IBM Business Process Manager
Version 8 Release 5

Übersicht über IBM Business Process Manager

PDF-Handbücher und Information Center

PDF-Handbücher erleichtern das Drucken sowie das Lesen im Offlinemodus. Die neuesten Informationen können Sie online im Information Center abrufen.

In der Gesamtheit bieten die PDF-Handbücher denselben Inhalt wie das Information Center. Einige Links in den PDF-Handbüchern wurden auf die Verwendung in den Information Centern zugeschnitten und funktionieren möglicherweise nicht ordnungsgemäß.

Die PDF-Dokumentation steht innerhalb von drei Monaten nach einem Hauptrelease des Information Centers (z. B. Version 7.0 oder Version 7.5) zur Verfügung.

Sie wird seltener als das Information Center, jedch häufiger als die Redbooks aktualisiert. PDF-Handbücher werden im Allgemeinen dann aktualisiert, wenn genügend Änderungen für ein Handbuch aufgelaufen sind.

Inhaltsverzeichnis

Kapitel 2. Weitere Informationen zu Schlüsselkonzepten 187

Kapitel 1. Erste Schritte mit IBM Business Process Manager

In diesem Abschnitt wird erläutert, welche Funktionen IBM® Business Process Manager für das Geschäftsprozessmanagement bereitstellt und wie die verschiedenen Phasen des Geschäftsprozessmanagements, z. B. das Erstellen und Implementieren von Prozessanwendungen, miteinander in Verbindung stehen.

Die Prozessanwendung ist der Basiscontainer für Prozesse und deren Komponenten in IBM Business Process Manager. Prozessentwickler erstellen Prozessanwendungen in den Authoring-Umgebungen und können dabei Services, Tasks und Artefakte einbeziehen, die zur Unterstützung der Ausführung benötigt werden.

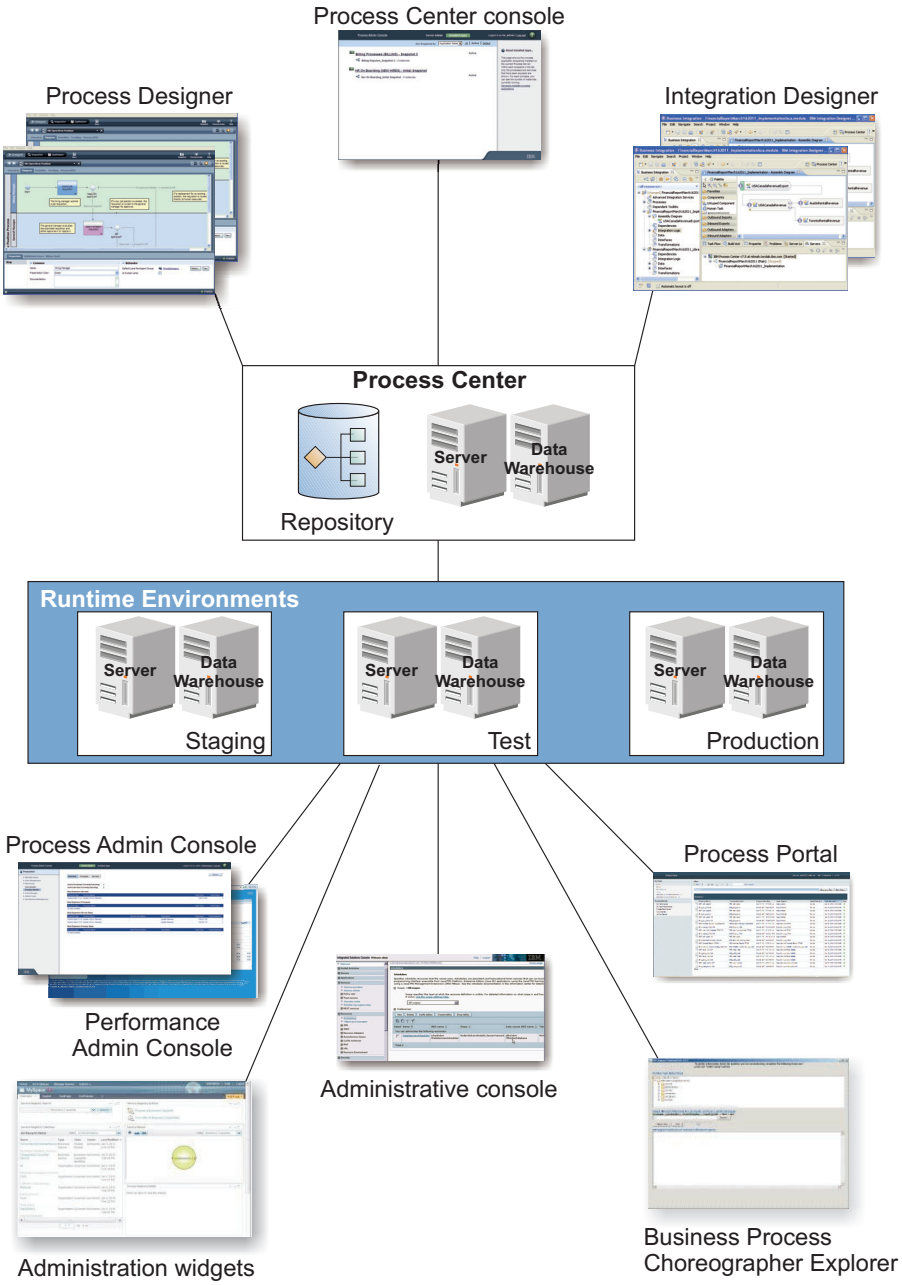
Erweiterte Integrationsservices werden in IBM Integration Designer implementiert und Prozessanwendungen zugeordnet. Mit Process Center werden Prozessanwendungen unter Process Server, der Prozesslaufzeitumgebung für IBM Business Process Manager, implementiert.

In ähnlicher Weise können automatisierte Prozesse, die in Integration Designer erstellt werden, Benutzerservices verwenden, die in IBM Process Designer entwickelt werden.

Produktübersicht

IBM Business Process Manager ist eine umfassende Plattform für das Geschäftsprozessmanagement, die vollständige Sichtbarkeit sowie Einblick in die Verwaltung von Geschäftsprozessen bietet. Es werden Tools, eine Laufzeitumgebung für Prozessdesign, -ausführung, -überwachung und -optimierung sowie eine Basisunterstützung für die Systemintegration bereitgestellt. Das Produkt kann so konfiguriert werden, dass verschiedene Stufen der Komplexität und der Einbeziehung in das Geschäftsprozessmanagement unterstützt werden.

Die Komponenten von IBM Business Process Manager umfassen ein einheitliches BPM-Repository, Tools für Autoren; Administratoren und Benutzer sowie eine Laufzeitplattform. Das Produkt kann so konfiguriert werden, dass verschiedene Stufen der Komplexität und der Einbeziehung in das Geschäftsprozessmanagement unterstützt werden. Das folgende Diagramm enthält eine typische IBM Business Process Manager-Konfiguration:



- Über die Authoring-Umgebungen von IBM Process Designer und IBM Integration Designer stellen Entwickler eine Verbindung zu IBM Process Center her. Entwickler können von jedem dieser GUI-basierten Tools für die Anwendungsentwicklung Geschäftsanwendungen erstellen, testen, debuggen und implementieren. Wählen Sie je nach Typ der Anwendung, die Sie entwickeln, ein beliebiges Tool aus. Es gibt auch Fälle, in denen beide Tools erhebliche Vorteile bieten.
- In den Authoring-Umgebungen von Process Designer und Integration Designer können Prozess- und Servicedesigner implementierbare Prozessanwendungen und wiederverwendbare Toolkits erstellen. Prozessanwendungen enthalten Prozessmodelle und Serviceimplementierungen sowie alle zugehörigen Unterstützungsdateien. Die Prozessanwendungen werden im Process Center-Repository gespeichert, so dass sie gemeinsam genutzt werden können.
- Das Process Center enthält zwei Server: Process Center-Server und Performance Data Warehouse-Server. Mit diesen Servern können Entwickler, die in Process Designer arbeiten, ihre Prozessanwendungen

ausführen und Leistungsdaten zum Testen und Wiedergeben während der Entwicklung speichern. Performance Data Warehouse ruft in regelmäßigen Intervallen verfolgte Daten vom Process Server oder Process Center-Server ab.

- Process Center unterstützt außerdem zahlreiche Verwaltungsfunktionen. Über Process Center Console installieren Administratoren Prozessanwendungen, die zur Bereitstellung, zum Test oder für die Produktion bereit sind, auf den Process Servern. Die Administratoren können auch ausgeführte Instanzen von Prozessanwendungen in konfigurierten Umgebungen verwalten.
- Implementieren Sie Prozessanwendungen auf einem Process Server zur Bereitstellung, zum Test oder für die Produktion. Die Laufzeitumgebungen unterstützen Business Process Model and Notation (BPMN) 2.0-Prozesse. IBM Business Process Manager Advanced unterstützt auch Business Process Execution Language (BPEL)-Prozesse.
- Über die Process Admin Console und die Performance Admin Console können Administratoren alle Laufzeitserver verwalten und pflegen. Verwenden Sie die Process Admin Console zum Verwalten des Process Center-Servers und der Process Server in Ihren Laufzeitumgebungen. Verwenden Sie die Performance Admin Console, um Leistungsengpässe zu identifizieren oder Instrumentierungsdaten für weitere Analysen zu erfassen.
- Über die Administrationskonsole können Sie Objekte wie Ressourcen, Anwendungen und Server erstellen und verwalten. Verwenden Sie darüber hinaus die Administrationskonsole zum Anzeigen von Produktnachrichten.
- Verwenden Sie Business Space, um angepasste Geschäftsbereiche zu erstellen, die Widgets zum Überwachen und Verwalten verschiedener Aspekte Ihres Systems bereitstellen. Sie können zum Beispiel Geschäftsaktivitäten, Services oder den Systemzustand überwachen oder Mediationsrichtlinien und Geschäftskalender verwalten. Sie können auch einen Geschäftsbereich mit den Widgets für die Benutzertaskverwaltung erstellen und ihn an Geschäftsprozessen teilhaben lassen.
- Prozessteilnehmer können mithilfe von Process Portal eine Verbindung zum Process Center-Server oder einen Process Server in einer beliebig konfigurierten Laufzeitumgebung herstellen, unabhängig davon, ob ein Prozess entwickelt, getestet oder in einer Produktionsumgebung veröffentlicht wurde.
- Verwalten Sie BPEL-Prozessinstanzen (Business Process Execution Language) im Business Process Choreographer Explorer oder in Business Space.

Konfigurationen von IBM Business Process Manager

Verschiedene Konfigurationen von IBM Business Process Manager entsprechen den typischen Einstiegspunkten oder Stufen in dem Geschäftsprozessmanagementprogramm eines Unternehmens.

Tabelle 1. Konfigurationen von IBM Business Process Manager

Konfiguration	**Phase**
Advanced	**Transformation** Vollständige Gruppe der Geschäftsprozessmanagementfunktionen • Erweiterte Unterstützung für Prozessautomation mit hohem Volumen • Integrierte SOA-Komponenten für umfassende unternehmensweite Serviceintegration und -koordination
Standard	**Programm** Konfiguriert für typische Geschäftsprozessmanagementprojekte • Für Verbesserungsprogramme, die mehrere Projekte umfassen; mit hoher Beteiligung des Unternehmens • Grundlegende Systemintegrationsunterstützung • Rapide Realisierungszeit und verbesserte Benutzerproduktivität

Tabelle 1. Konfigurationen von IBM Business Process Manager (Forts.)

Konfiguration	Phase
Express	**Projekt** Konfiguriert für ein erstes Geschäftsprozessmanagementprojekt • Rapide Realisierungszeit: Verbesserte Benutzerproduktivität • Niedrige Einstiegskosten • Einfache Installation und Konfiguration

Leistungsmerkmale in Konfigurationen von IBM Business Process Manager

In diesem Abschnitt wird erläutert, welche Produkte und Leistungsmerkmale oder Funktionen IBM für das Geschäftsprozessmanagement anbietet und wie Sie die richtige Wahl für Ihr Unternehmen treffen.

Bei IBM Business Process Manager werden benutzerorientierte und integrationsorientierte Funktionen in einem einheitlichen Produkt und einer einzigen BPM-Plattform kombiniert. Es stehen verschiedene Konfigurationen des Produkts zur Verfügung, die sich an unterschiedliche Benutzer richten und unterschiedliche Anforderungen im Unternehmen erfüllen. Die Produktkonfigurationen können für ein gemeinsames Authoring und für im Netz implementierte Laufzeitumgebungen kombiniert werden.

Tabelle 2. Leistungsmerkmale in Konfigurationen von IBM Business Process Manager

Funktion	Adv.	Std.	Express
WebSphere Lombardi Edition-kompatible Ausführung	X	X	X
Process Designer (BPMN)	X	X	X
Gemeinsame Bearbeitung / Sofortige Wiedergabe	X	X	X
Interaktive Benutzerschnittstellen ("Prozess-Coach")	X	X	X
ODM-basierte Verarbeitungsregeln	X	X	X
Process Portal	X	X	X
Echtzeitüberwachung und -berichterstellung	X	X	X
Leistungsanalyse und -optimierung	X	X	X
Performance Data Warehouse	X	X	X
Process Center / Repository für gemeinsam genutzte Assets	X	X	X
Uneingeschränkte Anzahl an Prozessautoren und Endbenutzern	X	X	200 Benutzer / 3 Autoren
Hochverfügbarkeit: Clustering und unbegrenzte Anzahl an Kernen	X	X	• 4 Kerne Produktion • 2 Kerne Entwicklung • Kein Cluster
WebSphere Process Server-kompatible Ausführung	X		
Integration Designer (BPEL / SOA)	X		
Integrierter Enterprise Service Bus (ESB)	X		
Transaktionsunterstützung	X		
Integrationsadapter	X		
Flexible Business Space-Benutzerschnittstelle	X		
Erweiterte Plattformunterstützung (Linux on System z, IBM AIX, Solaris)	X	X	*Hinweis

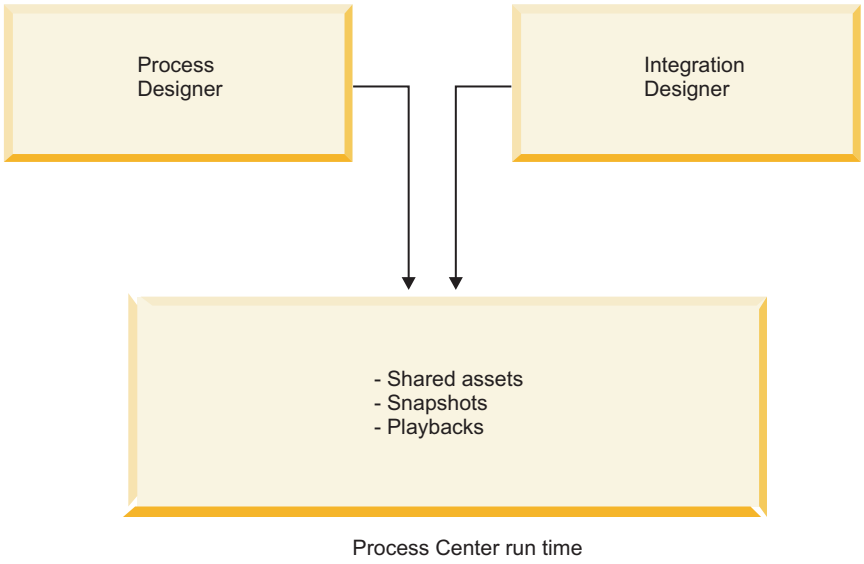
Anmerkung: IBM BPM Express wird auf AIX nur für IBM Master Data Management (MDM)-Kunden unterstützt.

Process Center-Repository

Process Center enthält ein Repository für alle Prozesse, Services und weiteren Assets, die in den IBM Business Process Manager-Authoring-Umgebungen Process Designer und Integration Designer erstellt werden.

Process Center ist eine Softwarekomponente, die als Server ausgeführt wird und bei der Process Designer und Integration Designer gemeinsam nutzen. Dies ermöglicht eine kooperative und hochinteraktive Entwicklung von Geschäftsprozessen.

Im folgenden Diagramm sehen Sie mehrere Komponenten, die zusammen die Erstellung komplexer Geschäftsprozesse ermöglichen.



Die Process Center Console stellt die Tools bereit, die Sie zur Verwaltung des Repositorys benötigen.

- Über die Process Center Console können Sie Prozessanwendungen und Toolkits erstellen sowie anderen Benutzern den Zugriff auf diese Prozessanwendungen und Toolkits gewähren.
- In den Authoring-Umgebungen erstellen Sie Prozessmodelle, Services und andere Ressourcen in Prozessanwendungen.
- Das Process Center enthält zwei Server: den Process Center-Server und den Performance Data Warehouse-Server. Mit diesem Process Center-Server können Entwickler, die in Process Designer arbeiten, ihre Prozessanwendungen ausführen und Leistungsdaten zum Testen und Wiedergeben während der Entwicklung speichern. Performance Data Warehouse ruft in regelmäßigen Intervallen verfolgte Daten vom Process Server oder Process Center-Server ab.
- Über die Process Center Console installieren Administratoren Prozessanwendungen, die zum Test oder für die Produktion bereit sind, auf den Process Servern in diesen Umgebungen.
- Über die Process Center Console verwalten Administratoren aktive Instanzen von Prozessanwendungen in den konfigurierten Umgebungen.

Die Process Center Console bietet eine effiziente Arbeitsumgebung zum Erstellen und Verwalten allgemeiner Container wie Prozessanwendungen und Toolkits. Administratoren, die nicht aktiv in der Designeransicht arbeiten, können die Process Center-Konsole verwenden, um ein Framework bereitzustellen, in dem BPM-Analysten und -Entwickler Ihre Prozesse und die zugrundeliegenden Implementierungen er-

stellen können. Eine weitere wichtige Aufgabe von Administratoren ist die Verwaltung des Zugriffs auf das Process Center Repository durch die Einrichtung geeigneter Berechtigungsstrukturen für Benutzer und Gruppen.

Benutzer mit den geeigneten Berechtigungen können bestimmte administrative Aufgaben direkt in Process Designer und Integration Designer erledigen. Ein Entwickler, der über einen Schreibzugriff auf die Prozessanwendung verfügt und den Status aller Prozessressourcen in einem bestimmten Stadium erfassen möchte, kann zum Beispiel bei der Arbeit in der Designer-Sicht eine Momentaufnahme (Snapshot) erstellen.

Process Server und Laufzeitumgebungen

Process Server enthält eine zentrale BPM-Laufzeitumgebung, die eine Vielzahl von Funktionen für Geschäftsprozesse und für die Servicekoordination und -integration unterstützt.

In Ihren Authoring-Umgebungen ermöglicht der in Process Center integrierte Process Server die Ausführung von Prozessen, während Sie diese erstellen. Sobald Sie diese Prozesse fertiggestellt haben, können Sie die Process Server in Ihren Laufzeitumgebungen zur Installation und Ausführung dieser Prozesse einsetzen. Die Komponente Business Performance Data Warehouse erfasst und aggregiert Daten aus den Prozessen, die auf den Process Servern ausgeführt werden. Sie können diese Daten dann zur Verbesserung Ihrer Geschäftsprozesse verwenden.

Mit der Process Admin Console verwalten Sie die Process Server in Ihren Laufzeitumgebungen (z. B. Staging-, Test- und Produktionsumgebungen) sowie den Process Server, der ein Bestandteil von Process Center ist.

Authoring-Umgebungen

In IBM Business Process Manager Advanced stehen zwei Authoring-Umgebungen zur Verfügung. Mithilfe von IBM Process Designer können Sie effizient Geschäftsprozesse modellieren, die Benutzertasks einbeziehen. Mit IBM Integration Designer können Sie Services erstellen, die eigenständig arbeiten oder andere bestehende Services (wie z. B. Web-Services, Unternehmensressourcenanwendungen oder Anwendungen in CICS und IMS) aufrufen.

- „Process Designer"
- „Integration Designer" auf Seite 7

Process Designer

Process Designer ist in allen Editionen des Produkts verfügbar. IBM Business Process Manager Advanced beinhaltet zudem Integration Designer mit den zugehörigen Editoren und Adaptern.

Ein Prozess ist die wichtigste Logikeinheit in IBM Business Process Manager. Er ist der Behälter für alle Komponenten einer Prozessdefinition, einschließlich Services, Aktivitäten und Gateways, Zeitgeber, Nachrichten und Ausnahmeereignisse, Sequenzlinien, Regeln und Variablen. Wenn Sie einen Prozess modellieren, erstellen Sie eine wieder verwendbare Geschäftsprozessdefinition (business process definition, BPD). Sowohl Process Designer als auch Integration Designer können Prozessmodelle erstellen, die Benutzertasks enthalten.

Process Designer unterstützt die Entwicklung von Geschäftsprozessen. Mit einem komfortablen, grafisch orientierten Tool können Sie eine Sequenz von Aktionen erstellen, die einen Geschäftsprozess bilden, und Sie können diesen Prozess neu zeichnen, wenn sich Umstände ändern. Erfordern Aktivitäten einen Zugriff auf große Back-End-Systeme oder Services, die Daten (z. B. Informationen zu Kunden) für den Geschäftsprozess bereitstellen, können Sie diesen Zweck mithilfe von Integration Designer realisieren. Mithilfe einer einfachen Schnittstelle kann eine Aktivität in Process Designer einen in Integration Designer erstellten Service aufrufen. Dieser Service kann Daten und Adapter mithilfe von Mediationsabläufen umsetzen, weiterleiten und erweitern, um eine ganze Reihe von Back-End-Systemen auf standardisierte Art

und Weise zu erreichen. Zusammenfassend lässt sich sagen, dass der Schwerpunkt von Process Designer auf den Geschäftsprozessen liegt, während bei Integration Designer automatisierte Services, die die Geschäftsprozesse ergänzen, im Mittelpunkt stehen. Siehe die Dokumentation mit der Einführung in IBM Process Designer.

Alle Process Designer-Projekte sind in Prozessanwendungen enthalten. Sie speichern diese Prozessanwendungen und die zugehörigen Artefakte im Process Center-Repository. Prozessanwendungen können Assets, die sich in Toolkits befinden, gemeinsam nutzen.

IBM Business Process Manager enthält mehrere Benutzerschnittstellen, in denen Sie Geschäftsprozesse modellieren, implementieren, simulieren und überprüfen können. In der Process Center-Konsole können Sie Prozessanwendungen, Toolkits, Verfolgungen und Snapshots erstellen und verwalten. In Process Designer können Sie Prozessmodelle, Berichte und einfache Services erstellen. Zum Ausführen und Debuggen von Prozessen dient der Inspector. Zur Ausführung von Simulationen ist das Dienstprogramm Optimizer geeignet.

Prozessanwendungen, die in Process Designer entwickelt werden, können jederzeit auf dem Process Center-Server ausgeführt oder als Snapshot gespeichert und dann auf dem Process Server implementiert werden. Dies trifft in gleicher Weise auf Services zu, die in Integration Designer entwickelt und Prozessanwendungen zugeordnet werden.

In Process Designer ist ein Erweiterungspunkt für den Anmeldungsauthentifikator definiert, um einen Anpassungspunkt für die Anmeldelogik von der Clientseite zu öffnen und damit besonderen Authentifizierungsvoraussetzungen der Serverseite zu entsprechen. Sobald die Authentifizierung ausgelöst wird, wird von Process Designer der Authentifikator abgerufen und die Anmeldelogik aufgerufen. Der Erweiterungspunkt für den Authentifikator wird im Eclipse-Plug-in-Format bereitgestellt. Informationen zur Verwendung des Authentifikator-Plug-ins finden Sie in IBM Process Designer installieren.

Integration Designer

Integration Designer enthält Editoren und Hilfsmittel, mit denen Entwickler komplexe automatisierte Prozesse und Services (wie z. B. SCA-Module, Mediationen oder BPEL-Prozesse) erstellen können. Das Produkt steht wahlweise als Teil des IBM Business Process Manager Advanced-Pakets oder als eigenständiges Toolset für andere Verwendungen bereit.

IBM Integration Designer wurde als vollständige Integrationsentwicklungsumgebung für Personen konzipiert, die integrierte Anwendungen erstellen. Integrierte Anwendungen sind nicht banal. Sie können Anwendungen auf unternehmensweiten Informationssystemen (EIS-Systemen) aufrufen, Geschäftsprozesse abteilungs- oder unternehmensübergreifend mit einbeziehen und Anwendungen lokal oder remote aufrufen, die in einer ganzen Reihe verschiedener Sprachen geschrieben sind und auf unterschiedlichen Betriebssystemen ausgeführt werden. Die Komponenten werden mithilfe visueller Editoren erstellt und zu anderen integrierten (d. h. aus einer Gruppe von Komponenten erstellten) Anwendungen zusammengesetzt. Die visuellen Editoren bilden eine Abstraktionsschicht zwischen den Komponenten und ihren Implementierungen. Ein Entwickler, der mit den Tools arbeitet, kann eine integrierte Anwendung zusammenstellen, ohne über detaillierte Kenntnisse über die zugrunde liegende Implementierung der einzelnen Komponenten verfügen zu müssen.

Die Tools von Integration Designer basieren auf einer serviceorientierten Architektur. Komponenten sind Services, und eine integrierte Anwendung mit vielen Komponenten ist ebenfalls ein Service. Die erstellten Services entsprechen den verbreiteten Standards in der Branche. BPEL-Prozesse, die ebenfalls zu Komponenten werden, werden auf ähnliche Weise mit benutzerfreundlichen visuellen Tools erstellt, die der standardisierten Sprache Business Process Execution Language (BPEL) entsprechen.

Das Konzept von Integration Designer sieht vor, dass Komponenten in Modulen zusammengefasst werden. Importe und Exporte dienen zum Austausch von Daten zwischen Modulen. Artefakte, die in einer Bibliothek abgelegt werden, können von mehreren Modulen gemeinsam genutzt werden.

Module und Bibliotheken können zur Nutzung mit Process Center einer Prozessanwendung zugeordnet werden und können von Prozessen, die in Process Designer erstellt werden, als Services verwendet werden. In diesen Fällen können sie auch mit der Prozessanwendung implementiert werden.

Alternativ hierzu können Module und Bibliotheken direkt in der Testumgebung oder auf dem Process Server implementiert werden. Mithilfe von Mediationsmodulen können Sie Mediationsabläufe erstellen, die dann in WebSphere Enterprise Service Bus oder auf dem Process Server implementiert werden können.

IBM Integration Designer enthält eine Funktion zur Erstellung von Datentypen und XML-Zuordnungen, die auf WebSphere DataPower-Geräten implementiert werden können. Es können auch Dateien in und aus WebSphere DataPower übertragen werden.

Verwaltungstools

IBM Business Process Manager enthält eine Gruppe von Verwaltungstools, die Sie bei der Durchführung einer Reihe von Tasks zu unterstützen sollen: vom Installieren und Verwalten von Snapshots, bis hin zum Steuern von Prozessen und Arbeiten mit den Ressourcen in Ihrer IT-Umgebung.

Befehlszeilentools

IBM Business Process Manager stellt Befehlszeilentools, Schnittstellen zur Scripterstellung und Programmierschnittstellen bereit, um die Laufzeitumgebung zu verwalten.

- Befehlszeilentools sind einfache Programme, die über die Eingabeaufforderung eines Betriebssystems ausgeführt werden, um bestimmte Tasks durchzuführen. Mit diesen Tools können Sie Anwendungsserver starten und stoppen, den Serverstatus überprüfen, Knoten hinzufügen oder entfernen und weitere Tasks durchführen.
- Das Scripterstellungsprogramm der WebSphere-Verwaltung (wsadmin) ist eine nicht-grafische Befehlsinterpreter-Umgebung, die es Ihnen ermöglicht, Verwaltungsoptionen in einer Scripting-Sprache auszuführen und Programme in einer Scripting-Sprache zur Ausführung zu übergeben. Es unterstützt dieselben Tasks wie die Administrationskonsole sowie viele der Tasks der Process Center-Konsole. Das Tool 'wsadmin' ist für Produktionsumgebungen und unbeaufsichtigte Operationen gedacht.
- Verwaltungsschnittstellen für die Programmierung sind eine Gruppe von Java-Klassen und -Methoden nach der JMX-Spezifikation (Java Management Extension), die Unterstützung für die Verwaltung von SCA-Objekten (Service Component Architecture) und Geschäftsobjekte bieten. Jede Programmierschnittstelle umfasst eine Beschreibung ihres jeweiligen Verwendungszwecks, ein Beispiel zur Veranschaulichung der Verwendung der Schnittstelle oder Klasse sowie Verweise auf die Beschreibungen der einzelnen Methoden.

Process Center-Konsole

Die Process Center-Konsole bietet Benutzern eine effiziente Arbeitsumgebung zum Erstellen und Verwalten allgemeiner Bibliothekselemente wie beispielsweise Prozessanwendungen und Toolkits. Es ist hilfreich, wenn ein Framework bereitgestellt werden soll, in dem BPM-Analysten und -Entwickler Ihre Prozesse und die zugrundeliegenden Implementierungen erstellen können. Darüber hinaus stellt die Process Center-Konsole Tools für die Verwaltung des Repositorys bereit, mit denen auch die entsprechenden Berechtigungen für Benutzer und Gruppen eingerichtet werden können.

Greifen Sie über einen Web-Browser auf die Process Center-Konsole zu (z. B. auf **http://host:9080/ProcessCenter**).

Process Admin Console

Die Process Admin Console wird verwendet, um die Process Server in Ihrer Umgebung zu verwalten, einschließlich der Benutzer und installierten Snapshots für die einzelnen Server. Darüber hinaus stellt diese Konsole Tools bereit, die Sie bei der Verwaltung von Warteschlangen und Cachespeichern unterstützen sollen.

Die Process Admin Console umfasst den Process Inspector, bei dem es sich um ein Tool handelt, mit dem Prozessinstanzen für Prozessanwendungen angezeigt und verwaltet werden können, die auf einem bestimmten Process Server ausgeführt werden.

Greifen Sie über einen Web-Browser auf die Process Admin Console zu (z. B. auf **http://host:9080/ProcessAdmin**).

Business Performance Admin Console
Die Business Performance Admin Console umfasst Tools zum Verwalten der Performance Data Warehouses in Ihrer Umgebung. Mithilfe dieses Tools können Sie Serverwarteschlangen verwalten und die Serverleistung überwachen.

Greifen Sie über einen Web-Browser auf die Business Performance Admin Console zu (z. B. auf **http://host:9080/PerformanceAdmin**).

WebSphere Application Server-Administrationskonsole
Die Administrationskonsole wird zur Verwaltung von Anwendungen, Services und anderen Ressourcen im Geltungsbereich einer Zelle, eines Knotens, eines Servers oder eines Clusters verwendet. Sie können die Konsole für eigenständige Server und für Deployment Manager verwenden, die alle Server in einer Zelle in einer Netzumgebung verwalten.

Wenn Sie ein eigenständiges Profil installiert haben, haben Sie einen einzelnen Knoten in einer eigenen Verwaltungsdomäne, die als Zelle bezeichnet wird. Verwenden Sie die Administrationskonsole, um Anwendungen, Busse, Server und Ressourcen innerhalb dieser Verwaltungsdomäne zu verwalten. Wenn Sie eine Network Deployment-Zelle installiert und konfiguriert haben, haben Sie dementsprechend einen Deployment Manager-Knoten sowie mindestens einen verwalteten Knoten in der Zelle. Verwenden Sie die Administrationskonsole, um Anwendungen zu verwalten, verwaltete Knoten in der Zelle einzurichten sowie diese Knoten und ihre Ressourcen zu überwachen und zu steuern.

Greifen Sie über einen Web-Browser auf diese Konsole zu (z. B. auf **http://host:9060/ibm/console** oder **https://host:9043/ibm/console**).

Business Process Choreographer Explorer und Business Process Archive Explorer
In Abhängigkeit von Ihrer Benutzerrolle können Sie diese Clientschnittstellen verwenden, um BPEL-Prozesse und Benutzertasks zu verwalten, die in IBM Integration Designer erstellt wurden, um die Ihnen zugewiesenen Tasks zu bearbeiten, um abgeschlossene BPEL-Prozesse und Benutzertasks anzuzeigen, die sich in der Archivdatenbank befinden, und um Prozesse und Tasks aus dem Archiv zu löschen.

Widgets 'Verwaltung'
Die Widgets 'Verwaltung' ermöglichen die Überwachung bestimmter Komponenten Ihrer Gesamtlösung für Business Process Management, einschließlich Module und Services des erweiterten Integrationsservice. Verwenden Sie diese Widgets in einem Business Space, um Transparenz in Ihrer Serviceanwendung und in Ihren Modulen zu erhalten und um folgende Fragen zu beantworten:

- Welche Services werden von einem Modul genutzt oder bereitgestellt? Wie hoch sind die Antwortzeit und der Durchsatz über eine definierten Zeitraum für diese Services?
- Welchen Status weist ein Modul auf?
- Enthält das Modul fehlgeschlagene Ereignisse?
- Welche Mediationsrichtlinien sind dem Modul zugeordnet?
- Welche BPEL-Prozesse und Benutzertasks werden in einem Modul verwendet?
- Sind Geschäftskalender oder Geschäftsregeln im Modul vorhanden?

Verwenden Sie ein oder mehrere Widgets, um einen Snapshot des gesamten Systemzustands Ihrer Geschäftslösung zu erhalten, einschließlich Status Ihrer Topologie (Implementierungsumgebung, Cluster), Systemanwendungen (z. B. Failed Event Manager oder Business Process Choreographer), Datenquellen, Messaging-Steuerkomponenten und Messaging-Warteschlangen.

Business Process Rules Manager

Der Business Process Rules Manager ist ein webbasiertes Tool, das Unternehmensanalysten beim Suchen nach und Ändern von Werten für Geschäftsregeln unterstützt. Bei diesem Tool handelt es sich um eine Option von IBM Process Server, die Sie zum Zeitpunkt der Profilerstellung oder nach der Installation des Servers zur Installation auswählen können.

Behindertengerechte Bedienung in IBM Business Process Manager

Eingabehilfefunktionen erleichtern Benutzern mit Behinderungen, wie beispielsweise eingeschränkter Mobilität oder Sehbehinderungen, die erfolgreiche Verwendung von Softwareprodukten.

IBM bemüht sich um ungehinderte Zugriffsmöglichkeiten für alle Benutzer unabhängig von Alter oder Fähigkeiten.

Details zu den Eingabehilfefunktionen dieses Produkts finden Sie in Funktionen zur behindertengerechten Bedienung in IBM Business Process Manager.

Verfügbarkeit landessprachlicher Versionen in IBM Business Process Manager

IBM Business Process Manager ist in einer Vielzahl von Sprachen vefügbar. Die Listen beschreiben die Unterstützungsstufe für eine bestimmte Sprache.

IBM Business Process Manager bietet Unterstützung für die folgenden Sprachen. Die Dokumentation ist möglicherweise nicht in allen Sprachen vollständig übersetzt.

- Vereinfachtes Chinesisch
- Traditionelles Chinesisch
- Tschechisch
- Englisch (US)
- Französisch
- Deutsch
- Ungarisch
- Italienisch
- Japanisch
- Koreanisch
- Polnisch
- Portugiesisch (Brasilien)
- Russisch
- Spanisch

IBM Business Process Manager bietet teilweise Unterstützung für die folgenden Sprachen. Die Dokumentation ist möglicherweise nicht in allen Sprachen vollständig übersetzt.

- Arabisch (übersetzt für BPEL-Benutzertaskwidgets, Business Process Choreographer Explorer-Widgets, Business Space-Framework und Business Space Monitor-Widgets)
- Dänisch (übersetzt für Business Space Monitor-Widgets und Business Space-Framework)
- Niederländisch (übersetzt für Process Designer, Process Center, BPD Modeler, Service Modeler, JSEditor, Process Designer, Business Space-Framework und Business Space Monitor-Widgets)
- Finnisch (übersetzt für Business Space Monitor-Widgets, Business Space-Framework, BPD Modeler, Service Modeler, JSEditor und Process Designer)
- Griechisch (übersetzt für Process Designer, Process Center und Business Space)
- Hebräisch (übersetzt für BPEL-Benutzertasks, Business Process Choreographer Explorer, Business Space-Framework und Business Space Monitor-Widgets)

- Norwegisch (übersetzt für Business Space Monitor-Widgets und Business Space-Framework)
- Portugiesisch (Portugal) (übersetzt für Process Designer, Process Center, BPD Modeler, Service Modeler und JSEditor)
- Rumänisch (übersetzt für Laufzeitoperationen)
- Slowakisch (übersetzt für Business Space, BPD Modeler, Service Modeler, JSEditor und Process Designer)
- Schwedisch (übersetzt für Business Space Monitor-Widgets und Business Space-Framework)
- Türkisch (übersetzt für Business Space)

  Anmerkung: Für die Ländereinstellung "Türkisch" muss der Eintrag **case-insensitive-security-cache** in der Datei `60Database.xml` auf **false** eingestellt sein, damit der Buchstabe *i* in Benutzernamen und Kennwörtern zulässig ist. Die Datei `60Database.xml` befindet sich im Verzeichnis *install_root*`\profiles\`*profile_name*`\config\cells\`*cell_name*`\nodes\`*node_name*`\servers\`*server_name*`\processcenter\config\system\`.

IBM Business Process Manager bietet Unterstützung für die Eingabe bidirektionaler Zeichenfolgen in der Process Designer-Umgebung, in Coachs und in Process Portal. Ferner werden JavaScript-APIs zur Testbearbeitung für bidirektionale Sprachen bereitgestellt.

Coachs und Process Portal unterstützen die Verwendung von hebräischen und arabischen Kalendern.

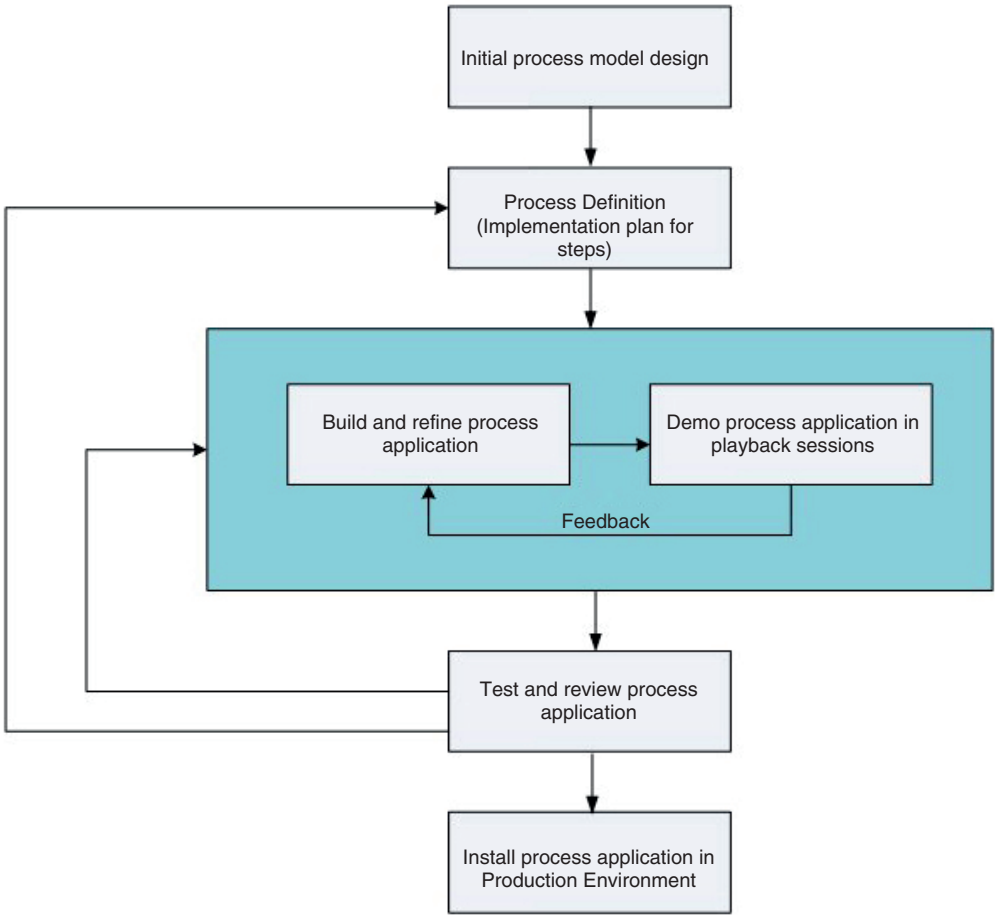
Übersicht über Geschäftsprozessmanagement

Bei der Entwicklung von Prozessen in Process Designer müssen Sie die abschließende Installation Ihrer Prozessanwendungen auf den Servern in Ihren Test- und Produktionsumgebungen planen.

Process Designer finden Sie in IBM Business Process Manager Express, IBM Business Process Manager Standard und IBM Business Process Manager Advanced. In diesem Abschnitt wird die erweiterte Version beschrieben, die für eine hohe Automatisierung und Verwendung komplexer Services, die in Integration Designer implementiert sind, konzipiert wurde. Die integrierten SOA-Komponenten können für eine umfassende, unternehmensübergreifende Serviceintegration verwendet werden. Die Standardversion kann von vielen Fachleuten kooperativ verwendet werden, um mehrere ausgereifte Prozesse zu entwickeln. Diese Version enthält eine grundlegende Systemintegration. Die Express-Version ist für eine kleine Anzahl von Benutzern auf einem Einzelserver geeignet, die ihre ersten Schritte mit Geschäftsprozessen machen oder keinen Zugriff auf viele externe Systeme benötigen.

Das folgende Diagramm zeigt den Lebenszyklus eines typischen Prozessentwicklungsverfahrens. Es zeigt auch Schritte zum Entwickeln und Optimieren eines Installationsservice, sodass Sie Ihre Prozessanwendungen in der Produktionsumgebung installieren können.

Wie das Diagramm zeigt, können Sie ausschließlich in Ihrer Entwicklungsumgebung arbeiten. Jedoch müssen Sie Process Server für Ihre Test- und Produktionsumgebungen konfigurieren.



Übersicht über die Prozessmodellierung

Bei einem Prozess handelt es sich um eine übergeordnete Logikeinheit in IBM Business Process Manager. Er ist der Behälter für alle Komponenten einer Prozessdefinition, einschließlich Services, Aktivitäten und Gateways, Zeitgeber, Nachrichten und Ausnahmeereignisse, Sequenzlinien, Regeln und Variablen. Wenn Sie einen Prozess modellieren, erstellen Sie eine wieder verwendbare Geschäftsprozessdefinition (Business Process Definition, BPD).

Prozesskomponenten ermöglichen Ihnen die Definition des Prozessarbeitsablaufs (Workflow) für Endanwender, indem Sie Logik im Prozess entwickeln und in andere Anwendungen und Datenquellen integrieren. Um zu verstehen, was zur Laufzeit innerhalb eines Prozesses abläuft, muss man die Komponenten kennen, die zum Entwicklungszeitpunkt in einen Prozess integriert werden.

Prozesse in IBM BPM erstellen

In die Entwicklung von Prozessen in IBM BPM werden viele Personen aus verschiedenen Organisationen einbezogen. Dabei muss unbedingt sichergestellt werden, dass Sie die bestmögliche Lösung entwickeln, um die beabsichtigten Ziele Ihres Projekts zu erreichen. Der Schlüssel zum Erfolg ist eine enge Kooperation der Mitglieder eines Teams bei der Erfassung der Prozessanforderungen sowie der iterativen Entwicklung des Modells und der zugehörigen Implementierungen.

Elemente in Process Designer wieder verwenden

Process Designer ermöglicht Prozessentwicklern die Wiederverwendung vorhandener Elemente in einer oder mehreren Prozessanwendungen. Wenn Sie zum Beispiel wissen, dass mehrere Services mit Coachs und anderen gemeinsam genutzten Elementen, die Sie und andere Entwickler benötigen, bereits vorhanden sind, können Sie auf diese Elemente zugreifen und sie wiederverwenden, indem Sie sie in ein Toolkit einfügen. Anschließend können Sie zu dem Toolkit, in dem sich die gemeinsam genutzten Elemente befinden, eine Abhängigkeit von Ihrer Prozessanwendung hinzufügen. Dieser Schritt ermöglicht Ihnen, bei der Auswahl einer Implementierung für eine Aktivität einen der vorhandenen Services auszuwählen. Die Elemente im Toolkit können auch von anderen Entwicklern verwendet werden, die an anderen Prozessanwendungen arbeiten.

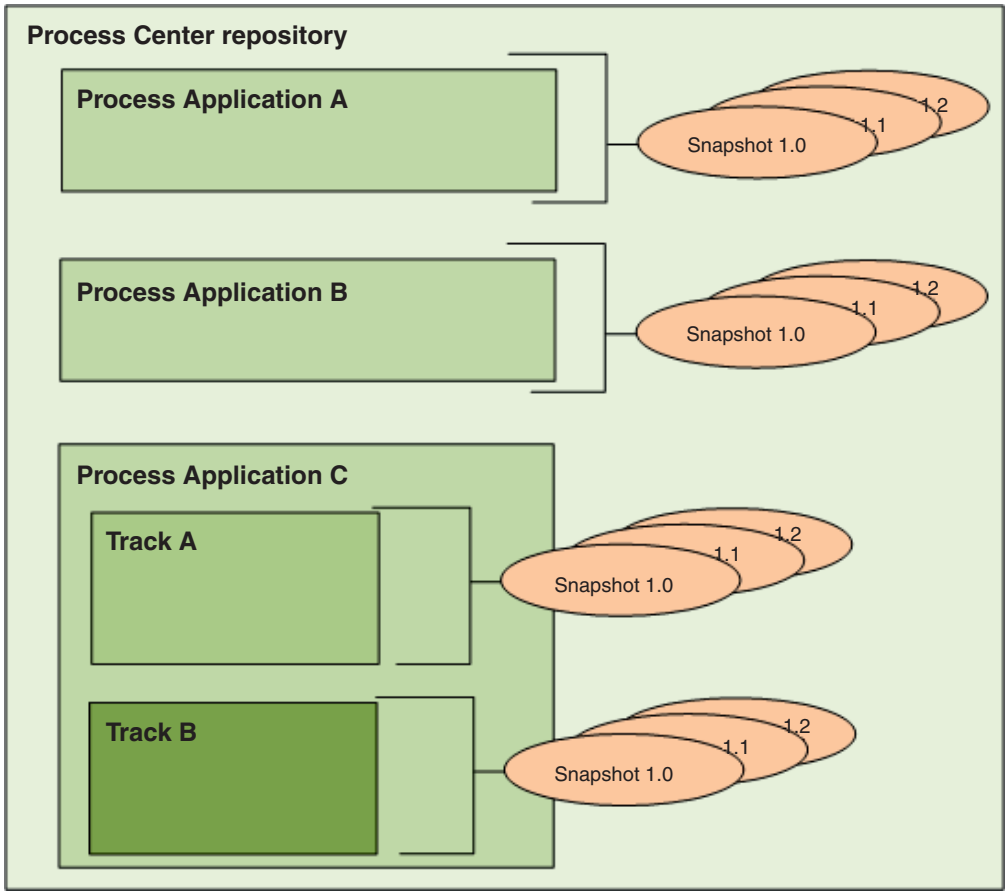
Designer in IBM Process Designer verwenden

Die Designer-Oberfläche stellt die Tools bereit, die Sie benötigen, um Ihre Prozesse in IBM BPM zu modellieren.

Prozessentwicklung mit Process Center

IBM Process Center dient als zentrales Repository für alle in Process Designer erstellten Projektressourcen. Werden mehrere Process Designer-Clients mit dem Process Center verbunden, können Benutzer Elemente, z. B. Prozesse und Services, gemeinsam nutzen und es werden ihnen die von anderen Benutzern vorgenommenen Änderungen sofort angezeigt. Das Process Center kann auch als Repository für Assets dienen, die in IBM Integration Designer erstellt werden.

Wenn Sie in Process Designer Prozesse entwickeln, steht im Process Center-Repository eine Hierarchie zur Verfügung, die beim Verwalten Ihrer Projekte helfen soll. Die folgende Abbildung bietet eine Übersicht der Konzepte der Repository-Hierarchie.



Wie dem vorherigen Diagramm zu entnehmen ist, umfasst das Process Center-Repository folgende Artefakte:

Inhaltstyp	Beschreibung
Prozessanwendungen	Container für Prozessmodelle und dazugehörige Implementierungen, die BPM-Analysten und Entwickler im Designer in IBM Process Designer entwickeln.
Verfolgungen	Optionale Teilbereiche in einer Prozessanwendung basierend auf Team-Tasks oder Versionen von Prozessanwendungen. Aktivierte Verfolgungen ermöglichen eine parallele Entwicklung. Administratoren legen fest, ob zusätzliche Verfolgungen erforderlich sind und somit für jede Prozessanwendung aktiviert wurden.
Snapshots	Zeichnen den Status der Elemente innerhalb einer Prozessanwendung oder einer Verfolgung zu einem bestimmten Zeitpunkt auf. Üblicherweise markieren Snapshots einen Meilenstein und werden zur Wiedergabe oder für Installationen verwendet. Sie können Snapshots vergleichen oder zu vorherigen Snapshots zurückkehren. Wenn ein Administrator Verfolgungen für eine Prozessanwendung aktiviert, wird ein Snapshot als Basis für eine neue Verfolgung verwendet.

Prozessanwendungen - Übersicht

Bei Prozessanwendungen handelt es sich um Container für Prozessmodelle und die zugehörigen unterstützenden Implementierungen. Prozessanwendungen werden im Repository gespeichert. Die Artefakte werden nach dem Erstellen (z. B. in Autorensystemumgebungen) in einer Prozessanwendung zusammengefügt.

Prozessanwendungen enthalten alle oder einige der folgenden Artefakte:

- Prozessmodelle, auch Geschäftsprozessdefinitionen (BPD = Business Process Definition) genannt
- Verweise auf Toolkits
- Die zum Implementieren von Aktivitäten bzw. für die Integration in andere Systeme erforderlichen Services (einschließlich erweiterte Integrationsservices)
- Mindestens eine Verfolgung
- SCA-Module und Bibliotheken, die in IBM Integration Designer erstellt wurden
- Alle anderen, zum Ausführen des Prozesses notwendigen Elemente

Eine Einführung in die iterative Prozessanwendung und Toolkit-Entwicklung mit Tipps, Snapshots und Aufzeichnungen bietet das Video „Iterative Prozessanwendung und Toolkit-Entwicklung", das auf YouTube oder im IBM Education Assistant Information Center zur Verfügung steht. Eine Mitschrift des Videos ist erhältlich.

Arbeitsversion, Snapshots und Verfolgungen der Prozessanwendung

Alle Änderungen werden dynamisch im Process Center-Repository in der Arbeitsversion gespeichert, die die aktuelle Arbeitsversion der Prozessanwendung ist. In Playback-Sitzungen kann die aktuelle Arbeitsversion der Prozessanwendung getestet und verwaltet werden.

Die Prozessanwendung bleibt so lange auf dieser Arbeitsversionsebene, bis Sie einen Snapshot erstellen, der den Status der Bibliothekselemente innerhalb einer Prozessanwendung oder Verfolgung zu einem bestimmten Zeitpunkt aufnimmt. Sie können in der Regel einen Snapshot jedes Mal aufnehmen, wenn Sie zum Testen der Integration bereit sind oder die Prozessanwendung auf einem Process Center-Server oder einen Process Server zum Testen, Bereitstellen oder Produzieren implementieren möchten.

Anmerkung: Bei der aktuellen Arbeitsversion handelt es sich um einen besonderen Snapshot; nämlich um den einzigen, dessen Inhalt bearbeitet werden kann. Allerdings kann er nur auf dem Process Center-Server ausgeführt werden. Sie können auf einem Process Server keine Arbeitsversion implementieren.

Jede Prozessanwendung hat standardmäßig eine einzelne Verfolgung mit dem Namen `Main`. Wenn Sie eine parallele Entwicklung auf einer Prozessanwendung zulassen möchten, können Sie weitere Verfolgungen erstellen. Diese optionalen Teilbereiche in der Prozessanwendung halten die Änderungen isoliert. Beispiel: Ihr Unternehmen möchte eine Markenumstrukturierung durchführen. Während dieses Übergangs müssen die aktuellen Prozessanwendungen beibehalten werden, während neue Versionen auf der Grundlage der aktualisierten Corporate Identity entwickelt werden. In dieser Situation behebt ein Team beispielsweise kleinere Fehler auf der aktuellen Version der Prozessanwendung (in der Verfolgung 'Main'), während das andere Team eine neue Version der Prozessanwendung in einer anderen Verfolgung entwickelt.

Toolkits für Prozessanwendungen

Toolkits sind Container, die Bibliothekselemente (z. B. BPDs) für die Wiederverwendung durch Prozessanwendungen oder andere Toolkits speichern. Prozessanwendungen können Bibliothekselemente zwischen einem oder mehreren Toolkits gemeinsam nutzen, und Toolkits können Bibliothekselemente von anderen Toolkits gemeinsam nutzen. Wenn Sie Zugriff auf ein Toolkit haben, können Sie eine Abhängigkeit von diesem Toolkit erstellen und die Bibliothekselemente dieses Toolkits in Ihrer Prozessanwendung verwenden.

Prozessanwendungen und Geschäftsanwendungen

Zu jeder Prozessanwendung gehört eine Geschäftsanwendung (BLA = Business Level Application), die als Container für die Prozessanwendung und die zugehörigen Ressourcen (z. B. SCA-Mmodule, Toolkits und Bibliotheken) agiert. Jeder Snapshot einer Prozessanwendung besitzt seine eigene BLA. Viele Verwaltungstasks für Snapshots (z. B. Stoppen und Starten von Snapshots auf einem Produktionsserver) werden auf der Geschäftsanwendungsebene vorgenommen, um eine zügigere und einfachere Verwaltung der Snapshots und der zugehörigen Assets zu ermöglichen.

Iterative Prozessanwendung und Toolkit-Entwicklung

Tabelle 3. Einführung

Szene	Ton	Zu sehende Handlung
1	Willkommen zu dieser Einführung in die iterative Prozessanwendung und Toolkit-Entwicklung in IBM Business Process Manager.	Die Eingangsanzeige zeigt den Titel des Videos, *Iterative Prozessanwendung und Toolkit-Entwicklung*, und den Untertitel, *Verwendung von Arbeitsversionen, Snapshots und Verfolgungen*. Copyright 2013, IBM Corporation.
2	In diesem Video erfahren Sie etwas über Prozessanwendungen und Toolkits und wie Sie Arbeitsweisen, Snapshots und Verfolgungen dafür verwenden können, deren Entwicklung und Implementierung zu verwalten.	Es wird eine Liste der Themen gezeigt, die in diesem Video behandelt werden.
3	Mit den veränderten Bedürfnissen Ihrer Organisation entwickeln sich auch die technischen Abläufe. Während Ihre Prozessanwendungen und Toolkits den Kreislauf von Entwicklung, Erprobung und Produktion durchlaufen, können Sie Prozess-Assets und Versionen mit dem IBM Business Process Manager verwalten.	Es wird ein Diagramm des iterativen Entwicklungszyklus gezeigt. Pfeile bewegen sich durch das Diagramm, um die Kreisläufe des Gestaltungsprozesses zu verdeutlichen.

Tabelle 3. Einführung (Forts.)

Szene	Ton	Zu sehende Handlung
4	Im IBM Business Process Manager dienen Prozessanwendungen als Container für Prozessmodelle und für die Unterstützung von Implementierungen, die die Entwickler mit IBM Process Designer erstellen.	Die Registerkarte 'Prozessanwendungen' in der Process Center-Konsole wird gezeigt. Der Cursor bewegt sich über dem Namen einer Prozessanwendung, dann über einem Link dorthin, wo der Benutzer sie mit IBM Process Designer öffnen kann. Der Text „Prozessanwendung: installierbarer Container für Prozessmodell" wird gezeigt.
5	Toolkits sind Container, die dieselben Artefakte enthalten wie Prozessanwendungen. Im Gegensatz zu Prozessanwendungen können sie jedoch nicht installiert und auf Process Server ausgeführt werden. Toolkits enthalten Artefakte, die von einer oder mehreren Prozessanwendungen wiederverwendet werden können. Anschließend werden die Toolkits indirekt mit jeder Prozessanwendung, die auf sie verweist, installiert.	Die Registerkarte 'Toolkits' in der Process Center-Konsole wird gezeigt. Der Cursor bewegt sich über dem Namen eines Toolkits, dann über einem Link dorthin, wo der Benutzer es mit IBM Process Designer öffnen kann. Der Text „Toolkit: Container für wiederverwendbare Artefakte" wird gezeigt.

Tabelle 4. Arbeitsweisen, Snapshots und Verfolgungen

Szene	Ton	Zu sehende Handlung
6	IBM Business Process Manager unterstützt ein iteratives Modell für die Prozessanwendungs- und Toolkitentwicklung mit Funktionen wie: • der Arbeitsweise, d.h. der aktuellen Arbeitsversion • Snapshots, die den Status von Bibliothekselementen zu einem bestimmten Zeitpunkt aufzeichnen • Verfolgungen, d.h. optionalen Verzweigungen innerhalb einer Prozessanwendung oder eines Toolkits, die Sie für die gleichzeitige Entwicklung mehrer Versionen einsetzen können	Eine Liste mit in IBM BPM enthaltenen Funktionen wird gezeigt, die ein iteratives Modell für die Prozessanwendung und Toolkitentwicklung unterstützen.
7	Während der Entwicklung einer Prozessanwendung oder eines Toolkits mit IBM Process Designer werden alle von Ihnen vorgenommen Änderungen am Process Center-Repository gespeichert. Diese aktuelle Arbeitsversion Ihrer Prozessanwendung oder Ihres Toolkits wird als Arbeitsweise bezeichnet.	Ein Prozessdiagramm in IBM Process Designer wird gezeigt. Es wird eine Diagrammkomponente geöffnet, und das Wort *Daten* ändert sich in einem Coach zu dem Wort *Datum*. Dann werden die Änderungen gespeichert. Der Text „Arbeitsweise: aktuelle Arbeitsversion" wird gezeigt.
8	Sie können für die Arbeitsweise Playback-Sitzungen verwenden, um die aktuelle Arbeitsversion der Prozessanwendung oder des Toolkits umgehend zu testen. Die Arbeitsweise kann nur für den Process Center-Server ausgeführt werden. Sie lässt sich nicht auf Process Server installieren.	Die Anfangsseite des Prozessdiagramms wird geöffnet und eine Playback-Sitzung gestartet. Damit öffnet sich die Inspector-Schnittstelle, und der geänderte Coach wird in einem Browserfenster angezeigt. Der Text „Arbeitsweise: Wiedergabe für umgehende Erprobung" wird gezeigt.

Tabelle 4. Arbeitsweisen, Snapshots und Verfolgungen (Forts.)

Szene	Ton	Zu sehende Handlung
9	Nachdem Sie den Änderungsvorgang an Ihrer Prozessanwendung oder Ihrem Toolkit abgeschlossen haben, können Sie einen Snapshot davon erstellen. Der Snapshot zeichnet den Status aller Elemente innerhalb einer Prozessanwendung oder eines Toolkits zu einem bestimmten Zeitpunkt auf. Nachdem der Snapshot erstellt ist, stehen Ihre Prozesse und zugehörigen Elemente weiterhin für die Bearbeitung der Arbeitsweise zur Verfügung. In Process Designer werden verfügbare Snapshots im Revisionsprotokoll angezeigt. Diese Snapshots lassen sich zwar anzeigen, aber nicht bearbeiten.	Das Browserfenster wird geschlossen und ein Snapshot von der Prozessanwendung erstellt. Die Designer-Oberfläche wird gezeigt, und der Cursor bewegt sich über dem Namen des neuen Snapshots im Abschnitt 'Revisionsprotokoll'. Der Text „Snapshot: zeichnet den Status aller Elemente auf" wird gezeigt.
10	Sie können die Snapshots auch vergleichen, um z.B. Informationen darüber zu erhalten, wann sie erstellt oder ob ihnen Prozesse hinzugefügt wurden.	Die Vergleichsansicht von Snapshots wird gezeigt. Der Cursor bewegt sich über dem Erstellungsdatum eines Snapshots und dem Namen eines Prozesselements, das geändert wurde. Der Text „Vergleich der Änderungen in mehreren Snapshots" wird gezeigt.
11	Beim Erstellen eines neuen Snapshots eines Toolkits verwenden die Prozessanwendungen, die auf das Toolkit verweisen, weiterhin den vorigen Snapshot. Wenn Sie die Prozessanwendung mit IBM Process Designer öffnen, um sie zu bearbeiten, benachrichtigt Sie eine Warnung, dass eine neue Version des Toolkits zur Verfügung steht.	Die Process Center-Konsole wird geöffnet. Es wird die Seite 'Toolkits' angezeigt und ein Toolkit geöffnet. Ein neuer Snapshot des Toolkits wird erstellt. Die Seite 'Prozessanwendungen' wird geöffnet, und eine Prozessanwendung in Process Designer wird geöffnet. Der Cursor bewegt sich über der Warnung, die erscheint, und den zugehörigen Menüelementen. Der Text „Prozessanwendungen behalten Verweise auf vorherigen Toolkit-Snapshot bei" wird gezeigt.
12	Sobald Ihre Prozessanwendung oder Ihr Toolkit für die Implementierung bereitsteht, kann ein Administrator mithilfe der Process Center-Konsole einen oder mehrere Snapshots auf Process Server installieren. Installierte Prozessanwendungen können nur auf Snapshots des Toolkits, nicht auf die aktuelle Arbeitsversion eines Toolkits, verweisen.	Die Process Center-Konsole und anschließend eine Prozessanwendung werden geöffnet. Der Cursor bewegt sich über einem Link eines Snapshots, auf den der Administrator klicken kann, um den Snapshot zu installieren. Der Text „Installieren von Snapshots einer Prozessanwendung auf Process Server" wird gezeigt.
13	Administratoren können Snapshots auch von der Process Center-Konsole aus verwalten, z.B. ihre Erstellung, Archivierung oder das Exportieren.	Das Dropdown-Menü neben dem Namen des Snapshots wird geöffnet. Der Cursor bewegt sich über alle Optionen des Menüs. Der Text „Verwaltung der Snapshots mithilfe der Process Center-Konsole" wird gezeigt.
14	Standardmäßig hat jede Prozessanwendung und jedes Toolkit eine einzige Verfolgung, Hauptzweig genannt. Der Administrator kann die Erstellung zusätzlicher Verfolgungen aktivieren, um parallele Entwicklungen zu ermöglichen. Diese optionalen Teilbereiche halten Änderungen isoliert.	Die Seite 'Verwalten' der Prozessanwendung wird geöffnet. Die Option „Benutzern das Erstellen von Tracks in dieser Prozessanwendung erlauben" ist aktiviert. Der Text „Verfolgungen: Optionale Teilbereiche halten Änderungen isoliert" wird gezeigt.
15	Sie erstellen aus dem Snapshot eine neue Verfolgung. Damit wird der ausgewählte Snapshot in die neue Verfolgung kopiert. Es werden keine weiteren Snapshots aus der Ausgangsverfolgung in die neue Verfolgung kopiert.	Die Seite 'Snapshots' der Prozessanwendung wird geöffnet. Eine neue Verfolgung, 'Verfolgung 2', wird geöffnet. Der Cursor bewegt sich über dem Namen des Snapshots, der überkopiert wurde. Der Text „Erstellen einer Verfolgung aus einem Snapshot" wird gezeigt.

Tabelle 4. Arbeitsweisen, Snapshots und Verfolgungen (Forts.)

Szene	Ton	Zu sehende Handlung
16	Jede Verfolgung hat eine bestimmte Arbeitsweise bzw. Arbeitsversion. In einer Prozessanwendung oder einem Toolkit mit mehreren Verfolgungen können Sie Assets aus einem Snapshot der einen Verfolgung in die Arbeitsweise einer anderen Verfolgung kopieren. Bei diesem Vorgang überschreiben die überarbeiteten Assets diejenigen Assets, die bereits in der Zielverfolgung vorhanden sind. Neue Assets werden hinzugefügt.	Der neue Snapshot wird geöffnet, und eine Seite mit den Einstellungen und Elementen der Prozessanwendung wird angezeigt. Die Geschäftsprozessdefinition wird gezeigt. Die Assets werden in die Arbeitsweise der Hauptzweig-Verfolgung kopiert. Anschließend werden die Seiten gezeigt, auf denen die Listen der neuen, aktualisierten oder in Konflikt stehenden Assets erscheinen. Der Text „Kopieren von Assets in andere Verfolgungen" wird gezeigt.
17	Hier wird die Hierarchie von Snapshots und Verfolgungen von Prozessanwendungen im Process Center-Repository gezeigt. In diesem Diagramm haben die Prozessanwendungen A und B standardgemäß eine Verfolgung, mehrere Snapshots und eine Arbeitsweise. Prozessanwendung C verfügt über zwei Verfolgungen, von denen jede mehrere Snapshots und eine Arbeitsweise hat. Die Snapshots und die Arbeitsweise für jede Verfolgung sind voneinander unabhängig.	Ein Diagramm der Beziehung zwischen Arbeitsweisen, Snapshots und Verfolgungen wird gezeigt. Das Diagramm zeigt Darstellungen der folgenden Prozessanwendungen: • Prozessanwendung A mit einer Verfolgung, vier Snapshots und einer Arbeitsweise • Prozessanwendung B mit einer Verfolgung, drei Snapshots und einer Arbeitsweise • Prozessanwendung C mit zwei Verfolgungen. Eine Verfolgung hat drei Snapshots und eine Arbeitsweise, die andere Verfolgung hat zwei Snapshots und eine Arbeitsweise.

Tabelle 5. Szenario der Verwendung von Arbeitsweisen, Snapshots und Verfolgungen

Szene	Ton	Zu sehende Handlung
18	Nachdem Sie nun wissen, was Prozessanwendungen und Toolkits sind und wie Arbeitsweisen, Snapshots und Verfolgungen verwendet werden, sehen wir uns die Anlässe für ihre Verwendung an. Dies ist Robert, Geschäftsprogrammierer für die Firma ABC.	Ein Foto eines lächelnden Mannes wird gezeigt. Das Foto ist mit „Robert, Geschäftsprogrammierer, Firma ABC" unterschrieben.

Tabelle 5. Szenario der Verwendung von Arbeitsweisen, Snapshots und Verfolgungen (Forts.)

Szene	Ton	Zu sehende Handlung
19	Robert entwickelt einen Prozess, der modellieren soll, wie seine Firma neue Mitarbeiter einstellt. Sein Prozess hat die Standardverfolgung, die der Hauptverzweigung. Robert nimmt in der Arbeitsweise Änderungen am Prozess vor. Um sicherzustellen, dass der Prozess problemlos verläuft, startet er eine Playback-Sitzung, die die Arbeitsweise für den Process Center-Server ausführt.	Ein Diagramm wird gezeigt, das eine Entwicklungsumgebung für eine Prozessanwendung darstellt. Unter dem Abschnitt 'Entwicklung' wird eine Gruppe von drei verschachtelten Rechtecken gezeigt. Das äußere Rechteck stellt das Process Center-Repository dar, das mittlere die Prozessanwendung und das innere die Hauptzweig-Verfolgung. Rechts von dem Abschnitt 'Entwicklung' befindet sich der Abschnitt 'Testen'. Dieser Abschnitt enthält nur ein Rechteck, das den Process Center-Server darstellt. Unterhalb des Abschnitts 'Testen' ist der Abschnitt 'Produktion'. Dieser Abschnitt enthält nur ein Rechteck, das Process Server darstellt. Um zu zeigen, wie Robert Änderungen an der Arbeitsweise vornimmt, wird in der Hauptzweig-Verfolgung ein Dreieck gezeigt, das die Arbeitsweise darstellt. Wenn Robert eine Playback-Sitzung ausführt, zeigt ein Pfeil, wie die Arbeitsweise in den Process Center-Server kopiert wird.
20	Robert und sein Team haben die Entwicklung und Erprobung der Prozessanwendung beendet, die jetzt in die Produktionsumgebung implementiert werden kann. Zu diesem Zweck erstellt Robert einen Snapshot für die Prozessanwendung. Dann installiert Alice als Administrator den Snapshot in Process Server. Firma ABC kann Roberts Prozess jetzt als Leitfaden für die Einstellung neuer Mitarbeiter verwenden.	Um zu zeigen, wie Robert einen Snapshot erstellt, erscheint neben der Arbeitsweise in der Hauptzweig-Verfolgung ein Oval, das einen Snapshot darstellt. Um zu zeigen, wie Alice den Snapshot installiert, zeigt ein Pfeil, wie der Snapshot in Process Server kopiert wird.
21	Später kündigt Firma ABC an, dass der Firmenname sich ändert und in Zukunft Firma DEF lautet. Während dieses Übergangs muss der Einstellungsprozess, den Robert erstellt hat, gepflegt werden. Gleichzeitig wird anhand der aktualisierten Corporate Identity eine neue Version entwickelt. Das Wartungsteam nimmt kleinere Korrekturen am ursprünglichen Prozess in der Hauptzweig-Verfolgung vor, während Robert und sein Team in der Verfolgung 'DEF Neuer Firmenname' eine Version der Einstellungs-Anwendung mit neuem Firmennamen erstellen.	Ein zusätzlicher Snapshot erscheint, der zeigt, dass an der Prozessanwendung weitere Entwicklungen stattgefunden haben. Ein Rechteck zeigt, dass die Verfolgung 'DEF Neuer Firmenname' innerhalb der Prozessanwendung hinzugefügt wurde. Die Verfolgung 'DEF Neuer Firmenname' hat anfangs einen Snapshot und eine Arbeitsweise. Dann wird ein weiterer Snapshot hinzugefügt, um die weitere Entwicklung in der Verfolgung zu zeigen.

Tabelle 5. Szenario der Verwendung von Arbeitsweisen, Snapshots und Verfolgungen (Forts.)

Szene	Ton	Zu sehende Handlung
22	Sowohl das Wartungsteam als auch das Team für die Umsetzung des neuen Firmennamens können auf dem Process Center-Server separate Playback-Sitzungen ausführen. Wenn die Prozessanwendungen für die Implementierung bereitstehen, kann jede Verfolgung einen oder mehrere Snapshots für Process Server installieren.	Pfeile zeigen die Richtung der Arbeitsweisen aus beiden Verfolgungen, die zur selben Zeit für den Process Center-Server ausgeführt werden. Anschließend zeigen Pfeile Snapshots aus allen Verfolgungen, die gleichzeitig für Process Server ausgeführt werden.

Tabelle 6. Schluss

Szene	Ton	Zu sehende Handlung
23	In diesem Video haben wir Prozessanwendungen und Toolkits behandelt, und wie IBM Business Process Manager die iterative Prozessentwicklung mit Arbeitsweisen, Snapshots und Verfolgungen unterstützt.	Es wird eine Liste der Themen gezeigt, die in diesem Video behandelt wurden.
24	Weitere Informationen zu IBM Business Process Manager und der Verwendung von Arbeitsweisen, Snapshots und Verfolgungen finden Sie beim Durchsuchen unserer YouTube-Videos oder beim Besuch unserer folgenden offiziellen Ressourcen.	Eine Liste mit den folgenden zusätzlichen Ressourcen wird gezeigt: • YouTube-Kanal 'WebSphereEducation' • IBM Business Process Manager V8.5 - Dokumentation • IBM developerWorks • IBM Education Assistant

Ausführen und Debugging von Prozessen

Mit dem Inspector können einzelne Entwickler Prozesse und Services auf dem Process Center Server oder Process Server der fernen Laufzeit ausführen.

Der Inspector in IBM Process Designer ist der Schlüssel zu einem iterativen Modell bei der Prozessentwicklung. Ein komplettes Entwicklerteam kann den Inspector verwenden, um aktuelle Prozessdesigns und Implementierungen in Playback-Sitzungen zu demonstrieren. Playback-Sitzungen helfen bei der Erfassung wichtiger Informationen von verschiedenen Beteiligten an einem Prozess, beispielsweise Management, Endbenutzer und Geschäftsanalysten. Ein iteratives Modell bei der Prozessentwicklung sorgt dafür, dass Ihre Prozessanwendungen die Ziele und Anforderungen aller Beteiligten erfüllen.

Der Inspector in IBM Process Designer enthält mehrere Tools, mit denen Sie in den jeweiligen Umgebungen folgende Aufgaben ausführen können:

Task	Beschreibung
Verwalten von Prozessinstanzen	Wenn Sie einen Prozess ausführen, können Sie alle zuvor ausgeführten und derzeit aktiven Instanzen auf den IBM Business Process Manager-Servern in Ihrer Umgebung anzeigen. Sie können laufende Instanzen verwalten, indem Sie sie beispielsweise anhalten und dann wieder aufnehmen. Sie können auch zuvor ausgeführte Instanzen verwalten, indem Sie bestimmte Datensätze filtern oder löschen.

Task	Beschreibung
Schrittweises Durchgehen und Fehlersuche in einem Prozess	Für eine ausgewählte Instanz sehen Sie den gerade ausgeführten Schritt und bewegen sich weiter durch den Prozess, während Sie die Prozessausführung Schritt für Schritt auswerten. Eine Anzeige des Prozesses in Form der Baumstruktur, kombiniert mit Indikatoren, die als Token bezeichnet werden, im Prozessdiagramm, ermöglichen eine klare Sicht auf die aktuelle Position im Prozess. Dabei sehen Sie auch die Variablen, die in jedem Schritt verwendet werden, gegebenenfalls mitsamt den entsprechenden Werten.

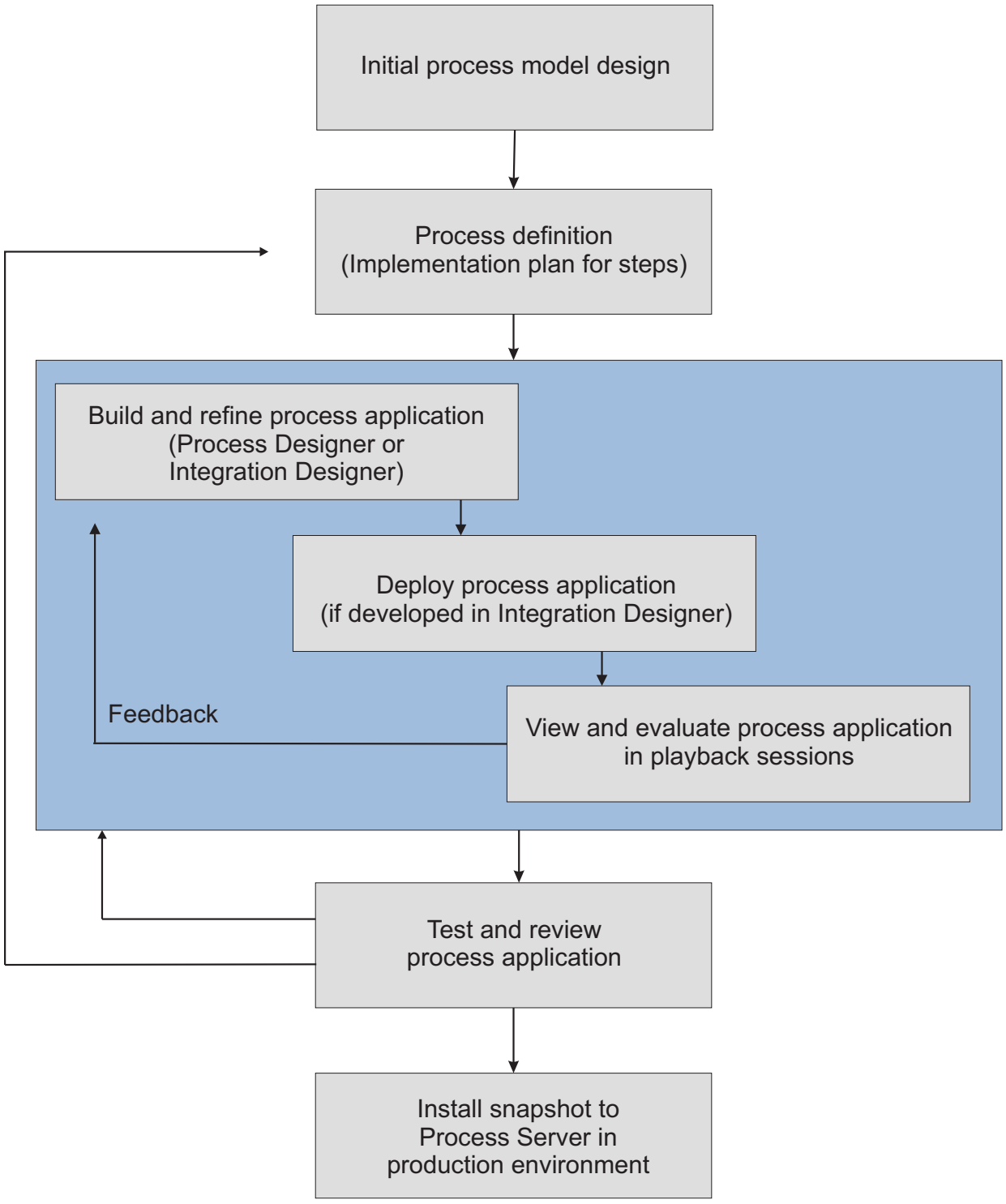
Wenn Sie in IBM Integration Designer arbeiten, können Sie den Inspector verwenden, wenn Ihr Projekt einer Prozessanwendung zugeordnet ist. Darüber hinaus stehen Ihnen noch weitere Debug- und Testtools zur Verfügung. Weitere Informationen zu diesen Integration Designer-Tools finden Sie in den Abschnitten zum Testen von Modulen und zur Verwendung des Integrationsdebuggers zur Problembestimmung, indem Sie den entsprechenden zugehörigen Links folgen.

Prozessanwendungen installieren und verwalten

Der Lebenszyklus einer Prozessanwendung umfasst die Installation, Verwaltung und Deimplementierung von Snapshots. Überlegungen hinsichtlich der Versionierung gehören ebenfalls zu diesem Lebenszyklus.

Bei der Prozessentwicklung können Sie die Vorteile des iterativen Ansatzes, der von den Process Designer-Tools unterstützt wird, optimal nutzen. Prozesse werden nach und nach weiterentwickelt - vom anfänglichen Entwicklungsstadium über die Testphase bis hin zur Produktion. Auch in der Produktion kann die Weiterentwicklung oder Anpassung Ihrer Prozesse an sich ändernde Anforderungen erforderlich sein. Daher empfiehlt es sich, auf den sich fortsetzenden Lebenszyklus Ihrer Prozesse vorbereitet zu sein, um von vornherein ein effizientes Design zu schaffen.

Die folgende Abbildung zeigt ein iteratives Modell bei der Prozessentwicklung.



Eine typische Business Process Manager-Konfiguration weist drei Umgebungen auf, die die Entwicklung und letztendliche Installation Ihrer Prozesse unterstützen.

Umgebung	Beschreibung
Entwicklung	Erstellen und optimieren Sie Ihre Prozessanwendungen in IBM Process Designer. Erstellen Sie Ihre Prozessmodelle und implementieren Sie die Schritte in diesen Modellen unter Einsatz von Designer. Mit dem Inspector können Sie in Playback-Sitzungen Ihren Entwicklungsfortschritt demonstrieren, damit Sie Ihren Prototyp schnell auswerten und optimieren können. Über die Process Center-Konsole können Sie Ihre Prozessanwendungen auf Process Servern installieren, die in Test- oder Produktionsumgebungen eingesetzt werden.
Test	Installieren Sie Ihre Prozessanwendungen mit der Process Center-Konsole auf dem Process Server in Ihrer Testumgebung, um formale Qualitätssicherungstests zu implementieren. Mit dem Inspector können Sie Probleme verifizieren und beheben.
Produktion	Wenn alle Probleme, die im Rahmen der formalen Tests gemeldet wurden, behoben sind, können Sie Ihre Prozessanwendungen mit der Process Center-Konsole auf dem Process Server in Ihrer Produktionsumgebung installieren. Mit dem Inspector können Sie alle Probleme überprüfen und beheben, über die in der Produktionsumgebung berichtet wurde.

Wenn Sie den Snapshot einer Prozessanwendung, die Inhalt von IBM Business Process Manager Advanced enthält, testen, installieren oder verwalten möchten, muss der Benutzer oder die Gruppe, zu der Sie gehören, der Verwaltungssicherheitsrolle Konfigurator, Operator *und* Implementierer zugewiesen sein. Wenn momentan keine Zuordnung zu allen drei Rollen besteht, klicken Sie in der WebSphere-Administrationskonsole auf **Benutzer und Gruppen**, um die Rollen für Benutzer oder Gruppen zu ändern. Weitere Informationen finden Sie unter Sicherheitsaufgabenbereiche in IBM Business Process Manager.

Release- und Installationsstrategien

Um sicherzustellen, dass die von Ihnen implementierten und installierten Prozessanwendungen die Qualitätsstandards Ihrer Organisation einhalten, ist es sinnvoll, eine Freigabe- und Installationsstrategie zu definieren. Wenn Sie die Ziele und Anforderungen für die Freigabe und Installation neuer und aktualisierter Prozessanwendungen ermittelt haben, können die erforderlichen Prozesse für die Genehmigung und Inbetriebnahme der Programme automatisieren.

Beispielsweise könnten Sie einen Prozess an Manager in verschiedenen Geschäftsbereichen weiterleiten. Erst wenn jeder Manager den neuen oder aktualisierten Prozess abgezeichnet hat, kann er in Ihrer Produktionsumgebung installiert und für Endbenutzer bereitgestellt werden. Sie können die notwendigen Schritte, die eine solche Prüfung umfasst, in IBM Business Process Manager Advanced erstellen und implementieren und somit sicherstellen, dass alle Unternehmensrichtlinien erfüllt werden und dass die entsprechenden Genehmigungen vorliegen. Der letzte Schritt in der Prüfungsroutine könnte die Benachrichtigung des IT-Teams sein, dass die genehmigte Prozessanwendung zur Installation bereitsteht.

Service erstellen, integrieren und auf sie zugreifen

Geschäftsprozesse verwenden häufig Services, die notwendige Funktionen für den Geschäftsprozess bereitstellen. Diese Services werden in einem Geschäftsprozessdiagramm als Aktivität oder Schritt angezeigt. Ein Service in Process Designer kann zum Beispiel einen externen Web-Service oder einen komplexen und automatisierten Service in Integration Designer aufrufen.

Auf externe Services für eine Anwendung zugreifen

In diesem Szenario werden verschiedene Möglichkeiten für den Zugriff auf Services erläutert, die für eine Anwendung extern sind. Außerdem sind die übergeordneten Tasks aufgeführt, die mit dem Zugriff auf diese externen Services verbunden sind.

Anmerkung: Dieses Szenario ist für IBM Business Process Manager Advanced anwendbar.

In einer integrierten Geschäftsanwendung wird eine erforderliche Funktion bereitgestellt, indem *Geschäftsservices* miteinander interagieren. Ein Geschäftsservice führt eine reproduzierbare Funktion oder Task aus, die zum Erreichen eines Geschäftsziels beiträgt. Das Suchen nach einem Service und das Herstellen einer

Verbindung zu diesem Service gehört jedoch nicht in den Aufgabenbereich der Geschäftsfunktion. Eine Lösung ist flexibler, wenn die Task für die Verwaltung von Serviceverbindungen nicht an die Geschäftsfunktion gekoppelt ist.

Die Serviceinteraktion beginnt, wenn ein *Serviceanforderer* eine Anforderung für die Ausführung einer Geschäftsfunktion an einen *Service-Provider* sendet. Diese Anforderung wird in Form einer *Nachricht* gesendet, in der die auszuführende Funktion definiert ist. Der Service-Provider führt die angeforderte Funktion aus und sendet das Ergebnis in einer Nachricht an den Serviceanforderer. Normalerweise müssen Nachrichten verarbeitet werden, damit Services Daten miteinander austauschen können und um andere untergeordnete IT-Funktionen implementieren zu können, die von den Geschäftsfunktionen und -daten unabhängig sind. Beispiele hierfür sind die Weiterleitung, die Protokollkonvertierung, die Transformation, die Wiederholung eines fehlgeschlagenen Aufrufs und der dynamische Serviceaufruf. Diese Verarbeitung wird als *Mediation* bezeichnet.

In IBM Integration Designer gibt es zwei Typen von Modulen. Dies sind zum einen Module (oder auch 'Geschäftsintegrationsmodule'), die primär für die Aufnahme von Geschäftslogik bestimmt sind (z. B. Geschäftsprozesse, Business-Regeln und Business-Statusmaschinen), und zum anderen Mediationsmodule, die Mediationsabläufe implementieren. Hinsichtlich der Funktionsweise gibt es zwar einige Überschneidungen zwischen den beiden Modultypen, aber es empfiehlt sich, die Geschäftslogik in Geschäftsmodulen zu isolieren und die Mediationslogik durch Mediationsmodule ausführen zu lassen.

Dennoch ist eine klare Trennung zwischen Geschäfts- und Mediationslogik nicht immer möglich. In solchen Fällen sollte der Umfang von *Statusangaben* oder Daten in Variablen berücksichtigt werden, der zwischen Serviceaufrufen verarbeitet werden muss. Wenn keine oder nur eine geringe Statusverarbeitung erforderlich ist, ist es in der Regel sinnvoll, eine Mediationsablaufkomponente zu verwenden. Falls es erforderlich ist, Statuswerte zwischen Serviceaufrufen zu speichern, oder falls Daten in Variablen gespeichert und verarbeitet werden müssen, ist die Verwendung einer Geschäftsprozesskomponente eine gute Entscheidung. Beispiel: Wenn Sie mehrere Services aufrufen und die von den einzelnen Services zurückgegebenen Informationen aufzeichnen, damit nach dem Aufruf aller Services die zurückgegebenen Daten weiter verarbeitet werden können, verwenden Sie einen Geschäftsprozess, in dem Sie die zurückgegebenen Daten ohne großen Aufwand zu Variablen zuordnen können. Anders ausgedrückt wird die Grenze zur Geschäftslogik dann überschritten, wenn zu viele Statusangaben vorliegen. In den folgenden Abschnitten wird diese Anweisung erläutert.

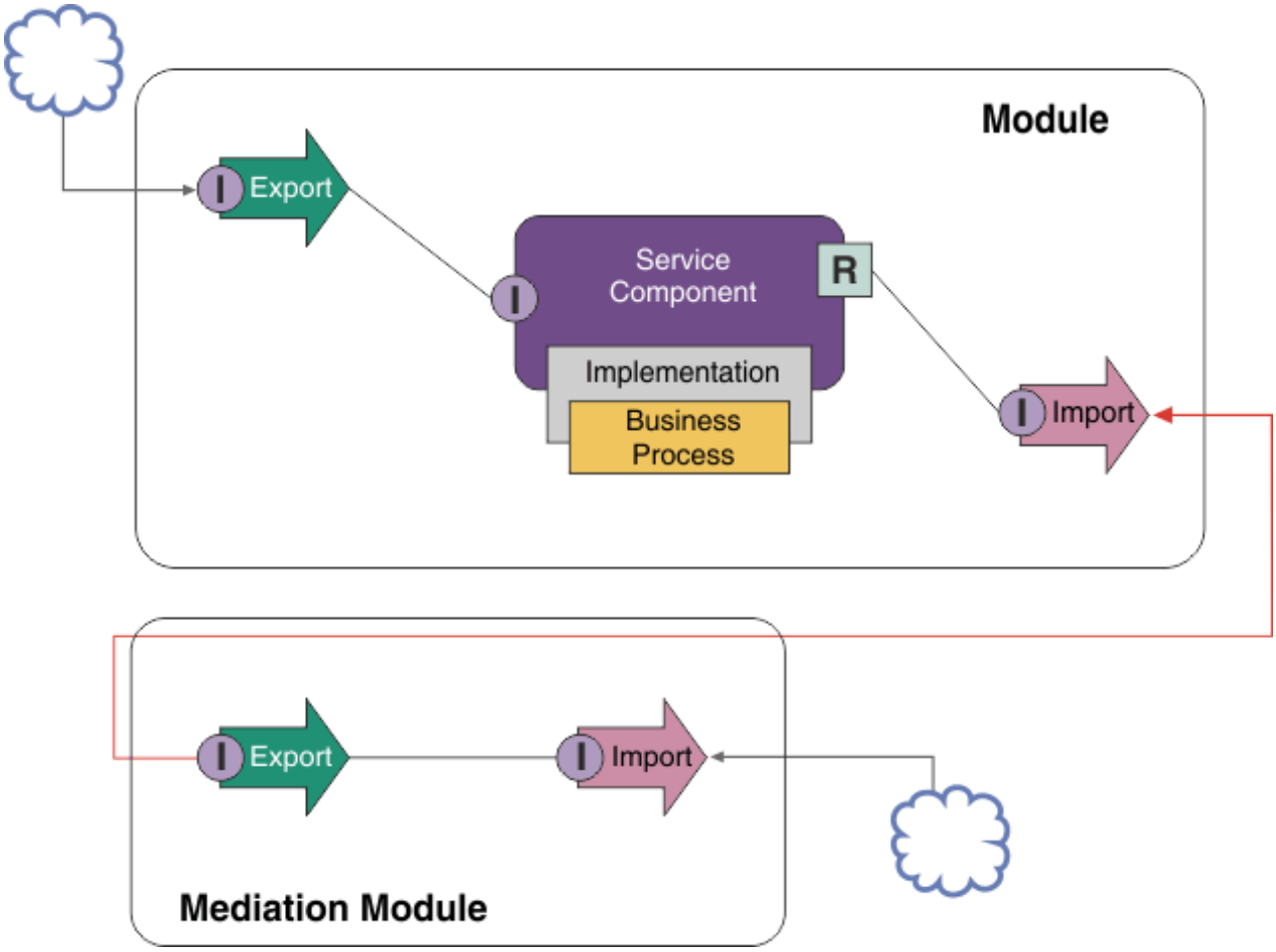
In diesem Zusammenhang gibt es kein allgemeingültiges Integrationsszenario und technisch gesehen keine falsche Antwort. Die hier erläuterten Richtlinien haben in der Praxis Flexibilität und Wiederverwendung ermöglicht und sollen als Anregung dienen. Wie bei jeder Entscheidung sollten Sie auch hier die Vor- und Nachteile einer Implementierung dieser Muster für Ihre Geschäftsintegrationsanwendung sorgfältig bedenken. Die folgenden Situationen können dabei hilfreich sein.

Auf eine SCA-Komponente zugreifen

Ein grundlegendes Beispiel für den Zugriff auf einen Service ist der Aufruf einer anderen SCA-Komponente durch einen Import, bei dem keine Datenkonvertierung erforderlich ist. Sogar in dieser Situation könnten Sie auf den externen Service über ein Mediationsmodul und nicht direkt aus einem Geschäftsmodul heraus zugreifen. Auf diese Weise bleiben Sie flexibel und können zu einem späteren Zeitpunkt den Serviceendpunkt, die Servicequalität oder die Governance ändern (z. B. durch das Hinzufügen einer Pro-

tokollierung), ohne hierdurch die Geschäftskomponenten zu beeinflussen, die den Service verwenden. Dieses Architekturmuster wird als 'Separation of Concerns' bezeichnet.

Bevor Sie sich für eine Implementierung dieses Musters entscheiden, sollten Sie die Vorzüge des Musters und die potenziellen Auswirkungen auf den Systemaufwand, die sich durch die Einführung eines weiteren Moduls ergeben, gegeneinander abwägen. Wenn Sie hauptsächlich Flexibilität anstreben und voraussichtlich häufig Änderungen an den Services vornehmen werden, auf die zugegriffen wird, kann es eine gute Entscheidung sein, ein separates Modul wie hier gezeigt zu verwenden. Wenn das Leistungsverhalten für Sie der wichtigste Aspekt ist und es für Sie akzeptabel ist, die Geschäftslogik zu aktualisieren und erneut zu implementieren, kann die Verwendung eines einzigen Moduls sinnvoll sein.



Das vorgestellte Beispiel wird durch die folgenden übergeordneten Tasks realisiert.

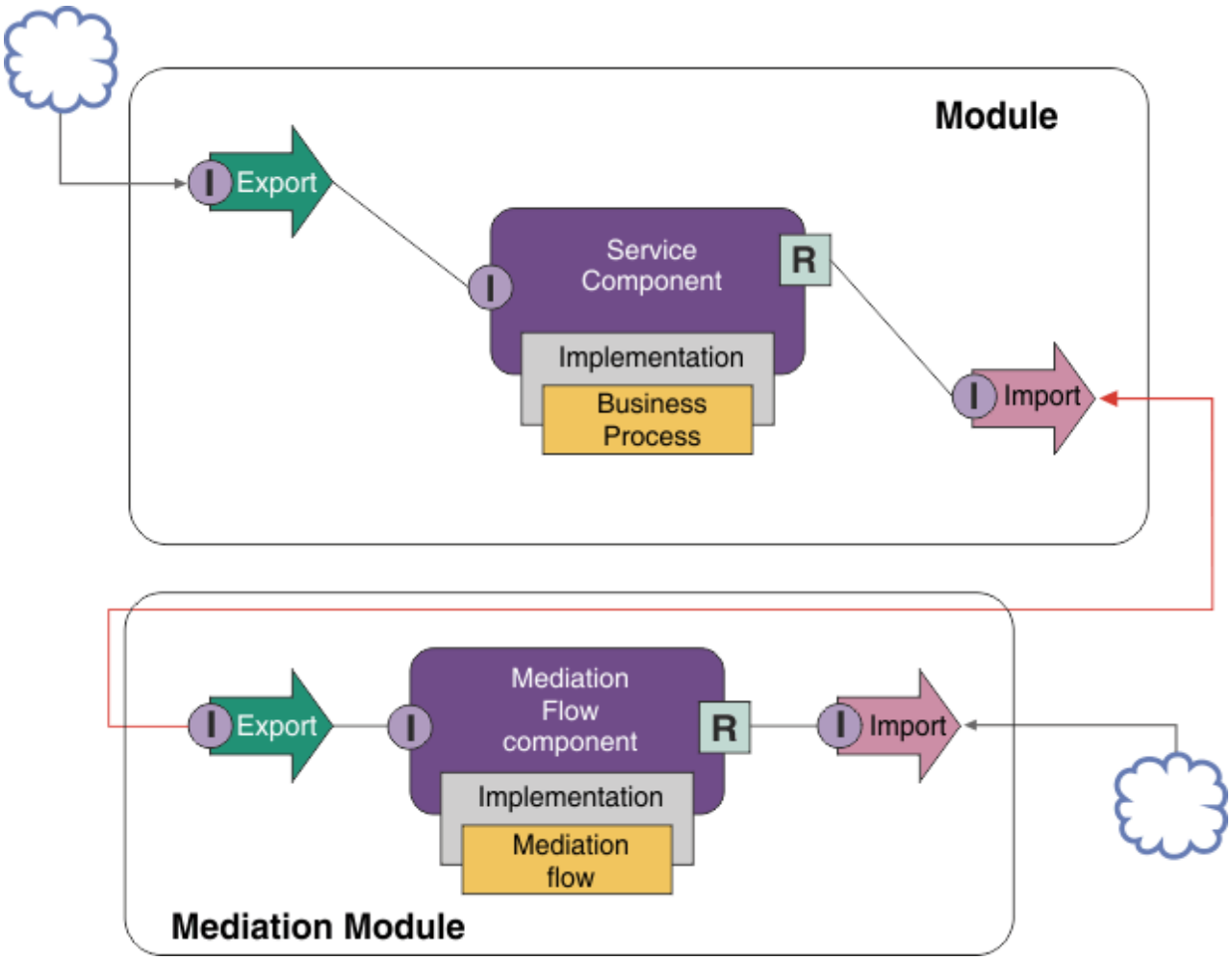
1. Erstellen Sie ein Mediationsmodul. Eine schrittweise Anleitung finden Sie unter Mediationsmodule erstellen.
2. Erstellen Sie im Mediationsmodul einen Import mit der entsprechenden Bindung für den externen Service, auf den Sie zugreifen möchten. Eine schrittweise Anleitung finden Sie unter Importe erstellen. Weitere Informationen zu Bindungen können Sie dem Thema Bindungen entnehmen.
3. Erstellen Sie einen Export und ordnen Sie ihm dieselbe Schnittstelle wie dem Import zu. Eine schrittweise Anleitung finden Sie unter Exporte erstellen.
4. Generieren Sie eine SCA-Bindung für den Export. Eine schrittweise Anleitung finden Sie unter SCA-Bindungen generieren.
5. Verbinden Sie den Export in der Assemblierung des Mediationsmoduls mit dem Import. Speichern Sie das Mediationsmodul.

6. Erstellen Sie ein Modul. Eine schrittweise Anleitung finden Sie unter Modul für Geschäftsservices erstellen.
7. Fügen Sie einen Export und eine Komponente hinzu.
8. Ziehen Sie den Export, den Sie (in Schritt 4) im Mediationsmodul erstellt haben, in der Sicht 'Geschäftsintegration' in die Modulassemblierung. Daraufhin wird ein Import mit derselben Bindung wie der Export erstellt.
9. Verbinden Sie den Export mit der Komponente und die Komponente mit dem Import.
10. Fügen Sie die Implementierung der Komponente hinzu. Informationen zu Implementierungstypen können Sie dem Thema Implementierungen entnehmen.

Später können Sie Mediationslogik wie beispielsweise Protokollierung oder Weiterleitung zum Mediationsmodul hinzufügen, ohne hierdurch das Geschäftsmodul zu beeinflussen.

Mediation hinzufügen

Manchmal reicht es nicht aus, einfach einen externen Service aufzurufen. Bisweilen müssen Sie zunächst eine Verarbeitung durchführen, indem Sie ein Mediationsmodul als Vermittler zwischen dem Serviceanforderer und dem Service-Provider hinzufügen.



Der vermittelnde Mediationsablauf führt unter anderem die folgenden Funktionen aus:

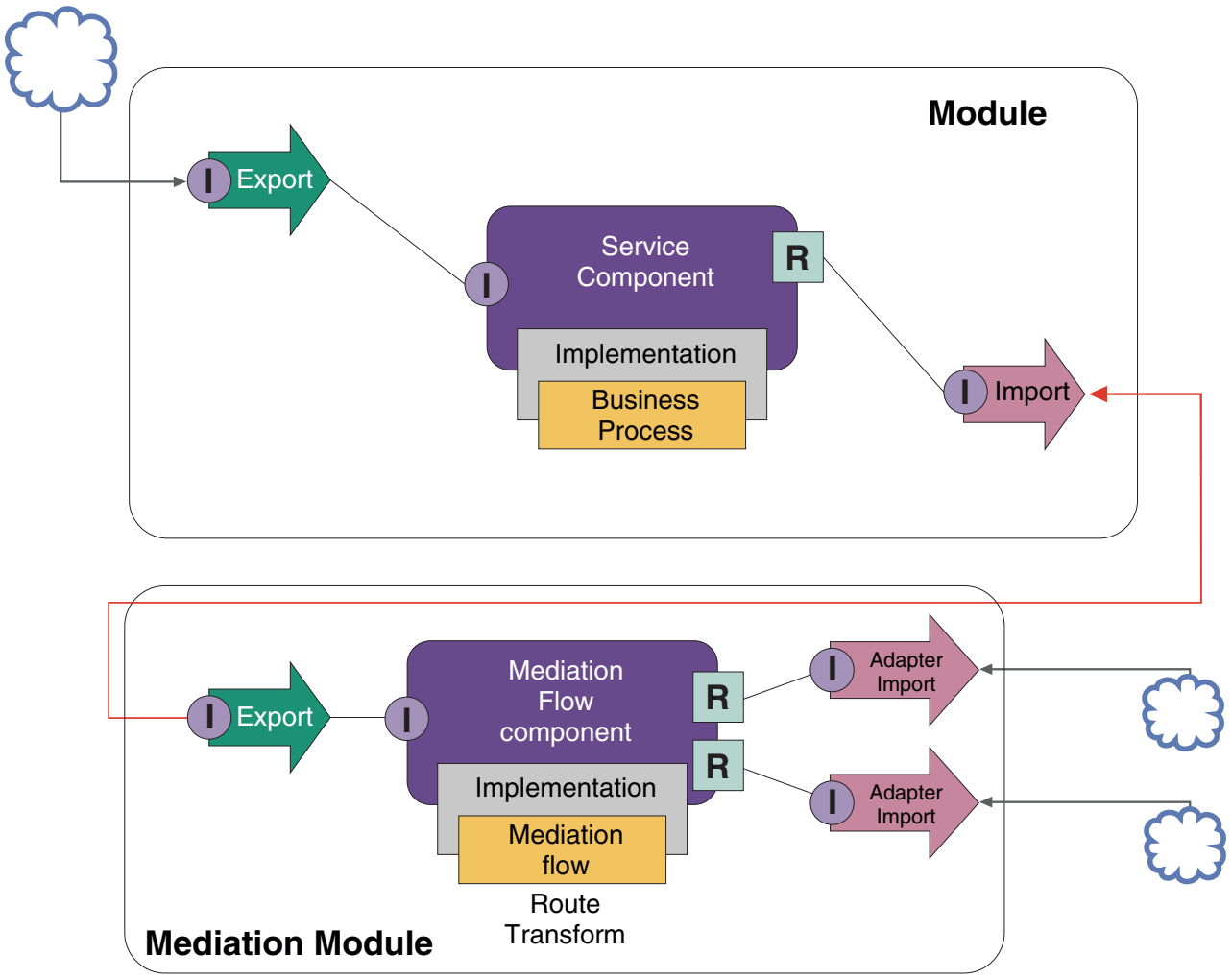
- Festlegung von Protokollheadern. Weitere Informationen finden Sie im Abschnitt Protocol Konvertierung im WebSphere Enterprise Service Bus Information Center.

- Schnittstellen- oder Parametertransformation unter Verwendung eines Basiselements für Geschäftsobjektzuordnung oder eines Basiselements für Zuordnung (siehe Umsetzen von Nachrichten).
- Auswahl eines bestimmten Service aus einer statischen Liste unter Verwendung des Basiselements 'Nachrichtenfilter' (siehe Nachrichtenfilter).
- Aufruf mehrerer Services für die Zusammenfassung der Ergebnisse unter Verwendung der Basiselemente 'Eingabefächerung' und 'Ausgabefächerung' (siehe Nachrichten zusammenfassen und rundsenden).
- Behandlung von fehlgeschlagenen Serviceaufrufen durch die Wiederholung desselben Service oder durch den Aufruf eines anderen Service unter Verwendung des Basiselements 'Serviceaufruf' (siehe Fehlgeschlagenen Serviceaufruf wiederholen).
- Dynamisches Routing durch die Auswahl des zu verwendenden Service zur Laufzeit und nicht zur Integrationszeit. Dies ermöglicht eine flexiblere Verbindung von Services und eine schnellere Reaktion von Unternehmen auf Änderungen. Neue Services können hinzugefügt werden, ohne dass eine Bearbeitung der Module erforderlich ist, die in der Laufzeitumgebung implementiert sind. Die Leistungsfähigkeit des dynamischen Routings zeigt sich insbesondere dann, wenn es in Kombination mit einer Registry verwendet wird, was die Verwendung des Mediationsbasiselements 'Endpunktsuche' erforderlich macht (siehe Endpunkte dynamisch auswählen).

Auf unternehmensweite Informationssysteme zugreifen

Services und Artefakte auf externen Systemen können in Integration Designer importiert werden. Ein Assistent erkennt Anwendungen und Daten in einem unternehmensweiten Informationssystem (Enterprise Information System, EIS) und ermöglicht die Generierung von Services aus den erkannten Anwendungen und Daten. Bei den generierten Artefakten handelt es sich um Schnittstellen und Geschäftsobjekte, die von Komponenten in einem Modul verwendet werden können.

Der Einsatz eines vermittelnden Mediationsmoduls zwischen einem Modul und einem Hostsystem steigert die Wiederverwendbarkeit des Moduls. Im nachfolgenden Beispiel wird ein Mediationsablauf zum Weiterleiten an das richtige Hostsystem und zum Umwandeln der Daten in das erforderliche Format verwendet.



Dieses Beispiel wird durch die folgenden übergeordneten Tasks realisiert:

1. Stellen Sie mit dem Assistenten 'Externer Service' eine Verbindung zum Hostsystem her. Die Verwendung des Assistenten 'Externer Service' ist ungeachtet des verwendeten Adapters immer ähnlich. Informationen zur Vorgehensweise bei der Verwendung des Assistenten 'Externer Service' finden Sie unter Zugriffsmuster für externe Services mit Adaptern.
2. Erstellen Sie ein Modul. Eine schrittweise Anleitung finden Sie unter Modul für Geschäftsservices erstellen.
3. Fügen Sie einen Export, eine Komponente und einen Import mit einer SCA-Bindung hinzu. Weitere Informationen finden Sie unter Services aufrufen.
4. Fügen Sie eine Schnittstelle zum Export hinzu und verbinden Sie den Export mit der Komponente.
5. Fügen Sie die Implementierung der Komponente hinzu. Legen Sie in der Implementierung eine Eigenschaft fest, die angibt, auf welchen Host-Service zugriffen wird. Informationen zu Implementierungstypen können Sie dem Thema Implementierungen entnehmen.
6. Erstellen Sie ein Mediationsmodul mit einem Export, der eine SCA-Bindung besitzt, und derselben Schnittstelle wie der Import des Moduls, das Sie in Schritt 2 erstellt haben.
7. Verbinden Sie den Export mit einer Mediationsablaufkomponente.
8. Erstellen Sie für jedes Hostsystem, auf das Sie zugreifen wollen, einen Import. Verwenden Sie hierbei den geeigneten abgehenden Adapter aus der Palette im Assembly-Editor.
9. Verbinden Sie die Mediationsablaufkomponente mit den Importen.

10. Implementieren Sie die Mediationsablaufkomponente. Verwenden Sie ein Basiselement 'Nachrichtenfilter', um den Import basierend auf einer Eigenschaftengruppe in der Geschäftslogik auszuwählen, und verwenden Sie für jeden Adapterimport ein Basiselement 'Zuordnung'.
11. Wählen Sie im Modul den Export des Mediationsmoduls als Service aus, der in das Modul importiert werden soll. Eine schrittweise Anleitung enthält der Abschnitt Service aus einem anderen Modul aufrufen.

Später können Sie mit nur geringfügigen Auswirkungen auf die Geschäftslogik Änderungen vornehmen, beispielsweise einen Adapter hinzufügen oder einen Adapter so ändern, dass er auf ein anderes Hostsystem verweist.

Auf Messaging-Systeme zugreifen

Damit ein SCA-Modul mit einem vorhandenen JMS-, MQ- oder MQ-JMS-Messaging-Client kommunizieren kann, müssen Sie Schnittstellen, Geschäftsobjekte sowie Bindungen für Importe und Exporte erstellen. Weitere Informationen finden Sie unter Nachricht einer SCA-Schnittstelle zuordnen.

Ein Mediationsablauf verwendet Nachrichten, die zusätzlich zu Geschäftsobjekten Zugriff auf Kontext- und Headerinformationen bieten. Wenn Sie auf JMS-Headerinformationen oder auf eine angepasste JMS-Eigenschaft zugreifen wollen, verwenden Sie einen Mediationsablauf. Wenn die die Integration mit einem MQ-System erfolgt und auf die MQ-Headerinformationen zugegriffen werden soll, müssen Sie einen Mediationsablauf verwenden.

Web-Service erstellen oder aufrufen

Web-Services sind eigenständige Anwendungen, die Geschäftsfunktionen ausführen. Dies kann von einer einfachen Abfrage bis hin zu komplexen Geschäftsprozessinteraktionen reichen. Sie können einen bestehenden Web-Service aufrufen oder einen neuen Web-Service Ihren Anforderungen entsprechend entwickeln. In diesem Szenario werden die zugehörigen Schritte beschrieben und weitere Informationen bereitgestellt.

Auch wenn Sie nicht alle Services von Grund auf mit IBM Integration Designer erstellen, müssen manche Services tatsächlich auf diese Weise erstellt werden. Wenn Sie Services in einem Geschäftsprozess mit dem Assembly-Editor oder dem Geschäftsprozesseditor assemblieren, werden Sie wahrscheinlich bemerken, dass manche Services fehlen. Es kann daher hilfreich sein, diese fehlenden Services mit den Tools von IBM Integration Designer zu erstellen. Gleiches gilt in umgekehrter Richtung: Möglicherweise stellen Sie nach der Erstellung eines neuen Prozesses fest, dass es sinnvoll wäre, alle oder einen Teil der Prozessoperationen in Form eines Service für Andere zur Verfügung zu stellen.

Anmerkung: Dieses Szenario ist für Benutzer von IBM Integration Designer für IBM Process Server anwendbar.

Es gibt verschiedene Gründe für die Entwicklung von Web-Services mit IBM Integration Designer:

- Bei der Erstellung von Services in IBM Integration Designer haben Sie die Möglichkeit, den Service unter Verwendung von Business-Regeln zu implementieren.
- Bei der Entwicklung in IBM Integration Designer können Sie einen Java™-Service entwickeln und diesen sowohl als Web-Service als auch über SCA zugänglich machen.
- Von Vorteil ist es, dass für die Schnittstellenzuordnung keine Codierung erstellt werden muss. Sie können alle Datenzuordnungen aus dem Java-Code herausnehmen und stellen dem Java-Entwickler dadurch eine einfache Funktionseinheit für ein Java-Programm zur Verfügung.
- In IBM Integration Designer werden alle Services und Beziehungen gemeinsam angezeigt.
- Des Weiteren unterstützt Sie die Refactoringfunktionalität bei der Entwicklung von Web-Services mit IBM Integration Designer.

Bitte bedenken Sie hierbei stets, dass Web-Services nicht als Lösung für alle Integrationsprobleme verstanden werden dürfen. Wie alle anderen Technologie- oder Architekturkonzepte bietet jedoch auch die Verwendung von Web-Services an der richtigen Stelle und zum richtigen Zeitpunkt spezielle Vorteile.

Exporte, Importe und Bindungen

In IBM Integration Designer können Sie standardmäßige Web-Services importieren und diese Services in Ihren Verbundanwendungen einsetzen.

Die Entwicklung von Services wird in IBM Integration Designer im Assembly-Editor vorgenommen. Für die Erstellung von Modulen, Mediationsmodulen, Bibliotheken und Komponenten kann ein Standardverfahren verwendet werden. Anschließend können Sie mit Exporten, Importen und Bindungen auf diese Services zugreifen und sie gemeinsam nutzen. Die Schritte für diese Basistasks sind nachfolgend aufgeführt. Über die Links können Sie detailliertere Informationen zu den einzelnen Tasks aufrufen.

Für Web-Services können Sie eine von zwei Bindungen verwenden, nämlich eine Web-Service-Bindung oder eine HTTP-Bindung. Eine Web-Service-Bindung stellt eine Spezifikation für die Übertragung von Nachrichten zu und von einem Web-Service bereit. Die Tools helfen Ihnen dabei, eine Web-Service-Bindung automatisch zu generieren. Eine HTTP-Bindung ist ein Standardprotokoll für Anforderungen und Antworten zwischen Clients und Servern; dies wurde in dem HTTP-Protokoll definiert, das vom World Wide Web Consortium (W3C) veröffentlicht wurde. Falls Sie eine HTTP-Bindung verwenden, müssen Sie einige anfängliche Bindungskonfigurationsinformationen bereitstellen.

1. Erstellen Sie einen Export, um den Service des Moduls zur Verwendung durch andere Module zu veröffentlichen.
2. Generieren Sie eine Bindung für den Export.
   - Generieren Sie eine Web-Service-Bindung für den Export.
   - Generieren Sie eine HTTP-Exportbindung.
3. Erstellen Sie einen Import, um einen vorhandenen Service aufzurufen, der nicht Bestandteil des von Ihnen assemblierten Moduls ist.
   - Generieren Sie eine Web-Service-Bindung für den Import.
   - Generieren Sie eine HTTP-Importbindung.

Lesen Sie das verlinkte Thema, wenn Sie einen Web-Service aus JavaServer Pages aufrufen wollen.

Funktionalität für die Web-Service-Entwicklung

Beim Öffnen eines Editors, der am Erstellungsprozess für Web-Services beteiligt ist, wird möglicherweise das Fenster 'Aktivierung bestätigen' angezeigt, das die folgenden Informationen enthält:

```
This action requires the enablement of "Web Services Deployment".
Enable the required capability? (Diese
Aktion erfordert die Aktivierung von "Web Services Deployment". Soll die erforderliche Funktionalität aktiviert werden?)
```

IBM Integration Designer bietet eine als *Leistungsspektrum* bezeichnete Filterfunktion. In den Einstellungen der Benutzervorgaben sind die Funktionen und Tools in Kategorien eingeteilt. Sie können Kategorien des Leistungsspektrums oder eine Untergruppe der Funktionen in einer Kategorie aktivieren bzw. inaktivieren. Weitere Informationen finden Sie im Thema über das Leistungsspektrum.

Weitere Informationen zu Schlüsselkonzepten

Dieser Abschnitt dient als Ausgangspunkt zum Recherchieren der Technologien, die in IBM Business Process Manager zum Einsatz kommen.

Authoringszenarios

Mithilfe von Szenarios können Sie ein besseres Verständnis der Komponenten und Produkte in der BPM-Produktfamilie gewinnen und mit diesen Komponenten und Produkten arbeiten.

Versionierung

Der Lebenszyklus einer Prozessanwendung beginnt mit deren Erstellung und setzt sich über einen Kreislauf aus Aktualisierung, Implementierung, Co-Implementierung, Deimplementierung und Archivierung fort. Der Begriff *Versionierung* (auch: Versionssteuerung) bezeichnet einen Mechanismus zur Steuerung des Lebenszyklus einer Prozessanwendung durch eindeutiges Identifizieren der einzelnen Versionen dieser Prozessanwendung.

Die Arbeitsweise der Versionierung in IBM Business Process Manager ist davon abhängig, ob Sie eine Prozessanwendung implementieren (aus dem Repository in IBM Process Center) oder eine Unternehmensanwendung (direkt aus IBM Integration Designer).

Die Prozessanwendungen und Toolkits, die Sie aus Process Center in einer Laufzeitumgebung implementieren, sind standardmäßig versioniert. Bei Unternehmensanwendungen können Sie auswählen, dass Module und Bibliotheken in IBM Integration Designer versioniert werden.

Darüber hinaus können Sie Versionen einer Benutzertask oder Statusmaschine erstellen, damit mehrere Versionen der Task oder Statusmaschine gemeinsam mit der Laufzeitumgebung existieren können.

Versionierung von Prozessanwendungen

Bei der Versionierung handelt es sich um die Funktionalität, die es der Laufzeitumgebung ermöglicht, Snapshots im Lebenszyklus einer Prozessanwendung zu identifizieren und mehrere Snapshots auf einem Prozessserver auszuführen.

Um zu verstehen, wie eine Versionierung für Prozessanwendungen durchgeführt wird, ist es wichtig zu beachten, dass eine Prozessanwendung ein Container ist, der verschiedene Artefakte enthält, die in der oder von der Prozessanwendung verwendet werden (zum Beispiel Prozessmodelle oder BPDs, Toolkit-Referenzen, Services, oder Verfolgungen). Eine Versionierung findet auf Container-Ebene und nicht auf der Ebene der einzelnen Artefakte statt. Bei Prozessanwendungen bedeutet dies, dass eine Versionierung stattfindet, wenn Sie einen Snapshot aufnehmen.

Sie können Snapshots vergleichen, um die Unterschiede zwischen den Versionen zu bestimmen. Hat ein Entwickler zum Beispiel ein Problem mit einem Service behoben und an dieser Stelle einen Snapshot der übergeordneten Prozessanwendung oder des übergeordneten Toolkits aufgenommen, damit anschließend ein anderer Entwickler mehrere zusätzliche Änderungen an demselben Service vornimmt und einen neuen Snapshot aufnimmt, so könnte der Projektmanager die beiden Snapshots miteinander vergleichen, um festzustellen, welche Änderungen wann und von wem vorgenommen wurden. Falls der Projektmanager die zusätzlichen Änderungen am Service als unnötig einschätzt, könnte er zum Snapshot der ursprünglichen Korrektur zurückkehren.

Sie können verschiedene Versionen (Snapshots) einer Prozessanwendung gleichzeitig auf einem Server ausführen. Wenn Sie einen neuen Snapshot installieren, entfernen Sie das Original oder führen Sie es weiterhin aus.

Versionskontext

Jeder Snapshot verfügt über eindeutige Metadaten, um die Version zu identifizieren (als Versionskontext bezeichnet). Diese Kennung wird von Ihnen zugeordnet; IBM empfiehlt hingegen die Verwendung eines numerischen Versionssystems, das aus drei Ziffern im Format <major>.<minor>.<service> besteht. Die Abschnitte zu Namenskonventionen enthalten eine ausführlichere Beschreibung dieses Versionierungsschemas.

IBM Business Process Manager ordnet jeder Prozessanwendung einen globalen Namensbereich zu. Der globale Namensbereich ist entweder ein Vorschlag der Prozessanwendung oder ein bestimmter Snapshot der Prozessanwendung. Der vom Server verwendete Versionsname darf aus maximal sieben Zeichen bestehen; deshalb ist der zugeordnete Name ein Akronym, das Zeichen aus dem angegebenen Snapshotnamen verwendet. Snapshotakronyme sind mit ihren Snapshotnamen identisch, wenn die Snapshotnamen dem empfohlenen IBM VRM-Stil entsprechen und aus maximal sieben Zeichen bestehen. Beispiel: Der Snapshotname 1.0.0 verfügt über das Akronym 1.0.0 und der Snapshotname 10.3.0 verfügt über das Akronym 10.3.0. Das Snapshotakronym ist innerhalb des Kontexts der Prozessanwendung im Geltungsbereich des Process Center Servers garantiert eindeutig. Aus diesem Grund kann das Snapshot-Akronym nicht bearbeitet werden.

Versionisierungsaspekte für Prozessanwendungen in mehreren Clustern

Es ist möglich, dieselbe Version einer Prozessanwendung auf verschiedenen Clustern derselben Zelle zu implementieren. Zur Differenzierung der einzelnen Implementierungen derselben Version von Prozessanwendung sollten Sie für jede Implementierung einen Snapshot erstellen und bei der Benennung des Snapshots eine für die Zelle eindeutige ID in den Namen einbinden (zum Beispiel 'v1.0_cell1_1' und 'v1.0_cell1_2'). Es handelt sich (aus der Perspektive des Lebenszyklusmanagements) bei jedem Snapshot um eine neue Version der Prozessanwendung, die denselben Inhalt und dieselbe Funktion besitzt.

Wenn Sie eine Prozessanwendung auf einem Cluster implementieren, wird eine automatische Synchronisation der Knoten ausgeführt.

Versionisierungsaspekte für Process Designer-Toolkits

Beachten Sie, dass Snapshots von Prozessanwendungen im Allgemeinen aufgenommen werden, wenn Sie getestet oder implementiert werden soll. Toolkit-Snapshots werden aber in der Regel aufgenommen, wenn dieses Toolkit von Prozessanwendungen verwendet werden soll. Falls Sie danach Aktualisierungen an dem Toolkit vornehmen möchten, müssen Sie, wenn Sie soweit sind, einen weiteren Snapshot der aktuellen Arbeitsversion erstellen und die Eigner von Prozessanwendungen und Toolkits können dann entscheiden, ob sie zu dem neuen Snapshot wechseln möchten.

Versionierung von Modulen und Bibliotheken

Wenn sich ein Modul oder eine Bibliothek in einer Prozessanwendung oder einem Toolkit befindet, übernimmt das Modul bzw. die Bibliothek den Lebenszyklus der betreffenden Prozessanwendung bzw. des betreffenden Toolkits (Versionen, Snapshots, Verfolgungen usw.). Modul- und Bibliotheksnamen müssen im Geltungsbereich einer Prozessanwendung oder eines Toolkits eindeutig sein.

In diesem Abschnitt wird die Versionierung von Modulen und Bibliotheken, die zusammen mit Prozessanwendungen verwendet werden, beschrieben. Hinweis: Wenn Sie Module direkt aus IBM Integration Designer in Process Server implementieren, können Sie weiterhin so verfahren, dass Sie Modulen während der Implementierung Versionsnummern zuweisen, wie im Abschnitt „Versionierte Module und Bibliotheken erstellen" beschrieben.

Die abhängigen Bibliotheken eines Moduls oder einer Bibliothek, das bzw. die IBM Process Center zugeordnet ist, müssen sich in derselben Prozessanwendung oder in einem abhängigen Toolkit befinden.

In der folgenden Tabelle sind die Optionen aufgeführt, die Sie im Abhängigkeitseditor von IBM Integration Designer auswählen können, wenn eine Bibliothek einer Prozessanwendung oder einem Toolkit zugeordnet ist:

Tabelle 7. Abhängigkeiten für Modul, Prozessanwendung oder Toolkit und globale Bibliotheken

Geltungsbereich der Bibliothek	Beschreibung	Kann abhängig sein von . . .
Modul	Eine Kopie dieser Bibliothek ist für jedes Modul, das sie verwendet, auf dem Server vorhanden.	Eine Bibliothek im Modulgeltungsbereich kann von allen Bibliothekstypen abhängig sein.
Prozessanwendung oder Toolkit	Die Bibliothek wird von allen Modulen im Geltungsbereich der Prozessanwendung bzw. des Toolkits gemeinsam genutzt. Diese Einstellung wird wirksam, wenn die Implementierung über IBM Process Center erfolgt. Wenn die Implementierung außerhalb von IBM Process Center erfolgt, dann wird die Bibliothek in jedes Modul kopiert. **Anmerkung:** Bibliotheken, die in IBM Integration Designer Version 8 erstellt werden, verfügen standardmäßig über die gemeinsame Nutzungsebene **Prozessanwendung oder Toolkit**.	Eine Bibliothek dieses Typs kann nur von globalen Bibliotheken abhängig sein.
Global	Die Bibliothek wird von allen aktiven Modulen gemeinsam genutzt.	Eine globale Bibliothek kann nur von anderen globalen Bibliotheken abhängig sein. **Anmerkung:** Zur Implementierung der globalen Bibliothek müssen Sie eine gemeinsam genutzte WebSphere-Bibliothek konfigurieren. Weitere Informationen finden Sie unter „Modul- und Bibliotheksabhängigkeiten".

Module und Bibliotheken, die Prozessanwendungen oder Toolkits zugeordnet sind

Für Module und Bibliotheken, die Prozessanwendungen oder Toolkits zugeordnet sind, ist keine Versionierung erforderlich.

Module und Bibliotheken, die einer Prozessanwendung oder einem Toolkit zugeordnet sind, brauchen nicht versioniert zu werden. Tatsächlich können Sie im Editor für Abhängigkeiten auch keine Version eines Moduls oder einer Bibliothek erstellen, das bzw. die einer Prozessanwendung oder einem Toolkit zugeordnet ist. Module und Bibliotheken, die einer Prozessanwendung oder einem Toolkit zugeordnet sind, arbeiten mit Snapshots (Momentaufnahmen), einer Funktion im Process Center, um dasselbe Ergebnis wie eine Version zu erzielen.

Bibliotheken, die einer Prozessanwendung oder einem Toolkit zugeordnet sind, haben keine erforderliche Versionsnummer im Abschnitt **Bibliotheken** des Editors für Abhängigkeiten, da keine Version benötigt wird.

Namenskonventionen

Eine Namenskonvention dient zur Unterscheidung der verschiedenen Versionen einer Prozessanwendung, wenn diese den Kreislauf aus Aktualisierung, Implementierung, Co-Implementierung, Deimplementierung und Archivierung durchläuft.

In diesem Abschnitt sind die Konventionen beschrieben, die zur eindeutigen Kennzeichnung der Versionen einer Prozessanwendung verwendet werden.

Ein *Versionskontext* ist eine Kombination von Akronymen, die eine Prozessanwendung oder ein Toolkit eindeutig beschreiben. Jeder Typ von Akronym hat eine Namenskonvention. Ein Akronym kann maximal sieben Zeichen aus dem Zeichensatz [A-Z0-9_] enthalten; eine Ausnahme bildet das Snapshotakronym, das außerdem einen Punkt enthalten kann.

- Das Akronym für die Prozessanwendung wird erstellt, wenn die Prozessanwendung erstellt wird. Es kann maximal sieben Zeichen lang sein.
- Das Akronym für den Snapshot wird bei der Erstellung des Snapshots automatisch erstellt. Es kann maximal sieben Zeichen lang sein.

  Falls der Snapshotname die Kriterien für ein gültiges Snapshotakronym erfüllt, sind der Snapshotname und das Akronym identisch.

  Anmerkung: Benennen Sie bei der versionssensitiven Weiterleitungsfunktion der Mediationsablaufkomponente Ihren Snapshot so, dass der Name mit dem Schema *<version>.<release>.<modifikation>* konform ist (z. B. **1.0.0**). Da das Snapshotakronym auf eine Länge von sieben Zeichen beschränkt ist, können maximal fünf Ziffern (plus zwei Punkte) verwendet werden. Daher sollten Sie bei der Erhöhung der Ziffernfelder sorgsam vorgehen, weil alle Zeichen nach den ersten sieben Zeichen abgeschnitten werden.

  Der Snapshotname **11.22.33** wird beispielsweise zum Snapshotakronym **11.22.3**.
- Das Verfolgungsakronym wird automatisch aus den Anfangsbuchstaben der Wörter des Verfolgungsnamens gebildet. Eine neue Verfolgung, die mit dem Namen **My New Track** erstellt wurde, hat beispielsweise den Akronymwert **MNT**.

  Der Standardname und das Standardakronym für die Verfolgung sind **Main**. Bei der Implementierung auf einem IBM Process Center-Server wird das Verfolgungsakronym in den Versionierungskontext aufgenommen, sofern es sich nicht um das Verfolgungsakronym **Main** handelt.

Eine Geschäftsprozessdefinition in einer Prozessanwendung wird für gewöhnlich durch das Akronym für den Prozessanwendungsnamen, das Snapshotakronym und den Namen der Geschäftsprozessdefinition gekennzeichnet. Wählen Sie für Ihre Geschäftsprozessdefinitionen nach Möglichkeit immer eindeutige Namen. Wenn Namen doppelt vorhanden sind, könnten die folgenden Probleme auftreten:

- Unter Umständen wird es nicht möglich sein, die Geschäftsprozessdefinitionen ohne eine Form der Mediation als Web-Services zugänglich zu machen.
- Unter Umständen wird es nicht möglich sein, eine in IBM Process Designer erstellte Geschäftsprozessdefinition aus einem in IBM Integration Designer erstellten BPEL-Prozess heraus aufzurufen.

Der Versionskontext variiert abhängig davon, wie die Prozessanwendung implementiert wird.

Namenskonventionen für Process Center-Serverimplementierungen:

Auf dem IBM Process Center-Server können Sie einen Snapshot einer Prozessanwendung sowie einen Snapshot eines Toolkits implementieren. Darüber hinaus können Sie die aktuelle Arbeitsversion einer Prozessanwendung oder eines Toolkits implementieren. Der Versionskontext variiert je nach Implementierungstyp.

Bei Prozessanwendungen wird die Arbeitsversion der Prozessanwendung oder der jeweilige Prozessanwendungssnapshot verwendet, um die Version eindeutig zu kennzeichnen.

Toolkits können mit einer Prozessanwendung oder mehreren Prozessanwendungen implementiert werden; der Lebenszyklus eines jeden Toolkits ist jedoch an den Lebenszyklus der Prozessanwendung gebunden. Jede Prozessanwendung verfügt über eine eigene Kopie der abhängigen Toolkits, die auf dem Server implementiert sind. Ein implementiertes Toolkit wird nicht von mehreren Prozessanwendungen gemeinsam genutzt.

Falls die Verfolgung, die der Prozessanwendung zugeordnet ist, einen anderen Namen als den Standardnamen **Main** (Hauptelement) hat, ist das Akronym der Verfolgung ebenfalls Bestandteil des Versionskontextes.

Weitere Informationen finden Sie unter „Beispiele” auf Seite 35 im weiteren Verlauf dieses Abschnitts.

Prozessanwendungssnapshots

Bei Implementierungen von Prozessanwendungssnapshots ist der Versionskontext eine Kombination aus den folgenden Elementen:

- Akronym des Prozessanwendungsnamens
- Akronym der Prozessanwendungsverfolgung (bei anderer Verfolgung als **Main**)
- Akronym des Prozessanwendungssnapshots

Eigenständige Toolkits

Bei Implementierungen von Toolkit-Snapshots ist der Versionskontext eine Kombination aus den folgenden Elementen:

- Akronym des Toolkitnamens
- Akronym der Toolkitverfolgung (bei anderer Verfolgung als **Main**)
- Akronym des Toolkit-Snapshots

Arbeitsversionen

Arbeitsversionen ('Tips') von Prozessanwendungen werden während der iterativen Erprobung in Process Designer verwendet. Sie können nur auf Process Center-Servern implementiert werden.

Bei Implementierungen von Arbeitsversionen der Prozessanwendungen ist der Versionskontext eine Kombination aus den folgenden Elementen:

- Akronym des Prozessanwendungsnamens
- Akronym der Prozessanwendungsverfolgung (bei anderer Verfolgung als **Main**)
- 'Tip'

Toolkitarbeitsversionen werden ebenfalls während der iterativen Erprobung in Process Designer verwendet. Sie werden nicht auf einem Produktionsserver implementiert.

Bei Implementierungen von Toolkitarbeitsversionen ist der Versionskontext eine Kombination aus den folgenden Elementen:

- Akronym des Toolkitnamens
- Akronym der Toolkitverfolgung (bei anderer Verfolgung als **Main**)
- 'Tip'

Beispiele

Ressourcen sollten unter Verwendung des Versionskontextes eindeutig benannt und extern gekennzeichnet werden.

- Die folgende Tabelle enthält einige Beispiele für eindeutige Namen. In diesem Beispiel verwendet eine aktuelle Arbeitsversion einer Prozessanwendung den Standardverfolgungsnamen (**Main**):

Tabelle 8. Aktuelle Arbeitsversion einer Prozessanwendung mit Standardverfolgungsnamen

Namenstyp	**Beispiel**
Prozessanwendungsname	**Process Application 1**
Akronym des Prozessanwendungsnamens	**PA1**
Prozessanwendungsverfolgung	**Main**
Akronym der Prozessanwendungsverfolgung	"" (bei Verfolgung **Main**)
Prozessanwendungssnapshot	

Tabelle 8. Aktuelle Arbeitsversion einer Prozessanwendung mit Standardverfolgungsnamen (Forts.)

Namenstyp	**Beispiel**
Akronym des Prozessanwendungssnapshots	**Tip**

Alle SCA-Module, die dieser aktuellen Arbeitsversion der Prozessanwendung zugeordnet sind, beinhalten den Versionskotext, wie in der folgenden Tabelle dargestellt:

Tabelle 9. SCA-Module und versionssensitive EAR-Dateien

Name des SCA-Moduls	**Versionssensitiver Name**	**Versionssensitiver EAR-/Anwendungsname**
M1	PA1-Tip-M1	PA1-Tip-M1.ear
M2	PA1-Tip-M2	PA1-Tip-M2.ear

- Die folgende Tabelle enthält ein Beispiel für eine aktuelle Arbeitsversion einer Prozessanwendung, die nicht den Standardverfolgungsnamen verwendet.

Tabelle 10. Aktuelle Arbeitsversion einer Prozessanwendung, die nicht den Standardverfolgungsnamen verwendet

Namenstyp	**Beispiel**
Prozessanwendungsname	**Process Application 1**
Akronym des Prozessanwendungsnamens	**PA1**
Prozessanwendungsverfolgung	**Track1**
Akronym der Prozessanwendungsverfolgung	**T1**
Prozessanwendungssnapshot	
Akronym des Prozessanwendungssnapshots	**Tip**

Alle SCA-Module, die dieser aktuellen Arbeitsversion der Prozessanwendung zugeordnet sind, beinhalten den Versionskotext, wie in der folgenden Tabelle dargestellt:

Tabelle 11. SCA-Module und versionssensitive EAR-Dateien

Name des SCA-Moduls	**Versionssensitiver Name**	**Versionssensitiver EAR-/Anwendungsname**
M1	PA1-T1-Tip-M1	PA1-T1-Tip-M1.ear
M2	PA1-T1-Tip-M2	PA1-T1-Tip-M2.ear

Ähnliche Namenskonventionen gelten auch für erweiterte Implementierungen von Toolkitarbeitsversionen und -snapshots. Sie gelten ebenso für erweiterte Snapshots bei einer Installation auf Process Server.

- Die folgende Tabelle enthält einige Beispiele für eindeutige Namen. In diesem Beispiel verwendet ein Prozessanwendungssnapshot den Standardverfolgungsnamen (**Main**):

Tabelle 12. Prozessanwendungssnapshot mit Standardverfolgungsnamen

Namenstyp	**Beispiel**
Prozessanwendungsname	**Process Application 1**
Akronym des Prozessanwendungsnamens	**PA1**
Prozessanwendungsverfolgung	**Main**
Akronym der Prozessanwendungsverfolgung	"" (bei Verfolgung **Main**)
Prozessanwendungssnapshot	**Prozessanwendungssnapshot V1**
Akronym des Prozessanwendungssnapshots	**PSV1**

Alle SCA-Module, die diesem Prozessanwendungssnapshot zugeordnet sind, beinhalten den Versionskotext, wie in der folgenden Tabelle dargestellt:

Tabelle 13. SCA-Module und versionssensitive EAR-Dateien

Name des SCA-Moduls	Versionssensitiver Name	Versionssensitiver EAR-/Anwendungsname
M1	PA1-PSV1-M1	PA1-PSV1-M1.ear
M2	PA1-PSV1-M2	PA1-PSV1-M2.ear

- Die folgende Tabelle enthält ein Beispiel für einen Prozessanwendungssnapshot, der nicht den Standardverfolgungsnamen verwendet.

Tabelle 14. Prozessanwendungssnapshot, der nicht den Standardverfolgungsnamen verwendet

Namenstyp	Beispiel
Prozessanwendungsname	**Process Application 1**
Akronym des Prozessanwendungsnamens	**PA1**
Prozessanwendungsverfolgung	**Track1**
Akronym der Prozessanwendungsverfolgung	**T1**
Prozessanwendungssnapshot	**Prozessanwendungssnapshot V1**
Akronym des Prozessanwendungssnapshots	**PSV1**

Alle SCA-Module, die diesem Prozessanwendungssnapshot zugeordnet sind, beinhalten den Versionskotext, wie in der folgenden Tabelle dargestellt:

Tabelle 15. SCA-Module und versionssensitive EAR-Dateien

Name des SCA-Moduls	Versionssensitiver Name	Versionssensitiver EAR-/Anwendungsname
M1	PA1-T1-PSV1-M1	PA1-T1-PSV1-M1.ear
M2	PA1-T1-PSV1-M2	PA1-T1-PSV1-M2.ear

Namenskonventionen für Process Server-Implementierungen:

Auf dem Process Server können Sie einen Snapshot der Prozessanwendung implementieren. Mit dem Akronym des Prozessanwendungssnapshots wird die Version eindeutig gekennzeichnet.

Bei Implementierungen von Prozessanwendungssnapshots ist der Versionskontext eine Kombination aus den folgenden Elementen:

- Akronym des Prozessanwendungsnamens
- Akronym des Prozessanwendungssnapshots

Ressourcen sollten unter Verwendung des Versionskontextes eindeutig benannt und extern gekennzeichnet werden. Die folgende Tabelle enthält einige Beispiele für eindeutige Namen:

Tabelle 16. Beispiele von Namen und Akronymen

Namenstyp	Beispiel
Prozessanwendungsname	**Process Application 1**
Akronym des Prozessanwendungsnamens	**PA1**
Prozessanwendungssnapshot	**1.0.0**
Akronym des Prozessanwendungssnapshots	**1.0.0**

Bei einer Ressource, wie z. B. einem Modul oder einer Bibliothek, ist der Versionskontext Bestandteil der Identität der Ressource.

Das Beispiel in der folgenden Tabelle umfasst zwei Module und beschreibt, wie die zugeordneten EAR-Dateien den Versionskontext beinhalten:

Tabelle 17. SCA-Module und versionssensitive EAR-Dateien

Name des SCA-Moduls	Versionssensitiver Name	Versionssensitiver EAR-/Anwendungsname
M1	PA1-1.0.0-M1	PA1-1.0.0-M1.ear
M2	PA1-1.0.0-M2	PA1-1.0.0-M2.ear

Das Beispiel in der folgenden Tabelle umfasst zwei Bibliotheken auf Prozessanwendungsebene und beschreibt, wie die zugeordneten JAR-Dateien den Versionskontext beinhalten:

Tabelle 18. Bibliotheken auf Prozessanwendungsebene und versionssensitive JAR-Dateien

Name der SCA-Bibliothek auf Prozessanwendungsebene	Versionssensitiver Name	Name der versionssensitiven JAR-Datei
Lib1	PA1-1.0.0-Lib1	PA1-1.0.0-Lib1.jar
Lib2	PA1-1.0.0-Lib2	PA1-1.0.0-Lib2.jar

Versionssensitive Bindungen

Prozessanwendungen können SCA-Module enthalten, die Import- und Exportbindungen umfassen. Wenn Sie Anwendungsversionen zusammen implementieren, muss die Bindung für jede Version der Anwendung eindeutig sein. Einige Bindungen werden während der Implementierung automatisch aktualisiert, um die Eindeutigkeit der Versionen sicherzustellen. In anderen Fällen müssen Sie die Bindung nach der Implementierung aktualisieren, damit die Eindeutigkeit gewährleistet ist.

Der Geltungsbereich einer *versionssensitiven* Bindung ist an eine bestimmte Version einer Prozessanwendung gebunden. Dies stellt ihre Eindeutigkeit bei Prozessanwendungen sicher. In den folgenden Abschnitten sind die Bindungen aufgeführt, die automatisch in versionssensitive Bindungen geändert werden, sowie sämtliche Aktionen, die Sie zur Laufzeit ausführen müssen, wenn eine Bindung nicht versionssensitiv ist. Überlegungen, die bei der Erstellung von Modulen berücksichtigt werden sollten, enthält der Abschnitt „Hinweise zur Verwendung von Bindungen".

SCA

Das Ziel einer SCA-Bindung wird zur Sicherstellung der Versionssensitivität automatisch während der Implementierung umbenannt, wenn die Import- und Exportbindungen des Moduls in demselben Geltungsbereich der Prozessanwendung definiert sind.

Falls die Bindungen nicht in demselben Geltungsbereich der Prozessanwendung definiert sind, wird eine Informationsnachricht protokolliert. Sie müssen die Importbindung nach der Implementierung bearbeiten, um die Endpunktzieladresse zu ändern. Um die Endpunktzieladresse zu ändern, können Sie die Administrationskonsole verwenden.

Web-Service (JAX-WS oder JAX-RPC)

Die Endpunktzieladresse einer Web-Service-Bindung wird während der Implementierung automatisch versionssensitiv umbenannt, wenn alle folgenden Bedingungen erfüllt sind:

- Die Standardnamenskonvention wurde für die Adresse berücksichtigt:
  http://*ip*:*port*/*modulname*Web/sca/*exportname*
- Die Endpunktadresse ist SOAP/HTTP.
- Die Import- und Exportbindungen des Moduls sind in demselben Geltungsbereich der Prozessanwendung definiert.

Falls diese Bedingungen nicht erfüllt werden, wird eine Informationsnachricht protokolliert. Die Maßnahme, die Sie ergreifen müssen, ist davon abhängig, wie Sie die Prozessanwendung implementieren:

- Falls Sie Ihre Prozessanwendung zusätzlich zu anderen Versionen implementieren, müssen Sie die SOAP/HTTP-Endpunkt-URL oder die SOAP/JMS-Zielwarteschlange manuell so umbenennen, dass sie bei den Versionen der Prozessanwendung eindeutig ist. Um die Endpunktzieladresse nach der Implementierung zu ändern, können Sie die Administrationskonsole verwenden.
- Falls Sie lediglich eine einzige Version der Prozessanwendung implementieren, können Sie diese Nachricht ignorieren.

Bei Co-Implementierungen von Snapshots von SOAP- oder JMS-Web-Service-Bindungen ist die Maßnahme, die Sie ergreifen müssen, davon abhängig, wie Sie die Prozessanwendung implementieren.

- Wenn sich Import und Zielexport in derselben Prozessanwendung befinden, führen Sie die folgenden Schritte aus, bevor Sie die Prozessanwendung im Process Center veröffentlichen und den Snapshot erstellen:
  1. Ändern Sie die Endpunkt-URL des Exports. Stellen Sie sicher, dass Ziel und Verbindungsfactory eindeutig sind.
  2. Ändern Sie die Endpunkt-URL des Imports in die gleiche URL, die Sie im vorherigen Schritt für den Export angegeben haben.
- Wenn sich Import und Zielexport in unterschiedlichen Prozessanwendungen befinden, führen Sie die folgenden Schritte aus:
  1. Ändern Sie die Endpunkt-URL des Exports. Stellen Sie sicher, dass Ziel und Verbindungsfactory eindeutig sind.
  2. Veröffentlichen Sie die Prozessanwendung im Process Center.
  3. Erstellen Sie den Snapshot.
  4. Implementieren Sie die Prozessanwendung auf dem Process Server.
  5. Ändern Sie die Endpunkt-URL des entsprechenden Imports mit der WebSphere-Administrationskonsole in die gleiche URL, die Sie für den Export angegeben haben.

HTTP

Die Endpunkt-URL-Adresse einer HTTP-Bindung wird während der Implementierung automatisch versionssensitiv umbenannt, wenn alle folgenden Bedingungen erfüllt sind:

- Die Standardnamenskonvention wurde für die Adresse berücksichtigt:

  http(s)://*ip*:*port*/*modulname*Web/*kontextpfad_im_export*
- Die Import- und Exportbindungen des Moduls sind in demselben Geltungsbereich der Prozessanwendung definiert.

Falls diese Bedingungen nicht erfüllt werden, wird eine Informationsnachricht protokolliert. Die Maßnahme, die Sie ergreifen müssen, ist davon abhängig, wie Sie die Prozessanwendung implementieren:

- Falls Sie Ihre Prozessanwendung zusätzlich zu anderen Versionen implementieren, müssen Sie die Endpunkt-URL manuell so umbenennen, dass sie bei den Versionen der Prozessanwendung eindeutig ist. Um die Endpunktzieladresse nach der Implementierung zu ändern, können Sie die Administrationskonsole verwenden.
- Falls Sie lediglich eine einzige Version der Prozessanwendung implementieren, können Sie diese Nachricht ignorieren.

JMS und generisches JMS

Systemgenerierte JMS-Bindungen und generische JMS-Bindungen sind automatisch versionssensitiv.

Anmerkung: Bei benutzerdefinierten JMS-Bindungen und generischen JMS-Bindungen findet während der Implementierung keine automatische Umbenennung zur Gewährleistung der Versionssensitivität statt.

Bei einer benutzerdefinierten Bindung müssen Sie die folgenden Attribute so umbenennen, dass sie bei den Versionen der Prozessanwendung eindeutig sind:

- Endpunktkonfiguration
- Empfangszielwarteschlange
- Listener-Port-Name (falls definiert)

Legen Sie das entsprechende Sendeziel fest, wenn Sie den Zielmodulendpunkt ändern.

MQ/JMS und MQ

Während der Implementierung findet keine automatische Umbenennung statt, um die Versionssensitivität von MQ/JMS- oder MQ-Bindungen zu aktivieren.

Sie müssen die folgenden Attribute so umbenennen, dass sie bei den Versionen der Prozessanwendung eindeutig sind:

- Endpunktkonfiguration
- Empfangszielwarteschlange

Legen Sie das entsprechende Sendeziel fest, wenn Sie den Zielmodulendpunkt ändern.

EJB

Während der Implementierung findet keine automatische Umbenennung statt, um die Versionssensitivität von EJB-Bindungen zu aktivieren.

Sie müssen die Attribute für die JNDI-Namen so umbenennen, dass sie bei den Versionen der Prozessanwendung eindeutig sind.

Bitte beachten Sie, dass Clientanwendungen ebenfalls so aktualisiert werden müssen, dass die neuen JNDI-Namen verwendet werden.

EIS

Ein Ressourcenadapter wird bei der Implementierung automatisch in eine versionssensitive Variante umbenannt, sofern der Standardressourcenname (***modulname*App:Adapter Description**) nicht geändert wurde.

Wenn der Standardressourcenname geändert wurde, müssen die Ressourcenadapternamen unter den Versionen der Prozessanwendung eindeutig sein.

Wenn die Ressourcenadapternamen nicht eindeutig sind, wird bei der Implementierung eine entsprechende Informationsnachricht protokolliert. Sie können die Ressourcenadapter nach der Implementierung in der Administrationskonsole umbenennen.

Versionssensitive dynamische Aufrufe

Sie können Mediationsablaufkomponenten so konfigurieren, dass Nachrichten an Endpunkte weitergeleitet werden, die zur Laufzeit dynamisch bestimmt werden. Wenn Sie das Mediationsmodul erstellen, konfigurieren Sie die Endpunktsuche für die Verwendung der versionssensitiven Weiterleitung.

Wenn Sie die Notation IBM_VRM (*<version>.<release>.<modifikation*) für den Snapshot verwenden, können Sie die EAR-Datei der Prozessanwendung an WebSphere Service Registry and Repository (WSRR) exportieren. Wenn Sie das Mediationsmodul erstellen, konfigurieren Sie die Endpunktsuche für die Verwendung der versionssensitiven Weiterleitung. Wählen Sie z. B. **Endpunkt zurückgeben, der mit der neuesten kompatiblen Version von SCA-Modul-basierten Services übereinstimmt** im Feld **Übereinstimmungsrichtlinie** aus und wählen Sie **SCA** im Feld **Bindingtyp** aus.

Zukünftige Versionen der Prozessanwendung werden auf dem Server implementiert und in WSRR veröffentlicht und die Endpunktsuche des Mediationsmoduls ruft dynamisch die letzte kompatible Version des Serviceendpunkts auf.

Alternativ hierzu können Sie das Ziel im SMO-Header setzen und den Wert in der Anforderungsnachricht übertragen.

Prozessanwendungen mit Java-Modulen und -Projekten implementieren

Prozessanwendungen können angepasste Java EE-Module und Java-Projekte enthalten. Wenn Sie Anwendungsversionen zusammen implementieren, muss das angepasste Java EE-Modul für jede Version der Anwendung eindeutig sein.

Beachten Sie, dass angepasste Java EE-Module und Java-Projekte auf einem Server implementiert werden, wenn sie mit einem SCA-Modul implementiert werden, in dem eine Abhängigkeit von dem Modul oder Projekt deklariert ist. Wenn Sie beim Deklarieren der Abhängigkeit nicht die Standardoption **Mit Modul implementieren** auswählen, müssen Sie das Modul oder Projekt manuell implementieren.

Prozessanwendungen mit Geschäftsregeln und Selektoren implementieren

Falls Sie mehrere Versionen einer Prozessanwendung implementieren, die eine Business-Regel oder eine Selektorkomponente enthält, müssen Sie beachten, wie die zugehörigen Metadaten durch die Versionen verwendet werden.

Die dynamischen Metadaten für eine Business-Regel oder eine Selektorkomponente werden zur Laufzeit durch den Komponentennamen, den Zielnamespace der Komponente und den Komponententyp bestimmt. Falls zwei oder mehr Versionen einer Prozessanwendung, die eine Business-Regel oder einen Selektor enthalten, in derselben Laufzeitumgebung implementiert werden, verwenden sie dieselben Metadaten für die Regellogik (Business-Regel) oder die Weiterleitung (Selektor).

Damit die einzelnen Versionen der Business-Regel oder der Selektorkomponente der Prozessanwendung jeweils eigene dynamische Metadaten (Regellogik oder Weiterleitung) verwenden können, muss der Zielnamespace so refaktoriert werden, dass er für jede Version der Prozessanwendung eindeutig ist.

Konfigurationsobjekte

Mithilfe der AdminConfig-Befehle des WebSphere-Befehlszeilenverwaltungstools (WSAdmin) können Sie auf Datenbank- und Sicherheitseigenschaften in IBM Business Process Manager zugreifen und diese modifizieren.

Der Ausdruck *Konfigurationsobjekt* bezeichnet ein Objekt, auf das mithilfe dieser WSAdmin-Bfehle zugegriffen wird. Weitere Informationen hierzu finden Sie in Sicherheitskonfigurationseigenschaften.

Implementierungsarchitektur

Die Implementierungsarchitektur von IBM Business Process Manager besteht aus Softwareprozessen (genannt 'Server'), aus topologischen Einheiten (genannt 'Knoten' und 'Zellen') sowie aus dem Konfigurationsrepository, das zum Speichern von Konfigurationsdaten verwendet wird.

Zellen

In IBM Business Process Manager bezeichnet der Begriff *Zellen* logische Gruppierungen von einem oder mehreren Knoten in einem verteilten Netz.

Eine Zelle ist ein Konfigurationskonzept, das Administratoren die Möglichkeit bietet, Knoten einander logisch zuzuordnen. Administratoren definieren die Knoten, die eine Zelle bilden, anhand spezieller Kriterien, die in den betreffenden Organisationsumgebungen sinnvoll sind.

Die Verwaltungskonfigurationsdaten werden in XML-Dateien gespeichert. Eine Zelle behält Hauptkonfigurationsdateien für jeden Server in jedem Knoten der Zelle bei. Jeder Knoten und jeder Server besitzen

außerdem eigene lokale Konfigurationsdateien. Änderungen an einer lokalen Konfigurationsdatei für einen Knoten oder einen Server sind temporär, wenn der Server zur Zelle gehört. Während sie wirksam sind, überschreiben lokale Änderungen die Zellenkonfigurationen. Änderungen an den Konfigurationsdateien für den Hauptserver und den Hauptknoten, die auf Zellenebene vorgenommen werden, ersetzen alle auf Knotenebene vorgenommenen temporären Änderungen, wenn die Konfigurationsdokumente für die Zelle mit den Knoten synchronisiert werden. Die Synchronisation findet bei festgelegten Ereignissen statt, beispielsweise beim Starten eines Servers.

Server

Server stellen die zentralen Funktionen von IBM Business Process Manager bereit. Process Server erweitern die Funktionalität eines Anwendungsservers hinsichtlich der Handhabung von SCA-Modulen. Andere Server (Deployment Manager und Knotenagenten) werden zur Verwaltung von Prozess-Servern verwendet.

Bei einem Prozess-Server kann es sich entweder um einen *eigenständigen Server* oder einen *verwalteten Server* handeln. Ein verwalteter Server kann optional Member eines *Clusters* sein. Eine Gruppe von verwalteten Servern, Clustern mit Servern und weiterer Middleware wird als *Implementierungsumgebung* bezeichnet. In einer Implementierungsumgebung wird jeder verwaltete Server oder Cluster für eine bestimmte Funktion innerhalb der Implementierungsumgebung konfiguriert (z. B. Zielhost, Anwendungsmodulhost oder CIM-Server). Ein eigenständiger Server wird so konfiguriert, dass alle erforderlichen Funktionen bereitgestellt werden.

Server stellen die Laufzeitumgebung für SCA-Module, die von diesen Modulen verwendeten Ressourcen (Datenquellen, Aktivierungsspezifikationen und JMS-Ziele) und die von IBM gelieferten Ressourcen (Nachrichtenziele, Business Process Choreographer Container und CIM-Server) bereit).

Ein *Knotenagent* ist ein Verwaltungsagent, der einen Knoten in Ihrem System darstellt und die Server auf diesem Knoten verwaltet. Knotenagenten überwachen die Server auf einem Hostsystem und leiten Verwaltungsanforderungen an die Server weiter. Der Knotenagent wird erstellt, wenn ein Knoten in einen Deployment Manager eingebunden wird.

Ein *Deployment Manager* ist ein Verwaltungsagent, der eine zentrale Managementsicht für mehrere Server und Cluster bereitstellt.

Ein eigenständiger Server wird durch ein eigenständiges Profil definiert, ein Deployment Manager durch ein Deployment Manager-Profil. Verwaltete Server werden innerhalb eines *verwalteten Knotens* erstellt, der durch ein Profil für den verwalteten Knoten definiert wird.

Eigenständige Server:

Ein eigenständiger Server stellt die Umgebung für die Implementierung von SCA-Modulen in einem Serverprozess bereit. Dieser Serverprozess umfasst unter anderem eine Administrationskonsole, ein Implementierungsziel, die Messaging-Unterstützung, den Business Process Rules Manager und einen Common Event Infrastructure-Server.

Ein eigenständiger Server ist einfach einzurichten und verfügt über eine Schnellstartkonsole, von der aus der Server gestartet und gestoppt und die Beispielsammlung und die Verwaltungskonsole geöffnet werden können. Wenn Sie die IBM Business Process Manager-Beispiele installieren und anschließend die Beispielgalerie öffnen, wird eine Beispiellösung auf dem eigenständigen Server implementiert. Sie können die Ressourcen für dieses Beispiel in der Administrationskonsole untersuchen.

Sie können zwar eigene Lösungen auf einem eigenständigen Server implementieren, jedoch kann ein solcher Server nicht die Kapazität, Skalierbarkeit und Zuverlässigkeit bieten, die in einer Produktionsumgebung gefordert werden. Für eine Produktionsumgebung ist eine Network Deployment-Umgebung die bessere Wahl.

Sie können mit einem eigenständigen Server beginnen und diesen später in eine Network Deployment-Umgebung aufnehmen, indem Sie ihn in eine Deployment Manager-Zelle einbinden. *Dies setzt voraus, dass noch keine anderen Knoten in diese Zelle eingebunden wurden*. Es ist nicht möglich, mehrere eigenständige Server in eine Zelle einzubinden. Sie können den eigenständigen Server mit der Administrationskonsole des Deployment Managers oder mit dem Befehl **addNode** einbinden. Der eigenständige Server darf nicht aktiv sein, wenn Sie ihn mit dem Befehl **addNode** einbinden.

Der eigenständige Server wird durch ein eigenständiges Serverprofil definiert.

Cluster:

Cluster sind Gruppen von Servern, die zusammen verwaltet werden und am Workload-Management teilnehmen.

Ein Cluster kann Knoten oder einzelne Anwendungsserver enthalten. Ein Knoten ist in der Regel ein physischer Computer mit einer eindeutigen IP-Hostadresse, auf dem einer oder mehrere Anwendungsserver ausgeführt werden. Cluster können unter der Konfiguration einer Zelle, die viele Server und Cluster mit unterschiedlichen Konfigurationen und Anwendungen einander logisch zuordnet, nach Ermessen des Administrators und in einer für die Organisationsumgebungen sinnvollen Weise gruppiert werden.

Cluster sorgen für eine gleichmäßige Lastverteilung unter Servern. Server, die Teil eines Clusters sind, werden Cluster-Member genannt. Wenn Sie eine Anwendung in einem Cluster installieren, wird sie automatisch auf jedem Cluster-Member installiert.

Da jeder Cluster-Member dieselben Anwendungen enthält, können Sie Client-Tasks gemäß der Kapazität der unterschiedlichen Systeme verteilen, indem Sie jedem Server eine Gewichtung zuweisen.

Das Leistungsverhalten und der Ausweichbetrieb (Failover) werden durch die Zuweisung von Gewichtungen zu Servern in einem Cluster verbessert. Tasks werden Servern zugewiesen, die die Kapazität für die Ausführung der Taskoperationen besitzen. Falls einer der Server nicht in der Lage ist, die Task auszuführen, wird sie einem anderen Cluster-Member zugewiesen. Diese Funktionalität der erneuten Zuweisung bietet offensichtliche Vorteile gegenüber der Ausführung eines einzigen Anwendungsservers, der bei zu vielen Anforderungen überlastet sein kann.

Profil

Ein Profil definiert eine eigene Laufzeitumgebung mit separaten Befehls-, Konfigurations- und Protokolldateien. Profile definieren drei verschiedene Umgebungstypen auf IBM Business Process Manager-Systemen: eigenständiger Server, Deployment Manager und verwalteter Knoten.

Mit Profilen können Sie mehrere Laufzeitumgebungen auf einem System ausführen, ohne dazu mehrere Kopien der Binärdateien von IBM Business Process Manager installieren zu müssen.

Verwenden Sie zum Erstellen von IBM BPM-Profilen das Befehlszeilendienstprogramm BPMConfig. Das Befehlszeilendienstprogramm **manageprofiles** bzw. seine grafische Benutzerschnittstelle, das Profile Management Tool (PMT), lässt sich als alternative Methode zum Erstellen von Profilen für den Deployment Manager oder verwaltete Knoten verwenden. PMT wird für die Erstellung von eigenständigen Profilen nicht mehr unterstützt.

Anmerkung: Auf verteilten Plattformen besitzt jedes Profil einen eindeutigen Namen. Unter z/OS haben alle Profile den Namen „default"; es ist nicht möglich, unter z/OS Profile umzubenennen, zu bearbeiten, zu kopieren oder zu löschen.

Profiltypen

Folgende IBM BPM-Profiltypen sind mit IBM Business Process Manager V8.5 verfügbar:

IBM BPM - Eigenständiges Profil
Mit dem eigenständigen Profil werden eigenständige Server mit dem Leistungsspektrum und der Funktionalität definiert, die für IBM BPM Express-Konfigurationen charakteristisch sind. Sie können das eigenständige Profil mithilfe der Profilschablone `BPM/BpmServer` erstellen, die nur in IBM BPM Express unterstützt wird.

IBM BPM - Deployment Manager-Profil
Das Deployment Manager-Profil definiert einen Deployment Manager, der genau eine Verwaltungsschnittstelle für eine logische Gruppe von Servern auf einer oder mehreren Workstations bereitstellt. Sie können das Deployment Manager-Profil mithilfe der Profilschablone `BPM/BpmDmgr` erstellen, die nur in IBM BPM Standard und IBM BPM Advanced unterstützt wird.

IBM BPM - Profil für verwalteten Knoten
Das Profil für den verwalteten Knoten definiert einen verwalteten Knoten, sofern dieser Knoten in einen Deployment Manager eingebunden ist. Sie können das Profil für den verwalteten Knoten mithilfe der Profilschablone `BPM/BpmNode` erstellen, die nur in IBM BPM Standard und IBM BPM Advanced unterstützt wird.

Mit jeder IBM BPM-Konfiguration sind bestimmte Profiltypen verfügbar.

Tabelle 19. Verfügbare IBM BPM-Profiltypen

	Profiltypen		
IBM BPM Konfiguration	**Eigenständig**	**Deployment Manager**	**Verwalteter Knoten**
IBM BPMExpress	Ja	Nein	Nein
IBM BPM Standard	Nein	Ja	Ja
IBM BPM Advanced	Nein	Ja	Ja
Komponententestumgebung (Unit Test Environment, UTE) für IBM Integration Designer	Ja	Optional	Optional

Profilverzeichnis

Jedes Profil auf einem System besitzt ein eigenes Verzeichnis mit allen zugehörigen Dateien. Sie können die Position des Profilverzeichnisses bei der Erstellung des Profils festlegen. Standardmäßig wird das Verzeichnis `profiles` in dem Verzeichnis verwendet, in dem IBM Business Process Manager installiert ist. Beispiel: Das Profil Dmgr befindet sich im Verzeichnis `C:\Program Files\IBM\WebSphere\ProcServer\profiles\Dmgr`.

Standardprofil

Das erste Profil, das Sie in einer Installation von IBM Business Process Manager erstellen, ist das *Standardprofil*. Das Standardprofil ist das Standardziel für Befehle, die im Unterverzeichnis `bin` des Installationsverzeichnisses von IBM Business Process Manager eingegeben werden. Ist auf einem System nur ein Profil vorhanden, dann wird jeder Befehl für dieses Profil ausgeführt. Wenn Sie ein weiteres Profil erstellen, können Sie dieses zum Standardprofil machen.

Anmerkung: Das Standardprofil muss nicht zwangsläufig den Namen 'default' haben.

Profile erweitern

Wenn Sie bereits ein Deployment Manager-Profil, ein Profil für den verwalteten Knoten oder ein eigenständiges Serverprofil besitzen, das für WebSphere Application Server Network Deployment erstellt wurde, können Sie es *erweitern*, so dass es zusätzlich zu seiner bestehenden Funktion auch IBM Business Process Manager unterstützt. Wenn Sie ein Profil erweitern möchten, installieren Sie zunächst IBM Business Process Manager. Verwenden Sie anschließend das Befehlszeilendienstprogramm **`manageprofiles`**, um ein

eigenständiges Profil zu erweitern, oder benutzen Sie wahlweise das Profile Management Tool oder das Befehlszeilendienstprogramm **manageprofiles**, um ein Profil für Deployment Manager-Profil oder einen erweiterten Knoten zu erweitern.

Wichtig: In einer Netzimplementierungsumgebung müssen Sie zunächst das Deployment Manager-Profil und anschließend die Profile von verwalteten Knoten erweitern.

Einschränkung: Sie können kein eigenständiges oder Deployment Manager-Profil erweitern, in dem die Standard-WebSphere VMM-Benutzerregistry geändert wurde, z.B. für die Verwendung von LDAP.

Deployment Manager

Ein Deployment Manager ist ein Server, der die Operationen für eine logische Gruppe anderer Server (Zelle) steuert. Der Deployment Manager ist ein zentraler Ort für die Verwaltung von Servern und Clustern.

Bei der Erstellung einer Implementierungsumgebung ist das Deployment Manager-Profil das erste Profil, das Sie erstellen. Jede Implementierungsumgebung, die Sie erstellen, verfügt über eine Schnellstartkonsole, von der aus der Deployment Manager gestartet und gestoppt und seine Administrationskonsole eingerichtet werden kann. Mit der Administrationskonsole des Deployment Managers können Sie die Server und Cluster in der Zelle steuern und verwalten. Sie können Server und Cluster konfigurieren, Server zu Clustern hinzufügen, Server und Cluster starten und stoppen und SCA-Module implementieren.

Obwohl es sich beim Deployment Manager prinzipiell um einen Server handelt, können Sie auf diesem keine Module implementieren.

Knoten

Ein *Knoten* ist eine logische Gruppierung von verwalteten Servern.

Ein Knoten entspricht in der Regel einem logischen oder physischen Computersystem mit einer eindeutigen IP-Hostadresse. Knoten können nicht mehrere Computer umfassen. Knotennamen sind gewöhnlich mit dem Hostnamen für den Computer identisch.

Knoten in einer Network Deployment-Topologie können verwaltet oder nicht verwaltet sein. Ein verwalteter Knoten hat einen Knotenagentenprozess, der die Konfiguration und die Server des Knotens verwaltet. Nicht verwaltete Knoten haben keinen Knotenagenten.

Verwaltete Knoten:

Ein *verwalteter Knoten* ist ein Knoten, der in einen Deployment Manager eingebunden wurde, einen Knotenagenten enthält und verwaltete Server enthalten kann. Auf einem verwalteten Knoten können Sie verwaltete Server konfigurieren und ausführen.

Die Server, die in einem verwalteten Knoten konfiguriert sind, bilden die Ressourcen Ihrer Implementierungsumgebung. Diese Server werden in der Administrationskonsole des Deployment Managers erstellt, konfiguriert, gestartet, gestoppt, verwaltet und gelöscht.

Ein verwalteter Knoten hat einen Knotenagenten, der alle Server auf einem Knoten verwaltet.

Beim Einbinden eines Knotens wird automatisch ein Knotenagentenprozess erstellt. Dieser Knotenagent muss aktiv sein, um die Konfiguration des Profils verwalten zu können. Dazu zählen unter anderem die folgenden Tasks:

- Serverprozesse starten und stoppen
- Konfigurationsdaten auf dem Deployment Manager mit der Kopie auf dem Knoten synchronisieren

Der Knotenagent muss jedoch nicht aktiv sein, um Anwendungen auszuführen oder um Ressourcen auf dem Knoten zu konfigurieren.

Ein verwalteter Knoten kann einen oder mehrere Server enthalten, die von einem Deployment Manager verwaltet werden. Sie können auf den Servern in einem verwalteten Knoten Lösungen implementieren. Der verwaltete Knoten enthält jedoch keine Sammlung von Beispielanwendungen. Der verwaltete Knoten wird durch ein Profil für den verwalteten Knoten definiert und verfügt über eine Schnellstartkonsole.

Nicht verwaltete Knoten:

Ein nicht verwalteter Knoten besitzt keinen Knotenagenten für die Verwaltung seiner Server.

Nicht verwaltete Knoten in der Network Deployment-Topologie können Serverdefinitionen wie zum Beispiel Webserver, aber keine Anwendungsserverdefinitionen besitzen. Nicht verwaltete Knoten können nicht in einem Verbund föderiert werden. Dies bedeutet, dass ein Knotenagent in keinem Fall zu einem nicht verwalteten Knoten hinzugefügt werden kann. Ein eigenständiger Server ist ein anderer Typ für einen nicht verwalteten Knoten. Der Deployment Manager kann diesen eigenständigen Server nicht verwalten, weil dieser der Zelle nicht bekannt ist. Ein eigenständiger Server kann in einem Verbund föderiert werden. Wenn er föderiert ist, wird automatisch ein Knotenagent erstellt. Der Knoten wird zu einem verwalteten Knoten in der Zelle.

Knotenagenten

Knotenagenten sind Verwaltungsagenten, die Verwaltungsanforderungen an Server weiterleiten.

Ein Knotenagent ist ein Server, der auf jedem Host-Computer-System ausgeführt wird, das Teil der Network Deployment-Konfiguration ist. Es handelt sich hierbei um einen reinen Verwaltungsagenten, der nicht in Anwendungsservingfunktionen einbezogen wird. Ein Knotenagent dient auch als Host für andere wichtige Verwaltungsfunktionen wie Dateiübertragungsservices, Konfigurationssynchronisation und Leistungsüberwachung.

Hinweise zur Benennung von Profilen, Knoten, Servern, Hosts und Zellen

Dieser Abschnitt enthält Informationen zu reservierten Begriffen sowie Hinweise, die Sie bei der Benennung von Profilen, Knoten, Servern, Hosts und Zellen (sofern zutreffend) berücksichtigen müssen. Dieser Abschnitt gilt für die verteilten Plattformen.

Hinweise zur Benennung von Profilen

Als Profilname kann mit folgenden Einschränkungen ein beliebiger eindeutiger Name verwendet werden. Verwenden Sie für Profilnamen keines der folgenden Zeichen:

- Leerzeichen
- Sonderzeichen, die im Namen von Verzeichnissen auf Ihrem Betriebssystem nicht zulässig sind. Beispiele: *, & oder ?
- Schrägstriche (/) oder umgekehrte Schrägstriche (\)

Doppelbytezeichen sind zulässig.

Hinweise zu Verzeichnispfaden: Der Pfad für das Installationsverzeichnis darf höchstens 60 Zeichen lang sein. Die Anzahl von Zeichen im Verzeichnispfad *profilverzeichnispfad**profilname* darf höchstens 80 Zeichen betragen.

Anmerkung: Verwenden Sie eine Namenskonvention mit kurzen Pfadnamen, wenn Sie ein Profil in einer Windows-Umgebung erstellen, um die Windows-Pfadlängenbegrenzung auf 255 Zeichen zu vermeiden.

Hinweise zur Benennung von Knoten, Servern, Hosts und Zellen

Reservierte Namen: Vermeiden Sie reservierte Namen als Feldwerte. Die Verwendung reservierter Namen kann zu unvorhersehbaren Ergebnissen führen. Die folgenden Wörter sind reserviert:

- cells

- nodes
- servers
- clusters
- applications
- deployments

Beschreibung der Felder auf der Seite mit den Namen für Knoten und Host und der Seite mit den Namen für Knoten, Host und Zelle: Beachten Sie beim Erstellen der Profile die entsprechenden Namensregeln.

- Eigenständige Serverprofile
- Deployment Manager-Profile
- Profile für verwaltete Knoten

Tabelle 20. Namensregeln für eigenständige Serverprofile

Feldname	Standardwert	Einschränkung	Beschreibung
Knotenname	. . . *kurzname_des_hosts* Node *knotennummer*, wobei Folgendes gilt: • *kurzname_des_hosts* ist der Kurzname des Hosts. • *knotennummer* ist eine fortlaufende Zahl, die bei 01 beginnt.	Verwenden Sie keine reservierten Namen.	Wählen Sie einen beliebigen Namen aus. Zur besseren Organisation Ihrer Installation sollten Sie einen eindeutigen Namen verwenden, falls Sie mehr als einen Server auf dem System installieren möchten.
Servername	. . . server1	Verwenden Sie einen eindeutigen Namen für den Server.	Der logische Name für den Server.
Hostname	. . . Die Langform des DNS-Namens (DNS = Domain Name Server, Domänennamensserver).	Verwenden Sie einen vollständig qualifizierten, über Ihr Netz erreichbaren Hostnamen.	Verwenden Sie den echten DNS-Namen oder die IP-Adresse Ihrer Workstation, um die Kommunikation mit dieser Workstation zu ermöglichen. Weitere Informationen zum Hostnamen finden Sie im Anschluss an diese Tabelle.

Tabelle 21. Namensregeln für Deployment Manager-Profile

Feldname	Standardwert	Einschränkung	Beschreibung
Knotenname	. . . *kurzname_des_hosts* Cell Manager*knotennummer*, wobei Folgendes gilt: • *kurzname_des_hosts* ist der Kurzname des Hosts. • *knotennummer* ist eine fortlaufende Zahl, die bei 01 beginnt.	Verwenden Sie für den Deployment Manager einen eindeutigen Namen. Verwenden Sie keine reservierten Namen.	Der Name wird für die Verwaltung in der Deployment Manager-Zelle verwendet.
Hostname	. . . Die Langform des DNS-Namens (DNS = Domain Name Server, Domänennamensserver).	Verwenden Sie einen vollständig qualifizierten, über Ihr Netz erreichbaren Hostnamen. Verwenden Sie keine reservierten Namen.	Verwenden Sie den echten DNS-Namen oder die IP-Adresse Ihrer Workstation, um die Kommunikation mit dieser Workstation zu ermöglichen. Weitere Informationen zum Hostnamen finden Sie im Anschluss an diese Tabelle.

Tabelle 21. Namensregeln für Deployment Manager-Profile (Forts.)

Feldname	Standardwert	Einschränkung	Beschreibung
Zellenname	. . . *kurzname_des_hosts* Cell *zellennummer*, wobei Folgendes gilt: • *kurzname_des_hosts* ist der Kurzname des Hosts. • *zellennummer* ist eine fortlaufende Zahl, die bei 01 beginnt.	Verwenden Sie einen eindeutigen Namen für die Deployment Manager-Zelle. Zellennamen müssen generell immer eindeutig sein, wenn das Produkt auf der gleichen physischen Workstation oder in einem Workstation-Cluster (wie z. B. einem Sysplex) ausgeführt wird. Zusätzlich muss ein Zellenname in allen Situationen eindeutig sein, in denen die Netzkonnektivität zwischen Entitäten entweder zwischen den Zellen oder von einem Client erforderlich ist, der mit jeder der Zellen kommunizieren muss. Zellennamen müssen auch eindeutig sein, wenn deren Namensbereiche in einen Verbund eingebunden werden sollen. Andernfalls können Symptome wie Ausnahmebedingungen vom Typ javax.naming.NameNotFoundException auftreten, die das Erstellen von eindeutig benannten Zellen erforderlich machen.	Alle eingebundenen Knoten werden Elemente der Deployment Manager-Zelle, die Sie auf der Seite für die Knoten-, Host- und Zellennamen im Profile Management Tool angeben.

Tabelle 22. Namensregeln für Profile von verwalteten Knoten

Feldname	Standardwert	Einschränkung	Beschreibung
Knotenname	. . . *kurzname_des_hosts* Node *knotennummer*, wobei Folgendes gilt: • *kurzname_des_hosts* ist der Kurzname des Hosts. • *knotennummer* ist eine fortlaufende Zahl, die bei 01 beginnt.	Verwenden Sie keine reservierten Namen. Verwenden Sie in der Deployment Manager-Zelle einen eindeutigen Namen.	Der Name wird für die Verwaltung innerhalb der Deployment Manager-Zelle verwendet, zu der das Profil für den verwalteten Knoten hinzugefügt wird. Verwenden Sie in der Deployment Manager-Zelle einen eindeutigen Namen.
Hostname	. . . Die Langform des DNS-Namens (DNS = Domain Name Server, Domänennamensserver).	Verwenden Sie einen vollständig qualifizierten, über Ihr Netz erreichbaren Hostnamen.	Verwenden Sie den echten DNS-Namen oder die IP-Adresse Ihrer Workstation, um die Kommunikation mit dieser Workstation zu ermöglichen. Weitere Informationen zum Hostnamen finden Sie im Anschluss an diese Tabelle.

Hinweise zu Hostnamen:

Der Hostname ist der Netzname für die physische Workstation, auf der der Knoten installiert ist. Der Hostname muss auf dem Server in einen physischen Netzknoten aufgelöst werden. Bei einem Server mit mehreren Netzkarten muss der Hostname oder die IP-Adresse in eine der Netzkarten aufgelöst werden. Ferne Knoten verwenden den Hostnamen, um mit diesem Knoten zu kommunizieren.

IBM Business Process Manager ist sowohl mit dem Internetprotokoll der Version 4 (IPv4) als auch mit Version 6 (IPv6) kompatibel. Die Eingabe von IP-Adressen in der Administrationskonsole oder an anderen Stellen kann wahlweise in einem der beiden Formate erfolgen. Beachten Sie, dass die Eingabe von IP-Adressen im IPv6-Format erfolgen muss, wenn IPv6 auf Ihrem System bereits implementiert ist. Wenn IPv6 auf Ihrem System noch nicht verfügbar ist, müssen Sie IP-Adressen im IPv4-Format eingeben. Weitere Informationen zu IPv6 enthält die folgende Beschreibung: IPv6.

Die folgenden Richtlinien können helfen, den geeigneten Hostnamen für Ihre Workstation festzulegen:

- Wählen Sie einen Host aus, den andere Workstations in Ihrem Netz erreichen können.
- Verwenden Sie als Wert nicht die generische ID 'localhost'.
- Versuchen Sie nicht, IBM Business Process Manager-Produkte auf einem Server mit einem Host zu installieren, in dessen Namen Doppelbytezeichen verwendet werden. Doppelbytezeichen werden in dem Hostnamen nicht unterstützt.
- Verwenden Sie in Servernamen keine Unterstreichungszeichen (_). Internetstandards geben vor, dass die Domänennamen mit den Anforderungen an Hostnamen konform sein müssen, die in den Internet Official Protocol Standards RFC 952 und RFC 1123 beschrieben werden. Domänennamen dürfen nur Buchstaben (in Groß- oder Kleinschreibung) sowie Ziffern enthalten. Domänennamen dürfen auch Gedankenstriche (-) enthalten, solange diese nicht am Ende des Namens stehen. Unterstreichungszeichen (_) werden im Hostnamen nicht unterstützt. Wenn Sie IBM Business Process Manager auf einem Server installiert haben, in dessen Namen ein Unterstreichungszeichen vorkommt, können Sie auf diesen Server so lange mit der entsprechenden IP-Adresse zugreifen, bis Sie ihn umbenennen.

Wenn Sie koexistierende Knoten auf demselben Computer mit eindeutigen IP-Adressen definieren, dann definieren Sie jede IP-Adresse in einer DNS-Referenztabelle (DNS = Domänennamensserver). Konfigurationsdateien für Server stellen keine DN-Auflösung für mehrere IP-Adressen auf einer Workstation mit nur einer Netzadresse bereit.

Der Wert, den Sie für den Hostnamen angeben, wird in Konfigurationsdokumenten als Wert für das Merkmal 'hostName' verwendet. Geben Sie den Wert für den Hostnamen in einem der folgenden Formate an:

- Zeichenfolge für einen vollständig qualifizierten DNS-Hostnamen (DNS = Domänennamensserver), wie zum Beispiel xmachine.manhattan.ibm.com
- Zeichenfolge für den DNS-Hostnamen in seiner Standardkurzform, wie zum Beispiel xmachine
- Numerische IP-Adresse, wie zum Beispiel 127.1.255.3

Der vollständig qualifizierte DNS-Hostname hat den Vorteil, völlig eindeutig und trotzdem flexibel zu sein. Sie haben die Möglichkeit, die tatsächliche IP-Adresse für das Hostsystem zu ändern, ohne dabei die Konfiguration des Servers ändern zu müssen. Dieser Wert für den Hostnamen ist besonders dann nützlich, wenn Sie die IP-Adresse mithilfe des Dynamic Host Configuration Protocol (DHCP) häufig ändern möchten. Ein Nachteil dieses Formats besteht in seiner Abhängigkeit vom DNS. Ohne DNS ist die Konnektivität beeinträchtigt.

Der Kurzname für den Host ist dynamisch auflösbar. Ein Kurznamensformat bietet die zusätzliche Möglichkeit einer Definitionsänderung in der Datei für die lokalen Hosts, sodass das System auch dann mit dem Server arbeiten kann, wenn keine Verbindung mehr zum Netz besteht. Definieren Sie in der Datei für die Hosts den Wert '127.0.0.1' (lokales Loopback) für den Kurznamen, um die Ausführung bei getrennter Verbindung anzugeben. Ein Nachteil des Kurznamensformats besteht darin, dass für den Remotezugriff ein DNS erforderlich ist. Ohne DNS ist die Konnektivität beeinträchtigt.

Eine numerische IP-Adresse hat den Vorteil, dass keine Namensauflösung über DNS erforderlich ist. Ein ferner Knoten kann mit dem Knoten, den Sie mit einer numerischen IP-Adresse bezeichnen, auch dann verbunden werden, wenn kein DNS verfügbar ist. Ein Nachteil dieses Formats besteht darin, dass die numerische IP-Adresse festgelegt ist. Wenn Sie die IP-Adresse der Workstation ändern, müssen Sie auch die Einstellung für das Merkmal 'hostName' in den Konfigurationsdokumenten ändern. Verwenden Sie deshalb nicht die numerische IP-Adresse, wenn Sie DHCP verwenden oder IP-Adressen regelmäßig ändern. Ein weiterer Nachteil dieses Formats besteht darin, dass Sie den Knoten nicht verwenden können, wenn keine Verbindung zwischen Host und Netz besteht.

BPMN 2.0

IBM Business Process Manager-Geschäftsprozessdefinitionen unterstützen die Unterklasse 'Common Executable' der Konformitätsklasse für die BPMN 2.0-Prozessmodellierung, die sich mit ausführbaren Modellen beschäftigt.

BPMN (Business Process Model and Notation) ist der grundlegende Standard für die Prozesse in IBM Process Designer und IBM Process Center. BPD-Diagramme (BPD - Business Process Definition, Geschäftsprozessdefinition) basieren auf der BPMN-Spezifikation. Dieser Abschnitt enthält einige Beispiele für die Verwendung von BPMN 2.0 in IBM Business Process Manager. Detaillierte Informationen zu BPMN finden Sie auf der Seite zur BPMN-Spezifikation unter http://www.bpmn.org/.

IBM Business Process Manager unterstützt die folgenden BPMN 2.0-Tasktypen:

- Keine (abstrakte Task in der BPMN 2.0-Spezifikation)
- Systemtask (Service-Task in der BPMN 2.0-Spezifikation)
- Benutzertask
- Script
- Entscheidungstask (Geschäftsregeltask in der BPMN 2.0-Spezifikation)

IBM BPM Nachrichten-Zwischenereignisse bieten ähnliche Funktionen wie die BPMN-Sende- und Empfangstask.

BPMN 2.0-Notation

Ab Version 7.5.1 werden Process Designer BPMN 2.0-Tasksymbole in den BPD-Diagrammen in einer vereinfachten Palette erfasst und in Prozessdiagrammen angezeigt. Anhand der Symbole können Sie feststellen, ob es sich bei der jeweiligen Aktivität um eine Systemtask, eine Benutzertask, eine Entscheidungstask, ein Script oder einen verlinkten Prozess handelt. Aktivitäten in Modellen, die in früheren Versionen erstellt wurden, verfügen ebenfalls über entsprechende BPMN 2.0-Tasktypen und -Tasksymbole, wenn sie in Version 7.5.1 oder höher angezeigt werden.

Aktivitäten und Tasks

Gegenüber früheren Versionen von Process Designer wurden einige Terminologieänderungen vorgenommen. Einige dieser Änderungen beziehen sich auf Aktivitätstypen, die umbenannt wurden.

- Aktivitäten vom Typ 'Service (automatisiert)' sind nun Systemtasks.
- Aktivitäten vom Typ 'Service (Task)' in einem systemfremden Verantwortlichkeitsbereich sind nun Benutzertasks.
- Aktivitäten vom Typ 'Service (Task)' in einem Verantwortlichkeitsbereich des Systems sind nun Entscheidungstasks, wenn sie auf einen Entscheidungsservice verweisen.
- Aktivitäten vom Typ 'Service (Task)' in einem Verantwortlichkeitsbereich des Systems sind nun Systemtasks, wenn sie auf einen anderen Service als den Entscheidungsservice verweisen.
- Javascript-Aktivitäten sind nun Script-Tasks.
- Aktivitäten in verschachtelten Prozessen sind nun verlinkte Prozesse.

- Externe Aktivitäten aus früheren Versionen von Process Designer stehen als externe Implementierungen für Benutzer- und Systemtasks zur Verfügung.

Gateways

Für die Gateways früherer Versionen gibt es keine Notationsänderungen. Es gibt jedoch drei Terminologieänderungen. Das Entscheidungsgateway ist nun ein *exklusives Gateway*, das einfache Split- oder Join-Gateway ist nun ein *paralleles Gateway* und das bedingte Split- oder Join-Gateway ist nun ein *inklusives Gateway*.

Es gibt auch einen neuen Gateway-Typ, das *Ereignis-Gateway*. Ein Ereignis-Gateway stellt einen Verzweigungspunkt in einem Prozess dar, bei dem die alternativen Pfade, die dem Gateway folgen, auf stattfindenden Ereignissen und nicht auf der Auswertung von Ausdrücken basieren, die Prozessdaten verwenden (wie bei einem exklusiven oder inklusiven Gateway). Ein bestimmtes Ereignis (in der Regel der Empfang einer Nachricht) legt fest, welcher Pfad verwendet wird.

Nicht unterbrechende Ereignisse

Mit BPMN 2.0 wird eine Notation für nicht unterbrechende Ereignisse hinzufügt. Im Normalfall unterbricht ein Grenzereignis die Aktivität, der es zugeordnet ist. Beim Auslösen des Ereignisses wird die Aktivität gestoppt und das Token wird an den abgehenden Sequenzfluss des Ereignisses weitergeleitet. Wenn das Ereignis als nicht unterbrechendes Ereignis definiert ist, wird die zugeordnete Aktivität beim Auslösen des Ereignisses weiterhin parallel ausgeführt, und es wird ein neues Token generiert, das mit dem abgehenden Sequenzfluss des Ereignisses weitergeleitet wird. Die Ereignisgrenze wird in eine gestrichelte Linie für nicht unterbrechende Ereignisse geändert.

Zwischenereignisse, die Aktivitäten zugeordnet sind, sind unterbrechende Zwischenereignisse, wenn sie die ihnen zugeordnete Aktivität schließen, oder nicht unterbrechende Zwischenereignisse, wenn sie die ihnen zugeordnete Aktivität nicht schließen.

Startereignis

Laut BPMN-Spezifikation müssen Symbole für Start- und Endereignisse in Prozessmodellen nicht verwendet werden. Process Designer setzt die Verwendung von Start- und Stoppereignissen in Prozessmodellen voraus.

In Process Designer stehen die folgenden Typen von Startereignissen zur Verfügung:

Prozesse

- Nicht
- Nachricht
- Ad-hoc

Unterprozesse

- Nicht

Ereignisunterprozesse

- Fehler
- Nachricht
- Zeitgeber

Der Typ eines Startereignisses kann geändert werden, indem die Eigenschaften für das Ereignis bearbeitet werden. Sie können in einem Prozess zahlreiche Nachrichtenstartereignisse, aber nur ein einziges Nicht-Startereignis verwenden.

Endereignisse

Es stehen vier Typen von Endereignissen zur Verfügung: *Nachricht*, *Beendigung*, *Fehler* und *Nicht*. Der Typ eines Endereignisses kann geändert werden.

Wenn ein übergeordneter Prozess einen untergeordneten Prozess aufruft und der untergeordnete Prozess eine Beendigungsereignisaktion ausführt, muss der untergeordnete Prozess gestoppt werden, während der übergeordnete Prozess mit den nächsten Schritten fortfährt.

Unterprozesse

Die BPMN-Spezifikation definiert zwei Typen von Unterprozessen, nämlich eingebettete und wiederverwendbare Unterprozesse. In Process Designer können beide Typen erstellt werden. Eingebettete Unterprozesse werden in Process Designer einfach als *Unterprozesse* bezeichnet und sind in Version 7.5.1 neu. Ein wiederverwendbarer Unterprozess in BPMN ist ein *verlinkter Prozess* in Process Designer.

Unterprozesse sind in einem übergeordneten Prozess enthalten und stellen eine Möglichkeit zur Gruppierung von Prozessschritten dar, um Diagramme weniger komplex und unübersichtlich zu gestalten. Unterprozesse komprimieren mehrere Schritte zu einer Aktivität. Der Unterprozess ist nur für den Prozess sichtbar, in dem er definiert ist. Ein Unterprozess existiert innerhalb des Geltungsbereichs des jeweiligen aufrufenden Programms und hat Zugriff auf alle Variablen in dieser Umgebung. Von einem eingebetteten Unterprozess werden keine Parameter übergeben oder empfangen.

Neben dem Unterprozess und dem verlinkten Prozess verfügt Process Designer über einen Ereignisunterprozess, bei dem es sich um einen speziellen Unterprozess für die ereignisgesteuerte Verarbeitung handelt. Er ist nicht mit anderen Aktivitäten über den Sequenzfluss verbunden und tritt nur dann auf, wenn das zugehörige Startereignis ausgelöst wird.

Verlinkte Prozesse

Ein wiederverwendbarer Unterprozess in BPMN wird in Process Designer als *verlinkter Prozess* bezeichnet. Es handelt sich um einen Prozess, der außerhalb des aktuellen Prozesses erstellt wird und vom aktuellen Prozess aufgerufen werden kann. Er ist wiederverwendbar, weil andere Prozessdefinitionen diesen Prozess ebenfalls aufrufen können. Der verlinkte Prozess definiert die zugehörigen Eingabe- und Ausgabeparameter und verfügt über keinen Zugriff auf den Geltungsbereich oder die Umgebung des aufrufenden Programms. Der verlinkte Prozess ähnelt dem verschachtelten Prozess, der in früheren Versionen zur Verfügung stand; das Verhalten der Aktivität ist unverändert. Vorherige verschachtelte Prozesse werden in verlinkte Prozesse migriert. Der verlinkte Prozess sieht wie ein Unterprozess mit einer dicken Grenze aus und wird im Inspector-Fenster hervorgehoben angezeigt.

Schleifen

BPMN enthält das Konzept sich wiederholender Aktivitäten. Eine Aktivität kann atomar sein, was bedeutet, dass die Aktivität wiederholt wird. Eine Aktivität kann aber auch ein Unterprozess sein, in den eine Reihe von Schritten eingebunden sind, die wiederholt werden. Wenn Sie die sich wiederholende Aktivität erweitern, werden die darin enthaltenen Aktivitäten angezeigt, die wiederholt ausgeführt werden sollen. Die Bedingung wird immer zu Beginn einer Schleifenwiederholung ausgewertet. Eine Auswertung am Ende einer Schleifenwiederholung ist nicht möglich.

IBM Business Process Manager verfügt über eine *Schleife mit mehreren Instanzen*, die mit begrenzter Häufigkeit ausgeführt wird, wobei die Ausführung der darin enthaltenen Aktivitäten sequenziell oder parallel erfolgt.

Nicht-BPMN-Prozesse importieren

In IBM WebSphere Business Modeler erstellte Modelle können importiert und in Process Designer verwendet werden. Informationen zum BPMN 2.0-Import finden Sie unter Zuordnung von IBM WebSphere Business Modeler-Elementen zu IBM Business Process Manager-Konstrukten. In IBM WebSphere Business Compass, Rational Software Architect oder anderen Modellierungsumgebungen erstellte BPMN 2.0-Modelle können ebenfalls importiert werden.

Geschäftsprozessdefinitionen (BPDs)

Um einen Prozess in IBM Process Designer zu modellieren, müssen Sie eine Geschäftsprozessdefinition (Business Process Definition, BPD) erstellen. Die Geschäftsprozessdefinition kann auf einem importierten BPMN-Modell basieren.

Bei einer Geschäftsprozessdefinition handelt es sich um ein wiederverwendbares Prozessmodell, mit dessen Hilfe alle Gemeinsamkeiten von Laufzeitinstanzen dieses Prozessmodells definiert werden. Eine Geschäftsprozessdefinition muss ein Startereignis, ein Endereignis, mindestens eine Bahn und mindestens eine Aktivität enthalten. Detaillierte Informationen zu den Einschränkungen bezüglich der verwendbaren Zeichen bei Geschäftsprozessdefinitionen finden Sie in den zugehörigen Links unter 'IBM Process Designer - Namenskonventionen'.

Eine Geschäftsprozessdefinition muss eine Bahn für alle Systeme und Benutzergruppen enthalten, die an einem Prozess teilnehmen. Eine Bahn kann eine Teilnehmer- oder eine Systembahn sein. Sie haben allerdings die Möglichkeit, Geschäftsprozessdefinitionen zu erstellen, die die Aktivitäten einer Gruppe und eines Systems in einer einzigen Spur gruppieren. Informationen zur Erstellung von Geschäftsprozessdefinitionen finden Sie in den zugehörigen Links unter "Erstellen einer Geschäftsprozessdefinition (BPD)".

Sie können angeben, dass eine bestimmte Person oder Gruppe für die Aktivitäten in einer Teilnehmerbahn verantwortlich sind. Jede Bahn, die Sie erstellen, wird standardmäßig der Gruppe "Alle Benutzer" zugewiesen. Sie können diese Standardgruppe verwenden, um Ihre BPD im Inspector auszuführen und zu testen. Die Gruppe "Alle Benutzer" umfasst alle Benutzer, die Mitglied der Sicherheitsgruppe **tw_allusers** sind. Diese Gruppe ist eine spezielle Sicherheitsgruppe, die automatisch alle Benutzer im System einschließt.

Eine Systembahn enthält Aktivitäten, die von einem bestimmten IBM Process Center-System verarbeitet werden. Jede Aktivität benötigt eine Implementierung, die die Aktivität definiert und die Eigenschaften für die Task festlegt. Während der Implementierung erstellt der Entwickler einen Service oder schreibt das zur Ausführung der Aktivitäten in einer Systembahn erforderliche JavaScript. Informationen zu Services finden Sie unter "Informationen zu Servicetypen" in den zugehörigen Links.

Für jede erstellte Geschäftsprozessdefinition müssen Sie Variablen deklarieren, um die Geschäftsdaten, die in Ihrem Prozess von einer Aktivität zur nächsten übergeben werden, zu erfassen. Weitere Informationen zur Implementierung von Variablen finden Sie unter "Variablen verwalten und zuordnen" in den zugehörigen Links.

Sie können auch Ereignisse zu einer Geschäftsprozessdefinition hinzufügen. Ereignisse in IBM BPM können können durch das Verstreichen eines Fälligkeitsdatums, eine Ausnahmebedingung oder eine eingehende Nachricht ausgelöst werden. Der benötigte Auslöser bestimmt den Ereignistyp, den Sie implementieren. Detaillierte Informationen zu den verfügbaren Ereignistypen und den zugehörigen Auslösern finden Sie unter "Modellieren von Ereignissen".

Bindungen

Kernpunkt einer serviceorientierten Architektur ist das Konzept eines *Service*, also einer Funktionalitätseinheit, die durch eine Interaktion zwischen Datenverarbeitungsgeräten gebildet wird. Ein *Export* definiert die externe Schnittstelle (den Zugriffspunkt) eines Moduls, damit die SCA-Komponenten innerhalb des Moduls ihre Services für externe Clients bereitstellen können. Ein *Import* definiert eine Schnittstelle für

modulexterne Services, damit die Services aus dem Modul heraus aufgerufen werden können. Sie verwenden protokollspezifische *Bindungen* bei Importen und Exporten, um anzugeben, mit welchem Transportmittel die Daten in das Modul oder aus dem Modul heraus transportiert werden.

Exporte

Externe Clients können SCA-Komponenten in einem Integrationsmodul über eine Vielzahl von Protokollen (z. B. HTTP, JMS, MQ und RMI/IIOP) mit Daten in vielfältigen Formaten wie beispielsweise XML, CSV, COBOL und JavaBeans) aufrufen. Exporte sind Komponenten, die diese Anforderungen von externen Quellen empfangen und dann unter Verwendung des SCA-Programmiermodells IBM Business Process Manager-Komponenten aufrufen.

Beispielsweise empfängt in der folgenden Abbildung ein Export von einer Clientanwendung eine Anforderung über das HTTP-Protokoll. Die Daten werden in ein Geschäftsobjekt transformiert. Dies ist das von der SCA-Komponente verwendete Format. Anschließend wird die Komponente mit diesem Datenobjekt aufgerufen.

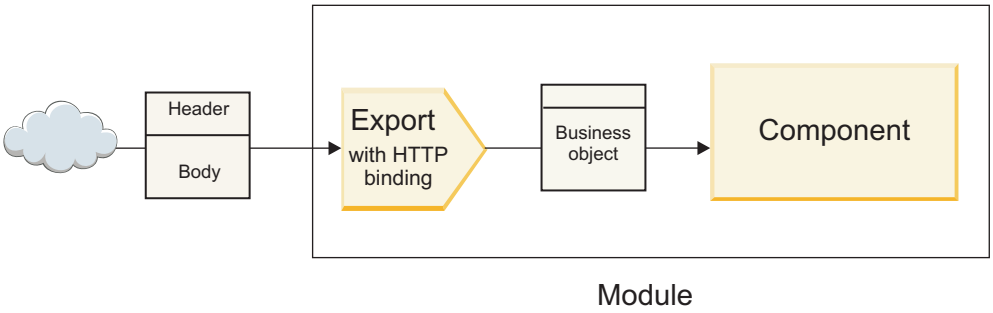


Abbildung 1. Export mit HTTP-Bindung

Importe

Es kann sein, dass eine SCA-Komponente einen externen Service aufrufen muss, der kein SCA-Service ist und Daten in einem anderen Format erwartet. Der externe Service wird von der SCA-Komponente unter Verwendung des SCA-Programmiermodells mit einem Import aufgerufen. Der Import ruft anschließend den Zielservice auf die Weise auf, die vom Service erwartet wird.

In der folgenden Abbildung wird beispielsweise eine Anforderung von einer SCA-Komponente durch den Import an einen externen Service gesendet. Das Geschäftsobjekt, also das von der SCA-Komponente erwartete Format, wird in das Format transformiert, das der Service erwartet, und der Service wird aufgerufen.

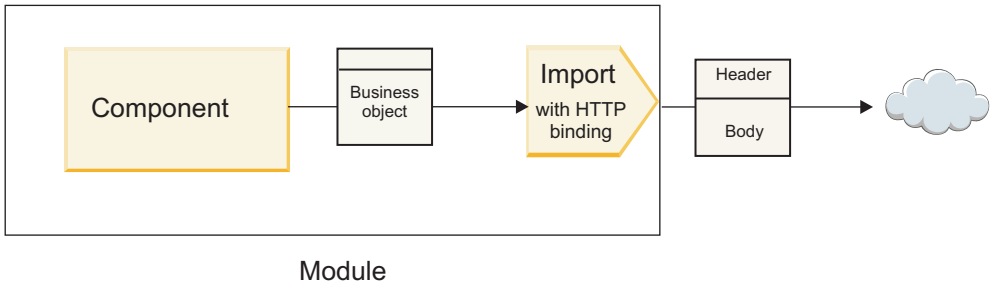


Abbildung 2. Import mit HTTP-Bindung

Liste der Bindungen

Mit Integration Designer können Sie eine Bindung für einen Import oder Export generieren und die Bindung konfigurieren. Die verfügbaren Bindungstypen sind in der folgenden Liste beschrieben.

- SCA

  Die SCA-Bindung, bei der es sich um die Standardbindung handelt, ermöglicht einem Service die Kommunikation mit Services in anderen SCA-Modulen. Einen Import mit SCA-Bindung verwenden Sie, um auf einen Service in einem anderen SCA-Modul zuzugreifen. Einen Export mit SCA-Bindung verwenden Sie, um einen Service für andere SCA-Module bereitzustellen.

- Web-Service

  Mit einer Web-Service-Bindung können Sie unter Verwendung von interoperablen SOAP-Nachrichten und Servicequalitäten auf einen externen Service zugreifen. Darüber hinaus können Sie mit Web-Service-Bindungen Anhänge als Teil der SOAP-Nachricht einbeziehen.

  Die Web-Service-Bindung kann das Übertragungsprotokoll SOAP/HTTP (SOAP über HTTP) oder SOAP/JMS (SOAP über JMS) verwenden. Unabhängig vom Transportprotokoll (HTTP oder JMS), mit dem die SOAP-Nachrichten übermittelt werden, verarbeiten Web-Service-Bindungen Anforderungs-/Antwortinteraktionen immer synchron.

- HTTP

  Mit der HTTP-Bindung können Sie mit dem HTTP-Protokoll auf einen externen Service zugreifen, wenn keine SOAP-Nachrichten verwendet werden oder ein direkter HTTP-Zugriff erforderlich ist. Diese Bindung wird verwendet, wenn Sie mit Web-Services arbeiten, die auf dem HTTP-Modell basieren, also anerkannte HTTP-Schnittstellenoperationen wie GET, PUT, DELETE usw. verwenden.

- Enterprise JavaBeans (EJB)

  Durch EJB-Bindungen können SCA-Komponenten mit Services interagieren, die durch Java EE-Geschäftslogik auf einem Java EEE-Server bereitgestellt werden.

- EIS

  Wenn die EIS-Bindung mit einem JCA-Ressourcenadapter verwendet wird, ermöglicht sie den Zugriff auf Services in einem unternehmensweiten Informationssystem (EIS) oder die Bereitstellung von Services für das EIS.

- JMS-Bindungen

  Java Message Service-Bindungen (JMS-Bindungen), generische JMS-Bindungen und WebSphere MQ-JMS-Bindungen (MQ-JMS-Bindungen) werden für Interaktionen mit Messaging-Systemen verwendet, wenn die asynchrone Übertragung über Nachrichtenwarteschlangen für die Zuverlässigkeit von wesentlicher Bedeutung ist.

  Ein Export mit einer der JMS-Bindungen überwacht eine Warteschlange auf das Eintreffen einer Nachricht und sendet die Antwort gegebenenfalls asynchron an die Antwortwarteschlange. Ein Import mit einer der JMS-Bindungen erstellt und sendet eine Nachricht an eine JMS-Warteschlange und überwacht eine Warteschlange auf das Eintreffen der Antwort (sofern vorhanden).

  – JMS

    Mit der JMS-Bindung können Sie auf den integrierten JMS-Provider von WebSphere zugreifen.

  – Generische JMS

    Mit der generischen JMS-Bindung können Sie auf ein Messaging-System eines anderen Herstellers als IBM zugreifen.

  – MQ-JMS

    Mit der MQ-JMS-Bindung können Sie auf die JMS-Untergruppe eines WebSphere MQ-Messaging-Systems zugreifen. Diese Bindung wird in der Regel verwendet, wenn die JMS-Untergruppe der Funktionen für eine Anwendung ausreichend ist.

- MQ

Die WebSphere MQ-Bindung ermöglicht die Kommunikation mit nativen MQ-Anwendungen, indem diese in das Framework der serviceorientierten Architektur integriert werden und der Zugriff auf MQ-spezifische Headerinformationen ermöglicht wird. Diese Bindung wird normalerweise verwendet, wenn native MQ-Funktionen benötigt werden.

Export- und Importbindungen - Übersicht

Mit einem Export können Sie Services in einem Integrationsmodul für externe Clients verfügbar machen. Ein Import ermöglicht den SCA-Komponenten in einem Integrationsmodul das Aufrufen externer Services. Die Bindung, die dem Export oder Import zugeordnet ist, gibt die Beziehung zwischen Protokollnachrichten und Geschäftsobjekten an. Sie gibt außerdem an, wie Operationen und Fehler ausgewählt werden.

Informationsfluss in einem Export

Ein Export empfängt eine Anforderung, die für die mit dem Export verbundene Komponente bestimmt ist, über einen bestimmten Transport, der durch die zugeordnete Bindung festgelegt wird (z. B. HTTP).

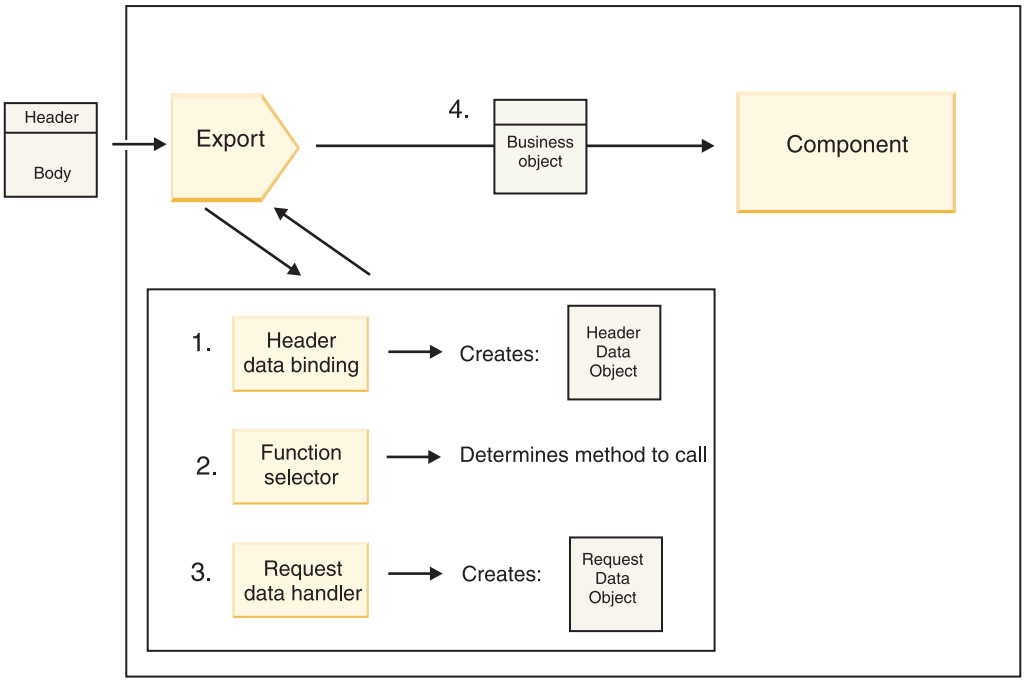


Abbildung 3. Fluss einer Anforderung durch den Export zu einer Komponente

Sobald der Export die Anforderung empfängt, findet die folgende Ereignissequenz statt:

1. Nur bei WebSphere MQ-Bindungen transformiert die Headerdatenbindung den Protokollheader in ein Headerdatenobjekt.
2. Der Funktionsselektor ermittelt den Namen der nativen Methode aus der Protokollnachricht. Der Name der nativen Methode wird durch die Exportkonfiguration zum Namen einer Operation für die Schnittstelle des Exports zugeordnet.
3. Der Anforderungsdatenhandler oder die Datenbindung für die Methode transformiert die Anforderung in ein Anforderungsgeschäftsobjekt.
4. Der Export ruft die Komponentenmethode mit dem Anforderungsgeschäftsobjekt auf.
   - Die HTTP-Exportbindung, die Web-Service-Exportbindung und die EJB-Exportbindung rufen die SCA-Komponente synchron auf.
   - Die generische JMS-, die JMS-, die MQ-JMS- und die WebSphere MQ-Exportbindung rufen die SCA-Komponente asynchron auf.

Bitte beachten Sie, dass ein Export die Header und Benutzereigenschaften, die er über das Protokoll empfängt, weitergeben kann, falls die Kontextweitergabe aktiviert ist. Komponenten, die mit dem Export verbunden sind, können dann auf diese Header und Benutzereigenschaften zugreifen. Weitere Informationen enthält das Thema über die Weiterleitung im Information Center von WebSphere Integration Developer.

Bei einer bidirektionalen Operation gibt die Komponente eine Antwort zurück.

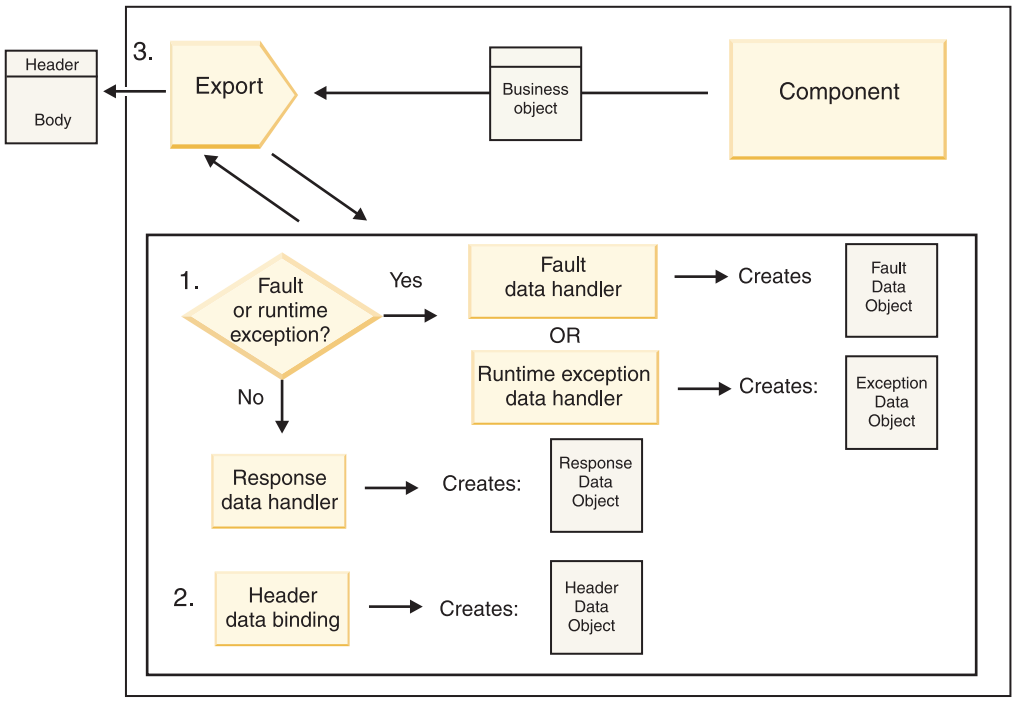


Abbildung 4. Rückfluss einer Antwort durch den Export

Die folgende Schrittsequenz findet statt:

1. Falls durch die Exportbindung eine normale Antwortnachricht empfangen wird, transformiert der Antwortdatenhandler oder die Datenbindung für die Methode das Geschäftsobjekt in eine Antwort.

   Handelt es sich bei der Antwort um einen Fehler, transformiert der Fehlerdatenhandler oder die Datenbindung für die Methode den Fehler in eine Fehlerantwort.

   Nur bei HTTP-Exportbindungen wird der Datenhandler für Laufzeitausnahmebedingungen (sofern konfiguriert) aufgerufen, falls die Antwort eine Laufzeitausnahmebedingung angibt.
2. Nur bei WebSphere MQ-Bindungen transformiert die Headerdatenbindung die Headerdatenobjekte in Protokollheader.
3. Der Export sendet die Serviceantwort über den Transport.

Informationsfluss in einem Import

Komponenten verwenden einen Import, um Anforderungen an modulexterne Services zu senden. Die Anforderung wird über einen bestimmten Transport gesendet, der durch die zugehörige Bindung festgelegt wird.

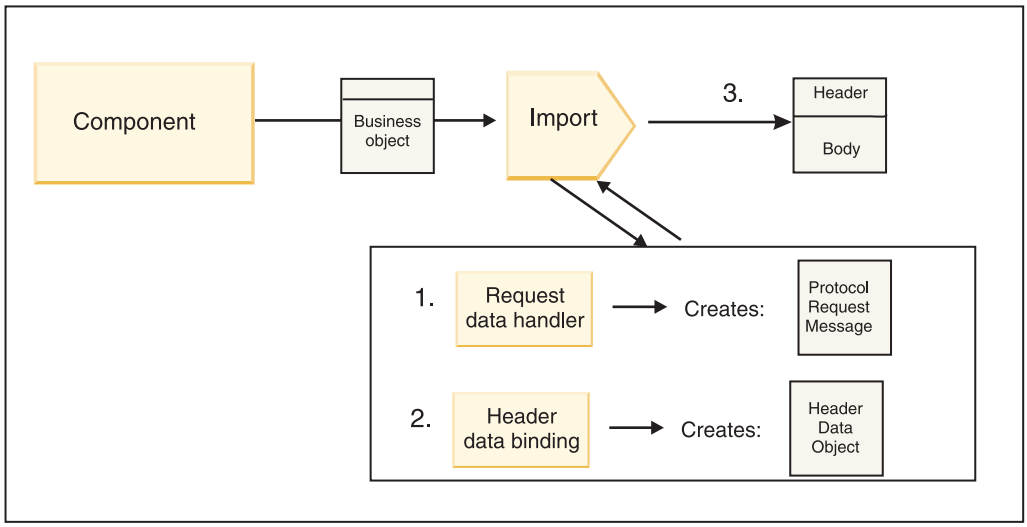


Abbildung 5. Fluss von einer Komponente über den Import zu einem Service

Die Komponente ruft den Import mit einem Anforderungsgeschäftsobjekt auf.

Anmerkung:

- Die HTTP-Importbindung, die Web-Service-Importbindung und die EJB-Importbindung sollten durch die aufrufende Komponente synchron aufgerufen werden.
- Die generische JMS-, die JMS-, die MQ-JMS- und die WebSphere MQ-Importbindung sollten asynchron aufgerufen werden.

Nachdem die Komponente den Import aufgerufen hat, findet die folgende Ereignissequenz statt:

1. Der Anforderungsdatenhandler oder die Datenbindung für die Methode transformiert das Anforderungsgeschäftsobjekt in eine Protokollanforderungsnachricht.
2. Nur bei WebSphere MQ-Bindungen legt die Headerdatenbindung für die Methode das Headerdatenobjekt im Protokollheader fest.
3. Der Import ruft den Service mit der Serviceanforderung über den Transport auf.

Bei einer bidirektionalen Operation gibt der Service eine Antwort zurück. Daraufhin findet die folgende Schrittsequenz statt:

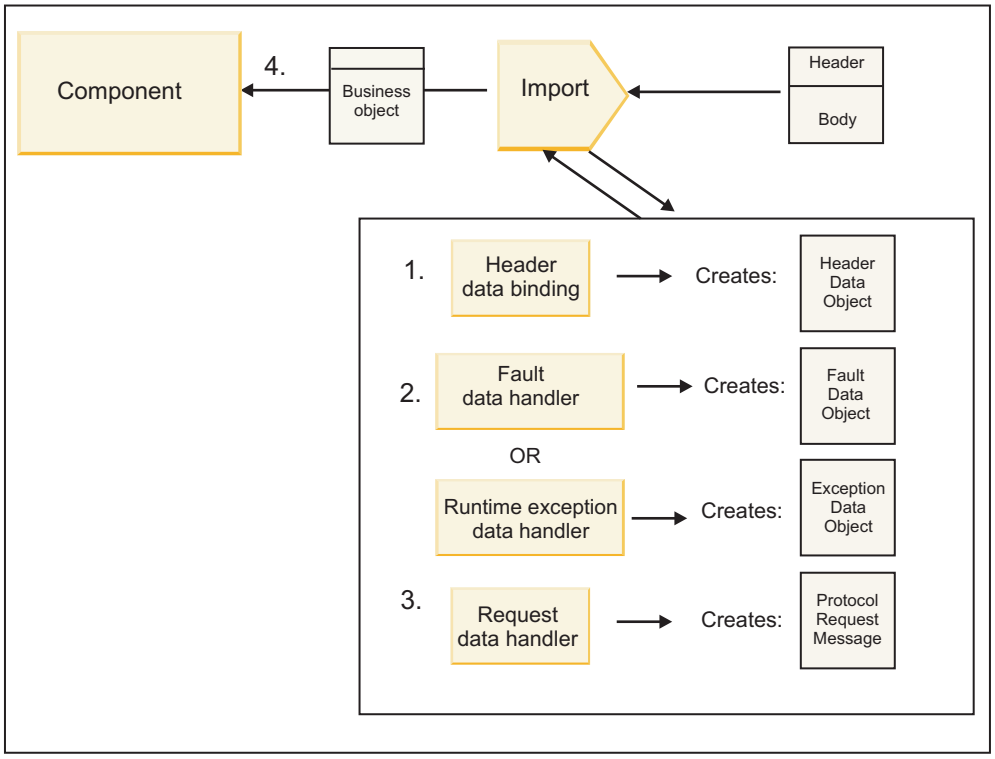


Abbildung 6. Rückfluss einer Antwort durch den Import

1. Nur bei WebSphere MQ-Bindungen transformiert die Headerdatenbindung den Protokollheader in ein Headerdatenobjekt.
2. Es wird ermittelt, ob es sich bei der Antwort um einen Fehler handelt.
   - Falls die Antwort einen Fehler angibt, überprüft der Fehlerselektor den Fehler und ermittelt, zu welchem WSDL-Fehler dieser zugeordnet ist. Der Fehlerdatenhandler für die Methode transformiert anschließend den Fehler in eine Fehlerantwort.
   - Falls die Antwort eine Laufzeitausnahmebedingung angibt, wird der Datenhandler für Laufzeitausnahmebedingungen (sofern konfiguriert) aufgerufen.
3. Der Antwortdatenhandler oder die Bindung für die Methode transformiert die Antwort in ein Antwortgeschäftsobjekt.
4. Der Import gibt das Antwortgeschäftsobjekt an die Komponente zurück.

Export- und Importbindungen - Konfiguration

Einer der wichtigsten Aspekte von Export- und Importbindungen ist die Datenformattransformation. Sie gibt an, wie Daten aus einem nativen Sendeformat zu einem Geschäftsobjekt zugeordnet (deserialisiert) werden oder wie sie aus einem Geschäftsobjekt zu einem nativen Sendeformat zugeordnet (serialisiert) werden. Bei Bindungen, die Exporten zugeordnet sind, können Sie außerdem durch die Auswahl eines Funktionsselektors angeben, welche Operation für die Daten ausgeführt werden soll. Bei Bindungen, die Exporten oder Importen zugeordnet sind, können Sie angeben, wie Fehler gehandhabt werden sollen, die während der Verarbeitung auftreten.

Darüber hinaus können Sie für Bindungen transportspezifische Informationen angeben. Für eine HTTP-Bindung geben Sie beispielsweise die Endpunkt-URL an. Die transportspezifischen Informationen für eine HTTP-Bindung sind in den Themen 'HTTP-Importbindung generieren' und 'HTTP-Exportbindung generieren' beschrieben. Im Information Center finden Sie außerdem Informationen zu anderen Bindungen.

Datenformattransformation bei Importen und Exporten:

Wenn Sie eine Export- oder Importbindung in IBM Integration Designer konfigurieren, geben Sie unter anderem die Konfigurationseigenschaft für das von der Bindung verwendete Datenformat an.

- Für Exportbindungen, bei denen eine Clientanwendung Anforderungen an eine SCA-Komponente sendet und von dieser Antworten empfängt, geben Sie das Format der nativen Daten an. Je nach Format wählt das System den geeigneten Datenhandler bzw. die geeignete Datenbindung aus, um die nativen Daten in ein Geschäftsobjekt zu transformieren (das von der SCA-Komponente verwendet wird) und um im Gegenzug das Geschäftsobjekt in native Daten zu transformieren (die die Antwort an die Clientanwendung darstellen).
- Für Importbindungen, bei denen eine SCA-Komponente Anforderungen an einen modulexternen Service sendet und von diesem Antworten empfängt, geben Sie das Datenformat der nativen Daten an. Je nach Format wählt das System den geeigneten Datenhandler bzw. die geeignete Datenbindung aus, um das Geschäftsobjekt in native Daten zu transformieren (und umgekehrt).

IBM Business Process Manager bietet eine Reihe von vordefinierten Datenformaten und entsprechenden Datenhandlern oder Datenbindungen, die die Formate unterstützen. Sie können auch eigene benutzerdefinierte Datenhandler erstellen und das Datenformat für diese Datenhandler registrieren. Weitere Informationen enthält das Thema über die Entwicklung von Datenhandlern im Information Center von IBM Integration Designer.

- *Datenhandler* sind protokollneutral und transformieren Daten aus einem Format in ein anderes Format. In IBM Business Process Manager transformieren Datenhandler normalerweise native Daten (z. B. XML, CVS und COBOL) in ein Geschäftsobjekt sowie ein Geschäftsobjekt in native Daten. Weil Datenhandler protokollneutral sind, können Sie denselben Datenhandler bei einer Vielzahl von Export- und Importbindungen wiederverwenden. Sie können beispielsweise denselben XML-Datenhandler bei einer HTTP-Bindung für den Export oder Import oder bei einer JMS-Bindung für den Export oder Import verwenden.
- *Datenbindungen* transformieren ebenfalls native Daten in ein Geschäftsobjekt (und umgekehrt), sind jedoch protokollspezifisch. Eine HTTP-Datenbindung kann beispielsweise nur bei einer HTTP-Exportbindung oder einer HTTP-Importbindung verwendet werden. Im Gegensatz zu Datenhandlern kann eine HTTP-Datenbindung nicht bei einer MQ-Exportbindung oder -Importbindung wiederverwendet werden.

  Anmerkung: Bei Version 7.0 von IBM Business Process Manager werden drei HTTP-Datenbindungen (HTTPStreamDataBindingSOAP, HTTPStreamDataBindingXML und HTTPServiceGatewayDataBinding) nicht mehr unterstützt. Verwenden Sie nach Möglichkeit immer Datenhandler.

Wie bereits erwähnt, können Sie bei Bedarf benutzerdefinierte Datenhandler erstellen. Sie können ebenfalls benutzerdefinierte Datenbindungen erstellen. Es empfiehlt sich jedoch, benutzerdefinierte Datenhandler zu erstellen, da diese für mehrere Bindungen verwendet werden können.

Datenhandler:

Datenhandler werden für Export- und Importbindungen konfiguriert, um Daten auf protokollneutrale Weise von einem Format in ein anderes Format zu transformieren. Im Rahmen des Produkts werden mehrere Datenhandler bereitgestellt. Bei Bedarf können Sie aber auch einen eigenen Datenhandler erstellen. Sie können einen Datenhandler einer Export- oder Importbindung auf einer von zwei Ebenen zuordnen. Entweder ordnen Sie ihn allen Operationen in der Schnittstelle des Exports oder Imports zu oder einer bestimmten Operation für die Anforderung bzw. Antwort.

Vordefinierte Datenhandler

Sie verwenden IBM Integration Designer, um den Datenhandler anzugeben, den Sie verwenden wollen.

Die vordefinierten Datenhandler sind in der folgenden Tabelle aufgeführt. Außerdem ist dort beschrieben, wie jeder Datenhandler eingehende und abgehende Daten transformiert.

Anmerkung: Wenn nicht explizit eine Ausnahme angegeben ist, können diese Datenhandler bei JMS-, generischen JMS-, MQ-JMS-, WebSphere MQ- und HTTP-Bindungen verwendet werden.
Detailinformationen enthält das Thema 'Datenhandler' im Information Center von Integration Designer.

Tabelle 23. Vordefinierte Datenhandler

Datenhandler	Native Daten in Geschäftsobjekt	Geschäftsobjekt in native Daten
ATOM	Führt eine Syntaxanalyse für Atom-Feeds durch und stellt sie in ein Geschäftsobjekt für Atom-Feeds.	Serialisiert ein Geschäftsobjekt für Atom-Feeds in Atom-Feeds.
Mit Begrenzer	Führt eine Syntaxanalyse von Daten mit Begrenzer durch und stellt sie in ein Geschäftsobjekt.	Serialisiert ein Geschäftsobjekt in Daten mit Begrenzer (inklusive CVS).
Feste Breite	Führt eine Syntaxanalyse von Daten mit fester Breite durch und stellt sie in ein Geschäftsobjekt.	Serialisiert ein Geschäftsobjekt in Daten mit fester Breite.
Verarbeitung durch WTX	Delegiert die Datenformattransformation an WebSphere Transformation Extender (WTX). Der WTX-Zuordnungsname wird durch den Datenhandler abgeleitet.	Delegiert die Datenformattransformation an WebSphere Transformation Extender (WTX). Der WTX-Zuordnungsname wird durch den Datenhandler abgeleitet.
Verarbeitung durch WTX Invoker	Delegiert die Datenformattransformation an WebSphere Transformation Extender (WTX). Der WTX-Zuordnungsname wird durch den Benutzer angegeben.	Delegiert die Datenformattransformation an WebSphere Transformation Extender (WTX). Der WTX-Zuordnungsname wird durch den Benutzer angegeben.
JAXB	Serialisiert Java-Beans in ein Geschäftsobjekt unter Verwendung der Zuordnungsregeln, die in der Spezifikation von Java Architecture for XML Binding (JAXB) definiert sind.	Deserialisiert ein Geschäftsobjekt in Java-Beans unter Verwendung der Zuordnungsregeln, die in der JAXB-Spezifikation definiert sind.
JAXWS **Anmerkung:** Der JAXWS-Datenhandler kann nur bei einer EJB-Bindung verwendet werden.	Wird von einer EJB-Bindung für die Transformation eines Java-Antwortobjekts oder eines Java-Ausnahmeobjekts in ein Antwortgeschäftsobjekt verwendet. Hierbei kommen die Zuordnungsregeln zum Einsatz, die in der Java API for XML Web Services (JAX-WS)-Spezifikation definiert sind.	Wird von einer EJB-Bindung für die Transformation eines Geschäftsobjekts in die abgehenden Java-Methodenparameter verwendet. Hierbei kommen die Zuordnungsregeln zum Einsatz, die in der JAX-WS-Spezifikation definiert sind.
JSON	Führt eine Syntaxanalyse von JSON-Daten durch und stellt sie in ein Geschäftsobjekt.	Serialisiert ein Geschäftsobjekt in JSON-Daten.
Nativer Hauptteil	Führt eine Syntaxanalyse der nativen Byte, des nativen Textes, der nativen Zuordnung, des nativen Datenstroms oder des nativen Objekts durch und stellt sie in eines von fünf Basisgeschäftsobjekten (Text, Byte, Zuordnung, Datenstrom oder Objekt).	Transformiert die fünf Basis-Geschäftsobjekte in Byte, Text, Zuordnung, Datenstrom oder Objekt.

Tabelle 23. Vordefinierte Datenhandler (Forts.)

Datenhandler	Native Daten in Geschäftsobjekt	Geschäftsobjekt in native Daten
SOAP	Führt eine Syntaxanalyse der SOAP-Nachricht (und des Headers) durch und stellt sie in ein Geschäftsobjekt.	Serialisiert ein Geschäftsobjekt in eine SOAP-Nachricht.
XML	Führt eine Syntaxanalyse von XML-Daten durch und stellt sie in ein Geschäftsobjekt.	Serialisiert ein Geschäftsobjekt in XML-Daten.
UTF8XMLDataHandler	Führt eine Syntaxanalyse von in UTF-8 codierten XML-Daten durch und stellt sie in ein Geschäftsobjekt.	Serialisiert ein Geschäftsobjekt beim Senden einer Nachricht in XML-Daten, die in UTF-8 codiert sind.

Datenhandler erstellen

Ausführliche Informationen zum Erstellen eines Datenhandlers enthält das Thema über die Entwicklung von Datenhandlern im Information Center von Integration Designer.

Datenbindungen:

Datenbindungen werden für Export- und Importbindungen konfiguriert, um Daten von einem Format in ein anderes Format zu transformieren. Datenbindungen sind protokollspezifisch. Im Rahmen des Produkts werden mehrere Datenbindungen bereitgestellt. Bei Bedarf können Sie aber auch eine eigene Datenbindung erstellen. Sie können eine Datenbindung einer Export- oder Importbindung auf einer von zwei Ebenen zuordnen. Entweder ordnen Sie sie allen Operationen in der Schnittstelle des Exports oder Imports zu oder einer bestimmten Operation für die Anforderung bzw. Antwort.

Sie verwenden IBM Integration Designer, um die zu verwendende Datenbindung anzugeben oder um eine eigene Datenbindung zu erstellen. Eine Erläuterung dazu, wie Datenbindungen erstellt werden, finden Sie im Abschnitt mit der Übersicht über JMS-, MQ-JMS- und generische JMS-Bindungen im Information Center von IBM Integration Designer.

JMS-Bindungen

In der folgenden Tabelle sind die Datenbindungen aufgeführt, die bei den folgenden Bindungen verwendet werden können:

- JMS-Bindungen
- Generische JMS-Bindungen
- WebSphere MQ-JMS-Bindungen

Außerdem enthält die Tabelle eine Beschreibung der Tasks, die von der Datenbindung ausgeführt werden.

Tabelle 24. Vordefinierte Datenbindungen für JMS-Bindungen

Datenbindung	Native Daten in Geschäftsobjekt	Geschäftsobjekt in native Daten
Serialisiertes Java-Objekt	Transformiert das serialisierte Java-Objekt in ein Geschäftsobjekt (das in WSDL als Eingabe- oder Ausgabetyp zugeordnet ist).	Serialisiert ein Geschäftsobjekt in das serialisierte Java-Objekt in der JMS-Objektnachricht.
Eingeschlossene Byte	Extrahiert die Byte aus der eingehenden JMS-Bytenachricht und schließt sie in das Geschäftsobjekt 'JMSBytesBody' ein.	Extrahiert die Byte aus dem Geschäftsobjekt 'JMSBytesBody' und schließt sie in die abgehende JMS-Bytenachricht ein.

Tabelle 24. Vordefinierte Datenbindungen für JMS-Bindungen (Forts.)

Datenbindung	Native Daten in Geschäftsobjekt	Geschäftsobjekt in native Daten
Eingeschlossener Zuordnungseintrag	Extrahiert die Informationen zu Name, Wert und Typ für jeden Eintrag in der eingehenden JMS-Zuordnungsnachricht und erstellt eine Liste mit Geschäftsobjekten 'MapEntry'. Anschließend wird die Liste in das Geschäftsobjekt 'JMSMapBody' eingeschlossen.	Extrahiert die Informationen zu Name, Wert und Typ aus der Liste 'MapEntry' im Geschäftsobjekt 'JMSMapBody' und erstellt die korrespondierenden Einträge in der abgehenden JMS-Zuordnungsnachricht.
Eingeschlossenes Objekt	Extrahiert das Objekt aus der eingehenden JSM-Objektnachricht und schließt es in das Geschäftsobjekt 'JMSObjectBody' ein.	Extrahiert das Objekt aus dem Geschäftsobjekt 'JMSObjectBody' und schließt es in die abgehende JMS-Objektnachricht ein.
Eingeschlossener Text	Extrahiert den Text aus der eingehenden JSM-Textnachricht und schließt ihn in das Geschäftsobjekt 'JMSTextBody' ein.	Extrahiert den Text aus dem Geschäftsobjekt 'JMSTextBody' und schließt ihn in die abgehende JMS-Textnachricht ein.

WebSphere MQ-Bindungen

In der folgenden Tabelle sind die Datenbindungen aufgeführt, die mit WebSphere MQ verwendet werden können. Außerdem sind die von den Datenbindungen ausgeführten Tasks beschrieben.

Tabelle 25. Vordefinierte Datenbindungen für WebSphere MQ-Bindungen

Datenbindung	Native Daten in Geschäftsobjekt	Geschäftsobjekt in native Daten
Serialisiertes Java-Objekt	Transformiert das serialisierte Java-Objekt aus der eingehenden Nachricht in ein Geschäftsobjekt (das in WSDL als Eingabe- oder Ausgabetyp zugeordnet ist).	Transformiert ein Geschäftsobjekt in das serialisierte Java-Objekt in der ausgehenden Nachricht.
Eingeschlossene Byte	Extrahiert die Byte aus der unstrukturierten MQ-Bytenachricht und schließt sie in das Geschäftsobjekt JMSBytesBody ein.	Extrahiert die Byte aus dem Geschäftsobjekt JMSBytesBody und schließt die Byte in die ausgehende unstrukturierte MQ-Bytenachricht ein.
Eingeschlossener Text	Extrahiert den Text aus einer unstrukturierten MQ-Textnachricht und schließt ihn in ein Geschäftsobjekt 'JMSTextBody' ein.	Extrahiert den Text aus einem Geschäftsobjekt 'JMSTextBody' und schließt ihn eine unstrukturierte MQ-Textnachricht ein.
Eingeschlossener Datenstromeintrag	Extrahiert die Informationen zu Name und Typ für jeden Eintrag in der eingehenden JMS-Datenstromnachricht und erstellt eine Liste der Geschäftsobjekte 'StreamEntry'. Anschließend wird die Liste in das Geschäftsobjekt 'JMSStreamBody' eingeschlossen.	Extrahiert die Informationen zu Name und Typ aus der Liste 'StreamEntry' im Geschäftsobjekt 'JMSStreamBody' und erstellt die korrespondierenden Einträge in der abgehenden JMS-Datenstromnachricht.

Neben den in Tabelle 25 aufgelisteten Datenbindungen verwendet WebSphere MQ außerdem Headerdatenbindungen. Ausführliche Informationen hierzu finden Sie im Information Center von IBM Integration Designer.

HTTP-Bindungen

In der folgenden Tabelle sind die Datenbindungen aufgeführt, die mit HTTP verwendet werden können. Außerdem sind die Tasks beschrieben, die die Datenbindungen ausführen.

Tabelle 26. Vordefinierte Datenbindungen für HTTP-Bindungen

Datenbindung	Native Daten in Geschäftsobjekt	Geschäftsobjekt in native Daten
Eingeschlossene Byte	Extrahiert die Byte aus der eingehenden HTTP-Nachricht und schließt sie in das Geschäftsobjekt 'HTTPBytes' ein.	Extrahiert die Byte aus dem Geschäftsobjekt 'HTTPBytes' und fügt sie zum Hauptteil der abgehenden HTTP-Nachricht hinzu.
Eingeschlossener Text	Extrahiert den Text aus dem Hauptteil der eingehenden HTTP-Nachricht und schließt ihn in das Geschäftsobjekt 'HTTPText' ein.	Extrahiert den Text aus dem Geschäftsobjekt 'HTTPText' und fügt ihn zum Hauptteil der abgehenden HTTP-Nachricht hinzu.

Funktionsselektoren in Exportbindungen:

Mit einem Funktionsselektor wird angegeben, welche Operation für die Daten einer Anforderungsnachricht ausgeführt werden soll. Funktionsselektoren werden im Rahmen einer Exportbindung konfiguriert.

Dies soll am Beispiel eines SCA-Exports veranschaulicht werden, der eine Schnittstelle zugänglich macht. Die Schnittstelle enthält die beiden Operationen 'Create' und 'Update'. Der Export besitzt eine JMS-Bindung, die Daten aus einer Warteschlange liest.

Wenn in der Warteschlange eine Nachricht eintrifft, werden an den Export die zugehörigen Daten übergeben. Es muss jedoch ermittelt werden, welche Operation aus der Schnittstelle des Exports für die verbundene Komponente aufgerufen werden muss. Die Operation wird durch den Funktionsselektor und durch die Konfiguration der Exportbindung bestimmt.

Der Funktionsselektor gibt den nativen Funktionsnamen zurück, also den Funktionsnamen im Clientsystem, das die Nachricht gesendet hat. Der native Funktionsname wird dann dem Operations- oder Funktionsnamen in der Schnittstelle zugeordnet, die dem Export zugeordnet ist. In der folgenden Abbildung gibt beispielsweise der Funktionsselektor den nativen Funktionsnamen (CRT) aus der eingehenden Nachricht zurück. Der native Funktionsname wird der Operation 'Create' zugeordnet und das Geschäftsobjekt wird mit der Operation 'Create' an die SCA-Komponente gesendet.

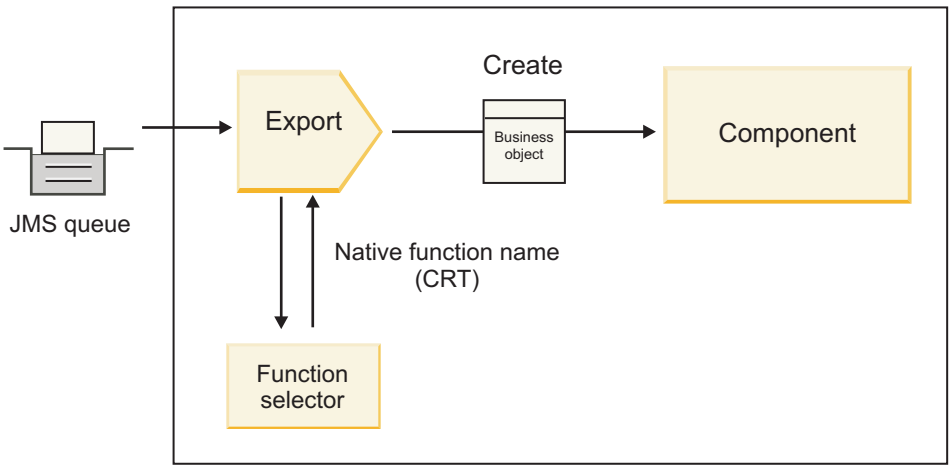


Abbildung 7. Funktionsselektor

Falls die Schnittstelle lediglich mit einer einzigen Operation ausgestattet ist, ist die Angabe eines Funktionsselektors nicht erforderlich.

Mehrere vordefinierte Funktionsselektoren sind verfügbar. Sie sind in den nachfolgenden Abschnitten aufgelistet.

JMS-Bindungen

In der nachstehenden Tabelle sind die Funktionsselektoren aufgeführt, die bei den folgenden Bindungen verwendet werden können:

- JMS-Bindungen
- Generische JMS-Bindungen
- WebSphere MQ-JMS-Bindungen

Tabelle 27. Vordefinierte Funktionsselektoren für JMS-Bindungen

Funktionsselektor	Beschreibung
JMS-Funktionsselektor für einfache JMS-Datenbindungen	Verwendet die Eigenschaft 'JMSType' der Nachricht, um den Operationsnamen auszuwählen.
Funktionsselektor für JMS-Headereigenschaft	Gibt den Wert der JMS-Zeichenfolgeeigenschaft 'TargetFunctionName' aus dem Header zurück.
Funktionsselektor für JMS-Service-Gateway	Ermittelt, ob es sich bei der Anforderung um eine unidirektionale oder um eine bidirektionale Operation handelt, indem die vom Client festgelegte Eigenschaft 'JMSReplyTo' untersucht wird.

WebSphere MQ-Bindungen

In der folgenden Tabelle sind die Funktionsselektoren aufgeführt, die bei WebSphere MQ-Bindungen verwendet werden können:

Tabelle 28. Vordefinierte Funktionsselektoren für WebSphere MQ-Bindungen

Funktionsselektor	Beschreibung
MQ-Funktionsselektor für 'handleMessage'	Gibt 'handleMessage' als einen Wert zurück, der mithilfe der Exportmethodenbindungen dem Namen einer Operation in der Schnittstelle zugeordnet wird.
MQ verwendet den JMS-Standardfunktionsselektor	Liest die native Operation aus der Eigenschaft 'TargetFunctionName' des Ordners eines MQRFH2-Headers.
MQ verwendet das Format des Nachrichtenhauptteils als native Funktion	Sucht nach dem Formatfeld des letzten Headers und gibt dieses Feld als Zeichenfolge zurück.
MQ-Funktionsselektor für 'type'	Erstellt in der Exportbindung eine Methode, indem eine URL abgerufen wird, die die Eigenschaften 'Msd', 'Set', 'Type' und 'Format' enthält, die im MQRFH2-Header gefunden wurden.
Funktionsselektor für MQ-Service-Gateway	Verwendet die Eigenschaft 'MsgType' im MQMD-Header, um den Operationsnamen zu ermitteln.

HTTP-Bindungen

In der folgenden Tabelle sind die Funktionsselektoren aufgeführt, die bei HTTP-Bindungen verwendet werden können:

Tabelle 29. Vordefinierte Funktionsselektoren für HTTP-Bindungen

Funktionsselektor	Beschreibung
HTTP-Funktionsselektor auf der Basis des Headers 'TargetFunctionName'	Verwendet die HTTP-Headereigenschaft 'TargetFunctionName' aus dem Client, um zu ermitteln, welche Operation zur Laufzeit aus dem Export aufgerufen werden soll.
HTTP-Funktionsselektor auf der Basis der URL und der HTTP-Methode	Verwendet den relativen Pfad der URL und hängt die HTTP-Methode des Clients an, um die native Operation festzustellen, die für den Export definiert ist.
Funktionsselektor für HTTP-Service-Gateway auf der Basis einer URL mit einem Operationsnamen	Ermittelt die aufzurufende Methode auf Basis der URL, falls 'perationMode = oneway' an die Anforderungs-URL angehängt worden ist.

Anmerkung: Mit IBM Integration Designer können Sie auch einen eigenen Funktionsselektor erstellen. Informationen zum Erstellen eines Funktionsselektors enthält das Information Center von IBM Integration Designer. Die Erstellung eines Funktionsselektors ist beispielsweise in der Übersicht über die MQ-Funktionsselektoren beschrieben.

Fehlerbehandlung:

Sie können Import- und Exportbindungen für den Umgang mit Fehlern (zum Beispiel Geschäftsausnahmebedingungen) konfigurieren, die während der Verarbeitung auftreten, indem Sie Fehlerdatenhandler angeben. Ein Fehlerdatenhandler kann auf drei Ebenen konfiguriert werden: Sie können einen Fehlerdatenhandler einem Fehler, einer Operation oder allen Operationen mit einer Bindung zuordnen.

Ein Fehlerdatenhandler verarbeitet Fehlerdaten und transformiert sie in das korrekte Format, das durch die Export- oder Importbindung gesendet werden muss.

- Bei einer Exportbindung transformiert der Fehlerdatenhandler das Ausnahmebedingungsgeschäftsobjekt, das von der Komponente gesendet wurde, in eine Antwortnachricht, die von der Clientanwendung verwendet werden kann.
- Bei einer Importbindung transformiert der Fehlerdatenhandler die Fehlerdaten oder die Antwortnachricht, die von einem Service gesendet wurden, in ein Ausnahmebedingungsgeschäftsobjekt, das von der SCA-Komponente verwendet werden kann.

Bei Importbindungen ruft die Bindung den Fehlerselektor auf. Dieser ermittelt, ob die Antwortnachricht eine normale Antwort, eine Geschäftsausnahmebedingung oder eine Laufzeitausnahmebedingung darstellt.

Sie können einen Fehlerdatenhandler für einen bestimmten Fehler, für eine Operation und für alle Operationen mit einer Bindung zuordnen.

- Falls der Fehlerdatenhandler auf allen drei Ebenen festgelegt ist, wird der Datenhandler aufgerufen, der einem bestimmten Fehler zugeordnet ist.
- Sind Fehlerdatenhandler auf Operations- und Bindungsebene festgelegt, wird der Datenhandler aufgerufen, der der Operation zugeordnet ist.

Zur Angabe der Fehlerbehandlung werden zwei Editoren in IBM Integration Designer verwendet. Mit dem Schnittstelleneditor wird angegeben, ob in einer Operation ein Fehler auftritt. Nachdem mit dieser Schnittstelle eine Bindung generiert wurde, können Sie im Editor der Eigenschaftensicht konfigurieren, wie der Fehler verarbeitet werden soll. Weitere Informationen enthält das Thema über Fehlerselektoren im Information Center von IBM Integration Designer.

Fehlerbehandlung in Exportbindungen:

Wenn beim Verarbeiten der Anforderung von einer Clientanwendung ein Fehler auftritt, kann die Exportbindung die Fehlerinformationen an den Client zurückgeben. Beim Konfigurieren der Exportbindung geben Sie an, wie der Fehler verarbeitet und an den Client zurückgegeben werden soll.

Zur Konfiguration der Exportbindung verwenden Sie IBM Integration Designer.

Während der Anforderungsverarbeitung ruft ein Client einen Export mit einer Anforderung auf. Der Export ruft dann die SCA-Komponente auf. Während der Verarbeitung der Anforderung kann die SCA-Komponente entweder eine Geschäftsantwort zurückgeben oder eine Geschäftsausnahmebedingung bzw. eine Laufzeitausnahmebedingung für den Service auslösen. In einem solchen Fall transformiert die Exportbindung die Ausnahmebedingung in eine Fehlernachricht und sendet diese an den Client. Dies ist in der folgenden Abbildung dargestellt und in den anschließenden Abschnitten beschrieben.

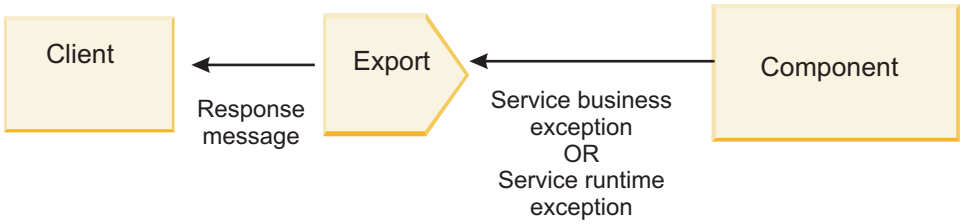


Abbildung 8. Verfahren beim Senden von Fehlerinformationen von der Komponente durch die Exportbindung an den Client

Zur Verarbeitung von Fehlern können Sie einen benutzerdefinierten Datenhandler oder eine benutzerdefinierte Datenbindung erstellen.

Geschäftsausnahmebedingungen

Geschäftsausnahmebedingungen sind Geschäftsfehler oder -ausnahmebedingungen, die während der Verarbeitung auftreten.

Dies soll am Beispiel der folgenden Schnittstelle erläutert werden, für die es eine Operation 'createCustomer' gibt. Für diese Operation sind zwei Geschäftsausnahmebedingungen definiert, nämlich 'CustomerAlreadyExists' (= Kunde ist bereits vorhanden) und 'MissingCustomerId' (= Kunden-ID fehlt).

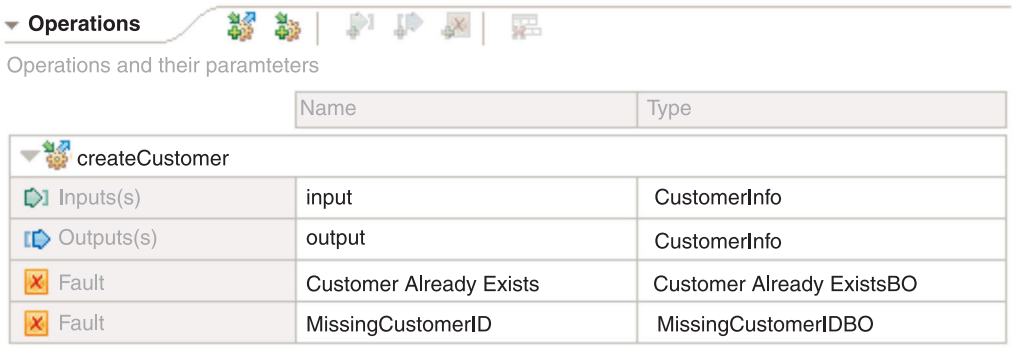


Abbildung 9. Schnittstelle mit zwei Ausnahmebedingungen

Falls in diesem Beispiel ein Client eine Anforderung zum Erstellen eines Kunden (an diese SCA-Komponente) sendet und der Kunde bereits vorhanden ist, löst die Komponente eine Ausnahmebedingung 'Cus-

tomerAlreadyExists' für den Export aus. Der Export muss diese Geschäftsausnahmebedingung an den aufrufenden Client zurückleiten. Zu diesem Zweck wird der Fehlerdatenhandler verwendet, der für die Exportbindung konfiguriert ist.

Sobald durch die Exportbindung eine Geschäftsausnahmebedingung empfangen wird, findet die folgende Verarbeitung statt:

1. Die Bindung stellt fest, welcher Fehlerdatendatenhandler zur Verarbeitung der Ausnahmebedingung aufgerufen werden muss. Falls die Geschäftsausnahmebedingung für den Service den Namen der Ausnahmebedingung enthält, wird der Datenhandler aufgerufen, der für die Ausnahmebedingung konfiguriert ist. Enthält die Geschäftsausnahmebedingung für den Service den Namen der Ausnahmebedingung nicht, wird der Name der Ausnahmebedingung durch einen Abgleich der Fehler- oder Ausnahmebedingungstypen abgeleitet.
2. Die Bindung ruft den Fehlerdatenhandler mit dem Datenobjekt aus der Geschäftsausnahmebedingung für den Service auf.
3. Der Fehlerdatenhandler transformiert das Fehlerdatenobjekt in eine Antwortnachricht und gibt diese an die Exportbindung zurück.
4. Der Export gibt die Antwortnachricht an den Client zurück.

Falls die Geschäftsausnahmebedingung für den Service den Namen der Ausnahmebedingung enthält, wird der Datenhandler aufgerufen, der für die Ausnahmebedingung konfiguriert ist. Enthält die Geschäftsausnahmebedingung für den Service den Namen der Ausnahmebedingung nicht, wird der Name der Ausnahmebedingung durch einen Abgleich der Fehler- oder Ausnahmebedingungstypen abgeleitet.

Laufzeitausnahmebedingung

Eine Laufzeitausnahmebedingung tritt in der SCA-Anwendung während der Verarbeitung einer Anforderung auf, die keiner Geschäftsausnahmebedingung entspricht. Im Gegensatz zu Geschäftsausnahmebedingungen sind Laufzeitausnahmebedingungen nicht in der Schnittstelle definiert.

In bestimmten Szenarios ist es wünschenswert, dass diese Laufzeitausnahmebedingungen an die Clientanwendung weitergegeben werden, damit die Clientanwendung eine geeignete Aktion ausführen kann.

Falls beispielsweise ein Client eine Anforderung zum Erstellen eines Kunden (an die SCA-Komponente) sendet und während der Verarbeitung der Anforderung ein Berechtigungsfehler auftritt, löst die Komponente eine Laufzeitausnahmebedingung aus. Diese Laufzeitausnahmebedingung muss an den aufrufenden Client zurückgegeben werden, damit dieser die geeignete Aktion hinsichtlich der Berechtigung ausführen kann. Dies wird durch den Datenhandler für Laufzeitausnahmebedingungen erreicht, der für die Exportbindung konfiguriert ist.

Anmerkung: Sie können einen Datenhandler für Laufzeitausnahmebedingungen nur bei HTTP-Bindungen konfigurieren.

Die Verarbeitung einer Laufzeitausnahmebedingung erfolgt ähnlich wie bei einer Geschäftsausnahmebedingung. Falls ein Datenhandler für Laufzeitausnahmebedingungen konfiguriert wurde, findet die folgende Verarbeitung statt:

1. Die Exportbindung ruft den entsprechenden Datenhandler mit der Laufzeitausnahmebedingung für den Service auf.
2. Der Datenhandler transformiert das Fehlerdatenobjekt in eine Antwortnachricht und gibt diese an die Exportbindung zurück.
3. Der Export gibt die Antwortnachricht an den Client zurück.

Die Fehlerbehandlung und die Behandlung von Laufzeitausnahmebedingungen sind optional. Wenn Sie nicht wollen, dass Fehler oder Laufzeitausnahmebedingungen an den aufrufenden Client zurückgegeben werden, konfigurieren Sie den Fehlerdatenhandler oder den Datenhandler für Laufzeitausnahmebedingungen nicht.

Fehlerbehandlung in Importbindungen:

Eine Komponente verwendet einen Import, um eine Anforderung an einen Service außerhalb des Moduls zu senden. Wenn während der Verarbeitung der Anforderung ein Fehler auftritt, gibt der Service den Fehler an die Importbindung zurück. Sie können beim Konfigurieren der Importbindung angeben, wie der Fehler verarbeitet und an die Komponente zurückgegeben werden soll.

Zur Konfiguration der Importbindung verwenden Sie IBM Integration Designer. Sie können einen Fehlerdatenhandler (oder eine Datenbindung) angeben. Außerdem geben Sie einen Fehlerselektor an.

Fehlerdatenhandler

Der Service, der die Anforderung verarbeitet, sendet Fehlerinformationen an die Importbindung in Form einer Ausnahmebedingung oder einer Antwortnachricht, die die Fehlerdaten enthält.

Die Importbindung transformiert die Serviceausnahmebedingung oder -antwortnachricht in eine Geschäftsausnahmebedingung oder Laufzeitausnahmebedingung für den Service. Dies ist in der folgenden Abbildung dargestellt und in den anschließenden Abschnitten beschrieben.

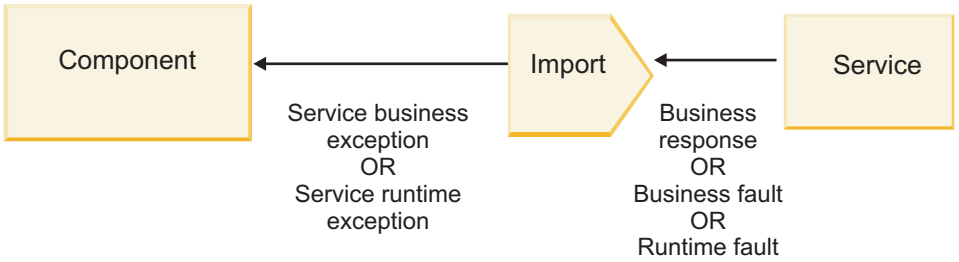


Abbildung 10. Verfahren beim Senden von Fehlerinformationen vom Service durch den Import an die Komponente

Zur Verarbeitung von Fehlern können Sie einen benutzerdefinierten Datenhandler oder eine benutzerdefinierte Datenbindung erstellen.

Fehlerselektoren

Beim Konfigurieren einer Importbindung können Sie einen Fehlerselektor angeben. Der Fehlerselektor ermittelt, ob es sich bei der Importantwort um eine echte Antwort, um eine Geschäftsausnahmebedingung oder um einen Laufzeitfehler handelt. Außerdem ermittelt er aus dem Antworthauptteil oder -header den nativen Fehlernamen, der durch die Bindungskonfiguration zu dem Namen eines Fehlers in der zugehörigen Schnittstelle zugeordnet wird.

Es gibt zwei Typen von vordefinierten Fehlerselektoren, die bei JMS-, MQ-JMS-, generischen JMS-, WebSphere MQ- und HTTP-Importen verwendet werden können:

Tabelle 30. Vordefinierte Fehlerselektoren

Fehlerselektortyp	Beschreibung
Auf Header basierend	Ermittelt anhand der Header in der eingehenden Antwortnachricht, ob es sich bei einer Antwortnachricht um eine Geschäftsausnahmebedingung, eine Laufzeitausnahmebedingung oder um eine normale Nachricht handelt.
SOAP	Ermittelt, ob die SOAP-Antwortnachricht eine normale Antwort, eine Geschäftsausnahmebedingung oder eine Laufzeitausnahmebedingung ist.

Die folgenden Beispiele veranschaulichen auf Headern basierende Fehlerselektoren und SOAP-Fehlerselektoren.

- Auf Header basierender Fehlerselektor

  Falls eine Anwendung angeben will, dass es sich bei der eingehenden Nachricht um eine Geschäftsausnahmebedingung handelt, muss die eingehende Nachricht - wie im Folgenden gezeigt - zwei Header für Geschäftsausnahmebedingungen enthalten:

  ```
  Header name = FaultType, Header value = Business
  Header name = FaultName, Header value = <benutzerdefinierter_nativer_fehlername>
  ```

  Falls eine Anwendung angeben will, dass es sich bei der eingehenden Antwortnachricht um eine Laufzeitausnahmebedingung handelt, muss die eingehende Nachricht - wie im Folgenden gezeigt - einen Header enthalten:

  ```
  Header name = FaultType, Header value = Runtime
  ```

- SOAP-Fehlerselektor

  Eine Geschäftsausnahmebedingung kann mit dem folgenden angepassten SOAP-Header im Rahmen einer SOAP-Nachricht gesendet werden. 'CustomerAlreadyExists' ist in diesem Fall der Name der Ausnahmebedingung.

  ```
  <ibmSoap:BusinessFaultName
  xmlns:ibmSoap="http://www.ibm.com/soap">CustomerAlreadyExists
  <ibmSoap:BusinessFaultName>
  ```

Der Fehlerselektor ist optional. Falls Sie keinen Fehlerselektor angeben, kann die Importbindung den Typ der Antwort nicht ermitteln. Die Bindung behandelt die Antwort daher als Geschäftsantwort und ruft den Antwortdatenhandler oder die Antwortdatenbindung auf.

Sie können einen benutzerdefinierten Fehlerselektor erstellen. Die Schritte zum Erstellen eines benutzerdefinierten Fehlerselektors sind im Thema über die Entwicklung eines angepassten oder benutzerdefinierten Fehlerselektors im Information Center von IBM Integration Designer beschrieben.

Geschäftsausnahmebedingungen

Eine Geschäftsausnahmebedingung kann auftreten, wenn bei der Verarbeitung einer Anforderung ein Fehler auftritt. Beispiel: Falls Sie eine Anforderung zum Erstellen eines Kunden senden und dieser Kunde bereits vorhanden ist, sendet der Service eine Geschäftsausnahmebedingung an die Importbindung.

Nachdem die Bindung eine Geschäftsausnahmebedingung empfangen hat, richten sich die Verarbeitungsschritte danach, ob für die Bindung ein Fehlerselektor definiert wurde.

- Falls kein Fehlerselektor definiert wurde, ruft die Bindung den Antwortdatenhandler oder die Antwortdatenbindung auf.
- Falls ein Fehlerselektor definiert wurde, findet die folgende Verarbeitung statt:
  1. Die Importbindung ruft den Fehlerselektor auf, um zu ermitteln, ob es sich bei der Antwort um eine Geschäftsausnahmebedingung, eine Geschäftsantwort oder einen Laufzeitfehler handelt.
  2. Ist die Antwort eine Geschäftsausnahmebedingung, ruft die Importbindung den Fehlerselektor auf, damit dieser den nativen Fehlernamen bereitstellt.
  3. Die Importbindung ermittelt anhand des nativen Fehlernamens, der vom Fehlerselektor zurückgegeben wurde, den WSDL-Fehler.
  4. Die Importbindung ermittelt den Fehlerdatenhandler, der für diesen WSDL-Fehler konfiguriert ist.
  5. Die Importbindung ruft diesen Fehlerdatenhandler mit den Fehlerdaten auf.
  6. Der Fehlerdatenhandler transformiert die Fehlerdaten in ein Datenobjekt und gibt dies an die Importbindung zurück.
  7. Die Importbindung erstellt ein Geschäftsausnahmebedingungsobjekt für den Service mit dem Datenobjekt und dem Fehlernamen.

8. Der Import gibt das Geschäftsausnahmebedingungsobjekt für den Service an die Komponente zurück.

Laufzeitausnahmebedingungen

Eine Laufzeitausnahmebedingung kann auftreten, wenn bei der Kommunikation mit dem Service ein Problem vorliegt. Die Verarbeitung einer Laufzeitausnahmebedingung erfolgt ähnlich wie bei einer Geschäftsausnahmebedingung. Falls ein Fehlerselektor definiert wurde, findet die folgende Verarbeitung statt:

1. Die Importbindung ruft den geeigneten Datenhandler für Laufzeitausnahmebedingungen mit den Ausnahmedaten auf.
2. Der Datenhandler für Laufzeitausnahmebedingungen transformiert die Ausnahmedaten in ein Laufzeitausnahmebedingungsobjekt für den Service und gibt dieses an die Importbindung zurück.
3. Der Import gibt das Laufzeitausnahmebedingungsobjekt für den Service an die Komponente zurück.

Interoperabilität zwischen SCA-Modulen und Open SCA-Services

IBM WebSphere Application Server V7.0 Feature Pack for Service Component Architecture (SCA) stellt ein einfaches, aber leistungsfähiges Programmiermodell zur Verfügung, mit dem Anwendungen basierend auf den Spezifikationen von 'Open SCA' erstellt werden können. Die SCA-Module von IBM Business Process Manager verwenden Import- und Exportbindungen für die Interaktion mit Open SCA-Services, die in einer Rational Application Developer-Umgebung entwickelt wurden und von WebSphere Application Server Feature Pack for Service Component Architecture per Hosting bereitgestellt werden.

Eine SCA-Anwendung ruft eine Open SCA-Anwendung mittels einer Importbindung auf. Eine SCA-Anwendung empfängt einen Aufruf von einer Open SCA-Anwendung über eine Exportbindung. Eine Liste der unterstützten Bindungen ist unter „Services über Bindungen für Interoperabilität aufrufen" auf Seite 73 aufgeführt.

Open SCA-Services aus SCA-Modulen aufrufen

Mit IBM Integration Designer entwickelte SCA-Anwendungen können Open SCA-Anwendungen aufrufen, die in einer Rational Application Developer-Umgebung entwickelt wurden. Dieser Abschnitt enthält ein Beispiel für den Aufruf eines Open SCA-Service aus einem SCA-Modul, für den eine SCA-Importbindung verwendet wird.

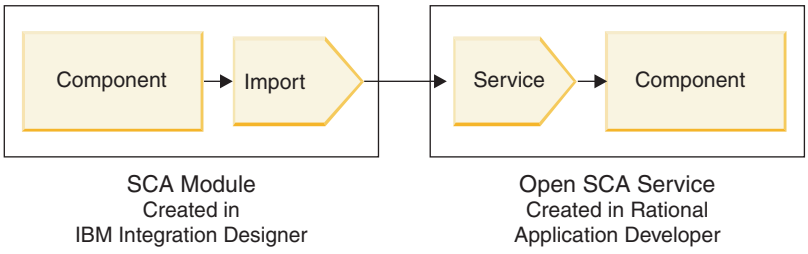


Abbildung 11. Komponente in einem SCA-Modul, die einen Open SCA-Service aufruft

Für den Aufruf eines Open SCA-Service ist keine spezielle Konfiguration erforderlich.

Um über eine SCA-Importbindung eine Verbindung zu einem Open SCA-Service herzustellen, geben Sie den Komponentennamen und den Servicenamen des Open SCA-Service in der Importbindung an.

1. Den Namen der Zielkomponente und des Zielservice im Open SCA-Verbund können Sie folgendermaßen abrufen:
   a. Stellen Sie sicher, dass die Registerkarte **Eigenschaften** geöffnet ist, indem Sie auf **Fenster** > **Sicht anzeigen** > **Eigenschaften** klicken.

b. Öffnen Sie den Verbundeditor, indem Sie doppelt auf das Verbunddiagramm klicken, das die Komponente und den Service enthält. Bei einer Komponente namens **customer** heißt das Verbunddiagramm beispielsweise **customer.composite_diagram**.
c. Klicken Sie auf die Zielkomponente.
d. Notieren Sie sich den Namen der Zielkomponente, der im Feld **Name** der Registerkarte **Eigenschaften** angegeben ist.
e. Klicken Sie auf das Servicesymbol, das der Komponente zugeordnet ist.
f. Notieren Sie sich den Namen des Service, der im Feld **Name** der Registerkarte **Eigenschaften** angegeben ist.

2. Gehen Sie folgendermaßen vor, um den IBM Business Process Manager-Import so zu konfigurieren, dass eine Verbindung zum Open SCA-Service hergestellt wird:
a. Navigieren Sie in IBM Integration Designer zur Registerkarte **Eigenschaften** des SCA-Imports, den Sie mit dem Open SCA-Service verbinden wollen.
b. Geben Sie im Feld **Modulname** den Komponentennamen ein, den Sie in 1d ermittelt haben.
c. Geben Sie im Feld **Exportname** den Servicenamen ein, den Sie in Schritt 1f ermittelt haben.
d. Speichern Sie Ihre Arbeit durch Drücken der Tastenkombination Strg+S.

SCA-Module aus Open SCA-Services aufrufen

Open SCA-Anwendungen, die in einer Rational Application Developer-Umgebung entwickelt wurden, können mit IBM Integration Designer entwickelte SCA-Anwendungen aufrufen. Dieser Abschnitt enthält ein Beispiel für den Aufruf eines SCA-Moduls aus einem Open SCA-Service, für den eine SCA-Exportbindung verwendet wird.

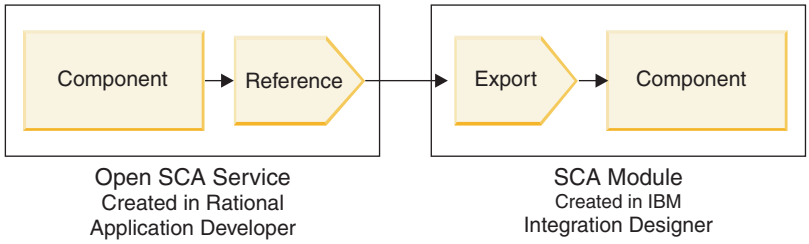


Abbildung 12. Open SCA-Service, der eine Komponente im SCA-Modul aufruft

Um eine Verbindung zu einer SCA-Komponente durch eine Open SCA-Referenzbindung herstellen zu können, müssen Sie den Modulnamen und den Exportnamen angeben.

1. Gehen Sie folgendermaßen vor, um den Namen des Zielmoduls und des Exports abzurufen:
a. Öffnen Sie in IBM Integration Designer das Modul im Assembly-Editor, indem Sie doppelt auf das Modul klicken.
b. Klicken Sie auf den Export.
c. Notieren Sie sich den Namen des Exports, der im Feld **Name** der Registerkarte **Eigenschaften** angegeben ist.

2. Konfigurieren Sie die Open SCA-Referenz, die eine Verbindung zum IBM Business Process Manager-Modul und -Export herstellen soll:
a. Öffnen Sie in Rational Application Developer den Verbundeditor, indem Sie doppelt auf das Verbunddiagramm klicken, das die Komponente und den Service enthält.
b. Klicken Sie auf das Referenzsymbol der Komponentenreferenz, um die Referenzeigenschaften auf der Registerkarte mit den **Eigenschaften** anzuzeigen.
c. Klicken Sie links auf der Seite auf die Registerkarte für die **Bindung**.
d. Klicken Sie auf **Bindungen** (oder 'Bindings') und dann auf **Hinzufügen**.

e. Wählen Sie die **SCA-Bindung** aus.
f. Geben Sie im Feld **URI** den Namen des IBM Business Process Manager-Moduls gefolgt von einem Schrägstrich ('/') und gefolgt vom Exportnamen ein (den Exportnamen haben Sie in Schritt 1c auf Seite 72 ermittelt).
g. Klicken Sie auf **OK**.
h. Speichern Sie Ihre Arbeit durch Drücken der Tastenkombination Strg+S.

Services über Bindungen für Interoperabilität aufrufen

Die folgenden Bindungen werden unterstützt, um eine Interoperabilität mit einem Open SCA-Service bereitzustellen.

- SCA-Bindung

  In IBM Business Process Manager werden die folgenden Aufrufstile unterstützt, wenn ein SCA-Modul einen Open SCA-Service über eine SCA-Importbindung aufruft:

  – Asynchron (unidirektional)
  – Synchron (Anforderung/Antwort)

  Die SCA-Importschnittstelle und die Open SCA-Serviceschnittstelle müssen eine WSDL-Schnittstelle verwenden, die mit WS-I (Web services interoperability) konform ist.

  Bitte beachten Sie, dass die SCA-Bindung die Weitergabe des Transaktions- und Sicherheitskontextes unterstützt.
- Web-Service-Bindung (JAX-WS) entweder mit dem SOAP1.1/HTTP- oder mit dem SOAP1.2/HTTP-Protokoll

  Die SCA-Importschnittstelle und die Open SCA-Serviceschnittstelle müssen eine WSDL-Schnittstelle verwenden, die mit WS-I (Web services interoperability) konform ist.

  Außerdem müssen die folgenden Servicequalitäten unterstützt werden:

  – Atomare Transaktion für Web-Services
  – Sicherheit für Web-Services
- EJB-Bindung

  Die Interaktion zwischen einem SCA-Modul und einem Open SCA-Service wird bei Verwendung einer EJB-Bindung mit einer Java-Schnittstelle definiert.

  Bitte beachten Sie, dass die EJB-Bindung die Weitergabe des Transaktions- und Sicherheitskontextes unterstützt.
- JMS-Bindungen

  Die SCA-Importschnittstelle und die Open SCA-Serviceschnittstelle müssen eine WSDL-Schnittstelle verwenden, die mit WS-I (Web services interoperability) konform ist.

  Die folgenden JMS-Provider werden unterstützt:

  – WebSphere Platform Messaging (JMS-Bindung)
  – WebSphere MQ (MQ-JMS-Bindung)

Anmerkung: Geschäftsgrafiken sind über SCA-Bindungen hinweg nicht interoperabel und werden daher in Schnittstellen, die für die Interoperabilität mit WebSphere Application Server Feature Pack for Service Component Architecture verwendet werden, nicht unterstützt.

Bindungstypen

Sie verwenden protokollspezifische *Bindungen* bei Importen und Exporten, um anzugeben, mit welchem Transportmittel Daten in ein Modul oder aus einem Modul heraus transportiert werden.

Geeignete Bindungen auswählen:

Bei der Erstellung einer Anwendung müssen Sie wissen, wie Sie das Binding auswählen können, das den Anforderungen Ihrer Anwendung am besten entspricht.

Die Palette der in IBM Integration Designer verfügbaren Bindings bieten eine große Bandbreite an Auswahlmöglichkeiten. Anhand dieser Liste können Sie erkennen, welcher Bindingtyp den Anforderungen der Anwendung am besten entspricht.

Verwenden Sie eine *SCA-Bindung*, wenn die folgenden Faktoren gegeben sind:

- Alle Services sind in Modulen enthalten. Es gibt keine externen Services.
- Die Funktionsweise soll auf verschiedene SCA-Module verteilt werden, die direkt miteinander interagieren.
- Die Module sind eng verbunden.

Verwenden Sie eine *Web-Service-Bindung*, wenn die folgenden Faktoren gegeben sind:

- Sie müssen über das Internet auf einen externen Provider zugreifen oder einen Service über das Internet bereitstellen.
- Die Services sind flexibel verbunden.
- Die synchrone Übertragung wird bevorzugt; dies bedeutet, dass eine Anforderung von einem Service auf eine Antwort von einem anderen Service warten kann.
- Die externen Services, auf die Sie zugreifen wollen, oder der Service, den Sie bereitstellen wollen, verwenden das Protokoll 'SOAP/HTTP' oder 'SOAP/JMS'.

Verwenden Sie eine *HTTP-Bindung*, wenn die folgenden Faktoren gegeben sind:

- Sie müssen auf einen externen Service über das Internet zugreifen oder einen Service über das Internet bereitstellen, und Sie arbeiten mit anderen Web-Services wie beispielsweise GET, PUT oder DELETE.
- Die Services sind flexibel verbunden.
- Die synchrone Übertragung wird bevorzugt; dies bedeutet, dass eine Anforderung von einem Service auf eine Antwort von einem anderen Service warten kann.

Verwenden Sie eine *EJB-Bindung*, wenn die folgenden Faktoren gegeben sind:

- Die Bindung ist für einen importierten Service gedacht, der entweder selbst eine EJB ist oder auf den von EJB-Clients zugegriffen werden muss.
- Der importierte Service ist flexibel verbunden.
- Interaktionen mit EJBs des Typs 'Stateful' sind nicht erforderlich.
- Die synchrone Übertragung wird bevorzugt; dies bedeutet, dass eine Anforderung von einem Service auf eine Antwort von einem anderen Service warten kann.

Verwenden Sie eine *EIS-Bindung*, wenn die folgenden Faktoren gegeben sind:

- Sie müssen unter Verwendung eines Ressourcenadapters auf einen Service auf einem EIS-System zugreifen.
- Die synchrone Datenübertragung wird der asynchronen Übertragung vorgezogen.

Verwenden Sie eine *JMS-Bindung*, wenn die folgenden Faktoren gegeben sind:

Wichtig: Es gibt mehrere Typen von JMS-Bindungen. Falls Sie voraussichtlich SOAP-Nachrichten mit JMS austauschen werden, kann die Web-Service-Bindung mit dem Protokoll 'SOAP/JMS' eine gute Wahl sein. Weitere Informationen hierzu finden Sie unter „Web-Service-Bindungen” auf Seite 75.

- Sie benötigen ein Messaging-System.
- Die Services sind flexibel verbunden.
- Die asynchrone Datenübertragung wird der synchronen Übertragung vorgezogen.

Verwenden Sie eine *JMS-Bindung*, wenn die folgenden Faktoren gegeben sind:

- Sie müssen auf ein Messaging-System eines anderen Herstellers als IBM zugreifen.
- Die Services sind flexibel verbunden.

- Die Zuverlässigkeit ist wichtiger als die Leistung; das heißt, die asynchrone Datenübertragung ist der synchronen vorzuziehen.

Verwenden Sie eine *MQ-Bindung*, wenn die folgenden Faktoren gegeben sind:
- Sie müssen auf ein WebSphere MQ-Messaging-System zugreifen und die nativen MQ-Funktionen verwenden.
- Die Services sind flexibel verbunden.
- Die Zuverlässigkeit ist wichtiger als die Leistung; das heißt, die asynchrone Datenübertragung ist der synchronen vorzuziehen.

Verwenden Sie eine *MQ-JMS-Bindung*, wenn die folgenden Faktoren gegeben sind:
- Sie müssen auf ein WebSphere MQ-Messaging-System zugreifen, können dies jedoch auch aus einem JMS-Kontext heraus ausführen, die JMS-Funktionsuntergruppe ist also für Ihre Anwendung ausreichend.
- Die Services sind flexibel verbunden.
- Die Zuverlässigkeit ist wichtiger als die Leistung; das heißt, die asynchrone Datenübertragung ist der synchronen vorzuziehen.

SCA-Bindungen:

Eine SCA-Bindung ermöglicht einem Service die Kommunikation mit anderen Services in anderen Modulen. Über einen Import mit einer SCA-Bindung können Sie auf einen Service in einem anderen SCA-Modul zugreifen. Ein Export mit einer SCA-Bindung ermöglicht es Ihnen, einen Service anderen Modulen anzubieten.

Verwenden Sie IBM Integration Designer, um SCA-Bindungen für Importe und Exporte in SCA-Modulen zu generieren und konfigurieren.

Falls Module auf demselben Server ausgeführt werden oder in demselben Cluster implementiert sind, ist die Verwendung einer SCA-Bindung die einfachste und schnellste Methode.

Nachdem das Modul, das die SCA-Bindung enthält, auf dem Server implementiert wurde, können Sie in der Administrationskonsole Informationen zur Bindung anzeigen oder bei einer Importbindung ausgewählte Eigenschaften der Bindung ändern.

Web-Service-Bindungen:

Eine Web-Service-Bindung ist ein Verfahren, mit dem Nachrichten von einer SCA-Komponente an einen Web-Service (und umgekehrt) übertragen werden können.

Web-Service-Bindungen - Übersicht:

Eine Web-Service-Import-Bindung ermöglicht Ihnen den Aufruf eines externen Web-Service aus SCA-Komponenten heraus. Mit einer Web-Service-Exportbindung können Sie Ihre SCA-Komponenten als Web-Services für Clients zugänglich machen.

Mit einer Web-Service-Bindung können Sie unter Verwendung von interoperablen SOAP-Nachrichten und Servicequalitäten (QoS) auf einen externen Service zugreifen.

Zum Generieren und Konfigurieren von Web-Service-Bindungen für Importe und Exporte in SCA-Modulen verwenden Sie Integration Designer. Die folgenden Typen von Web-Service-Bindungen sind verfügbar:
- SOAP1.2/HTTP und SOAP1.1/HTTP

Diese Bindungen basieren auf Java API for XML Web Services (JAX-WS), einer Java-Programmierungs-API für die Erstellung von Web-Services.

– Verwenden Sie SOAP1.2/HTTP, falls Ihr Web-Service der Spezifikation von SOAP 1.2 entspricht.
– Verwenden Sie SOAP1.1/HTTP, falls Ihr Web-Service der Spezifikation von SOAP 1.1 entspricht.

Wichtig: Wenn Sie eine Anwendung mit einer Web-Service-Bindung (JAX-WS-Bindung) implementieren, darf für den Zielserver nicht die Option für den **Start von Komponenten nach Bedarf** ausgewählt sein. Details finden Sie unter „Serverkonfiguration prüfen" auf Seite 84.

Wenn Sie eine dieser Bindungen auswählen, können Sie zusammen mit den SOAP-Nachrichten Anhänge senden.

Die Web-Service-Bindungen können bei SOAP-Standardnachrichten verwendet werden. Wenn Sie eine der JAX-WS-Bindungen für Web-Services verwenden, können Sie jedoch anpassen, wie diese SOAP-Nachrichten syntaktisch analysiert oder geschrieben werden. Sie können beispielsweise vom Standard abweichende Element in SOAP-Nachrichten verarbeiten oder eine zusätzliche Verarbeitung auf die SOAP-Nachricht anwenden. Beim Konfigurieren der Bindung geben Sie einen benutzerdefinierten Datenhandler an, der diese Verarbeitung für die SOAP-Nachricht ausführt.

Zusammen mit einer Web-Service-Bindung (JAX-WS) können Sie einen Richtliniensatz verwenden. Ein Richtliniensatz ist eine Sammlung von Richtlinientypen, die jeweils eine Servicequalität bereitstellen. Der Richtliniensatz 'WSAddressing' bietet beispielsweise ein transportneutrales Verfahren für die einheitliche Adressierung von Web-Services und Nachrichten. Zur Auswahl des Richtliniensatzes für die Bindung verwenden Sie Integration Designer.

Anmerkung: Falls Sie einen SAML-Richtliniensatz (SAML = Security Assertion Markup Language) verwenden wollen, müssen Sie einige zusätzliche Konfigurationsschritte ausführen, die unter „SAML-Richtliniensätze importieren" auf Seite 82 beschrieben sind.

- SOAP1.1/HTTP

  Verwenden Sie diese Bindung, wenn Sie Web-Services erstellen wollen, die eine SOAP-codierte Nachricht verwenden, die auf Java API for XML-based RPC (JAX-RPC) basiert.
- SOAP1.1/JMS

  Verwenden Sie diese Bindung, um SOAP-Nachrichten mit einem JMS-Ziel (JMS = Java Message Service) zu senden oder zu empfangen.

Unabhängig vom Transportprotokoll (HTTP oder JMS), mit dem die SOAP-Nachricht übermittelt wird, verarbeiten Web-Service-Bindungen Anforderungs-/Antwortinteraktionen immer synchron. Der Thread, der den Aufruf des Service-Providers ausführt, wird bis zum Empfang einer Antwort vom Provider geblockt. Weitere Informationen zur diesem Aufrufstil finden Sie im Thema über den synchronen Aufruf.

Wichtig: Die folgenden Kombinationen von Web-Service-Bindungen können für Exporte in demselben Modul nicht verwendet werden. Falls Sie Komponenten zugänglich machen müssen, die mehrere dieser Exportbindungen verwenden, müssen Sie für jede Komponente ein separates Modul verwenden und diese Module dann über eine SCA-Bindung mit Ihren Komponenten verbinden:

- SOAP 1.1/JMS und SOAP 1.1/HTTP mit JAX-RPC
- SOAP 1.1/HTTP mit JAX-RPC und SOAP 1.1/HTTP mit JAX-WS
- SOAP 1.1/HTTP mit JAX-RPC und SOAP 1.2/HTTP mit JAX-WS

Nachdem das SCA-Modul, das die Web-Service-Bindung enthält, auf dem Server implementiert wurde, können Sie in der Administrationskonsole die Informationen zur Bindung anzeigen oder ausgewählte Eigenschaften der Bindung ändern.

Anmerkung: Mithilfe von Web-Services können Anwendungen zusammenarbeiten, indem Standardbeschreibungen von Services und Standardformate für die ausgetauschten Nachrichten verwendet werden. Die Web-Service-Importbindungen und -Exportbindungen können beispielsweise mit Service interagieren,

die unter Verwendung von Web Services Enhancements (WSE) Version 3.5 und Windows Communication Foundation (WCF) Version 3.5 für Microsoft .NET implementiert wurden. Bei der Interaktion mit solchen Services muss Folgendes sichergestellt sein:

- Die WSDL-Datei, die für den Zugriff auf einen Web-Service verwendet wird, enthält für jede Operation in der Schnittstelle einen nicht leeren SOAP-Aktionswert.
- Der Web-Service-Client legt entweder den Header 'SOAPAction' oder den Header 'wsa:Action' fest, wenn Nachrichten an einen Web-Service-Export gesendet werden.

SOAP-Headerweitergabe:

Bei der Verarbeitung von SOAP-Nachrichten müssen Sie möglicherweise auf Informationen aus bestimmten SOAP-Headern in empfangenen Nachrichten zugreifen oder sicherstellen, dass Nachrichten mit SOAP-Headern mit bestimmten Werten gesendet werden, oder die Übergabe von SOAP-Headern über ein Modul zulassen.

Wenn Sie in Integration Designer eine Web-Service-Bindung konfigurieren, können Sie angeben, dass SOAP-Header weitergegeben werden sollen.

- Sobald Anforderungen bei einem Export oder Antworten bei einem Import empfangen werden, ist der Zugriff auf die SOAP-Headerinformationen möglich. Hierdurch können die Headerwerte für die Logik im Modul zugrunde gelegt werden und die Änderung dieser Header kann ermöglicht werden.
- Wenn Anforderungen von einem Export oder Antworten von einem Import aus gesendet werden, können in die entsprechenden Nachrichten SOAP-Header eingeschlossen werden.

Das Format und das Vorhandensein der weitergegebenen SOAP-Header kann durch Richtliniensätze beeinflusst werden, die für den Import oder Export konfiguriert sind (siehe Tabelle 31 auf Seite 78).

Um die Weitergabe von SOAP-Headern für einen Import oder Export zu konfigurieren, wählen Sie (in der Sicht 'Eigenschaften' von Integration Designer) die Registerkarte **Protokollheader weitergeben** und dann die benötigten Optionen aus.

Header 'WS-Addressing'

Der Header 'WS-Addressing' kann durch die Web-Service-Bindung (JAX-WS) weitergegeben werden.

Beim Weitergeben des Headers 'WS-Addressing' müssen Sie die folgenden Informationen beachten:

- Falls Sie die Weitergabe für den Header 'WS-Addressing' aktivieren, wird der Header unter den folgenden Umständen in das Modul weitergegeben:
  - Anforderungen werden bei einem Export empfangen.
  - Antworten werden bei einem Import empfangen.
- Der Header 'WS-Addressing' wird nicht in abgehende Nachrichten von IBM Business Process Manager weitergegeben (dies bedeutet, dass der Header nicht weitergegeben wird, wenn Anforderungen von einem Import oder Antworten vom Export aus gesendet werden).

Header 'WS-Security'

Der Header 'WS-Security' kann sowohl durch die Web-Service-Bindung (JAX-WS) als auch durch die Web-Service-Bindung (JAX-RPC) weitergegeben werden.

Die Spezifikation von 'WS-Security' für Web-Services beschreibt Erweiterungen für die SOAP-Nachrichtentübertragung, die ein Datenschutzniveau mittels Nachrichtenintegrität, Nachrichtenvertraulichkeit und Einzelnachrichtenauthentifizierung bereitstellen. Mit diesen Mechanismen kann eine Vielzahl von Sicherheitsmodellen und Verschlüsselungstechnologien einbezogen werden.

Beim Weitergeben des Headers WS-Security müssen Sie die folgenden Informationen beachten:

- Falls Sie die Weitergabe für den Header 'WS-Security' aktivieren, wird der Header unter den folgenden Umständen über das Modul weitergegeben:
  - Anforderungen werden bei einem Export empfangen.
  - Anforderungen werden von einem Import gesendet.
  - Antworten werden bei einem Import empfangen.
- Der Header wird standardmäßig *nicht* weitergegeben, wenn Antworten vom Export aus gesendet werden. Falls Sie jedoch für die JVM-Eigenschaft **WSSECURITY.ECHO.ENABLED** die Einstellung **true** festlegen, wird der Header weitergegeben, wenn Antworten vom Export aus gesendet werden. In diesem Fall werden Header 'WS-Security' möglicherweise automatisch aus Anforderungen in Antworten zurückgemeldet, falls der Header 'WS-Security' im Anforderungspfad nicht geändert wird.
- Das genaue Format der SOAP-Nachricht, die von einem Import für eine Anforderung oder von einem Export für eine Antwort gesendet wird, stimmt möglicherweise nicht präzise mit der ursprünglich empfangenen SOAP-Nachricht überein. Aus diesem Grund sollte davon ausgegangen werden, dass digitale Signaturen ungültig werden. Falls in gesendeten Nachrichten eine digitale Signatur erforderlich ist, muss sie mit dem entsprechenden Sicherheitsrichtliniensatz erstellt werden. Header 'WS-Security', die in empfangenen Nachrichten mit der digitalen Signatur verbunden sind, sollten im Modul entfernt werden.

Zur Weitergabe des Headers 'WS-Security' müssen Sie das Schema 'WS-Security' mit dem Anwendungsmodul einschließen. Die Vorgehensweise zum Einschließen des Schemas ist unter „Schema 'WS-Security' in ein Anwendungsmodul einschließen" auf Seite 79 beschrieben.

Methode für Weitergabe von Headern

Die Methode, mit der Header weitergegeben werden, richtet sich nach der Sicherheitsrichtlinieneinstellung für die Import- oder Exportbindung (siehe Tabelle 31):

Tabelle 31. Methode für Übergabe von Sicherheitsheadern

	Exportbindung ohne Sicherheitsrichtlinie	**Exportbindung mit Sicherheitsrichtlinie**
Importbindung ohne Sicherheitsrichtlinie	Sicherheitsheader werden unverändert über das Modul übergeben. Sie werden nicht entschlüsselt. Die Header werden abgehend in demselben Format gesendet, in dem sie empfangen wurden. Die digitale Signatur wird möglicherweise ungültig.	Sicherheitsheader werden entschlüsselt und über das Modul übergeben. Hierbei findet eine Prüfung und Authentifizierung der Signatur statt. Die entschlüsselten Header werden abgehend gesendet. Die digitale Signatur wird möglicherweise ungültig.
Importbindung mit Sicherheitsrichtlinie	Sicherheitsheader werden unverändert über das Modul übergeben. Sie werden nicht entschlüsselt. Die Header sollten nicht an den Import weitergegeben werden. Andernfalls tritt aufgrund einer Duplizierung ein Fehler auf.	Sicherheitsheader werden entschlüsselt und über das Modul übergeben. Hierbei findet eine Prüfung und Authentifizierung der Signatur statt. Die Header sollten nicht an den Import weitergegeben werden. Andernfalls tritt aufgrund einer Duplizierung ein Fehler auf.

Konfigurieren Sie entsprechende Richtliniensätze für die Export- und Importbindungen, da der Serviceanforderer hierdurch von Änderungen an der Konfiguration oder den QoS-Anforderungen des Service-Providers isoliert wird. Durch sichtbare SOAP-Standardheader in einem Modul kann dann die Verarbeitung

im Modul (z. B. Protokollierung und Traceverarbeitung) beeinflusst werden. Die Weitergabe von SOAP-Headern über ein Modul von einer empfangenen Nachricht an eine gesendete Nachricht bedeutet, dass die Isolationsvorteile des Moduls verringert werden.

Standardheader wie beispielsweise 'WS-Security' sollten nicht an eine Anforderung an einen Import oder an eine Antwort an einen Export weitergegeben werden, wenn dem Import oder Export ein Richtliniensatz zugeordnet ist, der normalerweise eine Generierung dieser Header bewirken würde. Andernfalls tritt ein Fehler auf, weil die Header doppelt vorhanden sind. Die Header sollten stattdessen explizit entfernt werden oder die Import- bzw. Exportbindung sollte so konfiguriert sein, dass die Weitergabe von Protokollheadern verhindert wird.

Auf SOAP-Header zugreifen

Wenn eine Nachricht, die SOAP-Header enthält, von einem Web-Service-Import oder -Export empfangen wird, werden die Header in den Headerabschnitt des Servicenachrichtenobjekts gestellt. Sie können auf die Headerinformationen zugreifen. Dies ist unter 'Zugriff auf SOAP-Headerdaten im SMO' beschrieben.

Schema 'WS-Security' in ein Anwendungsmodul einschließen

Die folgende Prozedur skizziert die Schritte, mit denen das Schema in das Anwendungsmodul eingeschlossen wird:

- Falls der Computer, auf dem Integration Designer ausgeführt wird, über einen Internetzugang verfügt, führen Sie die folgenden Schritte aus:
  1. Wählen Sie in der Perspektive 'Geschäftsintegration' für Ihr Projekt den Eintrag **Abhängigkeiten** aus.
  2. Erweitern Sie **Vordefinierte Ressourcen** und wählen Sie entweder **WS-Security 1.0-Schemadateien** oder **WS-Security 1.1-Schemadateien** aus, um das Schema in das Modul zu importieren.
  3. Bereinigen Sie das Projekt und erstellen Sie es erneut.
- Falls ein Computer, auf dem Integration Designer ausgeführt wird, nicht über einen Internetzugang verfügt, können Sie das Schema auf einen zweiten Computer mit Internetzugang herunterladen. Anschließend können Sie es auf den Computer kopieren, auf dem Integration Designer ausgeführt wird.
  1. Laden Sie das ferne Schema auf dem Computer mit Internetzugang herunter:
     a. Klicken Sie auf **Datei** > **Importieren** > **Geschäftsintegration** > **WSDL und XSD**.
     b. Wählen Sie **Ferne WSDL- oder XSD-Datei** aus.
     c. Importieren Sie die folgenden Schemas:

        http://www.w3.org/2003/05/soap-envelope/

        http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/xenc-schema.xsd

        http://www.w3.org/TR/xmldsig-core/xmldsig-core-schema.xsd
  2. Kopieren Sie die Schemas auf den Computer ohne Internetzugang.
  3. Importieren Sie das Schema auf dem Computer ohne Internetzugang:
     a. Klicken Sie auf **Datei** > **Importieren** > **Geschäftsintegration** > **WSDL und XSD**.
     b. Wählen Sie **Lokale WSDL-Datei oder XSD-Datei** aus.
  4. Ändern Sie die Schemapositionen für 'oasis-wss-wssecurity_secext-1.1.xsd':
     a. Öffnen Sie das Schema an der Position *arbeitsplatzposition/modulname/*`StandardImportFilesGen/oasis-wss-wssecurity-secext-1.1.xsd`.
     b. Ändern Sie

        ```
        <xs:import namespace='http://www.w3.org/2003/05/soap-envelope'
        schemaLocation='http://www.w3.org/2003/05/soap-envelope/'/>
        ```

        in:

        ```
        <xs:import namespace='http://www.w3.org/2003/05/soap-envelope'
        schemaLocation='../w3/_2003/_05/soap_envelope.xsd'/>
        ```

c. Ändern Sie

```
<xs:import namespace='http://www.w3.org/2001/04/xmlenc#'
schemaLocation='http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/xenc-schema.xsd'/>
```

in:

```
<xs:import namespace='http://www.w3.org/2001/04/xmlenc#'
schemaLocation='../w3/tr/_2002/rec_xmlenc_core_20021210/xenc-schema.xsd'/>
```

5. Ändern Sie die Schemaposition für 'oasis-200401-wss-wssecurity-secext-1.0.xsd':
   a. Öffnen Sie das Schema an der Position *arbeitsplatzposition/modulname/*`StandardImportFilesGen/oasis-200401-wss-wssecurity-secext-1.0.xsd`.
   b. Ändern Sie

```
<xsd:import namespace="http://www.w3.org/2000/09/xmldsig#"
schemaLocation="http://www.w3.org/TR/xmldsig-core/xmldsig-core-schema.xsd"/>
```

in:

```
<xsd:import namespace="http://www.w3.org/2000/09/xmldsig#"
schemaLocation="../w3/tr/_2002/rec_xmldsig_core_20020212/xmldsig-core-schema.xsd"/>
```

6. Bereinigen Sie das Projekt und erstellen Sie es erneut.

Transportheaderweitergabe:

Bei der Verarbeitung von SOAP-Nachrichten müssen Sie möglicherweise auf Informationen aus bestimmten Transportheadern in empfangenen Nachrichten zugreifen oder sicherstellen, dass Nachrichten mit Transportheadern mit bestimmten Werten gesendet werden, oder die Übergabe von Transportheadern über ein Modul zulassen.

Wenn Sie in Integration Designer eine Web-Service-Bindung konfigurieren, können Sie angeben, dass Transportheader weitergegeben werden sollen.

- Sobald Anforderungen bei einem Export oder Antworten bei einem Import empfangen werden, ist der Zugriff auf die Transportheaderinformationen möglich. Hierdurch können die Headerwerte für die Logik im Modul zugrunde gelegt werden und die Änderung dieser Header kann ermöglicht werden.
- Wenn Antworten von einem Export oder Anforderungen von einem Import aus gesendet werden, können in die entsprechenden Nachrichten Transportheader eingeschlossen werden.

Weitergabe von Headern angeben

Gehen Sie folgendermaßen vor, um die Weitergabe von Transportheadern für einen Import oder Export zu konfigurieren:

1. Wählen Sie in der Sicht 'Eigenschaften' von Integration Designer die Optionen **Binding** > **Weitergabe** aus.
2. Legen Sie die Option für die benötigte Transportheaderweitergabe fest.

Anmerkung: Die Transportheaderweitergabe ist standardmäßig inaktiviert und kann nur in einer Laufzeitumgebung mit Version 7.0.0.3 (oder höher) implementiert werden. Bitte beachten Sie außerdem, dass die Transportheaderweitergabe bei Version 7.0.0.3 auf HTTP-Transportheader beschränkt ist.

Falls Sie die Weitergabe von Transportheadern aktivieren, werden die Header über ein Modul aus empfangenen Nachrichten weitergegeben und bei nachfolgenden Aufrufen in demselben Thread verwendet, wenn Sie die Header nicht explizit entfernen.

Anmerkung: Transportheader können bei Verwendung einer Web-Service-Bindung (JAX-RPC) nicht weitergegeben werden.

Auf Headerinformationen zugreifen

Wenn die Transportheaderweitergabe für empfangene Nachrichten aktiviert ist, sind alle Transportheader (auch kundendefinierte Header) im Servicenachrichtenobjekt sichtbar. Sie können für die Header andere Werte angeben oder neue Header erstellen. Bitte beachten Sie jedoch, dass für die von Ihnen festgelegten Werte keine Überprüfung oder Validierung stattfindet. Falsche Header können Laufzeitprobleme für den Web-Service verursachen.

Berücksichtigen Sie beim Festlegen von HTTP-Headern die folgenden Informationen:

- Alle Änderungen an den Headern, die für die Web-Service-Steuerkomponente reserviert sind, werden in der abgehenden Nachricht nicht berücksichtigt. Beispielsweise sind die HTTP-Version oder die HTTP-Methode bzw. die Header 'Content-Type', 'Content-Length' und 'SOAPAction' für die Web-Service-Steuerkomponente reserviert.
- Falls ein Headerwert eine Zahl ist, sollte die Zahl (und nicht die Zeichenfolge) direkt gesendet werden. Verwenden Sie beispielsweise **Max-Forwards = 5** (anstelle von **Max-Forwards = Max-Forwards: 5**) und **Age = 300** (anstelle von **Age = Age: 300**).
- Falls die Anforderungsnachricht kleiner als 32 KB ist, entfernt die Web-Service-Steuerkomponente den Header 'Transfer-Encoding' und setzt den Header 'Content-Length' auf die feste Größe der Nachricht.
- Der Header 'Content-language' wird im Antwortpfad durch 'WAS.channel.http' zurückgesetzt.
- Eine ungültige Einstellung für 'Upgrade' führt zu einem Fehler 500.
- Die folgenden Header hängen den Wert, der durch die Web-Service-Steuerkomponente reserviert ist, an die Kundeneinstellungen an:
  - User-Agent
  - Cache-Control
  - Pragma
  - Accept
  - Connection

Zum Zugriff auf die Headerinformationen können Sie eines der folgenden Verfahren verwenden:

- Zugriff auf die Strukturen des Servicenachrichtenobjekts mit einem Mediationsbasiselement

  Nach Auswahl der Links zu den Referenzinformationen erhalten Sie Informationen zur Verwendung von Mediationsbasiselementen.
- Verwendung der Kontextservice-SPI

  Der folgende Beispielcode liest die HTTP-Transportheader aus dem Kontextservice:

```
HeadersType headerType = ContextService.INSTANCE.getHeaders();
HTTPHeaderType httpHeaderType = headerType.getHTTPHeader();
List HTTPHeader  httpHeaders = httpHeaderType.getHeader();
if(httpHeaders!=null){
 for(HTTPHeader httpHeader: httpHeaders){
   String httpHeadername = httpHeader.getName();
   String httpHeaderValue = httpHeader.getValue();
     }
}
List PropertyType  properties = headerType.getProperties();
if(properties!=null){
  for(PropertyType property: properties){
   String propertyName = property.getName();
   String propertyValue = property.getValue().toString();
     }
}
```

Fehlerbehebung

Falls beim Senden von überarbeiteten Headern Probleme auftreten, können Sie die TCP/IP-Nachrichten mithilfe von Tools wie der TCP/IP-Überwachung in Integration Designer abfangen. Auf die TCP/IP-Überwachung greifen Sie zu, indem Sie auf der Seite 'Benutzervorgaben' die Optionen **Ausführen/Debug** > **TCP/IP-Überwachung** auswählen.

Zum Anzeigen der Headerwerte können Sie auch den JAX-WS-Steuerkomponententrace verwenden: **org.apache.axis2.*=all: com.ibm.ws.websvcs.*=all:**

Mit Web-Service-Bindungen (JAX-WS) arbeiten:

Wenn Sie in Ihren Anwendungen Web-Service-Bindungen (JAX-WS) verwenden, können Sie eine SAML-Servicequalität (SAML = Security Assertion Markup Language) zur Bindung hinzufügen. Zunächst müssen Sie mit der Administrationskonsole den Richtliniensatz importieren. In der Administrationskonsole können Sie zudem sicherstellen, dass der Server ordnungsgemäß für die Verwendung der Web-Service-Bindung (JAX-WS) konfiguriert ist.

SAML-Richtliniensätze importieren:

Die Security Assertion Markup Language (Kurzform: SAML) ist ein XML-basierter OASIS-Standard für den Austausch von Informationen zu Benutzeridentitäts- und Sicherheitsattributen. Wenn Sie in Integration Designer eine Web-Service-Bindung (JAX-WS) konfigurieren, können Sie einen SAML-Richtliniensatz angeben. Zuerst machen Sie die SAML-Richtliniensätze mit der Administrationskonsole von IBM Business Process Manager verfügbar, sodass sie in Integration Designer importiert werden können.

Die SAML-Richtliniensätze befinden sich normalerweise im Verzeichnis für die Profilkonfiguration:

profilstammverzeichnis/config/templates/PolicySets

Bevor Sie diese Prozedur starten, müssen Sie sicherstellen, dass sich die folgenden Verzeichnisse, die die Richtliniensätze enthalten, im Verzeichnis für die Profilkonfiguration befinden:

- SAML11 Bearer WSHTTPS default
- SAML20 Bearer WSHTTPS default
- SAML11 Bearer WSSecurity default
- SAML20 Bearer WSSecurity default
- SAML11 HoK Public WSSecurity default
- SAML20 HoK Public WSSecurity default
- SAML11 HoK Symmetric WSSecurity default
- SAML20 HoK Symmetric WSSecurity default
- Benutzername WSHTTPS default

Wenn sich die Verzeichnisse nicht im Verzeichnis für die Profilkonfiguration befinden, kopieren Sie sie von der folgenden Position in dieses Verzeichnis:

Stammverzeichnis des Anwendungsservers/profileTemplates/default/documents/config/templates/PolicySets

Sie importieren die Richtliniensätze in die Administrationskonsole, wählen anschließend diejenigen Richtliniensätze aus, die für Integration Designer verfügbar gemacht werden sollen, und speichern dann für jeden dieser Richtliniensätze eine komprimierte Datei an einer Position, auf die von Integration Designer aus zugegriffen werden kann.

1. Importieren Sie die Richtliniensätze, indem Sie die folgenden Schritte ausführen:

a. Klicken Sie in der Administrationskonsole auf **Services** > **Richtliniensätze** > **Anwendungsrichtliniensätze**.
b. Klicken Sie auf **Importieren** > **Aus Standard-Repository**.
c. Wählen Sie die standardmäßigen SAML-Richtliniensätze aus und klicken Sie auf **OK**.

2. Exportieren Sie die Richtliniensätze, sodass sie von Integration Designer verwendet werden können:
   a. Wählen Sie auf der Seite für Anwendungsrichtliniensätze denjenigen SAML-Richtliniensatz aus, der exportiert werden soll, und klicken Sie auf **Exportieren**.

      Anmerkung: Wenn die Seite für Anwendungsrichtliniensätze gerade nicht angezeigt wird, klicken Sie in der Administrationskonsole auf **Services** > **Richtliniensätze** > **Anwendungsrichtliniensätze**.
   b. Klicken Sie auf der nächsten Seite für den Richtliniensatz auf den Link für die komprimierte Datei (ZIP).
   c. Klicken Sie im Fenster für den Dateidownload auf **Speichern** und geben Sie dann eine Position an, auf die von Integration Designer zugegriffen werden kann.
   d. Klicken Sie auf **Zurück**.
   e. Führen Sie die Schritte 2a bis 2d für alle Richtliniensätze aus, die exportiert werden sollen.

Die SAML-Richtliniensätze sind in komprimierten Dateien (ZIP) gespeichert und können nun in Integration Designer importiert werden.

Importieren Sie die Richtliniensätze in Integration Designer wie in „Richtliniensätze" beschrieben.

Web-Services aufrufen, die eine HTTP-Basisauthentifizierung erfordern:

Die HTTP-Basisauthentifizierung verwendet einen Benutzernamen und ein Kennwort für die Authentifizierung eines Service-Clients bei einem sicheren Endpunkt. Sie können die HTTP-Basisauthentifizierung beim Senden oder Empfangen von Web-Service-Anforderungen einrichten.

Sie richten die HTTP-Basisauthentifizierung zum Empfangen von Web-Service-Anforderungen ein, indem Sie die Exportbindung für die Java API for XML Web Services (JAX-WS) wie in Sicherheitsaufgabenbereiche erstellen und zu Web-Service-Exporten zuordnen erläutert konfigurieren.

Die HTTP-Basisauthentifizierung kann für Web-Service-Anforderungen, die von einer JAX-WS-Importbindung gesendet werden, auf eine von insgesamt zwei Arten aktiviert werden:

- Beim Konfigurieren der Importbindung in einem SCA-Modul können Sie den bereitgestellten Richtliniensatz für die HTTP-Authentifizierung namens 'BPMHTTPBasicAuthentication' auswählen, der mit der Web-Service-Importbindung (JAX-WS) zur Verfügung gestellt wird, oder Sie können einen beliebigen anderen Richtliniensatz auswählen, der die Richtlinie 'HTTPTransport' enthält.
- Beim Erstellen des SCA-Moduls können Sie die Funktionalität des Mediationsablaufs nutzen, um dynamisch einen neuen HTTP-Authentifizierungsheader zu erstellen und im Header den Benutzernamen und das Kennwort anzugeben.

Anmerkung: Der Richtliniensatz hat Vorrang vor dem im Header angegebenen Wert. Wenn zur Laufzeit der im HTTP-Authentifizierungsheader festgelegte Wert verwendet werden soll, hängen Sie keinen Richtliniensatz an, der die Richtlinie 'HTTPTransport' beinhaltet. Insbesondere müssen Sie auf die Verwendung des standardmäßigen Richtliniensatzes 'BPMHTTPBasicAuthentication' verzichten und sicherstellen, dass der definierte Richtliniensatz (sofern ein solcher vorhanden ist) die Richtlinie 'HTTPTransport' keinesfalls enthält.

Weitere Informationen zu Richtliniensätzen für Web-Services und Richtlinienbindungen sowie ihrer Verwendung finden Sie im Abschnitt über Richtliniensätze für Web-Services im WebSphere Application Server Information Center.

- Führen Sie die folgenden Schritte aus, um den bereitgestellten Richtliniensatz zu verwenden:

1. Optional: Erstellen Sie in der Administrationskonsole eine allgemeine Richtlinienbindung für den Client oder bearbeiten Sie eine bereits vorhandene Bindung, die die Richtlinie 'HTTPTransport' mit den erforderlichen Werten für Benutzer-ID und Kennwort enthält.
2. Generieren Sie in IBM Integration Designer eine Web-Service-Importbindung (JAX-WS) und hängen Sie den Richtliniensatz 'BPMHTTPBasicAuthentication' an.
3. Führen Sie *einen* der folgenden Schritte aus:
   – Geben Sie in IBM Integration Designer in den Eigenschaften für Web-Service-Importbindungen (JAX-WS) den Namen einer vorhandenen allgemeinen Richtlinienbindung für Clients an, die die Richtlinie 'HTTPTransport' einschließt.
   – Wählen Sie nach der Implementierung des SCA-Moduls in der Administrationskonsole entweder eine vorhandene Clientrichtlinienbindung aus oder erstellen Sie eine neue Clientrichtlinienbindung und ordnen Sie diese anschließend der Importbindung zu.
4. Optional: Bearbeiten Sie in der Administrationskonsole des Process Server die ausgewählte Richtliniensatzbindung, um die erforderliche ID und das erforderliche Kennwort anzugeben.

- Führen Sie eine der folgenden Gruppen von Schritten aus, um den Benutzernamen und das Kennwort im HTTP-Authentifizierungsheader anzugeben:
  – Erstellen Sie den HTTP-Authentifizierungsheader unter Verwendung des Mediationsbasiselements 'HTTP-Header-Setter' ('HTTP Header Setter' in IBM Integration Designer und geben Sie den Benutzernamen und das Kennwort an.
  – Falls zusätzliche Logik erforderlich ist, verwenden Sie Java-Code in einem angepassten Mediationsbasiselement, wie im folgenden Beispiel gezeigt, um Folgendes zu erzielen:
    1. Erstellen eines HTTP-Authentifizierungsheaders
    2. Angeben der Informationen für Benutzername und Kennwort
    3. Hinzufügen des neuen HTTP-Authentifizierungsheaders zu 'HTTPControl'
    4. Einsetzen des aktualisierten Elements 'HTTPControl' zurück in den Kontextservice

```
//Abrufen von 'HeaderInfoType' aus 'contextService'
ContextService contextService = (ContextService) ServiceManager.INSTANCE
.locateService("com/ibm/bpm/context/ContextService");
HeaderInfoType headers = contextService.getHeaderInfo();
if(headers == null){
    headers = ContextObjectFactory.eINSTANCE.createHeaderInfoType();
}
//Abrufen von HTTP-Header und 'HTTP Control' von 'HeaderInfoType'
HTTPHeaderType httpHeaderType = headers.getHTTPHeader();
HTTPControl cp = httpHeaderType.getControl();
HeadersFactory factory = HeadersFactory.eINSTANCE;
if(cp == null){
    cp = factory.createHTTPControl();
}
//Erstellen von neuem 'HTTPAuthentication' und Festlegen von 'HTTPCredentials'
HTTPAuthentication authorization = factory.createHTTPAuthentication();
HTTPCredentials credentials = factory.createHTTPCredentials();
authorization.setAuthenticationType(HTTPAuthenticationType.BASIC_LITERAL);
credentials.setUserId("USERNAME");
credentials.setPassword("PASSWORD");
authorization.setCredentials(credentials);
cp.setAuthentication(authorization);
httpHeaderType.setControl(cp);
// Header-Informationen wieder in aktuellen Ausführungskontext einfügen
contextService.setHeaderInfo(headers);
```

Serverkonfiguration prüfen:

Wenn Sie eine Anwendung mit einer Web-Service-Bindung (JAX-WS) implementieren, müssen Sie sicherstellen, dass auf dem Zielserver, auf dem die Anwendung implementiert wird, nicht die Option für den **Start von Komponenten nach Bedarf** ausgewählt ist.

Sie können überprüfen, ob diese Option ausgewählt ist, indem Sie in der Administrationskonsole die folgenden Schritte ausführen:

1. Klicken Sie auf **Server** > **Servertypen** > **WebSphere-Anwendungsserver**.
2. Klicken Sie auf den Namen des Servers.
3. Prüfen Sie auf der Registerkarte 'Konfiguration', ob die Option **Komponenten nach Bedarf starten** ausgewählt ist.
4. Führen Sie einen der folgenden Schritte aus:
   - Wenn die Option **Komponenten nach Bedarf starten** ausgewählt ist, wählen Sie sie durch Entfernen der entsprechenden Markierung ab und klicken Sie dann auf **Anwenden**.
   - Wenn die Option **Komponenten nach Bedarf starten** nicht ausgewählt ist, klicken Sie auf **Abbrechen**.

Anhänge in SOAP-Nachrichten:

Sie können SOAP-Nachrichten senden und empfangen, die binäre Daten (z. B. PDF-Dateien oder JPEG-Bilder) als Anhänge enthalten. Anhänge können *referenziert* (also in der Serviceschnittstelle explizit als Nachrichtenteile dargestellt) oder *nicht referenziert* sein (in diesem Fall können beliebige Anhangsmengen und -typen eingeschlossen werden.

Ein referenzierter Anhang kann mit einem der folgenden Verfahren dargestellt werden:

- MTOM-Anhänge verwenden die SOAP-MTOM-Codierung (MTOM - (Message Transmission Optimization Mechanism) (siehe http://www.w3.org/TR/soap12-mtom/). MTOM-Anhänge werden über eine Konfigurationsoption in den Import- und Exportbindungen aktiviert und stellen die empfohlene Methode zur Codierung von Anhängen für neue Anwendungen dar.
- Als Element des Typs 'wsi:swaRef' im Nachrichtenschema.

  Anhänge, die mit dem Typ 'wsi:swaRef' definiert sind, sind mit der WSI-Spezifikation *Attachments Profile Version 1.0* (http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html) konform. Diese Spezifikation definiert, wie Nachrichtenelemente zu MIME-Teilen in Beziehung stehen.
- Als Nachrichtenteil der höchsten Ebene unter Verwendung eines binären Schematyps

  Anhänge, die als Nachrichtenteile der höchsten Ebene dargestellt werden, sind mit der Spezifikation *SOAP Messages with Attachments* (http://www.w3.org/TR/SOAP-attachments) konform.

  Anhänge, die als Nachrichtenteile der höchsten Ebene dargestellt werden, können außerdem konfiguriert werden, um sicherzustellen, dass das WSDL-Dokument und die Nachrichten, die durch die Bindung erzeugt werden, mit den WS-I-Spezifikationen *Attachments Profile Version 1.0* und *Basic Profile Version 1.1* (http://www.ws-i.org/Profiles/BasicProfile-1.1.html) konform sind.

Ein nicht referenzierter Anhang wird in einer SOAP-Nachricht ohne jegliche Darstellung im Nachrichtenschema übertragen.

In allen Fällen mit Ausnahme von MTOM-Anhängen sollte die WSDL-SOAP-Bindung eine MIME-Bindung für zu verwendende Anhänge einschließen und die maximale Größe der Anhänge sollte 20 MB nicht überschreiten.

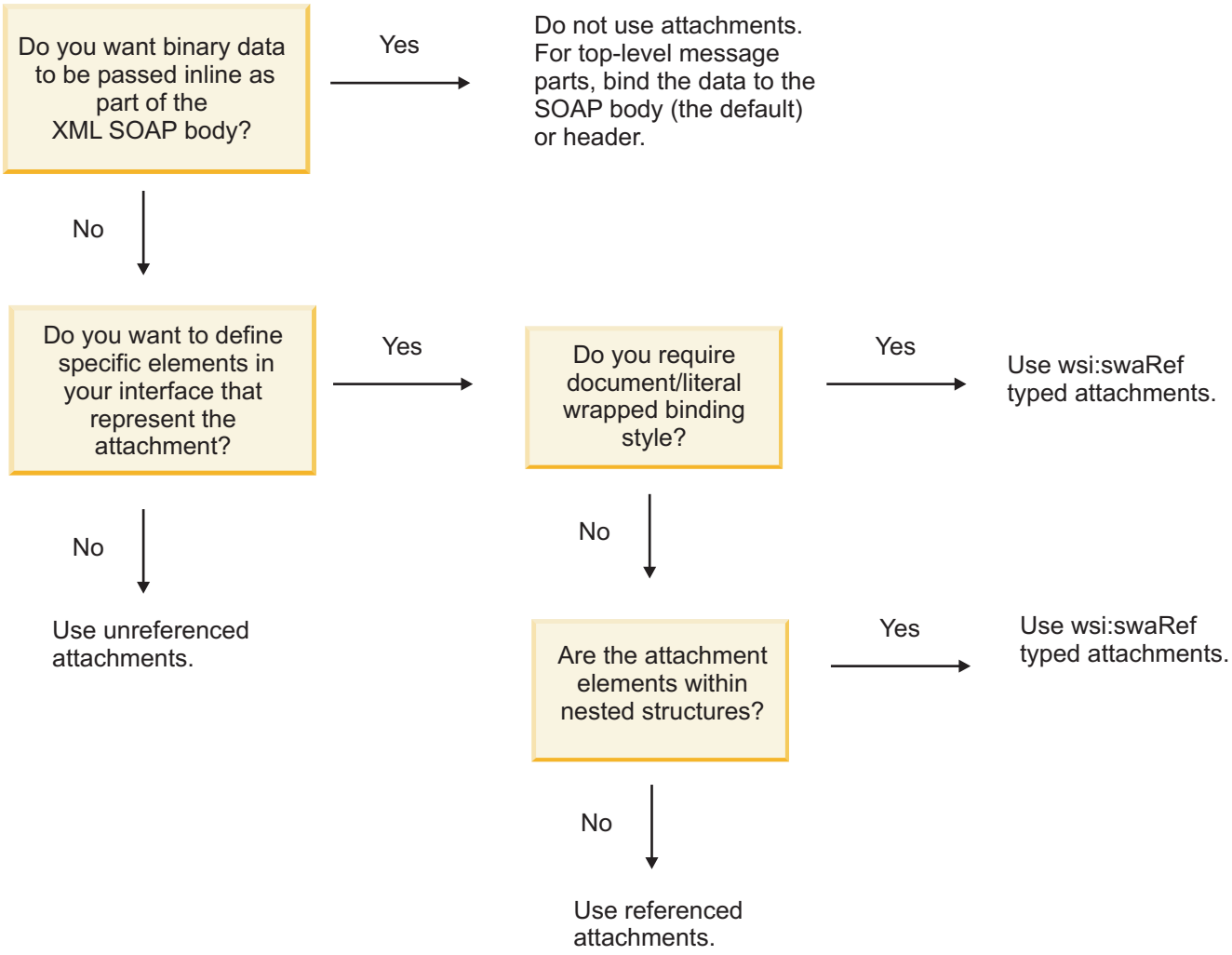
Anmerkung: Um SOAP-Nachrichten mit Anhängen zu senden oder zu empfangen, müssen Sie eine der Web-Service-Bindungen verwenden, die auf Java API for XML Web Services (JAX-WS) basieren.

Geeignete Anhangsdarstellung auswählen :

Beim Entwerfen einer neuen Serviceschnittstelle, die Binärdaten einbezieht, müssen Sie berücksichtigen, wie diese Binärdaten in den vom Service gesendeten und empfangenen SOAP-Nachrichten übertragen werden.

Für Anhänge sollte MTOM (Message Transmission Optimization Mechanism) verwendet werden, sofern die verbundene Web-Service-Anwendung MTOM unterstützt. Sollte dies nicht der Fall sein, zeigt das fol-

gende Diagramm, wie Sie andere Anhangsdarstellungen auswählen können. Ermitteln Sie die geeignete Anhangsdarstellung anhand der folgenden Fragen:



MTOM-Anhänge: Nachrichtenteile der höchsten Ebene:

Sie haben die Möglichkeit, Web-Service-Nachrichten mit SOAP-MTOM-Anhängen (MTOM - Message Transmission Optimization Mechanism) zu senden und zu empfangen. In SOAP-Nachrichten mit mehreren MIME-Teilen ist der SOAP-Hauptteil der erste Teil der Nachricht. Die Anhänge befinden sich in nachfolgenden Teilen.

Beim Senden oder Empfangen eines referenzierten Anhangs in einer SOAP-Nachricht befinden sich die Binärdaten, aus denen der (oftmals relativ große) Anhang besteht, nicht im Hauptteil der SOAP-Nachricht und müssen daher nicht als XML-Daten syntaktisch analysiert werden. Dies führt zu einer effizienteren Verarbeitung als bei Binärdaten, die sich in einem XML-Element befinden.

Beispiel für eine MTOM-SOAP-Nachricht:

```
... other transport headers ...
Content-Type: multipart/related; boundary=MIMEBoundaryurn_uuid_0FE43E4D025F0BF3DC11582467646812; type="application/xop+xml"
<0.urn:uuid:0FE43E4D025F0BF3DC11582467646813@apache.org>"; start-info="text/xml"; charset=UTF-8

--MIMEBoundaryurn_uuid_0FE43E4D025F0BF3DC11582467646812
content-type: application/xop+xml; charset=UTF-8; type="text/xml";
content-transfer-encoding: binary
content-id:
```

```
   <0.urn:uuid:0FE43E4D025F0BF3DC11582467646813@apache.org>

<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
      <soapenv:Header/>
      <soapenv:Body>
         <sendImage xmlns="http://org/apache/axis2/jaxws/sample/mtom">
            <input>
            <imageData><xop:Include xmlns:xop="http://www.w3.org/2004/08/xop/include" href="cid:1.urn:uuid:0FE43E4D025F0
            </input>
         </sendImage>
      </soapenv:Body>
   </soapenv:Envelope>
--MIMEBoundaryurn_uuid_0FE43E4D025F0BF3DC11582467646812
content-type: text/plain
content-transfer-encoding: binary
content-id:
         <1.urn:uuid:0FE43E4D025F0BF3DC11582467646811@apache.org>

... binary data goes here ...
--MIMEBoundaryurn_uuid_0FE43E4D025F0BF3DC11582467646812--
```

Beachten Sie in diesem MTOM-Beispiel, dass der Inhaltstyp ('content-type') für die SOAP-Rahmenanweisung **application/xop+xml** lautet und die Binärdaten durch ein **'xop:Include'**-Element wie das folgende ersetzt werden:

```
<xop:Include xmlns:xop="http://www.w3.org/2004/08/xop/include" href="cid:1.urn:uuid:0FE43E4D025F0BF3DC11582467646811@apac
```

Eingangsverarbeitung von referenzierten Anhängen

Wenn ein Client eine SOAP-Nachricht mit einem Anhang an eine SCA-Komponente übergibt, entfernt die Web-Service-Exportbindung (JAX-WS) zunächst den Anhang. Anschließend wird der SOAP-Teil der Nachricht syntaktisch analysiert und ein Geschäftsobjekt erstellt. Abschließend legt die Bindung die Anhangsbinärdaten im Geschäftsobjekt fest.

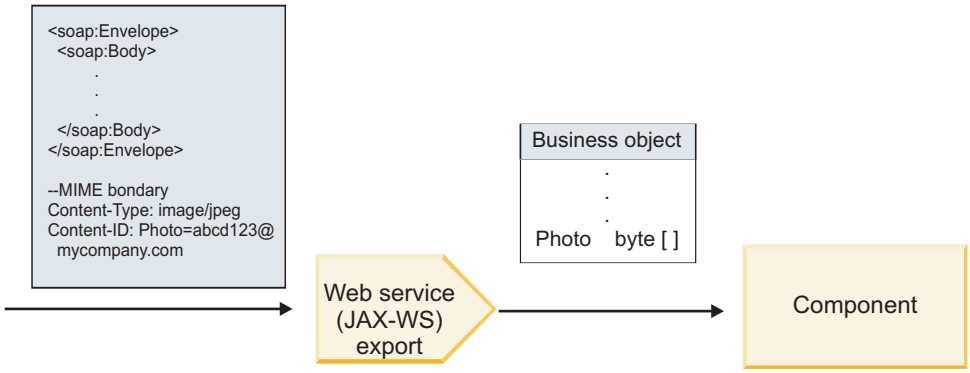


Abbildung 13. Verarbeitung einer SOAP-Nachricht mit einem referenzierten Anhang durch die Web-Service-Exportbindung (JAX-WS)

Attribute von MTOM-Anhängen

- MTOM unterstützt Anhangselemente in verschachtelten Strukturen.
- MTOM ist nur für den Typ 'base64Binary' verfügbar.
- MTOM unterstützt Anhangselemente in verschachtelten Strukturen; dies bedeutet, dass **bodyPath** für MTOM-Anhänge die **xpath**-Position für das Element darstellt, in dem sich der MTOM-Anhang befindet. Die Datenverarbeitungslogik für **bodyPath** hält sich streng an das Schema zum Generieren der **xpath**-Position (siehe folgende Beispiele):
  - Für Nicht-Array-Typen (**maxOccurs** = 1): /sendImage/input/imageData

– Für Array-Typen (**maxOccurs** > 1): /sendImage/input/imageData[1]

- Gemischte Anhangstypen werden nicht unterstützt; dies bedeutet, dass der MTOM-Anhang generiert wird, wenn MTOM für die Importbindung aktiviert ist. Wenn MTOM inaktiviert ist oder der MTOM-Konfigurationswert die Standardeinstellung für die Exportbindung behält, wird die eingehende MTOM-Nachricht nicht unterstützt.

Referenzierte Anhänge: swaRef-typisierte Elemente:

Sie können SOAP-Nachrichten mit eingeschlossenen Anhängen senden und empfangen, die in der Serviceschnittstelle als Elemente des Typs 'swaRef' dargestellt sind.

Ein Element des Typs 'swaRef' ist in Version 1.0 der Spezifikation *Attachments Profile* von Web Services Interoperability Organization (WS-I) definiert (http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html). Diese Spezifikation legt fest, wie Nachrichtenelemente zu MIME-Teilen in Beziehung gesetzt werden.

In der SOAP-Nachricht enthält der SOAP-Hauptteil ein Element des Typs 'swaRef', das die Inhalts-ID des Anhangs angibt.

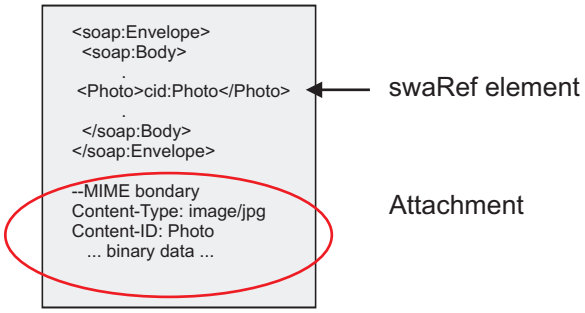


Abbildung 14. SOAP-Nachricht mit Element des Typs 'swaRef'

Die WSDL für diese SOAP-Nachricht enthält ein Element des Typs 'swaRef' innerhalb eines Nachrichtenteils, der den Anhang angibt.

```
<element name="sendPhoto">
  <complexType>
     <sequence>
       <element name="Photo" type="wsi:swaRef"/>
     </sequence>
  </complexType>
 </element>
```

Die WSDL sollte außerdem eine MIME-Bindung enthalten, die angibt, dass Nachrichten mit mehreren MIME-Teilen verwendet werden.

Anmerkung: Die WSDL enthält für das entsprechende Nachrichtenelement des Typs 'swaRef' *keine* MIME-Bindung, da MIME-Bindungen nur auf Nachrichtenteile der höchsten Ebene angewendet werden.

Anhänge, die als Elemente des Typs 'swaRef' dargestellt werden, können ausschließlich über Mediationsablaufkomponenten weitergegeben werden. Falls durch einen anderen Komponententyp auf einen Anhang zugegriffen bzw. ein Anhang weitergegeben werden muss, verwenden Sie eine Mediationsablaufkomponente, um den Anhang an eine Position zu versetzen, auf die die Komponente zugreifen kann.

Eingangsverarbeitung von Anhängen

Zum Konfigurieren einer Exportbindung für den Empfang des Anhangs verwenden Sie Integration Designer. Sie erstellen ein Modul und dessen zugehörige Schnittstelle und Operationen, inklusive einem Element des Typs 'swaRef'. Anschließend erstellen Sie eine Web-Service-Bindung (JAX-WS).

Anmerkung: Das Thema über die Arbeit mit Anhängen im Information Center von Integration Designer enthält detailliertere Informationen.

Wenn ein Client eine SOAP-Nachricht mit einem Anhang des Typs 'swaRef' an eine SCA-Komponente übergibt, entfernt die Web-Service-Exportbindung (JAX-WS) zunächst den Anhang. Anschließend wird der SOAP-Teil der Nachricht syntaktisch analysiert und ein Geschäftsobjekt erstellt. Abschließend legt die Bindung die Inhalts-ID des Anhangs im Geschäftsobjekt fest.

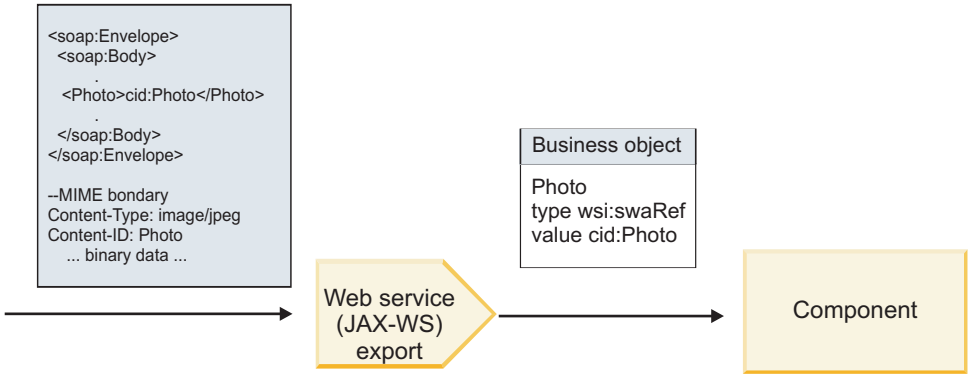


Abbildung 15. Verarbeitung einer SOAP-Nachricht mit einem Anhang des Typs 'swaRef' durch die Web-Service-Exportbindung (JAX-WS)

Auf Anhangsmetadaten in einer Mediationsablaufkomponente zugreifen

Aus Abb. 16 auf Seite 90 geht hervor, dass die Inhalts-ID des Anhangs als Element des Typs 'swaRef' dargestellt wird, wenn durch Komponenten auf Anhänge des Typs 'swaRef' zugegriffen wird.

Für jeden Anhang einer SOAP-Nachricht gibt es ein entsprechendes Element **attachments** im Servicenachrichtenobjekt. Bei Verwendung des WS-I-Typs 'swaRef' enthält das Element **attachments** den Inhaltstyp und die Inhalts-ID des Anhangs sowie die tatsächlichen Binärdaten des Anhangs.

Um den Wert eines Anhangs des Typs 'swaRef' zu erhalten, muss daher der Wert des Elements des Typs 'swaRef' abgerufen und dann nach dem Element **attachments** mit dem entsprechenden Wert für **contentID** gesucht werden. Bitte beachten Sie, dass beim Wert für **contentID** normalerweise das Präfix **cid:** aus dem Wert des Typs 'swaRef' entfernt wurde.

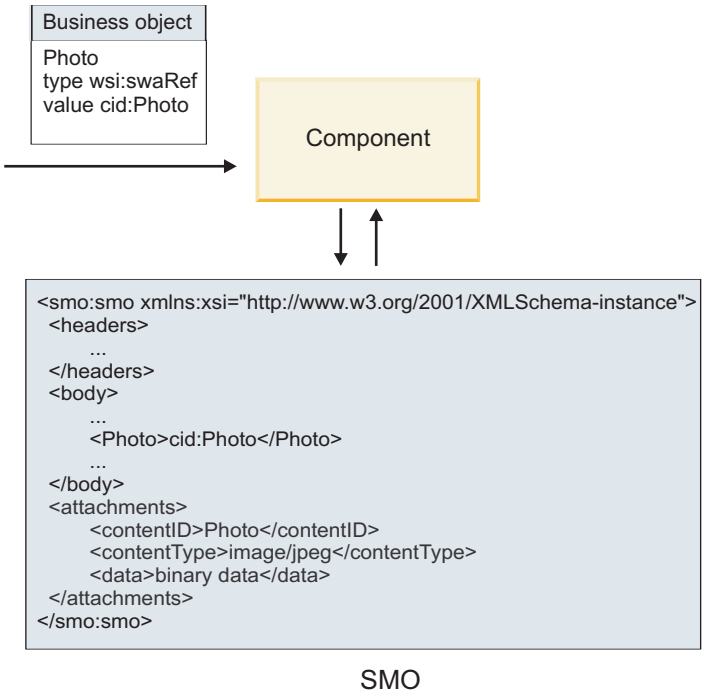


Abbildung 16. Darstellung von Anhängen des Typs 'swaRef' im Servicenachrichtenobjekt

Ausgangsverarbeitung

Zum Konfigurieren einer Web-Service-Importbindung (JAX-WS-) für den Aufruf eines externen Web-Service verwenden Sie Integration Designer. Die Importbindung wird mit einem WSDL-Dokument konfiguriert, das den aufzurufenden Web-Service beschreibt und den Anhang definiert, der an den Web-Service übergeben werden soll.

Wenn eine SCA-Nachricht von einer Web-Service-Importbindung (JAX-WS) empfangen wird, werden Elemente des Typs 'swaRef' als Anhänge gesendet, falls der Import mit einer Mediationsablaufkomponente verbunden ist und es für das Element des Typs 'swaRef' ein entsprechendes Element **attachments** gibt.

Bei der Ausgangsverarbeitung werden Elemente des Typs 'swaRef' immer mit ihren Werten für die Inhalts-ID gesendet. Das Mediationsmodul muss jedoch sicherstellen, dass es ein entsprechendes Element **attachments** mit einem übereinstimmenden Wert für **contentID** gibt.

Anmerkung: Zur Einhaltung der WS-I-Spezifikation für Anhangsprofile sollte der Wert für **content ID** auf die Codierung für den Teil 'content-id' folgen (siehe Abschnitt 3.8 der WS-I-Spezifikation *Attachments Profile 1.0*.

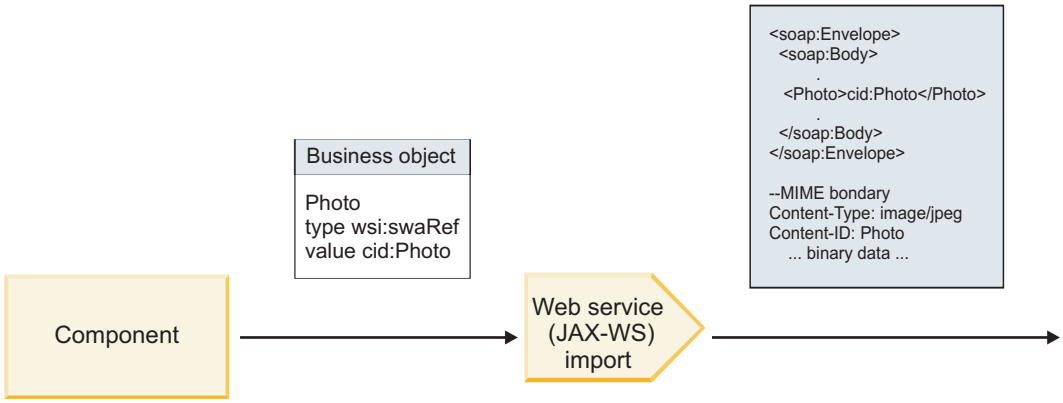


Abbildung 17. Generierung einer SOAP-Nachricht mit einem Anhang des Typs 'swaRef' durch eine Web-Service-Importbindung (JAX-WS)

Anhangsmetadaten in einer Mediationsablaufkomponente festlegen

Falls es im Servicenachrichtenobjekt einen Wert für das Element des Typs 'swaRef' und ein Element **attachments** gibt, bereitet die Bindung die SOAP-Nachricht (mit dem Anhang) vor und sendet sie an einen Empfänger.

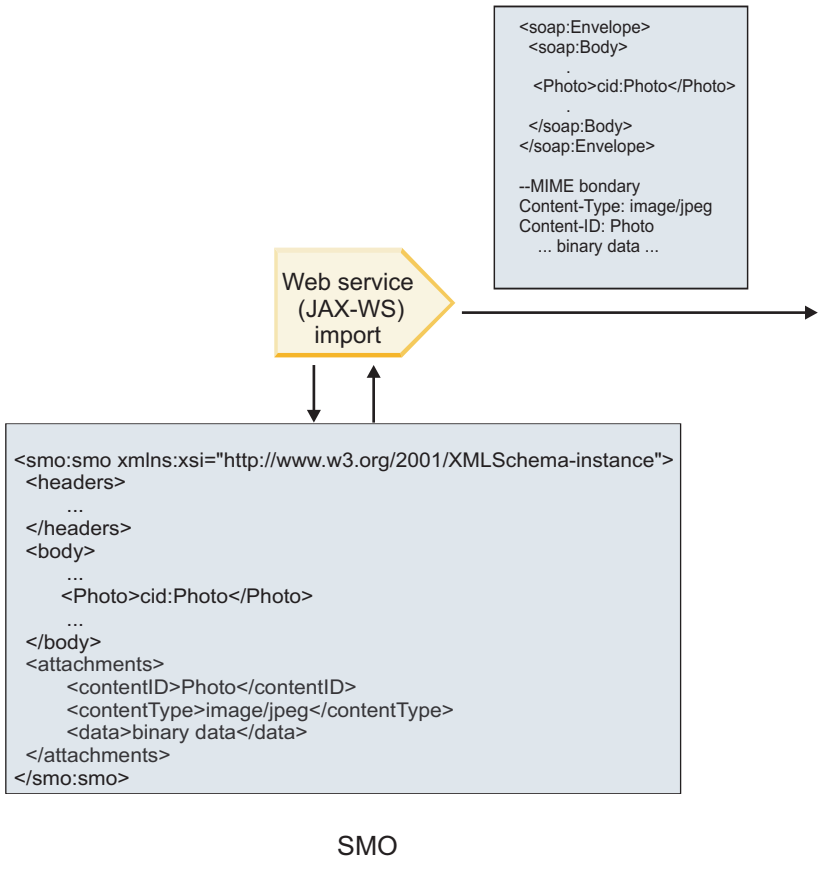


Abbildung 18. Zugriff auf Anhang des Typs 'swaRef' im Servicenachrichtenobjekt zur Erstellung der SOAP-Nachricht

Das Element **attachments** ist im Servicenachrichtenobjekt nur dann vorhanden, wenn eine Mediationsablaufkomponente direkt mit dem Import oder dem Export verbunden ist. Bei anderen Komponententypen wird das Element nicht übergeben. Falls die Werte in einem Modul benötigt werden, das andere Komponententypen enthält, sollten die Werte mit einer Mediationsablaufkomponente an eine Position kopiert

werden, an der das Modul auf sie zugreifen kann. Mit einer weiteren Mediationsablaufkomponente sollten die korrekten Werte festgelegt werden, bevor mittels eines Web-Service-Imports ein abgehender Aufruf erfolgt.

Wichtig: Wie im Thema über die XML-Darstellung des Servicenachrichtenobjekts (SMO) beschrieben, konvertiert das Mediationsbasiselement für Zuordnung Nachrichten unter Verwendung einer XSLT 1.0-Transformation. Die Transformation wird für eine XML-Serialisierung des Servicenachrichtenobjekts ausgeführt. Das Mediationsbasiselement für Zuordnung ermöglicht die Angabe des Stammelements für die Serialisierung. Das Stammelement des XML-Dokuments gibt dieses Stammelement wieder.

Wenn Sie SOAP-Nachrichten mit Anhängen senden, bestimmt das von Ihnen ausgewählte Stammelement, wie Anhänge weitergegeben werden.

- Falls Sie '/body' als Stammelement für die XML-Zuordnung verwenden, werden alle Anhänge standardmäßig über die Zuordnung weitergegeben.
- Falls Sie '/' als Stammelement der Zuordnung verwenden, können Sie die Weitergabe von Anhängen steuern.

Referenzierte Anhänge: Nachrichtenteile der höchsten Ebene:

Sie können SOAP-Nachrichten senden und empfangen, in denen binäre Anhänge enthalten sind, die als Teile in Ihrer Serviceschnittstelle deklariert sind.

In SOAP-Nachrichten mit mehreren MIME-Teilen ist der SOAP-Hauptteil der erste Teil der Nachricht. Der Anhang oder Anhänge befinden sich in nachfolgenden Teilen.

Welche Vorteile ergeben sich beim Senden und Empfangen eines referenzierten Anhangs in einer SOAP-Nachricht? Die Binärdaten, aus denen der (oftmals relativ große) Anhang besteht, werden vom Hauptteil der SOAP-Nachricht getrennt und müssen daher nicht als XML syntaktisch analysiert werden. Dies führt zu einer effizienteren Verarbeitung als bei Binärdaten, die sich in einem XML-Element befinden.

Typen von SOAP-Nachrichten mit referenzierten Anhängen

Ab Version 7.0.0.3 von IBM Business Process Manager können Sie auswählen, wie die SOAP-Nachricht generiert werden soll:

- **WS-I-konforme Nachrichten**

  Die Laufzeit kann SOAP-Nachrichten generieren, die mit *Attachments Profile Version 1.0* und *Basic Profile Version 1.1* von WS-I konform sind. In einer SOAP-Nachricht, die mit diesen Profilen konform ist, ist nur ein einziger Nachrichtenteil an den SOAP-Hauptteil gebunden. Bei den Teilen, die als Anhänge gebunden sind, wird der Anhang unter Verwendung der Codierung für den Teil 'content-id' (beschrieben in *Attachments Profile Version 1.0* von WS-I) auf den Nachrichtenteil bezogen.
- **Nicht WS-I-konforme Nachrichten**

  Die Laufzeit kann SOAP-Nachrichten generieren, die nicht mit den WS-I-Profilen konform, jedoch mit den Nachrichten kompatibel sind, die in Version 7.0 oder 7.0.0.2 von IBM Business Process Manager generiert werden. Die SOAP-Nachrichten verwenden nach dem Nachrichtenteil Elemente der höchsten Ebene mit einem Attribut **href**, das die Inhalts-ID (**content-id**) für den Anhang enthält. Die Codierung für den Teil 'content-i' (beschrieben in *Attachments Profile Version 1.0* von WS-I) wird jedoch nicht verwendet.

WS-I-Konformität für Web-Service-Exporte auswählen

Zum Konfigurieren einer Exportbindung verwenden Sie Integration Designer. Sie erstellen ein Modul und dessen zugehörige Schnittstelle und Operationen. Anschließend erstellen Sie eine Web-Service-Bindung (JAX-WS). Auf der Seite **Referenzierte Anhänge** werden alle binären Teile aus der erstellen Operation angezeigt. Sie wählen aus, bei welchen Teilen es sich um Anhänge handelt. Anschließend geben Sie auf der Seite **WS-I-Einhaltung angeben** von Integration Designer ein der folgenden Optionen an:

- **WS-I-konforme SOAP-Nachricht verwenden**

  Falls Sie diese Option auswählen, geben Sie außerdem an, welcher Nachrichtenteil an den SOAP-Hauptteil gebunden werden soll.

  Anmerkung: Diese Option kann nur verwendet werden, wenn die korrespondierende WSDL-Datei ebenfalls WS-I-konform ist.

  Eine von Integration Designer Version 7.0.0.3 generierte WSDL-Datei ist mit WS-I konform. Wenn Sie jedoch eine WSDL-Datei importieren, die nicht mit WS-I konform ist, können Sie diese Option nicht auswählen.
- **Nicht WS-I-konforme SOAP-Nachricht verwenden**

  Falls Sie diese Option auswählen, bei der es sich um die Standardoption handelt, wird der erste Nachrichtenteil an den SOAP-Hauptteil gebunden.

Anmerkung: Nur Nachrichtenteile der höchsten Ebene (also im WSDL-Element 'portType' als Teile in der Eingabe- oder Ausgabenachricht definierte Elemente), die einen binären Typ besitzen (entweder 'base64Binary' oder 'hexBinary') können als referenzierte Anhänge gesendet oder empfangen werden. Das Thema über die Arbeit mit Anhängen im Information Center von Integration Designer enthält detailliertere Informationen.

Bei WS-I-konformen Nachrichten stellt die Inhalts-ID, die in der SOAP-Nachricht generiert wird, eine Verkettung der folgenden Elemente dar:

- Wert des Attributs **name** des Elements **wsdl:part**, das von **mime:content** referenziert wird
- Zeichen **=**
- Global eindeutiger Wert, z. B. eine UUID
- Zeichen @
- Gültiger Domänenname

Eingangsverarbeitung von referenzierten Anhängen

Wenn ein Client eine SOAP-Nachricht mit einem Anhang an eine SCA-Komponente übergibt, entfernt die Web-Service-Exportbindung (JAX-WS) zunächst den Anhang. Anschließend wird der SOAP-Teil der Nachricht syntaktisch analysiert und ein Geschäftsobjekt erstellt. Abschließend legt die Bindung die Anhangsbinärdaten im Geschäftsobjekt fest.

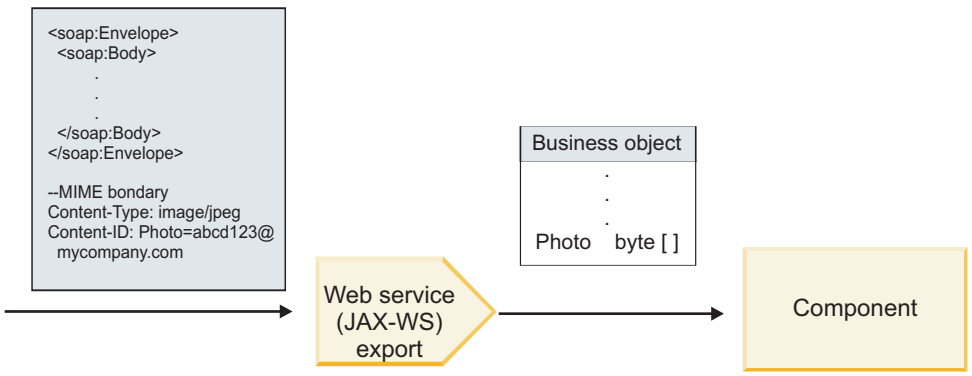


Abbildung 19. Verarbeitung einer WS-I-konformen SOAP-Nachricht mit einem referenzierten Anhang durch die Web-Service-Exportbindung (JAX-WS)

Auf Anhangsmetadaten in einer Mediationsablaufkomponente zugreifen

Aus Abb. 19 auf Seite 93 geht hervor, dass die Anhangsdaten als Byte-Array dargestellt werden, wenn durch Komponenten auf referenzierte Anhänge zugegriffen wird.

Für jeden referenzierten Anhang einer SOAP-Nachricht gibt es ein entsprechendes Element **attachments** im Servicenachrichtenobjekt. Das Element **attachments** enthält den Inhaltstyp des Anhangs und den Pfad zum Nachrichtenhauptteilelement, in dem sich der Anhang befindet.

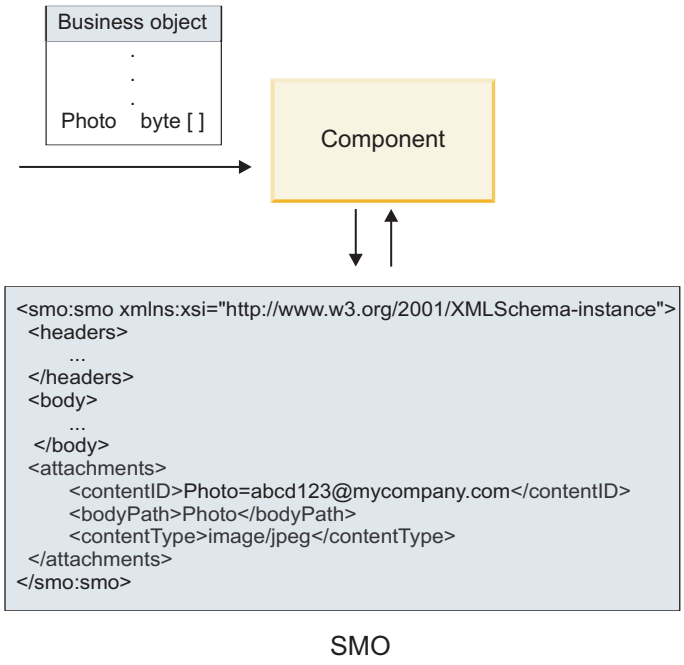


Abbildung 20. Darstellung referenzierter Anhänge im Servicenachrichtenobjekt

Wichtig: Der Pfad zum Nachrichtenhauptteilelement wird nicht automatisch aktualisiert, wenn die Nachricht transformiert und der Anhang versetzt wird. Sie können einen Mediationsablauf verwenden, um das Element **attachments** mit dem neuen Pfad zu aktualisieren (beispielsweise im Rahmen der Transformation oder durch die Verwendung eines separaten Setter-Nachrichtenelements).

Erstellung von abgehenden SOAP-Nachrichten

Zum Konfigurieren einer Web-Service-Importbindung (JAX-WS-) für den Aufruf eines externen Web-Service verwenden Sie Integration Designer. Die Importbindung wird mit einem WSDL-Dokument konfiguriert, das den aufzurufenden Web-Service beschreibt und definiert, welche Nachrichtenteile als Anhänge übergeben werden sollen. Auf der Seite **WS-I-Einhaltung angeben** von Integration Designer können Sie außerdem eine der folgenden Optionen angeben:

- **WS-I-konforme SOAP-Nachricht verwenden**

  Falls Sie diese Option auswählen, geben Sie außerdem an, welcher Nachrichtenteil an den SOAP-Hauptteil gebunden werden soll. Alle anderen Teile werden an Anhänge oder Header gebunden. Nachrichten, die an die Bindung gesendet werden, enthalten im SOAP-Hauptteil keine Elemente, die die Anhänge referenzieren. Die Beziehung wird durch die Inhalt-ID des Anhangs ausgedrückt, die den Namen des Nachrichtenteils enthält.
- **Nicht WS-I-konforme SOAP-Nachricht verwenden**

  Falls Sie diese Option auswählen, bei der es sich um die Standardoption handelt, wird der erste Nachrichtenteil an den SOAP-Hauptteil gebunden. Alle anderen Teile werden an Anhänge oder Header ge-

bunden. Nachrichten, die durch die Bindung gesendet werden, enthalten im SOAP-Hauptteil eines oder mehrere Elemente, die die Anhänge mittels eines Attributs **href** referenzieren.

Anmerkung: Der Teil, der einen Anhang darstellt (wie in WSDL definiert) muss einen einfachen Typ besitzen (entweder 'base64Binary' oder 'hexBinary'). Falls ein Teil durch einen komplexen Typ definiert wird, kann er nicht als Anhang gebunden werden.

Ausgangsverarbeitung von referenzierten Anhängen

Die Importbindung ermittelt anhand von Informationen im Servicenachrichtenobjekt, wie die binären Nachrichtenteile der höchsten Ebene als Anhänge gesendet werden.

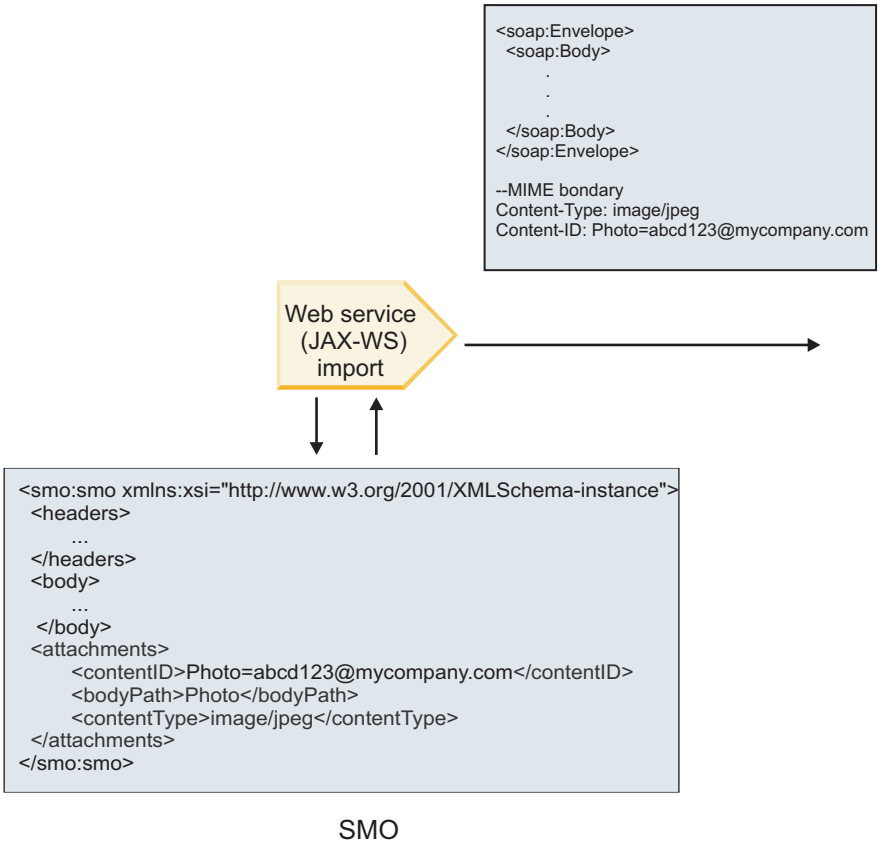


Abbildung 21. Zugriff auf den referenzierten Anhang im Servicenachrichtenobjekt zur Erstellung der SOAP-Nachricht

Das Element **attachments** ist im Servicenachrichtenobjekt nur dann vorhanden, wenn eine Mediationsablaufkomponente direkt mit dem Import oder dem Export verbunden ist. Bei anderen Komponententypen wird das Element nicht übergeben. Falls die Werte in einem Modul benötigt werden, das andere Komponententypen enthält, sollten die Werte mit einer Mediationsablaufkomponente an eine Position kopiert werden, an der das Modul auf sie zugreifen kann. Mit einer weiteren Mediationsablaufkomponente sollten die korrekten Werte festgelegt werden, bevor mittels eines Web-Service-Imports ein abgehender Aufruf erfolgt.

Die Bindung ermittelt anhand einer Kombination der folgenden Bedingungen, wie und ob die Nachricht gesendet wird:

- Gibt es für den binären Nachrichtenteil der höchsten Ebene eine WSDL-MIME-Bindung und, wenn ja, wie ist der Inhaltstyp definiert?
- Gibt es im Servicenachrichtenobjekt ein Element **attachments**, dessen Wert für **bodyPath** einen binären Teil der höchsten Ebene referenziert?

Erstellung von Anhängen bei vorhandenem Element attachment im Servicenachrichtenobjekt

Die folgende Tabelle zeigt, wie ein Anhang erstellt und gesendet wird, wenn das Servicenachrichtenobjekt ein Element **attachment** mit einem Wert für **bodyPath** enthält, der mit einem Nachrichtennamensteil übereinstimmt:

Tabelle 32. Generierung des Anhangs

Status der WSDL-MIME-Bindung für binären Nachrichtenteil der höchsten Ebene	Verfahren beim Erstellen und Senden der Nachricht
Mit einer der folgenden Bedingungen vorhanden: • Kein definierter Inhaltstyp für den Nachrichtenteil • Mehrere definierte Inhaltstypen • Definierter Platzhalter für den Inhaltstyp	Der Nachrichtenteil wird als Anhang gesendet. Der Wert für **Content-Id** wird auf den Wert im Element 'attachments' gesetzt, wenn dieses vorhanden ist; andernfalls wird ein Wert generiert. Der Wert für **Content-Type** wird auf den Wert im Element 'attachments' gesetzt, wenn dieses vorhanden ist; andernfalls wird er auf **application/octet-stream** gesetzt.
Mit Inhalt für Nachrichtenteil (kein Platzhalter) vorhanden	Der Nachrichtenteil wird als Anhang gesendet. Der Wert für **Content-Id** wird auf den Wert im Element 'attachments' gesetzt, wenn dieses vorhanden ist; andernfalls wird ein Wert generiert. Der Wert für **Content-Type** wird auf den Wert im Element 'attachments' gesetzt, wenn dieses vorhanden ist; andernfalls wird er auf den Typ gesetzt, der im WSDL-MIME-Inhaltselement definiert ist.
Nicht vorhanden	Der Nachrichtenteil wird als Anhang gesendet. Der Wert für **Content-Id** wird auf den Wert im Element 'attachments' gesetzt, wenn dieses vorhanden ist; andernfalls wird ein Wert generiert. Der Wert für **Content-Type** wird auf den Wert im Element 'attachments' gesetzt, wenn dieses vorhanden ist; andernfalls wird er auf **application/octet-stream** gesetzt. **Anmerkung:** Das Senden von Nachrichtenteilen als Anhänge, die als solche nicht in WSDL definiert sind, kann die Einhaltung von 'WS-I Attachments Profile 1.0' aufheben und sollte daher nach Möglichkeit vermieden werden.

Erstellung von Anhängen ohne Element attachment im Servicenachrichtenobjekt

Die folgende Tabelle zeigt, wie ein Anhang erstellt und gesendet wird, wenn das Servicenachrichtenobjekt kein Element **attachment** mit einem Wert für **bodyPath** enthält, der mit einem Nachrichtennamensteil übereinstimmt:

Tabelle 33. Generierung des Anhangs

Status der WSDL-MIME-Bindung für binären Nachrichtenteil der höchsten Ebene	Verfahren beim Erstellen und Senden der Nachricht
Mit einer der folgenden Bedingungen vorhanden: • Kein definierter Inhaltstyp für den Nachrichtenteil • Mehrere definierte Inhaltstypen • Definierter Platzhalter für den Inhaltstyp	Der Nachrichtenteil wird als Anhang gesendet. Der Wert für **Content-Id** wird generiert. Der Wert für **Content-Type** wird auf **application/octet-stream** gesetzt.

Tabelle 33. Generierung des Anhangs (Forts.)

Status der WSDL-MIME-Bindung für binären Nachrichtenteil der höchsten Ebene	**Verfahren beim Erstellen und Senden der Nachricht**
Mit Inhalt für Nachrichtenteil (kein Platzhalter) vorhanden	Der Nachrichtenteil wird als Anhang gesendet. Der Wert für **Content-Id** wird generiert. Der Wert für **Content-Type** wird auf den im WSDL-MIME-Inhaltselement definierten Typ gesetzt.
Nicht vorhanden	Der Nachrichtenteil wird nicht als Anhang gesendet.

Wichtig: Wie im Thema über die XML-Darstellung des Servicenachrichtenobjekts (SMO) beschrieben, konvertiert das Mediationsbasiselement für Zuordnung Nachrichten unter Verwendung einer XSLT 1.0-Transformation. Die Transformation wird für eine XML-Serialisierung des Servicenachrichtenobjekts ausgeführt. Das Mediationsbasiselement für Zuordnung ermöglicht die Angabe des Stammelements für die Serialisierung. Das Stammelement des XML-Dokuments gibt dieses Stammelement wieder.

Wenn Sie SOAP-Nachrichten mit Anhängen senden, bestimmt das von Ihnen ausgewählte Stammelement, wie Anhänge weitergegeben werden.

- Falls Sie '/body' als Stammelement für die XML-Zuordnung verwenden, werden alle Anhänge standardmäßig über die Zuordnung weitergegeben.
- Falls Sie '/' als Stammelement der Zuordnung verwenden, können Sie die Weitergabe von Anhängen steuern.

Nicht referenzierte Anhänge:

Sie können *nicht referenzierte* Anhänge senden und empfangen, die nicht in der Serviceschnittstelle deklariert sind.

In einer SOAP-Nachricht mit mehreren MIME-Teilen ist der SOAP-Hauptteil der erste Teil der Nachricht. Die Anhänge befinden sich in nachfolgenden Teilen. Der SOAP-Hauptteil enthält keine Referenz für den Anhang.

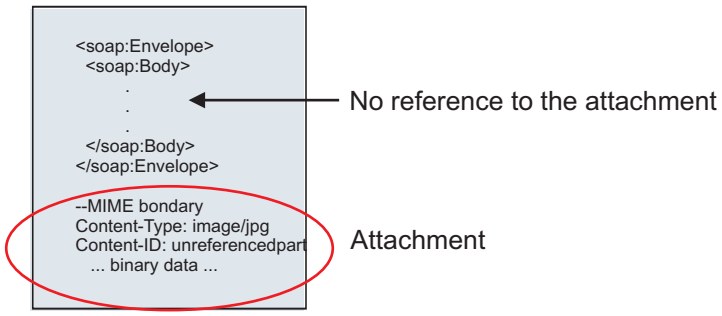


Abbildung 22. SOAP-Nachricht mit nicht referenziertem Anhang

Sie können eine SOAP-Nachricht mit einem nicht referenzierten Anhang über einen Web-Service-Export an einen Web-Service-Import senden. Die Ausgabenachricht, die an den Ziel-Web-Service gesendet wird, enthält den Anhang.

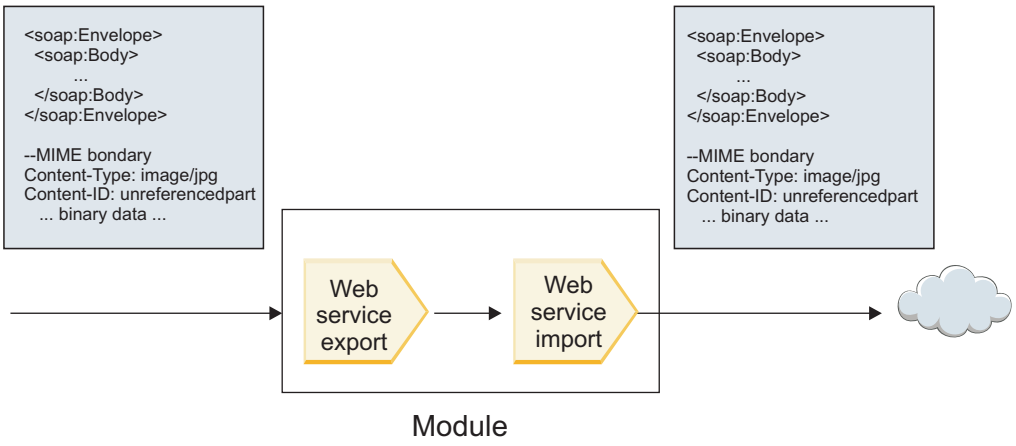


Abbildung 23. Übergabe eines Anhangs durch ein SCA-Modul

In Abb. 23 wird die SOAP-Nachricht mit dem Anhang unverändert übergeben.

Durch die Verwendung einer Mediationsablaufkomponente können Sie die SOAP-Nachricht auch ändern. Beispielsweise können Sie mit der Mediationsablaufkomponente Daten aus der SOAP-Nachricht extrahieren (in diesem Fall Binärdaten im Hauptteil der Nachricht) und eine SOAP-Nachricht mit Anhängen erstellen. Die Daten werden als Teil des Anhangselements eines Servicenachrichtenobjekts verarbeitet.

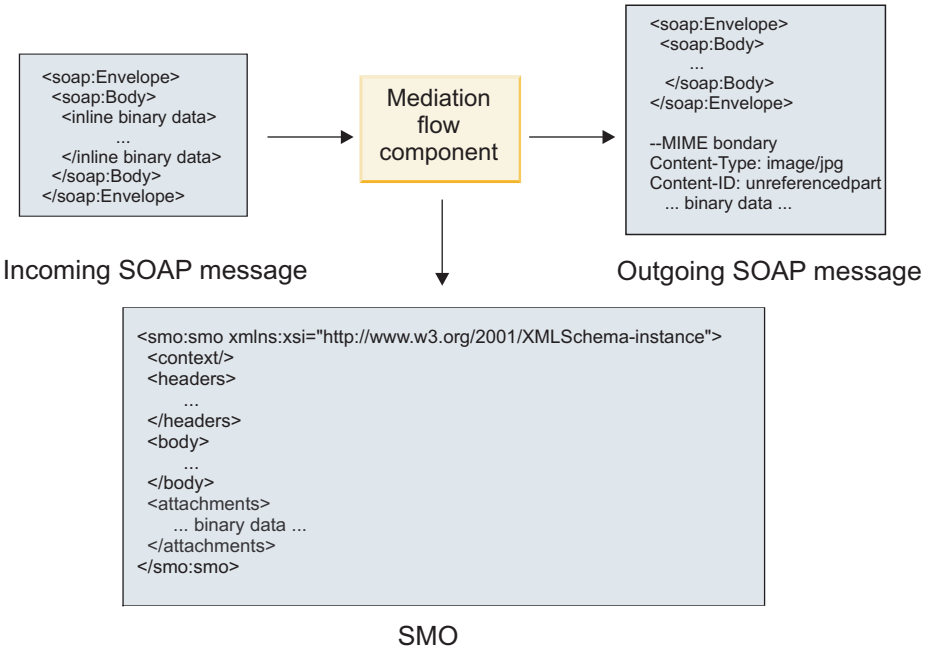


Abbildung 24. Durch eine Mediationsablaufkomponente verarbeitete Nachricht

In der entgegengesetzten Richtung kann die Mediationsablaufkomponente die eingehende Nachricht transformieren, indem sie den Anhang extrahiert und codiert und die Nachricht dann ohne Anhänge überträgt.

Statt Daten aus einer eingehenden SOAP-Nachricht zu extrahieren, um eine SOAP-Nachricht mit Anhängen zu bilden, können Sie die Anhangsdaten von einer fernen Quelle wie beispielsweise einer Datenbank beziehen.

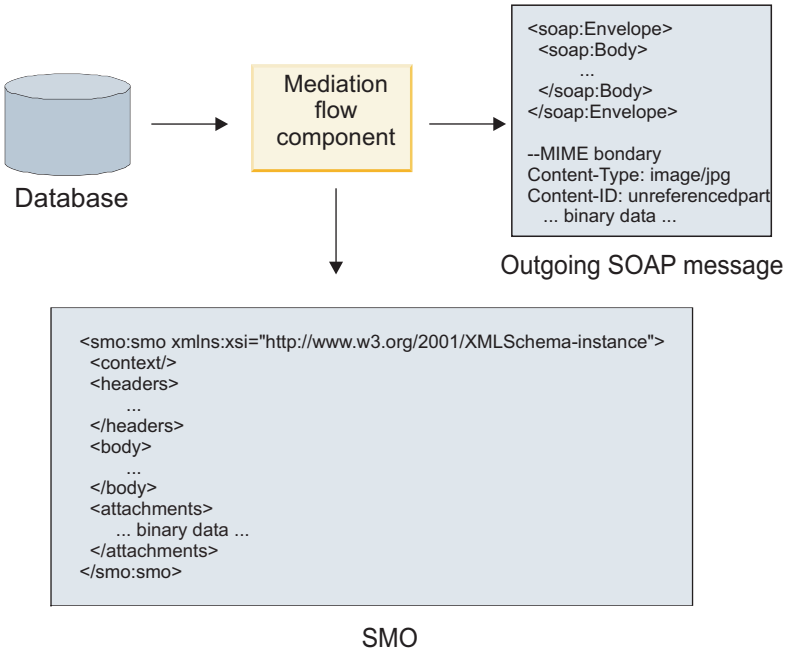


Abbildung 25. Aus einer Datenbank bezogener und zur SOAP-Nachricht hinzugefügter Anhang

In der entgegengesetzten Richtung kann die Mediationsablaufkomponente den Anhang aus einer eingehenden SOAP-Nachricht extrahieren und die Nachricht verarbeiten (z. B. den Anhang in einer Datenbank speichern).

Nicht referenzierte Anhänge können nur über Mediationsablaufkomponenten weitergegeben werden. Falls durch einen anderen Komponententyp auf einen Anhang zugegriffen bzw. ein Anhang weitergegeben werden muss, verwenden Sie eine Mediationsablaufkomponente, um den Anhang an eine Position zu versetzen, auf die die Komponente zugreifen kann.

Wichtig: Wie im Thema über die XML-Darstellung des Servicenachrichtenobjekts (SMO) beschrieben, konvertiert das Mediationsbasiselement für Zuordnung Nachrichten unter Verwendung einer XSLT 1.0-Transformation. Die Transformation wird für eine XML-Serialisierung des Servicenachrichtenobjekts ausgeführt. Das Mediationsbasiselement für Zuordnung ermöglicht die Angabe des Stammelements für die Serialisierung. Das Stammelement des XML-Dokuments gibt dieses Stammelement wieder.

Wenn Sie SOAP-Nachrichten mit Anhängen senden, bestimmt das von Ihnen ausgewählte Stammelement, wie Anhänge weitergegeben werden.

- Falls Sie '/body' als Stammelement für die XML-Zuordnung verwenden, werden alle Anhänge standardmäßig über die Zuordnung weitergegeben.
- Falls Sie '/' als Stammelement der Zuordnung verwenden, können Sie die Weitergabe von Anhängen steuern.

Verwendung der WSDL-Dokumentdarstellungsbindung mit mehrteiligen Nachrichten:

Die Organisation 'Web Services Interoperability Organization' (WS-I) hat eine Reihe von Regeln dafür definiert, wie Web-Services mittels WSDL beschrieben werden sollten und wie die entsprechenden SOAP-Nachrichten gebildet werden sollten, um eine Interoperabilität sicherzustellen.

Diese Regeln sind in der WS-I-Spezifikation *Basic Profile Version 1.1* angegeben (http://www.ws-i.org/Profiles/BasicProfile-1.1.html). Insbesondere gibt der Abschnitt 'R2712' der WS-I-Spezifikation 'Basic Profile 1.1' Folgendes an: "A document-literal binding MUST be serialized as an ENVELOPE with a soap:Body whose child element is an instance of the global element declaration referenced by the corresponding

wsdl:message part." (Eine Dokumentliteralbindung MUSS als Element ENVELOPE mit einem Element 'soap:Body' serialisiert werden, dessen untergeordnetes Element eine Instanz der globalen Elementdeklaration ist, die durch den entsprechenden Teil 'wsdl:message' referenziert wird.)

Dies bedeutet, dass bei Verwendung einer SOAP-Dokumentdarstellungsbindung für eine Operation mit Nachrichten (Eingabe-, Ausgabe- oder Fehlernachrichten), die mit mehreren Teilen definiert sind, nur einer einziger dieser Teile an den SOAP-Hauptteil gebunden werden darf, damit die WS-I-Spezifikation 'Basic Profile 1.1' eingehalten wird.

Desweiteren besagt der Abschnitt 'R2941' der WS-I-Spezifikation 'Attachments Profile 1.0' Folgendes: "A wsdl:binding in a DESCRIPTION SHOULD bind every wsdl:part of a wsdl:message in the wsdl:portType to which it refers to one of soapbind:body, soapbind:header, soapbind:fault , soapbind:headerfault, or mime:content." (Ein Element 'wsdl:binding' in einem Element DESCRIPTION sollte jedes Element 'wsdl-:part' eines Elements 'wsdl:message' im Element 'wsdl:portType', das es referenziert, an eines der folgenden Elemente binden: 'soapbind:body', 'soapbind:header', 'soapbind:fault', 'soapbind:headerfault' oder 'mime:content'.).

Dies bedeutet, dass bei Verwendung einer SOAP-Dokumentdarstellungsbindung für eine Operation mit Nachrichten (Eingabe-, Ausgabe- oder Fehlernachrichten), die mit mehreren Teilen definiert sind, alle anderen Teile als derjenige Teil, der an den SOAP-Hauptteil gebunden ist, als Anhänge oder Header gebunden werden müssen.

Beim Generieren von WSDL-Beschreibungen für Exporte mit Web-Service-Bindungen (JAX-WS und JAX-RPC) in diesem Fall wird das folgende Verfahren verwendet:

- Sie können auswählen, welcher Nachrichtenteil an den SOAP-Hauptteil gebunden wird, wenn es mehrere Element nicht binären Typs gibt. Gibt es lediglich ein einziges Element binären Typs, wird automatisch dieses Element an den SOAP-Hauptteil gebunden.
- Bei der JAX-WS-Bindung werden alle anderen Nachrichtenteile des Typs 'hexBinary' oder 'base64Binary' als referenzierte Anhänge gebunden. Weitere Informationen hierzu finden Sie im Abschnitt „Referenzierte Anhänge: Nachrichtenteile der höchsten Ebene” auf Seite 92.
- Alle anderen Nachrichtenteile werden als SOAP-Header gebunden.

Die JAX-RPC- und JAX-WS-Importbindungen berücksichtigen die SOAP-Bindung in einem vorhandenen WSDL-Dokument mit mehrteiligen Dokumentdarstellungsnachrichten auch dann, wenn durch sie mehrere Teile an den SOAP-Hauptteil gebunden werden. Für solche WSDL-Dokumente können jedoch in Rational Application Developer keine Web-Service-Clients generiert werden.

Anmerkung: Die JAX-RPC-Bindung unterstützt Anhänge nicht.

Bei der Verwendung mehrteiliger Nachrichten mit einer Operation, die eine SOAP-Dokumentdarstellungsbindung besitzt, wird daher das folgende Muster empfohlen:

1. Verwenden Sie die Darstellung mit eingeschlossenem Dokumentliteral. In diesem Fall besitzen Nachrichten immer einen einzigen Teil. Anhänge müssen in diesem Fall jedoch entweder nicht referenziert sein (siehe „Nicht referenzierte Anhänge” auf Seite 97) oder den Typ 'swaRef' aufweisen (siehe „Referenzierte Anhänge: swaRef-typisierte Elemente” auf Seite 88).
2. Verwenden Sie die Darstellung mit RPC/Literal. In diesem Fall bestehen für die WSDL-Bindung keine Einschränkungen hinsichtlich der Anzahl der Teile, die an den SOAP-Hauptteil gebunden sind. Die resultierende SOAP-Nachricht hat immer ein einziges untergeordnetes Element, das die aufzurufende Operation darstellt; die Nachrichtenteile sind untergeordnete Elemente dieses Elements.
3. Bei der JAX-WS-Bindung sollte höchstens ein einziger Nachrichtenteil verwendet werden, der nicht den Typ 'hexBinary' oder 'base64Binary' aufweist, sofern es nicht akzeptabel ist, die anderen nicht binären Teile an SOAP-Header zu binden.
4. Alle anderen Fälle unterliegen dem beschriebenen Verhalten.

Anmerkung: Zusätzliche Einschränkungen ergeben sich, wenn Sie SOAP-Nachrichten verwenden, die nicht mit der WS-I-Spezifikation *Basic Profile Version 1.1* konform sind.

- Der erste Nachrichtenteil sollte nicht binär sein.
- Beim Empfang von mehrteiligen SOAP-Dokumentdarstellungsnachrichten mit referenzierten Anhängen erwartet die JAX-WS-Bindung, dass jeder referenzierte Anhang als untergeordnetes Element des SOAP-Hauptteils mit einem Wert für das Attribut 'href' dargestellt ist, der den Anhang durch dessen Inhalts-ID angibt. Die JAX-WS-Bindung sendet referenzierte Anhänge für solche Nachrichten auf dieselbe Weise. Dieses Verhalten ist nicht mit der WS-I-Spezifikation 'Basic Profile' kompatibel.

  Um sicherzustellen, dass Ihre Nachrichten die Spezifikation 'Basic Profile' einhalten, verwenden Sie das Verfahren 1 auf Seite 100 oder 2 auf Seite 100 in der vorstehenden Liste bzw. vermeiden Sie die Verwendung von referenzierten Anhängen für solche Nachrichten und verwenden Sie stattdessen nicht referenzierte Anhänge oder Anhänge des Typs 'swaRef'.

HTTP-Bindungen:

Die HTTP-Bindung ist so konzipiert, dass sie für HTTP die Konnektivität von Service Component Architecture (SCA) bereitstellt. Infolgedessen können bestehende oder neu entwickelte HTTP-Anwendungen in Umgebungen mit serviceorientierter Architektur (Service Oriented Architecture - SOA) eingesetzt werden.

Das Hypertext Transfer Protocol (HTTP) ist ein weit verbreitetes Protokoll für die Datenübertragung im World Wide Web. Für die Arbeit mit einer externen Anwendung, die das HTTP-Protokoll verwendet, ist eine HTTP-Bindung erforderlich. Mithilfe der HTTP-Bindung werden die Daten, die in einer Nachricht in einem nativen Format weitergegeben werden, in ein Geschäftsobjekt in einer SCA-Anwendung transformiert. Von der HTTP-Bindung können auch Daten, die als Geschäftsobjekt übergeben wurden, in das native Format umgewandelt werden, das von der externen Anwendung erwartet wird.

Anmerkung: Wenn Sie mit Clients und Services interagieren wollen, die das SOAP-/HTTP-Protokoll für Web-Services verwenden, kann es sinnvoll sein, eine der Web-Service-Bindungen zu verwenden, die in Bezug auf die Behandlung der Standardservicequalitäten für Web-Services zusätzliche Funktionalität bieten.

Nachstehend sind einige allgemeine Szenarios für die Verwendung der HTTP-Bindung beschrieben:

- Services mit SCA-Basis können HTTP-Anwendungen mithilfe eines SCA-Imports aufrufen.
- Services mit SCA-Basis können sich selbst als HTTP-fähige Anwendungen zugänglich machen, damit sie (über einen HTTP-Export) von HTTP-Clients verwendet werden können.
- IBM Business Process Manager und Process Server können über eine HTTP-Infrastruktur miteinander kommunizieren. Benutzer können daher ihre Kommunikation gemäß der unternehmensweiten Standards verwalten.
- IBM Business Process Manager und Process Server können als Mediatoren der HTTP-Kommunikation agieren, indem sie Nachrichten transformieren und weiterleiten. Dies verbessert die Integration von Anwendungen bei Verwendung eines HTTP-Netzes.
- IBM Business Process Manager und Process Server können als Brücke zwischen HTTP und anderen Protokollen wie SOAP/HTTP-Web-Services Java Connector Architecture (JCA)-basierte Ressourcenadapter, JMS usw. eingesetzt werden.

Ausführliche Informationen zum Erstellen von HTTP-Importbindungen und -Exportbindungen finden Sie im Information Center von Integration Designer. Lesen Sie dort die Themen unter **Integrationsanwendungen entwickeln** > **Mit HTTP auf externe Services zugreifen**.

Übersicht über HTTP-Bindungen:

Eine HTTP-Bindung stellt Konnektivität für Anwendungen mit HTTP-Hosting bereit. Sie nimmt eine Meditation der Kommunikation zwischen HTTP-Anwendungen vor und ermöglicht den Aufruf vorhandener HTTP-basierter Anwendungen aus einem Modul heraus.

HTTP-Importbindungen

Die HTTP-Importbindung stellt eine abgehende Konnektivität von SCA-Anwendungen zu einem HTTP-Server oder HTTP-Anwendungen bereit.

Der Import ruft eine HTTP-Endpunkt-URL auf. Zur Angabe der URL gibt es drei Verfahren:

- Die URL kann in den HTTP-Headern durch die URL für dynamisches Überschreiben dynamisch festgelegt werden.
- Die URL kann im Zieladressenelement des Servicenachrichtenobjekts dynamisch festgelegt werden.
- Die URL kann als Konfigurationseigenschaft für den Import angegeben werden.

Dieser Aufruf erfolgt immer synchron.

Obwohl HTTP-Aufrufe immer Anforderungs-/Antwortaufrufe sind, unterstützt der HTTP-Import sowohl unidirektionale als auch bidirektionale Operationen und ignoriert bei einer unidirektionalen Operation die Antwort.

HTTP-Exportbindungen

Die HTTP-Exportbindung stellt die eingehende Konnektivität von HTTP-Anwendungen zu einer SCA-Anwendung bereit.

Im HTTP-Export ist eine URL definiert. HTTP-Anwendungen, die Anforderungsnachrichten an den Export senden wollen, rufen den Export über diese URL auf.

Der HTTP-Export unterstützt auch Pingbefehle.

HTTP-Bindungen zur Laufzeit

Im Verlauf eines Imports mit einer HTTP-Bindung zur Laufzeit wird eine Anforderung mit oder ohne Daten im Hauptteil der Nachricht von der SCA-Anwendung an den externen Web-Service gesendet. Die Anforderung wird von der SCA-Anwendung an den externen Web-Service ausgegeben (siehe Abb. 26).

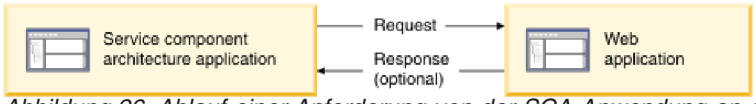


Abbildung 26. Ablauf einer Anforderung von der SCA-Anwendung an die Webanwendung

Optional können beim Import mit der HTTP-Bindung Daten von der Webanwendung in einer Antwort an die Anforderung zurückgesendet werden.

Im Verlauf eines Exports wird die Anforderung von einer Clientanwendung an einen Web-Service gesendet (siehe Abb. 27).

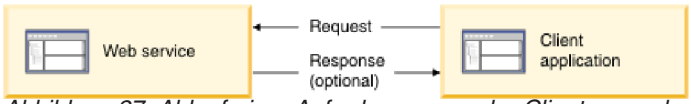


Abbildung 27. Ablauf einer Anforderung von der Clientanwendung an den Web-Service

Der Web-Service ist eine Webanwendung, die auf dem Server ausgeführt wird. Der Export wird in diese Webanwendung als Servlet implementiert, der Client sendet also seine Anforderung an eine URL-Adresse. Vom Servlet wird die Anforderung zur Laufzeit an die SCA-Anwendung weitergeleitet.

Optional können im Verlauf des Exports Daten als Antwort auf die Anforderung an die Clientanwendung gesendet werden.

HTTP-Header:

Bei HTTP-Importbindungen und -Exportbindungen können die HTTP-Header und deren Werte, die für abgehende Nachrichten verwendet werden, konfiguriert werden. Der HTTP-Import verwendet diese Header für Anforderungen. Der HTTP-Export verwendet sie für Antworten.

Statisch konfigurierte Header und Steuerinformationen haben Vorrang vor anderen Werten, die während der Laufzeit dynamisch festgelegt werden. Die Steuerwerte für die URL für dynamisches Überschreiben, die Version und die Methode überschreiben jedoch die statischen Werte, die ansonsten als Standardwerte betrachtet werden.

Die Bindung unterstützt die dynamische Spezifik der HTTP-Import-URL, indem der Wert für die HTTP-Ziel-URL, die Version und die Methode zur Laufzeit ermittelt werden. Zur Ermittlung dieser Werte wird der Wert für die Endpunktreferenz, die URL für dynamisches Überschreiben, die Version und die Methode extrahiert.

- Verwenden Sie für die Endpunktreferenz die APIs 'com.ibm.websphere.sca.addressing.EndpointReference' oder legen Sie das Feld '/headers/SMOHeader/Target/address' im Servicenachrichtenobjektheader fest.
- Für die URL für dynamisches Überschreiben, die Version und die Methode verwenden Sie den Abschnitt mit den HTTP-Steuerparametern in der SCA-Nachricht. Bitte beachten Sie, dass der Wert für die URL für dynamisches Überschreiben Vorrang vor der Zielendpunktreferenz hat. Die Endpunktreferenz gilt jedoch bindungsübergreifend. Sie stellt damit die bevorzugte Methode dar und sollte nach Möglichkeit verwendet werden.

Die Steuer- und Headerinformationen für abgehende Nachrichten unter HTTP-Exportbindungen und -Importbindungen werden in der nachstehenden Reihenfolge verarbeitet:

1. Header- und Steuerinformationen ohne HTTP-Wert für die URL für dynamisches Überschreiben, Version und Methode aus der SCA-Nachricht (niedrigste Priorität)
2. Änderungen aus der Administrationskonsole auf Export-/Importebene
3. Änderungen aus der Administrationskonsole auf der Methodenebene des Exports oder Imports
4. Zieladresse (angegeben durch Endpunktreferenz oder Servicenachrichtenobjektheader)
5. Wert für die URL für dynamisches Überschreiben, Version und Methode aus der SCA-Nachricht
6. Header und Steuerinformationen aus dem Datenhandler oder der Datenbindung (höchste Priorität)

Der HTTP-Export und -Import füllt Header und Steuerparameter in eingehender Richtung nur dann mit Daten aus der eingehenden Nachricht (HTTPExportRequest und HTTPImportResponse), wenn die Einstellung für die Weiterleitung des Protokollheaders auf **True** gesetzt ist. Analog lesen und verarbeiten der HTTP-Export und -Import abgehende Header und Steuerparameter (HTTPExportResponse und HTTPImportRequest) nur dann, wenn die Weiterleitung des Protokollheaders auf **True** gesetzt ist.

Anmerkung: Datenhandler- oder Datenbindungsänderungen an Headern oder Steuerparametern in der Importantwort oder der Exportanforderung ändern die Verarbeitungsanweisungen der Nachricht innerhalb der Import- oder Exportbindung nicht und sollten ausschließlich zu dem Zweck eingesetzt werden, geänderte Werte an nachgelagerte SCA-Komponenten weiterzugeben.

Der Kontextservice ist für die Weitergabe des Kontextes (inklusive der Protokollheader wie dem HTTP-Header und des Benutzerkontextes wie der Account-ID) über einen SCA-Aufrufpfad zuständig. Während

der Entwicklung in IBM Integration Designer können Sie die Weitergabe des Kontextes mittels Import- und Exporteigenschaften steuern. Weitere Details enthalten die Angaben über Import- und Exportbindungen im Information Center von IBM Integration Designer.

Bereitgestellte HTTP-Headerstrukturen und Unterstützung

In Tabelle 34 sind die Elemente für die Anforderungs-/Antwortparameter bei Anforderungen und Antworten von HTTP-Importen und -Exporten angegeben.

Tabelle 34. Bereitgestellte HTTP-Headerinformationen

Steuername	HTTP-Importanforderung	HTTP-Importantwort	HTTP-Exportanforderung	HTTP-Exportantwort
URL	Wird ignoriert.	Nicht festgelegt.	Wird aus der Anforderungsnachricht gelesen. **Anmerkung:** Die Abfragezeichenfolge ist ebenfalls Bestandteil des URL-Steuerparameters.	Wird ignoriert.
Version (mögliche Werte: 1.0, 1.1; Standardwert ist 1.1)	Wird ignoriert.	Nicht festgelegt.	Wird aus der Anforderungsnachricht gelesen.	Wird ignoriert.
Methode	Wird ignoriert.	Nicht festgelegt.	Wird aus der Anforderungsnachricht gelesen.	Wird ignoriert.
URL für dynamisches Überschreiben	Überschreibt die HTTP-Import-URL, falls im Datenhandler oder in der Datenbindung festgelegt. Wird in der Anforderungszeile in die Nachricht geschrieben. **Anmerkung:** Die Abfragezeichenfolge ist ebenfalls Bestandteil des URL-Steuerparameters.	Nicht festgelegt.	Nicht festgelegt.	Wird ignoriert.
Version für dynamisches Überschreiben	Überschreibt die HTTP-Importversion, falls festgelegt. Wird in der Anforderungszeile in die Nachricht geschrieben.	Nicht festgelegt.	Nicht festgelegt.	Wird ignoriert.
Methode für dynamisches Überschreiben	Überschreibt die HTTP-Importmethode, falls festgelegt. Wird in der Anforderungszeile in die Nachricht geschrieben.	Nicht festgelegt.	Nicht festgelegt.	Wird ignoriert.

Tabelle 34. Bereitgestellte HTTP-Headerinformationen (Forts.)

Steuername	HTTP-Importanforderung	HTTP-Importantwort	HTTP-Exportanforderung	HTTP-Exportantwort
Medientyp (dieser Steuerparameter überträgt einen Teil des Wertes für den HTTP-Header 'Content-Type')	Wird als Teil des Headers 'Content-Type' in die Nachricht geschrieben, falls vorhanden. **Anmerkung:** Dieser Steuerelementwert sollte vom Datenhandler oder von der Datenbindung bereitgestellt werden.	Wird aus dem Header 'Content-Type' der Antwortnachricht gelesen.	Wird aus dem Header 'Content-Type' der Anforderungsnachricht gelesen.	Wird als Teil des Headers 'Content-Type' in die Nachricht geschrieben, falls vorhanden. **Anmerkung:** Dieser Steuerelementwert sollte vom Datenhandler oder von der Datenbindung bereitgestellt werden.
Zeichensatz (Standardwert: UTF-8)	Wird als Teil des Headers 'Content-Type' in die Nachricht geschrieben, falls vorhanden. **Anmerkung:** Dieser Steuerelementwert sollte von der Datenbindung bereitgestellt werden.	Wird aus dem Header 'Content-Type' der Antwortnachricht gelesen.	Wird aus dem Header 'Content-Type' der Anforderungsnachricht gelesen.	Unterstützt; wird als Teil des Headers 'Content-Type' in die Nachricht geschrieben. **Anmerkung:** Dieser Steuerelementwert sollte von der Datenbindung bereitgestellt werden.
Übertragungsverschlüsselung (Mögliche Werte: chunked, identity; Standardwert ist identity)	Wird, falls vorhanden, als Header in die Nachricht geschrieben und steuert, wie die Nachrichtentransformation verschlüsselt wird.	Wird aus der Antwortnachricht gelesen.	Wird aus der Anforderungsnachricht gelesen.	Wird, falls vorhanden, als Header in die Nachricht geschrieben und steuert, wie die Nachrichtentransformation verschlüsselt wird.
Inhaltsverschlüsselung (Mögliche Werte: gzip, x-gzip, deflate, identity; Standardwert ist identity)	Wird, falls vorhanden, als Header in die Nachricht geschrieben und steuert, wie die Nutzdaten verschlüsselt werden.	Wird aus der Antwortnachricht gelesen.	Wird aus der Anforderungsnachricht gelesen.	Wird, falls vorhanden, als Header in die Nachricht geschrieben und steuert, wie die Nutzdaten verschlüsselt werden.
Inhaltslänge (Content-Length)	Wird ignoriert.	Wird aus der Antwortnachricht gelesen.	Wird aus der Anforderungsnachricht gelesen.	Wird ignoriert.
Statuscode (Standardwert: 200)	Wird nicht unterstützt.	Wird aus der Antwortnachricht gelesen.	Wird nicht unterstützt.	Wird, falls vorhanden, in der Antwortzeile in die Nachricht geschrieben.
ReasonPhrase (Standardwert: OK)	Wird nicht unterstützt.	Wird aus der Antwortnachricht gelesen.	Wird nicht unterstützt.	Der Steuerwert wird ignoriert. Der Wert für die Antwortzeile der Nachricht wird aus dem Statuscode generiert.

Tabelle 34. Bereitgestellte HTTP-Headerinformationen (Forts.)

Steuername	HTTP-Importanforderung	HTTP-Importantwort	HTTP-Exportanforderung	HTTP-Exportantwort
Authentifizierung (enthält mehrere Eigenschaften)	Wird, falls vorhanden, zum Erstellen des Basisauthentifizierungsheaders verwendet. **Anmerkung:** Der Wert für diesen Header wird nur im HTTP-Protokoll verschlüsselt. Bei SCA wird er entschlüsselt und als Klartext übergeben.	Nicht anwendbar	Wird aus dem Basisauthentifizierungsheader der Anforderungsnachricht gelesen. Das Vorhandensein dieses Headers gibt keinen Aufschluss darüber, ob der Benutzer authentifiziert wurde. die Authentifizierung sollte in der Servletkonfiguration gesteuert werden. **Anmerkung:** Der Wert für diesen Header wird nur im HTTP-Protokoll verschlüsselt. Bei SCA wird er entschlüsselt und als Klartext übergeben.	Nicht anwendbar
Proxy (enthält mehrere Eigenschaften: Host, Port, Authentifizierung)	Wird, falls vorhanden, zum Herstellen der Verbindung über den Proxy verwendet.	Nicht anwendbar	Nicht anwendbar	Nicht anwendbar
SSL (enthält mehrere Eigenschaften: Schlüsselspeicher, Schlüsselspeicherkennwort, Truststore, Truststore-Kennwort, Clientauthentifizierung)	Wenn dieser Wert angegeben ist und HTTPS als Ziel-URL verwendet wird, wird hiermit eine Verbindung über SSL aufgebaut.	Nicht anwendbar	Nicht anwendbar	Nicht anwendbar

HTTP-Datenbindungen:

Für jede unterschiedliche Zuordnung von Daten zwischen einer SCA-Nachricht und einer HTTP-Protokollnachricht muss ein Datenhandler oder eine HTTP-Datenbindung konfiguriert werden. Datenhandler stellen eine bindungsneutrale Schnittstelle zur Verfügung, die eine transportbindungsübergreifende Wiederverwendung ermöglicht. Die Verwendung von Datenhandlern wird empfohlen, da Datenbindungen speziell für eine bestimmte Transportbindung gelten. HTTP-spezifische Datenbindungsklassen werden bereitgestellt. Sie können aber auch benutzerdefinierte Datenhandler oder Datenbindungen schreiben.

Anmerkung: Die drei im vorliegenden Thema beschriebenen HTTP-Datenbindungsklassen (HTTPStreamDataBindingSOAP, HTTPStreamDataBindingXML und HTTPServiceGatewayDataBinding) werden ab IBM Business Process Manager Version 7.0 nicht mehr unterstützt. Anstelle der hier beschriebenen Datenbindungen sollten die folgenden Datenhandler verwendet werden:

- SOAPDataHandler anstelle von HTTPStreamDataBindingSOAP.
- UTF8XMLDataHandler anstelle von HTTPStreamDataBindingXML
- GatewayTextDataHandler anstelle von HTTPServiceGatewayDataBinding

Datenbindungen werden für die Verwendung bei HTTP-Importen und HTTP-Exporten bereitgestellt: Binärdatenbindung, XML-Datenbindung und SOAP-Datenbindung. Eine Antwortdatenbindung ist bei uni-

direktionalen Operationen nicht erforderlich. Eine Datenbindung wird durch den Namen einer Java-Klasse repräsentiert, deren Instanzen ein HTTP-Objekt in ein Servicedatenobjekt konvertieren können (und umgekehrt). Bei einem Export muss ein Funktionsselektor verwendet werden, der (zusammen mit Methodenbindungen) ermitteln kann, welche Datenbindung verwendet und welche Operation aufgerufen wird. Die folgenden Datenbindungen werden bereitgestellt:

- Binärdatenbindungen, die den Hauptteil wie unstrukturierte binäre Daten behandeln. Das XSD-Schema der Binärdatenbindung hat die folgende Implementierung:

```
<xsd:schema elementFormDefault="qualified"
  targetNamespace="http://com.ibm.websphere.http.data.bindings/schema"
  xmlns:tns="http://com.ibm.websphere.http.data.bindings/schema"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">

  <xsd:complexType name="HTTPBaseBody">
    <xsd:sequence/>
  </xsd:complexType>

  <xsd:complexType name="HTTPBytesBody">
    <xsd:complexContent>
      <xsd:extension base="tns:HTTPBaseBody">
        <xsd:sequence>
          <xsd:element name="value" type="xsd:hexBinary"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
```

- XML-Datenbindungen, die den Hauptteil wie XML-Daten behandeln. Die Implementierung der XML-Datenbindung hat Ähnlichkeit mit der JMS-XML-Datenbindung und weist keine Beschränkungen hinsichtlich des Schnittstellenschemas auf.
- SOAP-Datenbindungen, die den Hauptteil als SOAP-Daten unterstützen. Die Implementierung der SOAP-Datenbindung weist keine Beschränkungen hinsichtlich des Schnittstellenschemas auf.

Benutzerdefinierte HTTP-Datenbindungen implementieren

In diesem Abschnitt ist beschrieben, wie eine benutzerdefinierte HTTP-Datenbindung implementiert werden kann.

Anmerkung: Die empfohlene Strategie besteht in der Implementierung eines benutzerdefinierten Datenhandlers, da dieser transportbindungsübergreifend wiederverwendet werden kann.

HTTPStreamDataBinding ist die Hauptschnittstelle für die Behandlung von benutzerdefinierten HTTP-Nachrichten. Die Schnittstelle ist so konzipiert, dass sie die Verarbeitung umfangreicher Nutzdaten zulässt. Damit eine solche Implementierung funktioniert, muss diese Datenbindung die Steuerinformationen und Header zurückgeben, bevor die Nachricht in den Datenstrom geschrieben wird.

Die Methoden und ihre Ausführungsreihenfolge (nachfolgend aufgeführt) müssen durch die benutzerdefinierte Datenbindung implementiert werden.

Zum Anpassen einer Datenbindung schreiben Sie eine Klasse, die 'HTTPStreamDataBinding' implementiert. Die Datenbindung sollte vier private Eigenschaften besitzen:

- private DataObject pDataObject
- private HTTPControl pCtrl
- private HTTPHeaders pHeaders
- private yourNativeDataType nativeData

Die HTTP-Bindung verwendet beim Aufruf der benutzerdefinierten Datenbindung die folgende Reihenfolge:

- Ausgangsverarbeitung (Datenobjekt in natives Format):

1. setDataObject(...)
2. setHeaders(...)
3. setControlParameters(...)
4. setBusinessException(...)
5. convertToNativeData()
6. getControlParameters()
7. getHeaders()
8. write(...)

- Eingangsverarbeitung (natives Format in Datenobjekt):
  1. setControlParameters(...)
  2. setHeaders(...)
  3. convertFromNativeData(...)
  4. isBusinessException()
  5. getDataObject()
  6. getControlParameters()
  7. getHeaders()

Sie müssen die Methode 'setDataObject(...)' in der Methode 'convertFromNativeData(...)' aufrufen, um den Wert des Datenobjekts (dataObject) festzulegen, das aus nativen Daten in die private Eigenschaft 'pDataObject' konvertiert wird.

```
public void setDataObject(DataObject dataObject)
   throws DataBindingException {
  pDataObject = dataObject;

}
public void setControlParameters(HTTPControl arg0) {
  this.pCtrl = arg0;
}

public void setHeaders(HTTPHeaders arg0) {
  this.pHeaders = arg0;
}
/*
* Add http header "IsBusinessException" in pHeaders.
* Two steps:
* 1.Remove all the header with name IsBusinessException (case-insensitive) first.
*   This is to make sure only one header is present.
* 2.Add the new header "IsBusinessException"
*/
public void setBusinessException(boolean isBusinessException) {
  //remove all the header with name IsBusinessException (case-insensitive) first.
  //This is to make sure only one header is present.
  //add the new header "IsBusinessException", code example:
  HTTPHeader header=HeadersFactory.eINSTANCE.createHTTPHeader();
  header.setName("IsBusinessException");
  header.setValue(Boolean.toString(isBusinessException));
  this.pHeaders.getHeader().add(header);

}
public HTTPControl getControlParameters() {
  return pCtrl;
}
public HTTPHeaders getHeaders() {
  return pHeaders;
}
public DataObject getDataObject() throws DataBindingException {
  return pDataObject;
}
```

```
/*
* Get header "IsBusinessException" from pHeaders, return its boolean value
*/
public boolean isBusinessException() {
  String headerValue = getHeaderValue(pHeaders,"IsBusinessException");
  boolean result=Boolean.parseBoolean(headerValue);
  return result;
}
public void convertToNativeData() throws DataBindingException {
  DataObject dataObject = getDataObject();
  this.nativeData=realConvertWorkFromSDOToNativeData(dataObject);
}
public void convertFromNativeData(HTTPInputStream arg0){
  //Customer-developed method to
  //Read data from HTTPInputStream
  //Convert it to DataObject
  DataObject dataobject=realConvertWorkFromNativeDataToSDO(arg0);
  setDataObject(dataobject);
}
public void write(HTTPOutputStream output) throws IOException {
  if (nativeData != null)
  output.write(nativeData);
}
```

EJB-Bindungen:

EJB-Importbindungen (EJB = Enterprise JavaBeans) ermöglichen SCA-Komponenten den Aufruf von Services, die durch Java EE-Geschäftslogik auf einem Java EE-Server bereitgestellt werden. Mittels EJB-Exportbindungen können SCA-Komponenten als Enterprise JavaBeans zugänglich gemacht werden, damit Java EE-Geschäftslogik SCA-Komponenten aufrufen kann, die andernfalls für sie nicht verfügbar wären.

EJB-Importbindungen:

Mit EJB-Importbindungen kann ein SCA-Modul EJB-Implementierungen aufrufen, indem das Verfahren angegeben wird, mit dem das verarbeitende Modul an die externe EJB gebunden ist. Durch das Importieren von Services aus einer externen EJB-Implementierung können Benutzer ihre Geschäftslogik in die IBM Business Process Manager-Umgebung integrieren und an einem Geschäftsprozess mitwirken.

Zum Erstellen von EJB-Importbindungen verwenden Sie Integration Designer. Bei der Generierung der Bindungen können Sie eines der folgenden Verfahren verwenden:

- EJB-Import mit dem Assistenten 'Externer Service' erstellen

  Mit dem Assistenten 'Externer Service' können Sie in Integration Designer auf der Basis einer vorhandenen Implementierung einen EJB-Import erstellen. Der Assistent 'Externer Service' erstellt Services anhand der von Ihnen angegebenen Kriterien. Anschließend generiert er auf der Grundlage der erkannten Services Geschäftsobjekte, Schnittstellen und Importdateien.
- EJB-Import mit dem Assembly-Editor erstellen

  Zum Erstellen eines EJB-Imports können Sie den Assembly-Editor von Integration Designer verwenden. Sie können in der Palette entweder einen Import oder eine Java-Klasse verwenden, um die EJB-Bindung zu erstellen.

Der generierte Import besitzt Datenbindungen, die die Java-WSDL-Verbindung herstellen und keine Java-Brückenkomponente erforderlich machen. Sie können eine Komponente mit einer WSDL-Referenz direkt mit dem EJB-Import verbinden, der über eine Java-Schnittstelle mit einem EJB-basierten Service kommuniziert.

Der EJB-Import kann mit Java EE-Geschäftslogik entweder über das EJB 2.1- oder das EJB 3.0-Programmiermodell interagieren.

Der Aufruf der Java EE-Geschäftslogik kann lokal (nur bei EJB 3.0) oder über Fernzugriff erfolgen.

- Der lokale Aufruf wird verwendet, wenn Java EE-Geschäftslogik aufgerufen werden soll, die sich auf demselben Server wie der Import befindet.

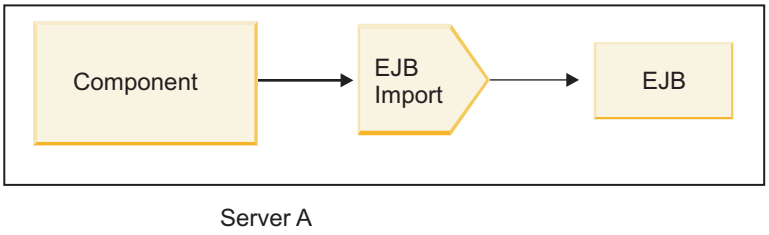


Abbildung 28. Lokaler Aufruf einer EJB (nur bei EJB 3.0)

- Der Fernaufruf wird verwendet, wenn Java EE-Geschäftslogik aufgerufen werden soll, die sich nicht auf demselben Server wie der Import befindet.

  In der folgenden Abbildung ist beispielsweise ein EJB-Import dargestellt, der eine EJB-Methode auf einem anderen Server über RMI/IIOP (Remote Method Invocation over Internet InterORB Protocol) aufruft.

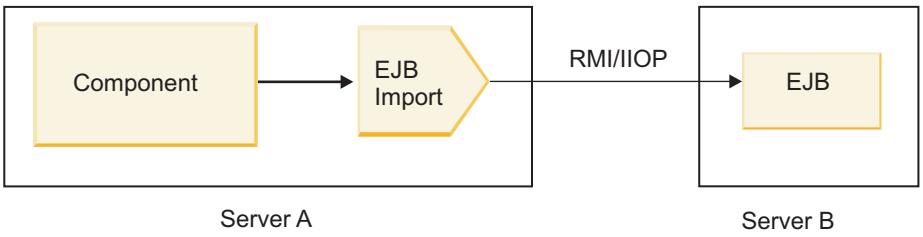


Abbildung 29. Aufruf einer EJB über Fernzugriff

Beim Konfigurieren der EJB-Bindung verwendet Integration Designer den JNDI-Namen, um die Version des EJB-Programmiermodells und den Typ des Aufrufs (lokal oder fern) zu ermitteln.

EJB-Importbindungen können die folgenden Hauptkomponenten enthalten:
- JAX-WS-Datenhandler
- EJB-Fehlerselektor
- EJB-Importfunktionsselektor

Falls Ihr Benutzerszenario nicht auf der JAX-WS-Zuordnung basiert, benötigen Sie möglicherweise einen angepassten Datenhandler, Funktionsselektor und Fehlerselektor, um die Tasks auszuführen, die andernfalls durch die Komponenten ausgeführt werden, die Bestandteil der EJB-Importbindungen sind. Hierzu gehört auch die Zuordnung, die normalerweise durch den benutzerdefinierten Zuordnungsalgorithmus vorgenommen wird.

EJB-Exportbindungen:

Externe Java EE-Anwendungen können über eine EJB-Exportbindung eine SCA-Komponente aufrufen. Durch einen EJB-Export können Sie SCA-Komponenten zugänglich machen, damit externe Java EE-Anwendungen diese Komponenten unter Verwendung des EJB-Programmiermodells aufrufen können.

Anmerkung: Der EJB-Export ist eine Stateless Bean (Bean ohne Statusaufzeichnung).

Zum Erstellen von EJB-Bindungen verwenden Sie Integration Designer. Bei der Generierung der Bindungen können Sie eines der folgenden Verfahren verwenden:
- EJB-Exportbindungen mit dem Assistenten 'Externer Service' erstellen

Mit dem Assistenten 'Externer Service' können Sie in Integration Designer auf der Basis einer vorhandenen Implementierung einen EJB-Exportservice erstellen. Der Assistent 'Externer Service' erstellt Services anhand der von Ihnen angegebenen Kriterien. Anschließend generiert er auf der Grundlage der erkannten Services Geschäftsobjekte, Schnittstellen und Exportdateien.

- EJB-Exportbindungen mit dem Assembly-Editor erstellen

  Zum Erstellen einer EJB-Exportbindung können Sie den Assembly-Editor von Integration Designer verwenden.

Wichtig: Ein J2SE-Client (J2SE = Java 2 Platform, Standard Edition) kann den in Integration Designer generierten EJB-Exportclient nicht aufrufen.

Sie können die Bindung ausgehend von einer vorhandenen SCA-Komponente generieren oder Sie können einen Export mit einer EJB-Bindung für eine Java-Schnittstelle generieren.

- Wenn Sie einen Export für eine vorhandene SCA-Komponente generieren, die mit einer vorhandenen WSDL-Schnittstelle ausgestattet ist, wird dem Export eine Java-Schnittstelle zugeordnet.
- Wenn Sie einen Export für eine Java-Schnittstelle erstellen, können Sie entweder eine WSDL- oder eine Java-Schnittstelle für den Export auswählen.

  Anmerkung: Eine Java-Schnittstelle, die zum Erstellen eines EJB-Exports verwendet wurde, unterliegt hinsichtlich der Objekte (Eingabe- und Ausgabeparameter sowie Ausnahmebedingungen), die als Parameter an einen fernen Aufruf übergeben werden, den folgenden Einschränkungen:
  - Die Objekte müssen einen konkreten Typ aufweisen (anstelle eines Schnittstellen- oder abstrakten Typs).
  - Die Objekte müssen mit der Enterprise JavaBeans-Spezifikation konform sein. Sie müssen serialisierbar sein und den Standardkonstruktor ohne Argument (no-argument) besitzen. Darüber hinaus muss auf alle Eigenschaften mit Getter- und Setter-Methoden zugegriffen werden können.

    Informationen zur Enterprise JavaBeans-Spezifikation finden Sie auf der Website von Sun Microsystems, Inc. unter der Adresse http://java.sun.com.

  Zusätzlich muss es sich bei der Ausnahmebedingung um eine geprüfte Ausnahmebedingung handeln, die von 'java.lang.Exception' übernommen wurde. Die Ausnahmebedingung muss zudem singulär sein (darf also nicht die Auslösung mehrerer Typen von geprüften Ausnahmebedingungen unterstützen).

  Bitte beachten Sie ebenfalls, dass eine Geschäftsschnittstelle einer Java EnterpriseBean eine normale Java-Schnittstelle ist und 'javax.ejb.EJBObject' oder 'javax.ejb.EJBLocalObject' nicht erweitern darf. Die Methoden der Geschäftsschnittstelle sollten keine Ausnahmebedingungen des Typs 'java.rmi.Remote-.Exception' auslösen.

Die EJB-Exportbindungen können mit Java EE-Geschäftslogik entweder über das EJB 2.1- oder das EJB 3.0-Programmiermodell interagieren.

Der Aufruf kann lokal (nur bei EJB 3.0) oder über Fernzugriff erfolgen.

- Der lokale Aufruf wird verwendet, wenn die Java EE-Geschäftslogik eine SCA-Komponente aufruft, die sich auf demselben Server wie der Export befindet.
- Der Fernaufruf wird verwendet, wenn sich die Java EE-Geschäftslogik nicht auf demselben Server befindet wie der Export.

  In der folgenden Abbildung ist beispielsweise eine EJB dargestellt, die RMI/IIOP verwendet, um eine SCA-Komponente auf einem anderen Server aufzurufen.

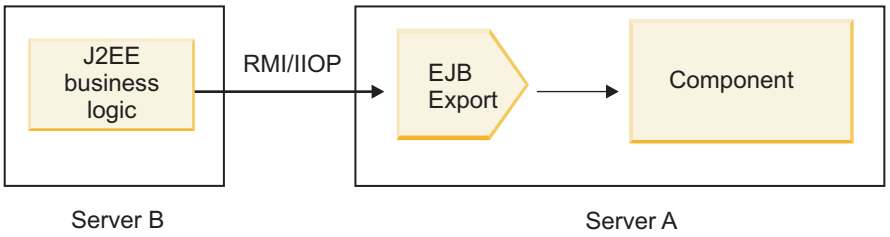


Abbildung 30. Fernaufruf von einem Client an eine SCA-Komponente mittels eines EJB-Exports

Beim Konfigurieren der EJB-Bindung verwendet Integration Designer den JNDI-Namen, um die Version des EJB-Programmiermodells und den Typ des Aufrufs (lokal oder fern) zu ermitteln.

EJB-Exportbindungen können die folgenden Hauptkomponenten enthalten:

- JAX-WS-Datenhandler
- EJB-Exportfunktionsselektor

Falls Ihr Benutzerszenario nicht auf der JAX-WS-Zuordnung basiert, benötigen Sie möglicherweise einen angepassten Datenhandler und Funktionsselektor, um die Tasks auszuführen, die andernfalls durch die Komponenten ausgeführt werden, die Bestandteil der EJB-Exportbindungen sind. Hierzu gehört auch die Zuordnung, die normalerweise durch den benutzerdefinierten Zuordnungsalgorithmus vorgenommen wird.

Eigenschaften von EJB-Bindungen:

EJB-Importbindungen verwenden ihre konfigurierten JNDI-Namen, um die Version des EJB-Programmiermodells und den Typ des Aufrufs (lokal oder fern) zu bestimmen. EJB-Importbindungen und -Exportbindungen verwenden zur Datentransformation den JAX-WS-Datenhandler. Die EJB-Importbindung verwendet einen EJB-Importfunktionsselektor und einen EJB-Fehlerselektor. Die EJB-Exportbindung verwendet einen Exportfunktionsselektor.

JNDI-Namen und EJB-Importbindungen:

Beim Konfigurieren der EJB-Bindung für einen Import verwendet Integration Designer den JNDI-Namen, um die Version des EJB-Programmiermodells und den Typ des Aufrufs (lokal oder fern) zu ermitteln.

Falls kein JNDI-Name angegeben ist, wird die Bindung für die EJB-Standardschnittstelle verwendet. Die erstellten Standardnamen sind davon abhängig, ob JavaBeans gemäß EJB 2.1 oder JavaBeans gemäß EJB 3.0 aufgerufen wird.

Anmerkung: Ausführliche Informationen zu den Namenskonventionen finden Sie in der Übersicht über EJB 3.0-Anwendungsbindungen im Information Center von WebSphere Application Server.

- JavaBeans gemäß EJB 2.1

  Der durch Integration Designer vorausgewählte standardmäßige JNDI-Name ist die EJB 2.1-Standardbindung. Diese hat das Format **ejb/** zuzüglich der Home-Schnittstelle (durch Schrägstriche getrennt).

  Für die Home-Schnittstelle von EJB 2.1 JavaBeans für 'com.mycompany.myremotebusinesshome' lautet die Standardbindung beispielsweise:

  ```
  ejb/com/mycompany/myremotebusinesshome
  ```

  Bei EJB 2.1 wird nur der ferne EJB-Aufruf unterstützt.

- JavaBeans gemäß EJB 3.0

  Der von Integration Designer für den lokalen JNDI-Namen vorausgewählte standardmäßige JNDI-Name ist der vollständig qualifizierte Klassenname der lokalen Schnittstelle, dem die Angabe **ejblocal:** vorangestellt ist. Für die vollständig qualifizierte Schnittstelle der lokalen Schnittstelle 'com.mycompany-.mylocalbusiness' wird beispielsweise der folgende EJB 3.0-JNDI-Name vorausgewählt:

```
ejblocal:com.mycompany.mylocalbusiness
```

Für die ferne Schnittstelle 'com.mycompany.myremotebusiness' ist der vorausgewählte EJB 3.0-JNDI-Name die vollständig qualifizierte Schnittstelle:

```
com.mycompany.myremotebusiness
```

Die EJB 3.0-Standard-Anwendungsbindungen werden unter der folgenden Adresse beschrieben: EJB 3.0-Anwendungsbindungen - Übersicht.

Integration Designer verwendet den Kurznamen als JNDI-Standardposition für EJBs, die das Programmiermodell von Version 3.0 verwenden.

Anmerkung: Falls die implementierte JNDI-Referenz der Ziel-EJB von der EJB-Standardbindungsposition abweicht, weil eine benutzerdefinierte Zuordnung verwendet oder konfiguriert wurde, muss der JNDI-Zielname richtig angegeben werden. Sie können den Namen in Integration Designer vor der Implementierung angeben. Bei der Importbindung haben Sie auch die Möglichkeit, den Namen (nach der Implementierung) in der Administrationskonsole so zu ändern, dass er mit dem JNDI-Namen der Ziel-EJB übereinstimmt.

Weitere Informationen zum Erstellen von EJB-Bindungen enthalten die Abschnitte über die Arbeit mit EJB-Bindungen im Information Center von Integration Designer.

JAX-WS-Datenhandler:

Die EJB-Importbindung (EJB = Enterprise JavaBeans) verwendet den JAX-WS-Datenhandler, um Anforderungsgeschäftsobjekte in Java-Objektparameter und den Rückgabewert des Java-Objekts in ein Antwortgeschäftsobjekt umzuwandeln. Die EJB-Exportbindung verwendet den JAX-WS-Datenhandler, um Anforderungs-EJBs in Anforderungsgeschäftsobjekte und die Antwortgeschäftsobjekte in einen Rückgabewert umzuwandeln.

Dieser Datenhandler ordnet Daten aus einer mit SCA angegebenen WSDL-Schnittstelle einer EJB-Java-Zielschnittstelle zu (und umgekehrt), unter Verwendung der Spezifikation von Java API for XML Web Services (JAX-WS) und der Spezifikation von Java Architecture for XML Binding (JAXB).

Anmerkung: Die Unterstützung ist gegenwärtig auf die Spezifikationen von JAX-WS 2.1.1 und JAXB 2.1.3 beschränkt.

Mit dem Datenhandler, der auf der Ebene der EJB-Bindung angegeben ist, wird die Verarbeitung von Anforderungen, Antworten, Fehlern und Laufzeitausnahmebedingungen ausgeführt.

Anmerkung: Für Fehler kann durch Angabe der Konfigurationseigenschaft 'faultBindingType' ein spezieller Datenhandler für jeden Fehler angegeben werden. Dies überschreibt den Wert, der auf der Ebene der EJB-Bindung angegeben ist.

Der JAX-WS-Datenhandler wird standardmäßig verwendet, wenn die EJB-Bindung eine WSDL-Schnittstelle besitzt. Dieser Datenhandler kann nicht verwendet werden, um eine SOAP-Nachricht zu transformieren, die einen JAX-WS-Aufruf eines Datenobjektes darstellt.

Die EJB-Importbindung verwendet einen Datenhandler, um ein Datenobjekt in einen Java-Objekt-Array (Object[]) zu transformieren. Während der abgehenden Kommunikation findet die folgende Verarbeitung statt:

1. Die EJB-Bindung legt den erwarteten Typ, das erwartete Element und den angestrebten Methodennamen im Element 'BindingContext' übereinstimmend mit den in WSDL angegebenen Elementen fest.
2. Die EJB-Bindung ruft die Transformationsmethode für das Datenobjekt auf, bei dem eine Datentransformation erfolgen muss.
3. Der Datenhandler gibt einen Array 'Object[]' zurück, der die Parameter der Methode (in der Reihenfolge ihrer Definition in der Methode) darstellt.

4. Die EJB-Bindung verwendet den Array 'Object[]', um die Methode für die EJB-Zielschnittstelle aufzurufen.

Die Bindung bereitet außerdem einen Array 'Object[]' zur Verarbeitung der Antwort für den EJB-Aufruf vor.

- Das erste Element im Array 'Object[]' ist der Rückgabewert vom Java-Methodenaufruf.
- Die nachfolgenden Werte stellen die Eingabeparameter für die Methode dar.

Dies ist erforderlich, damit Parameter des Typs 'Ein-/Ausgabe' sowie 'Ausgabe' unterstützt werden.

Bei Parametern des Typs 'Ausgabe' müssen die Werte im Antwortdatenobjekt zurückgegeben werden.

Der Datenhandler verarbeitet und transformiert Werte, die im Array 'Object[]' gefunden wurden, und gibt dann eine Antwort an das Datenobjekt zurück.

Der Datenhandler unterstützt neben anderen XSD-Datentypen auch 'xs:AnyType', 'xs:AnySimpleType' und 'xs:Any'. Um die Unterstützung für 'xs:Any' zu aktivieren, verwenden Sie die Angabe **@XmlAnyElement (lax=true)** für die JavaBeans-Eigenschaft im Java-Code wie im folgenden Beispiel gezeigt:

```
public class TestType {
 private Object[] object;

 @XmlAnyElement (lax=true)
 public Object[] getObject() {
  return object;
 )

 public void setObject (Object[] object) {
  this.object=object;
 )
)
```

Hierdurch wird das Eingeschaftsobjekt in 'TestType' zu einem Feld des Typs 'xs:any'. Der im Feld des Typs 'xs:any' verwendete Java-Klassenwert sollte mit der Annotation **@XmlAnyElement** versehen sein. Beispiel: Falls der Objektarray unter Verwendung der Java-Klasse 'Address' gefüllt wird, sollte die Klasse 'Address' mit der Annotation **@XmlRootElement** versehen sein.

Anmerkung: Um die Zuordnung des XSD-Typs zu den Java-Typen anzupassen, die in der JAX-WS-Spezifikation definiert sind, ändern Sie die JAXB-Annotationen so, dass Ihre Geschäftsanforderungen erfüllt werden. Der JAX-WS-Datenhandler unterstützt 'xs:any', 'xs:anyType' und 'xs:anySimpleType'.

Für den JAX-WS-Datenhandler gelten die folgenden Einschränkungen:

- Der Datenhandler bietet keine Unterstützung für die Headerattributannotation **@WebParam**.
- Der Namespace für Geschäftsobjektschemadateien (XSD-Dateien) enthält keine Standardzuordnung für den Java-Paketnamen. Die Annotation **@XMLSchema** in der Datei 'package-info.java' kann ebenfalls nicht verwendet werden. Die einzige Möglichkeit zur Erstellung von XSD mit einem Namespace besteht in der Verwendung der Annotationen **@XmlType** und **@XmlRootElement**. Die Annotation **@XmlRootElement** definiert den Zielnamespace für das globale Element in JavaBeans-Typen.
- Der EJB-Importassistent erstellt keine XSD-Dateien für zusammenhangslose Klassen. Version 2.0 unterstützt die Annotation **@XmlSeeAlso** nicht, weshalb kein XSD erstellt wird, falls die untergeordnete Klasse nicht direkt aus der übergeordneten Klasse referenziert wird. Dieses Problem kann durch die Ausführung von 'SchemaGen' für solche untergeordneten Klassen gelöst werden.

  Bei 'SchemaGen' handelt es sich um ein Befehlszeilendienstprogramm (Position ist das Verzeichnis '*WPS-installationsausgangsverzeichnis*/bin'), das zum Erstellen von XSD-Dateien für eine bestimmte Bean bereitgestellt wird. Diese XSD-Dateien müssen manuell in das Modul kopiert werden, damit das Modul funktionsfähig ist.

EJB-Fehlerselektor:

Der EJB-Fehlerselektor ermittelt, ob ein EJB-Aufruf zu einem Fehler, einer Laufzeitausnahmebedingung oder einer erfolgreichen Antwort geführt hat.

Falls ein Fehler festgestellt wird, gibt der EJB-Fehlerselektor den nativen Fehlernamen an die Bindungslaufzeit zurück, damit der JAX-WS-Datenhandler das Ausnahmebedingungsobjekt in ein Fehlergeschäftsobjekt konvertieren kann.

Bei einer erfolgreichen (fehlerfreien) Antwort assembliert die EJB-Importbindung einen Java-Objektarray (Object[]), um die Werte zurückzugeben.

- Das erste Element im Array 'Object[]' ist der Rückgabewert vom Java-Methodenaufruf.
- Die nachfolgenden Werte stellen die Eingabeparameter für die Methode dar.

Dies ist erforderlich, damit Parameter des Typs 'Ein-/Ausgabe' sowie 'Ausgabe' unterstützt werden.

In Szenarios mit Ausnahmebedingungen assembliert die Bindung einen Array 'Object[]' und das erste Element stellt die durch die Methode ausgelöste Ausnahmebedingung dar.

Der Fehlerselektor kann einen der folgenden Werte zurückgeben:

Tabelle 35. Rückgabewerte

Typ	Rückgabewert	Beschreibung
Fehler	**ResponseType.FAULT**	Wird zurückgegeben, wenn der übergebene Array 'Object[]' ein Ausnahmebedingungsobjekt enthält.
Laufzeitausnahmebedingung	**ResponseType.RUNTIME**	Wird zurückgegeben, falls das Ausnahmebedingungsobjekt mit keinem der für die Methode deklarierten Ausnahmebedingungstypen übereinstimmt.
Normale Antwort	**ResponseType.RESPONSE**	Wird in allen anderen Fällen zurückgegeben.

Falls der Fehlerselektor den Wert **ResponseType.FAULT** zurückgibt, wird der native Fehlername zurückgegeben. Anhand dieses nativen Fehlernamens ermittelt die Bindung den korrespondierenden WSDL-Fehlernamen und ruft den entsprechenden Fehlerdatenhandler auf.

EJB-Funktionsselektor:

Die EJB-Bindungen können einen Importfunktionsselektor (für die Ausgangsverarbeitung) oder einen Exportfunktionsselektor (für die Eingangsverarbeitung) verwenden, um die aufzurufende EJB-Methode zu ermitteln.

Importfunktionsselektor

Bei der Ausgangsverarbeitung leitet der Importfunktionsselektor den Typ der EJB-Methode basierend auf dem Namen der Operation ab, die durch die mit dem EJB-Import verbundene SCA-Komponente aufgerufen wird. Der Funktionsselektor sucht nach der Annotation '@WebMethod' für die in Integration Designer generierte JAX-WS-annotierte Java-Klasse, um den zugeordneten Zieloperationsnamen zu ermitteln.

- Falls die Annotation '@WebMethod' vorhanden ist, stellt der Funktionsselektor mithilfe der Annotation '@WebMethod' die korrekte Java-Methodenzuordnung für die WSDL-Methode fest.
- Wenn die Annotation @WebMethod nicht vorhanden ist, geht der Funktionsselektor davon aus, dass der Java-Methodenname mit dem Namen der aufgerufenen Operation übereinstimmt..

Anmerkung: Dieser Funktionsselektor ist nur bei einer WSDL-Schnittstelle für einen EJB-Import und nicht bei einer Java-Schnittstelle für einen EJB-Import gültig.

Der Funktionsselektor gibt ein Objekt 'java.lang.reflect.Method' zurück, das die Methode der EJB-Schnittstelle darstellt.

Der Funktionsselektor verwendet einen Java-Objektarray (Object[]) zur Aufnahme der Antwort von der Zielmethode. Das erste Element im Array 'Object[]' ist eine Java-Methode mit dem Namen aus WSDL, das zweite Element im Array 'Object[]' ist das Eingabegeschäftsobjekt.

Exportfunktionsselektor

Bei der Eingangsverarbeitung leitet der Exportfunktionsselektor die aufzurufende Zielmethode aus der Java-Methode ab.

Der Exportfunktionsselektor ordnet den Namen der vom EJB-Client aufgerufenen Java-Operation dem Namen der Operation in der Schnittstelle der Zielkomponente zu. Der Methodenname wird als Zeichenfolge zurückgegeben und durch die SCA-Laufzeit abhängig vom Schnittstellentyp der Zielkomponente aufgelöst.

EIS-Bindungen:

EIS-Bindungen stellen Konnektivität zwischen SCA-Komponenten und einem externen EIS (Enterprise Information System, unternehmensweites Informationssystem) bereit. Diese Kommunikation wird erreicht durch die Verwendung von EIS-Exporten und EIS-Importen, die JCA 1.5-Ressourcenadapter und WebSphere Adapters unterstützen.

Möglicherweise macht es eine SCA-Komponente erforderlich, dass Daten an ein externes EIS bzw. von diesem übertragen werden. Wenn Sie ein SCA-Modul erstellen, das eine solche Konnektivität benötigt, schließen Sie (neben Ihrer SCA-Komponente) einen Import oder Export mit einer EIS-Bindung für die Kommunikation mit einem bestimmten externen EIS ein.

Ressourcenadapter in IBM Integration Designer werden im Kontext eines Imports oder Exports verwendet. Sie entwickeln einen Import oder Export mit dem Assistenten 'Externer Service' und schließen bei seiner Entwicklung den Ressourcenadapter ein. Ein EIS-Import, mit dem eine Anwendung einen Service auf einem EIS-System aufrufen kann, oder ein EIS-Export, mit dem eine Anwendung auf einem EIS-System einen in IBM Integration Designer entwickelten Service aufrufen kann, wird mit einem Ressourcenadapter erstellt. Beispielsweise würden Sie einen Import mit dem JD Edwards-Adapter erstellen, um einen Service auf einem JD Edwards-System aufzurufen.

Wenn Sie den Assistenten 'Externer Service' verwenden, werden die Informationen zur EIS-Bindung automatisch erstellt. Bindungsinformationen können Sie auch mit einem anderen Tool hinzufügen oder ändern, nämlich dem Assembly-Editor. Weitere Informationen hierzu finden Sie unter Mit Adaptern auf externe Services zugreifen.

Nachdem das SCA-Modul, das die EIS-Bindung enthält, auf dem Server implementiert wurde, können Sie in der Administrationskonsole die Informationen zur Bindung anzeigen oder die Bindung konfigurieren.

EIS-Bindungen - Übersicht:

Wenn die EIS-Bindung (EIS = Enterprise Information System, unternehmensweites Informationssystem) zusammen mit einem JCA-Ressourcenadapter verwendet wird, können Sie auf Services in einem unternehmensweiten Informationssystem zugreifen oder Ihre Services für das EIS bereitstellen.

Die folgenden Beispiele veranschaulichen, wie ein SCA-Modul namens **ContactSyncModule** Kontaktinformationen zwischen einem Siebel-System und einem SAP-System synchronisiert.

1. Die SCA-Komponente namens **ContactSync** überwacht (durch einen EIS-Anwendungsexport namens 'Siebel Contact') Siebel-Kontakte auf Änderungen.
2. Die SCA-Komponente **ContactSync** selbst verwendet eine SAP-Anwendung (durch einen EIS-Anwendungsimport), um die SAP-Kontaktinformationen entsprechend zu aktualisieren.

Da die Datenstrukturen für die Speicherung von Kontakten bei Siebel- und SAP-Systemen unterschiedlich sind, muss die SCA-Komponente **ContactSync** eine Zuordnung bereitstellen.

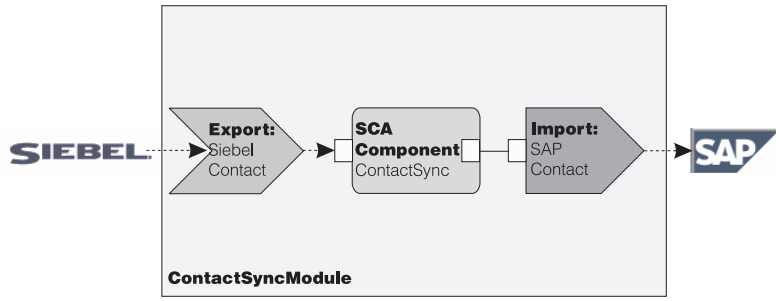


Abbildung 31. Fluss von einem Siebel-System an ein SAP-System

Für den Export 'Siebel Contact' und den Import 'SAP Contact' sind die geeigneten Ressourcenadapter konfiguriert.

Schlüsselfunktionen von EIS-Bindings:

Ein EIS-Import ist ein SCA-Import (SCA = Service Component Architecture), der es den Komponenten im SCA-Modul ermöglicht, EIS-Anwendungen zu verwenden, die außerhalb des SCA-Moduls definiert sind. Ein EIS-Import wird zum Übertragen der Daten von der SCA-Komponente an ein externes EIS verwendet; ein EIS-Export wird zum Übertragen der Daten von einem externen EIS in das SCA-Modul verwendet.

Importe

Ein EIS-Import dient dazu, die Lücke zwischen SCA-Komponenten und EIS-Systemen zu schließen. Externe Anwendungen können als ein EIS-Import behandelt werden. In diesem Fall werden von einem EIS-Import Daten an das externe EIS gesendet und optional Daten als Antwort empfangen.

Vom EIS-Import wird den SCA-Komponenten eine einheitliche Sicht der Anwendungen bereitgestellt, die sich außerhalb des Moduls befinden. So können die Komponenten mit einem externen EIS wie SAP, Siebel oder PeopleSoft über ein konsistentes SCA-Modell kommunizieren.

Auf der Clientseite des Imports befindet sich eine Schnittstelle, die von der EIS-Exportanwendung verfügbar gemacht wird; sie verfügt über mindestens eine Methode, von der Datenobjekte als Argumente angenommen und Werte zurückgegeben werden. Auf der Implementierungsseite befindet sich eine Common Client Interface (CCI), die von einem Ressourcenadapter implementiert wird.

Von der Laufzeitimplementierung eines EIS-Imports wird eine Verbindung zwischen der clientseitigen Schnittstelle und der CCI hergestellt. Vom Import wird der Aufruf der Methode in der Schnittstelle dem Aufruf in der CCI zugeordnet.

Bindings werden auf drei Ebenen erstellt: das Schnittstellenbinding, von dem die enthaltenen Methodenbindings verwendet werden, die wiederum die Datenbindings verwenden.

Vom Schnittstellenbinding wird die Schnittstelle des Imports der Verbindung dem EIS-System zugeordnet, von dem die Anwendung bereitgestellt wird. Dies spiegelt die Tatsache wider, dass die Anwendungen, die von der Schnittstelle dargestellt werden, von der konkreten Instanz des EIS bereitgestellt wird, und dass von der Verbindung der Zugriff auf diese Instanz bereitgestellt wird. Das Bindingelement enthält Eigenschaften, deren Informationen dazu ausreichen, die Verbindung herzustellen (diese Eigenschaften sind Bestandteil der Instanz javax.resource.spi.ManagedConnectionFactory).

Vom Methodenbinding wird die Methode mit der konkreten Interaktion dem EIS-System zugeordnet. Bei Verwendung von JCA ergibt sich die Interaktion aus den Eigenschaften der Schnittstellenimplementierung javax.resource.cci.InteractionSpec. Das Interaktionselement des Methodenbindings enthält diese Eigenschaften zusammen mit dem Namen der Klasse und stellt so ausreichend Informationen zum Ausführen der Interaktion bereit. Vom Methodenbinding werden Datenbindings verwendet, die die Zuordnung des Arguments und des Ergebnisses der Schnittstellenmethode zur EIS-Darstellung beschreiben.

Laufzeitszenario für einen EIS-Import:

1. Die Methode für die Importschnittstelle wird mithilfe des SCA-Programmiermodells aufgerufen.
2. In der Anforderung, die vom EIS-Import empfangen wird, sind der Name der Methode und die zugehörigen Argumente enthalten.
3. Vom Import wird zuerst eine Schnittstellenbindingimplementierung erstellt; anschließend wird unter Verwendung der Daten aus dem Importbinding die API für die Verbindungsfactory (ConnectionFactory) erstellt und beide werden einander zugeordnet. Dies bedeutet, dass vom Import die API setConnectionFactory im Schnittstellenbinding aufgerufen wird.
4. Die Methodenbindingimplementierung für die entsprechende aufgerufene Methode wird erstellt.
5. Die Instanz javax.resource.cci.InteractionSpec wird erstellt und gefüllt; anschließend werden die Methodenargumente mithilfe der Datenbindings an ein Format gebunden, das vom Ressourcenadapter verstanden wird.
6. Die CCI-Schnittstelle wird zum Ausführen der Interaktion verwendet.
7. Wenn der Aufruf zurückgegeben wird, wird das Datenbinding zum Erstellen des Ergebnisses des Aufrufs verwendet und das Ergebnis wird an das aufrufende Program zurückgegeben.

Exporte

Ein EIS-Export dient dazu, die Lücke zwischen einer SCA-Komponente und einem externen EIS zu schließen. Externe Anwendungen können als ein EIS-Export behandelt werden. In diesem Fall werden die Daten der externen Anwendung von ihr in Form regelmäßiger Benachrichtigungen gesendet. Ein EIS-Export ist sozusagen eine Subskriptionsanwendung, die für eine externe Anforderung von einem EIS empfangsbereit ist. Von der SCA-Komponente, die den EIS-Export verwendet, wird dieser als lokale Anwendung betrachtet.

Vom EIS-Export wird den SCA-Komponenten eine einheitliche Sicht der Anwendungen bereitgestellt, die sich außerhalb des Moduls befinden. So können die Komponenten mit einem EIS wie SAP, Siebel oder PeopleSoft über ein konsistentes SCA-Modell kommunizieren.

Vom Export wird die Implementierung eines Listeners bereitgestellt, von dem die Anforderungen des EIS empfangen werden. Vom Listener wird eine für den Ressourcenadapter spezifische Schnittstelle implementiert. Der Export enthält auch eine Komponenten implementierende Schnittstelle, die dem EIS über den Export verfügbar gemacht wird.

Die Laufzeitimplementierung eines EIS-Imports verbindet den Listener mit der Komponenten implementierenden Schnittstelle. Die EIS-Anforderung wird vom Export dem Aufruf der entsprechenden Operation für die Komponente zugeordnet. Bindings werden auf drei Ebenen erstellt: ein Listener-Binding, von dem anschließend ein enthaltenes natives Methodenbinding verwendet wird, von dem wiederum ein Datenbinding verwendet wird.

Vom Listener-Binding wird der Listener, der die Anforderungen empfängt, der Komponente zugeordnet, die über den Export zugänglich gemacht wird. In der Exportdefinition ist der Name der Komponente enthalten; sie wird von der Laufzeit gesucht und die Anforderungen werden an sie weitergeleitet.

Vom nativen Methodenbinding werden die vom Listener empfangenen nativen Methoden oder Ereignistypen den Operationen zugeordnet, die von der Komponente über den Export zugänglich gemacht werden. Es gibt keine Beziehung zwischen der für den Listener aufgerufenen Methode und dem Ereignistyp; alle Ereignisse werden über mindestens eine Methode des Listeners empfangen. Das native Methodenbinding verwendet den im Export definierten Funktionsselektor, um den nativen Methodennamen aus den eingehenden Daten zu extrahieren und die Datenbindings, um das Datenformat des EIS an ein Format zu binden, das von der Komponente verstanden wird.

Laufzeitszenario für einen EIS-Export:

1. Von einer EIS-Anforderung wird der Aufruf einer Methode für die Listenerimplementierung ausgelöst.
2. Vom Listener wird der Export gesucht und aufgerufen, alle Aufrufargumente werden an ihn übergeben.
3. Vom Export wird die Implementierung des Listener-Bindings erstellt.
4. Vom Export wird der Funktionsselektor instanziiert und für das Listener-Binding eingestellt.
5. Vom Export werden die nativen Methodenbindings initialisiert und zum Listener-Binding hinzugefügt. Für jedes Methodenbinding werden auch die Datenbindings initialisiert.
6. Vom Export wird das Listener-Binding aufgerufen.
7. Vom Listener-Binding werden exportierte Komponenten gesucht und der native Methodenname mit dem Funktionsselektor abgerufen.
8. Anhand dieses Namens wird das native Methodenbinding gesucht, von dem schließlich die Zielkomponente aufgerufen wird.

Der Adapterinteraktionsstil ermöglicht es dem EIS-Exportbinding, die Zielkomponente entweder asynchron (die Standardeinstellung) oder synchron aufzurufen.

Ressourcenadapter

Sie entwickeln einen Import oder Export mit dem Assistenten für externe Services und schließen bei der Entwicklung einen Ressourcenadapter mit ein. Die Adapter, die im Lieferumfang von IBM Integration Designer für den Zugriff auf CICS-, IMS-, JD Edwards-, PeopleSoft-, SAP- und Siebel-Systeme enthalten sind, sind nur für Entwicklungs- und Testzwecke vorgesehen. Sie können sie also nur zum Entwickeln und Testen der Anwendungen verwenden.

Sobald Sie eine Anwendung implementieren, benötigen Sie lizenzierte Laufzeitadapter zum Ausführen der Anwendung. Wenn Sie einen Service erstellen, können Sie den Adapter aber in den Service einbetten. Es kann vorkommen, dass die Verwendung des eingebetteten Adapters als lizenzierter Laufzeitadapter laut Adapterlizenzierung zulässig ist. Diese Adapter sind mit der Java EE Connector Architecture (JCA 1.5) kompatibel. JCA ist ein offener Standard und der Java EE-Standard für EIS-Verbindungen. Von JCA wird ein verwaltetes Framework bereitgestellt; dies bedeutet, dass die Qualität des Service (QoS) vom Anwendungsserver bereitgestellt wird, der Lebenszyklusverwaltung und Sicherheit für Transaktionen bietet. Mit Ausnahme von IBM CICS ECI Resource Adapter und IBM IMS Connector for Java sind Sie auch mit der Enterprise Metadata Discovery-Spezifikation kompatibel.

Auch WebSphere Business Integration Adapters, eine ältere Gruppe aus Adaptern, wird vom Assistenten unterstützt.

Java EE-Ressourcen

Das EIS-Modul, ein SCA-Modul, das dem EIS-Modulmuster entspricht, kann auf der Java EE-Plattform implementiert werden.

Das Ergebnis der Implementierung des EIS-Moduls auf der Java EE-Plattform ist eine Anwendung, die startbereit vorliegt, als EAR-Datei paketiert ist und auf dem Server implementiert wird. Alle Java EE-Artefakte und Ressourcen werden erstellt; die Anwendung ist konfiguriert und kann ausgeführt werden.

Dynamische Eigenschaften der JCA-Interaktionsspezifikation und -Verbindungsspezifikation:

Die EIS-Bindung kann Eingabe für die APIs **InteractionSpec** und **ConnectionSpec** akzeptieren, die durch ein klar strukturiertes untergeordnetes Datenobjekt angegeben werden, das die Nutzdaten begleitet. Dies ermöglicht dynamische Anforderungs-/Antwortinteraktionen mit einem Ressourcenadapter über die API **InteractionSpec** und die Interaktion für die Komponentenauthentifizierung über die API **ConnectionSpec**.

Die API **javax.cci.InteractionSpec** überträgt Informationen dazu, wie die Anforderung für die Interaktion mit dem Ressourcenadapter abgewickelt werden soll. Sie kann außerdem Informationen dazu übertragen, wie die Interaktion nach der Anforderung erreicht wurde. Diese wechselseitige Übertragung durch die Interaktionen wird manchmal auch als *Dialog* bezeichnet.

Die EIS-Bindung erwartet, dass die Nutzdaten, die das Argument für den Ressourcenadapter darstellen, ein untergeordnetes Datenobjekt namens **properties** enthalten. Dieses Eigenschaftsdatenobjekt enthält Name/Wert-Paare, bei denen die Namen der Eigenschaften für die Interaktionsspezifikation in einem bestimmten Format angegeben sind. In diesem Zusammenhang gelten die folgenden Formatierungsregeln:

- Namen müssen mit dem Präfix **IS** beginnen, auf das der Eigenschaftsname folgt. Beispiel: Eine Interaktionsspezifikation mit einer JavaBeans-Eigenschaft namens **InteractionId** würde den Eigenschaftsnamen mit **ISInteractionId** angeben.
- Das Name/Wert-Paar stellt den Namen und den Wert des einfachen Typs für die Eigenschaft der Interaktionsspezifikation dar.

In diesem Beispiel gibt eine Schnittstelle an, dass die Eingabe für eine Operation aus einem Datenobjekt **Account** besteht. Diese Schnittstelle ruft eine Anwendung mit EIS-Importbindung mit dem Ziel auf, eine dynamische Eigenschaft **InteractionSpec** namens **workingSet** mit dem Wert **xyz** zu senden und zu empfangen.

Die Geschäftsgrafik oder die Geschäftsobjekte im Server enthalten ein zugrunde liegendes Geschäftsobjekt **properties**, das das Senden von protokollspezifischen Daten mit den Nutzdaten zulässt. Dieses Geschäftsobjekt **properties** ist integriert und muss beim Erstellen eines Geschäftsobjekts nicht im XML-Schema angegeben werden. Es muss lediglich erstellt und verwendet werden. Falls Sie basierend auf einem XML-Schema eigene Datentypen definiert haben, müssen Sie ein Element **properties** angeben, dass die erwarteten Name/Wert-Paare enthält.

```
BOFactory dataFactory = (BOFactory) \
serviceManager.locateService("com/ibm/websphere/bo/BOFactory");
 //Wrapper for doc-lit wrapped style interfaces,
 //skip to payload for non doc-lit
 DataObject docLitWrapper = dataFactory.createByElement /
 ("http://mytest/eis/Account", "AccountWrapper");
```

Erstellen Sie die Nutzdaten.

```
DataObject account = docLitWrapper.createDataObject(0);
  DataObject accountInfo = account.createDataObject("AccountInfo");
  //Perform your setting up of payload
```

```
//Construct properties data for dynamic interaction

DataObject properties = account.createDataObject("properties");
```

Legen Sie für den Namen von 'workingSet' den erwarteten Wert fest (**xyz**).

```
properties.setString("ISworkingSet", "xyz");


//Invoke the service with argument

Service accountImport = (Service) \
serviceManager.locateService("AccountOutbound");
DataObject result = accountImport.invoke("createAccount", docLitWrapper);

//Get returned property
DataObject retProperties = result.getDataObject("properties");

String workingset = retProperties.getString("ISworkingSet");
```

Sie können die Eigenschaften der API **ConnectionSpec** für die dynamische Komponentenauthentifizierung verwenden. Hierbei gelten ebenfalls die vorgenannten Regeln, allerdings mit der Ausnahme, dass das Präfix **CS** (anstelle von **IS**) für den Eigenschaftsnamen verwendet werden muss. Die Eigenschaften der API **ConnectionSpec** sind nicht bidirektional. Ein einziges Datenobjekt **properties** kann sowohl IS- als auch CS-Eigenschaften enthalten.

Zur Verwendung der Eigenschaften für **ConnectionSpec** legen Sie **resAuth** in der Importbindung mit **Application** fest. Stellen Sie außerdem sicher, dass der Ressourcenadapter die Komponentenauthentifizierung unterstützt. Weitere Angaben finden Sie in Kapitel 8 des Dokuments J2EE Connector Architecture Specification.

Externe Clients mit EIS-Bindungen:

Der Server kann unter Verwendung von EIS-Bindungen Nachrichten an externe Clients senden bzw. von diesen empfangen.

Ein externer Client (beispielsweise ein Webportal oder ein EIS) muss eine Nachricht an ein SCA-Modul im Server senden oder muss von einer Komponente im Server aufgerufen werden.

Der Client ruft wie bei jeder anderen Anwendung den EIS-Import entweder mit der DII (Dynamic Invocation Interface - Schnittstelle für dynamischen Aufruf) oder der Java-Schnittstelle auf.

1. Der externe Client erstellt eine Instanz der API **ServiceManager** und sucht unter Verwendung des Referenznamens nach dem EIS-Import. Das Ergebnis der Suche ist eine Serviceschnittstellenimplementierung.
2. Der Client generiert ein Eingabeargument (ein generisches Datenobjekt), das dynamisch unter Verwendung des Datenobjektschemas erstellt wird. Dieser Schritt wird unter Verwendung der SDO-Schnittstellenimplementierung 'DataFactory' ausgeführt.
3. Der externe Client ruft das EIS auf und erhält die angeforderten Ergebnisse.

Alternativ kann der Client den EIS-Import unter Verwendung der Java-Schnittstelle aufrufen.

1. Der Client erstellt eine Instanz der API **ServiceManager** und sucht unter Verwendung des Referenznamens nach dem EIS-Import. Das Ergebnis der Suche ist eine Java-Schnittstelle des EIS-Imports.
2. Der Client erstellt ein Eingabeargument und ein typisiertes Datenobjekt.
3. Der Client ruft das EIS auf und erhält die angeforderten Ergebnisse.

Die EIS-Exportschnittstelle definiert die Schnittstelle der exportierten SCA-Komponente, die für die externen EIS-Anwendungen verfügbar ist. Diese Schnittstelle ist mit der Schnittstelle vergleichbar, die eine externe Anwendung (z. B. SAP oder PeopleSoft) durch die Implementierung der EIS-Exportanwendungslaufzeit aufruft.

Der Export verwendet die Schnittstelle **EISExportBinding**, um exportierte Services an die externe EIS-Anwendung zu binden. Er ermöglicht das Abonnieren einer im SCA-Modul enthaltenen Anwendung, um EIS-Serviceanforderungen zu überwachen. Die EIS-Exportbindung gibt in einer für den Ressourcenadapter verständlichen Weise (mit Schnittstellen von Java EE Connector Architecture) die Zuordnung zwischen eingehenden Ereignissen und dem Aufruf von SCA-Operationen an.

Die Schnittstelle **EISExportBinding** macht es erforderlich, dass externe EIS-Services auf den Verträgen für eingehende Daten von Java EE Connector Architecture 1.5 basieren. Die Schnittstelle **EISExportBinding** macht es ebenfalls erforderlich, dass ein Datenhandler oder eine Datenbindung entweder auf Bindungsebene oder auf Methodenebene angegeben ist.

JMS-Bindungen:

Ein JMS-Provider (JMS = Java Message Service) ermöglicht den Nachrichtenaustausch basierend auf der API und dem Programmiermodell von Java Messaging Service. Er stellt JMS-Verbindungsfactorys für die Erstellung von Verbindungen für JMS-Ziele sowie zum Senden und Empfangen von Nachrichten bereit.

JMS-Bindungen können verwendet werden, wenn Sie mit der Bindung des SIB-Providers (Service Integration Bus) interagieren und mit JMS und JCA 1.5 kompatibel sind.

Mithilfe der JMS-Exportbindungen und -Importbindungen kann ein SCA-Modul externe JMS-Systeme aufrufen und Nachrichten von diesen empfangen.

Die JMS-Importbindungen und -Exportbindungen ermöglichen die Integration bei JMS-Anwendungen, die den JCA 1.5-basierten SIB-JMS-Provider verwenden, der Teil von WebSphere Application Server ist. Andere JCA 1.5-basierte JMS-Ressourcenadapter werden nicht unterstützt.

JMS-Bindungen - Übersicht:

JMS-Bindungen stellen Konnektivität zwischen der SCA-Umgebung und JMS-Systemen bereit.

JMS-Bindungen

Die Hauptkomponenten sowohl bei JMS-Importbindungen als auch bei JMS-Exportbindungen sind Folgende:

- Ressourcenadapter: Er ermöglicht die verwaltete bidirektionale Konnektivität zwischen einem SCA-Modul und externen JMS-Systemen.
- Verbindungen: Sie binden eine virtuelle Verbindung zwischen einer Client- und einer Provideranwendung ein.
- Ziele: Sie werden von einem Client verwendet, um das Ziel der von ihm erstellten Nachrichten oder die Quelle der von ihm gelesenen Nachrichten anzugeben.
- Authentifizierungsdaten: Sie werden verwendet, um den Zugriff auf die Bindung zu schützen.

Schlüsselfunktionen von JMS-Bindings

Besondere Header

Die Eigenschaften besonderer Header werden in JMS-Importen und -Exporten dazu verwendet, dem Ziel mitzuteilen, wie die Nachricht verarbeitet werden soll.

Bei Verwendung von TargetFunctionName wird eine Zuordnung von der nativen Methode zur Operationsmethode vorgenommen.

Java EE-Ressourcen

Wenn JMS-Importe und -Exporte in einer Java EE-Umgebung implementiert werden, wird eine Reihe von Java EE-Ressourcen erstellt.

`ConnectionFactory`
Wird von Clients zum Erstellen einer Verbindung zum JMS-Provider verwendet.

`ActivationSpec`
Wird von Importen zum Empfangen der Antwort auf eine Anforderung verwendet; wird von Exporten zum Konfigurieren der Nachrichtenendpunkte verwendet, die Nachrichtenlistener in ihrer Interaktion mit dem Messaging-System darstellen.

`Ziele`

- Sendeziel: Für einen Import ist dies das Ziel, an das die Anforderung oder Nachricht gesendet wird; für einen Export ist dies das Ziel, an das die Antwortnachricht gesendet wird, falls diese nicht durch das Headerfeld JMSReplyTo in der eingehenden Nachricht außer Kraft gesetzt wird.
- Empfangsziel: Das Ziel, an das die eingehende Nachricht gesendet werden soll; bei Importen ist es eine Antwort, bei Exporten eine Anforderung.
- Callbackziel: Das SCA-JMS-Systemziel, das zum Speichern der Korrelationsinformationen verwendet wird. Nehmen Sie an dieser Adresse keine Lese- oder Schreiboperationen vor.

Von der Installationstask werden die Verbindungsfactory (ConnectionFactory) und drei Ziele erstellt. Außerdem wird von ihr die Aktivierungsspezifikation (ActivationSpec) zum Aktivieren des Laufzeitnachrichtenlisteners zur Überwachung auf Nachrichten am Empfangsziel erstellt. Die Eigenschaften dieser Ressourcen werden in der Import- oder Exportdatei angegeben.

JMS-Integration und Ressourcenadapter:

Java Message Service (JMS) ermöglicht die Integration durch einen verfügbaren, auf JMS JCA 1.5 basierenden Ressourcenadapter. Die vollständige Unterstützung für die JMS-Integration wird für den SIB-JMS-Ressourcenadapter bereitgestellt (SIB = Service Integration Bus).

Verwenden Sie einen Ressourcenadapter für einen JMS-Provider für JCA 1.5, wenn Sie eine Integration bei einem externen JCA 1.5-konformen JMS-System wünschen. Externe Services, die mit JCA 1.5 konform sind, können mit dem SIB-JMS-Ressourcenadapter Nachrichten empfangen und senden, um sich bei den SCA-Komponenten zu integrieren.

Die Verwendung anderer providerspezifischer JCA 1.5-Ressourcenadapter wird nicht unterstützt.

JMS-Importbindungen und -Exportbindungen:

Mithilfe von JMS-Importbindungen und -Exportbindungen können Sie dafür sorgen, dass SCA-Module mit Services interagieren, die von externen JMS-Anwendungen bereitgestellt werden.

JMS-Importbindungen

Verbindungen zum zugehörigen JMS-Provider von JMS-Zielen werden unter Verwendung einer JMS-Verbindungsfactory erstellt. Mit Verwaltungsobjekten für Verbindungsfactorys können Sie JMS-Verbindungsfactorys für den Standard-Messaging-Provider verwalten.

Die Interaktion mit externen JMS-Systemen umfasst auch die Verwendung von Zielen für das Senden von Anforderungen und das Empfangen von Antworten.

Es werden zwei Typen von Einsatzszenarios für JMS-Importbindungen unterstützt, die sich nach dem Typ der aufgerufenen Operation richten:

- Unidirektional: Der JMS-Import übergibt eine Nachricht an das Sendeziel, das in der Importbindung konfiguriert ist. Das Feld **replyTo** des JMS-Headers enthält keine Angaben.
- Bidirektional (Anforderung/Antwort): Der JMS-Import übergibt eine Nachricht an das Sendeziel und behält dann die von der SCA-Komponente empfangene Antwort bei.

  Die Importbindung kann (mit dem Feld **Korrelationsschema für Antwort** in Integration Designer) so konfiguriert werden, dass das Kopieren der Korrelations-ID für die Antwortnachricht aus der Anforderungsnachrichten-ID (Standardeinstellung) oder aus der Korrelations-ID für die Anforderungsnachricht erwartet wird. Außerdem kann für die Importbindung die Verwendung eines temporären dynamischen Antwortziels konfiguriert werden, um Antworten mit Anforderungen zu korrelieren. Für jede Anforderung wird ein temporäres Ziel erstellt. Der Import verwendet dieses Ziel, um die Antwort zu empfangen.

  Das Empfangsziel (**receive**) ist in der Headereigenschaft **replyTo** der abgehenden Nachricht festgelegt. Für die Überwachung des Empfangsziels wird ein Nachrichtenlistener implementiert. Sobald eine Antwort empfangen wird, übergibt der Nachrichtenlistener die Antwort zurück an die Komponente.

Sowohl bei unidirektionalen als auch bei bidirektionalen Einsatzszenarios können dynamische und statische Headereigenschaften angegeben werden. Statische Eigenschaften können über die Methodenbindung des JMS-Imports festgelegt werden. Manche dieser Eigenschaften haben in der SCA-JMS-Laufzeit eine besondere Bedeutung.

Es muss unbedingt beachtet werden, dass es sich bei JMS um eine asynchrone Bindung handelt. Falls eine aufrufende Komponente einen JMS-Import (bei einer bidirektionalen Operation) im Synchronmodus aufruft, wird die aufrufende Komponente blockiert, bis die Antwort durch den JMS-Service zurückgegeben wurde.

In Abb. 32 auf Seite 125 ist dargestellt, wie der Import mit dem externen Service verknüpft ist.

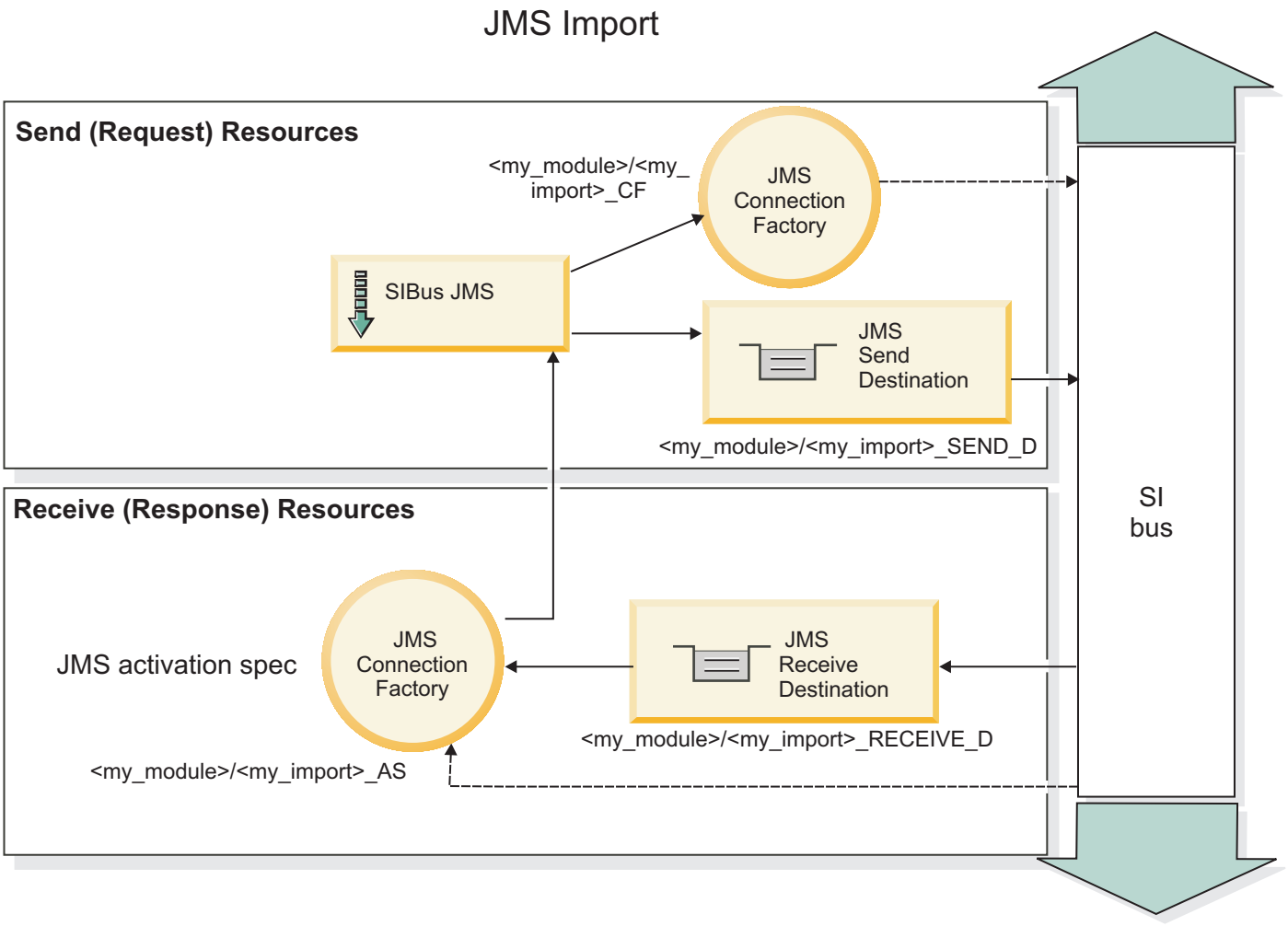


Abbildung 32. Ressourcen der JMS-Importbindung

JMS-Exportbindungen

JMS-Exportbindungen stellen SCA-Modulen ein Verfahren bereit, mit dem Services für externe JMS-Anwendungen angeboten werden können.

Bei der Verbindung, die Teil eines JMS-Exports ist, handelt es sich um eine konfigurierbare Aktivierungsspezifikation.

Ein JMS-Export besitzt Sende- und Empfangsziele.

- Das Empfangsziel ist die Position, an der die eingehende Nachricht für die Zielkomponente übergeben werden soll.
- Das Sendeziel ist die Position, an die die Antwort gesendet wird, sofern diese Angabe in der eingehenden Nachricht nicht durch die Headereigenschaft **replyTo** überschrieben wurde.

Zur Überwachung von Anforderungen, die an dem in der Exportbindung angegebenen **Empfangsziel** eingehen, wird ein Nachrichtenlistener implementiert. Das im Feld **send** angegebene Ziel wird verwendet, um die Antwort auf die eingehende Anforderung zu senden, falls die aufgerufene Komponente eine Antwort bereitstellt. Das im Feld **replyTo** der eingehenden Nachricht angegebene Ziel überschreibt das im Feld **send** angegebene Ziel.

In Abb. 33 auf Seite 126 ist dargestellt, wie der externe Anforderer mit dem Export verknüpft ist.

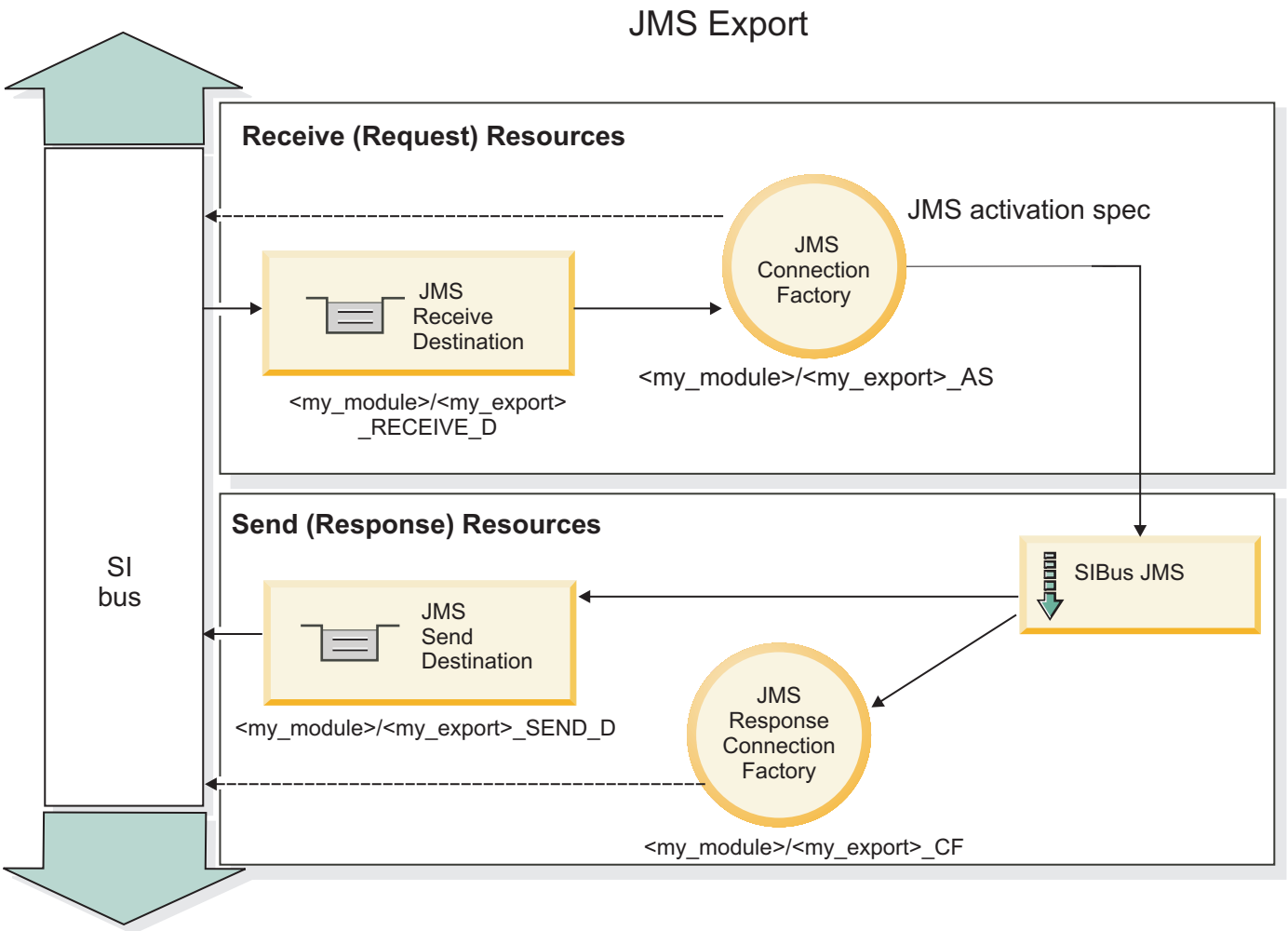


Abbildung 33. Ressourcen der JMS-Exportbindung

JMS-Header:

Eine JMS-Nachricht enthält zwei Typen von Headern, nämlich den JMS-Systemheader und mehrere JMS-Eigenschaften. Auf beide Headertypen kann entweder in einem Mediationsmodul im Servicenachrichtenobjekt (Service Message Object - SMO) oder unter Verwendung der API 'ContextService' zugegriffen werden.

JMS-Systemheader

Der JMS-Systemheader wird im Servicenachrichtenobjekt durch das Element 'JMSHeader' repräsentiert, das alle normalerweise in einem JMS-Header zu findenden Felder enthält. Obwohl diese Felder in der Mediation (oder der API 'ContextService') geändert werden können, werden einige der im Servicenachrichtenobjekt festgelegten Felder für den JMS-Systemheader nicht an die abgehende JMS-Nachricht weitergegeben, weil sie durch Systemwerte oder statische Werte überschrieben werden.

Die folgenden Schlüsselfelder im JMS-Systemheader können in einer Mediation (oder einer API 'ContextService') aktualisiert werden:

- **JMSType** und **JMSCorrelationID** - Werte der spezifischen vordefinierten Nachrichtenheadereigenschaften.
- **JMSDeliveryMode**: Werte für den Übermittlungsmodus ('persistent' oder 'nonpersistent'; Standardwert ist 'persistent').
- **JMSPriority**: Prioritätswert (0 bis 9; Standardwert ist 'JMS_Default_Priority').

JMS-Eigenschaften

JMS-Eigenschaften werden im Servicenachrichtenobjekt als Einträge in der Eigenschaftsliste ('Properties') dargestellt. Die Eigenschaften können in einer Mediation oder durch die Verwendung der API 'Context-Service' hinzugefügt, aktualisiert oder gelöscht werden.

Eigenschaften können auch in der JMS-Bindung statisch festgelegt sein. Statisch festgelegte Eigenschaften überschreiben Einstellungen desselben Namens, die dynamisch festgelegt werden.

Benutzereigenschaften, die von anderen Bindungen (z. B. einer HTTP-Bindung) weitergegeben werden, werden in der JMS-Bindung als JMS-Eigenschaften ausgegeben.

Einstellungen für Headerweitergabe

Die Weitergabe des JMS-Systemheaders und der JMS-Eigenschaften entweder von der eingehenden JMS-Nachricht an nachgelagerte Komponenten oder von vorgelagerten Komponenten an die abgehende JMS-Nachricht kann durch das Attribut 'Protokollheader weitergeben' der Bindung gesteuert werden.

Wenn das Attribut 'Protokollheader weitergeben' festgelegt ist, ist die Übermittlung von Headerinformationen an die Nachricht oder die Zielkomponente, wie in der folgenden Liste beschrieben, zulässig:

- JMS-Exportanforderung

  Der in der Nachricht empfangene JMS-Header wird mittels des Kontextservice an Zielkomponenten weitergegeben. In der Nachricht empfangene JMS-Eigenschaften werden mittels des Kontextservice an Zielkomponenten weitergegeben.
- JMS-Exportantwort

  Alle im Kontextservice definierten JMS-Headerfelder werden in der abgehenden Nachricht verwendet, sofern sie nicht durch statische Eigenschaften überschrieben werden, die in der JMS-Exportbindung festgelegt sind. Alle im Kontextservice definierten Eigenschaften werden in der abgehenden Nachricht verwendet, sofern sie nicht durch statische Eigenschaften überschrieben werden, die in der JMS-Exportbindung festgelegt sind.
- JMS-Importanforderung

  Alle im Kontextservice definierten JMS-Headerfelder werden in der abgehenden Nachricht verwendet, sofern sie nicht durch statische Eigenschaften überschrieben werden, die in der JMS-Importbindung festgelegt sind. Alle im Kontextservice definierten Eigenschaften werden in der abgehenden Nachricht verwendet, sofern sie nicht durch statische Eigenschaften überschrieben werden, die in der JMS-Importbindung festgelegt sind.
- JMS-Importantwort

  Der in der Nachricht empfangene JMS-Header wird mittels des Kontextservice an Zielkomponenten weitergegeben. In der Nachricht empfangene JMS-Eigenschaften werden mittels des Kontextservice an Zielkomponenten weitergegeben.

JMS-Korrelationsschema für temporäres dynamisches Antwortziel:

Das Korrelationsschema für temporäre dynamische Antwortziele bewirkt, dass für jede gesendete Anforderung eine eindeutige dynamische Warteschlange oder bzw. ein eindeutiges dynamisches Topic erstellt wird.

Anhand des im Import angegebenen statischen Antwortziels wird die Spezifik der temporären dynamischen Zielwarteschlange bzw. des Topics abgeleitet. Dies ist im Feld **ReplyTo** der Anforderung festgelegt. Der JMS-Import überwacht dieses Ziel auf Antworten. Sobald die Antwort empfangen wird, wird sie am statischen Antwortziel zur asynchronen Verarbeitung erneut in die Warteschlange eingereiht. Das Feld **CorrelationID** der Antwort wird nicht verwendet und muss nicht festgelegt sein.

Transaktionsbezogene Aspekte

Bei Verwendung eines temporären dynamischen Ziels muss die Antwort in demselben Thread wie die gesendete Antwort gelesen werden. Die Anforderung muss außerhalb der globalen Transaktion gesendet und festgeschrieben werden, bevor sie durch den Back-End-Service empfangen und eine Antwort zurückgegeben wird.

Persistenz

Temporäre dynamische Warteschlangen sind kurzlebige Entitäten, die nicht dasselbe Persistenzniveau wie eine statische Warteschlange oder ein statisches Topic gewährleisten. Eine temporäre dynamische Warteschlange bzw. ein solches Topic bleibt - wie auch die Nachrichten - bei einem Serverneustart nicht erhalten. Nachdem die Nachricht am statischen Antwortziel erneut in die Warteschlange eingereiht wurde, behält sie die in der Nachricht definierte Persistenz bei.

Zeitlimit

Der Import wartet für eine festgelegte Dauer auf eine Antwort am temporären dynamischen Antwortziel. Dieses Zeitintervall wird aus dem SCA-Qualifikationsmerkmal für den Ablauf der Antwort übernommen, falls es festgelegt wird. Andernfalls beträgt die Wartezeit standardmäßig 60 Sekunden. Wird die Wartezeit überschritten, löst der Import eine Ausnahmebedingung des Typs `ServiceTimeoutRuntimeException` aus.

Externe Clients:

Der Server kann mit JMS-Bindungen Nachrichten an externe Clients senden bzw. von diesen empfangen.

Ein externer Client (z. B. ein Webportal oder ein unternehmensweites Informationssystem) kann eine Nachricht an ein SCA-Modul im Server senden oder durch eine Komponente im Server aufgerufen werden.

Die JMS-Exportkomponenten implementieren Nachrichtenlistener, um Anforderungen zu überwachen, die bei dem in der Exportbindung definierten Empfangsziel eingehen. Das im Sendefeld angegebene Ziel wird verwendet, um die Antwort auf die eingehende Anforderung zu senden, falls die aufgerufene Anwendung eine Antwort bereitstellt. Ein externer Client kann somit Anwendungen über die Exportbindung aufrufen.

JMS-Importe interagieren mit externen Clients, indem sie Nachrichten an JMS-Warteschlangen senden oder aus diesen abrufen.

Mit externen Clients arbeiten:

Ein externer Client, d. h. ein Client außerhalb des Servers, muss unter Umständen mit einer Anwendung interagieren, die auf dem Server installiert ist.

Stellen Sie sich ein sehr einfaches Szenario vor, bei dem ein externer Client mit einer Anwendung auf dem Server interagieren möchte. Die Abbildung stellt ein typisches einfaches Szenario dar.

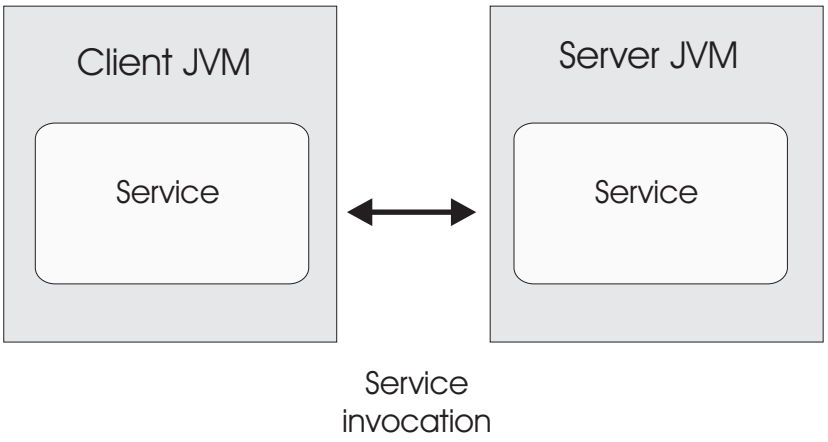


Abbildung 34. Einfaches Anwendungsfallszenario: Externer Client interagiert mit Serveranwendung

Die SCA-Anwendung beinhaltet einen Export mit einer JMS-Bindung. Hierdurch wird die Anwendung auch für externe Clients verfügbar.

Wenn Sie über einen externen Client in einer Java Virtual Machine (JVM) verfügen, der sich nicht auf Ihrem Server befindet, so müssen Sie mehrere Schritte ausführen, um eine Verbindung herzustellen und mit einem JMS-Export zu interagieren. Der Client bezieht einen Ausgangskontext (InitialContext) mit den korrekten Werten und sucht dann die Ressourcen über JNDI. Der Client greift unter Verwendung des JMS 1.1-Spezifikationsclients auf das Ziel zu und sendet bzw. empfängt Nachrichten auf dem Ziel.

Die JNDI-Standardnamen der Ressourcen, die automatisch von der Laufzeit erstellt werden, sind in diesem Abschnitt unter 'Konfiguration' aufgelistet. Wenn Sie jedoch im Vorfeld Ressourcen erstellt haben, verwenden Sie diese JNDI-Namen.

1. Konfigurieren Sie JMS-Ziele und die Verbindungsfactory für das Senden der Nachricht.
2. Stellen Sie sicher, dass der JNDI-Kontext, der Port für den SIB-Ressourcenadapter und der Messaging-Bootstrapping-Port korrekt sind.

   Der Server verwendet einige Standardports, aber wenn auf diesem System weitere Server installiert sind, werden zum Zeitpunkt der Installation alternative Ports erstellt, um eventuelle Konflikte mit anderen Serverinstanzen zu vermeiden. Über die Administrationskonsole können bestimmen, welche Ports Ihr Server verwendet. Gehen Sie zu **Server** > **Anwendungsserver** > ***Name_Ihres_Servers*** > **Konfiguration** und klicken Sie unter **Kommunikation** auf **Ports**. Dann können Sie den Port bearbeiten, der verwendet wird.
3. Der Client bezieht einen Ausgangskontext mit den korrekten Werten und sucht dann die Ressourcen über JNDI.
4. Der Client greift unter Verwendung von JMS 1.1-Spezifikationen auf die Ziele und auf die Sende- und Empfangsnachrichten auf den Zielen.

Fehlerbehebung für JMS-Bindungen:

Sie können Probleme im Zusammenhang mit JMS-Bindungen diagnostizieren und beheben.

Ausnahmebedingungen bei der Implementierung

Als Reaktion auf diverse Fehlerbedingungen kann die JMS-Implementierung für Importe und Exporte zwei Typen von Ausnahmebedingungen zurückgeben:

- Geschäftsausnahmebedingung für Service: Diese Ausnahmebedingung wird zurückgegeben, wenn der für die Geschäftsschnittstelle für den Service (Typ WSDL-Port) definierte Fehler aufgetreten ist.

- Laufzeitausnahmebedingung für den Service: Diese Ausnahmebedingung wird in allen übrigen Fällen ausgelöst. In den meisten Fällen enthält die Ausnahmebedingung vom Typ cause (Ursache) die ursprüngliche Ausnahmebedingung (JMSException).

  Ein Import erwartet zum Beispiel nur eine einzige Antwortnachricht für jede Anforderungsnachricht. Wenn mehr als eine Antwort eingeht oder wenn eine verspätete Antwort eingeht (d. h. eine Antwort, für die die definierte SCA-Antwortablauffrist verstrichen ist), so wird eine Laufzeitausnahmebedingung für den Service ausgelöst. Für die Transaktion wird ein Rollback ausgeführt und die Antwortnachricht wird aus der Warteschlange entfernt oder vom Failed Event Manager verarbeitet.

Primäre Fehlerbedingungen

Die primären Fehlerbedingungen von JMS-Bindungen werden durch Transaktionssemantik, durch die JMS-Providerkonfiguration oder durch Verweise auf bestehendes Verhalten in anderen Komponenten bestimmt. Zu den primären Fehlerbedingungen zählen die Folgenden:

- Die Herstellung der Verbindung zum JMS-Provider oder Ziel ist fehlgeschlagen.

  Das Fehlschlagen der Verbindungsherstellung zum JMS-Provider für den Empfang von Nachrichten hat zur Folge, dass der Nachrichtenlistener nicht gestartet werden kann. Dieser Zustand wird im WebSphere Application Server-Protokoll aufgezeichnet. Persistente Nachrichten verbleiben so lange au f dem Ziel, bis sie erfolgreich abgerufen wurden (oder abgelaufen sind).

  Das Fehlschlagen der Verbindungsherstellung zum JMS-Provider für den Versand abgehender Nachrichten bewirkt ein Rollback der Transaktion, die die Sendeaktion steuert.
- Die Ausführung der Syntaxanalyse für eine eingehende Nachricht oder die Erstellung einer abgehenden Nachricht ist fehlgeschlagen.

  Ein Fehler in der Datenbindung oder im Datenhandler bewirkt ein Rollback der Transaktion, die die Arbeit steuert.
- Das Senden der abgehenden Nachricht ist fehlgeschlagen.

  Das Fehlschlagen des Versands einer Nachricht bewirkt ein Rollback der relevanten Transaktion.
- Mehrere oder nicht erwartete verspätete Antwortnachrichten.

  Der Import erwartet für jede Anforderungsnachricht jeweils nur eine Antwortnachricht. Der Zeitraum, in dem der Empfang einer Antwort gültig ist, wird durch das SCA-Qualifikationsmerkmal für den Ablauf der Antwort bestimmt, das für die Anforderung festgelegt ist. Wenn eine Antwort eintrifft oder die Verfallszeit überschritten wird, so wird der Korrelationsdatensatz gelöscht. Treffen Antwortnachrichten unerwartet oder verspätet ein, so wird eine Laufzeitausnahmebedingung für den Service ausgelöst.
- Laufzeitausnahmebedingung durch Servicelimit, verursacht durch verspätete Antwort bei Verwendung des Korrelationsschemas für temporäre dynamische Antwortziele.

  Der JMS-Import überschreitet nach Ablauf einer durch das SCA-Qualifikationsmerkmal für den Ablauf der Antwort bestimmten Zeitspanne das Zeitlimit. Ist kein Zeitlimit definiert, gilt als Zeitlimit standardmäßig eine Zeitspanne von 60 Sekunden.

JMS-basierte SCA-Nachrichten werden im Failed Event Manager nicht angezeigt

Wenn die SCA-Nachrichten ihren Ursprung in einem Fehler oder einer Störung in der JMS-Interaktion haben, würden Sie erwarten, diese Nachrichten im Failed Event Manager vorzufinden. Werden keine derartigen Nachrichten im Failed Event Manager angezeigt, stellen Sie sicher, dass für das dem JMS-Ziel zugrunde liegende SIB-Ziel ein Wert größer als **1** für die maximale Anzahl fehlgeschlagener Zustellungen definiert ist. Durch Festlegen von **2** oder höher für diesen Wert wird eine Interaktion mit dem Failed Event Manager während SCA-Aufrufen für die JMS-Bindungen ermöglicht.

Ausnahmebedingungen verarbeiten:

Die Art und Weise, in der die Bindung konfiguriert ist, bestimmt, wie Ausnahmebedingungen verarbeitet werden, die von Datenhandlern oder Datenbindungen ausgelöst werden. Außerdem gibt die Spezifik des Mediationsablaufs das Verhalten des Systems vor, wenn eine solche Ausnahmebedingung ausgelöst wird.

Wenn ein Datenhandler oder eine Datenbindung durch eine Bindung aufgerufen wird, können verschiedene Probleme auftreten. Beispielsweise kann es sein, dass ein Datenhandler eine Nachricht mit beschädigten Nutzdaten empfängt oder er versucht, eine Nachricht mit einem falschen Format zu lesen.

Das Verfahren, mit dem die Bindung eine solche Ausnahmebedingung behandelt, wird dadurch bestimmt, wie Sie den Datenhandler oder die Datenbindung implementieren. Das empfohlene Verhalten besteht darin, die Datenbindung so zu konfigurieren, dass sie eine Ausnahmebedingung des Typs **DataBindingException** auslöst.

Wenn eine Laufzeitausnahmebedingung (inklusive **DataBindingException**) ausgelöst wird, geschieht Folgendes:

- Falls der Mediationsablauf transaktionsorientiert konfiguriert ist, wird die JMS-Nachricht standardmäßig in Failed Event Manager gespeichert, damit sie manuell wiedergegeben oder gelöscht werden kann.

  Anmerkung: Sie können den Fehlerbehebungsmodus für die Bindung so ändern, dass die Nachricht zurückgesetzt und nicht in Failed Event Manager gespeichert wird.
- Ist der Mediationsablauf nicht transaktionsorientiert, wird die Ausnahmebedingung protokolliert und die Nachricht ist nicht mehr vorhanden.

Bei einem Datenhandler ist die Situation ähnlich. Da der Datenhandler durch die Datenbindung aufgerufen wird, wird jede Ausnahmebedingung des Datenhandlers in eine Ausnahmebedingung der Datenbindung eingeschlossen. Daher wird eine Ausnahmebindung des Typs **DataHandlerException** als Typ **DataBindingException** gemeldet.

Generische JMS-Bindungen:

Die generische JMS-Bindung stellt Konnektivität für JMS 1.1-konforme Provider von Drittherstellern bereit. Der Operation der generischen JMS-Bindung ähnelt der Operation von JMS-Bindungen.

Mit dem durch eine JMS-Bindung bereitgestellten Service kann ein SCA-Modul Aufrufe an externe Systeme absetzen oder Nachrichten von diesen Systemen empfangen. Bei dem System kann es sich um ein externes JMS-System handeln.

Die generische JMS-Bindung ermöglicht die Integration bei nicht mit JCA 1.5-konformen JMS-Providern, die JMS 1.1 unterstützen und die optionale JMS-Anwendungsserverfunktion implementieren. Die generische JMS-Bindung unterstützt diejenigen JMS-Provider (einschließlich Oracle AQ, TIBCO, SonicMQ, WebMethods und BEA WebLogic), die JCA 1.5 nicht unterstützen, jedoch mit einer Unterstützung der Anwendungsserverfunktion gemäß der JMS 1.1-Spezifikation ausgestattet sind. Der in WebSphere integrierte JMS-Provider (SIBJMS), bei dem es sich um einen JCA 1.5-JMS-Provider handelt, wird durch diese Bindung nicht unterstützt. Bei Verwendung dieses Providers verwenden Sie die „JMS-Bindungen" auf Seite 122.

Verwenden Sie diese generische Bindung zur Integration bei einem nicht mit JCA 1.5 konformen JMS-basierten System in einer SCA-Umgebung. Die externen Zielanwendungen können dann zur Integration bei einer SCA-Komponente Nachrichten empfangen und senden.

Generische JMS-Bindungen - Übersicht:

Generische JMS-Bindungen sind JMS-Bindungen ohne JCA, die Konnektivität zwischen der SCA-Umgebung und JMS-Systemen bereitstellen, die mit JMS 1.1 konform sind und die optionale Anwendungsserverfunktion implementieren.

Generische JMS-Bindungen

Die Hauptaspekte von generischen JMS-Importbindungen und -Exportbindungen sind Folgende:

- Listener-Port: Er ermöglicht nicht auf JCA basierenden JMS-Providern den Empfang von Nachrichten und deren Zuteilung zu einer nachrichtengesteuerten Bean (Message Driven Bean - MDB).
- Verbindungen: Sie binden eine virtuelle Verbindung zwischen einer Client- und einer Provideranwendung ein.
- Ziele: Sie werden von einem Client verwendet, um das Ziel der von ihm erstellten Nachrichten oder die Quelle der von ihm gelesenen Nachrichten anzugeben.
- Authentifizierungsdaten: Sie werden verwendet, um den Zugriff auf die Bindung zu schützen.

Generische JMS-Importbindungen

Generische JMS-Importbindungen ermöglichen den Komponenten in einem SCA-Modul die Kommunikation mit Services, die durch externe und nicht mit JCA 1.5 konforme JMS-Provider bereitgestellt werden.

Der Verbindungsteil eines JMS-Imports ist eine Verbindungsfactory. Eine Verbindungsfactory, also das Objekt, mit dem der Client eine Verbindung zu einem Provider herstellt, bindet eine Reihe von Verbindungskonfigurationsparametern ein, die durch einen Administrator definiert werden. Jede Verbindungsfactory ist eine Instanz der Schnittstelle **ConnectionFactory**, **QueueConnectionFactory** oder **TopicConnectionFactory**.

Die Interaktion mit externen JMS-Systemen umfasst auch die Verwendung von Zielen für das Senden von Anforderungen und das Empfangen von Antworten.

Es werden zwei Typen von Einsatzszenarios für generische JMS-Importbindungen unterstützt, die sich nach dem Typ der aufgerufenen Operation richten:
- Unidirektional: Der generische JMS-Import übergibt eine Nachricht an das Sendeziel, das in der Importbindung konfiguriert ist. An das Feld **replyTo** des JMS-Headers werden keine Daten gesendet.
- Bidirektional (Anforderung/Antwort): Der generische JMS-Import übergibt eine Nachricht an das Sendeziel und behält dann die von der SCA-Komponente empfangene Antwort bei.

  Das Empfangsziel (**receive**) ist in der Headereigenschaft **replyTo** der abgehenden Nachricht festgelegt. Für die Überwachung des Empfangsziels wird eine nachrichtengesteuerte Bean implementiert. Sobald eine Antwort empfangen wird, übergibt die nachrichtengesteuerte Bean die Antwort zurück an die Komponente.

  Die Importbindung kann (mit dem Feld **Korrelationsschema für Antwort** in Integration Designer) so konfiguriert werden, dass das Kopieren der Korrelations-ID für die Antwortnachricht aus der Anforderungsnachrichten-ID (Standardeinstellung) oder aus der Korrelations-ID für die Anforderungsnachricht erwartet wird.

Sowohl bei unidirektionalen als auch bei bidirektionalen Einsatzszenarios können dynamische und statische Headereigenschaften angegeben werden. Statische Eigenschaften können über die Methodenbindung des generischen JMS-Imports festgelegt werden. Manche dieser Eigenschaften haben in der SCA-JMS-Laufzeit eine besondere Bedeutung.

Es muss unbedingt beachtet werden, dass es sich bei der generischen JMS-Bindung um eine asynchrone Bindung handelt. Falls eine aufrufende Komponente einen generischen JMS-Import (bei einer bidirektionalen Operation) im Synchronmodus aufruft, wird die aufrufende Komponente blockiert, bis die Antwort durch den JMS-Service zurückgegeben wurde.

In Abb. 35 auf Seite 133 ist dargestellt, wie der Import mit dem externen Service verknüpft ist.

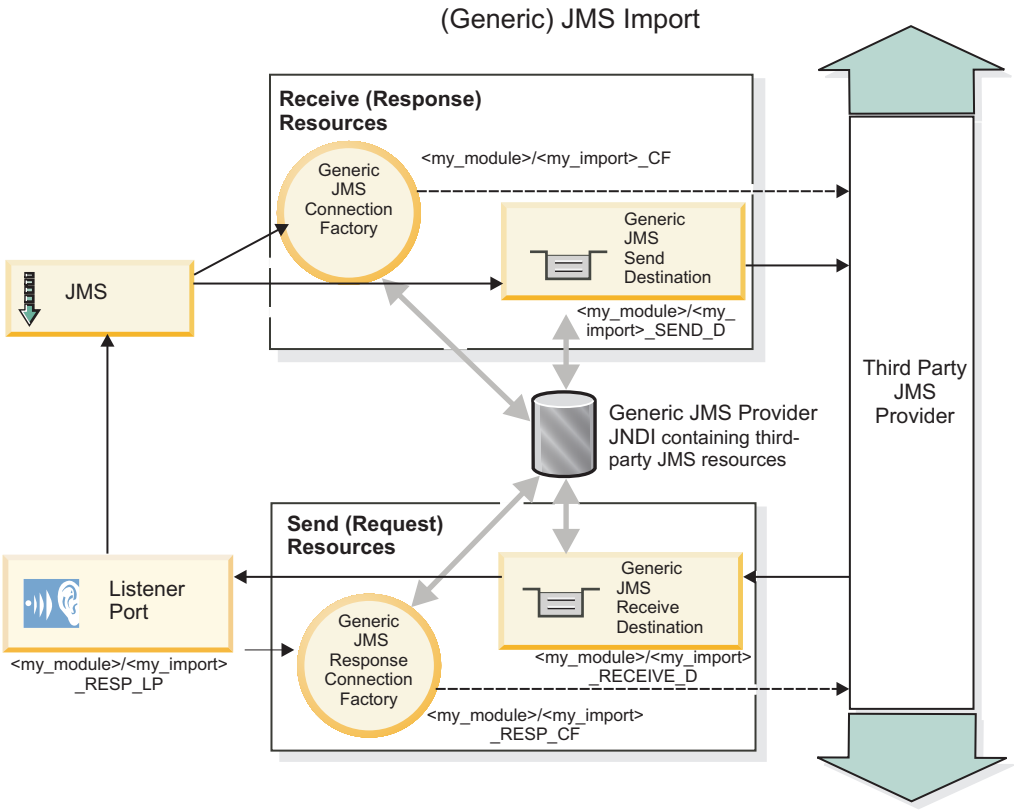


Abbildung 35. Ressourcen der generischen JMS-Importbindung

Generische JMS-Exportbindungen

Von generischen JMS-Exportbindungen wird SCA-Modulen ein Verfahren bereitgestellt, mit dem Services für externe JMS-Anwendungen bereitgestellt werden können.

Der Verbindungsteil eines JMS-Exports besteht aus einer Verbindungsfactory (ConnectionFactory) und einem Listener-Port (ListenerPort).

Ein generischer JMS-Export besitzt Sende- und Empfangsziele.

- Das Ziel receive ist die Position, an der die eingehende Nachricht für die Zielkomponente übergeben werden soll.
- Das Ziel send ist die Position, an die die Antwort gesendet wird, sofern diese Angabe in der eingehenden Nachricht nicht durch die Headereigenschaft replyTo überschrieben wurde.

Zur Überwachung von Anforderungen, die an dem in der Exportbindung angegebenen Ziel receive eingehen, wird eine nachrichtengesteuerte Bean implementiert.

- Das im Feld send angegebene Ziel wird verwendet, um die Antwort auf die eingehende Anforderung zu senden, falls die aufgerufene Komponente eine Antwort bereitstellt.
- Das im Feld replyTo der eingehenden Nachricht angegebene Ziel überschreibt das im Feld send angegebene Ziel.
- Bei Anforderungs-/Antwortszenarios kann die Importbindung (über das Feld **Korrelationsschema für Antwort** in Integration Designer) so konfiguriert werden, dass das Kopieren der Nachrichten-ID aus der Anforderung in das Feld **correlation ID** der Antwortnachricht (Standardverhalten) erwartet wird, oder die Antwort kann die Korrelations-ID der Anforderung in das Feld **correlation ID** der Antwortnachricht kopieren.

In Abb. 36 ist dargestellt, wie der externe Anforderer mit dem Export verknüpft ist.

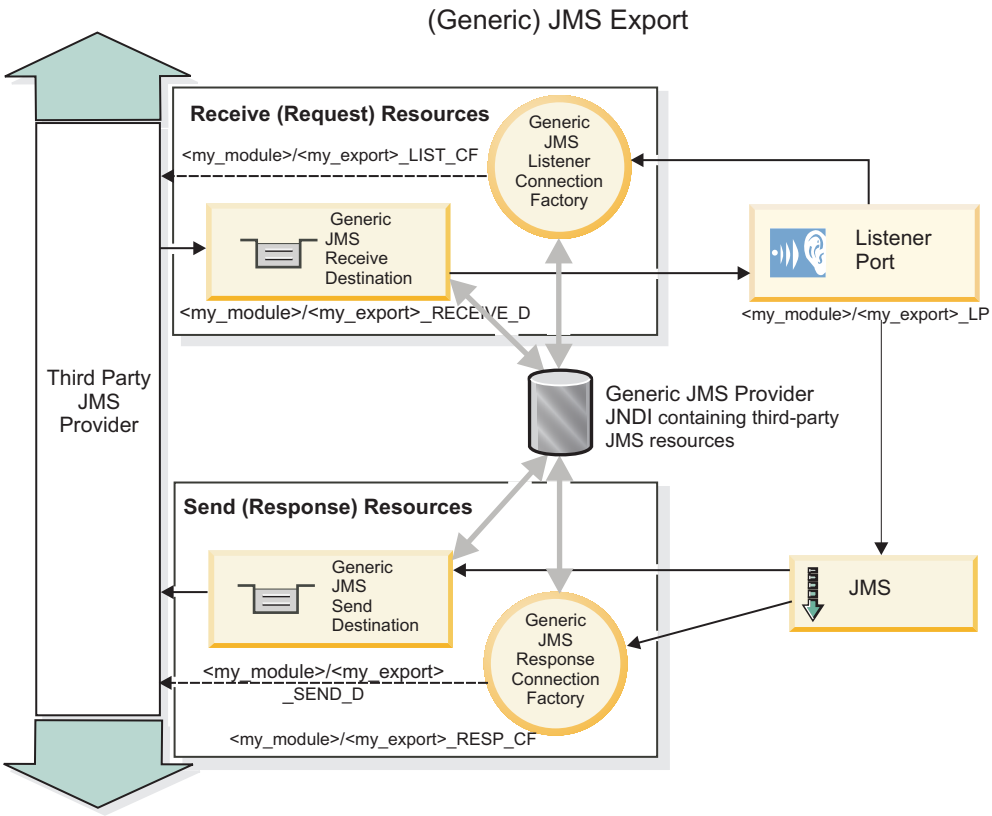


Abbildung 36. Ressourcen der generischen JMS-Exportbindung

Schlüsselfunktionen generischer JMS-Bindings:

Die Funktionen von generischen JMS-Importbindings und -Exportbindings sind mit den Funktionen der eingebetteten WebSphere JMS- und MQ-JMS-Importbindings konsistent. Zu den Schlüsselfunktionen gehören Headerdefinitionen und der Zugriff auf vorhandene Java EE-Ressourcen. Aufgrund ihrer generischen Natur verfügen sie jedoch nicht über Verbindungsoptionen, die für einen bestimmten JMS-Provider spezifisch sind und die Möglichkeiten eines solchen Bindings zum Generieren von Ressourcen im Rahmen der Implementierung und Installation sind eingeschränkt.

Generische Importe

Wie eine MQ-JMS-Importanwendung ist auch eine generische JMS-Implementierung asynchron und unterstützt drei Aufrufe: unidirektional, bidirektional (Request-Response) und Callback.

Wenn ein JMS-Import implementiert wird, wird eine von der Laufzeitumgebung bereitgestellte Message-driven Bean (MDB) implementiert. Von der MDB wird überwacht, ob Antworten auf die Anforderungsnachricht gesendet werden. Die MDB ist dem Ziel zugeordnet (überwacht es), das mit der Anforderung im Headerfeld replyTo der JMS-Nachricht gesendet wurde.

Generische Exporte

Generische JMS-Exportbindings unterscheiden sich von EIS-Exportbindings im Umgang mit der Rückgabe des Ergebnisses. Von einem generischen JMS-Export wird die Antwort explizit an das Ziel replyTo gesendet, das in der eingehenden Nachricht angegeben ist. Wenn keines angegeben ist, wird das Sendeziel verwendet.

Wenn ein generischer JMS-Export implementiert wird, wird eine Message-driven Bean (eine andere MDB als die für die generischen JMS-Importe) implementiert. Von ihr werden die eingehenden Anforderungen am Empfangsziel überwacht und anschließend die Anforderungen zur Verarbeitung durch die SCA-Laufzeit zugeteilt.

Besondere Header

Die Eigenschaften besonderer Header werden in generischen JMS-Importen und -Exporten dazu verwendet, dem Zielbinding mitzuteilen, wie die Nachricht verarbeitet werden soll.

Beispiel: Die Eigenschaft 'TargetFunctionName' wird standardmäßig vom Funktionsselektor dazu verwendet, den Namen der Operation in der Schnittstelle export zu ermitteln, die aufgerufen wird.

Anmerkung: Das Importbinding kann so konfiguriert werden, dass der Header 'TargetFunctionName' als Operationsname für jede Operation eingestellt wird.

Java EE-Ressourcen

Wenn ein JMS-Binding in einer Java EE-Umgebung implementiert wird, werden Java EE-Ressourcen erstellt.

- Listener-Port zum Überwachen am Empfangsziel (Antwort, nur bidirektional) für Importe und am Empfangsziel (Anforderung) für Exporte
- Generische JMS-Verbindungsfactory für abgehende Verbindung (Import) und eingehende Verbindung (Export)
- Generisches JMS-Ziel für Sendeziel (Import) und Empfangsziel (Export, nur bidirektional)
- Generische JMS-Verbindungsfactory für Antwortverbindung (bidirektional und optional; andernfalls wird die abgehende Verbindung für Exporte verwendet)
- Generisches JMS-Ziel für Empfangsziel (Import) und Sendeziel (Export, nur bidirektional)
- Callback-JMS-Ziel für Standard-Messaging-Provider für Zugriff auf Warteschlangenziel für SIB-Callback (nur bidirektional)
- Callback-JMS-Verbindungsfactory für Standard-Messaging-Provider für Zugriff auf Callback-JMS-Ziel (nur bidirektional)
- Warteschlangenziel für SIB-Callback zum Speichern der Anforderungsnachrichtendaten für die Antwortverarbeitung (nur bidirektional)

Von der Installationstask werden die Verbindungsfactory (ConnectionFactory), die drei Ziele und die Aktivierungsspezifikation (ActivationSpec) aus den Informationen in den Import- und Exportdateien erstellt.

Generische JMS-Header:

Generische JMS-Header sind Servicedatenobjekte (Service Data Objects - SDO), die alle Eigenschaften der generischen JMS-Nachrichteneigenschaften enthalten. Diese Eigenschaften können aus der eingehenden Nachricht stammen. Es kann sich aber auch um Eigenschaften handeln, die auf die abgehende Nachricht angewendet werden.

Eine JMS-Nachricht enthält zwei Typen von Headern, nämlich den JMS-Systemheader und mehrere JMS-Eigenschaften. Auf beide Headertypen kann entweder in einem Mediationsmodul im Servicenachrichtenobjekt (Service Message Object - SMO) oder unter Verwendung der API 'ContextService' zugegriffen werden.

Die folgenden Eigenschaften werden in der Methodenbindung (methodBinding) statisch festgelegt:

- JMSType
- JMSCorrelationID

- JMSDeliveryMode
- JMSPriority

Genauso wie die JMS- und die MQ-JMS-Bindung unterstützt die generische JMS-Bindung ebenfalls die dynamische Änderung von JMS-Headern und -Eigenschaften.

Einige generische JMS-Provider schränken die Eigenschaften und deren Kombination ein, die durch die Anwendung festgelegt werden können. Weitere Informationen entnehmen Sie der Dokumentation für das Produkt des Drittherstellers. Es wurde jedoch für 'methodBinding' die zusätzliche Eigenschaft 'ignoreInvalidOutboundJMSProperties' hinzugefügt, die eine Weitergabe aller Ausnahmebedingungen zulässt.

Die generischen JMS-Header und -Nachrichteneigenschaften werden nur dann verwendet, wenn der SCDL-Bindungsswitch der Basisservicekomponentenarchitektur aktiviert ist. Wenn der Switch aktiviert ist, werden Kontextinformationen weitergegeben. Dieser Switch ist standardmäßig aktiviert. Um die Weitergabe von Kontextinformationen zu verhindern, ändern Sie den Wert in **false**.

Bei aktivierter Kontextweitergabe können Headerinformationen an die Nachricht oder die Zielkomponente fließen. Um die Kontextweitergabe zu aktivieren oder zu inaktivieren, geben Sie den Wert **true** bzw. **false** für das Attribut 'contextPropagationEnabled' der Import- und Exportbindungen an. Beispiel:

```
<esbBinding xsi:type="eis:JMSImportBinding" contextProgagationEnabled="true">
```

Der Standardwert ist **true**.

Fehlerbehebung für generische JMS-Bindungen:

Sie können Probleme im Zusammenhang mit generischen JMS-Bindungen diagnostizieren und beheben.

Ausnahmebedingungen bei der Implementierung

Als Reaktion auf diverse Fehlerbedingungen kann die generische JMS-Implementierung für Importe und Exporte zwei Typen von Ausnahmebedingungen zurückgeben:

- Geschäftsausnahmebedingung für Service: Diese Ausnahmebedingung wird zurückgegeben, wenn der für die Geschäftsschnittstelle für den Service (Typ WSDL-Port) definierte Fehler aufgetreten ist.
- Laufzeitausnahmebedingung für den Service: Diese Ausnahmebedingung wird in allen übrigen Fällen ausgelöst. In den meisten Fällen enthält die Ausnahmebedingung vom Typ cause (Ursache) die ursprüngliche Ausnahmebedingung (JMSException).

Fehlerbehebung für den Ablauf generischer JMS-Nachrichten

Eine Anforderungsnachricht vom JMS-Provider kann ablaufen.

Die Bezeichnung *Anforderungsablauf* bezieht sich auf den Verfall der Gültigkeit oder das Ablaufen einer Anforderungsnachricht vom JMS-Provider, wenn die für JMSExpiration festgelegte Zeit für die Anforderungsnachricht erreicht wird. Wie bei anderen JMS-Bindungen handhabt die generische JMS-Bindung das Ablaufen der Anforderung, indem für die abgehende Anforderung denselben Wert für den Ablauf festlegt, wie für die Callback-Nachricht vom Import festgelegt wurde. Eine Benachrichtigung bezüglich des Ablaufs der Callback-Nachricht weist darauf hin, dass die Anforderungsnachricht abgelaufen ist und der Client über eine Geschäftsausnahmebedingung informiert werden sollte.

Wird das Callback-Ziel jedoch zu einem anderen Anbieter verlagert, wird diese Art von Anforderungsablauf (Gültigkeitsverfall von Anforderungen) nicht unterstützt.

Die Bezeichnung *Antwortablauf* bezieht sich auf den Verfall der Gültigkeit oder das Ablaufen einer Antwortnachricht vom JMS-Provider, wenn die für JMSExpiration festgelegte Zeit für die Antwortnachricht erreicht wird.

Die Funktion des Antwortablaufs für generische JMS-Bindungen wird nicht unterstützt, da das genaue Verhalten eines anderen JMS-Providers bezüglich des Verfalls der Gültigkeit nicht definiert ist. Sie können jedoch prüfen, ob die Antwort nicht abgelaufen ist, falls bzw. wenn sie empfangen wird.

Bei abgehenden Anforderungsnachrichten wird der Wert für JMSExpiration anhand der Wartezeit und der in asyncHeader mitgeführten Werte für requestExpiration berechnet, sofern diese festgesetzt wurden.

Fehlerbehebung für generische JMS-Verbindungsfactoryfehler

Wenn Sie bestimmte Typen von Verbindungsfactorys in Ihrem generischen JMS-Provider definieren, wird unter Umständen bei dem Versuch, eine Anwendung zu starten, eine Fehlernachricht angezeigt. Sie können die externe Verbindungsfactory solcherart ändern, dass dieses Problem vermieden wird.

Beim Starten einer Anwendung erhalten Sie möglicherweise eine Fehlernachricht ähnlich der folgenden:

```
Typ von 'JMSConnectionFactory' für MDB-Listener-Port stimmt nicht mit Typ von 'JMSDestination' überein
```

Dieses Problem kann beim Definieren externer Verbindungsfactorys auftreten. Die Ausnahmebedingung kann insbesondere dann ausgelöst werden, wenn Sie eine JMS 1.0.2-Topic-Verbindungsfactory anstelle einer (einheitlichen) JMS 1.1-Verbindungsfactory erstellen (d. h. anstelle einer Verbindungsfactory, die in der Lage ist, sowohl Punkt-zu-Punkt- als auch Publish/Subscribe-Kommunikation zu unterstützen).

Führen Sie die folgenden Schritte aus, um dieses Problem zu beheben:

1. Greifen Sie auf den generischen JMS-Provider zu, den Sie verwenden.
2. Ersetzen Sie die definierte JMS 1.0.2-Topic-Verbindungsfactory durch eine (einheitliche) JMS 1.1-Verbindungsfactory.

Wenn Sie die Anwendung nun mit der neu definierten JMS 1.1-Verbindungsfactory starten, dürfte Sie eigentlich keine Fehlernachricht mehr erhalten.

Generische JMS-basierte SCA-Nachrichten werden im Failed Event Manager nicht angezeigt

Wenn die SCA-Nachrichten ihren Ursprung in einem Fehler oder einer Störung in der generischen JMS-Interaktion haben, würden Sie erwarten, diese Nachrichten im Failed Event Manager vorzufinden. Werden keine derartigen Nachrichten im Failed Event Manager angezeigt, stellen Sie sicher, dass für den zugrunde liegenden Listener-Port ein Wert größer als 1 für die Eigenschaft der maximalen Anzahl von Wiederholungsversuchen definiert ist. Durch Festlegen von 2 oder höher für diesen Wert wird eine Interaktion mit dem Failed Event Manager während SCA-Aufrufen für die generischen JMS-Bindungen ermöglicht.

Ausnahmebedingungen verarbeiten:

Die Art und Weise, in der die Bindung konfiguriert ist, bestimmt, wie Ausnahmebedingungen verarbeitet werden, die von Datenhandlern oder Datenbindungen ausgelöst werden. Außerdem gibt die Spezifik des Mediationsablaufs das Verhalten des Systems vor, wenn eine solche Ausnahmebedingung ausgelöst wird.

Wenn ein Datenhandler oder eine Datenbindung durch eine Bindung aufgerufen wird, können verschiedene Probleme auftreten. Beispielsweise kann es sein, dass ein Datenhandler eine Nachricht mit beschädigten Nutzdaten empfängt oder er versucht, eine Nachricht mit einem falschen Format zu lesen.

Das Verfahren, mit dem die Bindung eine solche Ausnahmebedingung behandelt, wird dadurch bestimmt, wie Sie den Datenhandler oder die Datenbindung implementieren. Das empfohlene Verhalten besteht darin, die Datenbindung so zu konfigurieren, dass sie eine Ausnahmebedingung des Typs **DataBindingException** auslöst.

Bei einem Datenhandler ist die Situation ähnlich. Da der Datenhandler durch die Datenbindung aufgerufen wird, wird jede Ausnahmebedingung des Datenhandlers in eine Ausnahmebedingung der Datenbindung eingeschlossen. Daher wird eine Ausnahmebindung des Typs **DataHandlerException** als Typ **DataBindingException** gemeldet.

Wenn eine Laufzeitausnahmebedingung (inklusive **DataBindingException**) ausgelöst wird, geschieht Folgendes:

- Falls der Mediationsablauf transaktionsorientiert konfiguriert ist, wird die JMS-Nachricht standardmäßig in Failed Event Manager gespeichert, damit sie manuell wiedergegeben oder gelöscht werden kann.

  Anmerkung: Sie können den Fehlerbehebungsmodus für die Bindung so ändern, dass die Nachricht zurückgesetzt und nicht in failed event manager gespeichert wird.
- Ist der Mediationsablauf nicht transaktionsorientiert, wird die Ausnahmebedingung protokolliert und die Nachricht ist nicht mehr vorhanden.

Bei einem Datenhandler ist die Situation ähnlich. Da der Datenhandler durch die Datenbindung aufgerufen wird, wird jede Ausnahmebedingung des Datenhandlers in eine Ausnahmebedingung der Datenbindung eingeschlossen. Daher wird eine Ausnahmebindung des Typs **DataHandlerException** als Typ **DataBindingException** gemeldet.

WebSphere MQ-JMS-Bindungen:

Die WebSphere MQ-JMS-Bindung ermöglicht die Integration bei externen Anwendungen, die einen JMS-basierten WebSphere MQ-Provider verwenden.

Mit den WebSphere MQ-JMS-Exportbindungen und -Importbindungen können Sie eine direkte Integration bei externen JMS- oder MQ-JMS-Systemen aus Ihrer Serverumgebung heraus erreichen. Die Verwendung der Funktionen für die MQ- oder Clientverbindung von Service Integration Bus ist dann nicht mehr erforderlich.

Wenn eine Komponente mit einem JMS-basierten WebSphere MQ-Service über einen Import interagiert, verwendet die WebSphere MQ-JMS-Importbindung ein Ziel, an das Daten gesendet werden, und ein Ziel, an dem die Antwort empfangen werden kann. Die Konvertierung der Daten in eine und aus einer JMS-Nachricht wird durch die Ersterkennungskomponente des JMS-Datenhandlers oder der JMS-Datenbindung ausgeführt.

Wenn ein SCA-Modul einen Service für WebSphere MQ-JMS-Clients bereitstellt, verwendet die WebSphere MQ-JMS-Exportbindung ein Ziel, an dem die Anforderung empfangen und an das die Antwort gesendet werden kann. Die Konvertierung der Daten in eine und aus einer JMS-Nachricht wird durch den JMS-Datenhandler oder die JMS-Datenbindung vorgenommen.

Der Funktionsselektor stellt eine Zuordnung zu der Operation in der aufzurufenden Zielkomponente bereit.

WebSphere MQ-JMS-Bindungen - Übersicht:

Die WebSphere MQ-JMS-Bindung ermöglicht die Integration bei externen Anwendungen, die den WebSphere MQ-JMS-Provider verwenden.

WebSphere MQ-Verwaltungstasks

Der WebSphere MQ-Systemadministrator muss den zugrunde liegenden WebSphere MQ Queue Manager, den die WebSphere MQ-JMS-Bindungen verwenden, erstellen, bevor eine Anwendung ausgeführt wird, die diese Bindungen enthält.

WebSphere MQ-JMS-Importbindungen

Durch den WebSphere MQ-JMS-Import können Komponenten in Ihrem SCA-Modul mit Services kommunizieren, die von JMS-basierten WebSphere MQ-Providern bereitgestellt werden. Sie müssen eine unterstützte Version von WebSphere MQ verwenden. Ausführliche Angaben über die Hardware- und Softwarevoraussetzungen finden Sie auf den Seiten von IBM Support.

Es werden zwei Typen von Einsatzszenarios für WebSphere MQ-JMS-Importbindungen unterstützt, die sich nach dem Typ der aufgerufenen Operation richten:

- Unidirektional: Der WebSphere MQ-JMS-Import übergibt eine Nachricht an das Sendeziel, das in der Importbindung konfiguriert ist. An das Feld **replyTo** des JMS-Headers werden keine Daten gesendet.
- Bidirektional (Anforderung-Antwort): Der WebSphere MQ-JMS-Import übergibt eine Nachricht an das Sendeziel.

  Das Empfangsziel (**receive**) ist im Headerfeld **replyTo** festgelegt. Für die Überwachung des Empfangsziels wird eine nachrichtengesteuerte Bean implementiert. Sobald eine Antwort empfangen wird, übergibt die nachrichtengesteuerte Bean die Antwort zurück an die Komponente.

  Die Importbindung kann (mit dem Feld **Korrelationsschema für Antwort** in Integration Designer) so konfiguriert werden, dass das Kopieren der Korrelations-ID für die Antwortnachricht aus der Anforderungsnachrichten-ID (Standardeinstellung) oder aus der Korrelations-ID für die Anforderungsnachricht erwartet wird.

Sowohl bei unidirektionalen als auch bei bidirektionalen Einsatzszenarios können dynamische und statische Headereigenschaften angegeben werden. Statische Eigenschaften können über die Methodenbindung des JMS-Imports festgelegt werden. Manche dieser Eigenschaften haben in der SCA-JMS-Laufzeit eine besondere Bedeutung.

Es muss unbedingt beachtet werden, dass es sich bei WebSphere MQ-JMS um eine asynchrone Bindung handelt. Falls eine aufrufende Komponente einen WebSphere-JMS-Import (bei einer bidirektionalen Operation) im Synchronmodus aufruft, wird die aufrufende Komponente blockiert, bis die Antwort durch den JMS-Service zurückgegeben wurde.

In Abb. 37 auf Seite 140 ist dargestellt, wie der Import mit dem externen Service verknüpft ist.

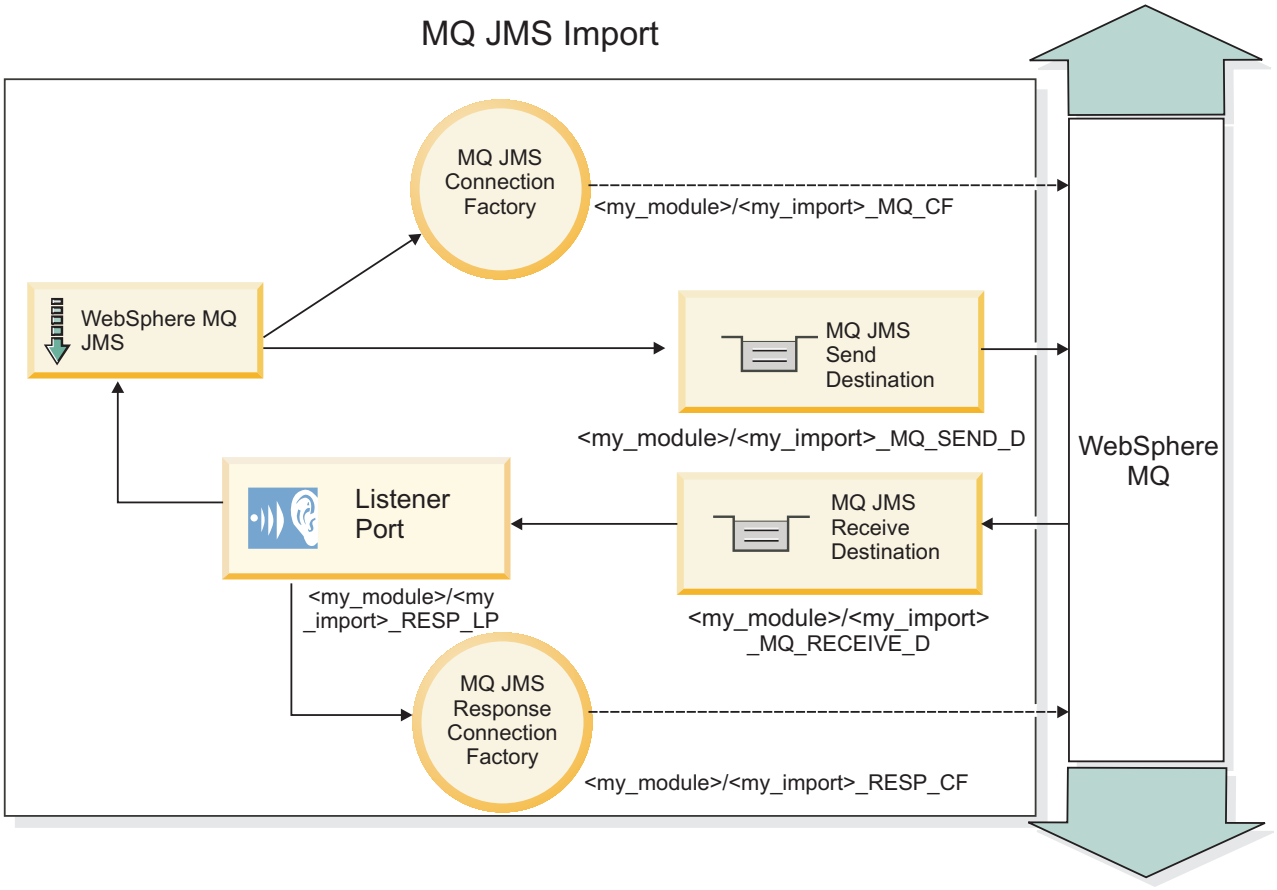


Abbildung 37. Ressourcen der WebSphere MQ-JMS-Importbindung

WebSphere MQ-JMS-Exportbindungen

Die WebSphere MQ-JMS-Exportbindung stellt für SCA-Module ein Verfahren bereit, mit dem Services für externe Anwendungen beim WebSphere MQ-basierten JMS-Provider angeboten werden können.

Zur Überwachung von Anforderungen, die an dem in der Exportbindung angegebenen Empfangsziel eingehen, wird eine nachrichtengesteuerte Bean implementiert. Das im Feld **send** angegebene Ziel wird verwendet, um die Antwort auf die eingehende Anforderung zu senden, falls die aufgerufene Komponente eine Antwort bereitstellt. Das im Feld **replyTo** der Antwortnachricht angegebene Ziel überschreibt das im Feld **send** angegebene Ziel.

In Abb. 38 auf Seite 141 ist dargestellt, wie der externe Anforderer mit dem Export verknüpft ist.

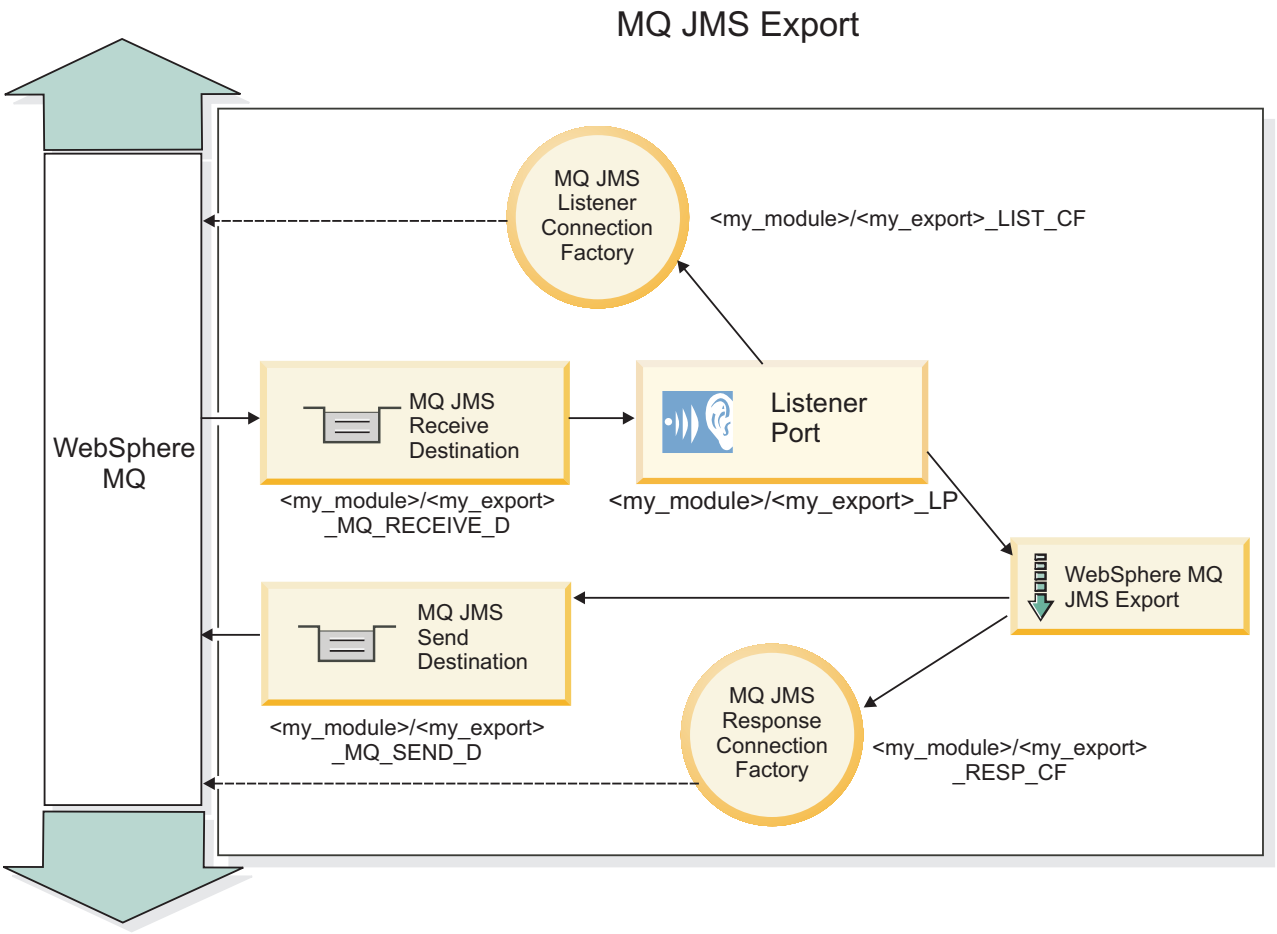


Abbildung 38. Ressourcen der WebSphere MQ-JMS-Exportbindung

Anmerkung: Abb. 37 auf Seite 140 und Abb. 38 veranschaulichen, wie eine Anwendung aus einer früheren Version von IBM Business Process Manager mit einem externen Service verbunden wird. Bei Anwendungen, die für IBM Business Process Manager Version 7.0 entwickelt werden, wird anstelle des Listener-Ports und der Verbindungsfactory die Aktivierungsspezifikation verwendet.

Schlüsselfunktionen von WebSphere MQ-JMS-Bindings:

Die Schlüsselfunktionen von WebSphere MQ JMS-Bindings umfassen Header, Java EE-Artefakte und erstellte Java EE-Ressourcen.

Header

Ein JMS-Nachrichtenheader enthält eine Reihe vordefinierter Felder, in denen Werte enthalten sind, die sowohl von den Clients als auch den Providern zum Angeben und Weiterleiten von Nachrichten verwendet werden. Sie können diese Header anhand der Bindingeigenschaften mit festen Werten konfigurieren, die Header können aber auch dynamisch zur Laufzeit angegeben werden.

`JMSCorrelationID`
Stellt eine Verbindung zur zugehörigen Nachricht her. In der Regel wird in diesem Feld die Zeichenfolge der Nachrichten-ID für die Nachricht eingestellt, auf die geantwortet wird.

`TargetFunctionName`
Dieser Header wird von einem der angegebenen Funktionsselektoren zum Angeben der Operation verwendet, die aufgerufen wird. Wenn die JMS-Headereigenschaft 'TargetFunctionName' in Nachrichten eingestellt wird, die an einen JMS-Export gesendet werden, kann dieser Funktionsselektor ver-

wendet werden. Die Eigenschaft kann direkt in den JMS-Clientanwendungen eingestellt werden oder wenn eine Verbindung zwischen einem Import mit einem JMS-Binding und einem solchen Export hergestellt wird. In diesem Fall muss das JMS-Importbinding so konfiguriert sein, dass der Header 'TargetFunctionName' für jede Operation in der Schnittstelle als Name der Operation festgelegt wird.

Korrelationsschemas

Von den WebSphere MQ-JMS-Bindings werden verschiedene Korrelationsschemas bereitgestellt, die dazu verwendet werden, festzustellen, wie Anforderungsnachrichten und Antwortnachrichten miteinander korrelieren.

RequestMsgIDToCorrelID
Die JMSMessageID wird in das Feld JMSCorrelationID kopiert. Dies ist die Standardeinstellung.

RequestCorrelIDToCorrelID
Die JMSCorrelationID wird in das Feld JMSCorrelationID kopiert.

Java EE-Ressourcen

Wenn ein MQ-JMS-Import in einer Java EE-Umgebung implementiert wird, werden Java EE-Ressourcen erstellt.

Parameter

MQ-Verbindungsfactory
Wird von Clients zum Erstellen einer Verbindung zum MQ-JMS-Provider verwendet.

Antwortverbindungsfactory
Wird von der SCA-MQ-JMS-Laufzeit verwendet, wenn sich das Sendeziel in einem anderen Queue Manager als das Empfangsziel befindet.

Aktivierungsspezifikation
Eine MQ-JMS-Aktivierungsspezifikation ist mindestens einer Message-driven Bean zugeordnet und stellt die Konfiguration bereit, die zum Empfangen von Nachrichten erforderlich ist.

Ziele

- Sendeziel:
  - Importe: Das Ziel, an das die Anforderung oder abgehende Nachricht gesendet wird.
  - Exporte: Das Ziel, an das die Antwortnachricht gesendet wird, falls diese nicht durch das Headerfeld JMSReplyTo der eingehenden Nachricht außer Kraft gesetzt wird.
- Empfangsziel:
  - Importe: Das Ziel, an das die Antwort oder eingehende Nachricht gesendet werden soll.
  - Exporte: Das Ziel, an das die eingehende oder Anforderungsnachricht gesendet werden soll.

JMS-Header:

Eine JMS-Nachricht enthält zwei Typen von Headern, nämlich den JMS-Systemheader und mehrere JMS-Eigenschaften. Auf beide Headertypen kann entweder in einem Mediationsmodul im Servicenachrichtenobjekt (Service Message Object - SMO) oder unter Verwendung der API 'ContextService' zugegriffen werden.

JMS-Systemheader

Der JMS-Systemheader wird im Servicenachrichtenobjekt durch das Element 'JMSHeader' repräsentiert, das alle normalerweise in einem JMS-Header zu findenden Felder enthält. Obwohl diese Felder in der Mediation (oder der API 'ContextService') geändert werden können, werden einige der im Servicenachrichtenobjekt festgelegten Felder für den JMS-Systemheader nicht an die abgehende JMS-Nachricht weitergegeben, weil sie durch Systemwerte oder statische Werte überschrieben werden.

Die folgenden Schlüsselfelder im JMS-Systemheader können in einer Mediation (oder einer API 'Context-Service') aktualisiert werden:

- **JMSType** und **JMSCorrelationID** - Werte der spezifischen vordefinierten Nachrichtenheadereigenschaften.
- **JMSDeliveryMode**: Werte für den Übermittlungsmodus ('persistent' oder 'nonpersistent'; Standardwert ist 'persistent').
- **JMSPriority**: Prioritätswert (0 bis 9; Standardwert ist 'JMS_Default_Priority').

JMS-Eigenschaften

JMS-Eigenschaften werden im Servicenachrichtenobjekt als Einträge in der Eigenschaftsliste ('Properties') dargestellt. Die Eigenschaften können in einer Mediation oder durch die Verwendung der API 'Context-Service' hinzugefügt, aktualisiert oder gelöscht werden.

Eigenschaften können auch in der JMS-Bindung statisch festgelegt sein. Statisch festgelegte Eigenschaften überschreiben Einstellungen desselben Namens, die dynamisch festgelegt werden.

Benutzereigenschaften, die von anderen Bindungen (z. B. einer HTTP-Bindung) weitergegeben werden, werden in der JMS-Bindung als JMS-Eigenschaften ausgegeben.

Einstellungen für Headerweitergabe

Die Weitergabe des JMS-Systemheaders und der JMS-Eigenschaften entweder von der eingehenden JMS-Nachricht an nachgelagerte Komponenten oder von vorgelagerten Komponenten an die abgehende JMS-Nachricht kann durch das Attribut 'Protokollheader weitergeben' der Bindung gesteuert werden.

Wenn das Attribut 'Protokollheader weitergeben' festgelegt ist, ist die Übermittlung von Headerinformationen an die Nachricht oder die Zielkomponente, wie in der folgenden Liste beschrieben, zulässig:

- JMS-Exportanforderung

  Der in der Nachricht empfangene JMS-Header wird mittels des Kontextservice an Zielkomponenten weitergegeben. In der Nachricht empfangene JMS-Eigenschaften werden mittels des Kontextservice an Zielkomponenten weitergegeben.
- JMS-Exportantwort

  Alle im Kontextservice definierten JMS-Headerfelder werden in der abgehenden Nachricht verwendet, sofern sie nicht durch statische Eigenschaften überschrieben werden, die in der JMS-Exportbindung festgelegt sind. Alle im Kontextservice definierten Eigenschaften werden in der abgehenden Nachricht verwendet, sofern sie nicht durch statische Eigenschaften überschrieben werden, die in der JMS-Exportbindung festgelegt sind.
- JMS-Importanforderung

  Alle im Kontextservice definierten JMS-Headerfelder werden in der abgehenden Nachricht verwendet, sofern sie nicht durch statische Eigenschaften überschrieben werden, die in der JMS-Importbindung festgelegt sind. Alle im Kontextservice definierten Eigenschaften werden in der abgehenden Nachricht verwendet, sofern sie nicht durch statische Eigenschaften überschrieben werden, die in der JMS-Importbindung festgelegt sind.
- JMS-Importantwort

  Der in der Nachricht empfangene JMS-Header wird mittels des Kontextservice an Zielkomponenten weitergegeben. In der Nachricht empfangene JMS-Eigenschaften werden mittels des Kontextservice an Zielkomponenten weitergegeben.

Externe Clients:

Der Server kann mit WebSphere MQ-JMS-Bindungen Nachrichten an externe Clients senden bzw. von diesen empfangen.

Ein externer Client (z. B. ein Webportal oder ein unternehmensweites Informationssystem) kann eine Nachricht an eine SCA-Komponente in der Anwendung durch einen Export senden oder von einer SCA-Komponente in der Anwendung durch einen Import aufgerufen werden.

Die WebSphere MQ-JMS-Exportbindung implementiert nachrichtengesteuerte Beans (Message driven bean - MDB), um Anforderungen zu überwachen, die bei dem in der Exportbindung definierten Empfangsziel eingehen. Das im Feld **send** angegebene Ziel wird verwendet, um die Antwort auf die eingehende Anforderung zu senden, falls die aufgerufene Anwendung eine Antwort bereitstellt. Ein externer Client kann somit Anwendungen über die Exportbindung aufrufen.

WebSphere MQ-JMS-Importe stellen eine Bindung zu externen Clients her und können Nachrichten an diese Clients übermitteln. Diese Nachrichten machen möglicherweise, jedoch nicht zwangsläufig, eine Antwort an den externen Client erforderlich.

Weitere Informationen zur Interaktion mit externen Clients unter Verwendung von WebSphere MQ finden Sie im Information Center von WebSphere MQ.

Fehlerbehebung für WebSphere MQ-JMS-Bindungen:

Sie können Probleme im Zusammenhang mit WebSphere MQ-JMS-Bindungen diagnostizieren und beheben.

Ausnahmebedingungen bei der Implementierung

Als Reaktion auf diverse Fehlerbedingungen kann die MQ-JMS-Implementierung für Importe und Exporte zwei Typen von Ausnahmebedingungen zurückgeben:

- Geschäftsausnahmebedingung für Service: Diese Ausnahmebedingung wird zurückgegeben, wenn der für die Geschäftsschnittstelle für den Service (Typ WSDL-Port) definierte Fehler aufgetreten ist.
- Laufzeitausnahmebedingung für den Service: Diese Ausnahmebedingung wird in allen übrigen Fällen ausgelöst. In den meisten Fällen enthält die Ausnahmebedingung vom Typ cause (Ursache) die ursprüngliche Ausnahmebedingung (JMSException).

  Ein Import erwartet zum Beispiel nur eine einzige Antwortnachricht für jede Anforderungsnachricht. Wenn mehr als eine Antwort eingeht oder wenn eine verspätete Antwort eingeht (d. h. eine Antwort, für die die definierte SCA-Antwortablauffrist verstrichen ist), so wird eine Laufzeitausnahmebedingung für den Service ausgelöst. Für die Transaktion wird ein Rollback ausgeführt und die Antwortnachricht wird aus der Warteschlange entfernt oder vom Failed Event Manager verarbeitet.

WebSphere MQ-JMS-basierte SCA-Nachrichten werden im Failed Event Manager nicht angezeigt

Wenn die SCA-Nachrichten ihren Ursprung in einem Fehler oder einer Störung in der WebSphere MQ-JMS-Interaktion haben, würden Sie erwarten, diese Nachrichten im Failed Event Manager vorzufinden. Werden keine derartigen Nachrichten im Failed Event Manager angezeigt, stellen Sie sicher, dass für den zugrunde liegenden Listener-Port ein Wert größer als **1** für die Eigenschaft der maximalen Anzahl von Wiederholungsversuchen definiert ist. Durch Festlegen von **2** oder höher für diesen Wert wird eine Interaktion mit dem Failed Event Manager während SCA-Aufrufen für die MQ-JMS-Bindungen ermöglicht.

Szenario falscher Verwendungen: Vergleich mit WebSphere MQ-Bindungen

Die WebSphere MQ-JMS-Bindung wurde dazu konzipiert, mit JMS-Anwendungen zu interagieren, die auf WebSphere MQ implementiert wurden, was eine Offenlegung von Nachrichten gemäß dem JMS-Nachrichtenmodell bewirkt. Import und Export von WebSphere MQ wurden jedoch in erster Linie entwickelt, um mit nativen WebSphere MQ-Anwendungen zu interagieren und den gesamten Inhalt des WebSphere MQ-Nachrichtenhauptteils gegenüber Mediationen verfügbar zu machen.

Die folgenden Szenarien sollten unter Verwendung der WebSphere MQ-JMS-Bindung erstellt werden und nicht mit der WebSphere MQ-Bindung:

- Aufrufen einer JMS-Message-driven Bean (MDB) von einem SCA-Modul, wenn die MDB auf dem WebSphere MQ-JMS-Provider implementiert ist. Verwenden Sie in diesem Fall einen WebSphere MQ-JMS-Import.
- Zulassen des Aufrufs des SCA-Moduls von einem Java EE-Komponentenservlet oder einer EJB anhand eines JMS. Verwenden Sie in diesem Fall einen WebSphere MQ-JMS-Export.
- Mediation des Inhalts einer JMS-Zuordnungsnachricht beim Durchqueren von WebSphere MQ betreiben. Verwenden Sie einen WebSphere MQ-JMS-Export und -Import in Verbindung mit dem entsprechenden Datenhandler oder der entsprechenden Datenbindung.

In gibt Fälle, in denen eine Interaktion zwischen WebSphere MQ-Bindung und WebSphere MQ-JMS-Bindung gegebenenfalls erwartet wird. Besonders bei Überbrückungen zwischen Java EE- und Nicht-Java-EE-Anwendungen von WebSphere MQ sollten Sie einen WebSphere MQ-Export und einen WebSphere MQ-JMS-Import (oder umgekehrt) in Verbindung mit den geeigneten Datenbindungen oder Mediationsmodulen (oder beiden) verwenden.

Ausnahmebedingungen verarbeiten:

Die Art und Weise, in der die Bindung konfiguriert ist, bestimmt, wie Ausnahmebedingungen verarbeitet werden, die von Datenhandlern oder Datenbindungen ausgelöst werden. Außerdem gibt die Spezifik des Mediationsablaufs das Verhalten des Systems vor, wenn eine solche Ausnahmebedingung ausgelöst wird.

Wenn ein Datenhandler oder eine Datenbindung durch eine Bindung aufgerufen wird, können verschiedene Probleme auftreten. Beispielsweise kann es sein, dass ein Datenhandler eine Nachricht mit beschädigten Nutzdaten empfängt oder er versucht, eine Nachricht mit einem falschen Format zu lesen.

Das Verfahren, mit dem die Bindung eine solche Ausnahmebedingung behandelt, wird dadurch bestimmt, wie Sie den Datenhandler oder die Datenbindung implementieren. Das empfohlene Verhalten besteht darin, die Datenbindung so zu konfigurieren, dass sie eine Ausnahmebedingung des Typs **DataBindingException** auslöst.

Bei einem Datenhandler ist die Situation ähnlich. Da der Datenhandler durch die Datenbindung aufgerufen wird, wird jede Ausnahmebedingung des Datenhandlers in eine Ausnahmebedingung der Datenbindung eingeschlossen. Daher wird eine Ausnahmebindung des Typs **DataHandlerException** als Typ **DataBindingException** gemeldet.

Wenn eine Laufzeitausnahmebedingung (inklusive **DataBindingException**) ausgelöst wird, geschieht Folgendes:

- Falls der Mediationsablauf transaktionsorientiert konfiguriert ist, wird die JMS-Nachricht standardmäßig in Failed Event Manager gespeichert, damit sie manuell wiedergegeben oder gelöscht werden kann.

  Anmerkung: Sie können den Fehlerbehebungsmodus für die Bindung so ändern, dass die Nachricht zurückgesetzt und nicht in failed event manager gespeichert wird.
- Ist der Mediationsablauf nicht transaktionsorientiert, wird die Ausnahmebedingung protokolliert und die Nachricht ist nicht mehr vorhanden.

Bei einem Datenhandler ist die Situation ähnlich. Da der Datenhandler durch die Datenbindung aufgerufen wird, wird jede Ausnahmebedingung des Datenhandlers in eine Ausnahmebedingung der Datenbindung eingeschlossen. Daher wird eine Ausnahmebindung des Typs **DataHandlerException** als Typ **DataBindingException** gemeldet.

WebSphere MQ-Bindungen:

Die WebSphere MQ-Bindung ist so konzipiert, dass sie SCA-Konnektivität mit WebSphere MQ-Anwendungen bereitstellt.

Mit den WebSphere MQ-Exportbindungen und -Importbindungen können Sie eine direkte Integration bei einem WebSphere MQ-basierten System aus Ihrer Serverumgebung heraus erreichen. Die Verwendung der Funktionen für die MQ- oder Clientverbindung von Service Integration Bus ist dann nicht mehr erforderlich.

Wenn eine Komponente mit einem WebSphere MQ-Service über einen Import interagiert, verwendet die WebSphere MQ-Importbindung eine Warteschlange, an die Daten gesendet werden, und eine Warteschlange, in der die Antwort empfangen werden kann.

Wenn ein SCA-Modul einen Service für WebSphere MQ-Clients bereitstellt, verwendet die WebSphere MQ-Exportbindung eine Warteschlange, in der Anforderung empfangen und an die die Antwort gesendet werden kann. Der Funktionsselektor stellt eine Zuordnung zu der Operation in der aufzurufenden Zielkomponente bereit.

Die Konvertierung der Nutzdaten in eine und aus einer MQ-Nachricht wird durch den Datenhandler oder die Datenbindung des MQ-Hauptteils vorgenommen. Die Konvertierung der Headerdaten in eine und aus einer MQ-Nachricht wird die durch Datenbindung des MQ-Headers vorgenommen.

Informationen zu den unterstützten WebSphere MQ-Versionen finden Sie auf der Webseite detailed system requirements.

WebSphere MQ-Bindungen - Übersicht:

Die WebSphere MQ-Bindung ermöglicht die Integration bei nativen MQ-basierten Anwendungen.

WebSphere MQ-Verwaltungstasks

Der WebSphere MQ-Systemadministrator muss den zugrunde liegenden WebSphere MQ Queue Manager, den die WebSphere MQ-Bindungen verwenden, erstellen, bevor eine Anwendung ausgeführt wird, die diese Bindungen enthält.

WebSphere-Verwaltungstasks

Sie müssen die Eigenschaft für den **Pfad der nativen Bibliothek** des MQ-Ressourcenadapters in WebSphere mit der vom Server unterstützten WebSphere MQ-Version festlegen und den Server erneut starten. Dies stellt sicher, dass die Bibliotheken einer unterstützten Version von WebSphere MQ verwendet werden. Detaillierte Informationen zu Hardware- und Softwarevoraussetzung finden Sie auf den IBM-Unterstützungsseiten.

WebSphere MQ-Importbindungen

Durch die WebSphere MQ-Importbindung können Komponenten in Ihrem SCA-Modul mit Services kommunizieren, die von externen WebSphere MQ-basierten Anwendungen bereitgestellt werden. Sie müssen eine unterstützte Version von WebSphere MQ verwenden. Detaillierte Informationen zu Hardware- und Softwarevoraussetzung finden Sie auf den IBM-Unterstützungsseiten.

Die Interaktion mit externen WebSphere-Systemen umfasst auch die Verwendung von Warteschlangen für das Senden von Anforderungen und das Empfangen von Antworten.

Es werden zwei Typen von Einsatzszenarios für die WebSphere MQ-Importbindungen unterstützt, die sich nach dem Typ der aufgerufenen Operation richten:

- Unidirektional: Der WebSphere MQ-Import stellt eine Nachricht in die Warteschlange, die im Feld für die **Sendezielwarteschlange** der Importbindung konfiguriert ist. An das Feld **replyTo** des MQMD-Headers werden keine Daten gesendet.
- Bidirektional: Der WebSphere MQ-Import stellt eine Nachricht in die Warteschlange, die im Feld für die **Sendezielwarteschlange** konfiguriert ist.

  Die Empfangswarteschlange (**receive**) ist im MQMD-Headerfeld **replyTo** festgelegt. Für die Überwachung der Empfangswarteschlange wird eine nachrichtengesteuerte Bean implementiert. Sobald eine Antwort empfangen wird, übergibt die nachrichtengesteuerte Bean die Antwort zurück an die Komponente.

  Die Importbindung kann (mit dem Feld **Korrelationsschema für Antwort**) so konfiguriert werden, dass das Kopieren der Korrelations-ID für die Antwortnachricht aus der Anforderungsnachrichten-ID (Standardeinstellung) oder aus der Korrelations-ID für die Anforderungsnachricht erwartet wird.

Es muss unbedingt beachtet werden, dass es sich bei WebSphere MQ um eine asynchrone Bindung handelt. Falls eine aufrufende Komponente einen WebSphere MQ-Import (bei einer bidirektionalen Operation) im Synchronmodus aufruft, wird die aufrufende Komponente blockiert, bis die Antwort durch den WebSphere MQ-Service zurückgegeben wurde.

In Abb. 39 ist dargestellt, wie der Import mit dem externen Service verknüpft ist.

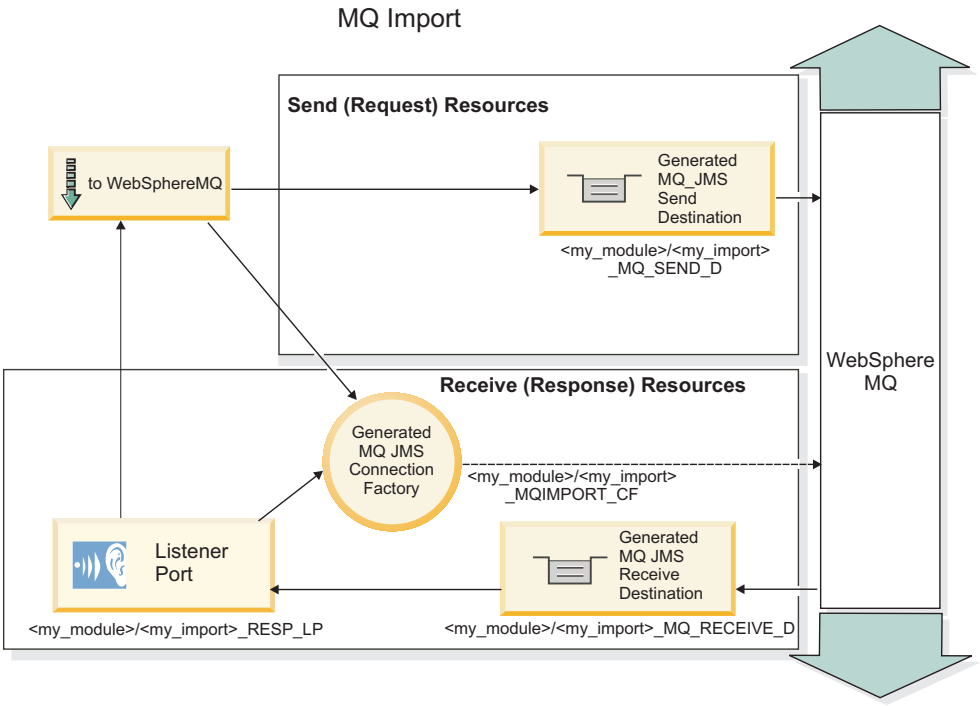


Abbildung 39. Ressourcen der WebSphere MQ-Importbindung

WebSphere MQ-Exportbindungen

Die WebSphere MQ-Exportbindung stellt SCA-Modulen ein Verfahren bereit, mit dem Services für externe WebSphere MQ-basierte Anwendungen angeboten werden können.

Zur Überwachung von Anforderungen, die in der in der Exportbindung angegebenen **Empfangszielwartschlange** eingehen, wird eine nachrichtengesteuerte Bean implementiert. Die im Feld für die **Sendezielwarteschlange** angegebene Warteschlange wird verwendet, um die Antwort auf die eingehende Anforde-

rung zu senden, falls die aufgerufene Komponente eine Antwort bereitstellt. Die im Feld **replyTo** der Antwortnachricht angegebene Warteschlange überschreibt die im Feld für die **Sendezielwarteschlange** angegebene Warteschlange.

In Abb. 40 ist dargestellt, wie der externe Anforderer mit dem Export verknüpft ist.

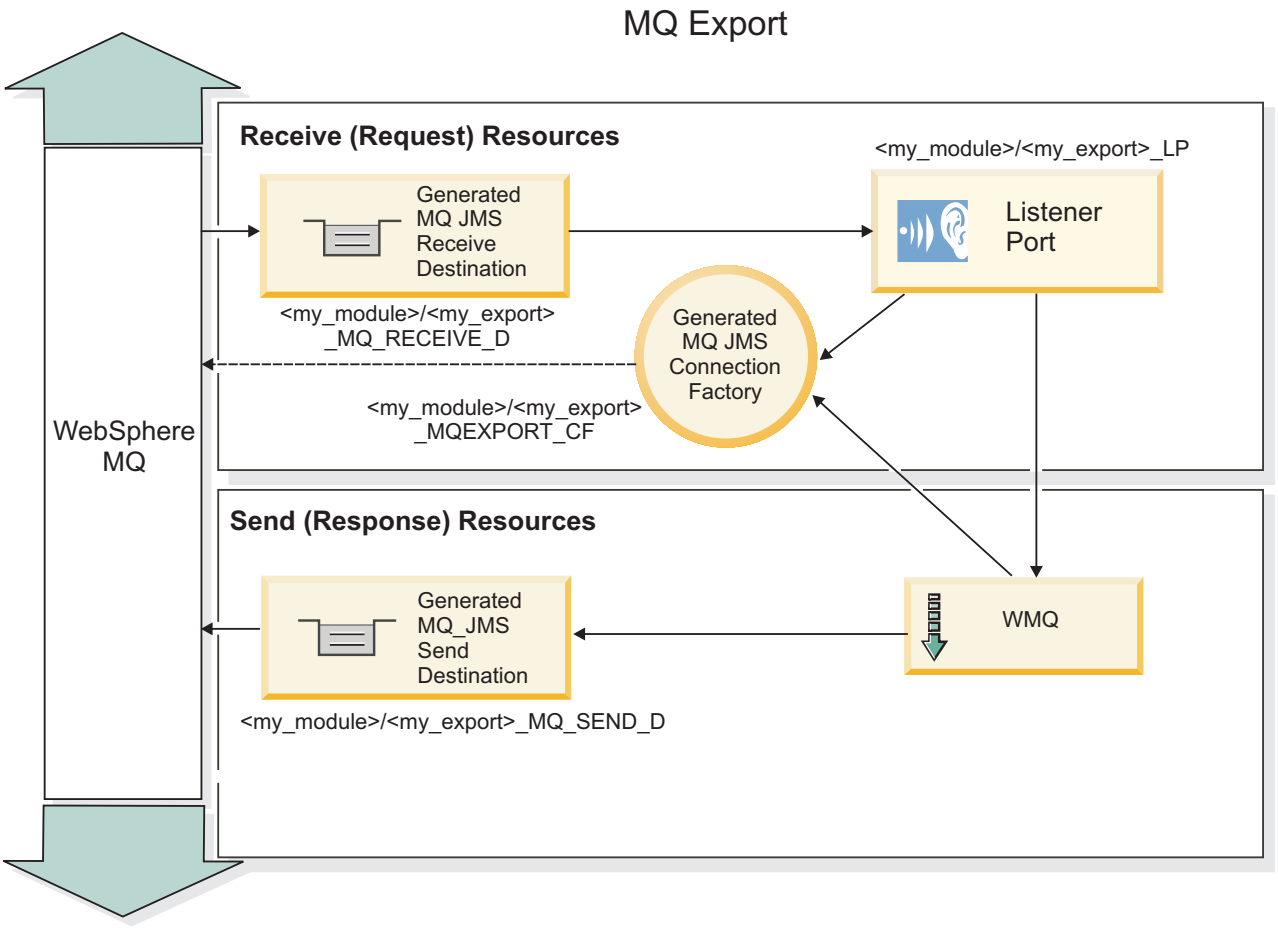


Abbildung 40. Ressourcen der WebSphere MQ-Exportbindung

Anmerkung: Abb. 39 auf Seite 147 und Abb. 40 veranschaulichen, wie eine Anwendung aus einer früheren Version von IBM Business Process Manager mit einem externen Service verbunden wird. Bei Anwendungen, die für IBM Business Process Manager Version 7.x oder aktuellere Versionen entwickelt werden, wird anstelle des Listener-Ports und der Verbindungsfactory die Aktivierungsspezifikation verwendet.

Schlüsselfunktionen von WebSphere MQ-Bindings:

Die Schlüsselfunktionen von WebSphere MQ-Bindings umfassen Header, Java EE-Artefakte und erstellte Java EE-Ressourcen.

Korrelationsschemas

Von einer WebSphere MQ-Anforderungs-/Antwortanwendung kann ein Verfahren aus einer Reihe verschiedener Verfahren zum Korrelieren von Antwortnachrichten mit Anforderungen verwendet werden; für die Erstellung werden die MQMD-Felder MessageID und CorrelID verwendet. Im Großteil der Fälle lässt der Anforderer den Warteschlangenmanager eine MessageID auswählen und erwartet, dass die antwortende Anwendung diese in die CorrelID der Antwort kopiert. In den meisten Fällen wissen der Anforderer und die antwortende Anwendung implizit, welches Korrelationsverfahren verwendet wird. In

manchen Fällen berücksichtigt die antwortende Anwendung verschiedene Markierungen im Feld Report der Anforderung, aus denen hervorgeht, wie mit diesen Feldern umgegangen werden soll.

Exportbindings für WebSphere MQ-Nachrichten können mit den folgenden Optionen konfiguriert werden:

Optionen für Nachrichten-ID der Antwort:

Neue Nachrichten-ID
Bei Auswahl dieser Option kann der Warteschlangenmanager eine eindeutige MsgId für die Antwort auswählen (Standardeinstellung).

Aus Nachrichten-ID der Anforderung kopieren
Kopiert die Angabe im Feld MsgId aus dem Feld MsgId in die Anforderung.

Aus SCA-Nachricht kopieren
Legt fest, dass die MsgId in den WebSphere MQ-Headern in die SCA-Antwortnachricht übertragen werden soll oder lässt den Warteschlangenmanager eine neue ID definieren, wenn kein Wert vorhanden ist.

Wie bei Berichtsoptionen
Überprüft das Feld Report von MQMD in der Anforderung auf einen Hinweis zum Umgang mit der MsgId. Die Optionen MQRO_NEW_MSG_ID und MQRO_PASS_MSG_ID werden unterstützt, ihr Verhalten entspricht New MsgId bzw. Copy from Request MsgID.

Optionen für Korrelations-ID der Antwort:

Aus Nachrichten-ID der Anforderung kopieren
Kopiert die Angabe im Feld CorrelId aus dem Feld MsgId in die Anforderung (Standardeinstellung).

Korrelations-ID-Kopie aus Anforderungskorrelations-ID
Kopiert die Angabe im Feld CorrelId aus dem Feld CorrelId in die Anforderung.

Aus SCA-Nachricht kopieren
Legt fest, dass die CorrelId in den WebSphere MQ-Headern in die SCA-Antwortnachricht übertragen werden soll oder lässt das Feld leer, wenn kein Wert vorhanden ist.

Wie bei Berichtsoptionen
Überprüft das Feld Report von MQMD in der Anforderung auf einen Hinweis zum Umgang mit der CorrelId. Die Optionen MQRO_COPY_MSG_ID_TO_CORREL_ID und MQRO_PASS_CORREL_ID werden unterstützt, ihr Verhalten entspricht dem von Copy from Request MsgID bzw. Copy from Request CorrelID.

Importbindings für WebSphere MQ-Nachrichten können mit den folgenden Optionen konfiguriert werden:

Optionen für Nachrichten-ID der Anforderung:

Neue Nachrichten-ID
Bei Auswahl dieser Option kann der Warteschlangenmanager eine eindeutige MsgId für die Anforderung auswählen (Standardeinstellung).

Aus SCA-Nachricht kopieren
Legt fest, dass die MsgId in den WebSphere MQ-Headern in die SCA-Anforderungsnachricht übertragen werden soll oder lässt den Warteschlangenmanager eine neue ID definieren, wenn kein Wert vorhanden ist.

Optionen für Korrelations-ID der Antwort:

Response has CorrelID copied from MsgId
Erwartet, dass in der Antwortnachricht ein Feld mit der Bezeichnung CorrelId eingestellt ist und dafür die MsgId der Anforderung verwendet wird (Standardeinstellung).

Response has MsgID copied from MsgId
Erwartet, dass in der Antwortnachricht ein Feld mit der Bezeichnung MsgId eingestellt ist und dafür die MsgId der Anforderung verwendet wird.

Response has CorrelID copied from CorrelId
Erwartet, dass in der Antwortnachricht ein Feld mit der Bezeichnung CorrelId eingestellt ist und dafür die CorrelId der Anforderung verwendet wird.

Java EE-Ressourcen

Wenn ein WebSphere MQ-Binding in einer Java EE-Umgebung implementiert wird, werden Java EE-Ressourcen erstellt.

Parameter

MQ-Verbindungsfactory
Wird von Clients zum Erstellen einer Verbindung zum WebSphere MQ-Provider verwendet.

Antwortverbindungsfactory
Wird von der SCA-MQ-Laufzeit verwendet, wenn sich das Sendeziel in einem anderen Queue Manager als das Empfangsziel befindet.

Aktivierungsspezifikation
Eine MQ-JMS-Aktivierungsspezifikation ist mindestens einer Message-driven Bean zugeordnet und stellt die Konfiguration bereit, die zum Empfangen von Nachrichten erforderlich ist.

Ziele

- Sendeziel: Das Ziel, an das die Anforderung oder Nachricht gesendet wird; das Ziel, an das die Antwortnachricht gesendet (exportiert) wird, falls diese nicht durch das MQMD-Headerfeld ReplyTo in der eingehenden Nachricht außer Kraft gesetzt wird.
- Empfangsziel: Das Ziel, an das die Anforderung bzw. Antwort oder eingehende Nachricht gesendet werden soll.

WebSphere MQ-Header:

WebSphere MQ-Header berücksichtigen bestimmte Konventionen für die Konvertierung in SCA-Nachrichten.

WebSphere MQ-Nachrichten bestehen aus einem Systemheader (MQMD), aus null oder mehr MQ-Headern (Systemheader oder benutzerdefinierte Header) sowie einem Nachrichtenhauptteil. Falls die Nachricht mehrere Nachrichtenheader enthält, ist die Reihenfolge der Header von Bedeutung.

Jeder Header enthält Informationen, die die Struktur des nachfolgenden Headers beschreiben. Der Header 'MQMD' beschreibt den ersten Header.

Syntaxanalyse bei MQ-Headern

Zur Syntaxanalyse von MQ-Headern wird eine MQ-Headerdatenbindung verwendet. Die folgenden Header werden automatisch unterstützt:

- MQRFH
- MQRFH2
- MQCIH
- MQIIH

Header, die mit den Zeichen **MQH** beginnen, werden anders gehandhabt. Bestimmte Felder des Headers werden nicht syntaktisch analysiert, sondern behalten das Format von nicht analysierten Byte bei.

Bei anderen MQ-Headern können Sie benutzerdefinierte MQ-Headerdatenbindungen schreiben, um diese Header syntaktisch zu analysieren.

Zugriff auf MQ-Header

Für den Zugriff auf MQ-Header gibt es im Produkt zwei mögliche Verfahren:
- Verwendung des Servicenachrichtenobjekts in einer Mediation
- Verwendung der API 'ContextService'

MQ-Header werden intern mit dem Element 'MQHeader' des Servicenachrichtenobjekts dargestellt. Das Element 'MQHeader' ist ein Container für Headerdaten, der 'MQControl' erweitert, jedoch ein Wertelement mit dem Typ 'anyType' enthält. Er enthält den Header 'MQMD', das Element 'MQControl' (Steuerinformationen für den MQ-Nachrichtenhauptteil) und eine Liste weiterer MQ-Header.
- Der Header 'MQMD' stellt den Inhalt der WebSphere MQ-Nachrichtenbeschreibung dar. Ausgenommen sind Informationen, die die Struktur und die Codierung des Hauptteils bestimmen.
- Das Element 'MQControl' enthält Informationen, die die Struktur und die Codierung eines Nachrichtenhauptteils bestimmen.
- Das Element 'MQHeaders' enthält eine Liste von Objekten des Typs 'MQHeader'.

Die MQ-Headerkette ist nicht verbunden. Daher enthält jeder MQ-Header im Servicenachrichtenobjekt seine eigenen Steuerinformationen (ID des codierten Zeichensatzes, Codierung und Format). Header können ohne großen Aufwand hinzugefügt oder gelöscht werden, ohne andere Headerdaten zu ändern.

Felder im Header 'MQMD' festlegen

Sie können den Header 'MQMD' unter Verwendung der Kontext-API oder über das Servicenachrichtenobjekt in einer Mediation aktualisieren. Die folgenden Felder werden automatisch an die abgehende MQ-Nachricht weitergegeben:
- Encoding
- CodedCharacterSet
- Format
- Report
- Expiry
- Feedback
- Priority
- Persistence
- CorrelId
- MsgFlags

Konfigurieren Sie die MQ-Bindung für einen Import oder Export, um die folgenden Eigenschaften in der abgehenden MQ-Nachricht weiterzugeben:

`MsgID`
Legen Sie für die **Anforderungsnachrichten-ID** die Einstellung für das `Kopieren aus der SCA-Nachricht` fest.

`MsgType`
Wählen Sie das Markierungsfeld für das **Festlegen des Nachrichtentyps auf MQMT_DATAGRAM oder MQMT_REQUEST für Anforderungs-/Antwortoperation** ab.

ReplyToQ
Wählen Sie das Markierungsfeld für das **Überschreiben der Empfangswarteschlange für Antworten auf die Anforderungsnachricht** ab.

ReplyToQMgr
Wählen Sie das Markierungsfeld für das **Überschreiben der Empfangswarteschlange für Antworten auf die Anforderungsnachricht** ab.

Ab Version 7.0 können Kontextfelder mithilfe einer benutzerdefinierten Eigenschaft für die JNDI-Zieldefinition überschrieben werden. Legen Sie die benutzerdefinierte Eigenschaft MDCTX mit dem Wert `SET_IDENTITY_CONTEXT` für das Sendeziel fest, damit die folgenden Felder an die abgehende MQ-Nachricht weitergegeben werden:

- UserIdentifier
- AppIdentityData

Legen Sie die benutzerdefinierte Eigenschaft MDCTX mit dem Wert `SET_ALL_CONTEXT` für das Sendeziel fest, damit die folgenden Eigenschaften an die abgehende MQ-Nachricht weitergegeben werden:

- UserIdentifier
- AppIdentityData
- PutApplType
- PutApplName
- ApplOriginData

Einige Felder werden nicht an die abgehende MQ-Nachricht weitergegeben. Die folgenden Felder werden beim Senden der Nachricht überschrieben:

- BackoutCount
- AccountingToken
- PutDate
- PutTime
- Offset
- OriginalLength

MQCIH statisch in einer WebSphere MQ-Bindung hinzufügen:

IBM Business Process Manager unterstützt das statische Hinzufügen von MQCIH-Headerinformationen ohne Verwendung eines Mediationsmoduls.

Es gibt mehrere Möglichkeiten, MQCIH-Headerinformationen zu einer Nachricht hinzuzufügen (z. B. durch Verwendung des Mediationsbasiselements 'Header-Setter'). Es könnte zweckmäßig sein, diese Headerinformationen statisch, ohne Verwendung eines zusätzlichen Mediationsmoduls, hinzuzufügen. Statische Headerinformationen, z. B. der CICS-Programmname, die Transaktions-ID und weitere Details zu Datenformatheadern, können im Rahmen der WebSphere MQ-Bindung definiert und hinzugefügt werden.

WebSphere MQ, MQ CICS Bridge und CICS müssen für das statische Hinzufügen von MQCIH-Headerinformationen konfiguriert werden.

Sie können Integration Designer verwenden, um den WebSphere MQ-Import mit den statischen Werten zu konfigurieren, die für die MQCIH-Headerinformationen erforderlich sind.

Wenn eine Nachricht empfangen und vom WebSphere MQ-Import verarbeitet wird, wird überprüft, ob die MQCIH-Headerinformationen in der Nachricht bereits vorhanden sind. Wenn der MQCIH vorhanden ist, werden die dynamischen Werte in der Nachricht mithilfe der im WebSphere MQ-Import definierten

statischen Werte überschrieben. Wenn der MQCIH nicht vorhanden ist, wird er in der Nachricht erstellt und die im WebSphere MQ-Import definierten statischen Werte werden hinzugefügt.

Die im WebSphere MQ-Import definierten statischen Werte beziehen sich auf eine bestimmte Methode. Für verschiedene Methoden innerhalb eines WebSphere MQ-Imports können unterschiedliche statische MQCIH-Werte angegeben werden.

Über diese Funktion werden keine Standardwerte bereitgestellt, wenn der MQCIH keine spezifischen Headerinformationen enthält, da ein im WebSphere MQ-Import definierter statischer Wert den entsprechenden Wert in der eingehenden Nachricht überschreibt.

Externe Clients:

IBM Business Process Manager kann mit WebSphere MQ-Bindungen Nachrichten an externe Clients senden bzw. von diesen empfangen.

Ein externer Client (z. B. ein Webportal oder ein unternehmensweites Informationssystem) kann eine Nachricht an eine SCA-Komponente in der Anwendung durch einen Export senden oder von einer SCA-Komponente in der Anwendung durch einen Import aufgerufen werden.

Die WebSphere MQ-Exportbindung implementiert nachrichtengesteuerte Beans (Message driven bean - MDB), um Anforderungen zu überwachen, die bei dem in der Exportbindung definierten Empfangsziel eingehen. Das im Feld **send** angegebene Ziel wird verwendet, um die Antwort auf die eingehende Anforderung zu senden, falls die aufgerufene Anwendung eine Antwort bereitstellt. Ein externer Client kann somit Anwendungen über die Exportbindung aufrufen.

WebSphere MQ-Importe stellen eine Bindung zu externen Clients her und können Nachrichten an diese Clients übermitteln. Diese Nachrichten machen möglicherweise, jedoch nicht zwangsläufig, eine Antwort an den externen Client erforderlich.

Weitere Informationen zur Interaktion mit externen Clients unter Verwendung von WebSphere MQ finden Sie im Information Center von WebSphere MQ.

Fehlerbehebung für WebSphere MQ-Bindungen:

Sie können Fehler und Fehlerbedingungen, die in Verbindung mit WebSphere MQ-Bindungen auftreten, diagnostizieren und beheben.

Primäre Fehlerbedingungen

Die primären Fehlerbedingungen von WebSphere MQ-Bindungen werden durch Transaktionssemantik, durch die WebSphere MQ-Konfiguration oder durch Verweise auf bestehendes Verhalten in anderen Komponenten bestimmt. Zu den primären Fehlerbedingungen zählen die Folgenden:

- Die Herstellung der Verbindung zum Warteschlangenmanager oder zur Warteschlange von WebSphere MQ schlägt fehl.

  Das Fehlschlagen der Verbindungsherstellung zum WebSphere MQ für den Empfang von Nachrichten hat zur Folge, dass der MDB-Listener-Port nicht gestartet werden kann. Dieser Zustand wird im WebSphere Application Server-Protokoll aufgezeichnet. Persistente Nachrichten verbleiben so lange in der WebSphere MQ-Warteschlange, bis sie erfolgreich abgerufen wurden (oder gemäß WebSphere MQ abgelaufen sind).

  Das Fehlschlagen der Verbindungsherstellung zu WebSphere MQ für den Versand abgehender Nachrichten bewirkt ein Rollback der Transaktion, die die Sendeaktion steuert.
- Die Ausführung der Syntaxanalyse für eine eingehende Nachricht oder die Erstellung einer abgehenden Nachricht ist fehlgeschlagen.

  Ein Fehler in der Datenbindung bewirkt ein Rollback der Transaktion, die die Arbeit steuert.

- Das Senden der abgehenden Nachricht ist fehlgeschlagen.

  Das Fehlschlagen des Versands einer Nachricht bewirkt ein Rollback der relevanten Transaktion.
- Mehrere oder nicht erwartete Antwortnachrichten.

  Der Import erwartet für jede Anforderungsnachricht jeweils nur eine Antwortnachricht. Wenn mehr als eine Antwort eingeht oder wenn eine verspätete Antwort eingeht (d. h. eine Antwort, für die die definierte SCA-Antwortablauffrist verstrichen ist), so wird eine Laufzeitausnahmebedingung für den Service ausgelöst. Für die Transaktion wird ein Rollback ausgeführt und die Antwortnachricht wird aus der Warteschlange entfernt oder vom Failed Event Manager verarbeitet.

Szenario falscher Verwendungen: Vergleich mit WebSphere MQ-JMS-Bindungen

Import und Export von WebSphere MQ wurden in erster Linie entwickelt, um mit nativen WebSphere MQ-Anwendungen zu interagieren und den gesamten Inhalt des WebSphere MQ-Nachrichtenhauptteils gegenüber Mediationen verfügbar zu machen. Die WebSphere MQ-JMS-Bindung wurde jedoch dazu konzipiert, mit JMS-Anwendungen zu interagieren, die auf WebSphere MQ implementiert wurden, was eine Offenlegung von Nachrichten gemäß dem JMS-Nachrichtenmodell bewirkt.

Die folgenden Szenarien sollten unter Verwendung der WebSphere MQ-JMS-Bindung erstellt werden und nicht mit der WebSphere MQ-Bindung:
- Aufrufen einer JMS-Message-driven Bean (MDB) von einem SCA-Modul, wenn die MDB auf dem WebSphere MQ-JMS-Provider implementiert ist. Verwenden Sie in diesem Fall einen WebSphere MQ-JMS-Import.
- Zulassen des Aufrufs des SCA-Moduls von einem Java EE-Komponentenservlet oder einer EJB anhand eines JMS. Verwenden Sie in diesem Fall einen WebSphere MQ-JMS-Export.
- Mediation des Inhalts einer JMS-Zuordnungsnachricht beim Durchqueren von WebSphere MQ betreiben. Verwenden Sie einen WebSphere MQ-JMS-Export und -Import in Verbindung mit der entsprechenden Datenbindung.

In gibt Fälle, in denen eine Interaktion zwischen WebSphere MQ-Bindung und WebSphere MQ-JMS-Bindung gegebenenfalls erwartet wird. Besonders bei Überbrückungen zwischen Java EE- und Nicht-Java-EE-Anwendungen von WebSphere MQ sollten Sie einen WebSphere MQ-Export und einen WebSphere MQ-JMS-Import (oder umgekehrt) in Verbindung mit den geeigneten Datenbindungen oder Mediationsmodulen (oder beiden) verwenden.

Nicht zugestellte Nachrichten

Wenn WebSphere MQ eine Nachricht ihrem Ziel nicht zustellen kann (beispielsweise wegen Konfigurationsfehlern), sendet es diese Nachrichten stattdessen an eine nominierte Warteschlange für nicht zustellbare Nachrichten.

Dabei wird dem Anfang des Nachrichtenhauptteils ein Header für nicht zustellbare Nachrichten hinzugefügt. Neben anderen Informationen enthält dieser Header die Gründe für das Fehlschlagen und das ursprüngliche Ziel.

MQ-basierte SCA-Nachrichten werden im Failed Event Manager nicht angezeigt

Wenn SCA-Nachrichten ihren Ursprung in einem Fehler oder einer Störung in der WebSphere MQ-Interaktion haben, würden Sie erwarten, diese Nachrichten im Failed Event Manager vorzufinden. Werden diese Nachrichten nicht im Failed Event Manager angezeigt, stellen Sie sicher, dass für das zugrunde liegende WebSphere MQ-Ziel ein Wert größer als 1 für die maximale Anzahl fehlgeschlagener Zustellungen definiert ist. Durch Festlegen von 2 oder höher für diesen Wert wird eine Interaktion mit dem Failed Event Manager während SCA-Aufrufen für die WebSphere MQ-Bindungen ermöglicht.

In MQ fehlgeschlagene Ereignisse werden im falschen Warteschlangenmanager wiedergegeben

Wenn eine vordefinierte Verbindungsfactory für abgehende Verbindungen verwendet werden soll, müssen die Verbindungseigenschaften mit denen übereinstimmen, die in der für abgehende Verbindungen verwendeten Aktivierungsspezifikation definiert sind.

Anhand der vordefinierten Verbindungsfactory wird bei der Wiedergabe eines fehlgeschlagenen Ereignisses eine Verbindung erstellt. Aus diesem Grund muss sie so konfiguriert sein, dass derselbe Warteschlangenmanager verwendet wird, von dem die Nachricht ursprünglich empfangen wurde.

Ausnahmebedingungen verarbeiten:

Die Art und Weise, in der die Bindung konfiguriert ist, bestimmt, wie Ausnahmebedingungen verarbeitet werden, die von Datenhandlern oder Datenbindungen ausgelöst werden. Außerdem gibt die Spezifik des Mediationsablaufs das Verhalten des Systems vor, wenn eine solche Ausnahmebedingung ausgelöst wird.

Wenn ein Datenhandler oder eine Datenbindung durch eine Bindung aufgerufen wird, können verschiedene Probleme auftreten. Beispielsweise kann es sein, dass ein Datenhandler eine Nachricht mit beschädigten Nutzdaten empfängt oder er versucht, eine Nachricht mit einem falschen Format zu lesen.

Das Verfahren, mit dem die Bindung eine solche Ausnahmebedingung behandelt, wird dadurch bestimmt, wie Sie den Datenhandler oder die Datenbindung implementieren. Das empfohlene Verhalten besteht darin, die Datenbindung so zu konfigurieren, dass sie eine Ausnahmebedingung des Typs **DataBindingException** auslöst.

Bei einem Datenhandler ist die Situation ähnlich. Da der Datenhandler durch die Datenbindung aufgerufen wird, wird jede Ausnahmebedingung des Datenhandlers in eine Ausnahmebedingung der Datenbindung eingeschlossen. Daher wird eine Ausnahmebindung des Typs **DataHandlerException** als Typ **DataBindingException** gemeldet.

Wenn eine Laufzeitausnahmebedingung (inklusive **DataBindingException**) ausgelöst wird, geschieht Folgendes:

- Falls der Mediationsablauf transaktionsorientiert konfiguriert ist, wird die JMS-Nachricht standardmäßig in Failed Event Manager gespeichert, damit sie manuell wiedergegeben oder gelöscht werden kann.

  Anmerkung: Sie können den Fehlerbehebungsmodus für die Bindung so ändern, dass die Nachricht zurückgesetzt und nicht in failed event manager gespeichert wird.
- Ist der Mediationsablauf nicht transaktionsorientiert, wird die Ausnahmebedingung protokolliert und die Nachricht ist nicht mehr vorhanden.

Bei einem Datenhandler ist die Situation ähnlich. Da der Datenhandler durch die Datenbindung aufgerufen wird, wird jede Ausnahmebedingung des Datenhandlers in eine Ausnahmebedingung der Datenbindung eingeschlossen. Daher wird eine Ausnahmebindung des Typs **DataHandlerException** als Typ **DataBindingException** gemeldet.

Einschränkungen bei Bindungen:

Für die Verwendung von Bindungen gelten einige Einschränkungen, die in diesem Thema aufgeführt sind.

Einschränkungen für MQ-Bindungen:

Für die Verwendung von MQ-Bindungen gelten einige Einschränkungen, die in diesem Thema aufgeführt sind.

Keine Nachrichtenverteilung mit Publish/Subscribe-Verfahren

Das Publish/Subscribe-Verfahren für die Verteilung von Nachrichten wird von der MQ-Bindung gegenwärtig nicht unterstützt, obwohl WebSphere MQ selbst das Publish/Subscribe-Verfahren unterstützt. Von der MQ-JMS-Bindung wird dieses Verteilungsverfahren jedoch unterstützt.

Gemeinsam genutzte Empfangswarteschlangen

Mehrere WebSphere MQ-Exportbindungen und Importbindungen erwarten, dass alle Nachrichten in ihrer konfigurierten Empfangswarteschlange für diesen Export bzw. Import bestimmt sind. Beim Konfigurieren von Import- und Exportbindungen sollten die folgenden Aspekte berücksichtigt werden:

- Jeder MQ-Import muss eine eigene separate Empfangswarteschlange besitzen, da die MQ-Importbindung davon ausgeht, dass alle Nachrichten in der Empfangswarteschlange Antworten auf die von ihr gesendeten Anforderungen sind. Wird die Empfangswarteschlange von mehreren Importen gemeinsam genutzt, könnten Antworten durch den falschen Import empfangen werden, was dazu führt, dass die Korrelation mit der ursprünglichen Anforderungsnachricht fehlschlägt.
- Jeder MQ-Export sollte eine eigene separate Empfangswarteschlange besitzen, da andernfalls nicht vorhergesagt werden kann, welcher Export eine bestimmte Anforderungsnachricht empfängt.
- MQ-Importe und -Exporte können auf dieselbe Sendewarteschlange verweisen.

Einschränkungen für JMS-, MQ-JMS- und generische JMS-Bindungen:

Die JMS- und MQ-JMS-Bindungen unterliegen einigen Einschränkungen.

Auswirkungen der Generierung von Standardbindungen

In den folgenden Abschnitten sind die Einschränkungen für die Verwendung von JMS-, MQ-JMS- und generischen JMS-Bindungen beschrieben:

- Auswirkungen der Generierung von Standardbindungen
- Antwortkorrelationsschema
- Bidirektionale Unterstützung

Wenn Sie eine Bindung generieren, werden mehrere Felder automatisch mit Standardwerten gefüllt, falls Sie die Werte nicht selbst eingeben. Beispielsweise wird automatisch ein Name für die Verbindungsfactory erstellt. Falls Sie wissen, dass Sie Ihre Anwendung auf einen Server stellen und mit einem Client über Fernzugriff auf sie zugreifen werden, sollten Sie bei der Bindungserstellung JNDI-Namen eingeben, statt die Standardwerte zu übernehmen, da diese Werte wahrscheinlich zur Ausführungszeit über die Administrationskonsole gesteuert werden müssen.

Wenn Sie jedoch die Standardwerte akzeptiert haben und später feststellen, dass Sie nicht von einem fernen Client aus auf Ihre Anwendung zugreifen können, können Sie den Wert für die Verbindungsfactory über die Administrationskonsole explizit festlegen. Suchen Sie in den Einstellungen für die Verbindungsfactory nach dem Feld für die Providerendpunkte und fügen Sie einen Wert wie '<serverhostname>:7276' hinzu (falls die Standardportnummer verwendet wird).

Antwortkorrelationsschema

Wenn Sie das Antwortkorrelationsschema 'CorrelationId To CorrelationId' verwenden, mit dem Nachrichten in einer Anforderungs-/Antwortoperation korreliert wurden, muss die Nachricht eine dynamische Korrelations-ID enthalten.

Wenn Sie in einem Mediationsmodul mithilfe des Mediationsablaufeditors eine dynamische Korrelations-ID erstellen möchten, fügen Sie vor dem Import mit der JMS-Bindung ein Mediationsbasiselement für Zuordnung hinzu. Öffnen Sie den Zuordnungseditor. In der Zielnachricht sind die bekannten SCA-Header

verfügbar. Ziehen Sie aus der Quellennachricht ein Feld, in dem eine eindeutige ID enthalten ist, in die Korrelations-ID im JMS-Header in der Zielnachricht.

Bidirektionale Unterstützung

Für JNDI-Namen (JNDI = Java Naming and Directory Interface) werden zur Laufzeit nur ASCII-Zeichen unterstützt.

Gemeinsam genutzte Empfangswarteschlangen

Mehrere Export- und Importbindungen erwarten, dass alle Nachrichten in ihrer konfigurierten Empfangswarteschlange für diesen Export bzw. Import bestimmt sind. Beim Konfigurieren von Import- und Exportbindungen sollten die folgenden Aspekte berücksichtigt werden:

- Jede Importbindung muss eine eigene separate Empfangswarteschlange besitzen, da die Importbindung davon ausgeht, dass alle Nachrichten in der Empfangswarteschlange Antworten auf die von ihr gesendeten Anforderungen sind. Wird die Empfangswarteschlange von mehreren Importen gemeinsam genutzt, könnten Antworten durch den falschen Import empfangen werden, was dazu führt, dass die Korrelation mit der ursprünglichen Anforderungsnachricht fehlschlägt.
- Jeder Export sollte eine eigene separate Empfangswarteschlange besitzen, da andernfalls nicht vorhergesagt werden kann, welcher Export eine bestimmte Anforderungsnachricht empfängt.
- Importe und Exporte können auf dieselbe Sendewarteschlange verweisen.

Geschäftsobjekte

Die Softwarebranche hat verschiedene Programmiermodelle und -frameworks entwickelt, bei denen *Geschäftsobjekte* eine einfach zu handhabende Darstellung der Geschäftsdaten für die Anwendungsverarbeitung liefern.

Für diese Geschäftsobjekte gilt im Wesentlichen Folgendes:

- Sie werden über Branchenstandards definiert.
- Sie ordnen Daten transparent zu Datenbanktabellen bzw. zu unternehmensweiten Informationssystemen zu.
- Sie unterstützen ferne Aufrufprotokolle.
- Sie liefern die Basis für die Datenprogrammiermodelle, die für die Anwendungsprogrammierung benötigt werden.

Process Designer und Integration Designer stellen Entwicklern ein auf diesen Merkmalen basierendes allgemeines Geschäftsobjektmodell bereit, das für die Darstellung von Geschäftsentitäten unterschiedlicher Arten aus verschiedenen Domänen ausgelegt ist.

In Process Designer sind die Geschäftsobjekte auf eine Datentypendarstellung ausgerichtet. Basis-Geschäftsobjekte (Variablentypen) werden in System-Toolkits bereitgestellt. Sie können auch angepasste Variablentypen erstellen, die dann als "angepasste Geschäftsobjekte" bezeichnet werden. Weitere Informationen hierzu finden Sie in Geschäftsobjekte und Variablen.

In Integration Designer, das nur zusammen mit IBM Business Process Manager Advanced erhältlich ist, können Geschäftsobjekte auch komplexere XSD-Konstrukte darstellen. In Integration Designer weisen die Geschäftsobjekte eine enge Affinität zu XML-Schemata auf. Informationen dazu, was bei der Integration von Geschäftsobjekten zu beachten ist, die in Integration Designer mit Geschäftsobjekten definiert sind, welche in Process Designer definiert sind, finden Sie in Spiegeln der Bibliothek und XML-Konstrukte nicht unterstützt.

Zur Entwicklungszeit in Integration Designer ermöglicht es das Geschäftsobjektmodell den Entwicklern, Geschäftsobjekte als XML-Schemadefinitionen zu definieren. Zur Laufzeit werden die von den XML-Schemadefinitionen definierten Geschäftsdaten als Java-Geschäftsobjekte dargestellt. Bei diesem Modell sind

Geschäftsobjekte flexibel an in einem frühen Stadium erstellten Entwürfen der SDO-Spezifikation (SDO = Service Data Object) ausgerichtet und stellen die gesamte Palette der Anwendungsschnittstellen für Programmiermodelle zur Verfügung, die zum Bearbeiten von Geschäftsdaten erforderlich sind. Die nachfolgenden Unterabschnitte geben weitere Beschreibungen, wie Geschäftsobjekte mit Integration Designer verwendet werden können.

Geschäftsobjekte definieren

Zum Definieren von Geschäftsobjekten verwenden Sie den Geschäftsobjekteditor in Integration Designer. Der Geschäftsobjekteditor speichert die Geschäftsobjekte als XML-Schemadefinitionen.

Die Verwendung des XML-Schemas zum Definieren von Geschäftsobjekten bietet mehrere Vorteile:

- Das XML-Schema bietet ein standardisiertes Datendefinitionsmodell und eine Grundlage für die Interoperabilität zwischen unterschiedlich und voneinander unabhängigen heterogenen Systemen und Anwendungen. XML-Schemas werden zusammen mit WSDL (Web Services Description Language) verwendet, um standardisierte Schnittstellenverträge zwischen Komponenten, Anwendungen und Systemen bereitzustellen.
- XML-Schemas definieren ein funktionsreiches Datendefinitionsmodell für die Darstellung von Geschäftsdaten. Dieses Modell schließt - neben anderen Funktionen - komplexe Typen, einfache Typen, benutzerdefinierte Datentypen, die Typübernahme und die Kardinalität ein.
- Geschäftsobjekte können durch Geschäftsschnittstellen und -daten definiert werden, die in WSDL (Web Services Description Language) definiert sind, sowie durch XML-Schemas von Industriestandardorganisationen oder von anderen Systemen und Anwendungen. Integration Designer kann diese Geschäftsobjekte direkt importieren.

Integration Designer unterstützt außerdem die Erkennung von Geschäftsdaten in Datenbanken und unternehmensweiten Informationssystemen und die anschließende Generierung der Geschäftsobjektdefinition als standardisiertem XML-Schema aus diesen Geschäftsdaten. Auf diese Weise generierte Geschäftsobjekte werden häufig als *anwendungsspezifische Geschäftsobjekte* bezeichnet, weil sie die Struktur der Geschäftsdaten abbilden, die im unternehmensweiten Informationssystem definiert ist.

Wenn ein Prozess Daten aus vielen verschiedenen Informationssystemen bearbeitet, kann es von Nutzen sein, die unterschiedlichen Darstellungen der Geschäftsdaten (z. B. 'CustomerEIS1' und 'CustomerEIS2' oder 'OrderEIS1' und 'OrderEIS2') in eine einzige kanonische Darstellung (z. B. 'Customer' oder 'Order') zu transformieren. Die kanonische Darstellung wird häufig als *generisches Geschäftsobjekt* bezeichnet.

Geschäftsobjektdefinitionen werden, insbesondere für generische Geschäftsobjekte, häufig von mehr als einer Anwendung verwendet. Zur Unterstützung dieser Wiederverwendung ermöglicht Integration Designer die Erstellung von Geschäftsobjekten in Bibliotheken, die anschließend mehreren Anwendungsmodulen zugeordnet werden können.

Anhand der Web Services Description Language (WSDL) werden die Verträge für die Services definiert, die über ein SCA-Anwendungsmodul bereitgestellt und verwendet werden, sowie die Verträge, die zum Erstellen der Komponenten in einem Anwendungsmodul verwendet werden. In einem Vertrag kann WSDL sowohl die Operationen als auch die Geschäftsobjekte darstellen (die durch das XML-Schema zur Darstellung der Geschäftsdaten definiert sind).

Mit Geschäftsobjekten arbeiten

SCA (Service Component Architecture - Servicekomponentenarchitektur) stellt das Framework für die Definition eines Anwendungsmoduls, der von ihm bereitgestellten Services, der von ihm verwendeten Services und der Zusammenstellung von Komponenten bereit, die die Geschäftslogik des Anwendungsmoduls zur Verfügung stellen. Geschäftsobjekte spielen in der Anwendung eine wichtige Rolle, denn sie definieren die Geschäftsdaten, mit denen die Service- und Komponentenverträge beschrieben werden, sowie die Geschäftsdaten, die von den Komponenten verarbeitet werden.

Im folgenden Diagramm, in dem ein SCA-Anwendungsmodul dargestellt ist, werden viele der Stellen erkennbar, an denen ein Entwickler mit Geschäftsobjekten arbeitet.

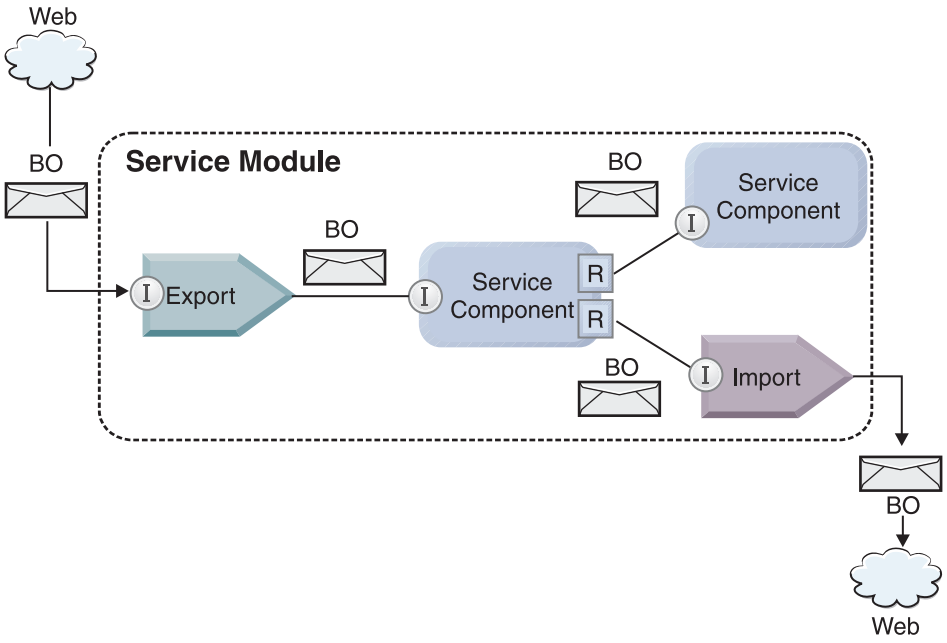


Abbildung 41. Geschäftsobjekte stellen die Daten dar, die in einer Anwendung zwischen Services fließen.

Anmerkung: In diesem Thema ist beschrieben, wie Geschäftsobjekte von SCA-Anwendungsmodulen verwendet werden. Falls Sie Java-Schnittstellen einsetzen, können die SCA-Anwendungsmodule auch Java-Objekte verarbeiten.

Programmiermodell für Geschäftsobjekte

Das Programmiermodell für Geschäftsobjekte besteht aus einer Reihe von Java-Schnittstellen, die Folgendes darstellen:

- Geschäftsobjektdefinition und Instanzdaten
- Gruppe von Services, die die Operationen für die Geschäftsobjekte unterstützen

Definitionen von Geschäftsobjekttypen werden durch die Schnittstellen 'commonj.sdo.Type' und 'commonj.sdo.Property' dargestellt. Das Programmiermodell für Geschäftsobjekte bietet eine Reihe von Regeln für die Zuordnung von XML-Schemainformationen für komplexe Typen zur Typschnittstelle und für die Zuordnung der einzelnen Elemente in der Definition des komplexen Typs zur Eigenschaftenschnittstelle.

Geschäftsobjektinstanzen werden durch die Schnittstelle 'commonj.sdo.DataObject' dargestellt. Das Programmiermodell für Geschäftsobjekte ist nicht typisiert. Dies bedeutet, dass dieselbe Schnittstelle 'commonj.sdo.DataObject' für die Darstellung verschiedener Geschäftsobjektdefinitionen (z. B. 'Customer' und 'Order') verwendet werden kann. Die Definition, welche Eigenschaften in jedem Geschäftsobjekt festgelegt und aus diesem abgerufen werden können, werden durch Typinformationen bestimmt, die in dem jedem Geschäftsobjekt zugeordneten XML-Schema definiert sind.

Das Verhalten des Programmiermodells für Geschäftsobjekte basiert auf der Spezifikation von 'Service Data Object 2.1'. Zusätzliche Informationen enthalten die Spezifikation, die Lernprogramme und Javadocs für 'SDO 2.1 for Java', die Sie im Internet unter der Adresse http://www.oasis-opencsa.org/ finden.

Geschäftsobjektservices unterstützen viele Lebenszyklusoperationen (z. B. Erstellung, Gleichheit, Parsing und Serialisierung) für Geschäftsobjekte.

Spezifikationen zum Programmiermodell für Geschäftsobjekte finden Sie unter Programmieren mit Geschäftsobjektservices und unter Generated API and SPI documentation on business objects.

Bindungen, Datenbindungen und Datenhandler

Wie in Abb. 41 auf Seite 159 gezeigt, werden Geschäftsdaten, mit denen von SCA-Anwendungsmodulen bereitgestellte Services aufgerufen werden, in Geschäftsobjekte transformiert, damit die SCA-Komponenten die Geschäftsdaten bearbeiten können. Die von SCA-Komponenten bearbeiteten Geschäftsobjekte werden analog in das Datenformat konvertiert, das von den externen Services benötigt wird.

In manchen Fällen (beispielsweise bei der Web-Service-Bindung) transformiert die Bindung, die zum Exportieren und Importieren von Services verwendet wird, die Daten automatisch in das geeignete Format. In anderen Fällen (beispielsweise bei der JMS-Bindung) können Entwickler eine Datenbindung oder einen Datenhandler bereitstellen, von der/dem nicht native Format in Geschäftsobjekte transformiert werden, die durch die Datenobjektschnittstelle (DataObject) dargestellt werden.

Weitere Informationen zum Entwickeln von Datenbindungen und Datenhandlern enthalten die Themen „Datenhandler” auf Seite 60 und „Datenbindungen” auf Seite 62.

Komponenten

SCA-Komponenten definieren ihre Bereitstellungs- und Verarbeitungsserviceverträge mit einer Kombination aus WSDL (Web Services Description Language) und XML-Schema. Die von SCA zwischen Komponenten übergebenen Geschäftsdaten werden unter Verwendung der Schnittstelle 'DataObject' als Geschäftsobjekte dargestellt. SCA prüft, ob diese Geschäftsobjekttypen mit dem Schnittstellenvertrag kompatibel sind, der durch die aufzurufende Komponente definiert ist.

Die Programmiermodellabstraktionen für die Bearbeitung von Geschäftsobjekten variieren je nach Komponente. Die POJO-Komponente und das Basiselement 'Angepasst' der Mediationsablaufkomponente ermöglichen eine direkte Bearbeitung der Geschäftsobjekte, da die Java-Programmierung durch die Verwendung der Programmierschnittstelle und Services für Geschäftsobjekte direkt möglich ist. Die meisten Komponenten bieten für die Bearbeitung von Geschäftsobjekten Abstraktionen höherer Ebene, aber auch Java-Code-Snippets, um angepasstes Verhalten in den Schnittstellen und Services von Geschäftsobjekten zu definieren.

Geschäftsobjekte können entweder mit einer Kombination aus den Komponenten für die Schnittstellenablaufmediation und die Geschäftsobjektzuordnung oder aus der Mediationsablaufkomponente und deren Basiselement 'XML-Zuordnung' transformiert werden. Diese Funktionen für die Geschäftsobjekttransformation sind zur Konvertierung von anwendungsspezifischen Geschäftsobjekten in generische Geschäftsobjekte (und umgekehrt) von Nutzen.

Spezielle Geschäftsobjekte

Servicenachrichtenobjekte und Geschäftsgrafiken sind zwei spezielle Typen von Geschäftsobjekten, die für bestimmte Anwendungszwecke eingesetzt werden.

Servicenachrichtenobjekt

Ein Servicenachrichtenobjekt ist ein spezialisiertes Geschäftsobjekt, mit dem in Mediationsablaufkomponenten die Erfassung der Daten dargestellt wird, die einem Serviceaufruf zugeordnet sind.

Ein Servicenachrichtenobjekt hat eine festgelegte Struktur der höchsten Ebene, die aus Headern, Kontext, Hauptteil und Anhängen (sofern vorhanden) besteht.

- Header übertragen Informationen im Zusammenhang mit dem Serviceaufruf über ein bestimmtes Protokoll bzw. eine bestimmte Bindung. Beispiele sind SOAP-Header und JMS-Header.

- Kontextdaten übertragen zusätzliche logische Informationen, die dem Aufruf zugeordnet sind, während dieser von der Mediationsablaufkomponente verarbeitet wird. Diese Informationen gehören normalerweise nicht zu den Anwendungsdaten, die von Clients gesendet oder empfangen werden.
- Der Hauptteil des Servicenachrichtenobjekts überträgt die Geschäftsnutzdaten, die die Kernanwendungsnachrichten- oder Aufrufdaten in Form eines Standardgeschäftsobjekts darstellen.

Das Servicenachrichtenobjekt kann außerdem Anhangsdaten für Web-Service-Aufrufe übertragen, die SOAP mit Anhängen verwenden.

Mediationsabläufe führen solche Tasks als Anforderungsweiterleitung und Datentransformation aus. Das Servicenachrichtenobjekt ermöglicht die kombinierte Sicht von Header- und Nutzdateninhalt in einer einzigen einheitlichen Struktur.

Geschäftsgrafik

Eine Geschäftsgrafik ist ein spezielles Geschäftsobjekt, mit dem die Unterstützung für die Datensynchronisation in Integrationsszenarios bereitgestellt wird.

Dies soll am Beispiel von zwei unternehmensweiten Informationssystemen erläutert werden, die eine Darstellung für einen bestimmten Auftrag enthalten. Wenn der Auftrag in einem System geändert wird, kann an das andere System eine Nachricht gesendet werden, um die Auftragsdaten zu synchronisieren. Geschäftsgrafiken unterstützen ein Konzept, bei dem lediglich der geänderte Teil des Auftrags an das andere System gesendet und mit Informationen zur Änderungsübersicht versehen ist, um den Typ der Änderung zu definieren.

In diesem Beispiel würde eine Auftragsgeschäftsgrafik dem anderen System mitteilen, dass eine der Artikelpositionen im Auftrag gelöscht und die Auftragseigenschaft für das voraussichtliche Lieferdatum aktualisiert wurde.

Geschäftsgrafiken können in Integration Designer ohne großen Aufwand zu Geschäftsobjekten hinzugefügt werden. Sie sind am häufigsten in Szenarios zu finden, in denen WebSphere-Adapter verwendet werden oder die Migration von WebSphere InterChange Server-Anwendungen unterstützt werden muss.

Parsingmodus für Geschäftsobjekte

Integration Designer bietet eine Eigenschaft für Module und Bibliotheken, mit der Sie für Geschäftsobjekte den XML-Parsingmodus 'Vollständig' oder 'Bedarfsorientiert' definieren können.

- Falls die Option auf *Vollständig* gesetzt ist, werden XML-Byteströme zur Erstellung des Geschäftsobjekts vollständig syntaktisch analysiert.
- Ist die Option auf *Bedarfsorientiert* gesetzt, wird das Geschäftsobjekt normal erstellt, das eigentliche Parsing des XML-Bytestroms wird jedoch verzögert und der Strom wird nur partiell analysiert, wenn auf die Eigenschaften des Geschäftsobjekts zugegriffen wird.

Daten, die keine XML-Daten sind, werden in beiden XML-Parsingmodi zur Erstellung des Geschäftsobjekts vollständig analysiert.

Hinweise zur Auswahl des Parsingmodus für Geschäftsobjekte:

Der Parsingmodus für Geschäftsobjekte legt fest, wie XML-Daten während der Laufzeit syntaktisch analysiert werden. Ein Parsingmodus für Geschäftsobjekte wird für ein Modul oder eine Bibliothek bei der Erstellung dieses Moduls bzw. dieser Bibliothek definiert. Sie können den Parsingmodus des Moduls oder der Bibliothek ändern. Jedoch sind hierbei gewisse Auswirkungen zu beachten.

Der Parsingmodus für ein Geschäftsobjekt wird auf der Modul- und Bibliotheksebene festgelegt. Module, die in einer Version von IBM Integration Designer vor Version 7 erstellt wurden, werden im vollständigen Parsingmodus ausgeführt, ohne dass Änderungen erforderlich sind. Standardmäßig werden Module und

Bibliotheken, die in IBM Integration Designer ab Version 7 erstellt werden, mit dem geeignetsten Parsingmodus abhängig von einer Reihe von Faktoren definiert. Solche Faktoren sind zum Beispiel der Parsingmodus der vorhandenen Projekte in Ihrem Arbeitsbereich oder der Parsingmodus abhängiger Projekte oder anderer Projekte in derselben Lösung usw. Sie können den Parsingmodus für Geschäftsobjekte eines Moduls oder einer Bibliothek an Ihre Implementierung anpassen, wobei jedoch die folgenden Punkte zu beachten sind.

Hinweise

- Der bedarfsorientierte Parsingmodus für Geschäftsobjekte verarbeitet XML-Daten schneller. Allerdings bestehen Kompatibilitätsunterschiede zwischen dem vollständigen Parsingmodus und dem bedarfsorientierten Modus, die zu beachten sind, bevor die Konfiguration eines Moduls oder einer Bibliothek geändert wird. Diese Unterschiede haben Auswirkungen auf das Laufzeitverhalten der Module. Informationen dazu, welcher Parsingmodus sich für Ihre Anwendung optimal eignet, finden Sie im Abschnitt über die Vorteile des bedarfsorientierten Parsingmodus im Vergleich zum vollständigen Parsingmodus in den zugehörigen Links.
- Ein Modul kann nur zur Ausführung in einem Parsingmodus konfiguriert werden. Bibliotheken können zur Unterstützung eines der beiden Parsingmodi oder zur Unterstützung beider Parsingmodi konfiguriert werden. Eine Bibliothek, die zur Unterstützung beider Parsingmodi konfiguriert ist, kann sowohl von einem Modul mit dem vollständigen Parsingmodus als auch von einem Modul mit dem bedarfsorientierten Parsingmodus referenziert werden. Der Parsingmodus einer Bibliothek zur Laufzeit wird durch die Module festgelegt, die die Bibliothek referenzieren. Zur Laufzeit deklariert ein Modul seinen Parsingmodus und dieser Parsingmodus wird von dem Modul und allen Bibliotheken verwendet, auf die das Modul zugreift.
- Module und Bibliotheken, die für verschiedene Parsingmodi konfiguriert sind, sind in den folgenden Fällen kompatibel:
  - Module und Bibliotheken, die mit dem bedarfsorientierten Parsingmodus konfiguriert sind, sind mit Bibliotheken kompatibel, die entweder den bedarfsorientierten Parsingmodus verwenden oder sowohl den vollständigen als auch den bedarfsorientierten Parsingmodus verwenden.
  - Module und Bibliotheken, die mit dem vollständigen Parsingmodus konfiguriert sind, sind mit Bibliotheken kompatibel, die entweder den vollständigen Parsingmodus verwenden oder sowohl den vollständigen als auch den bedarfsorientierten Parsingmodus verwenden.
  - Bibliotheken, die mit dem bedarfsorientierten und dem vollständigen Parsingmodus konfiguriert sind, sind nur mit Bibliotheken kompatibel, die sowohl den bedarfsorientierten Parsingmodus als auch den vollständigen Parsingmodus verwenden.
- Verwenden Sie denselben Parsingmodus für interagierende Module, die mithilfe der SCA-Bindung kommunizieren. Wenn Module mit verschiedenen Parsingmodi kommunizieren, kann es zu Leistungsproblemen kommen.

Zugehörige Konzepte:

„Vorteile des Parsingmodus 'Bedarfsorientiert' (Lazy) gegenüber dem Modus 'Vollständig' (Eager)"
Einige Anwendungen profitieren vom XML-Parsingmodus 'Bedarfsorientiert', während sich für andere Anwendungen beim Parsingmodus 'Vollständig' ein besseres Leistungsverhalten ergibt. Es empfiehlt sich, für Ihre Anwendungen einen Vergleichspunkt in beiden Parsingmodi zu ermitteln, um den für die speziellen Merkmale der Anwendung am besten geeigneten Modus festzustellen.

Vorteile des Parsingmodus 'Bedarfsorientiert' (Lazy) gegenüber dem Modus 'Vollständig' (Eager):

Einige Anwendungen profitieren vom XML-Parsingmodus 'Bedarfsorientiert', während sich für andere Anwendungen beim Parsingmodus 'Vollständig' ein besseres Leistungsverhalten ergibt. Es empfiehlt sich, für Ihre Anwendungen einen Vergleichspunkt in beiden Parsingmodi zu ermitteln, um den für die speziellen Merkmale der Anwendung am besten geeigneten Modus festzustellen.

Anwendungen, die umfangreiche XML-Datenströme syntaktisch analysieren (= parsen), bieten bei Verwendung des XML-Parsingmodus 'Bedarfsorientiert' wahrscheinlich eine gesteigerte Leistung. Mit stei-

gender Größe des XML-Bytestroms und mit abnehmendem Volumen der Daten aus dem Bytestrom, auf die durch die Anwendung zugegriffen wird, nimmt die Leistungsverbesserung zu.

Die folgenden Anwendungen zeigen im Parsingmodus 'Vollständig' wahrscheinlich ein besseres Leistungsverhalten:

- Anwendungen, die Datenströme syntaktisch analysieren, die keine XML-Datenströme sind
- Anwendungen, die Nachrichten verwenden, die mit dem Service 'BOFactory' erstellt wurden
- Anwendungen, die sehr kleine XML-Nachrichten syntaktisch analysieren

Zugehörige Verweise:

„Hinweise zur Auswahl des Parsingmodus für Geschäftsobjekte" auf Seite 161
Der Parsingmodus für Geschäftsobjekte legt fest, wie XML-Daten während der Laufzeit syntaktisch analysiert werden. Ein Parsingmodus für Geschäftsobjekte wird für ein Modul oder eine Bibliothek bei der Erstellung dieses Moduls bzw. dieser Bibliothek definiert. Sie können den Parsingmodus des Moduls oder der Bibliothek ändern. Jedoch sind hierbei gewisse Auswirkungen zu beachten.

Hinweise zur Anwendungsmigration und -entwicklung:

Falls Sie eine Anwendung, die ursprünglich für die Verwendung des Parsingmodus 'Vollständig' entwickelt wurde, so konfigurieren, dass künftig der Parsingmodus 'Bedarfsorientiert' verwendet werden soll, oder falls Sie beabsichtigen, bei einer Anwendung den Wechsel zwischen den Modi 'Vollständig' und 'Bedarfsorientiert' zu ermöglichen, müssen Sie die Unterschiede zwischen den Modi sowie die Hinweise für den Moduswechsel berücksichtigen.

Fehlerbehandlung

Falls der syntaktisch analysierte XML-Bytestrom falsch formatiert ist, treten Parsingausnahmebedingungen auf.

- Im XML-Parsingmodus 'Vollständig' treten diese Ausnahmebedingungen auf, sobald das Geschäftsobjekt aus dem eingehenden XML-Datenstrom syntaktisch analysiert wird.
- Ist der XML-Parsingmodus 'Bedarfsorientiert' konfiguriert, treten die Parsingausnahmebedingungen latent auf, wenn der Zugriff auf die Geschäftsobjekteigenschaften erfolgt und der falsch formatierte XML-Teil syntaktisch analysiert wird.

Zur Behandlung von falsch formatierten XML-Daten haben Sie die folgenden Möglichkeiten:

- Enterprise Service Bus für die Auswertung von eingehenden XML-Daten bei der Ersterkennung implementieren
- Logik für die verzögerte Fehlererkennung am Zugriffspunkt für Geschäftsobjekteigenschaften schreiben

Ausnahmebedingungsstacks und -nachrichten

Da den XML-Parsingmodi 'Vollständig' und 'Bedarfsorientiert' verschiedene Implementierungen zugrunde liegen, haben die durch die Programmierschnittstellen für Geschäftsobjekte ausgelösten Stack-Traces und Services denselben Ausnahmebedingungsklassennamen, enthalten jedoch möglicherweise nicht dieselbe Ausnahmebedingungsnachricht oder eingeschlossene Gruppe von implementierungsspezifischen Ausnahmebedingungsklassen.

XML-Serialisierungsformat

Der XML-Parsingmodus 'Bedarfsorientiert' bietet eine Leistungsoptimierung, die versucht, nicht geänderte XML-Datei aus dem eingehenden Bytestrom bei der Serialisierung in den ausgehenden Bytestrom zu kopieren. Dies hat eine Leistungssteigerung zur Folge, aber das Format des abgehenden XML-Bytestroms kann unterschiedlich sein, falls das gesamte Geschäftsobjekt im XML-Parsingmodus 'Bedarfsorientiert' aktualisiert oder im XML-Parsingmodus 'Vollständig' ausgeführt wurde.

Obwohl das XML-Serialisierungsformat unter Umständen syntaktisch nicht gänzlich äquivalent ist, ist der vom Geschäftsobjekt bereitgestellte semantische Wert ungeachtet der Parsingmodi äquivalent. Daher können XML-Daten ohne Weiteres mit semantischer Äquivalenz zwischen Anwendungen übergeben werden, die in unterschiedlichen Parsingmodi ausgeführt werden.

Prüfprogramm für Geschäftsobjektinstanz

Das Prüfprogramm für die Geschäftsobjektinstanz im XML-Parsingmodus 'Bedarfsorientiert' bietet eine Prüfung mit größerer Formattreue für Geschäftsobjekte, insbesondere in Bezug auf die Facettenprüfung bei Eigenschaftswerten. Aufgrund dieser Verbesserungen erfasst das Instanzprüfprogramm für den Parsingmodus 'Bedarfsorientiert' zusätzliche Aspekte, die im Parsingmodus 'Vollständig' nicht berücksichtigt werden, und bietet so genauere Fehlernachrichten.

XML-Zuordnungen in Version 602

Mediationsabläufe, die ursprünglich vor WebSphere Integration Developer Version 6.1 entwickelt wurden, enthalten möglicherweise Basiselemente vom Typ 'Zuordnung' mit einer Zuordnung oder einem Style-Sheet, die bzw. der nicht direkt im XML-Parsingmodus 'Bedarfsorientiert' ausgeführt werden kann. Wenn eine Anwendung für die Verwendung im XML-Parsingmodus 'Bedarfsorientiert' migriert wird, können die Zuordnungsdateien, die zu den Basiselementen vom Typ 'Zuordnung' gehören, vom Migrationsassistenten automatisch so aktualisiert werden, dass sie im neuen Modus ausgeführt werden. Wenn ein Basiselement vom Typ 'Zuordnung' jedoch direkt ein Style-Sheet referenziert, das manuell bearbeitet wurde, wird das Style-Sheet nicht migriert und kann nicht im XML-Parsingmodus 'Bedarfsorientiert' ausgeführt werden.

Private unveröffentlichte APIs

Falls eine Anwendung unveröffentlichte private implementierungsspezifische Programmierschnittstellen für Geschäftsobjekte nutzt, schlägt die Kompilierung der Anwendung bei einem Wechsel des Parsingmodus wahrscheinlich fehl. Beim Parsingmodus 'Vollständig' sind diese privaten Schnittstellen normalerweise Implementierungsklassen für Geschäftsobjekte, die durch EMF (Eclipse Modeling Framework) definiert sind.

In allen Fällen wird dringend empfohlen, private APIs aus der Anwendung zu entfernen.

EMF-APIs für Servicenachrichtenobjekte

Eine Mediationskomponente in IBM Integration Designer bietet die Möglichkeit, Nachrichteninhalte mithilfe der Java-Klassen und -Schnittstellen aus dem Paket 'com.ibm.websphere.sibx.smobo' zu manipulieren. Im XML-Parsingmodus 'Bedarfsorientiert' können weiterhin die Java-Schnittstellen aus dem Paket 'com.ibm.websphere.sibx.smobo' verwendet werden. Methoden, die EMF-Klassen und -Schnittstellen direkt referenzieren oder die von EMF-Schnittstellen geerbt wurden, schlagen jedoch wahrscheinlich fehl.

Das Servicenachrichtenobjekt und sein Inhalt können im XML-Parsingmodus 'Bedarfsorientiert' nicht in EMF-Objekte umgesetzt werden.

Service 'BOMode'

Mit dem Service 'BOMode' wird ermittelt, ob gegenwärtig der XML-Parsingmodus 'Vollständig' oder 'Bedarfsorientiert' ausgeführt wird.

Migration

Alle Anwendungen vor Version 7.0.0.0 werden im XML-Parsingmodus 'Vollständig' ausgeführt. Werden sie unter Verwendung der BPM-Laufzeitmigrationstools einer Laufzeitmigration unterzogen, erfolgt die Ausführung weiterhin im XML-Parsingmodus 'Vollständig'.

Damit eine Anwendung einer älteren Version als Version 7.0.0.0 für die Verwendung des XML-Parsingmodus 'Bedarfsorientiert' konfiguriert werden kann, müssen Sie zunächst die Artefakt der Anwendung mit Integration Designer migrieren. Nach der Migration konfigurieren Sie die Anwendung für das XML-Parsing 'Bedarfsorientiert'.

Das Thema Quellenartefakte migrieren enthält Informationen zum Migrieren von Artefakten in Integration Designer. Unter Parsingmodus des Geschäftsobjekts von Modulen und Bibliotheken konfigurieren finden Sie Angaben über das Festlegen des Parsingmodus.

Beziehungen

Eine Beziehung ist eine Zuordnung zwischen mindestens zwei Datenentitäten, die in der Regel Business-Objekte sind. In IBM Business Process Manager Advanced können Beziehungen zum Transformieren von Daten verwendet werden, die in Geschäftsobjekten oder anderen Daten gleich sind, aber unterschiedlich dargestellt werden; oder sie werden für die Erstellung von Zuordnungen zwischen verschiedenen Objekten in unterschiedlichen Anwendungen verwendet. Sie können über mehrere Anwendungen, Lösungen oder sogar Produkte hinweg gemeinsam genutzt werden.

Der Relationship Service in IBM Business Process Manager Advanced stellt die Infrastruktur und Operationen für die Verwaltung von Beziehungen bereit. Da er die Bearbeitung von Business-Objekten unabhängig von der jeweiligen Position ermöglicht, bietet er eine ganzheitliche Sicht auf alle Anwendungen in einem Unternehmen und dient zudem als Baustein für BPM-Lösungen. Da Beziehungen erweiterbar und einfach zu verwalten sind, können sie in komplexen Integrationslösungen verwendet werden.

Was sind Beziehungen?

Eine Beziehung ist eine Zuordnung zwischen Business-Objekten. Jedes Business-Objekt in einer Beziehung wird als *Teilnehmer* in der Beziehung bezeichnet. Die Teilnehmer in der Beziehung unterscheiden sich durch die Funktion oder *Rolle*, die sie in der Beziehung einnehmen. Eine Beziehung enthält eine Liste von Rollen.

Die *Beziehungsdefinition* beschreibt die einzelnen Rollen und gibt an, wie diese Rollen zueinander in Beziehung stehen. Ferner wird die 'Gesamtform' der Beziehung beschrieben. Es wäre beispielsweise möglich, dass eine Rolle nur einen Teilnehmer, eine andere Rolle dagegen so viele Teilnehmer wie nötig haben kann. Sie könnten eine *Auto-Eigentümer*-Beziehung definieren, in der ein Eigentümer möglicherweise mehrere Autos besitzt. Eine Instanz könnte beispielsweise die folgenden Teilnehmer für jede dieser Rollen aufweisen:

- Auto (Ferrari)
- Eigentümer (John)

Die Beziehungsdefinition ist eine Schablone für die *Beziehungsinstanz*. Die Instanz ist die Laufzeitinstanzierung der Beziehung. Im *Auto-Eigentümer*-Beispiel kann eine Instanz die folgenden Zuordnungen beschreiben:

- John ist Eigentümer von Ferrari.
- Sara ist Eigentümer von Mazda.
- Bob ist Eigentümer von Ferrari.

Bei Verwendung von Beziehungen ist die benutzerdefinierte Erstellung von Persistenz zur Beziehungsverfolgung innerhalb der Geschäftslogik nicht mehr erforderlich. In bestimmten Szenarios übernimmt der Relationship Service alle erforderlichen Arbeitsschritte. Siehe dazu das Beispiel im Abschnitt zu Identitätsbeziehungen.

Szenarios

Im Folgenden ist ein typisches Beispiel für eine Situation aufgeführt, in der eine Integrationslösung Beziehungen verwenden könnte. Ein Großunternehmen kauft mehrere Unternehmen oder Business-Units. Jede Business-Unit verwendet zur Überwachung von Personal und Notebooks unterschiedliche Software. Das Unternehmen sucht nach einer Möglichkeit zur Überwachung der Mitarbeiter und deren Notebooks. An die Lösung werden die folgenden Anforderungen gestellt:

- Alle Mitarbeiter in den verschiedenen Unternehmensbereichen sollen so angezeigt werden, als befänden sie sich in derselben Datenbank.
- Eine Gesamtansicht aller Notebooks soll möglich sein.
- Die Mitarbeiter sollen sich am System anmelden und ein Notebook erwerben können.
- Die verschiedenen Unternehmensanwendungssysteme sollen in die verschiedenen Geschäftsbereiche aufgenommen werden.

Um diese Ziele zu erreichen, muss das Unternehmen beispielsweise sicherstellen, dass John Smith und John A. Smith in verschiedenen Anwendungen als derselbe Mitarbeiter angezeigt wird. Es muss zum Beispiel eine Möglichkeit geben, eine einzelne Entität über den Geltungsbereich mehrerer Anwendungen hinweg zu konsolidieren.

Zu den komplexeren Beziehungsszenarios gehört die Erstellung von BPEL-Prozessen, die Beziehungen zwischen verschiedenen Objekten in mehreren Anwendungen herstellen. Bei komplexen Beziehungsszenarios befinden sich die Business-Objekte in der Integrationslösung und nicht in den Anwendungen. Der Relationship Service stellt eine Plattform zur persistenten Verwaltung von Beziehungen bereit. Vor dem Relationship Service muss jedoch ein eigener Persistenzservice für Objekte erstellt werden. Zwei Beispiele für komplexe Szenarios:

- Sie verfügen über das Business-Objekt **Auto** mit einer VIN-Nummer in einer SAP-Anwendung und möchten den Umstand protokollieren, dass dieses Auto einer anderen Person gehört. Die Eigentumsbeziehung besteht jedoch mit einer Person in einer PeopleSoft-Anwendung. In diesem Beziehungsmuster gibt es zwei Lösungen und Sie müssen eine Brücke zwischen diesen beiden Lösungen bauen.
- Ein großes Einzelhandelsunternehmen möchte die Waren überwachen, die gegen Rückerstattung oder Kredit zurückgegeben werden. Daran sind zwei unterschiedliche Anwendungen beteiligt: ein Auftragsbearbeitungssystem für Einkäufe und ein Retourenbearbeitungssystem für Retouren. Die Business-Objekte befinden sich in mehreren Anwendungen und Sie suchen nach einer Möglichkeit, die Beziehungen zwischen diesen Objekten anzuzeigen.

Allgemeine Verwendungsmuster

Die häufigsten Beziehungsmuster sind *Äquivalenzmuster*. Diese basieren auf der Arbeit mit Querverweisen oder Korrelationen. Es gibt zwei Typen von Beziehungen, die zu diesem Muster passen: *Nicht-Identitätsbeziehungen* und *Identitätsbeziehungen*.

- **Nicht-Identitätsbeziehungen** stellen Beziehungen zwischen Business-Objekten oder anderen Daten auf einer Eins-zu-viele- oder Viele-zu-viele-Basis her. Für jede Beziehungsinstanz können eine oder mehrere Instanzen jedes Teilnehmers existieren. Ein Typ von Nicht-Identitätsbeziehung ist eine statische Referenzbeziehung. Ein Beispiel für diesen Beziehungstyp ist eine Beziehung, bei der sich der Eintrag **CA** in einer SAP-Anwendung auf den Eintrag **California** in einer Siebel-Anwendung bezieht.
- **Identitätsbeziehungen** stellen Beziehungen zwischen Business-Objekten oder anderen Daten auf einer Eins-zu-eins-Basis her. Für jede Beziehungsinstanz kann nur eine Instanz jedes Teilnehmers existieren. Identitätsbeziehungen erfassen Querverweise zwischen Business-Objekten, die semantisch äquivalent sind, aber in verschiedenen Anwendungen unterschiedlich erkannt werden. Jeder Teilnehmer in der Beziehung ist einem Business-Objekt zugeordnet, das über einen Wert (oder eine Kombination von Werten) verfügt, der das Objekt eindeutig identifiziert. Identitätsbeziehungen transformieren normalerweise die Schlüsselattribute von Business-Objekten wie ID-Nummern und Produktcodes.

Wenn beispielsweise Business-Objekte vom Typ **Auto** in SAP-, PeopleSoft- und Siebel-Anwendungen enthalten sind und eine Lösung erstellt werden soll, die diese synchronisiert, müssen Sie normalerweise eine manuelle Beziehungssynchronisationslogik in sechs Zuordnungen einführen:

SAP -> generisch

generisch -> SAP

PeopleSoft -> generisch

generisch -> PeopleSoft

Siebel -> generisch

generisch -> Siebel

Wenn Sie jedoch Beziehungen in Ihrer Lösung verwenden, stellt der Relationship Service vordefinierte Musterimplementierungen bereit, die diese Beziehungsinstanzen für Sie verwalten.

Tools zum Arbeiten mit Beziehungen

Der *Relationship Editor* in Integration Designer ist das Tool zum Modellieren und Entwerfen von Business-Integrationsbeziehungen und Rollen. Detaillierte Hintergrund- und Taskinformationen zur Erstellung von Beziehungen und zur Verwendung des Relationship Editors finden Sie unter Beziehungen erstellen.

Der *Relationship Service* ist ein Infrastrukturservice in IBM Business Process Manager, der Beziehungen und Rollen im System verwaltet und Operationen für die Beziehungs- und Rollenverwaltung bereitstellt.

Der *Relationship Manager* ist die Verwaltungsschnittstelle zur Verwaltung von Beziehungen. Der Zugriff erfolgt über die Relationship Manager-Seiten der Administrationskonsole.

Beziehungen können programmgesteuert über Relationship Service-APIs aufgerufen werden.

Relationship Service

Der Relationship Service speichert Beziehungsdaten in Beziehungstabellen, über die anwendungsspezifische Werte aus verschiedenen Anwendungen oder Lösungen überwacht werden. Der Relationship Service stellt Operationen für die Beziehungs- und Rollenverwaltung bereit.

Funktionsweise von Beziehungen

Beziehungen und Rollen werden über die grafische Schnittstelle des Relationship Editors in Integration Designer definiert. Der Relationship Service speichert die Korrelationsdaten in Tabellen in der Beziehungsdatenbank in der Standarddatenquelle, die Sie beim Konfigurieren des Relationship Service angeben. In einer separaten Tabelle (manchmal auch als Teilnehmertabelle bezeichnet) werden Informationen zu jedem Teilnehmer in der Beziehung gespeichert. Der Relationship Service verwendet diese Beziehungstabellen zur Überwachung der zugehörigen anwendungsspezifischen Werte und zur Weitergabe aktualisierter Informationen an alle Lösungen.

Beziehungen sind Geschäftsartefakte, die innerhalb eines Projekts oder in einer gemeinsam genutzten Bibliothek implementiert werden. Bei der erstmaligen Implementierung füllt der Relationship Service die Daten.

Wenn Zuordnungen oder andere IBM Business Process Manager-Komponenten zur Laufzeit eine Beziehungsinstanz benötigen, werden die Instanzen der Beziehung je nach Szenario entweder aktualisiert oder abgerufen.

Zur Bearbeitung von Beziehungs- und Rolleninstanzdaten gibt es drei Möglichkeiten:

- Java-Snippet-Aufrufe der Relationship Service-APIs der IBM Business Process Manager-Komponente
- Beziehungsumsetzungen im Business-Objektzuordnungsservice von IBM Business Process Manager
- Relationship Manager-Tool

Detaillierte Hintergrund- und Taskinformationen zum Erstellen von Beziehungen, zum Angeben von Beziehungstypen und zur Verwendung des Relationship Editors finden Sie im Abschnitt Beziehungen erstellen.

Relationship Manager

Relationship Manager ist die Verwaltungsschnittstelle zur Verwaltung von Beziehungen. Der Zugriff erfolgt über die Relationship Manager-Seiten der Administrationskonsole.

Relationship Manager enthält eine grafische Benutzerschnittstelle für die Erstellung und Bearbeitung von Beziehungs- und Rollendaten zur Laufzeit. Beziehungsentitäten können auf allen Ebenen verwaltet werden: Beziehungsinstanzen, Rolleninstanzen, Attributdaten und Eigenschaftsdaten. Mit Relationship Manager können die folgenden Aktionen ausgeführt werden:

- Anzeigen einer Liste der Beziehungen im System sowie detaillierter Informationen für einzelne Beziehungen
- Verwaltung von Beziehungsinstanzen:
  - Abfragen von Beziehungsdaten zum Anzeigen von Teilmengen der Instanzdaten
  - Abfragen von Beziehungsdaten zum Anzeigen von Teilmengen der Instanzdaten mithilfe von Datenbanksichten
  - Anzeigen einer Liste von Beziehungsinstanzen, die mit einer einer Beziehungsabfrage übereinstimmen, sowie detaillierter Informationen zu einer Instanz
  - Ändern der Eigenschaftswerte für eine Beziehungsinstanz
  - Erstellen und Löschen von Beziehungsinstanzen
- Verwalten von Rollen und Rolleninstanzen
  - Anzeigen von Details zu einer Rolle oder Rolleninstanz
  - Bearbeiten der Eigenschaften von Rolleninstanzen
  - Erstellen und Löschen von Rolleninstanzen für eine Beziehung
  - Durchführen von Rollback-Operationen für Beziehungsinstanzdaten bis zu einem Zeitpunkt, zu dem die Daten garantiert zuverlässig sind
- Importieren von Daten aus einer vorhandenen statischen Beziehung in das System oder Exportieren von Daten aus einer vorhandenen statischen Beziehung in eine RI- oder CSV-Datei
- Entfernen von Beziehungsschemas und -daten aus dem Repository, wenn die darauf zugreifende Anwendung deinstalliert wird

Beziehungen in Network Deployment-Umgebungen

Beziehungen können ohne zusätzlichen Konfigurationsaufwand in Network Deployment-Umgebungen verwendet werden.

In Network Deployment-Umgebungen werden Beziehungen in einem Anwendungscluster installiert. Beziehungen sind dann innerhalb des Clusters sichtbar, und alle Server im Cluster haben Zugriff auf die Instanzdaten, die in der Beziehungsdatenbank gespeichert sind. Die Möglichkeit der Ausführung des Relationship Service in einer Network Deployment-Umgebung macht diese skalierbar und hoch verfügbar.

Mit Relationship Manager können Beziehungen in verschiedenen Clustern über eine zentrale Verwaltungsschnittstelle verwaltet werden. Die Verbindung zwischen Relationship Manager und einem Server in einem Cluster erfolgt über die Auswahl der zugehörigen Beziehungs-MBean.

Relationship Service-APIs

Beziehungen können innerhalb oder außerhalb von Business-Objektzuordnungen programmgesteuert über Relationship Service-APIs aufgerufen werden.

Es stehen drei API-Typen zur Verfügung:

- APIs zur Bearbeitung von Beziehungsinstanzen (einschließlich der direkten Erstellung, Aktualisierung und Löschung von Instanzdaten)
- APIs zur Unterstützung von Beziehungsmustern (einschließlich correlate(), correlateforeignKeyLookup)
- Muster für die Beziehungssuche (Such-APIs)

Enterprise Service Bus in IBM Business Process Manager

IBM Business Process Manager unterstützt die Integration von Anwendungsservices und enthält dieselben Funktionen wie WebSphere Enterprise Service Bus.

Services über einen Enterprise Service Bus verbinden

Mit einem Enterprise Service Bus (ESB) kann die Flexibilität einer SOA maximiert werden. Die Teilnehmer einer Serviceinteraktion werden mit dem ESB und nicht direkt miteinander verbunden.

Wenn der Serviceanforderer eine Verbindung zum ESB herstellt, übernimmt der ESB die Verantwortung für die Zustellung der entsprechenden Anforderungen. Diese Anforderungen werden mithilfe von Nachrichten einem Serviceanbieter zugestellt, der die erforderliche Funktion und Servicequalität bereitstellt. Der ESB erleichtert die Interaktionen zwischen Anforderer und Anbieter und adressiert abweichende Protokolle, Interaktionsmuster oder Servicekompetenzen. Ferner kann der ESB die Überwachung und Verwaltung aktivieren oder erweitern. Der ESB stellt Virtualisierungs- und Verwaltungsfunktionen bereit, die die zentralen Leistungsmerkmale der SOA implementieren und erweitern.

Der ESB zeichnet sich durch die folgenden Funktionen aus:

Position und Identität
: Die Teilnehmer müssen die Position oder Identität der anderen Teilnehmer nicht kennen. Ein Anforderer muss beispielsweise nicht wissen, dass eine Anforderung von mehreren Anbietern verarbeitet werden könnte; Serviceanbieter können ohne Unterbrechung hinzugefügt oder entfernt werden.

Interaktionsprotokoll
: Die Teilnehmer müssen nicht dasselbe Kommunikationsprotokoll und auch nicht denselben Interaktionsstil gemeinsam nutzen. Eine mit SOAP over HTTP gesendete Anforderung kann beispielsweise von einem Anbieter verarbeitet werden, der nur SOAP over JMS (Java Message Service) versteht.

Schnittstelle
: Anforderer und Anbieter müssen keine gemeinsame Schnittstelle verwenden. Der ESB gleicht Unterschiede durch das Transformieren von Anforderungs- und Antwortnachrichten in ein vom Anbieter erwartetes Format aus.

Servicequalität (Interaktionen)
: Teilnehmer oder Systemadministratoren deklarieren ihre Anforderungen an die Servicequalität, einschließlich der Autorisierung von Anforderungen, der Verschlüsselung und Entschlüsselung von Nachrichteninhalten, der automatischen Prüfung von Serviceinteraktionen, und geben an, wie die Anforderungen weitergeleitet werden sollen (z. B. Optimierung im Hinblick auf Geschwindigkeit oder Kosten).

Durch das Einfügen eines ESB zwischen die Teilnehmer kann die Interaktion zwischen den Teilnehmern über ein logisches Konstrukt namens *Mediation* moduliert werden. Mediationen arbeiten mit Nachrichten, die zwischen Anforderern und Anbietern noch nicht vollständig verarbeitet wurden. Mediationen können beispielsweise zum Suchen von Services mit bestimmten Merkmalen, die von einem Anforderer benötigt werden, oder zum Auflösen von Schnittstellendifferenzen zwischen Anforderern und Anbietern verwendet werden. Für komplexe Interaktionen können Mediationen sequenziell verkettet werden.

Unter Verwendung von Mediationen führt ein Enterprise Service Bus die folgenden Aktionen zwischen Anforderer und Service aus:

- *Weiterleitung* von Nachrichten zwischen Services. Ein Enterprise Service Bus bietet eine gemeinsame Kommunikationsinfrastruktur, die zum Verbinden von Services und daher auch zum Verbinden der von diesen Services dargestellten Geschäftsfunktionen verwendet werden kann, ohne dass der Programmierer komplexe Konnektivitätslogik entwickeln und verwalten muss.
- *Konvertieren* von Transportprotokollen zwischen Anforderer und Service. Ein Enterprise Service Bus bietet eine konsistente, standardisierte Möglichkeit zur Integration von Geschäftsfunktionen, die verschiedene IT-Standards verwenden. Dies ermöglicht die Integration von Geschäftsfunktionen, die normalerweise nicht kommunizieren könnten, z. B. das Verbinden von Anwendungen in Abteilungssilos oder das Aktivieren von Anwendungen in unterschiedlichen Unternehmen für die Teilnahme an Serviceinteraktionen.
- *Transformieren* von Nachrichtenformaten zwischen Anforderer und Service. Mit einem Enterprise Service Bus können Geschäftsfunktionen Informationen in unterschiedlichen Formaten austauschen, wobei der Bus sicherstellt, dass die an eine Geschäftsfunktion zugestellten Informationen das von der Anwendung geforderte Format aufweisen.
- *Verarbeiten* von Business-Ereignissen aus unterschiedlichen Quellen. Der Enterprise Service Bus unterstützt zur Bearbeitung von Serviceanforderungen zusätzlich zum Nachrichtenaustausch ereignisgesteuerte Interaktionen.

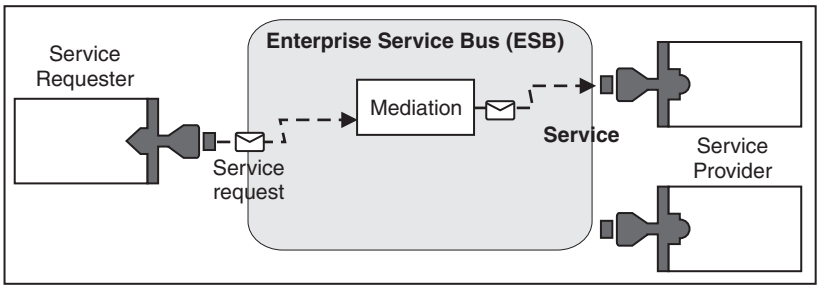


Abbildung 42. Ein Enterprise Service Bus. Der Enterprise Service Bus leitet Nachrichten zwischen Anwendungen weiter, die Anforderer oder Anbieter von Services sind. Der Bus konvertiert Transportprotokolle und transformiert Nachrichtenformate zwischen Anforderern und Anbietern. In dieser Abbildung verwendet jede Anwendung ein anderes Protokoll (dargestellt durch die verschiedenen geometrischen Formen der Connectors) und unterschiedliche Nachrichtenformate.

Wenn Sie einen Enterprise Service Bus verwenden, können Sie sich auf das Kerngeschäft konzentrieren, während die Computersysteme in den Hintergrund treten. Services können nach Bedarf geändert oder erweitert werden, um beispielsweise auf geänderte Geschäftsanforderungen zu reagieren oder zusätzliche Servicekapazität bzw. neue Funktionen hinzuzufügen. Sie können die erforderlichen Änderungen vornehmen, indem Sie den Bus neu konfigurieren, was wenig oder keine Auswirkungen auf die vorhandenen Services und Anwendungen hat, die den Bus verwenden.

Messaging-Infrastruktur für Enterprise Service Bus

IBM Business Process Manager enthält Enterprise Service Bus-Funktionen. IBM Business Process Manager unterstützt die Integration serviceorientierter, nachrichtenorientierter und ereignisgesteuerter Technologien zur Bereitstellung einer standardisierten Messaging-Infrastruktur in einem integrierten Enterprise Service Bus.

Die Enterprise Service-Funktionen, die Sie für Unternehmensanwendungen nutzen können, bieten eine Transportschicht und darüber hinaus Mediationsunterstützung zur Vereinfachung von Serviceinteraktionen. Der Enterprise Service Bus wurde für offene Standards und die serviceorientierte Architektur (Service-Oriented Architecture, SOA) entwickelt. Er basiert auf der leistungsfähigen Java EE-Infrastruktur und den zugehörigen Plattformservices, die durch IBM WebSphere Application Server Network Deployment bereitgestellt werden.

IBM Business Process Manager basiert auf derselben Technologie wie IBM WebSphere Enterprise Service Bus. Diese Funktionen sind Teil der zugrunde liegenden Funktionalität von IBM Business Process Manager; zur Nutzung dieser Funktionen ist keine zusätzliche Lizenz für WebSphere Enterprise Service Bus erforderlich.

Sie können jedoch zusätzliche Einzellizenzen für WebSphere Enterprise Service Bus in Ihrem Unternehmen implementieren, um die Konnektivität der auf IBM Business Process Manager basierenden Prozessintegrationslösungen zu erhöhen. Zum Beispiel könnte WebSphere Enterprise Service Bus näher bei einer SAP-Anwendung installiert werden, um einen IBM WebSphere Adapter for SAP aufzunehmen und SAP-Nachrichten umzusetzen, bevor diese Informationen über das Netz an einen von IBM Business Process Manager koordinierten Business-Prozess gesendet werden.

Messaging- oder Warteschlangenzielhosts:

Ein Host für Messaging oder Warteschlangenziele stellt die Nachrichtenübertragungsfunktion innerhalb eines Servers bereit. Ein Server wird zum Messaging-Zielhost, wenn Sie ihn als Messaging-Ziel konfigurieren.

Eine Messaging-Steuerkomponente wird auf einem Server ausgeführt. Die Messaging-Steuerkomponente stellt Messaging-Funktionen und einen Verbindungspunkt bereit, an dem Anwendungen eine Verbindung zum Bus herstellen. Die asynchrone SCA-Kommunikation, JMS-Importe und -Exporte sowie die asynchrone interne Verarbeitung verwenden Nachrichtenwarteschlangen.

Die Implementierungsumgebung verbindet die Nachrichtenquelle im Zuge der Implementierung der Anwendungsmodule über den Bus mit dem Nachrichtenziel. Wenn Nachrichtenquelle und -ziel bekannt sind, können Sie leichter bestimmen, welcher Typ von Implementierungsumgebung erforderlich ist.

Anwendungen können persistente Daten in einem Datenspeicher speichern. Hierbei handelt es sich um eine Gruppe von Tabellen in einer Datenbank oder einem Schema oder aber in einem Dateispeicher. Die Messaging-Steuerkomponente verwendet eine Instanz einer JDBC-Datenquelle, um mit dieser Datenbank zu interagieren.

Konfigurieren Sie den Messaging-Zielhost, wenn Sie Ihre Implementierungsumgebung über die Option **Server** in der Administrationskonsole definieren oder den Server bei der Softwareinstallation als Zielhost definieren.

Datenspeicher:

Jede Messaging-Steuerkomponente kann einen Datenspeicher verwenden. Hierbei handelt es sich um eine Gruppe von Tabellen in einer Datenbank oder einem Schema, in denen persistente Daten gespeichert werden.

Alle Tabellen im Datenspeicher befinden sich im selben Datenbankschema. Jeder Datenspeicher kann in einer separaten Datenbank erstellt werden. Alternativ können Sie mehrere Datenspeicher in derselben Datenbank erstellen, wobei jeder Datenspeicher ein anderes Schema verwendet.

Eine Messaging-Steuerkomponente verwendet eine Instanz einer JDBC-Datenquelle, um mit der Datenbank zu interagieren, die den Datenspeicher für diese Messaging-Steuerkomponente enthält.

JDBC-Provider:

Sie können JDBC-Provider für die Interaktion von Anwendungen mit relationalen Datenbanken verwenden.

JDBC-Provider werden von Anwendungen verwendet, damit diese mit relationalen Datenbanken interagieren können. Ein JDBC-Provider stellt die Implementierungsklasse eines JDBC-Treibers bereit, um den

Zugriff auf einen bestimmten Datenbanktyp zu ermöglichen. Sie ordnen dem JDBC-Provider eine Datenquelle zu, um einen Verbindungspool für die verwendete Datenbank zu erstellen. Die Kombination aus JDBC-Provider und Datenquellenobjekten entspricht funktional der JCA-Verbindungsfactory (JCA = Java EE Connector Architecture), die Verbindungen zu nicht relationalen Datenbanken bereitstellt.

Weitere Informationen hierzu finden Sie in den Beispielen für die typische Konfiguration einer eigenständigen Umgebung und einer Implementierungsumgebung im vorherigen Abschnitt.

Weitere Informationen zu JDBC-Providern enthält das Information Center von WebSphere Application Server unter dem Stichwort 'JDBC-Provider'.

Service Integration Bus (SIBus) für IBM Business Process Manager:

Ein SIBus ist ein verwalteter Kommunikationsmechanismus, der die Serviceintegration durch synchrones und asynchrones Messaging unterstützt. Ein Bus besteht aus verbundenen Messaging-Steuerkomponenten, die Busressourcen verwalten. Es handelt sich hierbei um eine der WebSphere Application Server-Technologien, auf denen IBM Business Process Manager basiert.

Ein einzelner SIBus und eine einzelne Messaging-Steuerkomponente verwenden standardmäßig dasselbe Datenbankschema wie die Produktdatenbank. Jeder Implementierungsumgebung verfügt über ihren eigenen Bus. Ein einzelner Bus hat den Namen **BPM.*deployment_environment_name*.Bus**.

Das Ziel eines Busses ist eine logische Adresse, der Anwendungen als Produzent und/oder als Konsument zugeordnet werden können. Das Ziel einer Warteschlange ist das Ziel eines Busses, das für das Punkt-zu-Punkt-Messaging verwendet wird.

Jeder Bus kann über ein oder mehrere Bus-Member verfügen, bei denen es sich entweder um Server oder Cluster handelt.

Als *Bustopologie* wird die physische Anordnung der Anwendungsserver, Messaging-Steuerkomponenten und WebSphere MQ-Warteschlangenmanager sowie das Muster der Busverbindungen und Links zwischen diesen Komponenten bezeichnet, das den Enterprise Service Bus ergibt.

Serviceanwendungen und -module

Ein Servicemodul ist ein SCA-Modul, das zur Laufzeit Services bereitstellt. Bei der Implementierung eines Servicemoduls in IBM Business Process Manager erstellen Sie eine zugeordnete Serviceanwendung, die als EAR-Datei gepackt wird.

Servicemodule sind die Basiseinheiten der Implementierung und können Komponenten, Bibliotheken und Staging-Module enthalten, die von der zugeordneten Serviceanwendung verwendet werden. Servicemodule verfügen über Exporte und optional auch über Importe, um die Beziehungen zwischen Modulen sowie Serviceanforderern und -anbietern zu definieren. WebSphere Process Server unterstützt Module für Business-Services und Mediationsmodule. Module und Mediationsmodule sind Typen von SCA-Modulen. Ein Mediationsmodul ermöglicht die Kommunikation zwischen Anwendungen durch das Transformieren des Serviceaufrufs in ein für das Ziel verständliches Format, wobei die Anforderung an das Ziel übergeben und das Ergebnis an den Absender zurückgegeben wird. Ein Modul für einen Business-Service implementiert die Logik eines Business-Prozesses. Ein Modul kann jedoch auch die Mediationslogik enthalten, die in ein Mediationsmodul gepackt werden kann.

Serviceanwendung implementieren

Der Prozess der Implementierung einer EAR-Datei, die eine Serviceanwendung enthält, entspricht dem Prozess der Implementierung jeder beliebigen EAR-Datei. Zur Implementierungszeit können Werte für Mediationsparameter geändert werden. Nach der Implementierung einer EAR-Datei mit einem SCA-Modul können Details zur Serviceanwendung und dem zugeordneten Modul angezeigt werden. Sie können

feststellen, wie ein Servicemodul mit Serviceanforderern (über Exporte) und mit Serviceanbietern (über Importe) verbunden ist.

SCA-Moduldetails anzeigen

Die Servicemoduldetails, die Sie anzeigen können, hängen vom jeweiligen SCA-Modul ab. Sie umfassen die folgenden Attribute:

- Name des SCA-Moduls
- Beschreibung des SCA-Moduls
- Zugeordneter Anwendungsname
- Versionsangaben zum SCA-Modul, sofern das Modul versioniert ist
- SCA-Modulimporte:
  - Importschnittstellen sind abstrakte Definitionen, die beschreiben, wie ein SCA-Modul auf einen Service zugreift.
  - Importbindungen sind konkrete Definitionen, die den physischen Mechanismus angeben, mit dessen Hilfe ein SCA-Modul auf einen Service zugreift. Beispiel: SOAP/HTTP.
- SCA-Modulexporte:
  - Exportschnittstellen sind abstrakte Definitionen, die beschreiben, wie ein Serviceanforderer auf ein SCA-Modul zugreift.
  - Exportbindungen sind konkrete Definitionen, die den physischen Mechanismus angeben, mit dessen Hilfe ein Serviceanforderer auf ein SCA-Modul und indirekt auf einen Service zugreift.
- Eigenschaften des SCA-Moduls

Importe und Importbindungen:

Importe definieren Interaktionen zwischen SCA-Modulen und Serviceanbietern. SCA-Module verwenden Importe, um Komponenten den Zugriff auf externe Services (Services außerhalb des SCA-Moduls) mit einer lokalen Darstellung zu ermöglichen. Importbindungen definieren die Art und Weise, wie der Zugriff auf einen externen Service erfolgt.

Wenn SCA-Module nicht auf externe Services zugreifen müssen, müssen sie über keine Importe verfügen. Mediationsmodule verfügen in der Regel über mindestens einen Import, der zum Übergeben von Nachrichten oder Anforderungen an die gewünschten Ziele verwendet wird.

Schnittstellen und Bindungen

Ein SCA-Modulimport benötigt mindestens eine Schnittstelle und verfügt über eine einzelne Bindung.

- Importschnittstellen sind abstrakte Definitionen, die eine Reihe von Operationen unter Verwendung von WSDL (Web Services Description Language) definieren. WSDL ist eine XML-Sprache zur Beschreibung von Web-Services. Ein SCA-Modul kann viele Importschnittstellen aufweisen.
- Importbindungen sind konkrete Definitionen, die den physischen Mechanismus angeben, der von SCA-Modulen für den Zugriff auf einen externen Service verwendet werden.

Unterstützte Importbindungen

IBM Business Process Manager unterstützt die folgenden Importbindungen:

- SCA-Bindungen verbinden SCA-Module mit anderen SCA-Modulen. SCA-Bindungen werden auch als Standardbindungen bezeichnet.
- Web-Service-Bindungen ermöglichen Komponenten das Aufrufen von Web-Services. Die unterstützten Protokolle sind SOAP1.1/HTTP, SOAP1.2/HTTP und SOAP1.1/JMS.

  Sie können eine SOAP1.1/HTTP- oder SOAP1.2/HTTP-Bindung basierend auf JAX-WS (Java API for XML Web Services) verwenden, die Interaktionen mit Services ermöglicht, die mit Dokument- oder lite-

ralen RPC-Bindungen arbeiten, und Aufrufe mithilfe von JAX-WS-Handlern anpasst. Für die Interaktion mit Services, die eine RPC-codierte Bindung verwenden, und für Fälle, in denen JAX-RPC-Handler zum Anpassen von Aufrufen verwendet werden müssen, wird eine separate SOAP1.1/HTTP-Bindung bereitgestellt.

- Mit HTTP-Bindungen kann über das HTTP-Protokoll auf Anwendungen zugegriffen werden.
- EJB-Importbindungen (EJB - Enterprise JavaBeans) ermöglichen SCA-Komponenten das Aufrufen von Services, die von der Java EE-Geschäftslogik auf einem Java EE-Server bereitgestellt werden.
- EIS-Bindungen stellen Konnektivität zwischen SCA-Komponenten und einem externen EIS (Enterprise Information System, unternehmensweites Informationssystem) bereit. Diese Kommunikation wird durch die Verwendung von Ressourcenadaptern ermöglicht.
- JMS 1.1-Bindungen (JMS - Java Message Service) ermöglichen Interoperabilität mit dem Standard-Messaging-Provider von WebSphere Application Server. JMS kann verschiedene Transporttypen nutzen, einschließlich TCP/IP und HTTP oder HTTPS. Die JMS-Nachrichtenklasse und die fünf zugehörigen Subtypen (Text, Bytes, Object, Stream und Map) werden automatisch unterstützt.
- Generische JMS-Bindungen ermöglichen Interoperabilität mit JMS-Providern anderer Anbieter, die über JMS-ASF (Application Server Facility) mit WebSphere Application Server integriert werden.
- WebSphere MQ-JMS-Bindungen ermöglichen Interoperabilität mit WebSphere MQ-basierten JMS-Providern. Die JMS-Nachrichtenklasse und die fünf zugehörigen Subtypen (Text, Bytes, Object, Stream und Map) werden automatisch unterstützt. Wenn Sie WebSphere MQ als JMS-Provider nutzen möchten, müssen Sie WebSphere MQ-JMS-Bindungen verwenden.
- WebSphere MQ-Bindungen ermöglichen Interoperabilität mit WebSphere MQ. WebSphere MQ-Bindungen können nur mit fernen Warteschlangenmanagern über eine WebSphere MQ-Clientverbindung verwendet werden; die Verwendung mit lokalen Warteschlangenmanagern ist nicht möglich. Verwenden Sie WebSphere MQ-Bindungen, wenn Sie mit nativen WebSphere MQ-Anwendungen kommunizieren möchten.

Dynamischer Aufruf von Services

Services können über eine unterstützte Importbindung aufgerufen werden. Ein Service befindet sich normalerweise an einem Endpunkt, der im Import angegeben ist. Dieser Endpunkt wird als statischer Endpunkt bezeichnet. Durch das Überschreiben des statischen Endpunkts kann ein anderer Service aufgerufen werden. Durch das dynamische Überschreiben von statischen Endpunkten können über unterstützte Importbindungen Services an anderen Endpunkten aufgerufen werden. Ferner kann durch das dynamische Aufrufen von Services auch ein Service aufgerufen werden, für den die unterstützte Importbindung keinen statischen Endpunkt aufweist.

Zum Angeben des Protokolls und der entsprechenden Konfiguration für einen dynamischen Aufruf wird ein Import mit einer zugeordneten Bindung verwendet. Der für den dynamischen Aufruf verwendete Import kann mit der aufrufenden Komponente verknüpft oder zur Laufzeit dynamisch ausgewählt werden.

Bei Web-Service- und SCA-Aufrufen ist es außerdem möglich, einen dynamischen Aufruf ohne einen Import auszuführen, wobei das zugehörige Protokoll und die Konfiguration aus der Endpunkt-URL abgeleitet werden. Der Zieltyp des Aufrufs wird über die Endpunkt-URL ermittelt. Bei Verwendung eines Imports muss die URL mit dem Protokoll der Importbindung kompatibel sein.

- Eine SCA-URL weist auf den Aufruf eines anderen SCA-Moduls hin.
- Eine HTTP- oder JMS-URL weist standardmäßig auf den Aufruf eines Web-Service hin; für diese URLs kann ein zusätzlicher Wert für den Bindungstyp angegeben werden, der darauf hinweist, dass die URL einen Aufruf mittels einer HTTP- oder JMS-Bindung darstellt.
- Für eine Web-Service-HTTP-URL wird standardmäßig SOAP 1.1 verwendet; es kann auch ein Bindungstypwert angegeben werden, der auf die Verwendung von SOAP 1.2 hinweist.

Exporte und Exportbindungen:

Exporte definieren Interaktionen zwischen SCA-Modulen und Serviceanforderern. SCA-Module verwenden Exporte, um Services bereitzustellen. Exportbindungen definieren die Art und Weise, wie der Zugriff auf ein SCA-Modul durch einen Serviceanforderer erfolgt.

Schnittstellen und Bindungen

Ein SCA-Modulexport benötigt mindestens eine Schnittstelle.

- Exportschnittstellen sind abstrakte Definitionen, die eine Reihe von Operationen unter Verwendung von WSDL (Web Services Description Language) definieren. WSDL ist eine XML-Sprache zur Beschreibung von Web-Services. Ein SCA-Modul kann viele Exportschnittstellen aufweisen.
- Exportbindungen sind konkrete Definitionen, die den physischen Mechanismus angeben, den der Serviceanforderer für den Zugriff auf einen Service verwendet. Normalerweise ist für einen SCA-Modulexport eine Bindung angegeben. Ein Export ohne angegebene Bindung wird von der Laufzeit als Export mit einer SCA-Bindung interpretiert.

Unterstützte Exportbindungen

IBM Business Process Manager unterstützt die folgenden Exportbindungen:

- SCA-Bindungen verbinden SCA-Module mit anderen SCA-Modulen. SCA-Bindungen werden auch als Standardbindungen bezeichnet.
- Mit Web-Service-Bindungen können Exporte als Web-Services aufgerufen werden. Die unterstützten Protokolle sind SOAP1.1/HTTP, SOAP1.2/HTTP und SOAP1.1/JMS.

  Sie können eine SOAP1.1/HTTP- oder SOAP1.2/HTTP-Bindung basierend auf JAX-WS (Java API for XML Web Services) verwenden, die Interaktionen mit Services ermöglicht, die mit Dokument- oder literalen RPC-Bindungen arbeiten, und Aufrufe mithilfe von JAX-WS-Handlern anpasst. Für die Interaktion mit Services, die eine RPC-codierte Bindung verwenden, und für Fälle, in denen JAX-RPC-Handler zum Anpassen von Aufrufen verwendet werden müssen, wird eine separate SOAP1.1/HTTP-Bindung bereitgestellt.
- Über HTTP-Bindungen kann mit dem HTTP-Protokoll auf Exporte zugegriffen werden.
- Durch EJB-Exportbindungen (EJB - JavaBeans) können SCA-Komponenten als EJBs zugänglich gemacht werden, sodass die Java EE-Geschäftslogik SCA-Komponenten aufrufen kann, die normalerweise nicht verfügbar wären.
- EIS-Bindungen stellen Konnektivität zwischen SCA-Komponenten und einem externen EIS (Enterprise Information System, unternehmensweites Informationssystem) bereit. Diese Kommunikation wird durch die Verwendung von Ressourcenadaptern ermöglicht.
- JMS 1.1-Bindungen (JMS - Java Message Service) ermöglichen Interoperabilität mit dem Standard-Messaging-Provider von WebSphere Application Server. JMS kann verschiedene Transporttypen nutzen, einschließlich TCP/IP und HTTP oder HTTPS. Die JMS-Nachrichtenklasse und die fünf zugehörigen Subtypen (Text, Bytes, Object, Stream und Map) werden automatisch unterstützt.
- Generische JMS-Bindungen ermöglichen Interoperabilität mit JMS-Providern anderer Anbieter, die über JMS-ASF (Application Server Facility) mit WebSphere Application Server integriert werden.
- WebSphere MQ-JMS-Bindungen ermöglichen Interoperabilität mit WebSphere MQ-basierten JMS-Providern. Die JMS-Nachrichtenklasse und die fünf zugehörigen Subtypen (Text, Bytes, Object, Stream und Map) werden automatisch unterstützt. Wenn Sie WebSphere MQ als JMS-Provider nutzen möchten, müssen Sie WebSphere MQ-JMS-Bindungen verwenden.
- WebSphere MQ-Bindungen ermöglichen Interoperabilität mit WebSphere MQ. Für die Verbindung zu einem MQ-Warteschlangenmanager auf einem fernen System wird eine ferne Verbindung (Clientverbindung) verwendet. Eine lokale Verbindung (Bindungsverbindung) ist eine Direktverbindung zu WebSphere MQ. Diese kann nur bei einer Verbindung zu einem MQ-Warteschlangenmanager auf demselben System verwendet werden. WebSphere MQ lässt beide Verbindungstypen zu, MQ-Bindungen unterstützen jedoch nur ferne Verbindungen (Clientverbindungen).

Mediationsmodule:

Mediationsmodule sind SCA-Module, die das Format, den Inhalt oder das Ziel von Serviceanforderungen ändern können.

Mediationsmodule arbeiten mit Nachrichten, die zwischen Serviceanforderern und Serviceanbietern noch nicht vollständig verarbeitet wurden. Sie können Nachrichten an unterschiedliche Serviceanbieter weiterleiten und den Nachrichteninhalt oder die Nachrichtenstruktur korrigieren. Von Mediationsmodulen können Funktionen wie beispielsweise Nachrichtenprotokollierung und Fehlerverarbeitung bereitgestellt werden, die auf Ihre Bedürfnisse zugeschnitten sind.

Bestimmte Aspekte von Mediationsmodulen können über die Administrationskonsole geändert werden, ohne das Modul erneut implementieren zu müssen.

Komponenten von Mediationsmodulen

Mediationsmodule enthalten die folgenden Elemente:

- Importe, die Interaktionen zwischen SCA-Modulen und Serviceanbietern definieren. Mithilfe dieser Importe können SCA-Module externe Services wie lokale Services aufrufen. Importe von Mediationsmodulen können angezeigt und die Bindung geändert werden.
- Exporte, die Interaktionen zwischen SCA-Modulen und Serviceanforderern definieren. Sie ermöglichen SCA-Modulen das Anbieten von Services und definieren die externen Schnittstellen (Zugriffspunkte) eines SCA-Moduls. Exporte von Mediationsmodulen können angezeigt werden.
- SCA-Komponenten, die Bausteine für SCA-Module (z. B. Mediationsmodule) sind. Mit Integration Designer können SCA-Module und Komponenten grafisch erstellt und angepasst werden. Nach der Implementierung können über die Administrationskonsole bestimmte Aspekte eines Mediationsmoduls angepasst werden, ohne das Modul erneut implementieren zu müssen.

  Normalerweise enthalten Mediationsmodule einen bestimmten Typ von SCA-Komponente, der als *Mediationsablaufkomponente* bezeichnet wird. Mediationsablaufkomponenten definieren Mediationsabläufe.

  Eine Mediationsablaufkomponente kann kein, ein oder mehrere Mediationsbasiselemente enthalten. IBM Business Process Manager unterstützt eine Gruppe von Mediationsbasiselementen, die Funktionalität für Nachrichtenweiterleitung und -umsetzung bereitstellen. Wird zusätzliche Flexibilität für Mediationsbasiselemente benötigt, können Sie mit dem angepassten Mediationsbasiselement angepasste Logik aufrufen.

  Mediationsmodule, die keine Mediationsablaufkomponenten enthalten, sind dafür vorgesehen, Serviceanforderungen aus einem Protokoll in einen anderes zu transformieren. Es könnte beispielsweise eine Serviceanforderung mit SOAP/JMS erstellt werden, die vor der Weiterleitung jedoch in SOAP/HTTP transformiert werden muss.

  Anmerkung: In IBM Business Process Manager können Mediationsmodule angezeigt und gewisse Änderungen vorgenommen werden. Die SCA-Komponenten in einem Modul können über IBM Business Process Manager jedoch nicht angezeigt oder geändert werden. Verwenden Sie Integration Designer zum Anpassen von SCA-Komponenten.

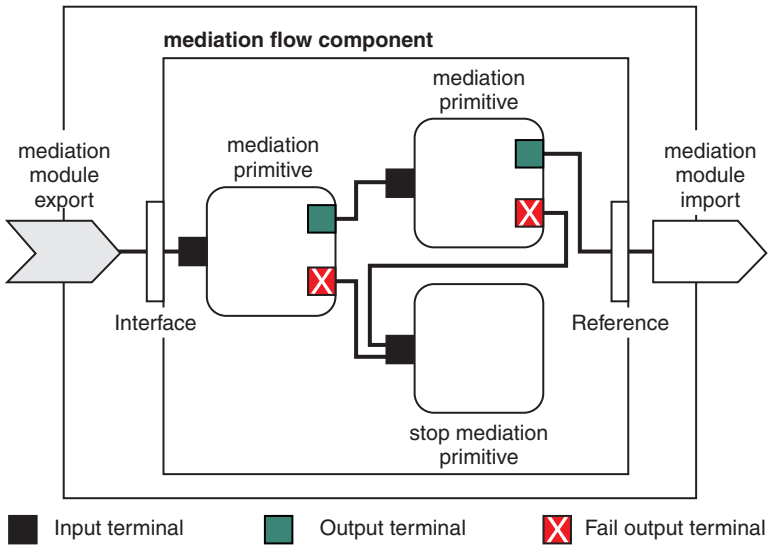


Abbildung 43. Vereinfachtes Beispiel eines Mediationsmoduls. Das Mediationsmodul enthält eine Mediationsablaufkomponente, die Mediationsbasiselemente enthält.

- Eigenschaften

  Mediationsbasiselemente verfügen über Eigenschaften; einige dieser Eigenschaften können in der Administrationskonsole als zusätzliche Eigenschaften eines SCA-Moduls angezeigt werden.

  Der Integrationsentwickler muss die Eigenschaften hochstufen, damit sie in der Administrationskonsole von IBM Business Process Manager angezeigt werden. Bestimmte Eigenschaften eignen sich für eine Verwaltungskonfiguration; Integration Designer bezeichnet diese Eigenschaften als hochstufbare Eigenschaften, da sie vom Integrationszyklus auf den Administrationszyklus hochgestuft werden können. Andere Eigenschaften sind nicht für die Verwaltungskonfiguration geeignet, da sich eine Änderung dieser Eigenschaften so auf den Mediationsablauf auswirken kann, dass das Mediationsmodul erneut implementiert werden muss. In Integration Designer sind die Eigenschaften aufgeführt, die unter den hochgestuften Eigenschaften eines Mediationsbasiselement hochgestuft werden können.

  Werte von hochgestufte Eigenschaften können über die Administrationskonsole von IBM Business Process Manager geändert werden, ohne das Mediationsmodul erneut implementieren oder den Server/das Modul erneut starten zu müssen.

  Im Allgemeinen werden Eigenschaftsänderungen von Mediationsabläufen umgehend verwendet. Wenn die Eigenschaftsänderungen jedoch in einer Deployment Manager-Zelle erfolgen, werden diese Änderungen bei der Synchronisation eines Knotens auf diesem Knoten wirksam. Mediationsabläufe, die noch ausgeführt werden, verwenden weiterhin die vorherigen Werte.

  Anmerkung: Über die Administrationskonsole können nur Eigenschaftswerte und keine Eigenschaftsgruppen, -namen oder -typen geändert werden. Zur Änderung von Eigenschaftsgruppen, -namen oder -typen muss Integration Designer verwendet werden.
- Ein Mediationsmodul oder eine abhängige Bibliothek kann auch Unterabläufe definieren. In einen Unterablauf ist eine Gruppe aus Mediationsbasiselementen eingebunden, die als wieder verwendbarer Teil der Integrationslogik miteinander verknüpft sind. Einem Mediationsablauf kann ein Basiselement hinzugefügt werden, um einen Unterablauf aufzurufen.

Mediationsmodule implementieren

Mediationsmodule werden mit Integration Designer erstellt und im Allgemeinen in Form einer EAR-Datei in IBM Business Process Manager implementiert.

Zur Implementierungszeit können die Werte von hochgestuften Eigenschaften geändert werden.

Sie können ein Mediationsmodul aus Integration Designer exportieren und Integration Designer dazu veranlassen, das Mediationsmodul in eine JAR-Datei und die JAR-Datei in eine EAR-Datei zu packen. Anschließend kann die EAR-Datei durch Installation einer neuen Anwendung über die Administrationskonsole implementiert werden.

Mediationsmodule können als eine Entität betrachtet werden. SCA-Module werden jedoch durch eine Reihe von XML-Dateien definiert, die in einer JAR-Datei gespeichert sind.

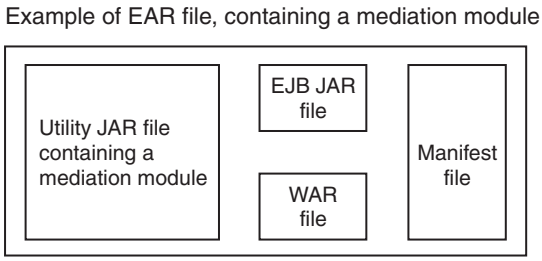


Abbildung 44. Vereinfachtes Beispiel einer EAR-Datei mit einem Mediationsmodul. Die EAR-Datei enthält JAR-Dateien. Die Dienstprogramm-JAR-Datei enthält ein Mediationsmodul.

Mediationsbasiselemente:

Mediationsablaufkomponenten arbeiten mit Nachrichtenflüssen zwischen Servicekomponenten. Die Funktionalität einer Mediationskomponente wird durch *Mediationsbasiselemente* implementiert, die Standard-Serviceimplementierungstypen implementieren.

Eine Mediationsablaufkomponente verfügt über mindestens einen Ablauf (z. B. einen Ablauf für Anforderungen und einen Ablauf für Antworten).

IBM Business Process Manager unterstützt eine Gruppe von bereitgestellten Mediationsbasiselementen, die Standardmediationsfunktionen für Mediationsmodule oder Module in IBM Business Process Manager implementieren. Wenn Sie besondere Mediationsfunktionen benötigen, können Sie eigene angepasste Mediationsbasiselemente entwickeln.

Ein Mediationsbasiselement definiert eine eingehende Operation, die Nachrichten verarbeitet oder ausführt, die durch Servicenachrichtenobjekte (Service Message Objects, SMOs) dargestellt werden. Ein Mediationsbasiselement kann auch abgehende Operationen definieren, die Nachrichten an andere Komponenten oder Module senden.

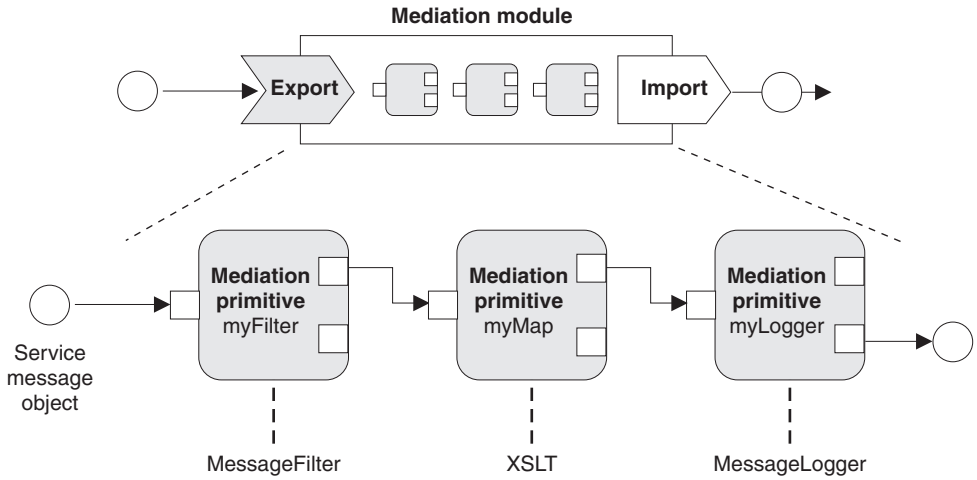


Abbildung 45. Mediationsmodul mit drei Mediationsbasiselementen

Mit Integration Designer können Mediationsbasiselemente konfiguriert und die zugehörigen Eigenschaften definiert werden. Einige dieser Eigenschaften können durch Hochstufen für den Laufzeitadministrator sichtbar gemacht werden. Hochstufbare Eigenschaften von Mediationsbasiselementen können auch dynamische Eigenschaften sein. Eine dynamische Eigenschaft kann zur Laufzeit mithilfe einer Richtliniendatei überschrieben werden.

Integration Designer ermöglicht ferner das grafische Modellieren und Assemblieren von Mediationsablaufkomponenten aus Mediationsbasiselementen sowie das Assemblieren von Mediationsmodulen oder Modulen aus Mediationsablaufkomponenten. Die Administrationskonsole bezeichnet Mediationsmodule und Module als SCA-Module.

Darüber hinaus ermöglicht Integration Designer das Definieren von Unterabläufen in Modulen oder den zugehörigen Bibliotheken. Ein Unterablauf kann alle Mediationsbasiselemente mit Ausnahme des Mediationsbasiselements für die Richtlinienauflösung enthalten. Ein Unterablauf wird über einen Anforderungs- oder Antwortablauf oder einen anderen Unterablauf mithilfe des Mediationsbasiselements für Unterabläufe aufgerufen. Eigenschaften, die aus Mediationsbasiselementen in einem Unterablauf hochgestuft wurden, werden in den Mediationsbasiselementen für Unterabläufe als Eigenschaften verfügbar gemacht. Diese können dann erneut hochgestuft werden, bis sie die Modulstufe erreichen, auf der sie vom Laufzeitadministrator geändert werden können.

Unterstützte Mediationsbasiselemente

Die folgenden Mediationsbasiselemente werden von IBM Business Process Manager unterstützt:

Business-Objektzuordnung
Transformiert Nachrichten.

- Definiert Nachrichtenumsetzungen mithilfe einer Business-Objektzuordnung, die wieder verwendet werden kann.
- Ermöglicht die grafische Definition von Nachrichtenumsetzungen unter Verwendung des Editors für Business-Objektzuordnungen.
- Kann den Inhalt einer Nachricht ändern.
- Kann einen Eingabenachrichtentyp in einen anderen Ausgabenachrichtentyp transformieren.

Angepasste Mediation
Ermöglicht die Implementierung eigener Mediationslogik in Java-Code. Das angepasste Mediationsbasiselement kombiniert die Flexibilität benutzerdefinierter Mediationsbasiselemente mit der

Einfachheit vordefinierter Mediationsbasiselemente. Durch die folgenden Aktionen können komplexe Transformationen und Weiterleitungsmuster erstellt werden:

- Erstellen von Java-Code.
- Erstellen eigener Eigenschaften.
- Hinzufügen neuer Terminals.

Sie können zwar einen Service über das angepasste Mediationsbasiselement aufrufen, zum Aufrufen von Services wurde jedoch das Mediationsbasiselement für Serviceaufrufe (Service Invoke) konzipiert, das zusätzliche Funktionalität (z. B. Wiederholungen) bereitstellt.

Datenhandler

Ermöglicht die Transformation von Nachrichtenteilen. Der Datenhandler wird zum Konvertieren eines Nachrichtenelements aus einem physischen Format in eine logische Struktur oder aus einer logischen Struktur in ein physisches Format verwendet. Die primäre Funktion des Basiselements besteht darin, ein physisches Format, z. B. eine Textzeichenfolge in einem JMS-Textnachrichtenobjekt, in eine logische Business-Objektstruktur zu konvertieren und anschließend zu rekonvertieren. Diese Mediation wird häufig für die folgenden Aktionen verwendet:

- Transformation eines Abschnitts der Eingabenachricht aus einer definierten Struktur in eine andere. Beispiel: Ein SMO enthält einen durch Kommas begrenzten Zeichenfolgewert, und Sie möchten diesen Wert in ein bestimmtes Business-Objekt (BO) parsen.
- Ändern des Nachrichtentyps. Beispiel: Ein JMS-Export wurde für die Verwendung einer JMS-Datenbindung (Basistyp) konfiguriert und innerhalb des Mediationsmoduls legt der Integrationsentwickler fest, dass der Inhalt um eine bestimmte BO-Struktur erweitert werden soll.

Datenbanksuche

Ändert Nachrichten anhand von Informationen aus einer vom Benutzer bereitgestellten Datenbank.

- Sie müssen eine Datenbank, eine Datenquelle und die Einstellungen zur Serverauthentifizierung definieren, die das Mediationsbasiselement für die Datenbanksuche (Database Lookup) verwenden kann. Die Administrationskonsole unterstützt Sie bei dieser Aufgabe.
- Das Mediationsbasiselement für die Datenbanksuche kann nur aus einer Tabelle lesen.
- Die angegebene Schlüsselspalte muss einen eindeutigen Wert enthalten.
- Die Daten in den Wertspalten müssen entweder einen einfachen XML-Schematyp oder einen XML-Schematyp aufweisen, der einen einfachen XML-Schematyp erweitert.

Endpunktsuche

Ermöglicht die dynamische Weiterleitung von Anforderungen durch die Suche nach Serviceendpunkten in einem Repository.

- Serviceendpunktinformationen werden aus WebSphere Service Registry and Repository (WSRR) abgerufen. Die WSRR-Registry kann lokal oder fern sein.
- Registry-Änderungen können über die WSRR-Administrationskonsole durchgeführt werden.
- IBM Business Process Manager muss wissen, welche Registry verwendet werden soll. Deshalb müssen über die IBM Business Process Manager-Administrationskonsole WSRR-Zugriffsdefinitionen erstellt werden.

Ereignisemitter

Erweitert die Überwachung durch die Möglichkeit, Ereignisse aus einer Mediationsablaufkomponente heraus zu senden.

- Die Mediationsaktion kann durch Inaktivieren des Kontrollkästchens ausgesetzt werden.
- Ereignisse des Ereignisemitters können mit dem CBE-Browser (CBE - Common Base Events) in IBM Business Process Manager angezeigt werden.
- Aus Leistungsgründen sollten Ereignisse nur an signifikanten Punkten in einem Mediationsablauf gesendet werden.
- Sie können die Teile der Nachricht definieren, die im Ereignis enthalten sind.

- Die Ereignisse werden in Form von Common Base Events an einen Common Event Infrastructure-Server gesendet.
- Ereigniskonsumenten müssen die Struktur der Common Base Events verstehen, um die Informationen des Ereignisemitters vollständig nutzen zu können. Die Common Base Events verfügen über ein Gesamtschema, das jedoch die anwendungsspezifischen Daten nicht modelliert, die in den erweiterten Datenelementen enthalten sind. Zur Modellierung der erweiterten Datenelemente generieren die Integration Designer-Tools für jedes konfigurierte Mediationsbasiselement 'Ereignisemitter' eine Definitionsdatei für den Common Event Infrastructure-Ereigniskatalog. Die Definitionsdateien für den Ereigniskatalog sind Exportartefakte, die zur Ihrer Unterstützung dienen; sie werden von Integration Designer oder der IBM Business Process Manager-Laufzeit nicht verwendet. Ziehen Sie die Definitionsdateien für den Ereigniskatalog zu Rate, wenn Sie Anwendungen zur Verarbeitung von Ereignissen des Ereignisemitters erstellen.
- In IBM Business Process Manager können Sie weitere Überwachungsoptionen angeben. Sie können beispielsweise Ereignisse überwachen, die von Importen und Exporten ausgegeben werden.

Fehlgeschlagen
Stoppt einen bestimmten Pfad im Ablauf und generiert eine Ausnahmebedingung.

Eingabefächerung
Hilft beim Aggregieren (Zusammenfassen) von Nachrichten.

- Kann nur in Verbindung mit dem Mediationsbasiselement 'Ausgabefächerung' verwendet werden.
- Zusammen ermöglichen die Mediationsbasiselemente für die Eingabe- und Ausgabefächerung das Aggregieren von Daten in einer Ausgabenachricht.
- Das Mediationsbasiselement 'Eingabefächerung' empfängt Nachrichten, bis ein Entscheidungspunkt erreicht wird; dann wird eine Nachricht ausgegeben.
- Der gemeinsam genutzte Kontext sollte für die Aggregationsdaten verwendet werden.

Ausgabefächerung
Hilft beim Aufteilen und Aggregieren (Zusammenfassen) von Nachrichten.

- Zusammen ermöglichen die Mediationsbasiselemente für die Eingabe- und Ausgabefächerung das Aggregieren von Daten in einer Ausgabenachricht.
- Im Iterationsmodus kann mit dem Mediationsbasiselement 'Ausgabefächerung' eine einzelne Eingabenachricht durchlaufen werden, die ein sich wiederholendes Element enthält. Für jedes Vorkommen des sich wiederholenden Elements wird eine Nachricht gesendet.
- Der gemeinsam genutzte Kontext sollte für die Aggregationsdaten verwendet werden.

HTTP-Header-Setter
Stellt einen Mechanismus zur Verwaltung von Headern in HTTP-Nachrichten bereit.

- Kann HTTP-Nachrichtenheader erstellen, definieren, kopieren oder löschen.
- Kann mehrere Aktionen zur Änderung mehrerer HTTP-Header definieren.

Zuordnung
Transformiert Nachrichten.

- Ermöglicht die Ausführung von XSL-Transformationen (XSL - Extensible Stylesheet Language) oder Transformationen von Geschäftsobjektzuordnungen.
- Nachrichten werden mithilfe einer XSLT 1.0- oder XSLT 2.0-Transformation oder einer Geschäftsobjektzuordnungstransformation transformiert. Die XSL-Transformationen arbeiten mit einer XML-Serialisierung der Nachricht, während die Transformation von Geschäftsobjektzuordnungen mit Service Data Objects (SDOs) arbeitet.

Nachrichtenelementsetter
Stellt einen einfachen Mechanismus zum Definieren von Nachrichteninhalten bereit.

- Kann Nachrichtenelemente ändern, hinzufügen oder löschen.

- Der Nachrichtentyp wird nicht geändert.
- Die Daten in den Wertspalten müssen entweder einen einfachen XML-Schematyp oder einen XML-Schematyp aufweisen, der einen einfachen XML-Schematyp erweitert.

Nachrichtenfilter
Leitet Nachrichten basierend auf dem Nachrichteninhalt an verschiedene Pfade weiter.

- Die Mediationsaktion kann durch Inaktivieren des Kontrollkästchens ausgesetzt werden.

Nachrichtenprotokollfunktion
Protokolliert Nachrichten in einer relationalen Datenbank oder über eine eigene angepasste Protokollfunktion. Die Nachrichten werden in XML gespeichert, sodass eine Nachverarbeitung der Daten durch XML-konforme Anwendungen möglich ist.

- Die Mediationsaktion kann durch Inaktivieren des Kontrollkästchens ausgesetzt werden.
- Das Schema der rationalen Datenbank (Tabellenstruktur) wird durch IBM definiert.
- Das Mediationsbasiselement für die Nachrichtenprotokollfunktion verwendet standardmäßig die Common-Datenbank. Die Laufzeitumgebung ordnet die Datenquelle unter **jdbc/mediation/messageLog** der Common-Datenbank zu.
- Zur Anpassung des Verhaltens der angepassten Protokollfunktion können Sie Implementierungsklassen für Handler definieren. Optional können Implementierungsklassen für Formatierungsprogramme und/oder Filter angegeben werden, um das Verhalten der angepassten Protokollfunktion anzupassen.

MQ-Header-Setter
Stellt einen Mechanismus zur Verwaltung von Headern in MQ-Nachrichten bereit.

- Kann MQ-Nachrichtenheader erstellen, definieren, kopieren oder löschen.
- Kann mehrere Aktionen zur Änderung mehrerer MQ-Header definieren.

Richtlinienauflösung
Ermöglicht die dynamische Konfiguration von Anforderungen durch die Suche nach Serviceendpunkten und zugehörigen Richtliniendateien in einem Repository.

- Mit einer Richtliniendatei können die hochgestuften Eigenschaften anderer Mediationsbasiselemente dynamisch überschrieben werden.
- Serviceendpunkt- und Richtlinieninformationen werden aus WebSphere Service Registry and Repository (WSRR) abgerufen. Die WSRR-Registry kann lokal oder fern sein.
- Registry-Änderungen können über die WSRR-Administrationskonsole durchgeführt werden.
- IBM Business Process Manager muss wissen, welche Registry verwendet werden soll. Deshalb müssen über die IBM Business Process Manager-Administrationskonsole WSRR-Zugriffsdefinitionen erstellt werden.

Serviceaufruf
Ruft einen Service aus einem Mediationsablauf heraus auf, anstatt bis zum Ende des Mediationsablaufs zu warten und den Callout-Mechanismus zu verwenden.

- Wenn der Service einen Fehler zurückgibt, kann der Aufruf für diesen Service wiederholt oder ein anderer Service aufgerufen werden.
- Das Mediationsbasiselement 'Serviceaufruf' ist ein leistungsstarkes Mediationsbasiselement, das für einfache Serviceaufrufe allein oder für komplexe Mediationen in Kombination mit anderen Mediationsbasiselementen verwendet werden kann.

Nachrichtentyp festlegen
Mit diesem Element können während der Integrationsentwicklung schwach typisierte Nachrichtenfelder wie stark typisierte Felder behandelt werden. Ein Feld ist schwach typisiert, wenn es mehr als einen Datentyp enthalten kann. Ein Feld ist stark typisiert, wenn der Typ und die interne Struktur des Felds bekannt sind.

- Zur Laufzeit kann mit dem Mediationsbasiselement 'Nachrichtentyp festlegen' überprüft werden, ob der Inhalt einer Nachricht mit dem erwarteten Datentypen übereinstimmt.

SOAP-Header-Setter
Stellt einen Mechanismus zur Verwaltung von Headern in SOAP-Nachrichten bereit.
- Kann SOAP-Nachrichtenheader erstellen, definieren, kopieren oder löschen.
- Kann mehrere Aktionen zur Änderung mehrerer SOAP-Header definieren.

Stoppen
Stoppt einen bestimmten Pfad im Ablauf und generiert keine Ausnahmebedingung.

Typfilter
Ermöglicht die typabhängige Weiterleitung von Nachrichten an verschiedene Pfade in einem Ablauf.

WebSphere eXtreme Scale - Abrufen
Sie können Informationen aus der Umgebung eines eXtreme Scale-Server-Cache abrufen.
- Sie können Werte im Cache suchen und diese Werte mithilfe eines Schlüssels als Elemente in der Nachricht speichern.
- Durch Kombinieren der eXtreme Scale-Mediationsbasiselemente für Speichern (Store) und Abrufen (Retrieve) können Sie die Antwort von einem Back-End-System in den Cache stellen. Zukünftige Anforderungen benötigen keinen Zugriff auf das betreffende Back-End-System.
- Sie müssen eXtreme Scale-Definitionen mithilfe der WebSphere ESB-Administrationskonsole erstellen, um angeben zu können, welcher eXtreme Scale-Server verwendet werden soll.

WebSphere eXtreme Scale - Speichern
Sie können Informationen in der Umgebung eines eXtreme Scale-Server-Cache speichern.
- Sie können Informationen mithilfe eines Schlüssels und Objekts in einem eXtreme Scale-Cache speichern.
- Durch Kombinieren der eXtreme Scale-Mediationsbasiselement für Speichern (Store) und Abrufen (Retrieve) können Sie mithilfe des Mediationsbasiselements 'Speichern' Daten in den Cache stellen und mithilfe des Mediationsbasiselements 'Abrufen' zuvor in den Cache gestellte Daten abrufen.
- Sie müssen eXtreme Scale-Definitionen mithilfe der WebSphere ESB-Administrationskonsole erstellen, um angeben zu können, welcher eXtreme Scale-Server verwendet werden soll.

Dynamische Weiterleitung:

Zur Weiterleitung von Nachrichten gibt es verschiedene Möglichkeiten. Dazu werden Endpunkte verwendet, die zur Integrationszeit oder dynamisch zur Laufzeit definiert werden.

Die dynamische Weiterleitung bezieht sich auf zwei Fälle der Nachrichtenweiterleitung:
- Nachrichtenweiterleitung, bei der der Ablauf dynamisch ist, alle möglichen Endpunkte aber in einem SCA-Modul vordefiniert sind.
- Nachrichtenweiterleitung, bei der sowohl der Ablauf als auch die Endpunktauswahl dynamisch sind. Die Serviceendpunkte werden zur Laufzeit aus einer externen Quelle ausgewählt.

Dynamische Endpunktauswahl

Die Laufzeit ist in der Lage, Anforderungs- und Antwortnachrichten an eine Endpunktadresse weiterzuleiten, die durch ein Nachrichtenheaderelement identifiziert wird. Dieses Nachrichtenheaderelement kann durch Mediationsbasiselemente in einem Mediationsablauf aktualisiert werden. Die Endpunktadresse kann mit Informationen aus einer Registry, einer Datenbank oder der Nachricht selbst aktualisiert werden. Antwortnachrichten werden nur dann weitergeleitet, wenn die Nachricht von einem Web-Service-Export (JAX-WS) gesendet wird.

Für das SCA-Modul muss die Eigenschaft `Dynamischen Endpunkt verwenden, falls im Nachrichtenheader festgelegt` definiert sein, damit die dynamische Weiterleitung von der Laufzeit für eine Anforderung oder eine Antwort implementiert werden kann. Integrationsentwickler können die Ei-

genschaft `Dynamischen Endpunkt verwenden, falls im Nachrichtenheader festgelegt` definieren oder hochstufen (zur Laufzeit sichtbar machen), damit sie vom Laufzeitadministrator definiert werden kann. Moduleigenschaften können im Fenster für die **Moduleigenschaften** angezeigt werden. Klicken Sie zum Anzeigen des Fensters auf **Anwendungen** > **SCA-Module** > **Moduleigenschaften**. Der Integrationsentwickler ordnet hochgestuften Eigenschaften Aliasnamen zu und diese Namen werden in der Administrationskonsole angezeigt.

Registry

Mit IBM WebSphere Service Registry and Repository (WSRR) können Sie Serviceendpunktinformationen speichern und dann SCA-Module erstellen, um Endpunkte aus der WSRR-Registry abzurufen.

Bei der Entwicklung von SCA-Modulen wird das Mediationsbasiselement für die Endpunktsuche verwendet, um einem Mediationsablauf das Abfragen einer WSRR-Registry bezüglich eines Serviceendpunkts oder einer Gruppe von Serviceendpunkten zu ermöglichen. Wenn ein SCA-Modul eine Gruppe von Endpunkten abruft, muss das Modul ein anderes Mediationsbasiselement verwenden, um den gewünschten Endpunkt auszuwählen.

Mediationsrichtliniensteuerung für Serviceanforderungen:

Mithilfe von Mediationsrichtlinien können Mediationsabläufe zwischen Serviceanforderern und Serviceanbietern gesteuert werden.

Mediationsabläufe können mithilfe von Mediationsrichtlinien gesteuert werden, die in IBM WebSphere Service Registry and Repository (WSRR) gespeichert sind. Die Implementierung der Servicerichtlinienverwaltung in WSRR basiert auf dem Web Services Policy Framework (WS-Policy).

Zur Steuerung von Serviceanforderungen mit Mediationsrichtlinien müssen in der WSRR-Registry geeignete SCA-Module und Mediationsrichtliniendokumente vorhanden sein.

Mediationsrichtlinie an eine Serviceanforderung anhängen

Bei der Entwicklung eines SCA-Moduls, das eine Mediationsrichtlinie verwenden soll, muss in den Mediationsablauf ein Mediationsbasiselement für die Richtlinienauflösung eingefügt werden. Während der Ausführung ruft das Mediationsbasiselement für die Richtlinienauflösung Mediationsrichtlinieninformationen aus der Registry ab. Deshalb muss das SCA-Modul eine Mediationsablaufkomponente enthalten, damit die Steuerung von Serviceanforderungen durch Mediationsrichtlinien unterstützt wird.

In der Registry kann mindestens eine Mediationsrichtlinie an ein SCA-Modul oder an einen vom SCA-Modul verwendeten Zielservice angehängt werden. Angehängte Mediationsrichtlinien können für alle von diesem SCA-Modul verarbeiteten Servicenachrichten verwendet werden (befinden sich innerhalb des Geltungsbereich). Die Mediationsrichtlinien können über Richtlinienanhänge verfügen, die Bedingungen definieren. Über Mediationsrichtlinienbedingungen können unterschiedliche Mediationsrichtlinien in verschiedenen Kontexten angewendet werden. Darüber hinaus können Mediationsrichtlinien über Klassifizierungen zum Angeben eines Governance-Zustands verfügen.

WebSphere Service Registry and Repository:

Mit WebSphere Service Registry and Repository (WSRR) können Sie Informationen zu Serviceendpunkten und Mediationsrichtlinien speichern und verwalten und auf diese Informationen zugreifen. Verwenden Sie WSRR, um Serviceanwendungen dynamischer zu gestalten und besser an sich ändernde Geschäftsbedingungen anzupassen.

Einführung

Mediationsabläufe können WSRR als Mechanismus für die dynamische Suche verwenden, der Informationen zu Serviceendpunkten oder Mediationsrichtlinien bereitstellt.

Zur Konfiguration des Zugriffs auf WSRR müssen über die Administrationskonsole WSRR-Definitionsdokumente erstellt werden. Alternativ dazu können Sie die WSRR-Verwaltungsbefehle über den Scripting-Client 'wsadmin' verwenden. Die WSRR-Definitionen und die zugehörigen Verbindungseigenschaften sind der Mechanismus, der für die Verbindung zu einer Registry-Instanz und zum Abrufen eines Serviceendpunkts oder einer Mediationsrichtlinie verwendet wird.

Serviceendpunkte

Mit WSRR können Informationen zu den Services gespeichert werden, die Sie bereits verwenden, in Zukunft verwenden möchten oder über die Sie Informationen benötigen. Diese Services können sich in Ihren oder in anderen Systemen befinden. Eine Anwendung könnte WSRR beispielsweise verwenden, um den Service zu suchen, der zur Erfüllung der jeweiligen Funktions- und Leistungsanforderungen am besten geeignet ist.

Bei der Entwicklung eines SCA-Moduls, das über WSRR auf Serviceendpunkte zugreifen soll, muss in den Mediationsablauf ein Mediationsbasiselement für die Endpunktsuche eingefügt werden. Zur Laufzeit ruft das Mediationsbasiselement für die Endpunktsuche Serviceendpunkte aus der Registry ab.

Mediationsrichtlinien

Sie können WSRR auch zum Speichern von Mediationsrichtlinieninformationen verwenden. Mediationsrichtlinien können Ihnen durch das dynamische Überschreiben von Moduleigenschaften bei der Steuerung von Serviceanforderungen helfen. Wenn WSRR Mediationsrichtlinien enthält, die an ein Objekt angehängt sind, das entweder das SCA-Modul oder den Zielservice darstellt, können die Moduleigenschaften von diesen Mediationsrichtlinien überschrieben werden. Durch die Erstellung von Mediationsrichtlinienbedingungen können in verschiedenen Kontexten unterschiedliche Mediationsrichtlinien verwendet werden.

Anmerkung: Mediationsrichtlinien befassen sich mit der Steuerung von Mediationsabläufen, aber nicht mit der Sicherheit.

Bei der Entwicklung eines SCA-Moduls, das eine Mediationsrichtlinie verwenden soll, muss in den Mediationsablauf ein Mediationsbasiselement für die Richtlinienauflösung eingefügt werden. Während der Ausführung ruft das Mediationsbasiselement für die Richtlinienauflösung Mediationsrichtlinieninformationen aus der Registry ab.

WebSphere eXtreme Scale:

Durch Verwendung des Produkts WebSphere eXtreme Scale (eXtreme Scale) können Sie ein Cachingsystem bereitstellen, das mit einer IBM Business Process Manager-Anwendung integriert werden kann. Die Verwendung von eXtreme Scale mit IBM Business Process Manager kann die Serviceantwortzeiten und -zuverlässigkeit verbessern und ermöglicht die Bereitstellung zusätzlicher Integrationsfunktionen.

eXtreme Scale fungiert als flexibles, skalierbares, speicherinternes Datengrid. Das Datengrid ermöglicht ein dynamisches Zwischenspeichern, Partitionieren, Replizieren und Verwalten von Anwendungsdaten und Geschäftslogik auf mehreren Servern. Mit eXtreme Scale können Sie auch für Servicequalität sorgen, wie beispielsweise für Transaktionsintegrität, hohe Verfügbarkeit und vorhersehbare Antwortzeiten.

Sie können Mediationsabläufe verwenden, um auf die Cachingfunktion von eXtreme Scale zuzugreifen, indem Sie die WebSphere eXtreme Scale-Mediationsbasiselemente in Ihren Ablauf integrieren. Bei der Entwicklung eines SCA-Moduls (Service Component Architecture), das Informationen in einem eXtreme Sca-

le-Cache speichern muss, ist es erforderlich, das WebSphere eXtreme Scale-Mediationsbasiselement für Speichern (Store) in den Mediationsablauf zu integrieren. Wenn Sie Informationen aus einem eXtreme Scale-Cache abrufen wollen, müssen Sie das WebSphere eXtreme Scale-Mediationsbasiselement für Abrufen (Retrieve) integrieren. Durch Kombinieren der beiden Mediationsbasiselemente in einem Mediationsablauf können Sie die Antwort von einem Back-End-System in den Cache stellen, sodass zukünftige Anforderungen diese Antwort aus dem Cache abrufen können.

Um den Zugriff auf eXtreme Scale zu konfigurieren, müssen Sie mithilfe der Administrationskonsole eine WebSphere eXtreme Scale-Definition erstellen. Alternativ dazu können Sie auch die WebSphere eXtreme Scale-Verwaltungsbefehle über den Scripting-Client 'wsadmin' verwenden. Bei einer eXtreme Scale-Definition handelt es sich um den Mechanismus, der von den WebSphere eXtreme Scale-Mediationsbasiselementen für Abrufen (Retrieve) und Speichern (Store) verwendet wird, um eine Verbindung zu einem eXtreme Scale-Server herzustellen.

Message Service Clients

Message Service Clients stehen für C/C++ und .NET zur Verfügung, um Nicht-Java-Anwendungen eine Verbindung zum Enterprise Service Bus zu ermöglichen.

Message Service Clients for C/C++ and .NET stellen eine API namens XMS bereit, die über dieselbe Gruppe von Schnittstellen wie die JMS-API (JMS - Java Message Service) verfügt. Message Service Client for C/C++ enthält zwei Implementierungen von XMS, eine für C-Anwendungen und eine weitere für C++-Anwendungen. Message Service Client for .NET enthält eine vollständig verwaltete Implementierung von XMS, die von allen .NET-konformen Sprachen verwendet werden kann.

Message Service Client for .NET erhalten Sie unter http://www-01.ibm.com/support/docview.wss?rs=0&q1=IA9H&uid=swg24011756&loc=en_US&cs=utf-8&cc=us&lang=en.

Message Service Client for C/C++ erhalten Sie unter http://www-01.ibm.com/support/docview.wss?rs=0&q1=ia94&uid=swg24007092&loc=en_US&cs=utf-8&cc=us&lang=en.

Sie können auch die Java EE-Clientunterstützung von WebSphere Application Server Network Deployment installieren und verwenden, einschließlich Web-Service-Client, EJB-Client und JMS-Client.

Kapitel 2. Weitere Informationen zu Schlüsselkonzepten

Dieser Abschnitt dient als Ausgangspunkt zum Recherchieren der Technologien, die in IBM Business Process Manager zum Einsatz kommen.

Authoringszenarios

Mithilfe von Szenarios können Sie ein besseres Verständnis der Komponenten und Produkte in der BPM-Produktfamilie gewinnen und mit diesen Komponenten und Produkten arbeiten.

Versionierung

Der Lebenszyklus einer Prozessanwendung beginnt mit deren Erstellung und setzt sich über einen Kreislauf aus Aktualisierung, Implementierung, Co-Implementierung, Deimplementierung und Archivierung fort. Der Begriff *Versionierung* (auch: Versionssteuerung) bezeichnet einen Mechanismus zur Steuerung des Lebenszyklus einer Prozessanwendung durch eindeutiges Identifizieren der einzelnen Versionen dieser Prozessanwendung.

Die Arbeitsweise der Versionierung in IBM Business Process Manager ist davon abhängig, ob Sie eine Prozessanwendung implementieren (aus dem Repository in IBM Process Center) oder eine Unternehmensanwendung (direkt aus IBM Integration Designer).

Die Prozessanwendungen und Toolkits, die Sie aus Process Center in einer Laufzeitumgebung implementieren, sind standardmäßig versioniert. Bei Unternehmensanwendungen können Sie auswählen, dass Module und Bibliotheken in IBM Integration Designer versioniert werden.

Darüber hinaus können Sie Versionen einer Benutzertask oder Statusmaschine erstellen, damit mehrere Versionen der Task oder Statusmaschine gemeinsam mit der Laufzeitumgebung existieren können.

Versionierung von Prozessanwendungen

Bei der Versionierung handelt es sich um die Funktionalität, die es der Laufzeitumgebung ermöglicht, Snapshots im Lebenszyklus einer Prozessanwendung zu identifizieren und mehrere Snapshots auf einem Prozessserver auszuführen.

Um zu verstehen, wie eine Versionierung für Prozessanwendungen durchgeführt wird, ist es wichtig zu beachten, dass eine Prozessanwendung ein Container ist, der verschiedene Artefakte enthält, die in der oder von der Prozessanwendung verwendet werden (zum Beispiel Prozessmodelle oder BPDs, Toolkit-Referenzen, Services, oder Verfolgungen). Eine Versionierung findet auf Container-Ebene und nicht auf der Ebene der einzelnen Artefakte statt. Bei Prozessanwendungen bedeutet dies, dass eine Versionierung stattfindet, wenn Sie einen Snapshot aufnehmen.

Sie können Snapshots vergleichen, um die Unterschiede zwischen den Versionen zu bestimmen. Hat ein Entwickler zum Beispiel ein Problem mit einem Service behoben und an dieser Stelle einen Snapshot der übergeordneten Prozessanwendung oder des übergeordneten Toolkits aufgenommen, damit anschließend ein anderer Entwickler mehrere zusätzliche Änderungen an demselben Service vornimmt und einen neuen Snapshot aufnimmt, so könnte der Projektmanager die beiden Snapshots miteinander vergleichen, um festzustellen, welche Änderungen wann und von wem vorgenommen wurden. Falls der Projektmanager die zusätzlichen Änderungen am Service als unnötig einschätzt, könnte er zum Snapshot der ursprünglichen Korrektur zurückkehren.

Sie können verschiedene Versionen (Snapshots) einer Prozessanwendung gleichzeitig auf einem Server ausführen. Wenn Sie einen neuen Snapshot installieren, entfernen Sie das Original oder führen Sie es weiterhin aus.

Versionskontext

Jeder Snapshot verfügt über eindeutige Metadaten, um die Version zu identifizieren (als Versionskontext bezeichnet). Diese Kennung wird von Ihnen zugeordnet; IBM empfiehlt hingegen die Verwendung eines numerischen Versionssystems, das aus drei Ziffern im Format <major>.<minor>.<service> besteht. Die Abschnitte zu Namenskonventionen enthalten eine ausführlichere Beschreibung dieses Versionierungsschemas.

IBM Business Process Manager ordnet jeder Prozessanwendung einen globalen Namensbereich zu. Der globale Namensbereich ist entweder ein Vorschlag der Prozessanwendung oder ein bestimmter Snapshot der Prozessanwendung. Der vom Server verwendete Versionsname darf aus maximal sieben Zeichen bestehen; deshalb ist der zugeordnete Name ein Akronym, das Zeichen aus dem angegebenen Snapshotnamen verwendet. Snapshotakronyme sind mit ihren Snapshotnamen identisch, wenn die Snapshotnamen dem empfohlenen IBM VRM-Stil entsprechen und aus maximal sieben Zeichen bestehen. Beispiel: Der Snapshotname 1.0.0 verfügt über das Akronym 1.0.0 und der Snapshotname 10.3.0 verfügt über das Akronym 10.3.0. Das Snapshotakronym ist innerhalb des Kontexts der Prozessanwendung im Geltungsbereich des Process Center Servers garantiert eindeutig. Aus diesem Grund kann das Snapshot-Akronym nicht bearbeitet werden.

Versionisierungsaspekte für Prozessanwendungen in mehreren Clustern

Es ist möglich, dieselbe Version einer Prozessanwendung auf verschiedenen Clustern derselben Zelle zu implementieren. Zur Differenzierung der einzelnen Implementierungen derselben Version von Prozessanwendung sollten Sie für jede Implementierung einen Snapshot erstellen und bei der Benennung des Snapshots eine für die Zelle eindeutige ID in den Namen einbinden (zum Beispiel 'v1.0_cell1_1' und 'v1.0_cell1_2'). Es handelt sich (aus der Perspektive des Lebenszyklusmanagements) bei jedem Snapshot um eine neue Version der Prozessanwendung, die denselben Inhalt und dieselbe Funktion besitzt.

Wenn Sie eine Prozessanwendung auf einem Cluster implementieren, wird eine automatische Synchronisation der Knoten ausgeführt.

Versionisierungsaspekte für Process Designer-Toolkits

Beachten Sie, dass Snapshots von Prozessanwendungen im Allgemeinen aufgenommen werden, wenn Sie getestet oder implementiert werden soll. Toolkit-Snapshots werden aber in der Regel aufgenommen, wenn dieses Toolkit von Prozessanwendungen verwendet werden soll. Falls Sie danach Aktualisierungen an dem Toolkit vornehmen möchten, müssen Sie, wenn Sie soweit sind, einen weiteren Snapshot der aktuellen Arbeitsversion erstellen und die Eigner von Prozessanwendungen und Toolkits können dann entscheiden, ob sie zu dem neuen Snapshot wechseln möchten.

Versionierung von Modulen und Bibliotheken

Wenn sich ein Modul oder eine Bibliothek in einer Prozessanwendung oder einem Toolkit befindet, übernimmt das Modul bzw. die Bibliothek den Lebenszyklus der betreffenden Prozessanwendung bzw. des betreffenden Toolkits (Versionen, Snapshots, Verfolgungen usw.). Modul- und Bibliotheksnamen müssen im Geltungsbereich einer Prozessanwendung oder eines Toolkits eindeutig sein.

In diesem Abschnitt wird die Versionierung von Modulen und Bibliotheken, die zusammen mit Prozessanwendungen verwendet werden, beschrieben. Hinweis: Wenn Sie Module direkt aus IBM Integration Designer in Process Server implementieren, können Sie weiterhin so verfahren, dass Sie Modulen während der Implementierung Versionsnummern zuweisen, wie im Abschnitt „Versionierte Module und Bibliotheken erstellen” beschrieben.

Die abhängigen Bibliotheken eines Moduls oder einer Bibliothek, das bzw. die IBM Process Center zugeordnet ist, müssen sich in derselben Prozessanwendung oder in einem abhängigen Toolkit befinden.

In der folgenden Tabelle sind die Optionen aufgeführt, die Sie im Abhängigkeitseditor von IBM Integration Designer auswählen können, wenn eine Bibliothek einer Prozessanwendung oder einem Toolkit zugeordnet ist:

Tabelle 36. Abhängigkeiten für Modul, Prozessanwendung oder Toolkit und globale Bibliotheken

Geltungsbereich der Bibliothek	**Beschreibung**	**Kann abhängig sein von . . .**
Modul	Eine Kopie dieser Bibliothek ist für jedes Modul, das sie verwendet, auf dem Server vorhanden.	Eine Bibliothek im Modulgeltungsbereich kann von allen Bibliothekstypen abhängig sein.
Prozessanwendung oder Toolkit	Die Bibliothek wird von allen Modulen im Geltungsbereich der Prozessanwendung bzw. des Toolkits gemeinsam genutzt. Diese Einstellung wird wirksam, wenn die Implementierung über IBM Process Center erfolgt. Wenn die Implementierung außerhalb von IBM Process Center erfolgt, dann wird die Bibliothek in jedes Modul kopiert. **Anmerkung:** Bibliotheken, die in IBM Integration Designer Version 8 erstellt werden, verfügen standardmäßig über die gemeinsame Nutzungsebene **Prozessanwendung oder Toolkit**.	Eine Bibliothek dieses Typs kann nur von globalen Bibliotheken abhängig sein.
Global	Die Bibliothek wird von allen aktiven Modulen gemeinsam genutzt.	Eine globale Bibliothek kann nur von anderen globalen Bibliotheken abhängig sein. **Anmerkung:** Zur Implementierung der globalen Bibliothek müssen Sie eine gemeinsam genutzte WebSphere-Bibliothek konfigurieren. Weitere Informationen finden Sie unter „Modul- und Bibliotheksabhängigkeiten”.

Module und Bibliotheken, die Prozessanwendungen oder Toolkits zugeordnet sind

Für Module und Bibliotheken, die Prozessanwendungen oder Toolkits zugeordnet sind, ist keine Versionierung erforderlich.

Module und Bibliotheken, die einer Prozessanwendung oder einem Toolkit zugeordnet sind, brauchen nicht versioniert zu werden. Tatsächlich können Sie im Editor für Abhängigkeiten auch keine Version eines Moduls oder einer Bibliothek erstellen, das bzw. die einer Prozessanwendung oder einem Toolkit zugeordnet ist. Module und Bibliotheken, die einer Prozessanwendung oder einem Toolkit zugeordnet sind, arbeiten mit Snapshots (Momentaufnahmen), einer Funktion im Process Center, um dasselbe Ergebnis wie eine Version zu erzielen.

Bibliotheken, die einer Prozessanwendung oder einem Toolkit zugeordnet sind, haben keine erforderliche Versionsnummer im Abschnitt **Bibliotheken** des Editors für Abhängigkeiten, da keine Version benötigt wird.

Namenskonventionen

Eine Namenskonvention dient zur Unterscheidung der verschiedenen Versionen einer Prozessanwendung, wenn diese den Kreislauf aus Aktualisierung, Implementierung, Co-Implementierung, Deimplementierung und Archivierung durchläuft.

In diesem Abschnitt sind die Konventionen beschrieben, die zur eindeutigen Kennzeichnung der Versionen einer Prozessanwendung verwendet werden.

Ein *Versionskontext* ist eine Kombination von Akronymen, die eine Prozessanwendung oder ein Toolkit eindeutig beschreiben. Jeder Typ von Akronym hat eine Namenskonvention. Ein Akronym kann maximal sieben Zeichen aus dem Zeichensatz [A-Z0-9_] enthalten; eine Ausnahme bildet das Snapshotakronym, das außerdem einen Punkt enthalten kann.

- Das Akronym für die Prozessanwendung wird erstellt, wenn die Prozessanwendung erstellt wird. Es kann maximal sieben Zeichen lang sein.
- Das Akronym für den Snapshot wird bei der Erstellung des Snapshots automatisch erstellt. Es kann maximal sieben Zeichen lang sein.

  Falls der Snapshotname die Kriterien für ein gültiges Snapshotakronym erfüllt, sind der Snapshotname und das Akronym identisch.

  Anmerkung: Benennen Sie bei der versionssensitiven Weiterleitungsfunktion der Mediationsablaufkomponente Ihren Snapshot so, dass der Name mit dem Schema <*version*>.<*release*>.<*modifikation*> konform ist (z. B. **1.0.0**). Da das Snapshotakronym auf eine Länge von sieben Zeichen beschränkt ist, können maximal fünf Ziffern (plus zwei Punkte) verwendet werden. Daher sollten Sie bei der Erhöhung der Ziffernfelder sorgsam vorgehen, weil alle Zeichen nach den ersten sieben Zeichen abgeschnitten werden.

  Der Snapshotname **11.22.33** wird beispielsweise zum Snapshotakronym **11.22.3**.
- Das Verfolgungsakronym wird automatisch aus den Anfangsbuchstaben der Wörter des Verfolgungsnamens gebildet. Eine neue Verfolgung, die mit dem Namen **My New Track** erstellt wurde, hat beispielsweise den Akronymwert **MNT**.

  Der Standardname und das Standardakronym für die Verfolgung sind **Main**. Bei der Implementierung auf einem IBM Process Center-Server wird das Verfolgungsakronym in den Versionierungskontext aufgenommen, sofern es sich nicht um das Verfolgungsakronym **Main** handelt.

Eine Geschäftsprozessdefinition in einer Prozessanwendung wird für gewöhnlich durch das Akronym für den Prozessanwendungsnamen, das Snapshotakronym und den Namen der Geschäftsprozessdefinition gekennzeichnet. Wählen Sie für Ihre Geschäftsprozessdefinitionen nach Möglichkeit immer eindeutige Namen. Wenn Namen doppelt vorhanden sind, könnten die folgenden Probleme auftreten:

- Unter Umständen wird es nicht möglich sein, die Geschäftsprozessdefinitionen ohne eine Form der Mediation als Web-Services zugänglich zu machen.
- Unter Umständen wird es nicht möglich sein, eine in IBM Process Designer erstellte Geschäftsprozessdefinition aus einem in IBM Integration Designer erstellten BPEL-Prozess heraus aufzurufen.

Der Versionskontext variiert abhängig davon, wie die Prozessanwendung implementiert wird.

Namenskonventionen für Process Center-Serverimplementierungen

Auf dem IBM Process Center-Server können Sie einen Snapshot einer Prozessanwendung sowie einen Snapshot eines Toolkits implementieren. Darüber hinaus können Sie die aktuelle Arbeitsversion einer Prozessanwendung oder eines Toolkits implementieren. Der Versionskontext variiert je nach Implementierungstyp.

Bei Prozessanwendungen wird die Arbeitsversion der Prozessanwendung oder der jeweilige Prozessanwendungssnapshot verwendet, um die Version eindeutig zu kennzeichnen.

Toolkits können mit einer Prozessanwendung oder mehreren Prozessanwendungen implementiert werden; der Lebenszyklus eines jeden Toolkits ist jedoch an den Lebenszyklus der Prozessanwendung gebunden. Jede Prozessanwendung verfügt über eine eigene Kopie der abhängigen Toolkits, die auf dem Server implementiert sind. Ein implementiertes Toolkit wird nicht von mehreren Prozessanwendungen gemeinsam genutzt.

Falls die Verfolgung, die der Prozessanwendung zugeordnet ist, einen anderen Namen als den Standardnamen **Main** (Hauptelement) hat, ist das Akronym der Verfolgung ebenfalls Bestandteil des Versionskontextes.

Weitere Informationen finden Sie unter „Beispiele” auf Seite 35 im weiteren Verlauf dieses Abschnitts.

Prozessanwendungssnapshots

Bei Implementierungen von Prozessanwendungssnapshots ist der Versionskontext eine Kombination aus den folgenden Elementen:

- Akronym des Prozessanwendungsnamens
- Akronym der Prozessanwendungsverfolgung (bei anderer Verfolgung als **Main**)
- Akronym des Prozessanwendungssnapshots

Eigenständige Toolkits

Bei Implementierungen von Toolkit-Snapshots ist der Versionskontext eine Kombination aus den folgenden Elementen:

- Akronym des Toolkitnamens
- Akronym der Toolkitverfolgung (bei anderer Verfolgung als **Main**)
- Akronym des Toolkit-Snapshots

Arbeitsversionen

Arbeitsversionen ('Tips') von Prozessanwendungen werden während der iterativen Erprobung in Process Designer verwendet. Sie können nur auf Process Center-Servern implementiert werden.

Bei Implementierungen von Arbeitsversionen der Prozessanwendungen ist der Versionskontext eine Kombination aus den folgenden Elementen:

- Akronym des Prozessanwendungsnamens
- Akronym der Prozessanwendungsverfolgung (bei anderer Verfolgung als **Main**)
- 'Tip'

Toolkitarbeitsversionen werden ebenfalls während der iterativen Erprobung in Process Designer verwendet. Sie werden nicht auf einem Produktionsserver implementiert.

Bei Implementierungen von Toolkitarbeitsversionen ist der Versionskontext eine Kombination aus den folgenden Elementen:

- Akronym des Toolkitnamens
- Akronym der Toolkitverfolgung (bei anderer Verfolgung als **Main**)
- 'Tip'

Beispiele

Ressourcen sollten unter Verwendung des Versionskontextes eindeutig benannt und extern gekennzeichnet werden.

- Die folgende Tabelle enthält einige Beispiele für eindeutige Namen. In diesem Beispiel verwendet eine aktuelle Arbeitsversion einer Prozessanwendung den Standardverfolgungsnamen (**Main**):

Tabelle 37. Aktuelle Arbeitsversion einer Prozessanwendung mit Standardverfolgungsnamen

Namenstyp	**Beispiel**
Prozessanwendungsname	**Process Application 1**

Tabelle 37. Aktuelle Arbeitsversion einer Prozessanwendung mit Standardverfolgungsnamen (Forts.)

Namenstyp	**Beispiel**
Akronym des Prozessanwendungsnamens	**PA1**
Prozessanwendungsverfolgung	**Main**
Akronym der Prozessanwendungsverfolgung	"" (bei Verfolgung **Main**)
Prozessanwendungssnapshot	
Akronym des Prozessanwendungssnapshots	**Tip**

Alle SCA-Module, die dieser aktuellen Arbeitsversion der Prozessanwendung zugeordnet sind, beinhalten den Versionskotext, wie in der folgenden Tabelle dargestellt:

Tabelle 38. SCA-Module und versionssensitive EAR-Dateien

Name des SCA-Moduls	**Versionssensitiver Name**	**Versionssensitiver EAR-/Anwendungsname**
M1	PA1-Tip-M1	PA1-Tip-M1.ear
M2	PA1-Tip-M2	PA1-Tip-M2.ear

- Die folgende Tabelle enthält ein Beispiel für eine aktuelle Arbeitsversion einer Prozessanwendung, die nicht den Standardverfolgungsnamen verwendet.

Tabelle 39. Aktuelle Arbeitsversion einer Prozessanwendung, die nicht den Standardverfolgungsnamen verwendet

Namenstyp	**Beispiel**
Prozessanwendungsname	**Process Application 1**
Akronym des Prozessanwendungsnamens	**PA1**
Prozessanwendungsverfolgung	**Track1**
Akronym der Prozessanwendungsverfolgung	**T1**
Prozessanwendungssnapshot	
Akronym des Prozessanwendungssnapshots	**Tip**

Alle SCA-Module, die dieser aktuellen Arbeitsversion der Prozessanwendung zugeordnet sind, beinhalten den Versionskotext, wie in der folgenden Tabelle dargestellt:

Tabelle 40. SCA-Module und versionssensitive EAR-Dateien

Name des SCA-Moduls	**Versionssensitiver Name**	**Versionssensitiver EAR-/Anwendungsname**
M1	PA1-T1-Tip-M1	PA1-T1-Tip-M1.ear
M2	PA1-T1-Tip-M2	PA1-T1-Tip-M2.ear

Ähnliche Namenskonventionen gelten auch für erweiterte Implementierungen von Toolkitarbeitsversionen und -snapshots. Sie gelten ebenso für erweiterte Snapshots bei einer Installation auf Process Server.

- Die folgende Tabelle enthält einige Beispiele für eindeutige Namen. In diesem Beispiel verwendet ein Prozessanwendungssnapshot den Standardverfolgungsnamen (**Main**):

Tabelle 41. Prozessanwendungssnapshot mit Standardverfolgungsnamen

Namenstyp	**Beispiel**
Prozessanwendungsname	**Process Application 1**
Akronym des Prozessanwendungsnamens	**PA1**
Prozessanwendungsverfolgung	**Main**
Akronym der Prozessanwendungsverfolgung	"" (bei Verfolgung **Main**)

Tabelle 41. Prozessanwendungssnapshot mit Standardverfolgungsnamen (Forts.)

Namenstyp	Beispiel
Prozessanwendungssnapshot	**Prozessanwendungssnapshot V1**
Akronym des Prozessanwendungssnapshots	**PSV1**

Alle SCA-Module, die diesem Prozessanwendungssnapshot zugeordnet sind, beinhalten den Versionskotext, wie in der folgenden Tabelle dargestellt:

Tabelle 42. SCA-Module und versionssensitive EAR-Dateien

Name des SCA-Moduls	Versionssensitiver Name	Versionssensitiver EAR-/Anwendungsname
M1	PA1-PSV1-M1	PA1-PSV1-M1.ear
M2	PA1-PSV1-M2	PA1-PSV1-M2.ear

- Die folgende Tabelle enthält ein Beispiel für einen Prozessanwendungssnapshot, der nicht den Standardverfolgungsnamen verwendet.

Tabelle 43. Prozessanwendungssnapshot, der nicht den Standardverfolgungsnamen verwendet

Namenstyp	Beispiel
Prozessanwendungsname	**Process Application 1**
Akronym des Prozessanwendungsnamens	**PA1**
Prozessanwendungsverfolgung	**Track1**
Akronym der Prozessanwendungsverfolgung	**T1**
Prozessanwendungssnapshot	**Prozessanwendungssnapshot V1**
Akronym des Prozessanwendungssnapshots	**PSV1**

Alle SCA-Module, die diesem Prozessanwendungssnapshot zugeordnet sind, beinhalten den Versionskotext, wie in der folgenden Tabelle dargestellt:

Tabelle 44. SCA-Module und versionssensitive EAR-Dateien

Name des SCA-Moduls	Versionssensitiver Name	Versionssensitiver EAR-/Anwendungsname
M1	PA1-T1-PSV1-M1	PA1-T1-PSV1-M1.ear
M2	PA1-T1-PSV1-M2	PA1-T1-PSV1-M2.ear

Namenskonventionen für Process Server-Implementierungen

Auf dem Process Server können Sie einen Snapshot der Prozessanwendung implementieren. Mit dem Akronym des Prozessanwendungssnapshots wird die Version eindeutig gekennzeichnet.

Bei Implementierungen von Prozessanwendungssnapshots ist der Versionskontext eine Kombination aus den folgenden Elementen:

- Akronym des Prozessanwendungsnamens
- Akronym des Prozessanwendungssnapshots

Ressourcen sollten unter Verwendung des Versionskontextes eindeutig benannt und extern gekennzeichnet werden. Die folgende Tabelle enthält einige Beispiele für eindeutige Namen:

Tabelle 45. Beispiele von Namen und Akronymen

Namenstyp	Beispiel
Prozessanwendungsname	**Process Application 1**
Akronym des Prozessanwendungsnamens	**PA1**

Tabelle 45. Beispiele von Namen und Akronymen (Forts.)

Namenstyp	Beispiel
Prozessanwendungssnapshot	**1.0.0**
Akronym des Prozessanwendungssnapshots	**1.0.0**

Bei einer Ressource, wie z. B. einem Modul oder einer Bibliothek, ist der Versionskontext Bestandteil der Identität der Ressource.

Das Beispiel in der folgenden Tabelle umfasst zwei Module und beschreibt, wie die zugeordneten EAR-Dateien den Versionskontext beinhalten:

Tabelle 46. SCA-Module und versionssensitive EAR-Dateien

Name des SCA-Moduls	Versionssensitiver Name	Versionssensitiver EAR-/Anwendungsname
M1	PA1-1.0.0-M1	PA1-1.0.0-M1.ear
M2	PA1-1.0.0-M2	PA1-1.0.0-M2.ear

Das Beispiel in der folgenden Tabelle umfasst zwei Bibliotheken auf Prozessanwendungsebene und beschreibt, wie die zugeordneten JAR-Dateien den Versionskontext beinhalten:

Tabelle 47. Bibliotheken auf Prozessanwendungsebene und versionssensitive JAR-Dateien

Name der SCA-Bibliothek auf Prozessanwendungsebene	Versionssensitiver Name	Name der versionssensitiven JAR-Datei
Lib1	PA1-1.0.0-Lib1	PA1-1.0.0-Lib1.jar
Lib2	PA1-1.0.0-Lib2	PA1-1.0.0-Lib2.jar

Versionssensitive Bindungen

Prozessanwendungen können SCA-Module enthalten, die Import- und Exportbindungen umfassen. Wenn Sie Anwendungsversionen zusammen implementieren, muss die Bindung für jede Version der Anwendung eindeutig sein. Einige Bindungen werden während der Implementierung automatisch aktualisiert, um die Eindeutigkeit der Versionen sicherzustellen. In anderen Fällen müssen Sie die Bindung nach der Implementierung aktualisieren, damit die Eindeutigkeit gewährleistet ist.

Der Geltungsbereich einer *versionssensitiven* Bindung ist an eine bestimmte Version einer Prozessanwendung gebunden. Dies stellt ihre Eindeutigkeit bei Prozessanwendungen sicher. In den folgenden Abschnitten sind die Bindungen aufgeführt, die automatisch in versionssensitive Bindungen geändert werden, sowie sämtliche Aktionen, die Sie zur Laufzeit ausführen müssen, wenn eine Bindung nicht versionssensitiv ist. Überlegungen, die bei der Erstellung von Modulen berücksichtigt werden sollten, enthält der Abschnitt „Hinweise zur Verwendung von Bindungen".

SCA

Das Ziel einer SCA-Bindung wird zur Sicherstellung der Versionssensitivität automatisch während der Implementierung umbenannt, wenn die Import- und Exportbindungen des Moduls in demselben Geltungsbereich der Prozessanwendung definiert sind.

Falls die Bindungen nicht in demselben Geltungsbereich der Prozessanwendung definiert sind, wird eine Informationsnachricht protokolliert. Sie müssen die Importbindung nach der Implementierung bearbeiten, um die Endpunktzieladresse zu ändern. Um die Endpunktzieladresse zu ändern, können Sie die Administrationskonsole verwenden.

Web-Service (JAX-WS oder JAX-RPC)

Die Endpunktzieladresse einer Web-Service-Bindung wird während der Implementierung automatisch versionssensitiv umbenannt, wenn alle folgenden Bedingungen erfüllt sind:

- Die Standardnamenskonvention wurde für die Adresse berücksichtigt:

  http://*ip*:*port*/*modulname*Web/sca/*exportname*
- Die Endpunktadresse ist SOAP/HTTP.
- Die Import- und Exportbindungen des Moduls sind in demselben Geltungsbereich der Prozessanwendung definiert.

Falls diese Bedingungen nicht erfüllt werden, wird eine Informationsnachricht protokolliert. Die Maßnahme, die Sie ergreifen müssen, ist davon abhängig, wie Sie die Prozessanwendung implementieren:

- Falls Sie Ihre Prozessanwendung zusätzlich zu anderen Versionen implementieren, müssen Sie die SOAP/HTTP-Endpunkt-URL oder die SOAP/JMS-Zielwarteschlange manuell so umbenennen, dass sie bei den Versionen der Prozessanwendung eindeutig ist. Um die Endpunktzieladresse nach der Implementierung zu ändern, können Sie die Administrationskonsole verwenden.
- Falls Sie lediglich eine einzige Version der Prozessanwendung implementieren, können Sie diese Nachricht ignorieren.

Bei Co-Implementierungen von Snapshots von SOAP- oder JMS-Web-Service-Bindungen ist die Maßnahme, die Sie ergreifen müssen, davon abhängig, wie Sie die Prozessanwendung implementieren.

- Wenn sich Import und Zielexport in derselben Prozessanwendung befinden, führen Sie die folgenden Schritte aus, bevor Sie die Prozessanwendung im Process Center veröffentlichen und den Snapshot erstellen:
  1. Ändern Sie die Endpunkt-URL des Exports. Stellen Sie sicher, dass Ziel und Verbindungsfactory eindeutig sind.
  2. Ändern Sie die Endpunkt-URL des Imports in die gleiche URL, die Sie im vorherigen Schritt für den Export angegeben haben.
- Wenn sich Import und Zielexport in unterschiedlichen Prozessanwendungen befinden, führen Sie die folgenden Schritte aus:
  1. Ändern Sie die Endpunkt-URL des Exports. Stellen Sie sicher, dass Ziel und Verbindungsfactory eindeutig sind.
  2. Veröffentlichen Sie die Prozessanwendung im Process Center.
  3. Erstellen Sie den Snapshot.
  4. Implementieren Sie die Prozessanwendung auf dem Process Server.
  5. Ändern Sie die Endpunkt-URL des entsprechenden Imports mit der WebSphere-Administrationskonsole in die gleiche URL, die Sie für den Export angegeben haben.

HTTP

Die Endpunkt-URL-Adresse einer HTTP-Bindung wird während der Implementierung automatisch versionssensitiv umbenannt, wenn alle folgenden Bedingungen erfüllt sind:

- Die Standardnamenskonvention wurde für die Adresse berücksichtigt:

  http(s)://*ip*:*port*/*modulname*Web/*kontextpfad_im_export*
- Die Import- und Exportbindungen des Moduls sind in demselben Geltungsbereich der Prozessanwendung definiert.

Falls diese Bedingungen nicht erfüllt werden, wird eine Informationsnachricht protokolliert. Die Maßnahme, die Sie ergreifen müssen, ist davon abhängig, wie Sie die Prozessanwendung implementieren:

- Falls Sie Ihre Prozessanwendung zusätzlich zu anderen Versionen implementieren, müssen Sie die Endpunkt-URL manuell so umbenennen, dass sie bei den Versionen der Prozessanwendung eindeutig ist. Um die Endpunktzieladresse nach der Implementierung zu ändern, können Sie die Administrationskonsole verwenden.
- Falls Sie lediglich eine einzige Version der Prozessanwendung implementieren, können Sie diese Nachricht ignorieren.

JMS und generisches JMS

Systemgenerierte JMS-Bindungen und generische JMS-Bindungen sind automatisch versionssensitiv.

Anmerkung: Bei benutzerdefinierten JMS-Bindungen und generischen JMS-Bindungen findet während der Implementierung keine automatische Umbenennung zur Gewährleistung der Versionssensitivität statt. Bei einer benutzerdefinierten Bindung müssen Sie die folgenden Attribute so umbenennen, dass sie bei den Versionen der Prozessanwendung eindeutig sind:

- Endpunktkonfiguration
- Empfangszielwarteschlange
- Listener-Port-Name (falls definiert)

Legen Sie das entsprechende Sendeziel fest, wenn Sie den Zielmodulendpunkt ändern.

MQ/JMS und MQ

Während der Implementierung findet keine automatische Umbenennung statt, um die Versionssensitivität von MQ/JMS- oder MQ-Bindungen zu aktivieren.

Sie müssen die folgenden Attribute so umbenennen, dass sie bei den Versionen der Prozessanwendung eindeutig sind:

- Endpunktkonfiguration
- Empfangszielwarteschlange

Legen Sie das entsprechende Sendeziel fest, wenn Sie den Zielmodulendpunkt ändern.

EJB

Während der Implementierung findet keine automatische Umbenennung statt, um die Versionssensitivität von EJB-Bindungen zu aktivieren.

Sie müssen die Attribute für die JNDI-Namen so umbenennen, dass sie bei den Versionen der Prozessanwendung eindeutig sind.

Bitte beachten Sie, dass Clientanwendungen ebenfalls so aktualisiert werden müssen, dass die neuen JNDI-Namen verwendet werden.

EIS

Ein Ressourcenadapter wird bei der Implementierung automatisch in eine versionssensitive Variante umbenannt, sofern der Standardressourcenname (***modulname*App:Adapter Description**) nicht geändert wurde.

Wenn der Standardressourcenname geändert wurde, müssen die Ressourcenadapternamen unter den Versionen der Prozessanwendung eindeutig sein.

Wenn die Ressourcenadapternamen nicht eindeutig sind, wird bei der Implementierung eine entsprechende Informationsnachricht protokolliert. Sie können die Ressourcenadapter nach der Implementierung in der Administrationskonsole umbenennen.

Versionssensitive dynamische Aufrufe

Sie können Mediationsablaufkomponenten so konfigurieren, dass Nachrichten an Endpunkte weitergeleitet werden, die zur Laufzeit dynamisch bestimmt werden. Wenn Sie das Mediationsmodul erstellen, konfigurieren Sie die Endpunktsuche für die Verwendung der versionssensitiven Weiterleitung.

Wenn Sie die Notation IBM_VRM (*<version>.<release>.<modifikation*) für den Snapshot verwenden, können Sie die EAR-Datei der Prozessanwendung an WebSphere Service Registry and Repository (WSRR) exportieren. Wenn Sie das Mediationsmodul erstellen, konfigurieren Sie die Endpunktsuche für die Verwendung der versionssensitiven Weiterleitung. Wählen Sie z. B. **Endpunkt zurückgeben, der mit der neuesten kompatiblen Version von SCA-Modul-basierten Services übereinstimmt** im Feld **Übereinstimmungsrichtlinie** aus und wählen Sie **SCA** im Feld **Bindingtyp** aus.

Zukünftige Versionen der Prozessanwendung werden auf dem Server implementiert und in WSRR veröffentlicht und die Endpunktsuche des Mediationsmoduls ruft dynamisch die letzte kompatible Version des Serviceendpunkts auf.

Alternativ hierzu können Sie das Ziel im SMO-Header setzen und den Wert in der Anforderungsnachricht übertragen.

Prozessanwendungen mit Java-Modulen und -Projekten implementieren

Prozessanwendungen können angepasste Java EE-Module und Java-Projekte enthalten. Wenn Sie Anwendungsversionen zusammen implementieren, muss das angepasste Java EE-Modul für jede Version der Anwendung eindeutig sein.

Beachten Sie, dass angepasste Java EE-Module und Java-Projekte auf einem Server implementiert werden, wenn sie mit einem SCA-Modul implementiert werden, in dem eine Abhängigkeit von dem Modul oder Projekt deklariert ist. Wenn Sie beim Deklarieren der Abhängigkeit nicht die Standardoption **Mit Modul implementieren** auswählen, müssen Sie das Modul oder Projekt manuell implementieren.

Prozessanwendungen mit Geschäftsregeln und Selektoren implementieren

Falls Sie mehrere Versionen einer Prozessanwendung implementieren, die eine Business-Regel oder eine Selektorkomponente enthält, müssen Sie beachten, wie die zugehörigen Metadaten durch die Versionen verwendet werden.

Die dynamischen Metadaten für eine Business-Regel oder eine Selektorkomponente werden zur Laufzeit durch den Komponentennamen, den Zielnamespace der Komponente und den Komponententyp bestimmt. Falls zwei oder mehr Versionen einer Prozessanwendung, die eine Business-Regel oder einen Selektor enthalten, in derselben Laufzeitumgebung implementiert werden, verwenden sie dieselben Metadaten für die Regellogik (Business-Regel) oder die Weiterleitung (Selektor).

Damit die einzelnen Versionen der Business-Regel oder der Selektorkomponente der Prozessanwendung jeweils eigene dynamische Metadaten (Regellogik oder Weiterleitung) verwenden können, muss der Zielnamespace so refaktoriert werden, dass er für jede Version der Prozessanwendung eindeutig ist.

Konfigurationsobjekte

Mithilfe der AdminConfig-Befehle des WebSphere-Befehlszeilenverwaltungstools (WSAdmin) können Sie auf Datenbank- und Sicherheitseigenschaften in IBM Business Process Manager zugreifen und diese modifizieren.

Der Ausdruck *Konfigurationsobjekt* bezeichnet ein Objekt, auf das mithilfe dieser WSAdmin-Bfehle zugegriffen wird. Weitere Informationen hierzu finden Sie in Sicherheitskonfigurationseigenschaften.

Implementierungsarchitektur

Die Implementierungsarchitektur von IBM Business Process Manager besteht aus Softwareprozessen (genannt 'Server'), aus topologischen Einheiten (genannt 'Knoten' und 'Zellen') sowie aus dem Konfigurationsrepository, das zum Speichern von Konfigurationsdaten verwendet wird.

Zellen

In IBM Business Process Manager bezeichnet der Begriff *Zellen* logische Gruppierungen von einem oder mehreren Knoten in einem verteilten Netz.

Eine Zelle ist ein Konfigurationskonzept, das Administratoren die Möglichkeit bietet, Knoten einander logisch zuzuordnen. Administratoren definieren die Knoten, die eine Zelle bilden, anhand spezieller Kriterien, die in den betreffenden Organisationsumgebungen sinnvoll sind.

Die Verwaltungskonfigurationsdaten werden in XML-Dateien gespeichert. Eine Zelle behält Hauptkonfigurationsdateien für jeden Server in jedem Knoten der Zelle bei. Jeder Knoten und jeder Server besitzen außerdem eigene lokale Konfigurationsdateien. Änderungen an einer lokalen Konfigurationsdatei für einen Knoten oder einen Server sind temporär, wenn der Server zur Zelle gehört. Während sie wirksam sind, überschreiben lokale Änderungen die Zellenkonfigurationen. Änderungen an den Konfigurationsdateien für den Hauptserver und den Hauptknoten, die auf Zellenebene vorgenommen werden, ersetzen alle auf Knotenebene vorgenommenen temporären Änderungen, wenn die Konfigurationsdokumente für die Zelle mit den Knoten synchronisiert werden. Die Synchronisation findet bei festgelegten Ereignissen statt, beispielsweise beim Starten eines Servers.

Server

Server stellen die zentralen Funktionen von IBM Business Process Manager bereit. Process Server erweitern die Funktionalität eines Anwendungsservers hinsichtlich der Handhabung von SCA-Modulen. Andere Server (Deployment Manager und Knotenagenten) werden zur Verwaltung von Prozess-Servern verwendet.

Bei einem Prozess-Server kann es sich entweder um einen *eigenständigen Server* oder einen *verwalteten Server* handeln. Ein verwalteter Server kann optional Member eines *Clusters* sein. Eine Gruppe von verwalteten Servern, Clustern mit Servern und weiterer Middleware wird als *Implementierungsumgebung* bezeichnet. In einer Implementierungsumgebung wird jeder verwaltete Server oder Cluster für eine bestimmte Funktion innerhalb der Implementierungsumgebung konfiguriert (z. B. Zielhost, Anwendungsmodulhost oder CIM-Server). Ein eigenständiger Server wird so konfiguriert, dass alle erforderlichen Funktionen bereitgestellt werden.

Server stellen die Laufzeitumgebung für SCA-Module, die von diesen Modulen verwendeten Ressourcen (Datenquellen, Aktivierungsspezifikationen und JMS-Ziele) und die von IBM gelieferten Ressourcen (Nachrichtenziele, Business Process Choreographer Container und CIM-Server) bereit).

Ein *Knotenagent* ist ein Verwaltungsagent, der einen Knoten in Ihrem System darstellt und die Server auf diesem Knoten verwaltet. Knotenagenten überwachen die Server auf einem Hostsystem und leiten Verwaltungsanforderungen an die Server weiter. Der Knotenagent wird erstellt, wenn ein Knoten in einen Deployment Manager eingebunden wird.

Ein *Deployment Manager* ist ein Verwaltungsagent, der eine zentrale Managementsicht für mehrere Server und Cluster bereitstellt.

Ein eigenständiger Server wird durch ein eigenständiges Profil definiert, ein Deployment Manager durch ein Deployment Manager-Profil. Verwaltete Server werden innerhalb eines *verwalteten Knotens* erstellt, der durch ein Profil für den verwalteten Knoten definiert wird.

Eigenständige Server

Ein eigenständiger Server stellt die Umgebung für die Implementierung von SCA-Modulen in einem Serverprozess bereit. Dieser Serverprozess umfasst unter anderem eine Administrationskonsole, ein Implementierungsziel, die Messaging-Unterstützung, den Business Process Rules Manager und einen Common Event Infrastructure-Server.

Ein eigenständiger Server ist einfach einzurichten und verfügt über eine Schnellstartkonsole, von der aus der Server gestartet und gestoppt und die Beispielsammlung und die Verwaltungskonsole geöffnet werden können. Wenn Sie die IBM Business Process Manager-Beispiele installieren und anschließend die Beispielgalerie öffnen, wird eine Beispiellösung auf dem eigenständigen Server implementiert. Sie können die Ressourcen für dieses Beispiel in der Administrationskonsole untersuchen.

Sie können zwar eigene Lösungen auf einem eigenständigen Server implementieren, jedoch kann ein solcher Server nicht die Kapazität, Skalierbarkeit und Zuverlässigkeit bieten, die in einer Produktionsumgebung gefordert werden. Für eine Produktionsumgebung ist eine Network Deployment-Umgebung die bessere Wahl.

Sie können mit einem eigenständigen Server beginnen und diesen später in eine Network Deployment-Umgebung aufnehmen, indem Sie ihn in eine Deployment Manager-Zelle einbinden. *Dies setzt voraus, dass noch keine anderen Knoten in diese Zelle eingebunden wurden*. Es ist nicht möglich, mehrere eigenständige Server in eine Zelle einzubinden. Sie können den eigenständigen Server mit der Administrationskonsole des Deployment Managers oder mit dem Befehl **addNode** einbinden. Der eigenständige Server darf nicht aktiv sein, wenn Sie ihn mit dem Befehl **addNode** einbinden.

Der eigenständige Server wird durch ein eigenständiges Serverprofil definiert.

Cluster

Cluster sind Gruppen von Servern, die zusammen verwaltet werden und am Workload-Management teilnehmen.

Ein Cluster kann Knoten oder einzelne Anwendungsserver enthalten. Ein Knoten ist in der Regel ein physischer Computer mit einer eindeutigen IP-Hostadresse, auf dem einer oder mehrere Anwendungsserver ausgeführt werden. Cluster können unter der Konfiguration einer Zelle, die viele Server und Cluster mit unterschiedlichen Konfigurationen und Anwendungen einander logisch zuordnet, nach Ermessen des Administrators und in einer für die Organisationsumgebungen sinnvollen Weise gruppiert werden.

Cluster sorgen für eine gleichmäßige Lastverteilung unter Servern. Server, die Teil eines Clusters sind, werden Cluster-Member genannt. Wenn Sie eine Anwendung in einem Cluster installieren, wird sie automatisch auf jedem Cluster-Member installiert.

Da jeder Cluster-Member dieselben Anwendungen enthält, können Sie Client-Tasks gemäß der Kapazität der unterschiedlichen Systeme verteilen, indem Sie jedem Server eine Gewichtung zuweisen.

Das Leistungsverhalten und der Ausweichbetrieb (Failover) werden durch die Zuweisung von Gewichtungen zu Servern in einem Cluster verbessert. Tasks werden Servern zugewiesen, die die Kapazität für die Ausführung der Taskoperationen besitzen. Falls einer der Server nicht in der Lage ist, die Task auszuführen, wird sie einem anderen Cluster-Member zugewiesen. Diese Funktionalität der erneuten Zuweisung bietet offensichtliche Vorteile gegenüber der Ausführung eines einzigen Anwendungsservers, der bei zu vielen Anforderungen überlastet sein kann.

Profil

Ein Profil definiert eine eigene Laufzeitumgebung mit separaten Befehls-, Konfigurations- und Protokolldateien. Profile definieren drei verschiedene Umgebungstypen auf IBM Business Process Manager-Systemen: eigenständiger Server, Deployment Manager und verwalteter Knoten.

Mit Profilen können Sie mehrere Laufzeitumgebungen auf einem System ausführen, ohne dazu mehrere Kopien der Binärdateien von IBM Business Process Manager installieren zu müssen.

Verwenden Sie zum Erstellen von IBM BPM-Profilen das Befehlszeilendienstprogramm BPMConfig. Das Befehlszeilendienstprogramm **manageprofiles** bzw. seine grafische Benutzerschnittstelle, das Profile Management Tool (PMT), lässt sich als alternative Methode zum Erstellen von Profilen für den Deployment Manager oder verwaltete Knoten verwenden. PMT wird für die Erstellung von eigenständigen Profilen nicht mehr unterstützt.

Anmerkung: Auf verteilten Plattformen besitzt jedes Profil einen eindeutigen Namen. Unter z/OS haben alle Profile den Namen „default"; es ist nicht möglich, unter z/OS Profile umzubenennen, zu bearbeiten, zu kopieren oder zu löschen.

Profiltypen

Folgende IBM BPM-Profiltypen sind mit IBM Business Process Manager V8.5 verfügbar:

IBM BPM - Eigenständiges Profil
Mit dem eigenständigen Profil werden eigenständige Server mit dem Leistungsspektrum und der Funktionalität definiert, die für IBM BPM Express-Konfigurationen charakteristisch sind. Sie können das eigenständige Profil mithilfe der Profilschablone `BPM/BpmServer` erstellen, die nur in IBM BPM Express unterstützt wird.

IBM BPM - Deployment Manager-Profil
Das Deployment Manager-Profil definiert einen Deployment Manager, der genau eine Verwaltungsschnittstelle für eine logische Gruppe von Servern auf einer oder mehreren Workstations bereitstellt. Sie können das Deployment Manager-Profil mithilfe der Profilschablone `BPM/BpmDmgr` erstellen, die nur in IBM BPM Standard und IBM BPM Advanced unterstützt wird.

IBM BPM - Profil für verwalteten Knoten
Das Profil für den verwalteten Knoten definiert einen verwalteten Knoten, sofern dieser Knoten in einen Deployment Manager eingebunden ist. Sie können das Profil für den verwalteten Knoten mithilfe der Profilschablone `BPM/BpmNode` erstellen, die nur in IBM BPM Standard und IBM BPM Advanced unterstützt wird.

Mit jeder IBM BPM-Konfiguration sind bestimmte Profiltypen verfügbar.

Tabelle 48. Verfügbare IBM BPM-Profiltypen

	Profiltypen		
IBM BPM Konfiguration	**Eigenständig**	**Deployment Manager**	**Verwalteter Knoten**
IBM BPMExpress	Ja	Nein	Nein
IBM BPM Standard	Nein	Ja	Ja
IBM BPM Advanced	Nein	Ja	Ja
Komponententestumgebung (Unit Test Environment, UTE) für IBM Integration Designer	Ja	Optional	Optional

Profilverzeichnis

Jedes Profil auf einem System besitzt ein eigenes Verzeichnis mit allen zugehörigen Dateien. Sie können die Position des Profilverzeichnisses bei der Erstellung des Profils festlegen. Standardmäßig wird das Verzeichnis `profiles` in dem Verzeichnis verwendet, in dem IBM Business Process Manager installiert ist. Beispiel: Das Profil Dmgr befindet sich im Verzeichnis `C:\Program Files\IBM\WebSphere\ProcServer\profiles\Dmgr`.

Standardprofil

Das erste Profil, das Sie in einer Installation von IBM Business Process Manager erstellen, ist das *Standardprofil*. Das Standardprofil ist das Standardziel für Befehle, die im Unterverzeichnis `bin` des Installationsverzeichnisses von IBM Business Process Manager eingegeben werden. Ist auf einem System nur ein Profil vorhanden, dann wird jeder Befehl für dieses Profil ausgeführt. Wenn Sie ein weiteres Profil erstellen, können Sie dieses zum Standardprofil machen.

Anmerkung: Das Standardprofil muss nicht zwangsläufig den Namen 'default' haben.

Profile erweitern

Wenn Sie bereits ein Deployment Manager-Profil, ein Profil für den verwalteten Knoten oder ein eigenständiges Serverprofil besitzen, das für WebSphere Application Server Network Deployment erstellt wurde, können Sie es *erweitern*, so dass es zusätzlich zu seiner bestehenden Funktion auch IBM Business Process Manager unterstützt. Wenn Sie ein Profil erweitern möchten, installieren Sie zunächst IBM Business Process Manager. Verwenden Sie anschließend das Befehlszeilendienstprogramm **manageprofiles**, um ein eigenständiges Profil zu erweitern, oder benutzen Sie wahlweise das Profile Management Tool oder das Befehlszeilendienstprogramm **manageprofiles**, um ein Profil für Deployment Manager-Profil oder einen erweiterten Knoten zu erweitern.

Wichtig: In einer Netzimplementierungsumgebung müssen Sie zunächst das Deployment Manager-Profil und anschließend die Profile von verwalteten Knoten erweitern.

Einschränkung: Sie können kein eigenständiges oder Deployment Manager-Profil erweitern, in dem die Standard-WebSphere VMM-Benutzerregistry geändert wurde, z.B. für die Verwendung von LDAP.

Deployment Manager

Ein Deployment Manager ist ein Server, der die Operationen für eine logische Gruppe anderer Server (Zelle) steuert. Der Deployment Manager ist ein zentraler Ort für die Verwaltung von Servern und Clustern.

Bei der Erstellung einer Implementierungsumgebung ist das Deployment Manager-Profil das erste Profil, das Sie erstellen. Jede Implementierungsumgebung, die Sie erstellen, verfügt über eine Schnellstartkonsole, von der aus der Deployment Manager gestartet und gestoppt und seine Administrationskonsole eingerichtet werden kann. Mit der Administrationskonsole des Deployment Managers können Sie die Server und Cluster in der Zelle steuern und verwalten. Sie können Server und Cluster konfigurieren, Server zu Clustern hinzufügen, Server und Cluster starten und stoppen und SCA-Module implementieren.

Obwohl es sich beim Deployment Manager prinzipiell um einen Server handelt, können Sie auf diesem keine Module implementieren.

Knoten

Ein *Knoten* ist eine logische Gruppierung von verwalteten Servern.

Ein Knoten entspricht in der Regel einem logischen oder physischen Computersystem mit einer eindeutigen IP-Hostadresse. Knoten können nicht mehrere Computer umfassen. Knotennamen sind gewöhnlich mit dem Hostnamen für den Computer identisch.

Knoten in einer Network Deployment-Topologie können verwaltet oder nicht verwaltet sein. Ein verwalteter Knoten hat einen Knotenagentenprozess, der die Konfiguration und die Server des Knotens verwaltet. Nicht verwaltete Knoten haben keinen Knotenagenten.

Verwaltete Knoten

Ein *verwalteter Knoten* ist ein Knoten, der in einen Deployment Manager eingebunden wurde, einen Knotenagenten enthält und verwaltete Server enthalten kann. Auf einem verwalteten Knoten können Sie verwaltete Server konfigurieren und ausführen.

Die Server, die in einem verwalteten Knoten konfiguriert sind, bilden die Ressourcen Ihrer Implementierungsumgebung. Diese Server werden in der Administrationskonsole des Deployment Managers erstellt, konfiguriert, gestartet, gestoppt, verwaltet und gelöscht.

Ein verwalteter Knoten hat einen Knotenagenten, der alle Server auf einem Knoten verwaltet.

Beim Einbinden eines Knotens wird automatisch ein Knotenagentenprozess erstellt. Dieser Knotenagent muss aktiv sein, um die Konfiguration des Profils verwalten zu können. Dazu zählen unter anderem die folgenden Tasks:

- Serverprozesse starten und stoppen
- Konfigurationsdaten auf dem Deployment Manager mit der Kopie auf dem Knoten synchronisieren

Der Knotenagent muss jedoch nicht aktiv sein, um Anwendungen auszuführen oder um Ressourcen auf dem Knoten zu konfigurieren.

Ein verwalteter Knoten kann einen oder mehrere Server enthalten, die von einem Deployment Manager verwaltet werden. Sie können auf den Servern in einem verwalteten Knoten Lösungen implementieren. Der verwaltete Knoten enthält jedoch keine Sammlung von Beispielanwendungen. Der verwaltete Knoten wird durch ein Profil für den verwalteten Knoten definiert und verfügt über eine Schnellstartkonsole.

Nicht verwaltete Knoten

Ein nicht verwalteter Knoten besitzt keinen Knotenagenten für die Verwaltung seiner Server.

Nicht verwaltete Knoten in der Network Deployment-Topologie können Serverdefinitionen wie zum Beispiel Webserver, aber keine Anwendungsserverdefinitionen besitzen. Nicht verwaltete Knoten können nicht in einem Verbund föderiert werden. Dies bedeutet, dass ein Knotenagent in keinem Fall zu einem nicht verwalteten Knoten hinzugefügt werden kann. Ein eigenständiger Server ist ein anderer Typ für einen nicht verwalteten Knoten. Der Deployment Manager kann diesen eigenständigen Server nicht verwalten, weil dieser der Zelle nicht bekannt ist. Ein eigenständiger Server kann in einem Verbund föderiert werden. Wenn er föderiert ist, wird automatisch ein Knotenagent erstellt. Der Knoten wird zu einem verwalteten Knoten in der Zelle.

Knotenagenten

Knotenagenten sind Verwaltungsagenten, die Verwaltungsanforderungen an Server weiterleiten.

Ein Knotenagent ist ein Server, der auf jedem Host-Computer-System ausgeführt wird, das Teil der Network Deployment-Konfiguration ist. Es handelt sich hierbei um einen reinen Verwaltungsagenten, der nicht in Anwendungsservingfunktionen einbezogen wird. Ein Knotenagent dient auch als Host für andere wichtige Verwaltungsfunktionen wie Dateiübertragungsservices, Konfigurationssynchronisation und Leistungsüberwachung.

Hinweise zur Benennung von Profilen, Knoten, Servern, Hosts und Zellen

Dieser Abschnitt enthält Informationen zu reservierten Begriffen sowie Hinweise, die Sie bei der Benennung von Profilen, Knoten, Servern, Hosts und Zellen (sofern zutreffend) berücksichtigen müssen. Dieser Abschnitt gilt für die verteilten Plattformen.

Hinweise zur Benennung von Profilen

Als Profilname kann mit folgenden Einschränkungen ein beliebiger eindeutiger Name verwendet werden. Verwenden Sie für Profilnamen keines der folgenden Zeichen:

- Leerzeichen
- Sonderzeichen, die im Namen von Verzeichnissen auf Ihrem Betriebssystem nicht zulässig sind. Beispiele: *, & oder ?
- Schrägstriche (/) oder umgekehrte Schrägstriche (\)

Doppelbytezeichen sind zulässig.

Hinweise zu Verzeichnispfaden: Der Pfad für das Installationsverzeichnis darf höchstens 60 Zeichen lang sein. Die Anzahl von Zeichen im Verzeichnispfad *profilverzeichnispfad**profilname* darf höchstens 80 Zeichen betragen.

Anmerkung: Verwenden Sie eine Namenskonvention mit kurzen Pfadnamen, wenn Sie ein Profil in einer Windows-Umgebung erstellen, um die Windows-Pfadlängenbegrenzung auf 255 Zeichen zu vermeiden.

Hinweise zur Benennung von Knoten, Servern, Hosts und Zellen

Reservierte Namen: Vermeiden Sie reservierte Namen als Feldwerte. Die Verwendung reservierter Namen kann zu unvorhersehbaren Ergebnissen führen. Die folgenden Wörter sind reserviert:

- cells
- nodes
- servers
- clusters
- applications
- deployments

Beschreibung der Felder auf der Seite mit den Namen für Knoten und Host und der Seite mit den Namen für Knoten, Host und Zelle: Beachten Sie beim Erstellen der Profile die entsprechenden Namensregeln.

- Eigenständige Serverprofile
- Deployment Manager-Profile
- Profile für verwaltete Knoten

Tabelle 49. Namensregeln für eigenständige Serverprofile

Feldname	Standardwert	Einschränkung	Beschreibung
Knotenname	*kurzname_des_hosts* Node *knotennummer*, wobei Folgendes gilt: • *kurzname_des_hosts* ist der Kurzname des Hosts. • *knotennummer* ist eine fortlaufende Zahl, die bei 01 beginnt.	Verwenden Sie keine reservierten Namen.	Wählen Sie einen beliebigen Namen aus. Zur besseren Organisation Ihrer Installation sollten Sie einen eindeutigen Namen verwenden, falls Sie mehr als einen Server auf dem System installieren möchten.

Tabelle 49. Namensregeln für eigenständige Serverprofile (Forts.)

Feldname	Standardwert	Einschränkung	Beschreibung
Servername	. . . server1	Verwenden Sie einen eindeutigen Namen für den Server.	Der logische Name für den Server.
Hostname	. . . Die Langform des DNS-Namens (DNS = Domain Name Server, Domänennamensserver).	Verwenden Sie einen vollständig qualifizierten, über Ihr Netz erreichbaren Hostnamen.	Verwenden Sie den echten DNS-Namen oder die IP-Adresse Ihrer Workstation, um die Kommunikation mit dieser Workstation zu ermöglichen. Weitere Informationen zum Hostnamen finden Sie im Anschluss an diese Tabelle.

Tabelle 50. Namensregeln für Deployment Manager-Profile

Feldname	Standardwert	Einschränkung	Beschreibung
Knotenname	. . . *kurzname_des_hosts* Cell Manager*knotennummer*, wobei Folgendes gilt: • *kurzname_des_hosts* ist der Kurzname des Hosts. • *knotennummer* ist eine fortlaufende Zahl, die bei 01 beginnt.	Verwenden Sie für den Deployment Manager einen eindeutigen Namen. Verwenden Sie keine reservierten Namen.	Der Name wird für die Verwaltung in der Deployment Manager-Zelle verwendet.
Hostname	. . . Die Langform des DNS-Namens (DNS = Domain Name Server, Domänennamensserver).	Verwenden Sie einen vollständig qualifizierten, über Ihr Netz erreichbaren Hostnamen. Verwenden Sie keine reservierten Namen.	Verwenden Sie den echten DNS-Namen oder die IP-Adresse Ihrer Workstation, um die Kommunikation mit dieser Workstation zu ermöglichen. Weitere Informationen zum Hostnamen finden Sie im Anschluss an diese Tabelle.

Tabelle 50. Namensregeln für Deployment Manager-Profile (Forts.)

Feldname	Standardwert	Einschränkung	Beschreibung
Zellenname	. . . *kurzname_des_hosts* Cell *zellennummer*, wobei Folgendes gilt: • *kurzname_des_hosts* ist der Kurzname des Hosts. • *zellennummer* ist eine fortlaufende Zahl, die bei 01 beginnt.	Verwenden Sie einen eindeutigen Namen für die Deployment Manager-Zelle. Zellennamen müssen generell immer eindeutig sein, wenn das Produkt auf der gleichen physischen Workstation oder in einem Workstation-Cluster (wie z. B. einem Sysplex) ausgeführt wird. Zusätzlich muss ein Zellenname in allen Situationen eindeutig sein, in denen die Netzkonnektivität zwischen Entitäten entweder zwischen den Zellen oder von einem Client erforderlich ist, der mit jeder der Zellen kommunizieren muss. Zellennamen müssen auch eindeutig sein, wenn deren Namensbereiche in einen Verbund eingebunden werden sollen. Andernfalls können Symptome wie Ausnahmebedingungen vom Typ javax.naming.NameNotFoundException auftreten, die das Erstellen von eindeutig benannten Zellen erforderlich machen.	Alle eingebundenen Knoten werden Elemente der Deployment Manager-Zelle, die Sie auf der Seite für die Knoten-, Host- und Zellennamen im Profile Management Tool angeben.

Tabelle 51. Namensregeln für Profile von verwalteten Knoten

Feldname	Standardwert	Einschränkung	Beschreibung
Knotenname	. . . *kurzname_des_hosts* Node *knotennummer*, wobei Folgendes gilt: • *kurzname_des_hosts* ist der Kurzname des Hosts. • *knotennummer* ist eine fortlaufende Zahl, die bei 01 beginnt.	Verwenden Sie keine reservierten Namen. Verwenden Sie in der Deployment Manager-Zelle einen eindeutigen Namen.	Der Name wird für die Verwaltung innerhalb der Deployment Manager-Zelle verwendet, zu der das Profil für den verwalteten Knoten hinzugefügt wird. Verwenden Sie in der Deployment Manager-Zelle einen eindeutigen Namen.
Hostname	. . . Die Langform des DNS-Namens (DNS = Domain Name Server, Domänennamensserver).	Verwenden Sie einen vollständig qualifizierten, über Ihr Netz erreichbaren Hostnamen.	Verwenden Sie den echten DNS-Namen oder die IP-Adresse Ihrer Workstation, um die Kommunikation mit dieser Workstation zu ermöglichen. Weitere Informationen zum Hostnamen finden Sie im Anschluss an diese Tabelle.

Hinweise zu Hostnamen:

Der Hostname ist der Netzname für die physische Workstation, auf der der Knoten installiert ist. Der Hostname muss auf dem Server in einen physischen Netzknoten aufgelöst werden. Bei einem Server mit mehreren Netzkarten muss der Hostname oder die IP-Adresse in eine der Netzkarten aufgelöst werden. Ferne Knoten verwenden den Hostnamen, um mit diesem Knoten zu kommunizieren.

IBM Business Process Manager ist sowohl mit dem Internetprotokoll der Version 4 (IPv4) als auch mit Version 6 (IPv6) kompatibel. Die Eingabe von IP-Adressen in der Administrationskonsole oder an anderen Stellen kann wahlweise in einem der beiden Formate erfolgen. Beachten Sie, dass die Eingabe von IP-Adressen im IPv6-Format erfolgen muss, wenn IPv6 auf Ihrem System bereits implementiert ist. Wenn IPv6 auf Ihrem System noch nicht verfügbar ist, müssen Sie IP-Adressen im IPv4-Format eingeben. Weitere Informationen zu IPv6 enthält die folgende Beschreibung: IPv6.

Die folgenden Richtlinien können helfen, den geeigneten Hostnamen für Ihre Workstation festzulegen:

- Wählen Sie einen Host aus, den andere Workstations in Ihrem Netz erreichen können.
- Verwenden Sie als Wert nicht die generische ID 'localhost'.
- Versuchen Sie nicht, IBM Business Process Manager-Produkte auf einem Server mit einem Host zu installieren, in dessen Namen Doppelbytezeichen verwendet werden. Doppelbytezeichen werden in dem Hostnamen nicht unterstützt.
- Verwenden Sie in Servernamen keine Unterstreichungszeichen (_). Internetstandards geben vor, dass die Domänennamen mit den Anforderungen an Hostnamen konform sein müssen, die in den Internet Official Protocol Standards RFC 952 und RFC 1123 beschrieben werden. Domänennamen dürfen nur Buchstaben (in Groß- oder Kleinschreibung) sowie Ziffern enthalten. Domänennamen dürfen auch Gedankenstriche (-) enthalten, solange diese nicht am Ende des Namens stehen. Unterstreichungszeichen (_) werden im Hostnamen nicht unterstützt. Wenn Sie IBM Business Process Manager auf einem Server installiert haben, in dessen Namen ein Unterstreichungszeichen vorkommt, können Sie auf diesen Server so lange mit der entsprechenden IP-Adresse zugreifen, bis Sie ihn umbenennen.

Wenn Sie koexistierende Knoten auf demselben Computer mit eindeutigen IP-Adressen definieren, dann definieren Sie jede IP-Adresse in einer DNS-Referenztabelle (DNS = Domänennamensserver). Konfigurationsdateien für Server stellen keine DN-Auflösung für mehrere IP-Adressen auf einer Workstation mit nur einer Netzadresse bereit.

Der Wert, den Sie für den Hostnamen angeben, wird in Konfigurationsdokumenten als Wert für das Merkmal 'hostName' verwendet. Geben Sie den Wert für den Hostnamen in einem der folgenden Formate an:

- Zeichenfolge für einen vollständig qualifizierten DNS-Hostnamen (DNS = Domänennamensserver), wie zum Beispiel xmachine.manhattan.ibm.com
- Zeichenfolge für den DNS-Hostnamen in seiner Standardkurzform, wie zum Beispiel xmachine
- Numerische IP-Adresse, wie zum Beispiel 127.1.255.3

Der vollständig qualifizierte DNS-Hostname hat den Vorteil, völlig eindeutig und trotzdem flexibel zu sein. Sie haben die Möglichkeit, die tatsächliche IP-Adresse für das Hostsystem zu ändern, ohne dabei die Konfiguration des Servers ändern zu müssen. Dieser Wert für den Hostnamen ist besonders dann nützlich, wenn Sie die IP-Adresse mithilfe des Dynamic Host Configuration Protocol (DHCP) häufig ändern möchten. Ein Nachteil dieses Formats besteht in seiner Abhängigkeit vom DNS. Ohne DNS ist die Konnektivität beeinträchtigt.

Der Kurzname für den Host ist dynamisch auflösbar. Ein Kurznamensformat bietet die zusätzliche Möglichkeit einer Definitionsänderung in der Datei für die lokalen Hosts, sodass das System auch dann mit dem Server arbeiten kann, wenn keine Verbindung mehr zum Netz besteht. Definieren Sie in der Datei für die Hosts den Wert '127.0.0.1' (lokales Loopback) für den Kurznamen, um die Ausführung bei getrennter Verbindung anzugeben. Ein Nachteil des Kurznamensformats besteht darin, dass für den Remotezugriff ein DNS erforderlich ist. Ohne DNS ist die Konnektivität beeinträchtigt.

Eine numerische IP-Adresse hat den Vorteil, dass keine Namensauflösung über DNS erforderlich ist. Ein ferner Knoten kann mit dem Knoten, den Sie mit einer numerischen IP-Adresse bezeichnen, auch dann verbunden werden, wenn kein DNS verfügbar ist. Ein Nachteil dieses Formats besteht darin, dass die numerische IP-Adresse festgelegt ist. Wenn Sie die IP-Adresse der Workstation ändern, müssen Sie auch die Einstellung für das Merkmal 'hostName' in den Konfigurationsdokumenten ändern. Verwenden Sie deshalb nicht die numerische IP-Adresse, wenn Sie DHCP verwenden oder IP-Adressen regelmäßig ändern. Ein weiterer Nachteil dieses Formats besteht darin, dass Sie den Knoten nicht verwenden können, wenn keine Verbindung zwischen Host und Netz besteht.

BPMN 2.0

IBM Business Process Manager-Geschäftsprozessdefinitionen unterstützen die Unterklasse 'Common Executable' der Konformitätsklasse für die BPMN 2.0-Prozessmodellierung, die sich mit ausführbaren Modellen beschäftigt.

BPMN (Business Process Model and Notation) ist der grundlegende Standard für die Prozesse in IBM Process Designer und IBM Process Center. BPD-Diagramme (BPD - Business Process Definition, Geschäftsprozessdefinition) basieren auf der BPMN-Spezifikation. Dieser Abschnitt enthält einige Beispiele für die Verwendung von BPMN 2.0 in IBM Business Process Manager. Detaillierte Informationen zu BPMN finden Sie auf der Seite zur BPMN-Spezifikation unter http://www.bpmn.org/.

IBM Business Process Manager unterstützt die folgenden BPMN 2.0-Tasktypen:

- Keine (abstrakte Task in der BPMN 2.0-Spezifikation)
- Systemtask (Service-Task in der BPMN 2.0-Spezifikation)
- Benutzertask
- Script
- Entscheidungstask (Geschäftsregeltask in der BPMN 2.0-Spezifikation)

IBM BPM Nachrichten-Zwischenereignisse bieten ähnliche Funktionen wie die BPMN-Sende- und Empfangstask.

BPMN 2.0-Notation

Ab Version 7.5.1 werden Process Designer BPMN 2.0-Tasksymbole in den BPD-Diagrammen in einer vereinfachten Palette erfasst und in Prozessdiagrammen angezeigt. Anhand der Symbole können Sie feststellen, ob es sich bei der jeweiligen Aktivität um eine Systemtask, eine Benutzertask, eine Entscheidungstask, ein Script oder einen verlinkten Prozess handelt. Aktivitäten in Modellen, die in früheren Versionen erstellt wurden, verfügen ebenfalls über entsprechende BPMN 2.0-Tasktypen und -Tasksymbole, wenn sie in Version 7.5.1 oder höher angezeigt werden.

Aktivitäten und Tasks

Gegenüber früheren Versionen von Process Designer wurden einige Terminologieänderungen vorgenommen. Einige dieser Änderungen beziehen sich auf Aktivitätstypen, die umbenannt wurden.

- Aktivitäten vom Typ 'Service (automatisiert)' sind nun Systemtasks.
- Aktivitäten vom Typ 'Service (Task)' in einem systemfremden Verantwortlichkeitsbereich sind nun Benutzertasks.
- Aktivitäten vom Typ 'Service (Task)' in einem Verantwortlichkeitsbereich des Systems sind nun Entscheidungstasks, wenn sie auf einen Entscheidungsservice verweisen.
- Aktivitäten vom Typ 'Service (Task)' in einem Verantwortlichkeitsbereich des Systems sind nun Systemtasks, wenn sie auf einen anderen Service als den Entscheidungsservice verweisen.
- Javascript-Aktivitäten sind nun Script-Tasks.
- Aktivitäten in verschachtelten Prozessen sind nun verlinkte Prozesse.

- Externe Aktivitäten aus früheren Versionen von Process Designer stehen als externe Implementierungen für Benutzer- und Systemtasks zur Verfügung.

Gateways

Für die Gateways früherer Versionen gibt es keine Notationsänderungen. Es gibt jedoch drei Terminologieänderungen. Das Entscheidungsgateway ist nun ein *exklusives Gateway*, das einfache Split- oder Join-Gateway ist nun ein *paralleles Gateway* und das bedingte Split- oder Join-Gateway ist nun ein *inklusives Gateway*.

Es gibt auch einen neuen Gateway-Typ, das *Ereignis-Gateway*. Ein Ereignis-Gateway stellt einen Verzweigungspunkt in einem Prozess dar, bei dem die alternativen Pfade, die dem Gateway folgen, auf stattfindenden Ereignissen und nicht auf der Auswertung von Ausdrücken basieren, die Prozessdaten verwenden (wie bei einem exklusiven oder inklusiven Gateway). Ein bestimmtes Ereignis (in der Regel der Empfang einer Nachricht) legt fest, welcher Pfad verwendet wird.

Nicht unterbrechende Ereignisse

Mit BPMN 2.0 wird eine Notation für nicht unterbrechende Ereignisse hinzufügt. Im Normalfall unterbricht ein Grenzereignis die Aktivität, der es zugeordnet ist. Beim Auslösen des Ereignisses wird die Aktivität gestoppt und das Token wird an den abgehenden Sequenzfluss des Ereignisses weitergeleitet. Wenn das Ereignis als nicht unterbrechendes Ereignis definiert ist, wird die zugeordnete Aktivität beim Auslösen des Ereignisses weiterhin parallel ausgeführt, und es wird ein neues Token generiert, das mit dem abgehenden Sequenzfluss des Ereignisses weitergeleitet wird. Die Ereignisgrenze wird in eine gestrichelte Linie für nicht unterbrechende Ereignisse geändert.

Zwischenereignisse, die Aktivitäten zugeordnet sind, sind unterbrechende Zwischenereignisse, wenn sie die ihnen zugeordnete Aktivität schließen, oder nicht unterbrechende Zwischenereignisse, wenn sie die ihnen zugeordnete Aktivität nicht schließen.

Startereignis

Laut BPMN-Spezifikation müssen Symbole für Start- und Endereignisse in Prozessmodellen nicht verwendet werden. Process Designer setzt die Verwendung von Start- und Stoppereignissen in Prozessmodellen voraus.

In Process Designer stehen die folgenden Typen von Startereignissen zur Verfügung:

Prozesse

- Nicht
- Nachricht
- Ad-hoc

Unterprozesse

- Nicht

Ereignisunterprozesse

- Fehler
- Nachricht
- Zeitgeber

Der Typ eines Startereignisses kann geändert werden, indem die Eigenschaften für das Ereignis bearbeitet werden. Sie können in einem Prozess zahlreiche Nachrichtenstartereignisse, aber nur ein einziges Nicht-Startereignis verwenden.

Endereignisse

Es stehen vier Typen von Endereignissen zur Verfügung: *Nachricht*, *Beendigung*, *Fehler* und *Nicht*. Der Typ eines Endereignisses kann geändert werden.

Wenn ein übergeordneter Prozess einen untergeordneten Prozess aufruft und der untergeordnete Prozess eine Beendigungsereignisaktion ausführt, muss der untergeordnete Prozess gestoppt werden, während der übergeordnete Prozess mit den nächsten Schritten fortfährt.

Unterprozesse

Die BPMN-Spezifikation definiert zwei Typen von Unterprozessen, nämlich eingebettete und wiederverwendbare Unterprozesse. In Process Designer können beide Typen erstellt werden. Eingebettete Unterprozesse werden in Process Designer einfach als *Unterprozesse* bezeichnet und sind in Version 7.5.1 neu. Ein wiederverwendbarer Unterprozess in BPMN ist ein *verlinkter Prozess* in Process Designer.

Unterprozesse sind in einem übergeordneten Prozess enthalten und stellen eine Möglichkeit zur Gruppierung von Prozessschritten dar, um Diagramme weniger komplex und unübersichtlich zu gestalten. Unterprozesse komprimieren mehrere Schritte zu einer Aktivität. Der Unterprozess ist nur für den Prozess sichtbar, in dem er definiert ist. Ein Unterprozess existiert innerhalb des Geltungsbereichs des jeweiligen aufrufenden Programms und hat Zugriff auf alle Variablen in dieser Umgebung. Von einem eingebetteten Unterprozess werden keine Parameter übergeben oder empfangen.

Neben dem Unterprozess und dem verlinkten Prozess verfügt Process Designer über einen Ereignisunterprozess, bei dem es sich um einen speziellen Unterprozess für die ereignisgesteuerte Verarbeitung handelt. Er ist nicht mit anderen Aktivitäten über den Sequenzfluss verbunden und tritt nur dann auf, wenn das zugehörige Startereignis ausgelöst wird.

Verlinkte Prozesse

Ein wiederverwendbarer Unterprozess in BPMN wird in Process Designer als *verlinkter Prozess* bezeichnet. Es handelt sich um einen Prozess, der außerhalb des aktuellen Prozesses erstellt wird und vom aktuellen Prozess aufgerufen werden kann. Er ist wiederverwendbar, weil andere Prozessdefinitionen diesen Prozess ebenfalls aufrufen können. Der verlinkte Prozess definiert die zugehörigen Eingabe- und Ausgabeparameter und verfügt über keinen Zugriff auf den Geltungsbereich oder die Umgebung des aufrufenden Programms. Der verlinkte Prozess ähnelt dem verschachtelten Prozess, der in früheren Versionen zur Verfügung stand; das Verhalten der Aktivität ist unverändert. Vorherige verschachtelte Prozesse werden in verlinkte Prozesse migriert. Der verlinkte Prozess sieht wie ein Unterprozess mit einer dicken Grenze aus und wird im Inspector-Fenster hervorgehoben angezeigt.

Schleifen

BPMN enthält das Konzept sich wiederholender Aktivitäten. Eine Aktivität kann atomar sein, was bedeutet, dass die Aktivität wiederholt wird. Eine Aktivität kann aber auch ein Unterprozess sein, in den eine Reihe von Schritten eingebunden sind, die wiederholt werden. Wenn Sie die sich wiederholende Aktivität erweitern, werden die darin enthaltenen Aktivitäten angezeigt, die wiederholt ausgeführt werden sollen. Die Bedingung wird immer zu Beginn einer Schleifenwiederholung ausgewertet. Eine Auswertung am Ende einer Schleifenwiederholung ist nicht möglich.

IBM Business Process Manager verfügt über eine *Schleife mit mehreren Instanzen*, die mit begrenzter Häufigkeit ausgeführt wird, wobei die Ausführung der darin enthaltenen Aktivitäten sequenziell oder parallel erfolgt.

Nicht-BPMN-Prozesse importieren

In IBM WebSphere Business Modeler erstellte Modelle können importiert und in Process Designer verwendet werden. Informationen zum BPMN 2.0-Import finden Sie unter Zuordnung von IBM WebSphere Business Modeler-Elementen zu IBM Business Process Manager-Konstrukten. In IBM WebSphere Business Compass, Rational Software Architect oder anderen Modellierungsumgebungen erstellte BPMN 2.0-Modelle können ebenfalls importiert werden.

Geschäftsprozessdefinitionen (BPDs)

Um einen Prozess in IBM Process Designer zu modellieren, müssen Sie eine Geschäftsprozessdefinition (Business Process Definition, BPD) erstellen. Die Geschäftsprozessdefinition kann auf einem importierten BPMN-Modell basieren.

Bei einer Geschäftsprozessdefinition handelt es sich um ein wiederverwendbares Prozessmodell, mit dessen Hilfe alle Gemeinsamkeiten von Laufzeitinstanzen dieses Prozessmodells definiert werden. Eine Geschäftsprozessdefinition muss ein Startereignis, ein Endereignis, mindestens eine Bahn und mindestens eine Aktivität enthalten. Detaillierte Informationen zu den Einschränkungen bezüglich der verwendbaren Zeichen bei Geschäftsprozessdefinitionen finden Sie in den zugehörigen Links unter 'IBM Process Designer - Namenskonventionen'.

Eine Geschäftsprozessdefinition muss eine Bahn für alle Systeme und Benutzergruppen enthalten, die an einem Prozess teilnehmen. Eine Bahn kann eine Teilnehmer- oder eine Systembahn sein. Sie haben allerdings die Möglichkeit, Geschäftsprozessdefinitionen zu erstellen, die die Aktivitäten einer Gruppe und eines Systems in einer einzigen Spur gruppieren. Informationen zur Erstellung von Geschäftsprozessdefinitionen finden Sie in den zugehörigen Links unter "Erstellen einer Geschäftsprozessdefinition (BPD)".

Sie können angeben, dass eine bestimmte Person oder Gruppe für die Aktivitäten in einer Teilnehmerbahn verantwortlich sind. Jede Bahn, die Sie erstellen, wird standardmäßig der Gruppe "Alle Benutzer" zugewiesen. Sie können diese Standardgruppe verwenden, um Ihre BPD im Inspector auszuführen und zu testen. Die Gruppe "Alle Benutzer" umfasst alle Benutzer, die Mitglied der Sicherheitsgruppe **tw_allusers** sind. Diese Gruppe ist eine spezielle Sicherheitsgruppe, die automatisch alle Benutzer im System einschließt.

Eine Systembahn enthält Aktivitäten, die von einem bestimmten IBM Process Center-System verarbeitet werden. Jede Aktivität benötigt eine Implementierung, die die Aktivität definiert und die Eigenschaften für die Task festlegt. Während der Implementierung erstellt der Entwickler einen Service oder schreibt das zur Ausführung der Aktivitäten in einer Systembahn erforderliche JavaScript. Informationen zu Services finden Sie unter "Informationen zu Servicetypen" in den zugehörigen Links.

Für jede erstellte Geschäftsprozessdefinition müssen Sie Variablen deklarieren, um die Geschäftsdaten, die in Ihrem Prozess von einer Aktivität zur nächsten übergeben werden, zu erfassen. Weitere Informationen zur Implementierung von Variablen finden Sie unter "Variablen verwalten und zuordnen" in den zugehörigen Links.

Sie können auch Ereignisse zu einer Geschäftsprozessdefinition hinzufügen. Ereignisse in IBM BPM können können durch das Verstreichen eines Fälligkeitsdatums, eine Ausnahmebedingung oder eine eingehende Nachricht ausgelöst werden. Der benötigte Auslöser bestimmt den Ereignistyp, den Sie implementieren. Detaillierte Informationen zu den verfügbaren Ereignistypen und den zugehörigen Auslösern finden Sie unter "Modellieren von Ereignissen".

Bindungen

Kernpunkt einer serviceorientierten Architektur ist das Konzept eines *Service*, also einer Funktionalitätseinheit, die durch eine Interaktion zwischen Datenverarbeitungsgeräten gebildet wird. Ein *Export* definiert die externe Schnittstelle (den Zugriffspunkt) eines Moduls, damit die SCA-Komponenten innerhalb des Moduls ihre Services für externe Clients bereitstellen können. Ein *Import* definiert eine Schnittstelle für modulexterne Services, damit die Services aus dem Modul heraus aufgerufen werden können. Sie verwenden protokollspezifische *Bindungen* bei Importen und Exporten, um anzugeben, mit welchem Transportmittel die Daten in das Modul oder aus dem Modul heraus transportiert werden.

Exporte

Externe Clients können SCA-Komponenten in einem Integrationsmodul über eine Vielzahl von Protokollen (z. B. HTTP, JMS, MQ und RMI/IIOP) mit Daten in vielfältigen Formaten wie beispielsweise XML, CSV, COBOL und JavaBeans) aufrufen. Exporte sind Komponenten, die diese Anforderungen von externen Quellen empfangen und dann unter Verwendung des SCA-Programmiermodells IBM Business Process Manager-Komponenten aufrufen.

Beispielsweise empfängt in der folgenden Abbildung ein Export von einer Clientanwendung eine Anforderung über das HTTP-Protokoll. Die Daten werden in ein Geschäftsobjekt transformiert. Dies ist das von der SCA-Komponente verwendete Format. Anschließend wird die Komponente mit diesem Datenobjekt aufgerufen.

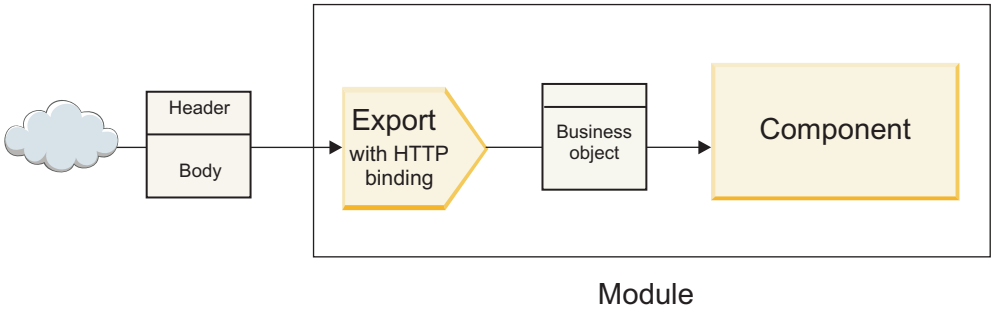


Abbildung 46. Export mit HTTP-Bindung

Importe

Es kann sein, dass eine SCA-Komponente einen externen Service aufrufen muss, der kein SCA-Service ist und Daten in einem anderen Format erwartet. Der externe Service wird von der SCA-Komponente unter Verwendung des SCA-Programmiermodells mit einem Import aufgerufen. Der Import ruft anschließend den Zielservice auf die Weise auf, die vom Service erwartet wird.

In der folgenden Abbildung wird beispielsweise eine Anforderung von einer SCA-Komponente durch den Import an einen externen Service gesendet. Das Geschäftsobjekt, also das von der SCA-Komponente erwartete Format, wird in das Format transformiert, das der Service erwartet, und der Service wird aufgerufen.

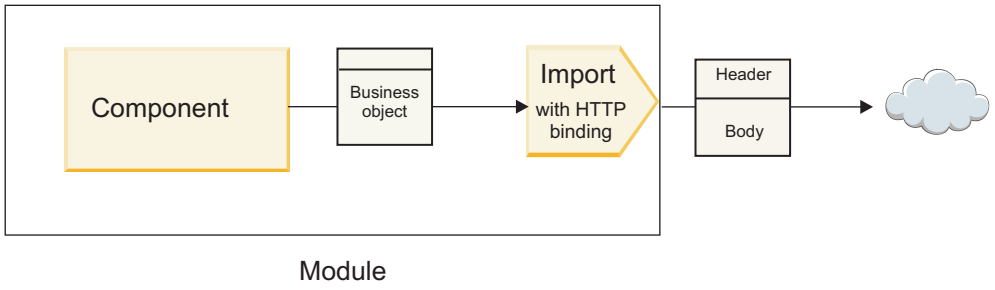


Abbildung 47. Import mit HTTP-Bindung

Liste der Bindungen

Mit Integration Designer können Sie eine Bindung für einen Import oder Export generieren und die Bindung konfigurieren. Die verfügbaren Bindungstypen sind in der folgenden Liste beschrieben.

- SCA

  Die SCA-Bindung, bei der es sich um die Standardbindung handelt, ermöglicht einem Service die Kommunikation mit Services in anderen SCA-Modulen. Einen Import mit SCA-Bindung verwenden Sie, um auf einen Service in einem anderen SCA-Modul zuzugreifen. Einen Export mit SCA-Bindung verwenden Sie, um einen Service für andere SCA-Module bereitzustellen.
- Web-Service

  Mit einer Web-Service-Bindung können Sie unter Verwendung von interoperablen SOAP-Nachrichten und Servicequalitäten auf einen externen Service zugreifen. Darüber hinaus können Sie mit Web-Service-Bindungen Anhänge als Teil der SOAP-Nachricht einbeziehen.

  Die Web-Service-Bindung kann das Übertragungsprotokoll SOAP/HTTP (SOAP über HTTP) oder SOAP/JMS (SOAP über JMS) verwenden. Unabhängig vom Transportprotokoll (HTTP oder JMS), mit dem die SOAP-Nachrichten übermittelt werden, verarbeiten Web-Service-Bindungen Anforderungs-/Antwortinteraktionen immer synchron.
- HTTP

  Mit der HTTP-Bindung können Sie mit dem HTTP-Protokoll auf einen externen Service zugreifen, wenn keine SOAP-Nachrichten verwendet werden oder ein direkter HTTP-Zugriff erforderlich ist. Diese Bindung wird verwendet, wenn Sie mit Web-Services arbeiten, die auf dem HTTP-Modell basieren, also anerkannte HTTP-Schnittstellenoperationen wie GET, PUT, DELETE usw. verwenden.
- Enterprise JavaBeans (EJB)

  Durch EJB-Bindungen können SCA-Komponenten mit Services interagieren, die durch Java EE-Geschäftslogik auf einem Java EEE-Server bereitgestellt werden.
- EIS

  Wenn die EIS-Bindung mit einem JCA-Ressourcenadapter verwendet wird, ermöglicht sie den Zugriff auf Services in einem unternehmensweiten Informationssystem (EIS) oder die Bereitstellung von Services für das EIS.
- JMS-Bindungen

  Java Message Service-Bindungen (JMS-Bindungen), generische JMS-Bindungen und WebSphere MQ-JMS-Bindungen (MQ-JMS-Bindungen) werden für Interaktionen mit Messaging-Systemen verwendet, wenn die asynchrone Übertragung über Nachrichtenwarteschlangen für die Zuverlässigkeit von wesentlicher Bedeutung ist.

  Ein Export mit einer der JMS-Bindungen überwacht eine Warteschlange auf das Eintreffen einer Nachricht und sendet die Antwort gegebenenfalls asynchron an die Antwortwarteschlange. Ein Import mit einer der JMS-Bindungen erstellt und sendet eine Nachricht an eine JMS-Warteschlange und überwacht eine Warteschlange auf das Eintreffen der Antwort (sofern vorhanden).
  - JMS

    Mit der JMS-Bindung können Sie auf den integrierten JMS-Provider von WebSphere zugreifen.

– Generische JMS

  Mit der generischen JMS-Bindung können Sie auf ein Messaging-System eines anderen Herstellers als IBM zugreifen.

– MQ-JMS

  Mit der MQ-JMS-Bindung können Sie auf die JMS-Untergruppe eines WebSphere MQ-Messaging-Systems zugreifen. Diese Bindung wird in der Regel verwendet, wenn die JMS-Untergruppe der Funktionen für eine Anwendung ausreichend ist.

- MQ

  Die WebSphere MQ-Bindung ermöglicht die Kommunikation mit nativen MQ-Anwendungen, indem diese in das Framework der serviceorientierten Architektur integriert werden und der Zugriff auf MQ-spezifische Headerinformationen ermöglicht wird. Diese Bindung wird normalerweise verwendet, wenn native MQ-Funktionen benötigt werden.

Export- und Importbindungen - Übersicht

Mit einem Export können Sie Services in einem Integrationsmodul für externe Clients verfügbar machen. Ein Import ermöglicht den SCA-Komponenten in einem Integrationsmodul das Aufrufen externer Services. Die Bindung, die dem Export oder Import zugeordnet ist, gibt die Beziehung zwischen Protokollnachrichten und Geschäftsobjekten an. Sie gibt außerdem an, wie Operationen und Fehler ausgewählt werden.

Informationsfluss in einem Export

Ein Export empfängt eine Anforderung, die für die mit dem Export verbundene Komponente bestimmt ist, über einen bestimmten Transport, der durch die zugeordnete Bindung festgelegt wird (z. B. HTTP).

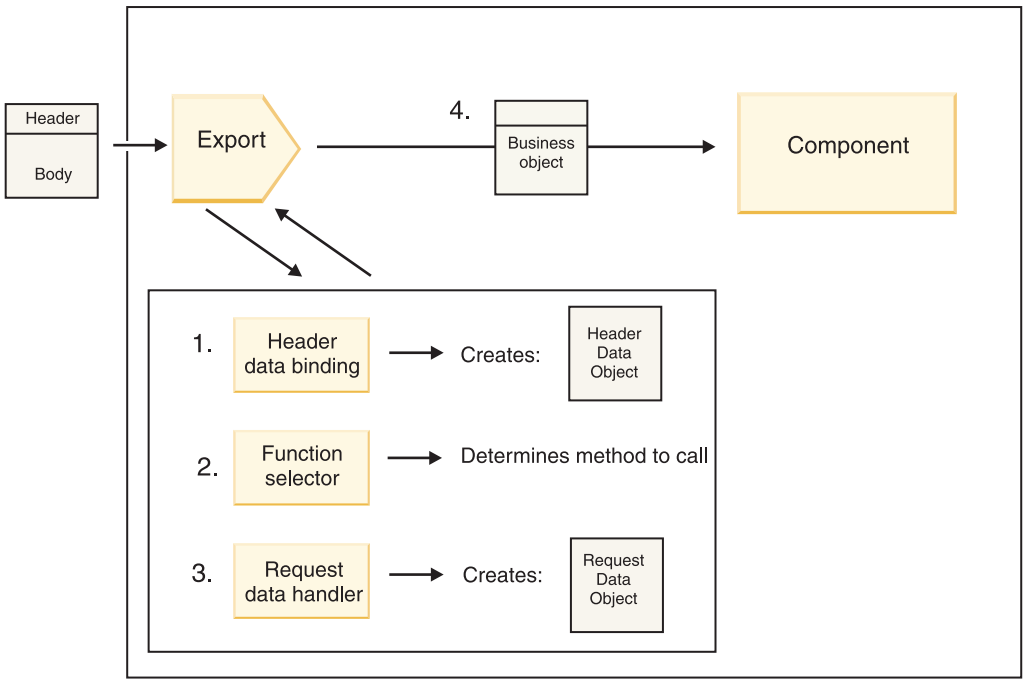


Abbildung 48. Fluss einer Anforderung durch den Export zu einer Komponente

Sobald der Export die Anforderung empfängt, findet die folgende Ereignissequenz statt:

1. Nur bei WebSphere MQ-Bindungen transformiert die Headerdatenbindung den Protokollheader in ein Headerdatenobjekt.

2. Der Funktionsselektor ermittelt den Namen der nativen Methode aus der Protokollnachricht. Der Name der nativen Methode wird durch die Exportkonfiguration zum Namen einer Operation für die Schnittstelle des Exports zugeordnet.
3. Der Anforderungsdatenhandler oder die Datenbindung für die Methode transformiert die Anforderung in ein Anforderungsgeschäftsobjekt.
4. Der Export ruft die Komponentenmethode mit dem Anforderungsgeschäftsobjekt auf.
   - Die HTTP-Exportbindung, die Web-Service-Exportbindung und die EJB-Exportbindung rufen die SCA-Komponente synchron auf.
   - Die generische JMS-, die JMS-, die MQ-JMS- und die WebSphere MQ-Exportbindung rufen die SCA-Komponente asynchron auf.

Bitte beachten Sie, dass ein Export die Header und Benutzereigenschaften, die er über das Protokoll empfängt, weitergeben kann, falls die Kontextweitergabe aktiviert ist. Komponenten, die mit dem Export verbunden sind, können dann auf diese Header und Benutzereigenschaften zugreifen. Weitere Informationen enthält das Thema über die Weiterleitung im Information Center von WebSphere Integration Developer.

Bei einer bidirektionalen Operation gibt die Komponente eine Antwort zurück.

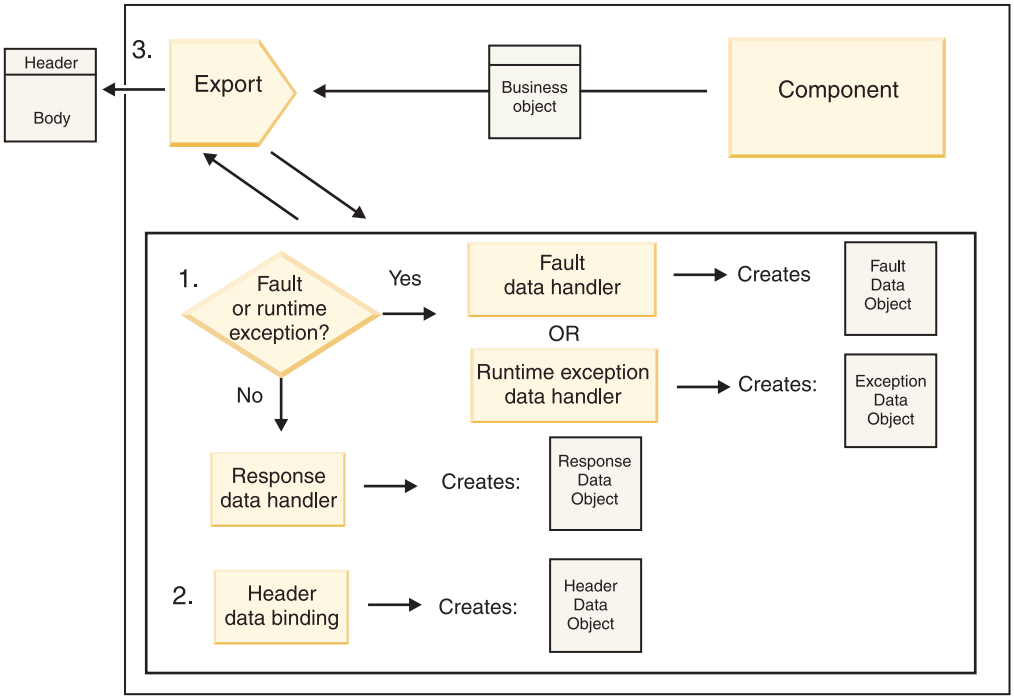


Abbildung 49. Rückfluss einer Antwort durch den Export

Die folgende Schrittsequenz findet statt:

1. Falls durch die Exportbindung eine normale Antwortnachricht empfangen wird, transformiert der Antwortdatenhandler oder die Datenbindung für die Methode das Geschäftsobjekt in eine Antwort.

   Handelt es sich bei der Antwort um einen Fehler, transformiert der Fehlerdatenhandler oder die Datenbindung für die Methode den Fehler in eine Fehlerantwort.

   Nur bei HTTP-Exportbindungen wird der Datenhandler für Laufzeitausnahmebedingungen (sofern konfiguriert) aufgerufen, falls die Antwort eine Laufzeitausnahmebedingung angibt.
2. Nur bei WebSphere MQ-Bindungen transformiert die Headerdatenbindung die Headerdatenobjekte in Protokollheader.
3. Der Export sendet die Serviceantwort über den Transport.

Informationsfluss in einem Import

Komponenten verwenden einen Import, um Anforderungen an modulexterne Services zu senden. Die Anforderung wird über einen bestimmten Transport gesendet, der durch die zugehörige Bindung festgelegt wird.

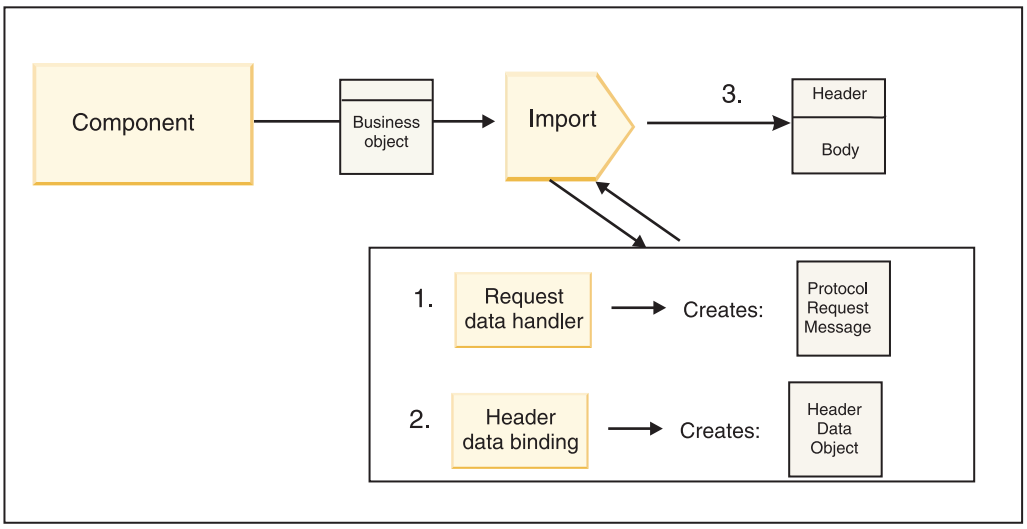


Abbildung 50. Fluss von einer Komponente über den Import zu einem Service

Die Komponente ruft den Import mit einem Anforderungsgeschäftsobjekt auf.

Anmerkung:

- Die HTTP-Importbindung, die Web-Service-Importbindung und die EJB-Importbindung sollten durch die aufrufende Komponente synchron aufgerufen werden.
- Die generische JMS-, die JMS-, die MQ-JMS- und die WebSphere MQ-Importbindung sollten asynchron aufgerufen werden.

Nachdem die Komponente den Import aufgerufen hat, findet die folgende Ereignissequenz statt:

1. Der Anforderungsdatenhandler oder die Datenbindung für die Methode transformiert das Anforderungsgeschäftsobjekt in eine Protokollanforderungsnachricht.
2. Nur bei WebSphere MQ-Bindungen legt die Headerdatenbindung für die Methode das Headerdatenobjekt im Protokollheader fest.
3. Der Import ruft den Service mit der Serviceanforderung über den Transport auf.

Bei einer bidirektionalen Operation gibt der Service eine Antwort zurück. Daraufhin findet die folgende Schrittsequenz statt:

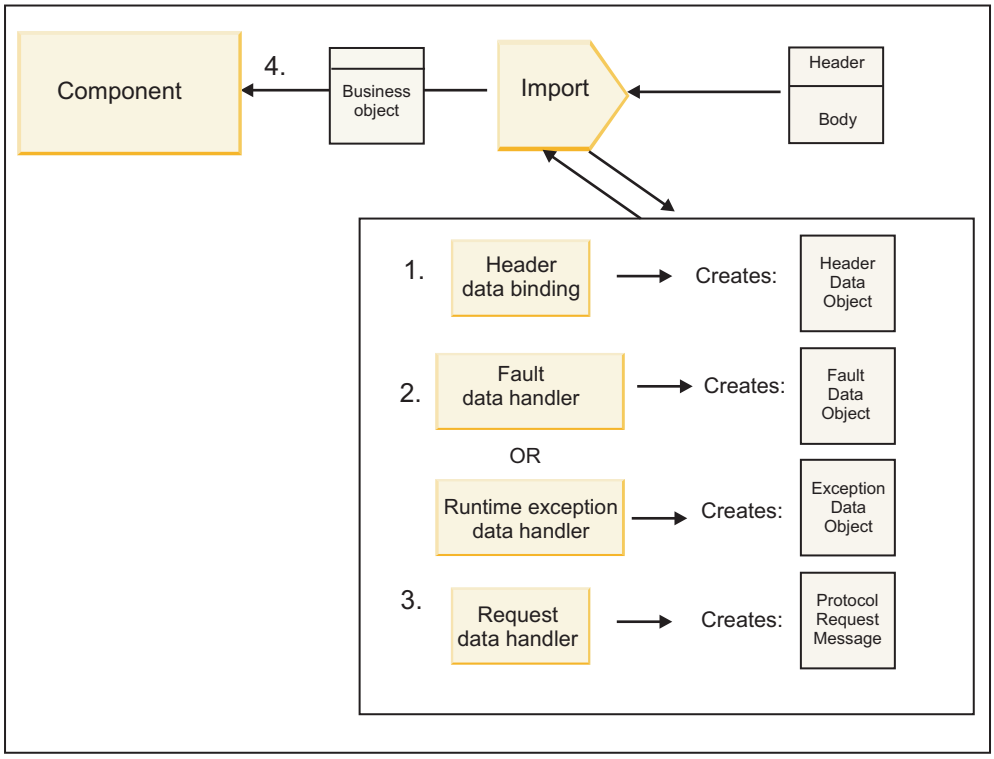


Abbildung 51. Rückfluss einer Antwort durch den Import

1. Nur bei WebSphere MQ-Bindungen transformiert die Headerdatenbindung den Protokollheader in ein Headerdatenobjekt.
2. Es wird ermittelt, ob es sich bei der Antwort um einen Fehler handelt.
   - Falls die Antwort einen Fehler angibt, überprüft der Fehlerselektor den Fehler und ermittelt, zu welchem WSDL-Fehler dieser zugeordnet ist. Der Fehlerdatenhandler für die Methode transformiert anschließend den Fehler in eine Fehlerantwort.
   - Falls die Antwort eine Laufzeitausnahmebedingung angibt, wird der Datenhandler für Laufzeitausnahmebedingungen (sofern konfiguriert) aufgerufen.
3. Der Antwortdatenhandler oder die Bindung für die Methode transformiert die Antwort in ein Antwortgeschäftsobjekt.
4. Der Import gibt das Antwortgeschäftsobjekt an die Komponente zurück.

Export- und Importbindungen - Konfiguration

Einer der wichtigsten Aspekte von Export- und Importbindungen ist die Datenformattransformation. Sie gibt an, wie Daten aus einem nativen Sendeformat zu einem Geschäftsobjekt zugeordnet (deserialisiert) werden oder wie sie aus einem Geschäftsobjekt zu einem nativen Sendeformat zugeordnet (serialisiert) werden. Bei Bindungen, die Exporten zugeordnet sind, können Sie außerdem durch die Auswahl eines Funktionsselektors angeben, welche Operation für die Daten ausgeführt werden soll. Bei Bindungen, die Exporten oder Importen zugeordnet sind, können Sie angeben, wie Fehler gehandhabt werden sollen, die während der Verarbeitung auftreten.

Darüber hinaus können Sie für Bindungen transportspezifische Informationen angeben. Für eine HTTP-Bindung geben Sie beispielsweise die Endpunkt-URL an. Die transportspezifischen Informationen für eine HTTP-Bindung sind in den Themen 'HTTP-Importbindung generieren' und 'HTTP-Exportbindung generieren' beschrieben. Im Information Center finden Sie außerdem Informationen zu anderen Bindungen.

Datenformattransformation bei Importen und Exporten

Wenn Sie eine Export- oder Importbindung in IBM Integration Designer konfigurieren, geben Sie unter anderem die Konfigurationseigenschaft für das von der Bindung verwendete Datenformat an.

- Für Exportbindungen, bei denen eine Clientanwendung Anforderungen an eine SCA-Komponente sendet und von dieser Antworten empfängt, geben Sie das Format der nativen Daten an. Je nach Format wählt das System den geeigneten Datenhandler bzw. die geeignete Datenbindung aus, um die nativen Daten in ein Geschäftsobjekt zu transformieren (das von der SCA-Komponente verwendet wird) und um im Gegenzug das Geschäftsobjekt in native Daten zu transformieren (die die Antwort an die Clientanwendung darstellen).
- Für Importbindungen, bei denen eine SCA-Komponente Anforderungen an einen modulexternen Service sendet und von diesem Antworten empfängt, geben Sie das Datenformat der nativen Daten an. Je nach Format wählt das System den geeigneten Datenhandler bzw. die geeignete Datenbindung aus, um das Geschäftsobjekt in native Daten zu transformieren (und umgekehrt).

IBM Business Process Manager bietet eine Reihe von vordefinierten Datenformaten und entsprechenden Datenhandlern oder Datenbindungen, die die Formate unterstützen. Sie können auch eigene benutzerdefinierte Datenhandler erstellen und das Datenformat für diese Datenhandler registrieren. Weitere Informationen enthält das Thema über die Entwicklung von Datenhandlern im Information Center von IBM Integration Designer.

- *Datenhandler* sind protokollneutral und transformieren Daten aus einem Format in ein anderes Format. In IBM Business Process Manager transformieren Datenhandler normalerweise native Daten (z. B. XML, CVS und COBOL) in ein Geschäftsobjekt sowie ein Geschäftsobjekt in native Daten. Weil Datenhandler protokollneutral sind, können Sie denselben Datenhandler bei einer Vielzahl von Export- und Importbindungen wiederverwenden. Sie können beispielsweise denselben XML-Datenhandler bei einer HTTP-Bindung für den Export oder Import oder bei einer JMS-Bindung für den Export oder Import verwenden.
- *Datenbindungen* transformieren ebenfalls native Daten in ein Geschäftsobjekt (und umgekehrt), sind jedoch protokollspezifisch. Eine HTTP-Datenbindung kann beispielsweise nur bei einer HTTP-Exportbindung oder einer HTTP-Importbindung verwendet werden. Im Gegensatz zu Datenhandlern kann eine HTTP-Datenbindung nicht bei einer MQ-Exportbindung oder -Importbindung wiederverwendet werden.

  Anmerkung: Bei Version 7.0 von IBM Business Process Manager werden drei HTTP-Datenbindungen (HTTPStreamDataBindingSOAP, HTTPStreamDataBindingXML und HTTPServiceGatewayDataBinding) nicht mehr unterstützt. Verwenden Sie nach Möglichkeit immer Datenhandler.

Wie bereits erwähnt, können Sie bei Bedarf benutzerdefinierte Datenhandler erstellen. Sie können ebenfalls benutzerdefinierte Datenbindungen erstellen. Es empfiehlt sich jedoch, benutzerdefinierte Datenhandler zu erstellen, da diese für mehrere Bindungen verwendet werden können.

Datenhandler:

Datenhandler werden für Export- und Importbindungen konfiguriert, um Daten auf protokollneutrale Weise von einem Format in ein anderes Format zu transformieren. Im Rahmen des Produkts werden mehrere Datenhandler bereitgestellt. Bei Bedarf können Sie aber auch einen eigenen Datenhandler erstellen. Sie können einen Datenhandler einer Export- oder Importbindung auf einer von zwei Ebenen zuordnen. Entweder ordnen Sie ihn allen Operationen in der Schnittstelle des Exports oder Imports zu oder einer bestimmten Operation für die Anforderung bzw. Antwort.

Vordefinierte Datenhandler

Sie verwenden IBM Integration Designer, um den Datenhandler anzugeben, den Sie verwenden wollen.

Die vordefinierten Datenhandler sind in der folgenden Tabelle aufgeführt. Außerdem ist dort beschrieben, wie jeder Datenhandler eingehende und abgehende Daten transformiert.

Anmerkung: Wenn nicht explizit eine Ausnahme angegeben ist, können diese Datenhandler bei JMS-, generischen JMS-, MQ-JMS-, WebSphere MQ- und HTTP-Bindungen verwendet werden.
Detailinformationen enthält das Thema 'Datenhandler' im Information Center von Integration Designer.

Tabelle 52. Vordefinierte Datenhandler

Datenhandler	Native Daten in Geschäftsobjekt	Geschäftsobjekt in native Daten
ATOM	Führt eine Syntaxanalyse für Atom-Feeds durch und stellt sie in ein Geschäftsobjekt für Atom-Feeds.	Serialisiert ein Geschäftsobjekt für Atom-Feeds in Atom-Feeds.
Mit Begrenzer	Führt eine Syntaxanalyse von Daten mit Begrenzer durch und stellt sie in ein Geschäftsobjekt.	Serialisiert ein Geschäftsobjekt in Daten mit Begrenzer (inklusive CVS).
Feste Breite	Führt eine Syntaxanalyse von Daten mit fester Breite durch und stellt sie in ein Geschäftsobjekt.	Serialisiert ein Geschäftsobjekt in Daten mit fester Breite.
Verarbeitung durch WTX	Delegiert die Datenformattransformation an WebSphere Transformation Extender (WTX). Der WTX-Zuordnungsname wird durch den Datenhandler abgeleitet.	Delegiert die Datenformattransformation an WebSphere Transformation Extender (WTX). Der WTX-Zuordnungsname wird durch den Datenhandler abgeleitet.
Verarbeitung durch WTX Invoker	Delegiert die Datenformattransformation an WebSphere Transformation Extender (WTX). Der WTX-Zuordnungsname wird durch den Benutzer angegeben.	Delegiert die Datenformattransformation an WebSphere Transformation Extender (WTX). Der WTX-Zuordnungsname wird durch den Benutzer angegeben.
JAXB	Serialisiert Java-Beans in ein Geschäftsobjekt unter Verwendung der Zuordnungsregeln, die in der Spezifikation von Java Architecture for XML Binding (JAXB) definiert sind.	Deserialisiert ein Geschäftsobjekt in Java-Beans unter Verwendung der Zuordnungsregeln, die in der JAXB-Spezifikation definiert sind.
JAXWS **Anmerkung:** Der JAXWS-Datenhandler kann nur bei einer EJB-Bindung verwendet werden.	Wird von einer EJB-Bindung für die Transformation eines Java-Antwortobjekts oder eines Java-Ausnahmeobjekts in ein Antwortgeschäftsobjekt verwendet. Hierbei kommen die Zuordnungsregeln zum Einsatz, die in der Java API for XML Web Services (JAX-WS)-Spezifikation definiert sind.	Wird von einer EJB-Bindung für die Transformation eines Geschäftsobjekts in die abgehenden Java-Methodenparameter verwendet. Hierbei kommen die Zuordnungsregeln zum Einsatz, die in der JAX-WS-Spezifikation definiert sind.
JSON	Führt eine Syntaxanalyse von JSON-Daten durch und stellt sie in ein Geschäftsobjekt.	Serialisiert ein Geschäftsobjekt in JSON-Daten.
Nativer Hauptteil	Führt eine Syntaxanalyse der nativen Byte, des nativen Textes, der nativen Zuordnung, des nativen Datenstroms oder des nativen Objekts durch und stellt sie in eines von fünf Basisgeschäftsobjekten (Text, Byte, Zuordnung, Datenstrom oder Objekt).	Transformiert die fünf Basis-Geschäftsobjekte in Byte, Text, Zuordnung, Datenstrom oder Objekt.
SOAP	Führt eine Syntaxanalyse der SOAP-Nachricht (und des Headers) durch und stellt sie in ein Geschäftsobjekt.	Serialisiert ein Geschäftsobjekt in eine SOAP-Nachricht.

Tabelle 52. Vordefinierte Datenhandler (Forts.)

Datenhandler	Native Daten in Geschäftsobjekt	Geschäftsobjekt in native Daten
XML	Führt eine Syntaxanalyse von XML-Daten durch und stellt sie in ein Geschäftsobjekt.	Serialisiert ein Geschäftsobjekt in XML-Daten.
UTF8XMLDataHandler	Führt eine Syntaxanalyse von in UTF-8 codierten XML-Daten durch und stellt sie in ein Geschäftsobjekt.	Serialisiert ein Geschäftsobjekt beim Senden einer Nachricht in XML-Daten, die in UTF-8 codiert sind.

Datenhandler erstellen

Ausführliche Informationen zum Erstellen eines Datenhandlers enthält das Thema über die Entwicklung von Datenhandlern im Information Center von Integration Designer.

Datenbindungen:

Datenbindungen werden für Export- und Importbindungen konfiguriert, um Daten von einem Format in ein anderes Format zu transformieren. Datenbindungen sind protokollspezifisch. Im Rahmen des Produkts werden mehrere Datenbindungen bereitgestellt. Bei Bedarf können Sie aber auch eine eigene Datenbindung erstellen. Sie können eine Datenbindung einer Export- oder Importbindung auf einer von zwei Ebenen zuordnen. Entweder ordnen Sie sie allen Operationen in der Schnittstelle des Exports oder Imports zu oder einer bestimmten Operation für die Anforderung bzw. Antwort.

Sie verwenden IBM Integration Designer, um die zu verwendende Datenbindung anzugeben oder um eine eigene Datenbindung zu erstellen. Eine Erläuterung dazu, wie Datenbindungen erstellt werden, finden Sie im Abschnitt mit der Übersicht über JMS-, MQ-JMS- und generische JMS-Bindungen im Information Center von IBM Integration Designer.

JMS-Bindungen

In der folgenden Tabelle sind die Datenbindungen aufgeführt, die bei den folgenden Bindungen verwendet werden können:

- JMS-Bindungen
- Generische JMS-Bindungen
- WebSphere MQ-JMS-Bindungen

Außerdem enthält die Tabelle eine Beschreibung der Tasks, die von der Datenbindung ausgeführt werden.

Tabelle 53. Vordefinierte Datenbindungen für JMS-Bindungen

Datenbindung	Native Daten in Geschäftsobjekt	Geschäftsobjekt in native Daten
Serialisiertes Java-Objekt	Transformiert das serialisierte Java-Objekt in ein Geschäftsobjekt (das in WSDL als Eingabe- oder Ausgabetyp zugeordnet ist).	Serialisiert ein Geschäftsobjekt in das serialisierte Java-Objekt in der JMS-Objektnachricht.
Eingeschlossene Byte	Extrahiert die Byte aus der eingehenden JMS-Bytenachricht und schließt sie in das Geschäftsobjekt 'JMSBytesBody' ein.	Extrahiert die Byte aus dem Geschäftsobjekt 'JMSBytesBody' und schließt sie in die abgehende JMS-Bytenachricht ein.

Tabelle 53. Vordefinierte Datenbindungen für JMS-Bindungen (Forts.)

Datenbindung	Native Daten in Geschäftsobjekt	Geschäftsobjekt in native Daten
Eingeschlossener Zuordnungseintrag	Extrahiert die Informationen zu Name, Wert und Typ für jeden Eintrag in der eingehenden JMS-Zuordnungsnachricht und erstellt eine Liste mit Geschäftsobjekten 'MapEntry'. Anschließend wird die Liste in das Geschäftsobjekt 'JMSMapBody' eingeschlossen.	Extrahiert die Informationen zu Name, Wert und Typ aus der Liste 'MapEntry' im Geschäftsobjekt 'JMSMapBody' und erstellt die korrespondierenden Einträge in der abgehenden JMS-Zuordnungsnachricht.
Eingeschlossenes Objekt	Extrahiert das Objekt aus der eingehenden JSM-Objektnachricht und schließt es in das Geschäftsobjekt 'JMSObjectBody' ein.	Extrahiert das Objekt aus dem Geschäftsobjekt 'JMSObjectBody' und schließt es in die abgehende JMS-Objektnachricht ein.
Eingeschlossener Text	Extrahiert den Text aus der eingehenden JSM-Textnachricht und schließt ihn in das Geschäftsobjekt 'JMSTextBody' ein.	Extrahiert den Text aus dem Geschäftsobjekt 'JMSTextBody' und schließt ihn in die abgehende JMS-Textnachricht ein.

WebSphere MQ-Bindungen

In der folgenden Tabelle sind die Datenbindungen aufgeführt, die mit WebSphere MQ verwendet werden können. Außerdem sind die von den Datenbindungen ausgeführten Tasks beschrieben.

Tabelle 54. Vordefinierte Datenbindungen für WebSphere MQ-Bindungen

Datenbindung	Native Daten in Geschäftsobjekt	Geschäftsobjekt in native Daten
Serialisiertes Java-Objekt	Transformiert das serialisierte Java-Objekt aus der eingehenden Nachricht in ein Geschäftsobjekt (das in WSDL als Eingabe- oder Ausgabetyp zugeordnet ist).	Transformiert ein Geschäftsobjekt in das serialisierte Java-Objekt in der ausgehenden Nachricht.
Eingeschlossene Byte	Extrahiert die Byte aus der unstrukturierten MQ-Bytenachricht und schließt sie in das Geschäftsobjekt JMSBytesBody ein.	Extrahiert die Byte aus dem Geschäftsobjekt JMSBytesBody und schließt die Byte in die ausgehende unstrukturierte MQ-Bytenachricht ein.
Eingeschlossener Text	Extrahiert den Text aus einer unstrukturierten MQ-Textnachricht und schließt ihn in ein Geschäftsobjekt 'JMSTextBody' ein.	Extrahiert den Text aus einem Geschäftsobjekt 'JMSTextBody' und schließt ihn eine unstrukturierte MQ-Textnachricht ein.
Eingeschlossener Datenstromeintrag	Extrahiert die Informationen zu Name und Typ für jeden Eintrag in der eingehenden JMS-Datenstromnachricht und erstellt eine Liste der Geschäftsobjekte 'StreamEntry'. Anschließend wird die Liste in das Geschäftsobjekt 'JMSStreamBody' eingeschlossen.	Extrahiert die Informationen zu Name und Typ aus der Liste 'StreamEntry' im Geschäftsobjekt 'JMSStreamBody' und erstellt die korrespondierenden Einträge in der abgehenden JMS-Datenstromnachricht.

Neben den in Tabelle 25 auf Seite 63 aufgelisteten Datenbindungen verwendet WebSphere MQ außerdem Headerdatenbindungen. Ausführliche Informationen hierzu finden Sie im Information Center von IBM Integration Designer.

HTTP-Bindungen

In der folgenden Tabelle sind die Datenbindungen aufgeführt, die mit HTTP verwendet werden können. Außerdem sind die Tasks beschrieben, die die Datenbindungen ausführen.

Tabelle 55. Vordefinierte Datenbindungen für HTTP-Bindungen

Datenbindung	Native Daten in Geschäftsobjekt	Geschäftsobjekt in native Daten
Eingeschlossene Byte	Extrahiert die Byte aus der eingehenden HTTP-Nachricht und schließt sie in das Geschäftsobjekt 'HTTPBytes' ein.	Extrahiert die Byte aus dem Geschäftsobjekt 'HTTPBytes' und fügt sie zum Hauptteil der abgehenden HTTP-Nachricht hinzu.
Eingeschlossener Text	Extrahiert den Text aus dem Hauptteil der eingehenden HTTP-Nachricht und schließt ihn in das Geschäftsobjekt 'HTTPText' ein.	Extrahiert den Text aus dem Geschäftsobjekt 'HTTPText' und fügt ihn zum Hauptteil der abgehenden HTTP-Nachricht hinzu.

Funktionsselektoren in Exportbindungen

Mit einem Funktionsselektor wird angegeben, welche Operation für die Daten einer Anforderungsnachricht ausgeführt werden soll. Funktionsselektoren werden im Rahmen einer Exportbindung konfiguriert.

Dies soll am Beispiel eines SCA-Exports veranschaulicht werden, der eine Schnittstelle zugänglich macht. Die Schnittstelle enthält die beiden Operationen 'Create' und 'Update'. Der Export besitzt eine JMS-Bindung, die Daten aus einer Warteschlange liest.

Wenn in der Warteschlange eine Nachricht eintrifft, werden an den Export die zugehörigen Daten übergeben. Es muss jedoch ermittelt werden, welche Operation aus der Schnittstelle des Exports für die verbundene Komponente aufgerufen werden muss. Die Operation wird durch den Funktionsselektor und durch die Konfiguration der Exportbindung bestimmt.

Der Funktionsselektor gibt den nativen Funktionsnamen zurück, also den Funktionsnamen im Clientsystem, das die Nachricht gesendet hat. Der native Funktionsname wird dann dem Operations- oder Funktionsnamen in der Schnittstelle zugeordnet, die dem Export zugeordnet ist. In der folgenden Abbildung gibt beispielsweise der Funktionsselektor den nativen Funktionsnamen (CRT) aus der eingehenden Nachricht zurück. Der native Funktionsname wird der Operation 'Create' zugeordnet und das Geschäftsobjekt wird mit der Operation 'Create' an die SCA-Komponente gesendet.

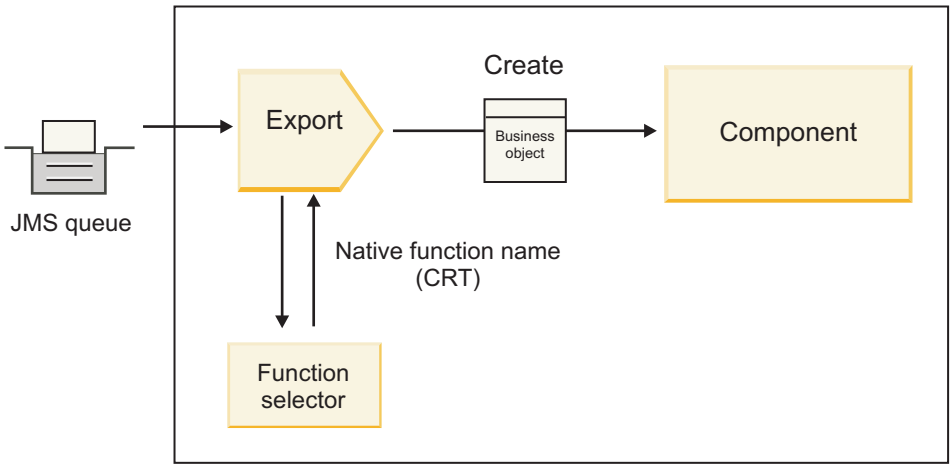


Abbildung 52. Funktionsselektor

Falls die Schnittstelle lediglich mit einer einzigen Operation ausgestattet ist, ist die Angabe eines Funktionsselektors nicht erforderlich.

Mehrere vordefinierte Funktionsselektoren sind verfügbar. Sie sind in den nachfolgenden Abschnitten aufgelistet.

JMS-Bindungen

In der nachstehenden Tabelle sind die Funktionsselektoren aufgeführt, die bei den folgenden Bindungen verwendet werden können:

- JMS-Bindungen
- Generische JMS-Bindungen
- WebSphere MQ-JMS-Bindungen

Tabelle 56. Vordefinierte Funktionsselektoren für JMS-Bindungen

Funktionsselektor	Beschreibung
JMS-Funktionsselektor für einfache JMS-Datenbindungen	Verwendet die Eigenschaft 'JMSType' der Nachricht, um den Operationsnamen auszuwählen.
Funktionsselektor für JMS-Headereigenschaft	Gibt den Wert der JMS-Zeichenfolgeeigenschaft 'TargetFunctionName' aus dem Header zurück.
Funktionsselektor für JMS-Service-Gateway	Ermittelt, ob es sich bei der Anforderung um eine unidirektionale oder um eine bidirektionale Operation handelt, indem die vom Client festgelegte Eigenschaft 'JMSReplyTo' untersucht wird.

WebSphere MQ-Bindungen

In der folgenden Tabelle sind die Funktionsselektoren aufgeführt, die bei WebSphere MQ-Bindungen verwendet werden können:

Tabelle 57. Vordefinierte Funktionsselektoren für WebSphere MQ-Bindungen

Funktionsselektor	Beschreibung
MQ-Funktionsselektor für 'handleMessage'	Gibt 'handleMessage' als einen Wert zurück, der mithilfe der Exportmethodenbindungen dem Namen einer Operation in der Schnittstelle zugeordnet wird.
MQ verwendet den JMS-Standardfunktionsselektor	Liest die native Operation aus der Eigenschaft 'TargetFunctionName' des Ordners eines MQRFH2-Headers.
MQ verwendet das Format des Nachrichtenhauptteils als native Funktion	Sucht nach dem Formatfeld des letzten Headers und gibt dieses Feld als Zeichenfolge zurück.
MQ-Funktionsselektor für 'type'	Erstellt in der Exportbindung eine Methode, indem eine URL abgerufen wird, die die Eigenschaften 'Msd', 'Set', 'Type' und 'Format' enthält, die im MQRFH2-Header gefunden wurden.
Funktionsselektor für MQ-Service-Gateway	Verwendet die Eigenschaft 'MsgType' im MQMD-Header, um den Operationsnamen zu ermitteln.

HTTP-Bindungen

In der folgenden Tabelle sind die Funktionsselektoren aufgeführt, die bei HTTP-Bindungen verwendet werden können:

Tabelle 58. Vordefinierte Funktionsselektoren für HTTP-Bindungen

Funktionsselektor	Beschreibung
HTTP-Funktionsselektor auf der Basis des Headers 'TargetFunctionName'	Verwendet die HTTP-Headereigenschaft 'TargetFunctionName' aus dem Client, um zu ermitteln, welche Operation zur Laufzeit aus dem Export aufgerufen werden soll.
HTTP-Funktionsselektor auf der Basis der URL und der HTTP-Methode	Verwendet den relativen Pfad der URL und hängt die HTTP-Methode des Clients an, um die native Operation festzustellen, die für den Export definiert ist.
Funktionsselektor für HTTP-Service-Gateway auf der Basis einer URL mit einem Operationsnamen	Ermittelt die aufzurufende Methode auf Basis der URL, falls 'perationMode = oneway' an die Anforderungs-URL angehängt worden ist.

Anmerkung: Mit IBM Integration Designer können Sie auch einen eigenen Funktionsselektor erstellen. Informationen zum Erstellen eines Funktionsselektors enthält das Information Center von IBM Integration Designer. Die Erstellung eines Funktionsselektors ist beispielsweise in der Übersicht über die MQ-Funktionsselektoren beschrieben.

Fehlerbehandlung

Sie können Import- und Exportbindungen für den Umgang mit Fehlern (zum Beispiel Geschäftsausnahmebedingungen) konfigurieren, die während der Verarbeitung auftreten, indem Sie Fehlerdatenhandler angeben. Ein Fehlerdatenhandler kann auf drei Ebenen konfiguriert werden: Sie können einen Fehlerdatenhandler einem Fehler, einer Operation oder allen Operationen mit einer Bindung zuordnen.

Ein Fehlerdatenhandler verarbeitet Fehlerdaten und transformiert sie in das korrekte Format, das durch die Export- oder Importbindung gesendet werden muss.

- Bei einer Exportbindung transformiert der Fehlerdatenhandler das Ausnahmebedingungsgeschäftsobjekt, das von der Komponente gesendet wurde, in eine Antwortnachricht, die von der Clientanwendung verwendet werden kann.
- Bei einer Importbindung transformiert der Fehlerdatenhandler die Fehlerdaten oder die Antwortnachricht, die von einem Service gesendet wurden, in ein Ausnahmebedingungsgeschäftsobjekt, das von der SCA-Komponente verwendet werden kann.

Bei Importbindungen ruft die Bindung den Fehlerselektor auf. Dieser ermittelt, ob die Antwortnachricht eine normale Antwort, eine Geschäftsausnahmebedingung oder eine Laufzeitausnahmebedingung darstellt.

Sie können einen Fehlerdatenhandler für einen bestimmten Fehler, für eine Operation und für alle Operationen mit einer Bindung zuordnen.

- Falls der Fehlerdatenhandler auf allen drei Ebenen festgelegt ist, wird der Datenhandler aufgerufen, der einem bestimmten Fehler zugeordnet ist.
- Sind Fehlerdatenhandler auf Operations- und Bindungsebene festgelegt, wird der Datenhandler aufgerufen, der der Operation zugeordnet ist.

Zur Angabe der Fehlerbehandlung werden zwei Editoren in IBM Integration Designer verwendet. Mit dem Schnittstelleneditor wird angegeben, ob in einer Operation ein Fehler auftritt. Nachdem mit dieser Schnittstelle eine Bindung generiert wurde, können Sie im Editor der Eigenschaftensicht konfigurieren, wie der Fehler verarbeitet werden soll. Weitere Informationen enthält das Thema über Fehlerselektoren im Information Center von IBM Integration Designer.

Fehlerbehandlung in Exportbindungen:

Wenn beim Verarbeiten der Anforderung von einer Clientanwendung ein Fehler auftritt, kann die Exportbindung die Fehlerinformationen an den Client zurückgeben. Beim Konfigurieren der Exportbindung geben Sie an, wie der Fehler verarbeitet und an den Client zurückgegeben werden soll.

Zur Konfiguration der Exportbindung verwenden Sie IBM Integration Designer.

Während der Anforderungsverarbeitung ruft ein Client einen Export mit einer Anforderung auf. Der Export ruft dann die SCA-Komponente auf. Während der Verarbeitung der Anforderung kann die SCA-Komponente entweder eine Geschäftsantwort zurückgeben oder eine Geschäftsausnahmebedingung bzw. eine Laufzeitausnahmebedingung für den Service auslösen. In einem solchen Fall transformiert die Exportbindung die Ausnahmebedingung in eine Fehlernachricht und sendet diese an den Client. Dies ist in der folgenden Abbildung dargestellt und in den anschließenden Abschnitten beschrieben.

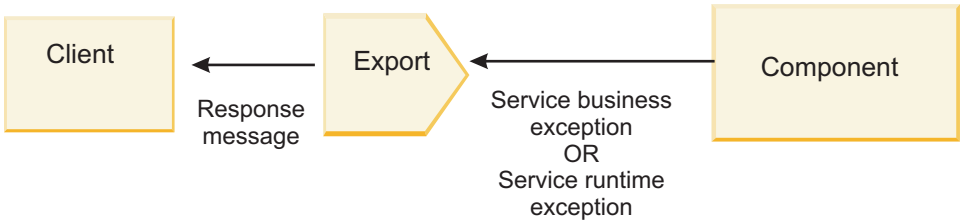


Abbildung 53. Verfahren beim Senden von Fehlerinformationen von der Komponente durch die Exportbindung an den Client

Zur Verarbeitung von Fehlern können Sie einen benutzerdefinierten Datenhandler oder eine benutzerdefinierte Datenbindung erstellen.

Geschäftsausnahmebedingungen

Geschäftsausnahmebedingungen sind Geschäftsfehler oder -ausnahmebedingungen, die während der Verarbeitung auftreten.

Dies soll am Beispiel der folgenden Schnittstelle erläutert werden, für die es eine Operation 'createCustomer' gibt. Für diese Operation sind zwei Geschäftsausnahmebedingungen definiert, nämlich 'CustomerAlreadyExists' (= Kunde ist bereits vorhanden) und 'MissingCustomerId' (= Kunden-ID fehlt).

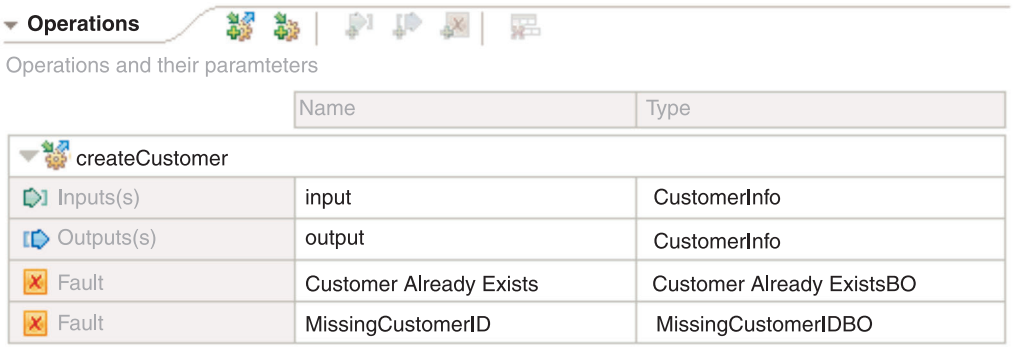


Abbildung 54. Schnittstelle mit zwei Ausnahmebedingungen

Falls in diesem Beispiel ein Client eine Anforderung zum Erstellen eines Kunden (an diese SCA-Komponente) sendet und der Kunde bereits vorhanden ist, löst die Komponente eine Ausnahmebedingung 'Cus-

tomerAlreadyExists' für den Export aus. Der Export muss diese Geschäftsausnahmebedingung an den aufrufenden Client zurückleiten. Zu diesem Zweck wird der Fehlerdatenhandler verwendet, der für die Exportbindung konfiguriert ist.

Sobald durch die Exportbindung eine Geschäftsausnahmebedingung empfangen wird, findet die folgende Verarbeitung statt:

1. Die Bindung stellt fest, welcher Fehlerdatendatenhandler zur Verarbeitung der Ausnahmebedingung aufgerufen werden muss. Falls die Geschäftsausnahmebedingung für den Service den Namen der Ausnahmebedingung enthält, wird der Datenhandler aufgerufen, der für die Ausnahmebedingung konfiguriert ist. Enthält die Geschäftsausnahmebedingung für den Service den Namen der Ausnahmebedingung nicht, wird der Name der Ausnahmebedingung durch einen Abgleich der Fehler- oder Ausnahmebedingungstypen abgeleitet.
2. Die Bindung ruft den Fehlerdatenhandler mit dem Datenobjekt aus der Geschäftsausnahmebedingung für den Service auf.
3. Der Fehlerdatenhandler transformiert das Fehlerdatenobjekt in eine Antwortnachricht und gibt diese an die Exportbindung zurück.
4. Der Export gibt die Antwortnachricht an den Client zurück.

Falls die Geschäftsausnahmebedingung für den Service den Namen der Ausnahmebedingung enthält, wird der Datenhandler aufgerufen, der für die Ausnahmebedingung konfiguriert ist. Enthält die Geschäftsausnahmebedingung für den Service den Namen der Ausnahmebedingung nicht, wird der Name der Ausnahmebedingung durch einen Abgleich der Fehler- oder Ausnahmebedingungstypen abgeleitet.

Laufzeitausnahmebedingung

Eine Laufzeitausnahmebedingung tritt in der SCA-Anwendung während der Verarbeitung einer Anforderung auf, die keiner Geschäftsausnahmebedingung entspricht. Im Gegensatz zu Geschäftsausnahmebedingungen sind Laufzeitausnahmebedingungen nicht in der Schnittstelle definiert.

In bestimmten Szenarios ist es wünschenswert, dass diese Laufzeitausnahmebedingungen an die Clientanwendung weitergegeben werden, damit die Clientanwendung eine geeignete Aktion ausführen kann.

Falls beispielsweise ein Client eine Anforderung zum Erstellen eines Kunden (an die SCA-Komponente) sendet und während der Verarbeitung der Anforderung ein Berechtigungsfehler auftritt, löst die Komponente eine Laufzeitausnahmebedingung aus. Diese Laufzeitausnahmebedingung muss an den aufrufenden Client zurückgegeben werden, damit dieser die geeignete Aktion hinsichtlich der Berechtigung ausführen kann. Dies wird durch den Datenhandler für Laufzeitausnahmebedingungen erreicht, der für die Exportbindung konfiguriert ist.

Anmerkung: Sie können einen Datenhandler für Laufzeitausnahmebedingungen nur bei HTTP-Bindungen konfigurieren.

Die Verarbeitung einer Laufzeitausnahmebedingung erfolgt ähnlich wie bei einer Geschäftsausnahmebedingung. Falls ein Datenhandler für Laufzeitausnahmebedingungen konfiguriert wurde, findet die folgende Verarbeitung statt:

1. Die Exportbindung ruft den entsprechenden Datenhandler mit der Laufzeitausnahmebedingung für den Service auf.
2. Der Datenhandler transformiert das Fehlerdatenobjekt in eine Antwortnachricht und gibt diese an die Exportbindung zurück.
3. Der Export gibt die Antwortnachricht an den Client zurück.

Die Fehlerbehandlung und die Behandlung von Laufzeitausnahmebedingungen sind optional. Wenn Sie nicht wollen, dass Fehler oder Laufzeitausnahmebedingungen an den aufrufenden Client zurückgegeben werden, konfigurieren Sie den Fehlerdatenhandler oder den Datenhandler für Laufzeitausnahmebedingungen nicht.

Fehlerbehandlung in Importbindungen:

Eine Komponente verwendet einen Import, um eine Anforderung an einen Service außerhalb des Moduls zu senden. Wenn während der Verarbeitung der Anforderung ein Fehler auftritt, gibt der Service den Fehler an die Importbindung zurück. Sie können beim Konfigurieren der Importbindung angeben, wie der Fehler verarbeitet und an die Komponente zurückgegeben werden soll.

Zur Konfiguration der Importbindung verwenden Sie IBM Integration Designer. Sie können einen Fehlerdatenhandler (oder eine Datenbindung) angeben. Außerdem geben Sie einen Fehlerselektor an.

Fehlerdatenhandler

Der Service, der die Anforderung verarbeitet, sendet Fehlerinformationen an die Importbindung in Form einer Ausnahmebedingung oder einer Antwortnachricht, die die Fehlerdaten enthält.

Die Importbindung transformiert die Serviceausnahmebedingung oder -antwortnachricht in eine Geschäftsausnahmebedingung oder Laufzeitausnahmebedingung für den Service. Dies ist in der folgenden Abbildung dargestellt und in den anschließenden Abschnitten beschrieben.

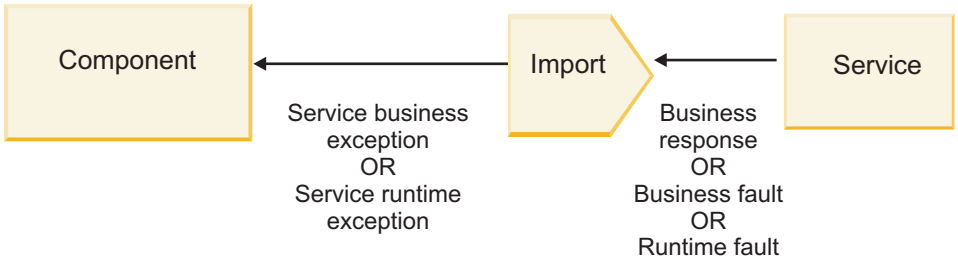


Abbildung 55. Verfahren beim Senden von Fehlerinformationen vom Service durch den Import an die Komponente

Zur Verarbeitung von Fehlern können Sie einen benutzerdefinierten Datenhandler oder eine benutzerdefinierte Datenbindung erstellen.

Fehlerselektoren

Beim Konfigurieren einer Importbindung können Sie einen Fehlerselektor angeben. Der Fehlerselektor ermittelt, ob es sich bei der Importantwort um eine echte Antwort, um eine Geschäftsausnahmebedingung oder um einen Laufzeitfehler handelt. Außerdem ermittelt er aus dem Antworthauptteil oder -header den nativen Fehlernamen, der durch die Bindungskonfiguration zu dem Namen eines Fehlers in der zugehörigen Schnittstelle zugeordnet wird.

Es gibt zwei Typen von vordefinierten Fehlerselektoren, die bei JMS-, MQ-JMS-, generischen JMS-, WebSphere MQ- und HTTP-Importen verwendet werden können:

Tabelle 59. Vordefinierte Fehlerselektoren

Fehlerselektortyp	Beschreibung
Auf Header basierend	Ermittelt anhand der Header in der eingehenden Antwortnachricht, ob es sich bei einer Antwortnachricht um eine Geschäftsausnahmebedingung, eine Laufzeitausnahmebedingung oder um eine normale Nachricht handelt.
SOAP	Ermittelt, ob die SOAP-Antwortnachricht eine normale Antwort, eine Geschäftsausnahmebedingung oder eine Laufzeitausnahmebedingung ist.

Die folgenden Beispiele veranschaulichen auf Headern basierende Fehlerselektoren und SOAP-Fehlerselektoren.

- Auf Header basierender Fehlerselektor

  Falls eine Anwendung angeben will, dass es sich bei der eingehenden Nachricht um eine Geschäftsausnahmebedingung handelt, muss die eingehende Nachricht - wie im Folgenden gezeigt - zwei Header für Geschäftsausnahmebedingungen enthalten:

  ```
  Header name = FaultType, Header value = Business
  Header name = FaultName, Header value = <benutzerdefinierter_nativer_fehlername>
  ```

  Falls eine Anwendung angeben will, dass es sich bei der eingehenden Antwortnachricht um eine Laufzeitausnahmebedingung handelt, muss die eingehende Nachricht - wie im Folgenden gezeigt - einen Header enthalten:

  ```
  Header name = FaultType, Header value = Runtime
  ```

- SOAP-Fehlerselektor

  Eine Geschäftsausnahmebedingung kann mit dem folgenden angepassten SOAP-Header im Rahmen einer SOAP-Nachricht gesendet werden. 'CustomerAlreadyExists' ist in diesem Fall der Name der Ausnahmebedingung.

  ```
  <ibmSoap:BusinessFaultName
  xmlns:ibmSoap="http://www.ibm.com/soap">CustomerAlreadyExists
  <ibmSoap:BusinessFaultName>
  ```

Der Fehlerselektor ist optional. Falls Sie keinen Fehlerselektor angeben, kann die Importbindung den Typ der Antwort nicht ermitteln. Die Bindung behandelt die Antwort daher als Geschäftsantwort und ruft den Antwortdatenhandler oder die Antwortdatenbindung auf.

Sie können einen benutzerdefinierten Fehlerselektor erstellen. Die Schritte zum Erstellen eines benutzerdefinierten Fehlerselektors sind im Thema über die Entwicklung eines angepassten oder benutzerdefinierten Fehlerselektors im Information Center von IBM Integration Designer beschrieben.

Geschäftsausnahmebedingungen

Eine Geschäftsausnahmebedingung kann auftreten, wenn bei der Verarbeitung einer Anforderung ein Fehler auftritt. Beispiel: Falls Sie eine Anforderung zum Erstellen eines Kunden senden und dieser Kunde bereits vorhanden ist, sendet der Service eine Geschäftsausnahmebedingung an die Importbindung.

Nachdem die Bindung eine Geschäftsausnahmebedingung empfangen hat, richten sich die Verarbeitungsschritte danach, ob für die Bindung ein Fehlerselektor definiert wurde.

- Falls kein Fehlerselektor definiert wurde, ruft die Bindung den Antwortdatenhandler oder die Antwortdatenbindung auf.
- Falls ein Fehlerselektor definiert wurde, findet die folgende Verarbeitung statt:
  1. Die Importbindung ruft den Fehlerselektor auf, um zu ermitteln, ob es sich bei der Antwort um eine Geschäftsausnahmebedingung, eine Geschäftsantwort oder einen Laufzeitfehler handelt.
  2. Ist die Antwort eine Geschäftsausnahmebedingung, ruft die Importbindung den Fehlerselektor auf, damit dieser den nativen Fehlernamen bereitstellt.
  3. Die Importbindung ermittelt anhand des nativen Fehlernamens, der vom Fehlerselektor zurückgegeben wurde, den WSDL-Fehler.
  4. Die Importbindung ermittelt den Fehlerdatenhandler, der für diesen WSDL-Fehler konfiguriert ist.
  5. Die Importbindung ruft diesen Fehlerdatenhandler mit den Fehlerdaten auf.
  6. Der Fehlerdatenhandler transformiert die Fehlerdaten in ein Datenobjekt und gibt dies an die Importbindung zurück.
  7. Die Importbindung erstellt ein Geschäftsausnahmebedingungsobjekt für den Service mit dem Datenobjekt und dem Fehlernamen.

8. Der Import gibt das Geschäftsausnahmebedingungsobjekt für den Service an die Komponente zurück.

Laufzeitausnahmebedingungen

Eine Laufzeitausnahmebedingung kann auftreten, wenn bei der Kommunikation mit dem Service ein Problem vorliegt. Die Verarbeitung einer Laufzeitausnahmebedingung erfolgt ähnlich wie bei einer Geschäftsausnahmebedingung. Falls ein Fehlerselektor definiert wurde, findet die folgende Verarbeitung statt:

1. Die Importbindung ruft den geeigneten Datenhandler für Laufzeitausnahmebedingungen mit den Ausnahmedaten auf.
2. Der Datenhandler für Laufzeitausnahmebedingungen transformiert die Ausnahmedaten in ein Laufzeitausnahmebedingungsobjekt für den Service und gibt dieses an die Importbindung zurück.
3. Der Import gibt das Laufzeitausnahmebedingungsobjekt für den Service an die Komponente zurück.

Interoperabilität zwischen SCA-Modulen und Open SCA-Services

IBM WebSphere Application Server V7.0 Feature Pack for Service Component Architecture (SCA) stellt ein einfaches, aber leistungsfähiges Programmiermodell zur Verfügung, mit dem Anwendungen basierend auf den Spezifikationen von 'Open SCA' erstellt werden können. Die SCA-Module von IBM Business Process Manager verwenden Import- und Exportbindungen für die Interaktion mit Open SCA-Services, die in einer Rational Application Developer-Umgebung entwickelt wurden und von WebSphere Application Server Feature Pack for Service Component Architecture per Hosting bereitgestellt werden.

Eine SCA-Anwendung ruft eine Open SCA-Anwendung mittels einer Importbindung auf. Eine SCA-Anwendung empfängt einen Aufruf von einer Open SCA-Anwendung über eine Exportbindung. Eine Liste der unterstützten Bindungen ist unter „Services über Bindungen für Interoperabilität aufrufen" auf Seite 73 aufgeführt.

Open SCA-Services aus SCA-Modulen aufrufen

Mit IBM Integration Designer entwickelte SCA-Anwendungen können Open SCA-Anwendungen aufrufen, die in einer Rational Application Developer-Umgebung entwickelt wurden. Dieser Abschnitt enthält ein Beispiel für den Aufruf eines Open SCA-Service aus einem SCA-Modul, für den eine SCA-Importbindung verwendet wird.

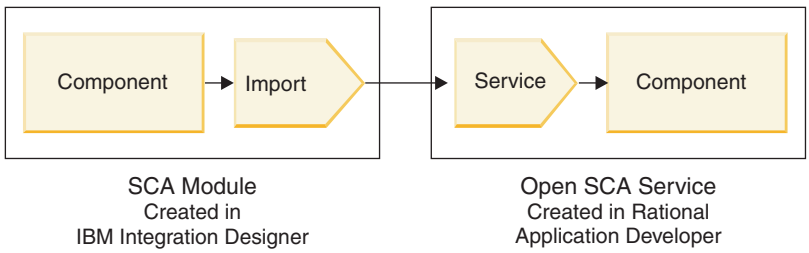


Abbildung 56. Komponente in einem SCA-Modul, die einen Open SCA-Service aufruft

Für den Aufruf eines Open SCA-Service ist keine spezielle Konfiguration erforderlich.

Um über eine SCA-Importbindung eine Verbindung zu einem Open SCA-Service herzustellen, geben Sie den Komponentennamen und den Servicenamen des Open SCA-Service in der Importbindung an.

1. Den Namen der Zielkomponente und des Zielservice im Open SCA-Verbund können Sie folgendermaßen abrufen:

a. Stellen Sie sicher, dass die Registerkarte **Eigenschaften** geöffnet ist, indem Sie auf **Fenster** > **Sicht anzeigen** > **Eigenschaften** klicken.
b. Öffnen Sie den Verbundeditor, indem Sie doppelt auf das Verbunddiagramm klicken, das die Komponente und den Service enthält. Bei einer Komponente namens **customer** heißt das Verbunddiagramm beispielsweise **customer.composite_diagram**.
c. Klicken Sie auf die Zielkomponente.
d. Notieren Sie sich den Namen der Zielkomponente, der im Feld **Name** der Registerkarte **Eigenschaften** angegeben ist.
e. Klicken Sie auf das Servicesymbol, das der Komponente zugeordnet ist.
f. Notieren Sie sich den Namen des Service, der im Feld **Name** der Registerkarte **Eigenschaften** angegeben ist.

2. Gehen Sie folgendermaßen vor, um den IBM Business Process Manager-Import so zu konfigurieren, dass eine Verbindung zum Open SCA-Service hergestellt wird:
a. Navigieren Sie in IBM Integration Designer zur Registerkarte **Eigenschaften** des SCA-Imports, den Sie mit dem Open SCA-Service verbinden wollen.
b. Geben Sie im Feld **Modulname** den Komponentennamen ein, den Sie in 1d auf Seite 72 ermittelt haben.
c. Geben Sie im Feld **Exportname** den Servicenamen ein, den Sie in Schritt 1f auf Seite 72 ermittelt haben.
d. Speichern Sie Ihre Arbeit durch Drücken der Tastenkombination Strg+S.

SCA-Module aus Open SCA-Services aufrufen

Open SCA-Anwendungen, die in einer Rational Application Developer-Umgebung entwickelt wurden, können mit IBM Integration Designer entwickelte SCA-Anwendungen aufrufen. Dieser Abschnitt enthält ein Beispiel für den Aufruf eines SCA-Moduls aus einem Open SCA-Service, für den eine SCA-Exportbindung verwendet wird.

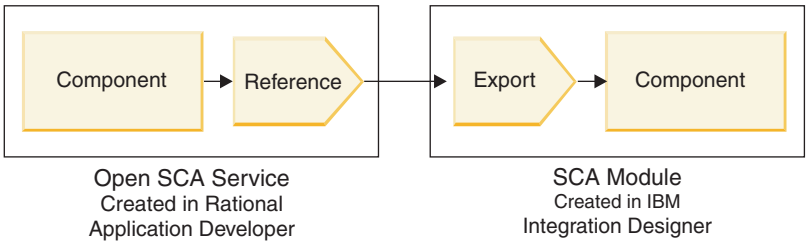


Abbildung 57. Open SCA-Service, der eine Komponente im SCA-Modul aufruft

Um eine Verbindung zu einer SCA-Komponente durch eine Open SCA-Referenzbindung herstellen zu können, müssen Sie den Modulnamen und den Exportnamen angeben.

1. Gehen Sie folgendermaßen vor, um den Namen des Zielmoduls und des Exports abzurufen:
a. Öffnen Sie in IBM Integration Designer das Modul im Assembly-Editor, indem Sie doppelt auf das Modul klicken.
b. Klicken Sie auf den Export.
c. Notieren Sie sich den Namen des Exports, der im Feld **Name** der Registerkarte **Eigenschaften** angegeben ist.

2. Konfigurieren Sie die Open SCA-Referenz, die eine Verbindung zum IBM Business Process Manager-Modul und -Export herstellen soll:
a. Öffnen Sie in Rational Application Developer den Verbundeditor, indem Sie doppelt auf das Verbunddiagramm klicken, das die Komponente und den Service enthält.

b. Klicken Sie auf das Referenzsymbol der Komponentenreferenz, um die Referenzeigenschaften auf der Registerkarte mit den **Eigenschaften** anzuzeigen.
c. Klicken Sie links auf der Seite auf die Registerkarte für die **Bindung**.
d. Klicken Sie auf **Bindungen** (oder 'Bindings') und dann auf **Hinzufügen**.
e. Wählen Sie die **SCA-Bindung** aus.
f. Geben Sie im Feld **URI** den Namen des IBM Business Process Manager-Moduls gefolgt von einem Schrägstrich ('/') und gefolgt vom Exportnamen ein (den Exportnamen haben Sie in Schritt 1c auf Seite 72 ermittelt).
g. Klicken Sie auf **OK**.
h. Speichern Sie Ihre Arbeit durch Drücken der Tastenkombination Strg+S.

Services über Bindungen für Interoperabilität aufrufen

Die folgenden Bindungen werden unterstützt, um eine Interoperabilität mit einem Open SCA-Service bereitzustellen.

- SCA-Bindung

  In IBM Business Process Manager werden die folgenden Aufrufstile unterstützt, wenn ein SCA-Modul einen Open SCA-Service über eine SCA-Importbindung aufruft:

  – Asynchron (unidirektional)
  – Synchron (Anforderung/Antwort)

  Die SCA-Importschnittstelle und die Open SCA-Serviceschnittstelle müssen eine WSDL-Schnittstelle verwenden, die mit WS-I (Web services interoperability) konform ist.

  Bitte beachten Sie, dass die SCA-Bindung die Weitergabe des Transaktions- und Sicherheitskontextes unterstützt.
- Web-Service-Bindung (JAX-WS) entweder mit dem SOAP1.1/HTTP- oder mit dem SOAP1.2/HTTP-Protokoll

  Die SCA-Importschnittstelle und die Open SCA-Serviceschnittstelle müssen eine WSDL-Schnittstelle verwenden, die mit WS-I (Web services interoperability) konform ist.

  Außerdem müssen die folgenden Servicequalitäten unterstützt werden:

  – Atomare Transaktion für Web-Services
  – Sicherheit für Web-Services
- EJB-Bindung

  Die Interaktion zwischen einem SCA-Modul und einem Open SCA-Service wird bei Verwendung einer EJB-Bindung mit einer Java-Schnittstelle definiert.

  Bitte beachten Sie, dass die EJB-Bindung die Weitergabe des Transaktions- und Sicherheitskontextes unterstützt.
- JMS-Bindungen

  Die SCA-Importschnittstelle und die Open SCA-Serviceschnittstelle müssen eine WSDL-Schnittstelle verwenden, die mit WS-I (Web services interoperability) konform ist.

  Die folgenden JMS-Provider werden unterstützt:

  – WebSphere Platform Messaging (JMS-Bindung)
  – WebSphere MQ (MQ-JMS-Bindung)

Anmerkung: Geschäftsgrafiken sind über SCA-Bindungen hinweg nicht interoperabel und werden daher in Schnittstellen, die für die Interoperabilität mit WebSphere Application Server Feature Pack for Service Component Architecture verwendet werden, nicht unterstützt.

Bindungstypen

Sie verwenden protokollspezifische *Bindungen* bei Importen und Exporten, um anzugeben, mit welchem Transportmittel Daten in ein Modul oder aus einem Modul heraus transportiert werden.

Geeignete Bindungen auswählen

Bei der Erstellung einer Anwendung müssen Sie wissen, wie Sie das Binding auswählen können, das den Anforderungen Ihrer Anwendung am besten entspricht.

Die Palette der in IBM Integration Designer verfügbaren Bindings bieten eine große Bandbreite an Auswahlmöglichkeiten. Anhand dieser Liste können Sie erkennen, welcher Bindingtyp den Anforderungen der Anwendung am besten entspricht.

Verwenden Sie eine *SCA-Bindung*, wenn die folgenden Faktoren gegeben sind:

- Alle Services sind in Modulen enthalten. Es gibt keine externen Services.
- Die Funktionsweise soll auf verschiedene SCA-Module verteilt werden, die direkt miteinander interagieren.
- Die Module sind eng verbunden.

Verwenden Sie eine *Web-Service-Bindung*, wenn die folgenden Faktoren gegeben sind:

- Sie müssen über das Internet auf einen externen Provider zugreifen oder einen Service über das Internet bereitstellen.
- Die Services sind flexibel verbunden.
- Die synchrone Übertragung wird bevorzugt; dies bedeutet, dass eine Anforderung von einem Service auf eine Antwort von einem anderen Service warten kann.
- Die externen Services, auf die Sie zugreifen wollen, oder der Service, den Sie bereitstellen wollen, verwenden das Protokoll 'SOAP/HTTP' oder 'SOAP/JMS'.

Verwenden Sie eine *HTTP-Bindung*, wenn die folgenden Faktoren gegeben sind:

- Sie müssen auf einen externen Service über das Internet zugreifen oder einen Service über das Internet bereitstellen, und Sie arbeiten mit anderen Web-Services wie beispielsweise GET, PUT oder DELETE.
- Die Services sind flexibel verbunden.
- Die synchrone Übertragung wird bevorzugt; dies bedeutet, dass eine Anforderung von einem Service auf eine Antwort von einem anderen Service warten kann.

Verwenden Sie eine *EJB-Bindung*, wenn die folgenden Faktoren gegeben sind:

- Die Bindung ist für einen importierten Service gedacht, der entweder selbst eine EJB ist oder auf den von EJB-Clients zugegriffen werden muss.
- Der importierte Service ist flexibel verbunden.
- Interaktionen mit EJBs des Typs 'Stateful' sind nicht erforderlich.
- Die synchrone Übertragung wird bevorzugt; dies bedeutet, dass eine Anforderung von einem Service auf eine Antwort von einem anderen Service warten kann.

Verwenden Sie eine *EIS-Bindung*, wenn die folgenden Faktoren gegeben sind:

- Sie müssen unter Verwendung eines Ressourcenadapters auf einen Service auf einem EIS-System zugreifen.
- Die synchrone Datenübertragung wird der asynchronen Übertragung vorgezogen.

Verwenden Sie eine *JMS-Bindung*, wenn die folgenden Faktoren gegeben sind:

Wichtig: Es gibt mehrere Typen von JMS-Bindungen. Falls Sie voraussichtlich SOAP-Nachrichten mit JMS austauschen werden, kann die Web-Service-Bindung mit dem Protokoll 'SOAP/JMS' eine gute Wahl sein. Weitere Informationen hierzu finden Sie unter „Web-Service-Bindungen” auf Seite 75.

- Sie benötigen ein Messaging-System.
- Die Services sind flexibel verbunden.
- Die asynchrone Datenübertragung wird der synchronen Übertragung vorgezogen.

Verwenden Sie eine *JMS-Bindung*, wenn die folgenden Faktoren gegeben sind:
- Sie müssen auf ein Messaging-System eines anderen Herstellers als IBM zugreifen.
- Die Services sind flexibel verbunden.
- Die Zuverlässigkeit ist wichtiger als die Leistung; das heißt, die asynchrone Datenübertragung ist der synchronen vorzuziehen.

Verwenden Sie eine *MQ-Bindung*, wenn die folgenden Faktoren gegeben sind:
- Sie müssen auf ein WebSphere MQ-Messaging-System zugreifen und die nativen MQ-Funktionen verwenden.
- Die Services sind flexibel verbunden.
- Die Zuverlässigkeit ist wichtiger als die Leistung; das heißt, die asynchrone Datenübertragung ist der synchronen vorzuziehen.

Verwenden Sie eine *MQ-JMS-Bindung*, wenn die folgenden Faktoren gegeben sind:
- Sie müssen auf ein WebSphere MQ-Messaging-System zugreifen, können dies jedoch auch aus einem JMS-Kontext heraus ausführen, die JMS-Funktionsuntergruppe ist also für Ihre Anwendung ausreichend.
- Die Services sind flexibel verbunden.
- Die Zuverlässigkeit ist wichtiger als die Leistung; das heißt, die asynchrone Datenübertragung ist der synchronen vorzuziehen.

SCA-Bindungen

Eine SCA-Bindung ermöglicht einem Service die Kommunikation mit anderen Services in anderen Modulen. Über einen Import mit einer SCA-Bindung können Sie auf einen Service in einem anderen SCA-Modul zugreifen. Ein Export mit einer SCA-Bindung ermöglicht es Ihnen, einen Service anderen Modulen anzubieten.

Verwenden Sie IBM Integration Designer, um SCA-Bindungen für Importe und Exporte in SCA-Modulen zu generieren und konfigurieren.

Falls Module auf demselben Server ausgeführt werden oder in demselben Cluster implementiert sind, ist die Verwendung einer SCA-Bindung die einfachste und schnellste Methode.

Nachdem das Modul, das die SCA-Bindung enthält, auf dem Server implementiert wurde, können Sie in der Administrationskonsole Informationen zur Bindung anzeigen oder bei einer Importbindung ausgewählte Eigenschaften der Bindung ändern.

Web-Service-Bindungen

Eine Web-Service-Bindung ist ein Verfahren, mit dem Nachrichten von einer SCA-Komponente an einen Web-Service (und umgekehrt) übertragen werden können.

Web-Service-Bindungen - Übersicht:

Eine Web-Service-Import-Bindung ermöglicht Ihnen den Aufruf eines externen Web-Service aus SCA-Komponenten heraus. Mit einer Web-Service-Exportbindung können Sie Ihre SCA-Komponenten als Web-Services für Clients zugänglich machen.

Mit einer Web-Service-Bindung können Sie unter Verwendung von interoperablen SOAP-Nachrichten und Servicequalitäten (QoS) auf einen externen Service zugreifen.

Zum Generieren und Konfigurieren von Web-Service-Bindungen für Importe und Exporte in SCA-Modulen verwenden Sie Integration Designer. Die folgenden Typen von Web-Service-Bindungen sind verfügbar:

- SOAP1.2/HTTP und SOAP1.1/HTTP

  Diese Bindungen basieren auf Java API for XML Web Services (JAX-WS), einer Java-Programmierungs-API für die Erstellung von Web-Services.

  – Verwenden Sie SOAP1.2/HTTP, falls Ihr Web-Service der Spezifikation von SOAP 1.2 entspricht.
  – Verwenden Sie SOAP1.1/HTTP, falls Ihr Web-Service der Spezifikation von SOAP 1.1 entspricht.

  Wichtig: Wenn Sie eine Anwendung mit einer Web-Service-Bindung (JAX-WS-Bindung) implementieren, darf für den Zielserver nicht die Option für den **Start von Komponenten nach Bedarf** ausgewählt sein. Details finden Sie unter „Serverkonfiguration prüfen" auf Seite 84.

  Wenn Sie eine dieser Bindungen auswählen, können Sie zusammen mit den SOAP-Nachrichten Anhänge senden.

  Die Web-Service-Bindungen können bei SOAP-Standardnachrichten verwendet werden. Wenn Sie eine der JAX-WS-Bindungen für Web-Services verwenden, können Sie jedoch anpassen, wie diese SOAP-Nachrichten syntaktisch analysiert oder geschrieben werden. Sie können beispielsweise vom Standard abweichende Element in SOAP-Nachrichten verarbeiten oder eine zusätzliche Verarbeitung auf die SOAP-Nachricht anwenden. Beim Konfigurieren der Bindung geben Sie einen benutzerdefinierten Datenhandler an, der diese Verarbeitung für die SOAP-Nachricht ausführt.

  Zusammen mit einer Web-Service-Bindung (JAX-WS) können Sie einen Richtliniensatz verwenden. Ein Richtliniensatz ist eine Sammlung von Richtlinientypen, die jeweils eine Servicequalität bereitstellen. Der Richtliniensatz 'WSAddressing' bietet beispielsweise ein transportneutrales Verfahren für die einheitliche Adressierung von Web-Services und Nachrichten. Zur Auswahl des Richtliniensatzes für die Bindung verwenden Sie Integration Designer.

  Anmerkung: Falls Sie einen SAML-Richtliniensatz (SAML = Security Assertion Markup Language) verwenden wollen, müssen Sie einige zusätzliche Konfigurationsschritte ausführen, die unter „SAML-Richtliniensätze importieren" auf Seite 82 beschrieben sind.
- SOAP1.1/HTTP

  Verwenden Sie diese Bindung, wenn Sie Web-Services erstellen wollen, die eine SOAP-codierte Nachricht verwenden, die auf Java API for XML-based RPC (JAX-RPC) basiert.
- SOAP1.1/JMS

  Verwenden Sie diese Bindung, um SOAP-Nachrichten mit einem JMS-Ziel (JMS = Java Message Service) zu senden oder zu empfangen.

Unabhängig vom Transportprotokoll (HTTP oder JMS), mit dem die SOAP-Nachricht übermittelt wird, verarbeiten Web-Service-Bindungen Anforderungs-/Antwortinteraktionen immer synchron. Der Thread, der den Aufruf des Service-Providers ausführt, wird bis zum Empfang einer Antwort vom Provider geblockt. Weitere Informationen zur diesem Aufrufstil finden Sie im Thema über den synchronen Aufruf.

Wichtig: Die folgenden Kombinationen von Web-Service-Bindungen können für Exporte in demselben Modul nicht verwendet werden. Falls Sie Komponenten zugänglich machen müssen, die mehrere dieser Exportbindungen verwenden, müssen Sie für jede Komponente ein separates Modul verwenden und diese Module dann über eine SCA-Bindung mit Ihren Komponenten verbinden:

- SOAP 1.1/JMS und SOAP 1.1/HTTP mit JAX-RPC
- SOAP 1.1/HTTP mit JAX-RPC und SOAP 1.1/HTTP mit JAX-WS
- SOAP 1.1/HTTP mit JAX-RPC und SOAP 1.2/HTTP mit JAX-WS

Nachdem das SCA-Modul, das die Web-Service-Bindung enthält, auf dem Server implementiert wurde, können Sie in der Administrationskonsole die Informationen zur Bindung anzeigen oder ausgewählte Eigenschaften der Bindung ändern.

Anmerkung: Mithilfe von Web-Services können Anwendungen zusammenarbeiten, indem Standardbeschreibungen von Services und Standardformate für die ausgetauschten Nachrichten verwendet werden. Die Web-Service-Importbindungen und -Exportbindungen können beispielsweise mit Service interagieren, die unter Verwendung von Web Services Enhancements (WSE) Version 3.5 und Windows Communication Foundation (WCF) Version 3.5 für Microsoft .NET implementiert wurden. Bei der Interaktion mit solchen Services muss Folgendes sichergestellt sein:

- Die WSDL-Datei, die für den Zugriff auf einen Web-Service verwendet wird, enthält für jede Operation in der Schnittstelle einen nicht leeren SOAP-Aktionswert.
- Der Web-Service-Client legt entweder den Header 'SOAPAction' oder den Header 'wsa:Action' fest, wenn Nachrichten an einen Web-Service-Export gesendet werden.

SOAP-Headerweitergabe:

Bei der Verarbeitung von SOAP-Nachrichten müssen Sie möglicherweise auf Informationen aus bestimmten SOAP-Headern in empfangenen Nachrichten zugreifen oder sicherstellen, dass Nachrichten mit SOAP-Headern mit bestimmten Werten gesendet werden, oder die Übergabe von SOAP-Headern über ein Modul zulassen.

Wenn Sie in Integration Designer eine Web-Service-Bindung konfigurieren, können Sie angeben, dass SOAP-Header weitergegeben werden sollen.

- Sobald Anforderungen bei einem Export oder Antworten bei einem Import empfangen werden, ist der Zugriff auf die SOAP-Headerinformationen möglich. Hierdurch können die Headerwerte für die Logik im Modul zugrunde gelegt werden und die Änderung dieser Header kann ermöglicht werden.
- Wenn Anforderungen von einem Export oder Antworten von einem Import aus gesendet werden, können in die entsprechenden Nachrichten SOAP-Header eingeschlossen werden.

Das Format und das Vorhandensein der weitergegebenen SOAP-Header kann durch Richtliniensätze beeinflusst werden, die für den Import oder Export konfiguriert sind (siehe Tabelle 31 auf Seite 78).

Um die Weitergabe von SOAP-Headern für einen Import oder Export zu konfigurieren, wählen Sie (in der Sicht 'Eigenschaften' von Integration Designer) die Registerkarte **Protokollheader weitergeben** und dann die benötigten Optionen aus.

Header 'WS-Addressing'

Der Header 'WS-Addressing' kann durch die Web-Service-Bindung (JAX-WS) weitergegeben werden.

Beim Weitergeben des Headers 'WS-Addressing' müssen Sie die folgenden Informationen beachten:

- Falls Sie die Weitergabe für den Header 'WS-Addressing' aktivieren, wird der Header unter den folgenden Umständen in das Modul weitergegeben:
  - Anforderungen werden bei einem Export empfangen.
  - Antworten werden bei einem Import empfangen.
- Der Header 'WS-Addressing' wird nicht in abgehende Nachrichten von IBM Business Process Manager weitergegeben (dies bedeutet, dass der Header nicht weitergegeben wird, wenn Anforderungen von einem Import oder Antworten vom Export aus gesendet werden).

Header 'WS-Security'

Der Header 'WS-Security' kann sowohl durch die Web-Service-Bindung (JAX-WS) als auch durch die Web-Service-Bindung (JAX-RPC) weitergegeben werden.

Die Spezifikation von 'WS-Security' für Web-Services beschreibt Erweiterungen für die SOAP-Nachrichtentübertragung, die ein Datenschutzniveau mittels Nachrichtenintegrität, Nachrichtenvertraulichkeit und Einzelnachrichtenauthentifizierung bereitstellen. Mit diesen Mechanismen kann eine Vielzahl von Sicherheitsmodellen und Verschlüsselungstechnologien einbezogen werden.

Beim Weitergeben des Headers WS-Security müssen Sie die folgenden Informationen beachten:

- Falls Sie die Weitergabe für den Header 'WS-Security' aktivieren, wird der Header unter den folgenden Umständen über das Modul weitergegeben:
  - Anforderungen werden bei einem Export empfangen.
  - Anforderungen werden von einem Import gesendet.
  - Antworten werden bei einem Import empfangen.
- Der Header wird standardmäßig *nicht* weitergegeben, wenn Antworten vom Export aus gesendet werden. Falls Sie jedoch für die JVM-Eigenschaft **WSSECURITY.ECHO.ENABLED** die Einstellung **true** festlegen, wird der Header weitergegeben, wenn Antworten vom Export aus gesendet werden. In diesem Fall werden Header 'WS-Security' möglicherweise automatisch aus Anforderungen in Antworten zurückgemeldet, falls der Header 'WS-Security' im Anforderungspfad nicht geändert wird.
- Das genaue Format der SOAP-Nachricht, die von einem Import für eine Anforderung oder von einem Export für eine Antwort gesendet wird, stimmt möglicherweise nicht präzise mit der ursprünglich empfangenen SOAP-Nachricht überein. Aus diesem Grund sollte davon ausgegangen werden, dass digitale Signaturen ungültig werden. Falls in gesendeten Nachrichten eine digitale Signatur erforderlich ist, muss sie mit dem entsprechenden Sicherheitsrichtliniensatz erstellt werden. Header 'WS-Security', die in empfangenen Nachrichten mit der digitalen Signatur verbunden sind, sollten im Modul entfernt werden.

Zur Weitergabe des Headers 'WS-Security' müssen Sie das Schema 'WS-Security' mit dem Anwendungsmodul einschließen. Die Vorgehensweise zum Einschließen des Schemas ist unter „Schema 'WS-Security' in ein Anwendungsmodul einschließen” auf Seite 79 beschrieben.

Methode für Weitergabe von Headern

Die Methode, mit der Header weitergegeben werden, richtet sich nach der Sicherheitsrichtlinieneinstellung für die Import- oder Exportbindung (siehe Tabelle 31 auf Seite 78):

Tabelle 60. Methode für Übergabe von Sicherheitsheadern

	Exportbindung ohne Sicherheitsrichtlinie	**Exportbindung mit Sicherheitsrichtlinie**
Importbindung ohne Sicherheitsrichtlinie	Sicherheitsheader werden unverändert über das Modul übergeben. Sie werden nicht entschlüsselt. Die Header werden abgehend in demselben Format gesendet, in dem sie empfangen wurden. Die digitale Signatur wird möglicherweise ungültig.	Sicherheitsheader werden entschlüsselt und über das Modul übergeben. Hierbei findet eine Prüfung und Authentifizierung der Signatur statt. Die entschlüsselten Header werden abgehend gesendet. Die digitale Signatur wird möglicherweise ungültig.
Importbindung mit Sicherheitsrichtlinie	Sicherheitsheader werden unverändert über das Modul übergeben. Sie werden nicht entschlüsselt. Die Header sollten nicht an den Import weitergegeben werden. Andernfalls tritt aufgrund einer Duplizierung ein Fehler auf.	Sicherheitsheader werden entschlüsselt und über das Modul übergeben. Hierbei findet eine Prüfung und Authentifizierung der Signatur statt. Die Header sollten nicht an den Import weitergegeben werden. Andernfalls tritt aufgrund einer Duplizierung ein Fehler auf.

Konfigurieren Sie entsprechende Richtliniensätze für die Export- und Importbindungen, da der Serviceanforderer hierdurch von Änderungen an der Konfiguration oder den QoS-Anforderungen des Service-Providers isoliert wird. Durch sichtbare SOAP-Standardheader in einem Modul kann dann die Verarbeitung

im Modul (z. B. Protokollierung und Traceverarbeitung) beeinflusst werden. Die Weitergabe von SOAP-Headern über ein Modul von einer empfangenen Nachricht an eine gesendete Nachricht bedeutet, dass die Isolationsvorteile des Moduls verringert werden.

Standardheader wie beispielsweise 'WS-Security' sollten nicht an eine Anforderung an einen Import oder an eine Antwort an einen Export weitergegeben werden, wenn dem Import oder Export ein Richtliniensatz zugeordnet ist, der normalerweise eine Generierung dieser Header bewirken würde. Andernfalls tritt ein Fehler auf, weil die Header doppelt vorhanden sind. Die Header sollten stattdessen explizit entfernt werden oder die Import- bzw. Exportbindung sollte so konfiguriert sein, dass die Weitergabe von Protokollheadern verhindert wird.

Auf SOAP-Header zugreifen

Wenn eine Nachricht, die SOAP-Header enthält, von einem Web-Service-Import oder -Export empfangen wird, werden die Header in den Headerabschnitt des Servicenachrichtenobjekts gestellt. Sie können auf die Headerinformationen zugreifen. Dies ist unter 'Zugriff auf SOAP-Headerdaten im SMO' beschrieben.

Schema 'WS-Security' in ein Anwendungsmodul einschließen

Die folgende Prozedur skizziert die Schritte, mit denen das Schema in das Anwendungsmodul eingeschlossen wird:

- Falls der Computer, auf dem Integration Designer ausgeführt wird, über einen Internetzugang verfügt, führen Sie die folgenden Schritte aus:
  1. Wählen Sie in der Perspektive 'Geschäftsintegration' für Ihr Projekt den Eintrag **Abhängigkeiten** aus.
  2. Erweitern Sie **Vordefinierte Ressourcen** und wählen Sie entweder **WS-Security 1.0-Schemadateien** oder **WS-Security 1.1-Schemadateien** aus, um das Schema in das Modul zu importieren.
  3. Bereinigen Sie das Projekt und erstellen Sie es erneut.
- Falls ein Computer, auf dem Integration Designer ausgeführt wird, nicht über einen Internetzugang verfügt, können Sie das Schema auf einen zweiten Computer mit Internetzugang herunterladen. Anschließend können Sie es auf den Computer kopieren, auf dem Integration Designer ausgeführt wird.
  1. Laden Sie das ferne Schema auf dem Computer mit Internetzugang herunter:
     a. Klicken Sie auf **Datei** > **Importieren** > **Geschäftsintegration** > **WSDL und XSD**.
     b. Wählen Sie **Ferne WSDL- oder XSD-Datei** aus.
     c. Importieren Sie die folgenden Schemas:

        http://www.w3.org/2003/05/soap-envelope/

        http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/xenc-schema.xsd

        http://www.w3.org/TR/xmldsig-core/xmldsig-core-schema.xsd
  2. Kopieren Sie die Schemas auf den Computer ohne Internetzugang.
  3. Importieren Sie das Schema auf dem Computer ohne Internetzugang:
     a. Klicken Sie auf **Datei** > **Importieren** > **Geschäftsintegration** > **WSDL und XSD**.
     b. Wählen Sie **Lokale WSDL-Datei oder XSD-Datei** aus.
  4. Ändern Sie die Schemapositionen für 'oasis-wss-wssecurity_secext-1.1.xsd':
     a. Öffnen Sie das Schema an der Position *arbeitsplatzposition/modulname/*`StandardImportFilesGen/oasis-wss-wssecurity-secext-1.1.xsd`.
     b. Ändern Sie

        ```
        <xs:import namespace='http://www.w3.org/2003/05/soap-envelope'
        schemaLocation='http://www.w3.org/2003/05/soap-envelope/'/>
        ```

        in:

        ```
        <xs:import namespace='http://www.w3.org/2003/05/soap-envelope'
        schemaLocation='../w3/_2003/_05/soap_envelope.xsd'/>
        ```

c. Ändern Sie

```
<xs:import namespace='http://www.w3.org/2001/04/xmlenc#'
schemaLocation='http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/xenc-schema.xsd'/>
```

in:

```
<xs:import namespace='http://www.w3.org/2001/04/xmlenc#'
schemaLocation='../w3/tr/_2002/rec_xmlenc_core_20021210/xenc-schema.xsd'/>
```

5. Ändern Sie die Schemaposition für 'oasis-200401-wss-wssecurity-secext-1.0.xsd':
   a. Öffnen Sie das Schema an der Position *arbeitsplatzposition/modulname/*`StandardImportFilesGen/oasis-200401-wss-wssecurity-secext-1.0.xsd`.
   b. Ändern Sie

```
<xsd:import namespace="http://www.w3.org/2000/09/xmldsig#"
schemaLocation="http://www.w3.org/TR/xmldsig-core/xmldsig-core-schema.xsd"/>
```

in:

```
<xsd:import namespace="http://www.w3.org/2000/09/xmldsig#"
schemaLocation="../w3/tr/_2002/rec_xmldsig_core_20020212/xmldsig-core-schema.xsd"/>
```

6. Bereinigen Sie das Projekt und erstellen Sie es erneut.

Transportheaderweitergabe:

Bei der Verarbeitung von SOAP-Nachrichten müssen Sie möglicherweise auf Informationen aus bestimmten Transportheadern in empfangenen Nachrichten zugreifen oder sicherstellen, dass Nachrichten mit Transportheadern mit bestimmten Werten gesendet werden, oder die Übergabe von Transportheadern über ein Modul zulassen.

Wenn Sie in Integration Designer eine Web-Service-Bindung konfigurieren, können Sie angeben, dass Transportheader weitergegeben werden sollen.

- Sobald Anforderungen bei einem Export oder Antworten bei einem Import empfangen werden, ist der Zugriff auf die Transportheaderinformationen möglich. Hierdurch können die Headerwerte für die Logik im Modul zugrunde gelegt werden und die Änderung dieser Header kann ermöglicht werden.
- Wenn Antworten von einem Export oder Anforderungen von einem Import aus gesendet werden, können in die entsprechenden Nachrichten Transportheader eingeschlossen werden.

Weitergabe von Headern angeben

Gehen Sie folgendermaßen vor, um die Weitergabe von Transportheadern für einen Import oder Export zu konfigurieren:

1. Wählen Sie in der Sicht 'Eigenschaften' von Integration Designer die Optionen **Binding** > **Weitergabe** aus.
2. Legen Sie die Option für die benötigte Transportheaderweitergabe fest.

Anmerkung: Die Transportheaderweitergabe ist standardmäßig inaktiviert und kann nur in einer Laufzeitumgebung mit Version 7.0.0.3 (oder höher) implementiert werden. Bitte beachten Sie außerdem, dass die Transportheaderweitergabe bei Version 7.0.0.3 auf HTTP-Transportheader beschränkt ist.

Falls Sie die Weitergabe von Transportheadern aktivieren, werden die Header über ein Modul aus empfangenen Nachrichten weitergegeben und bei nachfolgenden Aufrufen in demselben Thread verwendet, wenn Sie die Header nicht explizit entfernen.

Anmerkung: Transportheader können bei Verwendung einer Web-Service-Bindung (JAX-RPC) nicht weitergegeben werden.

Auf Headerinformationen zugreifen

Wenn die Transportheaderweitergabe für empfangene Nachrichten aktiviert ist, sind alle Transportheader (auch kundendefinierte Header) im Servicenachrichtenobjekt sichtbar. Sie können für die Header andere Werte angeben oder neue Header erstellen. Bitte beachten Sie jedoch, dass für die von Ihnen festgelegten Werte keine Überprüfung oder Validierung stattfindet. Falsche Header können Laufzeitprobleme für den Web-Service verursachen.

Berücksichtigen Sie beim Festlegen von HTTP-Headern die folgenden Informationen:

- Alle Änderungen an den Headern, die für die Web-Service-Steuerkomponente reserviert sind, werden in der abgehenden Nachricht nicht berücksichtigt. Beispielsweise sind die HTTP-Version oder die HTTP-Methode bzw. die Header 'Content-Type', 'Content-Length' und 'SOAPAction' für die Web-Service-Steuerkomponente reserviert.
- Falls ein Headerwert eine Zahl ist, sollte die Zahl (und nicht die Zeichenfolge) direkt gesendet werden. Verwenden Sie beispielsweise **Max-Forwards = 5** (anstelle von **Max-Forwards = Max-Forwards: 5**) und **Age = 300** (anstelle von **Age = Age: 300**).
- Falls die Anforderungsnachricht kleiner als 32 KB ist, entfernt die Web-Service-Steuerkomponente den Header 'Transfer-Encoding' und setzt den Header 'Content-Length' auf die feste Größe der Nachricht.
- Der Header 'Content-language' wird im Antwortpfad durch 'WAS.channel.http' zurückgesetzt.
- Eine ungültige Einstellung für 'Upgrade' führt zu einem Fehler 500.
- Die folgenden Header hängen den Wert, der durch die Web-Service-Steuerkomponente reserviert ist, an die Kundeneinstellungen an:
  - User-Agent
  - Cache-Control
  - Pragma
  - Accept
  - Connection

Zum Zugriff auf die Headerinformationen können Sie eines der folgenden Verfahren verwenden:

- Zugriff auf die Strukturen des Servicenachrichtenobjekts mit einem Mediationsbasiselement

  Nach Auswahl der Links zu den Referenzinformationen erhalten Sie Informationen zur Verwendung von Mediationsbasiselementen.
- Verwendung der Kontextservice-SPI

  Der folgende Beispielcode liest die HTTP-Transportheader aus dem Kontextservice:

```
HeadersType headerType = ContextService.INSTANCE.getHeaders();
HTTPHeaderType httpHeaderType = headerType.getHTTPHeader();
List HTTPHeader  httpHeaders = httpHeaderType.getHeader();
if(httpHeaders!=null){
 for(HTTPHeader httpHeader: httpHeaders){
   String httpHeadername = httpHeader.getName();
   String httpHeaderValue = httpHeader.getValue();
     }
}
List PropertyType  properties = headerType.getProperties();
if(properties!=null){
  for(PropertyType property: properties){
   String propertyName = property.getName();
   String propertyValue = property.getValue().toString();
     }
}
```

Fehlerbehebung

Falls beim Senden von überarbeiteten Headern Probleme auftreten, können Sie die TCP/IP-Nachrichten mithilfe von Tools wie der TCP/IP-Überwachung in Integration Designer abfangen. Auf die TCP/IP-Überwachung greifen Sie zu, indem Sie auf der Seite 'Benutzervorgaben' die Optionen **Ausführen/Debug** > **TCP/IP-Überwachung** auswählen.

Zum Anzeigen der Headerwerte können Sie auch den JAX-WS-Steuerkomponententrace verwenden: **org.apache.axis2.*=all: com.ibm.ws.websvcs.*=all:**

Mit Web-Service-Bindungen (JAX-WS) arbeiten:

Wenn Sie in Ihren Anwendungen Web-Service-Bindungen (JAX-WS) verwenden, können Sie eine SAML-Servicequalität (SAML = Security Assertion Markup Language) zur Bindung hinzufügen. Zunächst müssen Sie mit der Administrationskonsole den Richtliniensatz importieren. In der Administrationskonsole können Sie zudem sicherstellen, dass der Server ordnungsgemäß für die Verwendung der Web-Service-Bindung (JAX-WS) konfiguriert ist.

SAML-Richtliniensätze importieren:

Die Security Assertion Markup Language (Kurzform: SAML) ist ein XML-basierter OASIS-Standard für den Austausch von Informationen zu Benutzeridentitäts- und Sicherheitsattributen. Wenn Sie in Integration Designer eine Web-Service-Bindung (JAX-WS) konfigurieren, können Sie einen SAML-Richtliniensatz angeben. Zuerst machen Sie die SAML-Richtliniensätze mit der Administrationskonsole von IBM Business Process Manager verfügbar, sodass sie in Integration Designer importiert werden können.

Die SAML-Richtliniensätze befinden sich normalerweise im Verzeichnis für die Profilkonfiguration:

profilstammverzeichnis/config/templates/PolicySets

Bevor Sie diese Prozedur starten, müssen Sie sicherstellen, dass sich die folgenden Verzeichnisse, die die Richtliniensätze enthalten, im Verzeichnis für die Profilkonfiguration befinden:

- SAML11 Bearer WSHTTPS default
- SAML20 Bearer WSHTTPS default
- SAML11 Bearer WSSecurity default
- SAML20 Bearer WSSecurity default
- SAML11 HoK Public WSSecurity default
- SAML20 HoK Public WSSecurity default
- SAML11 HoK Symmetric WSSecurity default
- SAML20 HoK Symmetric WSSecurity default
- Benutzername WSHTTPS default

Wenn sich die Verzeichnisse nicht im Verzeichnis für die Profilkonfiguration befinden, kopieren Sie sie von der folgenden Position in dieses Verzeichnis:

Stammverzeichnis des Anwendungsservers/profileTemplates/default/documents/config/templates/PolicySets

Sie importieren die Richtliniensätze in die Administrationskonsole, wählen anschließend diejenigen Richtliniensätze aus, die für Integration Designer verfügbar gemacht werden sollen, und speichern dann für jeden dieser Richtliniensätze eine komprimierte Datei an einer Position, auf die von Integration Designer aus zugegriffen werden kann.

1. Importieren Sie die Richtliniensätze, indem Sie die folgenden Schritte ausführen:

a. Klicken Sie in der Administrationskonsole auf **Services** > **Richtliniensätze** > **Anwendungsrichtliniensätze**.
b. Klicken Sie auf **Importieren** > **Aus Standard-Repository**.
c. Wählen Sie die standardmäßigen SAML-Richtliniensätze aus und klicken Sie auf **OK**.

2. Exportieren Sie die Richtliniensätze, sodass sie von Integration Designer verwendet werden können:
   a. Wählen Sie auf der Seite für Anwendungsrichtliniensätze denjenigen SAML-Richtliniensatz aus, der exportiert werden soll, und klicken Sie auf **Exportieren**.

      Anmerkung: Wenn die Seite für Anwendungsrichtliniensätze gerade nicht angezeigt wird, klicken Sie in der Administrationskonsole auf **Services** > **Richtliniensätze** > **Anwendungsrichtliniensätze**.
   b. Klicken Sie auf der nächsten Seite für den Richtliniensatz auf den Link für die komprimierte Datei (ZIP).
   c. Klicken Sie im Fenster für den Dateidownload auf **Speichern** und geben Sie dann eine Position an, auf die von Integration Designer zugegriffen werden kann.
   d. Klicken Sie auf **Zurück**.
   e. Führen Sie die Schritte 2a auf Seite 83 bis 2d auf Seite 83 für alle Richtliniensätze aus, die exportiert werden sollen.

Die SAML-Richtliniensätze sind in komprimierten Dateien (ZIP) gespeichert und können nun in Integration Designer importiert werden.

Importieren Sie die Richtliniensätze in Integration Designer wie in „Richtliniensätze” beschrieben.

Web-Services aufrufen, die eine HTTP-Basisauthentifizierung erfordern:

Die HTTP-Basisauthentifizierung verwendet einen Benutzernamen und ein Kennwort für die Authentifizierung eines Service-Clients bei einem sicheren Endpunkt. Sie können die HTTP-Basisauthentifizierung beim Senden oder Empfangen von Web-Service-Anforderungen einrichten.

Sie richten die HTTP-Basisauthentifizierung zum Empfangen von Web-Service-Anforderungen ein, indem Sie die Exportbindung für die Java API for XML Web Services (JAX-WS) wie in Sicherheitsaufgabenbereiche erstellen und zu Web-Service-Exporten zuordnen erläutert konfigurieren.

Die HTTP-Basisauthentifizierung kann für Web-Service-Anforderungen, die von einer JAX-WS-Importbindung gesendet werden, auf eine von insgesamt zwei Arten aktiviert werden:

- Beim Konfigurieren der Importbindung in einem SCA-Modul können Sie den bereitgestellten Richtliniensatz für die HTTP-Authentifizierung namens 'BPMHTTPBasicAuthentication' auswählen, der mit der Web-Service-Importbindung (JAX-WS) zur Verfügung gestellt wird, oder Sie können einen beliebigen anderen Richtliniensatz auswählen, der die Richtlinie 'HTTPTransport' enthält.
- Beim Erstellen des SCA-Moduls können Sie die Funktionalität des Mediationsablaufs nutzen, um dynamisch einen neuen HTTP-Authentifizierungsheader zu erstellen und im Header den Benutzernamen und das Kennwort anzugeben.

Anmerkung: Der Richtliniensatz hat Vorrang vor dem im Header angegebenen Wert. Wenn zur Laufzeit der im HTTP-Authentifizierungsheader festgelegte Wert verwendet werden soll, hängen Sie keinen Richtliniensatz an, der die Richtlinie 'HTTPTransport' beinhaltet. Insbesondere müssen Sie auf die Verwendung des standardmäßigen Richtliniensatzes 'BPMHTTPBasicAuthentication' verzichten und sicherstellen, dass der definierte Richtliniensatz (sofern ein solcher vorhanden ist) die Richtlinie 'HTTPTransport' keinesfalls enthält.

Weitere Informationen zu Richtliniensätzen für Web-Services und Richtlinienbindungen sowie ihrer Verwendung finden Sie im Abschnitt über Richtliniensätze für Web-Services im WebSphere Application Server Information Center.

- Führen Sie die folgenden Schritte aus, um den bereitgestellten Richtliniensatz zu verwenden:
  1. Optional: Erstellen Sie in der Administrationskonsole eine allgemeine Richtlinienbindung für den Client oder bearbeiten Sie eine bereits vorhandene Bindung, die die Richtlinie 'HTTPTransport' mit den erforderlichen Werten für Benutzer-ID und Kennwort enthält.
  2. Generieren Sie in IBM Integration Designer eine Web-Service-Importbindung (JAX-WS) und hängen Sie den Richtliniensatz 'BPMHTTPBasicAuthentication' an.
  3. Führen Sie *einen* der folgenden Schritte aus:
     – Geben Sie in IBM Integration Designer in den Eigenschaften für Web-Service-Importbindungen (JAX-WS) den Namen einer vorhandenen allgemeinen Richtlinienbindung für Clients an, die die Richtlinie 'HTTPTransport' einschließt.
     – Wählen Sie nach der Implementierung des SCA-Moduls in der Administrationskonsole entweder eine vorhandene Clientrichtlinienbindung aus oder erstellen Sie eine neue Clientrichtlinienbindung und ordnen Sie diese anschließend der Importbindung zu.
  4. Optional: Bearbeiten Sie in der Administrationskonsole des Process Server die ausgewählte Richtliniensatzbindung, um die erforderliche ID und das erforderliche Kennwort anzugeben.
- Führen Sie eine der folgenden Gruppen von Schritten aus, um den Benutzernamen und das Kennwort im HTTP-Authentifizierungsheader anzugeben:
  – Erstellen Sie den HTTP-Authentifizierungsheader unter Verwendung des Mediationsbasiselements 'HTTP-Header-Setter' ('HTTP Header Setter' in IBM Integration Designer und geben Sie den Benutzernamen und das Kennwort an.
  – Falls zusätzliche Logik erforderlich ist, verwenden Sie Java-Code in einem angepassten Mediationsbasiselement, wie im folgenden Beispiel gezeigt, um Folgendes zu erzielen:
    1. Erstellen eines HTTP-Authentifizierungsheaders
    2. Angeben der Informationen für Benutzername und Kennwort
    3. Hinzufügen des neuen HTTP-Authentifizierungsheaders zu 'HTTPControl'
    4. Einsetzen des aktualisierten Elements 'HTTPControl' zurück in den Kontextservice

```
//Abrufen von 'HeaderInfoType' aus 'contextService'
ContextService contextService = (ContextService) ServiceManager.INSTANCE
.locateService("com/ibm/bpm/context/ContextService");
HeaderInfoType headers = contextService.getHeaderInfo();
if(headers == null){
    headers = ContextObjectFactory.eINSTANCE.createHeaderInfoType();
}
//Abrufen von HTTP-Header und 'HTTP Control' von 'HeaderInfoType'
HTTPHeaderType httpHeaderType = headers.getHTTPHeader();
HTTPControl cp = httpHeaderType.getControl();
HeadersFactory factory = HeadersFactory.eINSTANCE;
if(cp == null){
    cp = factory.createHTTPControl();
}
//Erstellen von neuem 'HTTPAuthentication' und Festlegen von 'HTTPCredentials'
HTTPAuthentication authorization = factory.createHTTPAuthentication();
HTTPCredentials credentials = factory.createHTTPCredentials();
authorization.setAuthenticationType(HTTPAuthenticationType.BASIC_LITERAL);
credentials.setUserId("USERNAME");
credentials.setPassword("PASSWORD");
authorization.setCredentials(credentials);
cp.setAuthentication(authorization);
httpHeaderType.setControl(cp);
// Header-Informationen wieder in aktuellen Ausführungskontext einfügen
contextService.setHeaderInfo(headers);
```

Serverkonfiguration prüfen:

Wenn Sie eine Anwendung mit einer Web-Service-Bindung (JAX-WS) implementieren, müssen Sie sicherstellen, dass auf dem Zielserver, auf dem die Anwendung implementiert wird, nicht die Option für den **Start von Komponenten nach Bedarf** ausgewählt ist.

Sie können überprüfen, ob diese Option ausgewählt ist, indem Sie in der Administrationskonsole die folgenden Schritte ausführen:

1. Klicken Sie auf **Server** > **Servertypen** > **WebSphere-Anwendungsserver**.
2. Klicken Sie auf den Namen des Servers.
3. Prüfen Sie auf der Registerkarte 'Konfiguration', ob die Option **Komponenten nach Bedarf starten** ausgewählt ist.
4. Führen Sie einen der folgenden Schritte aus:
   - Wenn die Option **Komponenten nach Bedarf starten** ausgewählt ist, wählen Sie sie durch Entfernen der entsprechenden Markierung ab und klicken Sie dann auf **Anwenden**.
   - Wenn die Option **Komponenten nach Bedarf starten** nicht ausgewählt ist, klicken Sie auf **Abbrechen**.

Anhänge in SOAP-Nachrichten:

Sie können SOAP-Nachrichten senden und empfangen, die binäre Daten (z. B. PDF-Dateien oder JPEG-Bilder) als Anhänge enthalten. Anhänge können *referenziert* (also in der Serviceschnittstelle explizit als Nachrichtenteile dargestellt) oder *nicht referenziert* sein (in diesem Fall können beliebige Anhangsmengen und -typen eingeschlossen werden.

Ein referenzierter Anhang kann mit einem der folgenden Verfahren dargestellt werden:

- MTOM-Anhänge verwenden die SOAP-MTOM-Codierung (MTOM - (Message Transmission Optimization Mechanism) (siehe http://www.w3.org/TR/soap12-mtom/). MTOM-Anhänge werden über eine Konfigurationsoption in den Import- und Exportbindungen aktiviert und stellen die empfohlene Methode zur Codierung von Anhängen für neue Anwendungen dar.
- Als Element des Typs 'wsi:swaRef' im Nachrichtenschema.

  Anhänge, die mit dem Typ 'wsi:swaRef' definiert sind, sind mit der WSI-Spezifikation *Attachments Profile Version 1.0* (http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html) konform. Diese Spezifikation definiert, wie Nachrichtenelemente zu MIME-Teilen in Beziehung stehen.
- Als Nachrichtenteil der höchsten Ebene unter Verwendung eines binären Schematyps

  Anhänge, die als Nachrichtenteile der höchsten Ebene dargestellt werden, sind mit der Spezifikation *SOAP Messages with Attachments* (http://www.w3.org/TR/SOAP-attachments) konform.

  Anhänge, die als Nachrichtenteile der höchsten Ebene dargestellt werden, können außerdem konfiguriert werden, um sicherzustellen, dass das WSDL-Dokument und die Nachrichten, die durch die Bindung erzeugt werden, mit den WS-I-Spezifikationen *Attachments Profile Version 1.0* und *Basic Profile Version 1.1* (http://www.ws-i.org/Profiles/BasicProfile-1.1.html) konform sind.

Ein nicht referenzierter Anhang wird in einer SOAP-Nachricht ohne jegliche Darstellung im Nachrichtenschema übertragen.

In allen Fällen mit Ausnahme von MTOM-Anhängen sollte die WSDL-SOAP-Bindung eine MIME-Bindung für zu verwendende Anhänge einschließen und die maximale Größe der Anhänge sollte 20 MB nicht überschreiten.

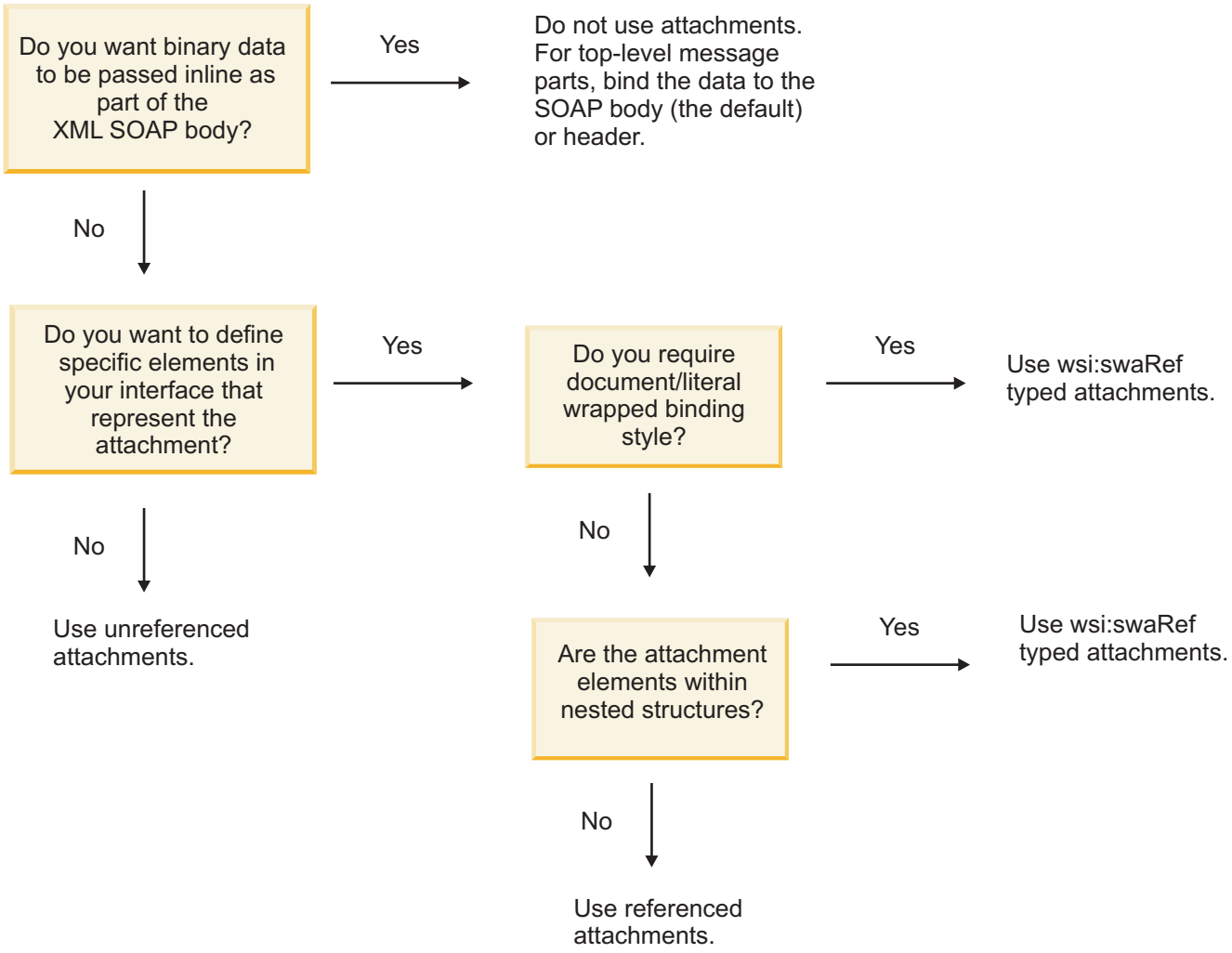
Anmerkung: Um SOAP-Nachrichten mit Anhängen zu senden oder zu empfangen, müssen Sie eine der Web-Service-Bindungen verwenden, die auf Java API for XML Web Services (JAX-WS) basieren.

Geeignete Anhangsdarstellung auswählen :

Beim Entwerfen einer neuen Serviceschnittstelle, die Binärdaten einbezieht, müssen Sie berücksichtigen, wie diese Binärdaten in den vom Service gesendeten und empfangenen SOAP-Nachrichten übertragen werden.

Für Anhänge sollte MTOM (Message Transmission Optimization Mechanism) verwendet werden, sofern die verbundene Web-Service-Anwendung MTOM unterstützt. Sollte dies nicht der Fall sein, zeigt das fol-

gende Diagramm, wie Sie andere Anhangsdarstellungen auswählen können. Ermitteln Sie die geeignete Anhangsdarstellung anhand der folgenden Fragen:



MTOM-Anhänge: Nachrichtenteile der höchsten Ebene:

Sie haben die Möglichkeit, Web-Service-Nachrichten mit SOAP-MTOM-Anhängen (MTOM - Message Transmission Optimization Mechanism) zu senden und zu empfangen. In SOAP-Nachrichten mit mehreren MIME-Teilen ist der SOAP-Hauptteil der erste Teil der Nachricht. Die Anhänge befinden sich in nachfolgenden Teilen.

Beim Senden oder Empfangen eines referenzierten Anhangs in einer SOAP-Nachricht befinden sich die Binärdaten, aus denen der (oftmals relativ große) Anhang besteht, nicht im Hauptteil der SOAP-Nachricht und müssen daher nicht als XML-Daten syntaktisch analysiert werden. Dies führt zu einer effizienteren Verarbeitung als bei Binärdaten, die sich in einem XML-Element befinden.

Beispiel für eine MTOM-SOAP-Nachricht:

```
... other transport headers ...
Content-Type: multipart/related; boundary=MIMEBoundaryurn_uuid_0FE43E4D025F0BF3DC11582467646812; type="application/xop+xm
<0.urn:uuid:0FE43E4D025F0BF3DC11582467646813@apache.org>"; start-info="text/xml"; charset=UTF-8

--MIMEBoundaryurn_uuid_0FE43E4D025F0BF3DC11582467646812
content-type: application/xop+xml; charset=UTF-8; type="text/xml";
content-transfer-encoding: binary
content-id:
```

```
   <0.urn:uuid:0FE43E4D025F0BF3DC115824676468l3@apache.org>

<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
      <soapenv:Header/>
      <soapenv:Body>
         <sendImage xmlns="http://org/apache/axis2/jaxws/sample/mtom">
            <input>
            <imageData><xop:Include xmlns:xop="http://www.w3.org/2004/08/xop/include" href="cid:1.urn:uuid:0FE43E4D025F0BF3
            </input>
         </sendImage>
      </soapenv:Body>
   </soapenv:Envelope>
--MIMEBoundaryurn_uuid_0FE43E4D025F0BF3DC11582467646812
content-type: text/plain
content-transfer-encoding: binary
content-id:
         <1.urn:uuid:0FE43E4D025F0BF3DC11582467646811@apache.org>

... binary data goes here ...
--MIMEBoundaryurn_uuid_0FE43E4D025F0BF3DC11582467646812--
```

Beachten Sie in diesem MTOM-Beispiel, dass der Inhaltstyp ('content-type') für die SOAP-Rahmenanweisung **application/xop+xml** lautet und die Binärdaten durch ein **'xop:Include'**-Element wie das folgende ersetzt werden:

```
<xop:Include xmlns:xop="http://www.w3.org/2004/08/xop/include" href="cid:1.urn:uuid:0FE43E4D025F0BF3DC115824676468l1@apache
```

Eingangsverarbeitung von referenzierten Anhängen

Wenn ein Client eine SOAP-Nachricht mit einem Anhang an eine SCA-Komponente übergibt, entfernt die Web-Service-Exportbindung (JAX-WS) zunächst den Anhang. Anschließend wird der SOAP-Teil der Nachricht syntaktisch analysiert und ein Geschäftsobjekt erstellt. Abschließend legt die Bindung die Anhangsbinärdaten im Geschäftsobjekt fest.

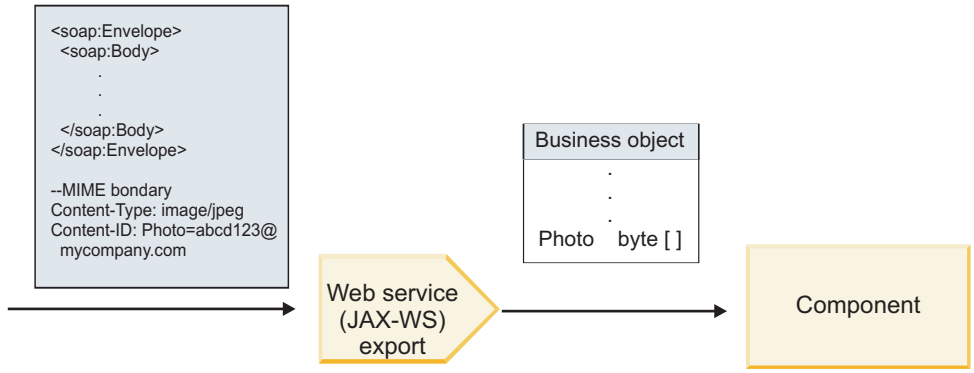


Abbildung 58. Verarbeitung einer SOAP-Nachricht mit einem referenzierten Anhang durch die Web-Service-Exportbindung (JAX-WS)

Attribute von MTOM-Anhängen

- MTOM unterstützt Anhangselemente in verschachtelten Strukturen.
- MTOM ist nur für den Typ 'base64Binary' verfügbar.
- MTOM unterstützt Anhangselemente in verschachtelten Strukturen; dies bedeutet, dass **bodyPath** für MTOM-Anhänge die **xpath**-Position für das Element darstellt, in dem sich der MTOM-Anhang befindet. Die Datenverarbeitungslogik für **bodyPath** hält sich streng an das Schema zum Generieren der **xpath**-Position (siehe folgende Beispiele):
  - Für Nicht-Array-Typen (**maxOccurs** = 1): /sendImage/input/imageData

  - Für Array-Typen (**maxOccurs** > 1): /sendImage/input/imageData[1]
- Gemischte Anhangstypen werden nicht unterstützt; dies bedeutet, dass der MTOM-Anhang generiert wird, wenn MTOM für die Importbindung aktiviert ist. Wenn MTOM inaktiviert ist oder der MTOM-Konfigurationswert die Standardeinstellung für die Exportbindung behält, wird die eingehende MTOM-Nachricht nicht unterstützt.

Referenzierte Anhänge: swaRef-typisierte Elemente:

Sie können SOAP-Nachrichten mit eingeschlossenen Anhängen senden und empfangen, die in der Serviceschnittstelle als Elemente des Typs 'swaRef' dargestellt sind.

Ein Element des Typs 'swaRef' ist in Version 1.0 der Spezifikation *Attachments Profile* von Web Services Interoperability Organization (WS-I) definiert (http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html). Diese Spezifikation legt fest, wie Nachrichtenelemente zu MIME-Teilen in Beziehung gesetzt werden.

In der SOAP-Nachricht enthält der SOAP-Hauptteil ein Element des Typs 'swaRef', das die Inhalts-ID des Anhangs angibt.

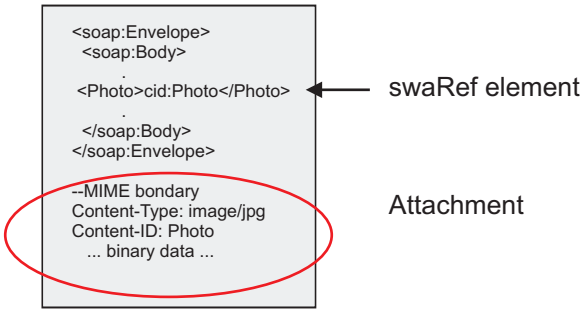


Abbildung 59. SOAP-Nachricht mit Element des Typs 'swaRef'

Die WSDL für diese SOAP-Nachricht enthält ein Element des Typs 'swaRef' innerhalb eines Nachrichtenteils, der den Anhang angibt.

```
<element name="sendPhoto">
  <complexType>
     <sequence>
       <element name="Photo" type="wsi:swaRef"/>
     </sequence>
  </complexType>
 </element>
```

Die WSDL sollte außerdem eine MIME-Bindung enthalten, die angibt, dass Nachrichten mit mehreren MIME-Teilen verwendet werden.

Anmerkung: Die WSDL enthält für das entsprechende Nachrichtenelement des Typs 'swaRef' *keine* MIME-Bindung, da MIME-Bindungen nur auf Nachrichtenteile der höchsten Ebene angewendet werden.

Anhänge, die als Elemente des Typs 'swaRef' dargestellt werden, können ausschließlich über Mediationsablaufkomponenten weitergegeben werden. Falls durch einen anderen Komponententyp auf einen Anhang zugegriffen bzw. ein Anhang weitergegeben werden muss, verwenden Sie eine Mediationsablaufkomponente, um den Anhang an eine Position zu versetzen, auf die die Komponente zugreifen kann.

Eingangsverarbeitung von Anhängen

Zum Konfigurieren einer Exportbindung für den Empfang des Anhangs verwenden Sie Integration Designer. Sie erstellen ein Modul und dessen zugehörige Schnittstelle und Operationen, inklusive einem Element des Typs 'swaRef'. Anschließend erstellen Sie eine Web-Service-Bindung (JAX-WS).

Anmerkung: Das Thema über die Arbeit mit Anhängen im Information Center von Integration Designer enthält detailliertere Informationen.

Wenn ein Client eine SOAP-Nachricht mit einem Anhang des Typs 'swaRef' an eine SCA-Komponente übergibt, entfernt die Web-Service-Exportbindung (JAX-WS) zunächst den Anhang. Anschließend wird der SOAP-Teil der Nachricht syntaktisch analysiert und ein Geschäftsobjekt erstellt. Abschließend legt die Bindung die Inhalts-ID des Anhangs im Geschäftsobjekt fest.

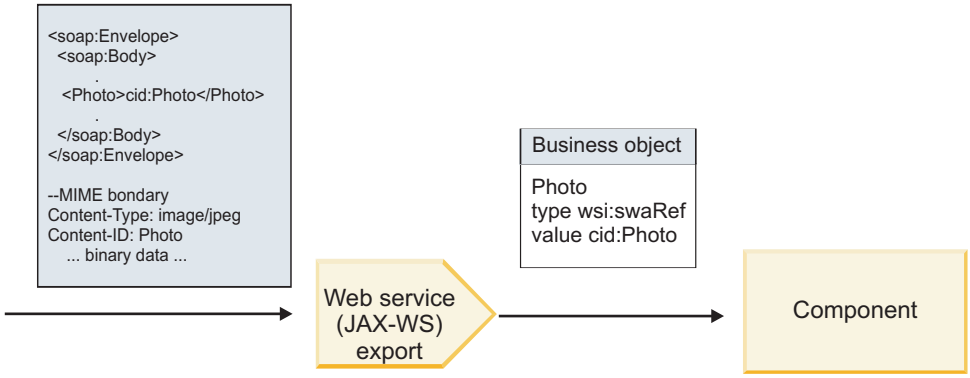


Abbildung 60. Verarbeitung einer SOAP-Nachricht mit einem Anhang des Typs 'swaRef' durch die Web-Service-Exportbindung (JAX-WS)

Auf Anhangsmetadaten in einer Mediationsablaufkomponente zugreifen

Aus Abb. 16 auf Seite 90 geht hervor, dass die Inhalts-ID des Anhangs als Element des Typs 'swaRef' dargestellt wird, wenn durch Komponenten auf Anhänge des Typs 'swaRef' zugegriffen wird.

Für jeden Anhang einer SOAP-Nachricht gibt es ein entsprechendes Element **attachments** im Servicenachrichtenobjekt. Bei Verwendung des WS-I-Typs 'swaRef' enthält das Element **attachments** den Inhaltstyp und die Inhalts-ID des Anhangs sowie die tatsächlichen Binärdaten des Anhangs.

Um den Wert eines Anhangs des Typs 'swaRef' zu erhalten, muss daher der Wert des Elements des Typs 'swaRef' abgerufen und dann nach dem Element **attachments** mit dem entsprechenden Wert für **contentID** gesucht werden. Bitte beachten Sie, dass beim Wert für **contentID** normalerweise das Präfix **cid:** aus dem Wert des Typs 'swaRef' entfernt wurde.

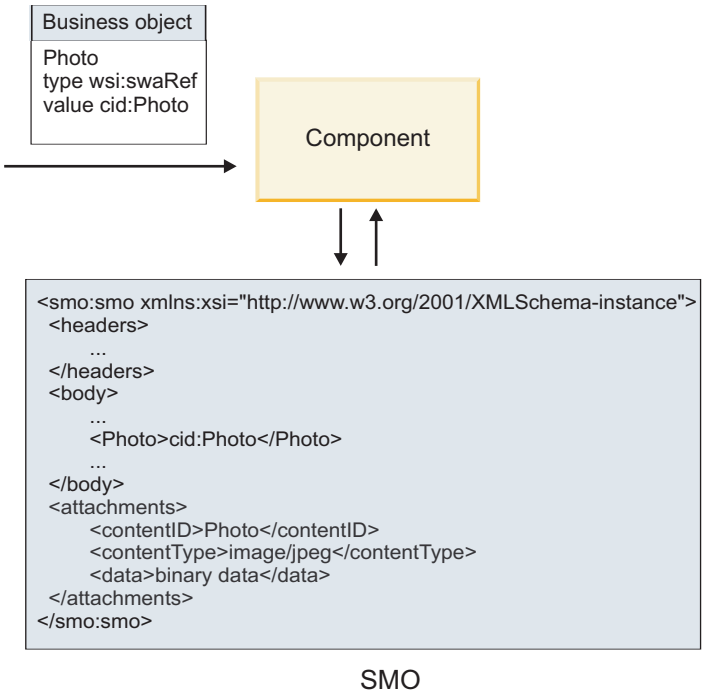


Abbildung 61. Darstellung von Anhängen des Typs 'swaRef' im Servicenachrichtenobjekt

Ausgangsverarbeitung

Zum Konfigurieren einer Web-Service-Importbindung (JAX-WS-) für den Aufruf eines externen Web-Service verwenden Sie Integration Designer. Die Importbindung wird mit einem WSDL-Dokument konfiguriert, das den aufzurufenden Web-Service beschreibt und den Anhang definiert, der an den Web-Service übergeben werden soll.

Wenn eine SCA-Nachricht von einer Web-Service-Importbindung (JAX-WS) empfangen wird, werden Elemente des Typs 'swaRef' als Anhänge gesendet, falls der Import mit einer Mediationsablaufkomponente verbunden ist und es für das Element des Typs 'swaRef' ein entsprechendes Element **attachments** gibt.

Bei der Ausgangsverarbeitung werden Elemente des Typs 'swaRef' immer mit ihren Werten für die Inhalts-ID gesendet. Das Mediationsmodul muss jedoch sicherstellen, dass es ein entsprechendes Element **attachments** mit einem übereinstimmenden Wert für **contentID** gibt.

Anmerkung: Zur Einhaltung der WS-I-Spezifikation für Anhangsprofile sollte der Wert für **content ID** auf die Codierung für den Teil 'content-id' folgen (siehe Abschnitt 3.8 der WS-I-Spezifikation *Attachments Profile 1.0*.

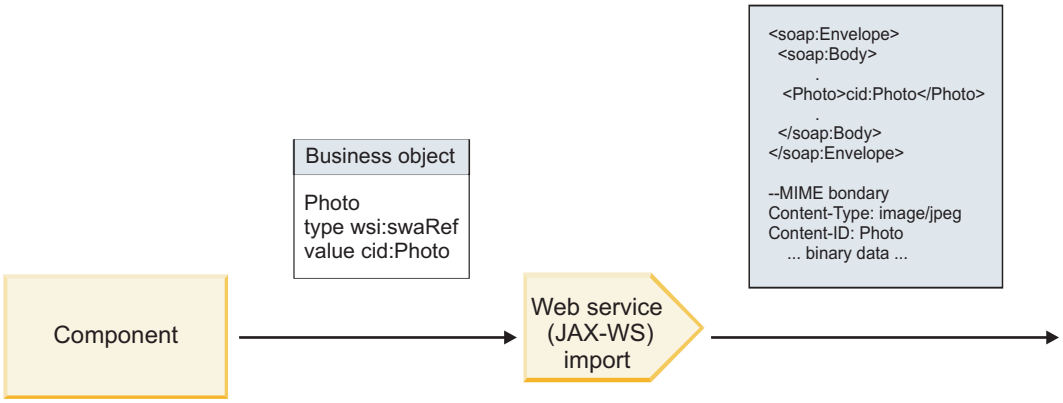


Abbildung 62. Generierung einer SOAP-Nachricht mit einem Anhang des Typs 'swaRef' durch eine Web-Service-Importbindung (JAX-WS)

Anhangsmetadaten in einer Mediationsablaufkomponente festlegen

Falls es im Servicenachrichtenobjekt einen Wert für das Element des Typs 'swaRef' und ein Element **attachments** gibt, bereitet die Bindung die SOAP-Nachricht (mit dem Anhang) vor und sendet sie an einen Empfänger.

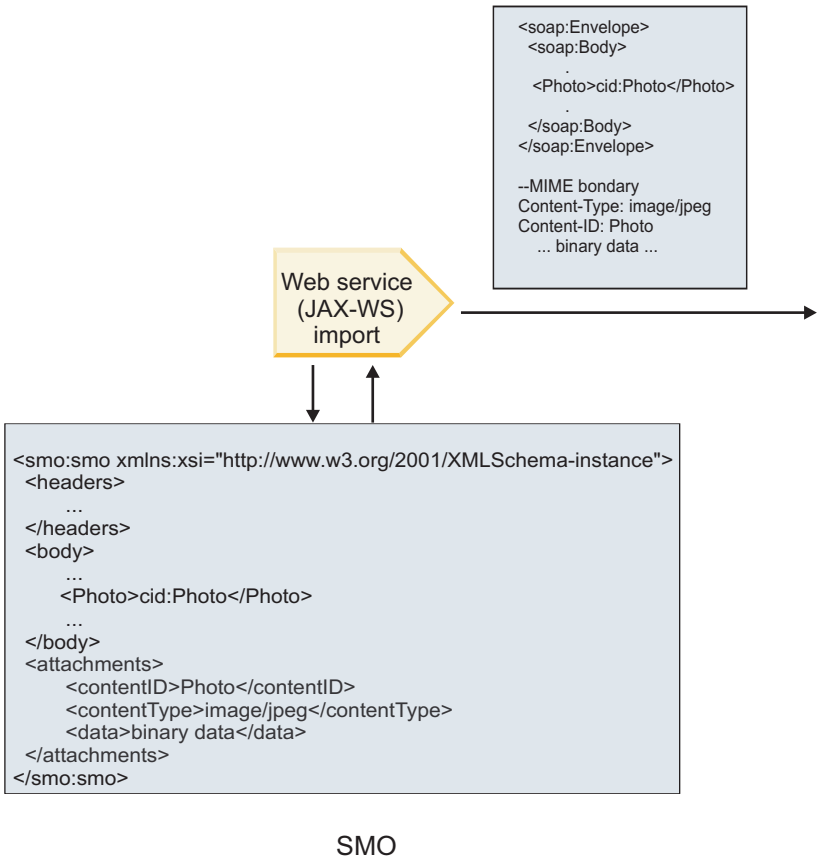


Abbildung 63. Zugriff auf Anhang des Typs 'swaRef' im Servicenachrichtenobjekt zur Erstellung der SOAP-Nachricht

Das Element **attachments** ist im Servicenachrichtenobjekt nur dann vorhanden, wenn eine Mediationsablaufkomponente direkt mit dem Import oder dem Export verbunden ist. Bei anderen Komponententypen wird das Element nicht übergeben. Falls die Werte in einem Modul benötigt werden, das andere Komponententypen enthält, sollten die Werte mit einer Mediationsablaufkomponente an eine Position kopiert

werden, an der das Modul auf sie zugreifen kann. Mit einer weiteren Mediationsablaufkomponente sollten die korrekten Werte festgelegt werden, bevor mittels eines Web-Service-Imports ein abgehender Aufruf erfolgt.

Wichtig: Wie im Thema über die XML-Darstellung des Servicenachrichtenobjekts (SMO) beschrieben, konvertiert das Mediationsbasiselement für Zuordnung Nachrichten unter Verwendung einer XSLT 1.0-Transformation. Die Transformation wird für eine XML-Serialisierung des Servicenachrichtenobjekts ausgeführt. Das Mediationsbasiselement für Zuordnung ermöglicht die Angabe des Stammelements für die Serialisierung. Das Stammelement des XML-Dokuments gibt dieses Stammelement wieder.

Wenn Sie SOAP-Nachrichten mit Anhängen senden, bestimmt das von Ihnen ausgewählte Stammelement, wie Anhänge weitergegeben werden.

- Falls Sie '/body' als Stammelement für die XML-Zuordnung verwenden, werden alle Anhänge standardmäßig über die Zuordnung weitergegeben.
- Falls Sie '/' als Stammelement der Zuordnung verwenden, können Sie die Weitergabe von Anhängen steuern.

Referenzierte Anhänge: Nachrichtenteile der höchsten Ebene:

Sie können SOAP-Nachrichten senden und empfangen, in denen binäre Anhänge enthalten sind, die als Teile in Ihrer Serviceschnittstelle deklariert sind.

In SOAP-Nachrichten mit mehreren MIME-Teilen ist der SOAP-Hauptteil der erste Teil der Nachricht. Der Anhang oder Anhänge befinden sich in nachfolgenden Teilen.

Welche Vorteile ergeben sich beim Senden und Empfangen eines referenzierten Anhangs in einer SOAP-Nachricht? Die Binärdaten, aus denen der (oftmals relativ große) Anhang besteht, werden vom Hauptteil der SOAP-Nachricht getrennt und müssen daher nicht als XML syntaktisch analysiert werden. Dies führt zu einer effizienteren Verarbeitung als bei Binärdaten, die sich in einem XML-Element befinden.

Typen von SOAP-Nachrichten mit referenzierten Anhängen

Ab Version 7.0.0.3 von IBM Business Process Manager können Sie auswählen, wie die SOAP-Nachricht generiert werden soll:

- **WS-I-konforme Nachrichten**

  Die Laufzeit kann SOAP-Nachrichten generieren, die mit *Attachments Profile Version 1.0* und *Basic Profile Version 1.1* von WS-I konform sind. In einer SOAP-Nachricht, die mit diesen Profilen konform ist, ist nur ein einziger Nachrichtenteil an den SOAP-Hauptteil gebunden. Bei den Teilen, die als Anhänge gebunden sind, wird der Anhang unter Verwendung der Codierung für den Teil 'content-id' (beschrieben in *Attachments Profile Version 1.0* von WS-I) auf den Nachrichtenteil bezogen.
- **Nicht WS-I-konforme Nachrichten**

  Die Laufzeit kann SOAP-Nachrichten generieren, die nicht mit den WS-I-Profilen konform, jedoch mit den Nachrichten kompatibel sind, die in Version 7.0 oder 7.0.0.2 von IBM Business Process Manager generiert werden. Die SOAP-Nachrichten verwenden nach dem Nachrichtenteil Elemente der höchsten Ebene mit einem Attribut **href**, das die Inhalts-ID (**content-id**) für den Anhang enthält. Die Codierung für den Teil 'content-i' (beschrieben in *Attachments Profile Version 1.0* von WS-I) wird jedoch nicht verwendet.

WS-I-Konformität für Web-Service-Exporte auswählen

Zum Konfigurieren einer Exportbindung verwenden Sie Integration Designer. Sie erstellen ein Modul und dessen zugehörige Schnittstelle und Operationen. Anschließend erstellen Sie eine Web-Service-Bindung (JAX-WS). Auf der Seite **Referenzierte Anhänge** werden alle binären Teile aus der erstellen Operation angezeigt. Sie wählen aus, bei welchen Teilen es sich um Anhänge handelt. Anschließend geben Sie auf der Seite **WS-I-Einhaltung angeben** von Integration Designer ein der folgenden Optionen an:

- **WS-I-konforme SOAP-Nachricht verwenden**
  Falls Sie diese Option auswählen, geben Sie außerdem an, welcher Nachrichtenteil an den SOAP-Hauptteil gebunden werden soll.

  Anmerkung: Diese Option kann nur verwendet werden, wenn die korrespondierende WSDL-Datei ebenfalls WS-I-konform ist.

  Eine von Integration Designer Version 7.0.0.3 generierte WSDL-Datei ist mit WS-I konform. Wenn Sie jedoch eine WSDL-Datei importieren, die nicht mit WS-I konform ist, können Sie diese Option nicht auswählen.
- **Nicht WS-I-konforme SOAP-Nachricht verwenden**
  Falls Sie diese Option auswählen, bei der es sich um die Standardoption handelt, wird der erste Nachrichtenteil an den SOAP-Hauptteil gebunden.

Anmerkung: Nur Nachrichtenteile der höchsten Ebene (also im WSDL-Element 'portType' als Teile in der Eingabe- oder Ausgabenachricht definierte Elemente), die einen binären Typ besitzen (entweder 'base64Binary' oder 'hexBinary') können als referenzierte Anhänge gesendet oder empfangen werden. Das Thema über die Arbeit mit Anhängen im Information Center von Integration Designer enthält detailliertere Informationen.

Bei WS-I-konformen Nachrichten stellt die Inhalts-ID, die in der SOAP-Nachricht generiert wird, eine Verkettung der folgenden Elemente dar:
- Wert des Attributs **name** des Elements **wsdl:part**, das von **mime:content** referenziert wird
- Zeichen **=**
- Global eindeutiger Wert, z. B. eine UUID
- Zeichen **@**
- Gültiger Domänenname

Eingangsverarbeitung von referenzierten Anhängen

Wenn ein Client eine SOAP-Nachricht mit einem Anhang an eine SCA-Komponente übergibt, entfernt die Web-Service-Exportbindung (JAX-WS) zunächst den Anhang. Anschließend wird der SOAP-Teil der Nachricht syntaktisch analysiert und ein Geschäftsobjekt erstellt. Abschließend legt die Bindung die Anhangsbinärdaten im Geschäftsobjekt fest.

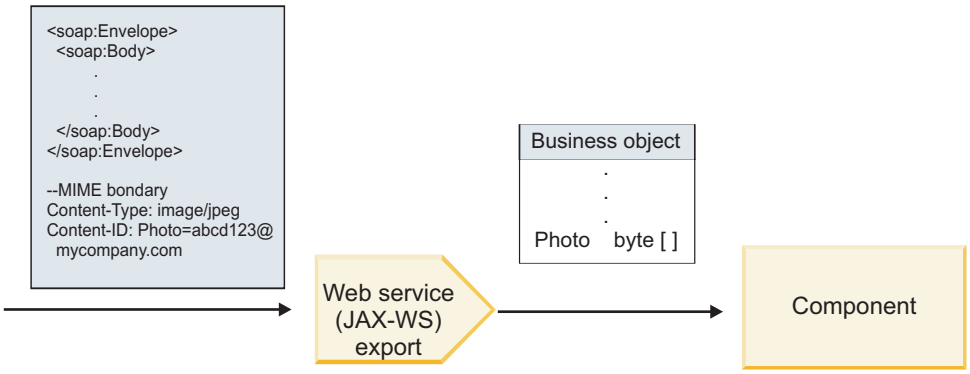


Abbildung 64. Verarbeitung einer WS-I-konformen SOAP-Nachricht mit einem referenzierten Anhang durch die Web-Service-Exportbindung (JAX-WS)

Auf Anhangsmetadaten in einer Mediationsablaufkomponente zugreifen

Aus Abb. 19 auf Seite 93 geht hervor, dass die Anhangsdaten als Byte-Array dargestellt werden, wenn durch Komponenten auf referenzierte Anhänge zugegriffen wird.

Für jeden referenzierten Anhang einer SOAP-Nachricht gibt es ein entsprechendes Element **attachments** im Servicenachrichtenobjekt. Das Element **attachments** enthält den Inhaltstyp des Anhangs und den Pfad zum Nachrichtenhauptteilelement, in dem sich der Anhang befindet.

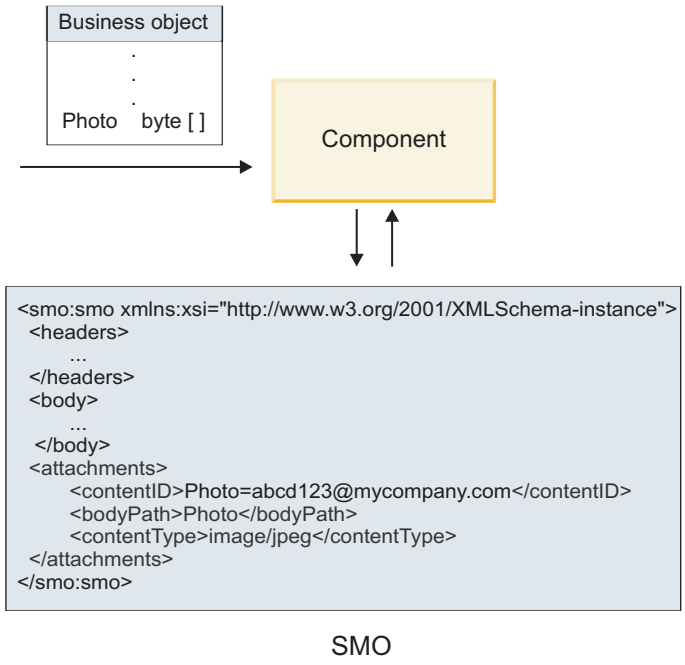


Abbildung 65. Darstellung referenzierter Anhänge im Servicenachrichtenobjekt

Wichtig: Der Pfad zum Nachrichtenhauptteilelement wird nicht automatisch aktualisiert, wenn die Nachricht transformiert und der Anhang versetzt wird. Sie können einen Mediationsablauf verwenden, um das Element **attachments** mit dem neuen Pfad zu aktualisieren (beispielsweise im Rahmen der Transformation oder durch die Verwendung eines separaten Setter-Nachrichtenelements).

Erstellung von abgehenden SOAP-Nachrichten

Zum Konfigurieren einer Web-Service-Importbindung (JAX-WS-) für den Aufruf eines externen Web-Service verwenden Sie Integration Designer. Die Importbindung wird mit einem WSDL-Dokument konfiguriert, das den aufzurufenden Web-Service beschreibt und definiert, welche Nachrichtenteile als Anhänge übergeben werden sollen. Auf der Seite **WS-I-Einhaltung angeben** von Integration Designer können Sie außerdem eine der folgenden Optionen angeben:

- **WS-I-konforme SOAP-Nachricht verwenden**

  Falls Sie diese Option auswählen, geben Sie außerdem an, welcher Nachrichtenteil an den SOAP-Hauptteil gebunden werden soll. Alle anderen Teile werden an Anhänge oder Header gebunden. Nachrichten, die an die Bindung gesendet werden, enthalten im SOAP-Hauptteil keine Elemente, die die Anhänge referenzieren. Die Beziehung wird durch die Inhalt-ID des Anhangs ausgedrückt, die den Namen des Nachrichtenteils enthält.
- **Nicht WS-I-konforme SOAP-Nachricht verwenden**

  Falls Sie diese Option auswählen, bei der es sich um die Standardoption handelt, wird der erste Nachrichtenteil an den SOAP-Hauptteil gebunden. Alle anderen Teile werden an Anhänge oder Header ge-

bunden. Nachrichten, die durch die Bindung gesendet werden, enthalten im SOAP-Hauptteil eines oder mehrere Elemente, die die Anhänge mittels eines Attributs **href** referenzieren.

Anmerkung: Der Teil, der einen Anhang darstellt (wie in WSDL definiert) muss einen einfachen Typ besitzen (entweder 'base64Binary' oder 'hexBinary'). Falls ein Teil durch einen komplexen Typ definiert wird, kann er nicht als Anhang gebunden werden.

Ausgangsverarbeitung von referenzierten Anhängen

Die Importbindung ermittelt anhand von Informationen im Servicenachrichtenobjekt, wie die binären Nachrichtenteile der höchsten Ebene als Anhänge gesendet werden.

Abbildung 66. Zugriff auf den referenzierten Anhang im Servicenachrichtenobjekt zur Erstellung der SOAP-Nachricht

Das Element **attachments** ist im Servicenachrichtenobjekt nur dann vorhanden, wenn eine Mediationsablaufkomponente direkt mit dem Import oder dem Export verbunden ist. Bei anderen Komponententypen wird das Element nicht übergeben. Falls die Werte in einem Modul benötigt werden, das andere Komponententypen enthält, sollten die Werte mit einer Mediationsablaufkomponente an eine Position kopiert werden, an der das Modul auf sie zugreifen kann. Mit einer weiteren Mediationsablaufkomponente sollten die korrekten Werte festgelegt werden, bevor mittels eines Web-Service-Imports ein abgehender Aufruf erfolgt.

Die Bindung ermittelt anhand einer Kombination der folgenden Bedingungen, wie und ob die Nachricht gesendet wird:

- Gibt es für den binären Nachrichtenteil der höchsten Ebene eine WSDL-MIME-Bindung und, wenn ja, wie ist der Inhaltstyp definiert?
- Gibt es im Servicenachrichtenobjekt ein Element **attachments**, dessen Wert für **bodyPath** einen binären Teil der höchsten Ebene referenziert?

Erstellung von Anhängen bei vorhandenem Element attachment im Servicenachrichtenobjekt

Die folgende Tabelle zeigt, wie ein Anhang erstellt und gesendet wird, wenn das Servicenachrichtenobjekt ein Element **attachment** mit einem Wert für **bodyPath** enthält, der mit einem Nachrichtennamensteil übereinstimmt:

Tabelle 61. Generierung des Anhangs

Status der WSDL-MIME-Bindung für binären Nachrichtenteil der höchsten Ebene	Verfahren beim Erstellen und Senden der Nachricht
Mit einer der folgenden Bedingungen vorhanden: • Kein definierter Inhaltstyp für den Nachrichtenteil • Mehrere definierte Inhaltstypen • Definierter Platzhalter für den Inhaltstyp	Der Nachrichtenteil wird als Anhang gesendet. Der Wert für **Content-Id** wird auf den Wert im Element 'attachments' gesetzt, wenn dieses vorhanden ist; andernfalls wird ein Wert generiert. Der Wert für **Content-Type** wird auf den Wert im Element 'attachments' gesetzt, wenn dieses vorhanden ist; andernfalls wird er auf **application/octet-stream** gesetzt.
Mit Inhalt für Nachrichtenteil (kein Platzhalter) vorhanden	Der Nachrichtenteil wird als Anhang gesendet. Der Wert für **Content-Id** wird auf den Wert im Element 'attachments' gesetzt, wenn dieses vorhanden ist; andernfalls wird ein Wert generiert. Der Wert für **Content-Type** wird auf den Wert im Element 'attachments' gesetzt, wenn dieses vorhanden ist; andernfalls wird er auf den Typ gesetzt, der im WSDL-MIME-Inhaltselement definiert ist.
Nicht vorhanden	Der Nachrichtenteil wird als Anhang gesendet. Der Wert für **Content-Id** wird auf den Wert im Element 'attachments' gesetzt, wenn dieses vorhanden ist; andernfalls wird ein Wert generiert. Der Wert für **Content-Type** wird auf den Wert im Element 'attachments' gesetzt, wenn dieses vorhanden ist; andernfalls wird er auf **application/octet-stream** gesetzt. **Anmerkung:** Das Senden von Nachrichtenteilen als Anhänge, die als solche nicht in WSDL definiert sind, kann die Einhaltung von 'WS-I Attachments Profile 1.0' aufheben und sollte daher nach Möglichkeit vermieden werden.

Erstellung von Anhängen ohne Element attachment im Servicenachrichtenobjekt

Die folgende Tabelle zeigt, wie ein Anhang erstellt und gesendet wird, wenn das Servicenachrichtenobjekt kein Element **attachment** mit einem Wert für **bodyPath** enthält, der mit einem Nachrichtennamensteil übereinstimmt:

Tabelle 62. Generierung des Anhangs

Status der WSDL-MIME-Bindung für binären Nachrichtenteil der höchsten Ebene	Verfahren beim Erstellen und Senden der Nachricht
Mit einer der folgenden Bedingungen vorhanden: • Kein definierter Inhaltstyp für den Nachrichtenteil • Mehrere definierte Inhaltstypen • Definierter Platzhalter für den Inhaltstyp	Der Nachrichtenteil wird als Anhang gesendet. Der Wert für **Content-Id** wird generiert. Der Wert für **Content-Type** wird auf **application/octet-stream** gesetzt.

Tabelle 62. Generierung des Anhangs (Forts.)

Status der WSDL-MIME-Bindung für binären Nachrichtenteil der höchsten Ebene	**Verfahren beim Erstellen und Senden der Nachricht**
Mit Inhalt für Nachrichtenteil (kein Platzhalter) vorhanden	Der Nachrichtenteil wird als Anhang gesendet. Der Wert für **Content-Id** wird generiert. Der Wert für **Content-Type** wird auf den im WSDL-MIME-Inhaltselement definierten Typ gesetzt.
Nicht vorhanden	Der Nachrichtenteil wird nicht als Anhang gesendet.

Wichtig: Wie im Thema über die XML-Darstellung des Servicenachrichtenobjekts (SMO) beschrieben, konvertiert das Mediationsbasiselement für Zuordnung Nachrichten unter Verwendung einer XSLT 1.0-Transformation. Die Transformation wird für eine XML-Serialisierung des Servicenachrichtenobjekts ausgeführt. Das Mediationsbasiselement für Zuordnung ermöglicht die Angabe des Stammelements für die Serialisierung. Das Stammelement des XML-Dokuments gibt dieses Stammelement wieder.

Wenn Sie SOAP-Nachrichten mit Anhängen senden, bestimmt das von Ihnen ausgewählte Stammelement, wie Anhänge weitergegeben werden.

- Falls Sie '/body' als Stammelement für die XML-Zuordnung verwenden, werden alle Anhänge standardmäßig über die Zuordnung weitergegeben.
- Falls Sie '/' als Stammelement der Zuordnung verwenden, können Sie die Weitergabe von Anhängen steuern.

Nicht referenzierte Anhänge:

Sie können *nicht referenzierte* Anhänge senden und empfangen, die nicht in der Serviceschnittstelle deklariert sind.

In einer SOAP-Nachricht mit mehreren MIME-Teilen ist der SOAP-Hauptteil der erste Teil der Nachricht. Die Anhänge befinden sich in nachfolgenden Teilen. Der SOAP-Hauptteil enthält keine Referenz für den Anhang.

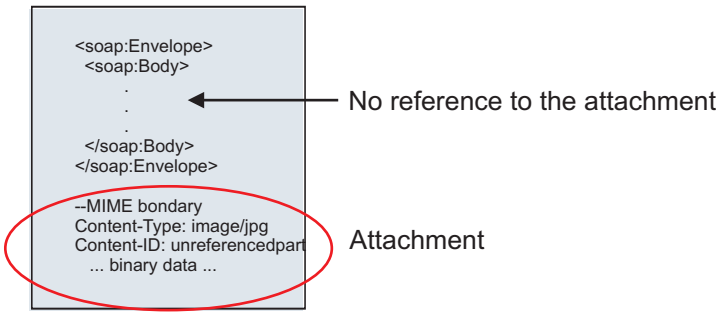


Abbildung 67. SOAP-Nachricht mit nicht referenziertem Anhang

Sie können eine SOAP-Nachricht mit einem nicht referenzierten Anhang über einen Web-Service-Export an einen Web-Service-Import senden. Die Ausgabenachricht, die an den Ziel-Web-Service gesendet wird, enthält den Anhang.

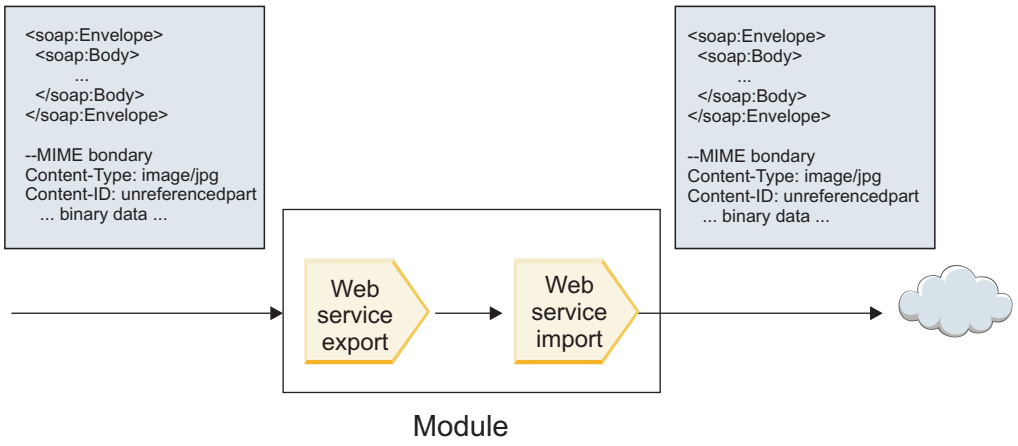


Abbildung 68. Übergabe eines Anhangs durch ein SCA-Modul

In Abb. 23 auf Seite 98 wird die SOAP-Nachricht mit dem Anhang unverändert übergeben.

Durch die Verwendung einer Mediationsablaufkomponente können Sie die SOAP-Nachricht auch ändern. Beispielsweise können Sie mit der Mediationsablaufkomponente Daten aus der SOAP-Nachricht extrahieren (in diesem Fall Binärdaten im Hauptteil der Nachricht) und eine SOAP-Nachricht mit Anhängen erstellen. Die Daten werden als Teil des Anhangselements eines Servicenachrichtenobjekts verarbeitet.

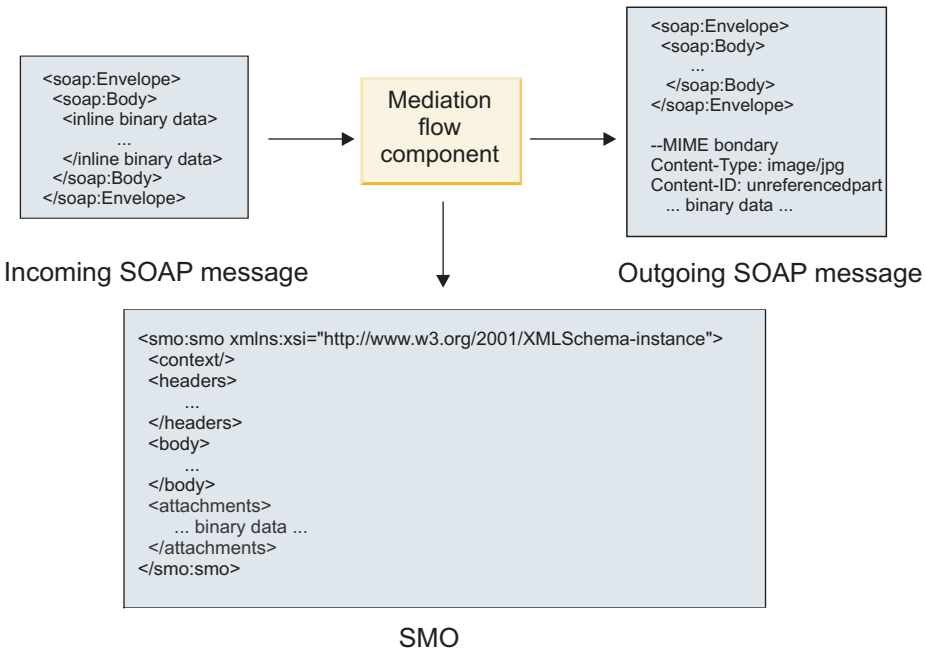


Abbildung 69. Durch eine Mediationsablaufkomponente verarbeitete Nachricht

In der entgegengesetzten Richtung kann die Mediationsablaufkomponente die eingehende Nachricht transformieren, indem sie den Anhang extrahiert und codiert und die Nachricht dann ohne Anhänge überträgt.

Statt Daten aus einer eingehenden SOAP-Nachricht zu extrahieren, um eine SOAP-Nachricht mit Anhängen zu bilden, können Sie die Anhangsdaten von einer fernen Quelle wie beispielsweise einer Datenbank beziehen.

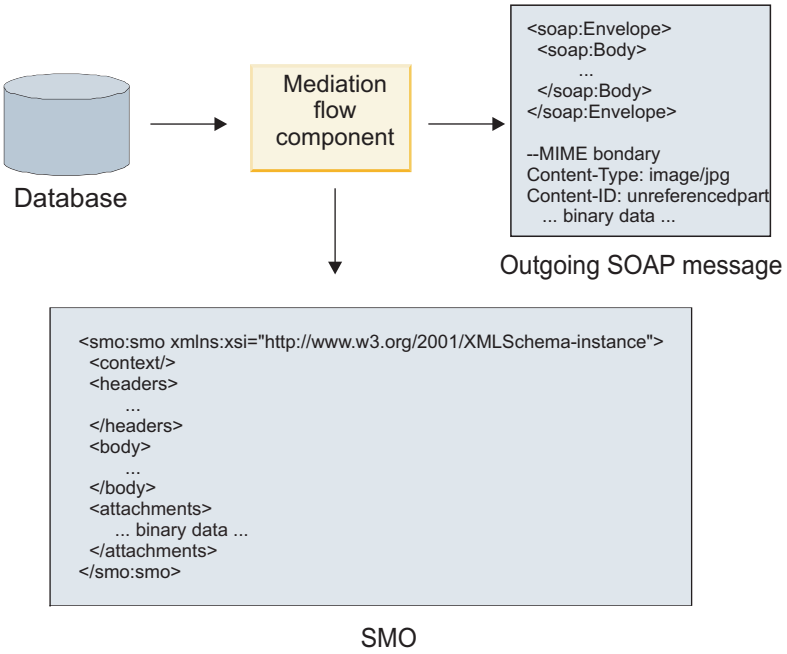


Abbildung 70. Aus einer Datenbank bezogener und zur SOAP-Nachricht hinzugefügter Anhang

In der entgegengesetzten Richtung kann die Mediationsablaufkomponente den Anhang aus einer eingehenden SOAP-Nachricht extrahieren und die Nachricht verarbeiten (z. B. den Anhang in einer Datenbank speichern).

Nicht referenzierte Anhänge können nur über Mediationsablaufkomponenten weitergegeben werden. Falls durch einen anderen Komponententyp auf einen Anhang zugegriffen bzw. ein Anhang weitergegeben werden muss, verwenden Sie eine Mediationsablaufkomponente, um den Anhang an eine Position zu versetzen, auf die die Komponente zugreifen kann.

Wichtig: Wie im Thema über die XML-Darstellung des Servicenachrichtenobjekts (SMO) beschrieben, konvertiert das Mediationsbasiselement für Zuordnung Nachrichten unter Verwendung einer XSLT 1.0-Transformation. Die Transformation wird für eine XML-Serialisierung des Servicenachrichtenobjekts ausgeführt. Das Mediationsbasiselement für Zuordnung ermöglicht die Angabe des Stammelements für die Serialisierung. Das Stammelement des XML-Dokuments gibt dieses Stammelement wieder.

Wenn Sie SOAP-Nachrichten mit Anhängen senden, bestimmt das von Ihnen ausgewählte Stammelement, wie Anhänge weitergegeben werden.

- Falls Sie '/body' als Stammelement für die XML-Zuordnung verwenden, werden alle Anhänge standardmäßig über die Zuordnung weitergegeben.
- Falls Sie '/' als Stammelement der Zuordnung verwenden, können Sie die Weitergabe von Anhängen steuern.

Verwendung der WSDL-Dokumentdarstellungsbindung mit mehrteiligen Nachrichten:

Die Organisation 'Web Services Interoperability Organization' (WS-I) hat eine Reihe von Regeln dafür definiert, wie Web-Services mittels WSDL beschrieben werden sollten und wie die entsprechenden SOAP-Nachrichten gebildet werden sollten, um eine Interoperabilität sicherzustellen.

Diese Regeln sind in der WS-I-Spezifikation *Basic Profile Version 1.1* angegeben (http://www.ws-i.org/Profiles/BasicProfile-1.1.html). Insbesondere gibt der Abschnitt 'R2712' der WS-I-Spezifikation 'Basic Profile 1.1' Folgendes an: "A document-literal binding MUST be serialized as an ENVELOPE with a soap:Body whose child element is an instance of the global element declaration referenced by the corresponding

wsdl:message part." (Eine Dokumentliteralbindung MUSS als Element ENVELOPE mit einem Element 'soap:Body' serialisiert werden, dessen untergeordnetes Element eine Instanz der globalen Elementdeklaration ist, die durch den entsprechenden Teil 'wsdl:message' referenziert wird.)

Dies bedeutet, dass bei Verwendung einer SOAP-Dokumentdarstellungsbindung für eine Operation mit Nachrichten (Eingabe-, Ausgabe- oder Fehlernachrichten), die mit mehreren Teilen definiert sind, nur einer einziger dieser Teile an den SOAP-Hauptteil gebunden werden darf, damit die WS-I-Spezifikation 'Basic Profile 1.1' eingehalten wird.

Desweiteren besagt der Abschnitt 'R2941' der WS-I-Spezifikation 'Attachments Profile 1.0' Folgendes: "A wsdl:binding in a DESCRIPTION SHOULD bind every wsdl:part of a wsdl:message in the wsdl:portType to which it refers to one of soapbind:body, soapbind:header, soapbind:fault , soapbind:headerfault, or mime:content." (Ein Element 'wsdl:binding' in einem Element DESCRIPTION sollte jedes Element 'wsdl-:part' eines Elements 'wsdl:message' im Element 'wsdl:portType', das es referenziert, an eines der folgenden Elemente binden: 'soapbind:body', 'soapbind:header', 'soapbind:fault', 'soapbind:headerfault' oder 'mime:content'.).

Dies bedeutet, dass bei Verwendung einer SOAP-Dokumentdarstellungsbindung für eine Operation mit Nachrichten (Eingabe-, Ausgabe- oder Fehlernachrichten), die mit mehreren Teilen definiert sind, alle anderen Teile als derjenige Teil, der an den SOAP-Hauptteil gebunden ist, als Anhänge oder Header gebunden werden müssen.

Beim Generieren von WSDL-Beschreibungen für Exporte mit Web-Service-Bindungen (JAX-WS und JAX-RPC) in diesem Fall wird das folgende Verfahren verwendet:

- Sie können auswählen, welcher Nachrichtenteil an den SOAP-Hauptteil gebunden wird, wenn es mehrere Element nicht binären Typs gibt. Gibt es lediglich ein einziges Element binären Typs, wird automatisch dieses Element an den SOAP-Hauptteil gebunden.
- Bei der JAX-WS-Bindung werden alle anderen Nachrichtenteile des Typs 'hexBinary' oder 'base64Binary' als referenzierte Anhänge gebunden. Weitere Informationen hierzu finden Sie im Abschnitt „Referenzierte Anhänge: Nachrichtenteile der höchsten Ebene” auf Seite 92.
- Alle anderen Nachrichtenteile werden als SOAP-Header gebunden.

Die JAX-RPC- und JAX-WS-Importbindungen berücksichtigen die SOAP-Bindung in einem vorhandenen WSDL-Dokument mit mehrteiligen Dokumentdarstellungsnachrichten auch dann, wenn durch sie mehrere Teile an den SOAP-Hauptteil gebunden werden. Für solche WSDL-Dokumente können jedoch in Rational Application Developer keine Web-Service-Clients generiert werden.

Anmerkung: Die JAX-RPC-Bindung unterstützt Anhänge nicht.

Bei der Verwendung mehrteiliger Nachrichten mit einer Operation, die eine SOAP-Dokumentdarstellungsbindung besitzt, wird daher das folgende Muster empfohlen:

1. Verwenden Sie die Darstellung mit eingeschlossenem Dokumentliteral. In diesem Fall besitzen Nachrichten immer einen einzigen Teil. Anhänge müssen in diesem Fall jedoch entweder nicht referenziert sein (siehe „Nicht referenzierte Anhänge” auf Seite 97) oder den Typ 'swaRef' aufweisen (siehe „Referenzierte Anhänge: swaRef-typisierte Elemente” auf Seite 88).
2. Verwenden Sie die Darstellung mit RPC/Literal. In diesem Fall bestehen für die WSDL-Bindung keine Einschränkungen hinsichtlich der Anzahl der Teile, die an den SOAP-Hauptteil gebunden sind. Die resultierende SOAP-Nachricht hat immer ein einziges untergeordnetes Element, das die aufzurufende Operation darstellt; die Nachrichtenteile sind untergeordnete Elemente dieses Elements.
3. Bei der JAX-WS-Bindung sollte höchstens ein einziger Nachrichtenteil verwendet werden, der nicht den Typ 'hexBinary' oder 'base64Binary' aufweist, sofern es nicht akzeptabel ist, die anderen nicht binären Teile an SOAP-Header zu binden.
4. Alle anderen Fälle unterliegen dem beschriebenen Verhalten.

Anmerkung: Zusätzliche Einschränkungen ergeben sich, wenn Sie SOAP-Nachrichten verwenden, die nicht mit der WS-I-Spezifikation *Basic Profile Version 1.1* konform sind.

- Der erste Nachrichtenteil sollte nicht binär sein.
- Beim Empfang von mehrteiligen SOAP-Dokumentdarstellungsnachrichten mit referenzierten Anhängen erwartet die JAX-WS-Bindung, dass jeder referenzierte Anhang als untergeordnetes Element des SOAP-Hauptteils mit einem Wert für das Attribut 'href' dargestellt ist, der den Anhang durch dessen Inhalts-ID angibt. Die JAX-WS-Bindung sendet referenzierte Anhänge für solche Nachrichten auf dieselbe Weise. Dieses Verhalten ist nicht mit der WS-I-Spezifikation 'Basic Profile' kompatibel.

  Um sicherzustellen, dass Ihre Nachrichten die Spezifikation 'Basic Profile' einhalten, verwenden Sie das Verfahren 1 auf Seite 100 oder 2 auf Seite 100 in der vorstehenden Liste bzw. vermeiden Sie die Verwendung von referenzierten Anhängen für solche Nachrichten und verwenden Sie stattdessen nicht referenzierte Anhänge oder Anhänge des Typs 'swaRef'.

HTTP-Bindungen

Die HTTP-Bindung ist so konzipiert, dass sie für HTTP die Konnektivität von Service Component Architecture (SCA) bereitstellt. Infolgedessen können bestehende oder neu entwickelte HTTP-Anwendungen in Umgebungen mit serviceorientierter Architektur (Service Oriented Architecture - SOA) eingesetzt werden.

Das Hypertext Transfer Protocol (HTTP) ist ein weit verbreitetes Protokoll für die Datenübertragung im World Wide Web. Für die Arbeit mit einer externen Anwendung, die das HTTP-Protokoll verwendet, ist eine HTTP-Bindung erforderlich. Mithilfe der HTTP-Bindung werden die Daten, die in einer Nachricht in einem nativen Format weitergegeben werden, in ein Geschäftsobjekt in einer SCA-Anwendung transformiert. Von der HTTP-Bindung können auch Daten, die als Geschäftsobjekt übergeben wurden, in das native Format umgewandelt werden, das von der externen Anwendung erwartet wird.

Anmerkung: Wenn Sie mit Clients und Services interagieren wollen, die das SOAP-/HTTP-Protokoll für Web-Services verwenden, kann es sinnvoll sein, eine der Web-Service-Bindungen zu verwenden, die in Bezug auf die Behandlung der Standardservicequalitäten für Web-Services zusätzliche Funktionalität bieten.

Nachstehend sind einige allgemeine Szenarios für die Verwendung der HTTP-Bindung beschrieben:

- Services mit SCA-Basis können HTTP-Anwendungen mithilfe eines SCA-Imports aufrufen.
- Services mit SCA-Basis können sich selbst als HTTP-fähige Anwendungen zugänglich machen, damit sie (über einen HTTP-Export) von HTTP-Clients verwendet werden können.
- IBM Business Process Manager und Process Server können über eine HTTP-Infrastruktur miteinander kommunizieren. Benutzer können daher ihre Kommunikation gemäß der unternehmensweiten Standards verwalten.
- IBM Business Process Manager und Process Server können als Mediatoren der HTTP-Kommunikation agieren, indem sie Nachrichten transformieren und weiterleiten. Dies verbessert die Integration von Anwendungen bei Verwendung eines HTTP-Netzes.
- IBM Business Process Manager und Process Server können als Brücke zwischen HTTP und anderen Protokollen wie SOAP/HTTP-Web-Services Java Connector Architecture (JCA)-basierte Ressourcenadapter, JMS usw. eingesetzt werden.

Ausführliche Informationen zum Erstellen von HTTP-Importbindungen und -Exportbindungen finden Sie im Information Center von Integration Designer. Lesen Sie dort die Themen unter **Integrationsanwendungen entwickeln** > **Mit HTTP auf externe Services zugreifen**.

Übersicht über HTTP-Bindungen:

Eine HTTP-Bindung stellt Konnektivität für Anwendungen mit HTTP-Hosting bereit. Sie nimmt eine Meditation der Kommunikation zwischen HTTP-Anwendungen vor und ermöglicht den Aufruf vorhandener HTTP-basierter Anwendungen aus einem Modul heraus.

HTTP-Importbindungen

Die HTTP-Importbindung stellt eine abgehende Konnektivität von SCA-Anwendungen zu einem HTTP-Server oder HTTP-Anwendungen bereit.

Der Import ruft eine HTTP-Endpunkt-URL auf. Zur Angabe der URL gibt es drei Verfahren:

- Die URL kann in den HTTP-Headern durch die URL für dynamisches Überschreiben dynamisch festgelegt werden.
- Die URL kann im Zieladressenelement des Servicenachrichtenobjekts dynamisch festgelegt werden.
- Die URL kann als Konfigurationseigenschaft für den Import angegeben werden.

Dieser Aufruf erfolgt immer synchron.

Obwohl HTTP-Aufrufe immer Anforderungs-/Antwortaufrufe sind, unterstützt der HTTP-Import sowohl unidirektionale als auch bidirektionale Operationen und ignoriert bei einer unidirektionalen Operation die Antwort.

HTTP-Exportbindungen

Die HTTP-Exportbindung stellt die eingehende Konnektivität von HTTP-Anwendungen zu einer SCA-Anwendung bereit.

Im HTTP-Export ist eine URL definiert. HTTP-Anwendungen, die Anforderungsnachrichten an den Export senden wollen, rufen den Export über diese URL auf.

Der HTTP-Export unterstützt auch Pingbefehle.

HTTP-Bindungen zur Laufzeit

Im Verlauf eines Imports mit einer HTTP-Bindung zur Laufzeit wird eine Anforderung mit oder ohne Daten im Hauptteil der Nachricht von der SCA-Anwendung an den externen Web-Service gesendet. Die Anforderung wird von der SCA-Anwendung an den externen Web-Service ausgegeben (siehe Abb. 26 auf Seite 102).

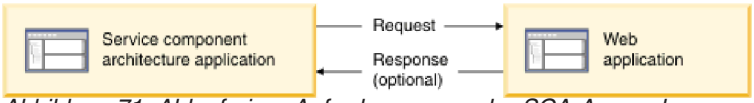


Abbildung 71. Ablauf einer Anforderung von der SCA-Anwendung an die Webanwendung

Optional können beim Import mit der HTTP-Bindung Daten von der Webanwendung in einer Antwort an die Anforderung zurückgesendet werden.

Im Verlauf eines Exports wird die Anforderung von einer Clientanwendung an einen Web-Service gesendet (siehe Abb. 27 auf Seite 102).

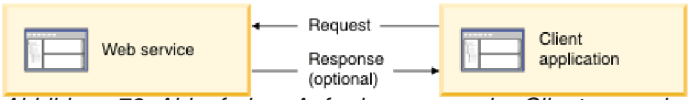


Abbildung 72. Ablauf einer Anforderung von der Clientanwendung an den Web-Service

Der Web-Service ist eine Webanwendung, die auf dem Server ausgeführt wird. Der Export wird in diese Webanwendung als Servlet implementiert, der Client sendet also seine Anforderung an eine URL-Adresse. Vom Servlet wird die Anforderung zur Laufzeit an die SCA-Anwendung weitergeleitet.

Optional können im Verlauf des Exports Daten als Antwort auf die Anforderung an die Clientanwendung gesendet werden.

HTTP-Header:

Bei HTTP-Importbindungen und -Exportbindungen können die HTTP-Header und deren Werte, die für abgehende Nachrichten verwendet werden, konfiguriert werden. Der HTTP-Import verwendet diese Header für Anforderungen. Der HTTP-Export verwendet sie für Antworten.

Statisch konfigurierte Header und Steuerinformationen haben Vorrang vor anderen Werten, die während der Laufzeit dynamisch festgelegt werden. Die Steuerwerte für die URL für dynamisches Überschreiben, die Version und die Methode überschreiben jedoch die statischen Werte, die ansonsten als Standardwerte betrachtet werden.

Die Bindung unterstützt die dynamische Spezifik der HTTP-Import-URL, indem der Wert für die HTTP-Ziel-URL, die Version und die Methode zur Laufzeit ermittelt werden. Zur Ermittlung dieser Werte wird der Wert für die Endpunktreferenz, die URL für dynamisches Überschreiben, die Version und die Methode extrahiert.

- Verwenden Sie für die Endpunktreferenz die APIs 'com.ibm.websphere.sca.addressing.EndpointReference' oder legen Sie das Feld '/headers/SMOHeader/Target/address' im Servicenachrichtenobjektheader fest.
- Für die URL für dynamisches Überschreiben, die Version und die Methode verwenden Sie den Abschnitt mit den HTTP-Steuerparametern in der SCA-Nachricht. Bitte beachten Sie, dass der Wert für die URL für dynamisches Überschreiben Vorrang vor der Zielendpunktreferenz hat. Die Endpunktreferenz gilt jedoch bindungsübergreifend. Sie stellt damit die bevorzugte Methode dar und sollte nach Möglichkeit verwendet werden.

Die Steuer- und Headerinformationen für abgehende Nachrichten unter HTTP-Exportbindungen und -Importbindungen werden in der nachstehenden Reihenfolge verarbeitet:

1. Header- und Steuerinformationen ohne HTTP-Wert für die URL für dynamisches Überschreiben, Version und Methode aus der SCA-Nachricht (niedrigste Priorität)
2. Änderungen aus der Administrationskonsole auf Export-/Importebene
3. Änderungen aus der Administrationskonsole auf der Methodenebene des Exports oder Imports
4. Zieladresse (angegeben durch Endpunktreferenz oder Servicenachrichtenobjektheader)
5. Wert für die URL für dynamisches Überschreiben, Version und Methode aus der SCA-Nachricht
6. Header und Steuerinformationen aus dem Datenhandler oder der Datenbindung (höchste Priorität)

Der HTTP-Export und -Import füllt Header und Steuerparameter in eingehender Richtung nur dann mit Daten aus der eingehenden Nachricht (HTTPExportRequest und HTTPImportResponse), wenn die Einstellung für die Weiterleitung des Protokollheaders auf **True** gesetzt ist. Analog lesen und verarbeiten der HTTP-Export und -Import abgehende Header und Steuerparameter (HTTPExportResponse und HTTPImportRequest) nur dann, wenn die Weiterleitung des Protokollheaders auf **True** gesetzt ist.

Anmerkung: Datenhandler- oder Datenbindungsänderungen an Headern oder Steuerparametern in der Importantwort oder der Exportanforderung ändern die Verarbeitungsanweisungen der Nachricht innerhalb der Import- oder Exportbindung nicht und sollten ausschließlich zu dem Zweck eingesetzt werden, geänderte Werte an nachgelagerte SCA-Komponenten weiterzugeben.

Der Kontextservice ist für die Weitergabe des Kontextes (inklusive der Protokollheader wie dem HTTP-Header und des Benutzerkontextes wie der Account-ID) über einen SCA-Aufrufpfad zuständig. Während der Entwicklung in IBM Integration Designer können Sie die Weitergabe des Kontextes mittels Import- und Exporteigenschaften steuern. Weitere Details enthalten die Angaben über Import- und Exportbindungen im Information Center von IBM Integration Designer.

Bereitgestellte HTTP-Headerstrukturen und Unterstützung

In Tabelle 34 auf Seite 104 sind die Elemente für die Anforderungs-/Antwortparameter bei Anforderungen und Antworten von HTTP-Importen und -Exporten angegeben.

Tabelle 63. Bereitgestellte HTTP-Headerinformationen

Steuername	HTTP-Importanforderung	HTTP-Importantwort	HTTP-Exportanforderung	HTTP-Exportantwort
URL	Wird ignoriert.	Nicht festgelegt.	Wird aus der Anforderungsnachricht gelesen. **Anmerkung:** Die Abfragezeichenfolge ist ebenfalls Bestandteil des URL-Steuerparameters.	Wird ignoriert.
Version (mögliche Werte: 1.0, 1.1; Standardwert ist 1.1)	Wird ignoriert.	Nicht festgelegt.	Wird aus der Anforderungsnachricht gelesen.	Wird ignoriert.
Methode	Wird ignoriert.	Nicht festgelegt.	Wird aus der Anforderungsnachricht gelesen.	Wird ignoriert.
URL für dynamisches Überschreiben	Überschreibt die HTTP-Import-URL, falls im Datenhandler oder in der Datenbindung festgelegt. Wird in der Anforderungszeile in die Nachricht geschrieben. **Anmerkung:** Die Abfragezeichenfolge ist ebenfalls Bestandteil des URL-Steuerparameters.	Nicht festgelegt.	Nicht festgelegt.	Wird ignoriert.
Version für dynamisches Überschreiben	Überschreibt die HTTP-Importversion, falls festgelegt. Wird in der Anforderungszeile in die Nachricht geschrieben.	Nicht festgelegt.	Nicht festgelegt.	Wird ignoriert.
Methode für dynamisches Überschreiben	Überschreibt die HTTP-Importmethode, falls festgelegt. Wird in der Anforderungszeile in die Nachricht geschrieben.	Nicht festgelegt.	Nicht festgelegt.	Wird ignoriert.

Tabelle 63. Bereitgestellte HTTP-Headerinformationen (Forts.)

Steuername	HTTP-Importanforderung	HTTP-Importantwort	HTTP-Exportanforderung	HTTP-Exportantwort
Medientyp (dieser Steuerparameter überträgt einen Teil des Wertes für den HTTP-Header 'Content-Type')	Wird als Teil des Headers 'Content-Type' in die Nachricht geschrieben, falls vorhanden. **Anmerkung:** Dieser Steuerelementwert sollte vom Datenhandler oder von der Datenbindung bereitgestellt werden.	Wird aus dem Header 'Content-Type' der Antwortnachricht gelesen.	Wird aus dem Header 'Content-Type' der Anforderungsnachricht gelesen.	Wird als Teil des Headers 'Content-Type' in die Nachricht geschrieben, falls vorhanden. **Anmerkung:** Dieser Steuerelementwert sollte vom Datenhandler oder von der Datenbindung bereitgestellt werden.
Zeichensatz (Standardwert: UTF-8)	Wird als Teil des Headers 'Content-Type' in die Nachricht geschrieben, falls vorhanden. **Anmerkung:** Dieser Steuerelementwert sollte von der Datenbindung bereitgestellt werden.	Wird aus dem Header 'Content-Type' der Antwortnachricht gelesen.	Wird aus dem Header 'Content-Type' der Anforderungsnachricht gelesen.	Unterstützt; wird als Teil des Headers 'Content-Type' in die Nachricht geschrieben. **Anmerkung:** Dieser Steuerelementwert sollte von der Datenbindung bereitgestellt werden.
Übertragungsverschlüsselung (Mögliche Werte: chunked, identity; Standardwert ist identity)	Wird, falls vorhanden, als Header in die Nachricht geschrieben und steuert, wie die Nachrichtentransformation verschlüsselt wird.	Wird aus der Antwortnachricht gelesen.	Wird aus der Anforderungsnachricht gelesen.	Wird, falls vorhanden, als Header in die Nachricht geschrieben und steuert, wie die Nachrichtentransformation verschlüsselt wird.
Inhaltsverschlüsselung (Mögliche Werte: gzip, x-gzip, deflate, identity; Standardwert ist identity)	Wird, falls vorhanden, als Header in die Nachricht geschrieben und steuert, wie die Nutzdaten verschlüsselt werden.	Wird aus der Antwortnachricht gelesen.	Wird aus der Anforderungsnachricht gelesen.	Wird, falls vorhanden, als Header in die Nachricht geschrieben und steuert, wie die Nutzdaten verschlüsselt werden.
Inhaltslänge (Content-Length)	Wird ignoriert.	Wird aus der Antwortnachricht gelesen.	Wird aus der Anforderungsnachricht gelesen.	Wird ignoriert.
Statuscode (Standardwert: 200)	Wird nicht unterstützt.	Wird aus der Antwortnachricht gelesen.	Wird nicht unterstützt.	Wird, falls vorhanden, in der Antwortzeile in die Nachricht geschrieben.
ReasonPhrase (Standardwert: OK)	Wird nicht unterstützt.	Wird aus der Antwortnachricht gelesen.	Wird nicht unterstützt.	Der Steuerwert wird ignoriert. Der Wert für die Antwortzeile der Nachricht wird aus dem Statuscode generiert.

Tabelle 63. Bereitgestellte HTTP-Headerinformationen (Forts.)

Steuername	HTTP-Importanforderung	HTTP-Importantwort	HTTP-Exportanforderung	HTTP-Exportantwort
Authentifizierung (enthält mehrere Eigenschaften)	Wird, falls vorhanden, zum Erstellen des Basisauthentifizierungsheaders verwendet. **Anmerkung:** Der Wert für diesen Header wird nur im HTTP-Protokoll verschlüsselt. Bei SCA wird er entschlüsselt und als Klartext übergeben.	Nicht anwendbar	Wird aus dem Basisauthentifizierungsheader der Anforderungsnachricht gelesen. Das Vorhandensein dieses Headers gibt keinen Aufschluss darüber, ob der Benutzer authentifiziert wurde. die Authentifizierung sollte in der Servletkonfiguration gesteuert werden. **Anmerkung:** Der Wert für diesen Header wird nur im HTTP-Protokoll verschlüsselt. Bei SCA wird er entschlüsselt und als Klartext übergeben.	Nicht anwendbar
Proxy (enthält mehrere Eigenschaften: Host, Port, Authentifizierung)	Wird, falls vorhanden, zum Herstellen der Verbindung über den Proxy verwendet.	Nicht anwendbar	Nicht anwendbar	Nicht anwendbar
SSL (enthält mehrere Eigenschaften: Schlüsselspeicher, Schlüsselspeicherkennwort, Truststore, Truststore-Kennwort, Clientauthentifizierung)	Wenn dieser Wert angegeben ist und HTTPS als Ziel-URL verwendet wird, wird hiermit eine Verbindung über SSL aufgebaut.	Nicht anwendbar	Nicht anwendbar	Nicht anwendbar

HTTP-Datenbindungen:

Für jede unterschiedliche Zuordnung von Daten zwischen einer SCA-Nachricht und einer HTTP-Protokollnachricht muss ein Datenhandler oder eine HTTP-Datenbindung konfiguriert werden. Datenhandler stellen eine bindungsneutrale Schnittstelle zur Verfügung, die eine transportbindungsübergreifende Wiederverwendung ermöglicht. Die Verwendung von Datenhandlern wird empfohlen, da Datenbindungen speziell für eine bestimmte Transportbindung gelten. HTTP-spezifische Datenbindungsklassen werden bereitgestellt. Sie können aber auch benutzerdefinierte Datenhandler oder Datenbindungen schreiben.

Anmerkung: Die drei im vorliegenden Thema beschriebenen HTTP-Datenbindungsklassen (HTTPStreamDataBindingSOAP, HTTPStreamDataBindingXML und HTTPServiceGatewayDataBinding) werden ab IBM Business Process Manager Version 7.0 nicht mehr unterstützt. Anstelle der hier beschriebenen Datenbindungen sollten die folgenden Datenhandler verwendet werden:

- SOAPDataHandler anstelle von HTTPStreamDataBindingSOAP.
- UTF8XMLDataHandler anstelle von HTTPStreamDataBindingXML
- GatewayTextDataHandler anstelle von HTTPServiceGatewayDataBinding

Datenbindungen werden für die Verwendung bei HTTP-Importen und HTTP-Exporten bereitgestellt: Binärdatenbindung, XML-Datenbindung und SOAP-Datenbindung. Eine Antwortdatenbindung ist bei uni-

direktionalen Operationen nicht erforderlich. Eine Datenbindung wird durch den Namen einer Java-Klasse repräsentiert, deren Instanzen ein HTTP-Objekt in ein Servicedatenobjekt konvertieren können (und umgekehrt). Bei einem Export muss ein Funktionsselektor verwendet werden, der (zusammen mit Methodenbindungen) ermitteln kann, welche Datenbindung verwendet und welche Operation aufgerufen wird. Die folgenden Datenbindungen werden bereitgestellt:

- Binärdatenbindungen, die den Hauptteil wie unstrukturierte binäre Daten behandeln. Das XSD-Schema der Binärdatenbindung hat die folgende Implementierung:

```
<xsd:schema elementFormDefault="qualified"
  targetNamespace="http://com.ibm.websphere.http.data.bindings/schema"
  xmlns:tns="http://com.ibm.websphere.http.data.bindings/schema"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">

  <xsd:complexType name="HTTPBaseBody">
    <xsd:sequence/>
  </xsd:complexType>

  <xsd:complexType name="HTTPBytesBody">
    <xsd:complexContent>
      <xsd:extension base="tns:HTTPBaseBody">
        <xsd:sequence>
          <xsd:element name="value" type="xsd:hexBinary"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
```

- XML-Datenbindungen, die den Hauptteil wie XML-Daten behandeln. Die Implementierung der XML-Datenbindung hat Ähnlichkeit mit der JMS-XML-Datenbindung und weist keine Beschränkungen hinsichtlich des Schnittstellenschemas auf.
- SOAP-Datenbindungen, die den Hauptteil als SOAP-Daten unterstützen. Die Implementierung der SOAP-Datenbindung weist keine Beschränkungen hinsichtlich des Schnittstellenschemas auf.

Benutzerdefinierte HTTP-Datenbindungen implementieren

In diesem Abschnitt ist beschrieben, wie eine benutzerdefinierte HTTP-Datenbindung implementiert werden kann.

Anmerkung: Die empfohlene Strategie besteht in der Implementierung eines benutzerdefinierten Datenhandlers, da dieser transportbindungsübergreifend wiederverwendet werden kann.

HTTPStreamDataBinding ist die Hauptschnittstelle für die Behandlung von benutzerdefinierten HTTP-Nachrichten. Die Schnittstelle ist so konzipiert, dass sie die Verarbeitung umfangreicher Nutzdaten zulässt. Damit eine solche Implementierung funktioniert, muss diese Datenbindung die Steuerinformationen und Header zurückgeben, bevor die Nachricht in den Datenstrom geschrieben wird.

Die Methoden und ihre Ausführungsreihenfolge (nachfolgend aufgeführt) müssen durch die benutzerdefinierte Datenbindung implementiert werden.

Zum Anpassen einer Datenbindung schreiben Sie eine Klasse, die 'HTTPStreamDataBinding' implementiert. Die Datenbindung sollte vier private Eigenschaften besitzen:

- private DataObject pDataObject
- private HTTPControl pCtrl
- private HTTPHeaders pHeaders
- private yourNativeDataType nativeData

Die HTTP-Bindung verwendet beim Aufruf der benutzerdefinierten Datenbindung die folgende Reihenfolge:

- Ausgangsverarbeitung (Datenobjekt in natives Format):

1. setDataObject(...)
2. setHeaders(...)
3. setControlParameters(...)
4. setBusinessException(...)
5. convertToNativeData()
6. getControlParameters()
7. getHeaders()
8. write(...)

- Eingangsverarbeitung (natives Format in Datenobjekt):
  1. setControlParameters(...)
  2. setHeaders(...)
  3. convertFromNativeData(...)
  4. isBusinessException()
  5. getDataObject()
  6. getControlParameters()
  7. getHeaders()

Sie müssen die Methode 'setDataObject(...)' in der Methode 'convertFromNativeData(...)' aufrufen, um den Wert des Datenobjekts (dataObject) festzulegen, das aus nativen Daten in die private Eigenschaft 'pData-Object' konvertiert wird.

```
public void setDataObject(DataObject dataObject)
   throws DataBindingException {
  pDataObject = dataObject;

}
public void setControlParameters(HTTPControl arg0) {
  this.pCtrl = arg0;
}

public void setHeaders(HTTPHeaders arg0) {
  this.pHeaders = arg0;
}
/*
* Add http header "IsBusinessException" in pHeaders.
* Two steps:
* 1.Remove all the header with name IsBusinessException (case-insensitive) first.
*   This is to make sure only one header is present.
* 2.Add the new header "IsBusinessException"
*/
public void setBusinessException(boolean isBusinessException) {
  //remove all the header with name IsBusinessException (case-insensitive) first.
  //This is to make sure only one header is present.
  //add the new header "IsBusinessException", code example:
  HTTPHeader header=HeadersFactory.eINSTANCE.createHTTPHeader();
  header.setName("IsBusinessException");
  header.setValue(Boolean.toString(isBusinessException));
  this.pHeaders.getHeader().add(header);

}
public HTTPControl getControlParameters() {
  return pCtrl;
}
public HTTPHeaders getHeaders() {
  return pHeaders;
}
public DataObject getDataObject() throws DataBindingException {
  return pDataObject;
}
```

```
/*
* Get header "IsBusinessException" from pHeaders, return its boolean value
*/
public boolean isBusinessException() {
  String headerValue = getHeaderValue(pHeaders,"IsBusinessException");
  boolean result=Boolean.parseBoolean(headerValue);
  return result;
}
public void convertToNativeData() throws DataBindingException {
  DataObject dataObject = getDataObject();
  this.nativeData=realConvertWorkFromSDOToNativeData(dataObject);
}
public void convertFromNativeData(HTTPInputStream arg0){
  //Customer-developed method to
  //Read data from HTTPInputStream
  //Convert it to DataObject
  DataObject dataobject=realConvertWorkFromNativeDataToSDO(arg0);
  setDataObject(dataobject);
}
public void write(HTTPOutputStream output) throws IOException {
  if (nativeData != null)
  output.write(nativeData);
}
```

EJB-Bindungen

EJB-Importbindungen (EJB = Enterprise JavaBeans) ermöglichen SCA-Komponenten den Aufruf von Services, die durch Java EE-Geschäftslogik auf einem Java EE-Server bereitgestellt werden. Mittels EJB-Exportbindungen können SCA-Komponenten als Enterprise JavaBeans zugänglich gemacht werden, damit Java EE-Geschäftslogik SCA-Komponenten aufrufen kann, die andernfalls für sie nicht verfügbar wären.

EJB-Importbindungen:

Mit EJB-Importbindungen kann ein SCA-Modul EJB-Implementierungen aufrufen, indem das Verfahren angegeben wird, mit dem das verarbeitende Modul an die externe EJB gebunden ist. Durch das Importieren von Services aus einer externen EJB-Implementierung können Benutzer ihre Geschäftslogik in die IBM Business Process Manager-Umgebung integrieren und an einem Geschäftsprozess mitwirken.

Zum Erstellen von EJB-Importbindungen verwenden Sie Integration Designer. Bei der Generierung der Bindungen können Sie eines der folgenden Verfahren verwenden:

- EJB-Import mit dem Assistenten 'Externer Service' erstellen

  Mit dem Assistenten 'Externer Service' können Sie in Integration Designer auf der Basis einer vorhandenen Implementierung einen EJB-Import erstellen. Der Assistent 'Externer Service' erstellt Services anhand der von Ihnen angegebenen Kriterien. Anschließend generiert er auf der Grundlage der erkannten Services Geschäftsobjekte, Schnittstellen und Importdateien.
- EJB-Import mit dem Assembly-Editor erstellen

  Zum Erstellen eines EJB-Imports können Sie den Assembly-Editor von Integration Designer verwenden. Sie können in der Palette entweder einen Import oder eine Java-Klasse verwenden, um die EJB-Bindung zu erstellen.

Der generierte Import besitzt Datenbindungen, die die Java-WSDL-Verbindung herstellen und keine Java-Brückenkomponente erforderlich machen. Sie können eine Komponente mit einer WSDL-Referenz direkt mit dem EJB-Import verbinden, der über eine Java-Schnittstelle mit einem EJB-basierten Service kommuniziert.

Der EJB-Import kann mit Java EE-Geschäftslogik entweder über das EJB 2.1- oder das EJB 3.0-Programmiermodell interagieren.

Der Aufruf der Java EE-Geschäftslogik kann lokal (nur bei EJB 3.0) oder über Fernzugriff erfolgen.

- Der lokale Aufruf wird verwendet, wenn Java EE-Geschäftslogik aufgerufen werden soll, die sich auf demselben Server wie der Import befindet.

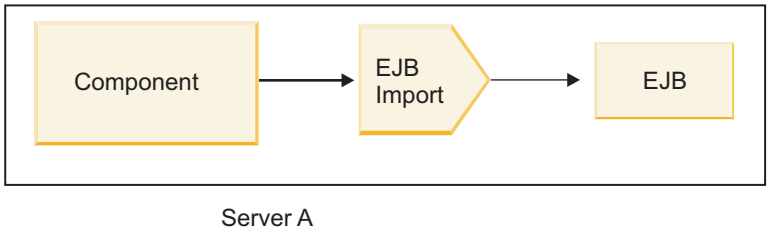


Abbildung 73. Lokaler Aufruf einer EJB (nur bei EJB 3.0)

- Der Fernaufruf wird verwendet, wenn Java EE-Geschäftslogik aufgerufen werden soll, die sich nicht auf demselben Server wie der Import befindet.

  In der folgenden Abbildung ist beispielsweise ein EJB-Import dargestellt, der eine EJB-Methode auf einem anderen Server über RMI/IIOP (Remote Method Invocation over Internet InterORB Protocol) aufruft.

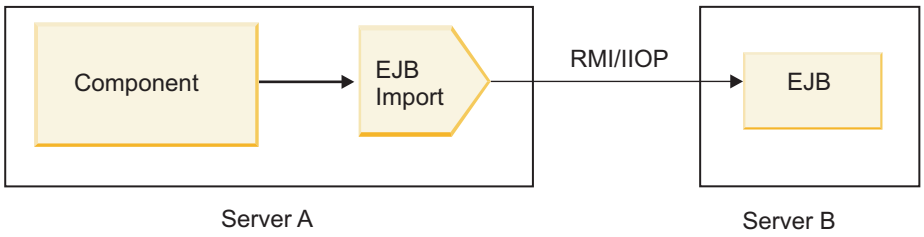


Abbildung 74. Aufruf einer EJB über Fernzugriff

Beim Konfigurieren der EJB-Bindung verwendet Integration Designer den JNDI-Namen, um die Version des EJB-Programmiermodells und den Typ des Aufrufs (lokal oder fern) zu ermitteln.

EJB-Importbindungen können die folgenden Hauptkomponenten enthalten:
- JAX-WS-Datenhandler
- EJB-Fehlerselektor
- EJB-Importfunktionsselektor

Falls Ihr Benutzerszenario nicht auf der JAX-WS-Zuordnung basiert, benötigen Sie möglicherweise einen angepassten Datenhandler, Funktionsselektor und Fehlerselektor, um die Tasks auszuführen, die andernfalls durch die Komponenten ausgeführt werden, die Bestandteil der EJB-Importbindungen sind. Hierzu gehört auch die Zuordnung, die normalerweise durch den benutzerdefinierten Zuordnungsalgorithmus vorgenommen wird.

EJB-Exportbindungen:

Externe Java EE-Anwendungen können über eine EJB-Exportbindung eine SCA-Komponente aufrufen. Durch einen EJB-Export können Sie SCA-Komponenten zugänglich machen, damit externe Java EE-Anwendungen diese Komponenten unter Verwendung des EJB-Programmiermodells aufrufen können.

Anmerkung: Der EJB-Export ist eine Stateless Bean (Bean ohne Statusaufzeichnung).

Zum Erstellen von EJB-Bindungen verwenden Sie Integration Designer. Bei der Generierung der Bindungen können Sie eines der folgenden Verfahren verwenden:
- EJB-Exportbindungen mit dem Assistenten 'Externer Service' erstellen

Mit dem Assistenten 'Externer Service' können Sie in Integration Designer auf der Basis einer vorhandenen Implementierung einen EJB-Exportservice erstellen. Der Assistent 'Externer Service' erstellt Services anhand der von Ihnen angegebenen Kriterien. Anschließend generiert er auf der Grundlage der erkannten Services Geschäftsobjekte, Schnittstellen und Exportdateien.

- EJB-Exportbindungen mit dem Assembly-Editor erstellen

  Zum Erstellen einer EJB-Exportbindung können Sie den Assembly-Editor von Integration Designer verwenden.

Wichtig: Ein J2SE-Client (J2SE = Java 2 Platform, Standard Edition) kann den in Integration Designer generierten EJB-Exportclient nicht aufrufen.

Sie können die Bindung ausgehend von einer vorhandenen SCA-Komponente generieren oder Sie können einen Export mit einer EJB-Bindung für eine Java-Schnittstelle generieren.

- Wenn Sie einen Export für eine vorhandene SCA-Komponente generieren, die mit einer vorhandenen WSDL-Schnittstelle ausgestattet ist, wird dem Export eine Java-Schnittstelle zugeordnet.
- Wenn Sie einen Export für eine Java-Schnittstelle erstellen, können Sie entweder eine WSDL- oder eine Java-Schnittstelle für den Export auswählen.

  Anmerkung: Eine Java-Schnittstelle, die zum Erstellen eines EJB-Exports verwendet wurde, unterliegt hinsichtlich der Objekte (Eingabe- und Ausgabeparameter sowie Ausnahmebedingungen), die als Parameter an einen fernen Aufruf übergeben werden, den folgenden Einschränkungen:
  - Die Objekte müssen einen konkreten Typ aufweisen (anstelle eines Schnittstellen- oder abstrakten Typs).
  - Die Objekte müssen mit der Enterprise JavaBeans-Spezifikation konform sein. Sie müssen serialisierbar sein und den Standardkonstruktor ohne Argument (no-argument) besitzen. Darüber hinaus muss auf alle Eigenschaften mit Getter- und Setter-Methoden zugegriffen werden können.

    Informationen zur Enterprise JavaBeans-Spezifikation finden Sie auf der Website von Sun Microsystems, Inc. unter der Adresse http://java.sun.com.

  Zusätzlich muss es sich bei der Ausnahmebedingung um eine geprüfte Ausnahmebedingung handeln, die von 'java.lang.Exception' übernommen wurde. Die Ausnahmebedingung muss zudem singulär sein (darf also nicht die Auslösung mehrerer Typen von geprüften Ausnahmebedingungen unterstützen).

  Bitte beachten Sie ebenfalls, dass eine Geschäftsschnittstelle einer Java EnterpriseBean eine normale Java-Schnittstelle ist und 'javax.ejb.EJBObject' oder 'javax.ejb.EJBLocalObject' nicht erweitern darf. Die Methoden der Geschäftsschnittstelle sollten keine Ausnahmebedingungen des Typs 'java.rmi.Remote-.Exception' auslösen.

Die EJB-Exportbindungen können mit Java EE-Geschäftslogik entweder über das EJB 2.1- oder das EJB 3.0-Programmiermodell interagieren.

Der Aufruf kann lokal (nur bei EJB 3.0) oder über Fernzugriff erfolgen.

- Der lokale Aufruf wird verwendet, wenn die Java EE-Geschäftslogik eine SCA-Komponente aufruft, die sich auf demselben Server wie der Export befindet.
- Der Fernaufruf wird verwendet, wenn sich die Java EE-Geschäftslogik nicht auf demselben Server befindet wie der Export.

  In der folgenden Abbildung ist beispielsweise eine EJB dargestellt, die RMI/IIOP verwendet, um eine SCA-Komponente auf einem anderen Server aufzurufen.

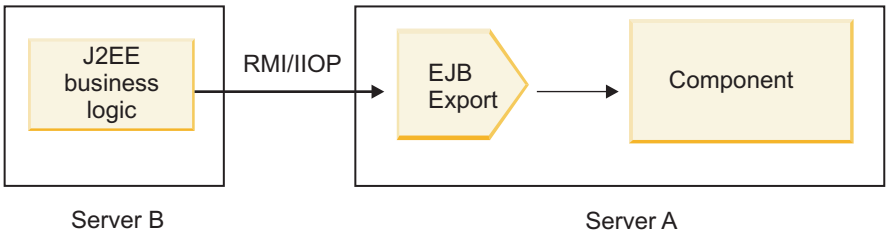


Abbildung 75. Fernaufruf von einem Client an eine SCA-Komponente mittels eines EJB-Exports

Beim Konfigurieren der EJB-Bindung verwendet Integration Designer den JNDI-Namen, um die Version des EJB-Programmiermodells und den Typ des Aufrufs (lokal oder fern) zu ermitteln.

EJB-Exportbindungen können die folgenden Hauptkomponenten enthalten:

- JAX-WS-Datenhandler
- EJB-Exportfunktionsselektor

Falls Ihr Benutzerszenario nicht auf der JAX-WS-Zuordnung basiert, benötigen Sie möglicherweise einen angepassten Datenhandler und Funktionsselektor, um die Tasks auszuführen, die andernfalls durch die Komponenten ausgeführt werden, die Bestandteil der EJB-Exportbindungen sind. Hierzu gehört auch die Zuordnung, die normalerweise durch den benutzerdefinierten Zuordnungsalgorithmus vorgenommen wird.

Eigenschaften von EJB-Bindungen:

EJB-Importbindungen verwenden ihre konfigurierten JNDI-Namen, um die Version des EJB-Programmiermodells und den Typ des Aufrufs (lokal oder fern) zu bestimmen. EJB-Importbindungen und -Exportbindungen verwenden zur Datentransformation den JAX-WS-Datenhandler. Die EJB-Importbindung verwendet einen EJB-Importfunktionsselektor und einen EJB-Fehlerselektor. Die EJB-Exportbindung verwendet einen Exportfunktionsselektor.

JNDI-Namen und EJB-Importbindungen:

Beim Konfigurieren der EJB-Bindung für einen Import verwendet Integration Designer den JNDI-Namen, um die Version des EJB-Programmiermodells und den Typ des Aufrufs (lokal oder fern) zu ermitteln.

Falls kein JNDI-Name angegeben ist, wird die Bindung für die EJB-Standardschnittstelle verwendet. Die erstellten Standardnamen sind davon abhängig, ob JavaBeans gemäß EJB 2.1 oder JavaBeans gemäß EJB 3.0 aufgerufen wird.

Anmerkung: Ausführliche Informationen zu den Namenskonventionen finden Sie in der Übersicht über EJB 3.0-Anwendungsbindungen im Information Center von WebSphere Application Server.

- JavaBeans gemäß EJB 2.1

  Der durch Integration Designer vorausgewählte standardmäßige JNDI-Name ist die EJB 2.1-Standardbindung. Diese hat das Format **ejb/** zuzüglich der Home-Schnittstelle (durch Schrägstriche getrennt).

  Für die Home-Schnittstelle von EJB 2.1 JavaBeans für 'com.mycompany.myremotebusinesshome' lautet die Standardbindung beispielsweise:

  ```
  ejb/com/mycompany/myremotebusinesshome
  ```

  Bei EJB 2.1 wird nur der ferne EJB-Aufruf unterstützt.
- JavaBeans gemäß EJB 3.0

  Der von Integration Designer für den lokalen JNDI-Namen vorausgewählte standardmäßige JNDI-Name ist der vollständig qualifizierte Klassenname der lokalen Schnittstelle, dem die Angabe **ejblocal:** vorangestellt ist. Für die vollständig qualifizierte Schnittstelle der lokalen Schnittstelle 'com.mycompany-.mylocalbusiness' wird beispielsweise der folgende EJB 3.0-JNDI-Name vorausgewählt:

`ejblocal:com.mycompany.mylocalbusiness`

Für die ferne Schnittstelle 'com.mycompany.myremotebusiness' ist der vorausgewählte EJB 3.0-JNDI-Name die vollständig qualifizierte Schnittstelle:

`com.mycompany.myremotebusiness`

Die EJB 3.0-Standard-Anwendungsbindungen werden unter der folgenden Adresse beschrieben: EJB 3.0-Anwendungsbindungen - Übersicht.

Integration Designer verwendet den Kurznamen als JNDI-Standardposition für EJBs, die das Programmiermodell von Version 3.0 verwenden.

Anmerkung: Falls die implementierte JNDI-Referenz der Ziel-EJB von der EJB-Standardbindungsposition abweicht, weil eine benutzerdefinierte Zuordnung verwendet oder konfiguriert wurde, muss der JNDI-Zielname richtig angegeben werden. Sie können den Namen in Integration Designer vor der Implementierung angeben. Bei der Importbindung haben Sie auch die Möglichkeit, den Namen (nach der Implementierung) in der Administrationskonsole so zu ändern, dass er mit dem JNDI-Namen der Ziel-EJB übereinstimmt.

Weitere Informationen zum Erstellen von EJB-Bindungen enthalten die Abschnitte über die Arbeit mit EJB-Bindungen im Information Center von Integration Designer.

JAX-WS-Datenhandler:

Die EJB-Importbindung (EJB = Enterprise JavaBeans) verwendet den JAX-WS-Datenhandler, um Anforderungsgeschäftsobjekte in Java-Objektparameter und den Rückgabewert des Java-Objekts in ein Antwortgeschäftsobjekt umzuwandeln. Die EJB-Exportbindung verwendet den JAX-WS-Datenhandler, um Anforderungs-EJBs in Anforderungsgeschäftsobjekte und die Antwortgeschäftsobjekte in einen Rückgabewert umzuwandeln.

Dieser Datenhandler ordnet Daten aus einer mit SCA angegebenen WSDL-Schnittstelle einer EJB-Java-Zielschnittstelle zu (und umgekehrt), unter Verwendung der Spezifikation von Java API for XML Web Services (JAX-WS) und der Spezifikation von Java Architecture for XML Binding (JAXB).

Anmerkung: Die Unterstützung ist gegenwärtig auf die Spezifikationen von JAX-WS 2.1.1 und JAXB 2.1.3 beschränkt.

Mit dem Datenhandler, der auf der Ebene der EJB-Bindung angegeben ist, wird die Verarbeitung von Anforderungen, Antworten, Fehlern und Laufzeitausnahmebedingungen ausgeführt.

Anmerkung: Für Fehler kann durch Angabe der Konfigurationseigenschaft 'faultBindingType' ein spezieller Datenhandler für jeden Fehler angegeben werden. Dies überschreibt den Wert, der auf der Ebene der EJB-Bindung angegeben ist.

Der JAX-WS-Datenhandler wird standardmäßig verwendet, wenn die EJB-Bindung eine WSDL-Schnittstelle besitzt. Dieser Datenhandler kann nicht verwendet werden, um eine SOAP-Nachricht zu transformieren, die einen JAX-WS-Aufruf eines Datenobjektes darstellt.

Die EJB-Importbindung verwendet einen Datenhandler, um ein Datenobjekt in einen Java-Objekt-Array (Object[]) zu transformieren. Während der abgehenden Kommunikation findet die folgende Verarbeitung statt:

1. Die EJB-Bindung legt den erwarteten Typ, das erwartete Element und den angestrebten Methodennamen im Element 'BindingContext' übereinstimmend mit den in WSDL angegebenen Elementen fest.
2. Die EJB-Bindung ruft die Transformationsmethode für das Datenobjekt auf, bei dem eine Datentransformation erfolgen muss.
3. Der Datenhandler gibt einen Array 'Object[]' zurück, der die Parameter der Methode (in der Reihenfolge ihrer Definition in der Methode) darstellt.

4. Die EJB-Bindung verwendet den Array 'Object[]', um die Methode für die EJB-Zielschnittstelle aufzurufen.

Die Bindung bereitet außerdem einen Array 'Object[]' zur Verarbeitung der Antwort für den EJB-Aufruf vor.

- Das erste Element im Array 'Object[]' ist der Rückgabewert vom Java-Methodenaufruf.
- Die nachfolgenden Werte stellen die Eingabeparameter für die Methode dar.

Dies ist erforderlich, damit Parameter des Typs 'Ein-/Ausgabe' sowie 'Ausgabe' unterstützt werden.

Bei Parametern des Typs 'Ausgabe' müssen die Werte im Antwortdatenobjekt zurückgegeben werden.

Der Datenhandler verarbeitet und transformiert Werte, die im Array 'Object[]' gefunden wurden, und gibt dann eine Antwort an das Datenobjekt zurück.

Der Datenhandler unterstützt neben anderen XSD-Datentypen auch 'xs:AnyType', 'xs:AnySimpleType' und 'xs:Any'. Um die Unterstützung für 'xs:Any' zu aktivieren, verwenden Sie die Angabe **@XmlAnyElement (lax=true)** für die JavaBeans-Eigenschaft im Java-Code wie im folgenden Beispiel gezeigt:

```
public class TestType {
 private Object[] object;

 @XmlAnyElement (lax=true)
 public Object[] getObject() {
  return object;
 )

 public void setObject (Object[] object) {
  this.object=object;
 )
)
```

Hierdurch wird das Eingeschaftsobjekt in 'TestType' zu einem Feld des Typs 'xs:any'. Der im Feld des Typs 'xs:any' verwendete Java-Klassenwert sollte mit der Annotation **@XmlAnyElement** versehen sein. Beispiel: Falls der Objektarray unter Verwendung der Java-Klasse 'Address' gefüllt wird, sollte die Klasse 'Address' mit der Annotation **@XmlRootElement** versehen sein.

Anmerkung: Um die Zuordnung des XSD-Typs zu den Java-Typen anzupassen, die in der JAX-WS-Spezifikation definiert sind, ändern Sie die JAXB-Annotationen so, dass Ihre Geschäftsanforderungen erfüllt werden. Der JAX-WS-Datenhandler unterstützt 'xs:any', 'xs:anyType' und 'xs:anySimpleType'.

Für den JAX-WS-Datenhandler gelten die folgenden Einschränkungen:

- Der Datenhandler bietet keine Unterstützung für die Headerattributannotation **@WebParam**.
- Der Namespace für Geschäftsobjektschemadateien (XSD-Dateien) enthält keine Standardzuordnung für den Java-Paketnamen. Die Annotation **@XMLSchema** in der Datei 'package-info.java' kann ebenfalls nicht verwendet werden. Die einzige Möglichkeit zur Erstellung von XSD mit einem Namespace besteht in der Verwendung der Annotationen **@XmlType** und **@XmlRootElement**. Die Annotation **@XmlRootElement** definiert den Zielnamespace für das globale Element in JavaBeans-Typen.
- Der EJB-Importassistent erstellt keine XSD-Dateien für zusammenhangslose Klassen. Version 2.0 unterstützt die Annotation **@XmlSeeAlso** nicht, weshalb kein XSD erstellt wird, falls die untergeordnete Klasse nicht direkt aus der übergeordneten Klasse referenziert wird. Dieses Problem kann durch die Ausführung von 'SchemaGen' für solche untergeordneten Klassen gelöst werden.

  Bei 'SchemaGen' handelt es sich um ein Befehlszeilendienstprogramm (Position ist das Verzeichnis '*WPS-installationsausgangsverzeichnis*/bin'), das zum Erstellen von XSD-Dateien für eine bestimmte Bean bereitgestellt wird. Diese XSD-Dateien müssen manuell in das Modul kopiert werden, damit das Modul funktionsfähig ist.

EJB-Fehlerselektor:

Der EJB-Fehlerselektor ermittelt, ob ein EJB-Aufruf zu einem Fehler, einer Laufzeitausnahmebedingung oder einer erfolgreichen Antwort geführt hat.

Falls ein Fehler festgestellt wird, gibt der EJB-Fehlerselektor den nativen Fehlernamen an die Bindungslaufzeit zurück, damit der JAX-WS-Datenhandler das Ausnahmebedingungsobjekt in ein Fehlergeschäftsobjekt konvertieren kann.

Bei einer erfolgreichen (fehlerfreien) Antwort assembliert die EJB-Importbindung einen Java-Objektarray (Object[]), um die Werte zurückzugeben.

- Das erste Element im Array 'Object[]' ist der Rückgabewert vom Java-Methodenaufruf.
- Die nachfolgenden Werte stellen die Eingabeparameter für die Methode dar.

Dies ist erforderlich, damit Parameter des Typs 'Ein-/Ausgabe' sowie 'Ausgabe' unterstützt werden.

In Szenarios mit Ausnahmebedingungen assembliert die Bindung einen Array 'Object[]' und das erste Element stellt die durch die Methode ausgelöste Ausnahmebedingung dar.

Der Fehlerselektor kann einen der folgenden Werte zurückgeben:

Tabelle 64. Rückgabewerte

Typ	Rückgabewert	Beschreibung
Fehler	**ResponseType.FAULT**	Wird zurückgegeben, wenn der übergebene Array 'Object[]' ein Ausnahmebedingungsobjekt enthält.
Laufzeitausnahmebedingung	**ResponseType.RUNTIME**	Wird zurückgegeben, falls das Ausnahmebedingungsobjekt mit keinem der für die Methode deklarierten Ausnahmebedingungstypen übereinstimmt.
Normale Antwort	**ResponseType.RESPONSE**	Wird in allen anderen Fällen zurückgegeben.

Falls der Fehlerselektor den Wert **ResponseType.FAULT** zurückgibt, wird der native Fehlername zurückgegeben. Anhand dieses nativen Fehlernamens ermittelt die Bindung den korrespondierenden WSDL-Fehlernamen und ruft den entsprechenden Fehlerdatenhandler auf.

EJB-Funktionsselektor:

Die EJB-Bindungen können einen Importfunktionsselektor (für die Ausgangsverarbeitung) oder einen Exportfunktionsselektor (für die Eingangsverarbeitung) verwenden, um die aufzurufende EJB-Methode zu ermitteln.

Importfunktionsselektor

Bei der Ausgangsverarbeitung leitet der Importfunktionsselektor den Typ der EJB-Methode basierend auf dem Namen der Operation ab, die durch die mit dem EJB-Import verbundene SCA-Komponente aufgerufen wird. Der Funktionsselektor sucht nach der Annotation '@WebMethod' für die in Integration Designer generierte JAX-WS-annotierte Java-Klasse, um den zugeordneten Zieloperationsnamen zu ermitteln.

- Falls die Annotation '@WebMethod' vorhanden ist, stellt der Funktionsselektor mithilfe der Annotation '@WebMethod' die korrekte Java-Methodenzuordnung für die WSDL-Methode fest.
- Wenn die Annotation @WebMethod nicht vorhanden ist, geht der Funktionsselektor davon aus, dass der Java-Methodenname mit dem Namen der aufgerufenen Operation übereinstimmt..

Anmerkung: Dieser Funktionsselektor ist nur bei einer WSDL-Schnittstelle für einen EJB-Import und nicht bei einer Java-Schnittstelle für einen EJB-Import gültig.

Der Funktionsselektor gibt ein Objekt 'java.lang.reflect.Method' zurück, das die Methode der EJB-Schnittstelle darstellt.

Der Funktionsselektor verwendet einen Java-Objektarray (Object[]) zur Aufnahme der Antwort von der Zielmethode. Das erste Element im Array 'Object[]' ist eine Java-Methode mit dem Namen aus WSDL, das zweite Element im Array 'Object[]' ist das Eingabegeschäftsobjekt.

Exportfunktionsselektor

Bei der Eingangsverarbeitung leitet der Exportfunktionsselektor die aufzurufende Zielmethode aus der Java-Methode ab.

Der Exportfunktionsselektor ordnet den Namen der vom EJB-Client aufgerufenen Java-Operation dem Namen der Operation in der Schnittstelle der Zielkomponente zu. Der Methodenname wird als Zeichenfolge zurückgegeben und durch die SCA-Laufzeit abhängig vom Schnittstellentyp der Zielkomponente aufgelöst.

EIS-Bindungen

EIS-Bindungen stellen Konnektivität zwischen SCA-Komponenten und einem externen EIS (Enterprise Information System, unternehmensweites Informationssystem) bereit. Diese Kommunikation wird erreicht durch die Verwendung von EIS-Exporten und EIS-Importen, die JCA 1.5-Ressourcenadapter und WebSphere Adapters unterstützen.

Möglicherweise macht es eine SCA-Komponente erforderlich, dass Daten an ein externes EIS bzw. von diesem übertragen werden. Wenn Sie ein SCA-Modul erstellen, das eine solche Konnektivität benötigt, schließen Sie (neben Ihrer SCA-Komponente) einen Import oder Export mit einer EIS-Bindung für die Kommunikation mit einem bestimmten externen EIS ein.

Ressourcenadapter in IBM Integration Designer werden im Kontext eines Imports oder Exports verwendet. Sie entwickeln einen Import oder Export mit dem Assistenten 'Externer Service' und schließen bei seiner Entwicklung den Ressourcenadapter ein. Ein EIS-Import, mit dem eine Anwendung einen Service auf einem EIS-System aufrufen kann, oder ein EIS-Export, mit dem eine Anwendung auf einem EIS-System einen in IBM Integration Designer entwickelten Service aufrufen kann, wird mit einem Ressourcenadapter erstellt. Beispielsweise würden Sie einen Import mit dem JD Edwards-Adapter erstellen, um einen Service auf einem JD Edwards-System aufzurufen.

Wenn Sie den Assistenten 'Externer Service' verwenden, werden die Informationen zur EIS-Bindung automatisch erstellt. Bindungsinformationen können Sie auch mit einem anderen Tool hinzufügen oder ändern, nämlich dem Assembly-Editor. Weitere Informationen hierzu finden Sie unter Mit Adaptern auf externe Services zugreifen.

Nachdem das SCA-Modul, das die EIS-Bindung enthält, auf dem Server implementiert wurde, können Sie in der Administrationskonsole die Informationen zur Bindung anzeigen oder die Bindung konfigurieren.

EIS-Bindungen - Übersicht:

Wenn die EIS-Bindung (EIS = Enterprise Information System, unternehmensweites Informationssystem) zusammen mit einem JCA-Ressourcenadapter verwendet wird, können Sie auf Services in einem unternehmensweiten Informationssystem zugreifen oder Ihre Services für das EIS bereitstellen.

Die folgenden Beispiele veranschaulichen, wie ein SCA-Modul namens **ContactSyncModule** Kontaktinformationen zwischen einem Siebel-System und einem SAP-System synchronisiert.

1. Die SCA-Komponente namens **ContactSync** überwacht (durch einen EIS-Anwendungsexport namens 'Siebel Contact') Siebel-Kontakte auf Änderungen.
2. Die SCA-Komponente **ContactSync** selbst verwendet eine SAP-Anwendung (durch einen EIS-Anwendungsimport), um die SAP-Kontaktinformationen entsprechend zu aktualisieren.

Da die Datenstrukturen für die Speicherung von Kontakten bei Siebel- und SAP-Systemen unterschiedlich sind, muss die SCA-Komponente **ContactSync** eine Zuordnung bereitstellen.

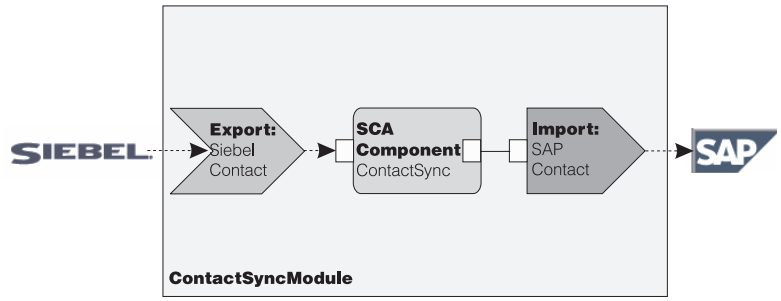


Abbildung 76. Fluss von einem Siebel-System an ein SAP-System

Für den Export 'Siebel Contact' und den Import 'SAP Contact' sind die geeigneten Ressourcenadapter konfiguriert.

Schlüsselfunktionen von EIS-Bindings:

Ein EIS-Import ist ein SCA-Import (SCA = Service Component Architecture), der es den Komponenten im SCA-Modul ermöglicht, EIS-Anwendungen zu verwenden, die außerhalb des SCA-Moduls definiert sind. Ein EIS-Import wird zum Übertragen der Daten von der SCA-Komponente an ein externes EIS verwendet; ein EIS-Export wird zum Übertragen der Daten von einem externen EIS in das SCA-Modul verwendet.

Importe

Ein EIS-Import dient dazu, die Lücke zwischen SCA-Komponenten und EIS-Systemen zu schließen. Externe Anwendungen können als ein EIS-Import behandelt werden. In diesem Fall werden von einem EIS-Import Daten an das externe EIS gesendet und optional Daten als Antwort empfangen.

Vom EIS-Import wird den SCA-Komponenten eine einheitliche Sicht der Anwendungen bereitgestellt, die sich außerhalb des Moduls befinden. So können die Komponenten mit einem externen EIS wie SAP, Siebel oder PeopleSoft über ein konsistentes SCA-Modell kommunizieren.

Auf der Clientseite des Imports befindet sich eine Schnittstelle, die von der EIS-Exportanwendung verfügbar gemacht wird; sie verfügt über mindestens eine Methode, von der Datenobjekte als Argumente angenommen und Werte zurückgegeben werden. Auf der Implementierungsseite befindet sich eine Common Client Interface (CCI), die von einem Ressourcenadapter implementiert wird.

Von der Laufzeitimplementierung eines EIS-Imports wird eine Verbindung zwischen der clientseitigen Schnittstelle und der CCI hergestellt. Vom Import wird der Aufruf der Methode in der Schnittstelle dem Aufruf in der CCI zugeordnet.

Bindings werden auf drei Ebenen erstellt: das Schnittstellenbinding, von dem die enthaltenen Methodenbindings verwendet werden, die wiederum die Datenbindings verwenden.

Vom Schnittstellenbinding wird die Schnittstelle des Imports der Verbindung dem EIS-System zugeordnet, von dem die Anwendung bereitgestellt wird. Dies spiegelt die Tatsache wider, dass die Anwendungen, die von der Schnittstelle dargestellt werden, von der konkreten Instanz des EIS bereitgestellt wird,

und dass von der Verbindung der Zugriff auf diese Instanz bereitgestellt wird. Das Bindingelement enthält Eigenschaften, deren Informationen dazu ausreichen, die Verbindung herzustellen (diese Eigenschaften sind Bestandteil der Instanz javax.resource.spi.ManagedConnectionFactory).

Vom Methodenbinding wird die Methode mit der konkreten Interaktion dem EIS-System zugeordnet. Bei Verwendung von JCA ergibt sich die Interaktion aus den Eigenschaften der Schnittstellenimplementierung javax.resource.cci.InteractionSpec. Das Interaktionselement des Methodenbindings enthält diese Eigenschaften zusammen mit dem Namen der Klasse und stellt so ausreichend Informationen zum Ausführen der Interaktion bereit. Vom Methodenbinding werden Datenbindings verwendet, die die Zuordnung des Arguments und des Ergebnisses der Schnittstellenmethode zur EIS-Darstellung beschreiben.

Laufzeitszenario für einen EIS-Import:

1. Die Methode für die Importschnittstelle wird mithilfe des SCA-Programmiermodells aufgerufen.
2. In der Anforderung, die vom EIS-Import empfangen wird, sind der Name der Methode und die zugehörigen Argumente enthalten.
3. Vom Import wird zuerst eine Schnittstellenbindingimplementierung erstellt; anschließend wird unter Verwendung der Daten aus dem Importbinding die API für die Verbindungsfactory (ConnectionFactory) erstellt und beide werden einander zugeordnet. Dies bedeutet, dass vom Import die API setConnectionFactory im Schnittstellenbinding aufgerufen wird.
4. Die Methodenbindingimplementierung für die entsprechende aufgerufene Methode wird erstellt.
5. Die Instanz javax.resource.cci.InteractionSpec wird erstellt und gefüllt; anschließend werden die Methodenargumente mithilfe der Datenbindings an ein Format gebunden, das vom Ressourcenadapter verstanden wird.
6. Die CCI-Schnittstelle wird zum Ausführen der Interaktion verwendet.
7. Wenn der Aufruf zurückgegeben wird, wird das Datenbinding zum Erstellen des Ergebnisses des Aufrufs verwendet und das Ergebnis wird an das aufrufende Program zurückgegeben.

Exporte

Ein EIS-Export dient dazu, die Lücke zwischen einer SCA-Komponente und einem externen EIS zu schließen. Externe Anwendungen können als ein EIS-Export behandelt werden. In diesem Fall werden die Daten der externen Anwendung von ihr in Form regelmäßiger Benachrichtigungen gesendet. Ein EIS-Export ist sozusagen eine Subskriptionsanwendung, die für eine externe Anforderung von einem EIS empfangsbereit ist. Von der SCA-Komponente, die den EIS-Export verwendet, wird dieser als lokale Anwendung betrachtet.

Vom EIS-Export wird den SCA-Komponenten eine einheitliche Sicht der Anwendungen bereitgestellt, die sich außerhalb des Moduls befinden. So können die Komponenten mit einem EIS wie SAP, Siebel oder PeopleSoft über ein konsistentes SCA-Modell kommunizieren.

Vom Export wird die Implementierung eines Listeners bereitgestellt, von dem die Anforderungen des EIS empfangen werden. Vom Listener wird eine für den Ressourcenadapter spezifische Schnittstelle implementiert. Der Export enthält auch eine Komponenten implementierende Schnittstelle, die dem EIS über den Export verfügbar gemacht wird.

Die Laufzeitimplementierung eines EIS-Imports verbindet den Listener mit der Komponenten implementierenden Schnittstelle. Die EIS-Anforderung wird vom Export dem Aufruf der entsprechenden Operation für die Komponente zugeordnet. Bindings werden auf drei Ebenen erstellt: ein Listener-Binding, von dem anschließend ein enthaltenes natives Methodenbinding verwendet wird, von dem wiederum ein Datenbinding verwendet wird.

Vom Listener-Binding wird der Listener, der die Anforderungen empfängt, der Komponente zugeordnet, die über den Export zugänglich gemacht wird. In der Exportdefinition ist der Name der Komponente enthalten; sie wird von der Laufzeit gesucht und die Anforderungen werden an sie weitergeleitet.

Vom nativen Methodenbinding werden die vom Listener empfangenen nativen Methoden oder Ereignistypen den Operationen zugeordnet, die von der Komponente über den Export zugänglich gemacht werden. Es gibt keine Beziehung zwischen der für den Listener aufgerufenen Methode und dem Ereignistyp; alle Ereignisse werden über mindestens eine Methode des Listeners empfangen. Das native Methodenbinding verwendet den im Export definierten Funktionsselektor, um den nativen Methodennamen aus den eingehenden Daten zu extrahieren und die Datenbindings, um das Datenformat des EIS an ein Format zu binden, das von der Komponente verstanden wird.

Laufzeitszenario für einen EIS-Export:

1. Von einer EIS-Anforderung wird der Aufruf einer Methode für die Listenerimplementierung ausgelöst.
2. Vom Listener wird der Export gesucht und aufgerufen, alle Aufrufargumente werden an ihn übergeben.
3. Vom Export wird die Implementierung des Listener-Bindings erstellt.
4. Vom Export wird der Funktionsselektor instanziiert und für das Listener-Binding eingestellt.
5. Vom Export werden die nativen Methodenbindings initialisiert und zum Listener-Binding hinzugefügt. Für jedes Methodenbinding werden auch die Datenbindings initialisiert.
6. Vom Export wird das Listener-Binding aufgerufen.
7. Vom Listener-Binding werden exportierte Komponenten gesucht und der native Methodenname mit dem Funktionsselektor abgerufen.
8. Anhand dieses Namens wird das native Methodenbinding gesucht, von dem schließlich die Zielkomponente aufgerufen wird.

Der Adapterinteraktionsstil ermöglicht es dem EIS-Exportbinding, die Zielkomponente entweder asynchron (die Standardeinstellung) oder synchron aufzurufen.

Ressourcenadapter

Sie entwickeln einen Import oder Export mit dem Assistenten für externe Services und schließen bei der Entwicklung einen Ressourcenadapter mit ein. Die Adapter, die im Lieferumfang von IBM Integration Designer für den Zugriff auf CICS-, IMS-, JD Edwards-, PeopleSoft-, SAP- und Siebel-Systeme enthalten sind, sind nur für Entwicklungs- und Testzwecke vorgesehen. Sie können sie also nur zum Entwickeln und Testen der Anwendungen verwenden.

Sobald Sie eine Anwendung implementieren, benötigen Sie lizenzierte Laufzeitadapter zum Ausführen der Anwendung. Wenn Sie einen Service erstellen, können Sie den Adapter aber in den Service einbetten. Es kann vorkommen, dass die Verwendung des eingebetteten Adapters als lizenzierter Laufzeitadapter laut Adapterlizenzierung zulässig ist. Diese Adapter sind mit der Java EE Connector Architecture (JCA 1.5) kompatibel. JCA ist ein offener Standard und der Java EE-Standard für EIS-Verbindungen. Von JCA wird ein verwaltetes Framework bereitgestellt; dies bedeutet, dass die Qualität des Service (QoS) vom Anwendungsserver bereitgestellt wird, der Lebenszyklusverwaltung und Sicherheit für Transaktionen bietet. Mit Ausnahme von IBM CICS ECI Resource Adapter und IBM IMS Connector for Java sind Sie auch mit der Enterprise Metadata Discovery-Spezifikation kompatibel.

Auch WebSphere Business Integration Adapters, eine ältere Gruppe aus Adaptern, wird vom Assistenten unterstützt.

Java EE-Ressourcen

Das EIS-Modul, ein SCA-Modul, das dem EIS-Modulmuster entspricht, kann auf der Java EE-Plattform implementiert werden.

Das Ergebnis der Implementierung des EIS-Moduls auf der Java EE-Plattform ist eine Anwendung, die startbereit vorliegt, als EAR-Datei paketiert ist und auf dem Server implementiert wird. Alle Java EE-Ar-

tefakte und Ressourcen werden erstellt; die Anwendung ist konfiguriert und kann ausgeführt werden.

Dynamische Eigenschaften der JCA-Interaktionsspezifikation und -Verbindungsspezifikation:

Die EIS-Bindung kann Eingabe für die APIs **InteractionSpec** und **ConnectionSpec** akzeptieren, die durch ein klar strukturiertes untergeordnetes Datenobjekt angegeben werden, das die Nutzdaten begleitet. Dies ermöglicht dynamische Anforderungs-/Antwortinteraktionen mit einem Ressourcenadapter über die API **InteractionSpec** und die Interaktion für die Komponentenauthentifizierung über die API **ConnectionSpec**.

Die API **javax.cci.InteractionSpec** überträgt Informationen dazu, wie die Anforderung für die Interaktion mit dem Ressourcenadapter abgewickelt werden soll. Sie kann außerdem Informationen dazu übertragen, wie die Interaktion nach der Anforderung erreicht wurde. Diese wechselseitige Übertragung durch die Interaktionen wird manchmal auch als *Dialog* bezeichnet.

Die EIS-Bindung erwartet, dass die Nutzdaten, die das Argument für den Ressourcenadapter darstellen, ein untergeordnetes Datenobjekt namens **properties** enthalten. Dieses Eigenschaftsdatenobjekt enthält Name/Wert-Paare, bei denen die Namen der Eigenschaften für die Interaktionsspezifikation in einem bestimmten Format angegeben sind. In diesem Zusammenhang gelten die folgenden Formatierungsregeln:

- Namen müssen mit dem Präfix **IS** beginnen, auf das der Eigenschaftsname folgt. Beispiel: Eine Interaktionsspezifikation mit einer JavaBeans-Eigenschaft namens **InteractionId** würde den Eigenschaftsnamen mit **ISInteractionId** angeben.
- Das Name/Wert-Paar stellt den Namen und den Wert des einfachen Typs für die Eigenschaft der Interaktionsspezifikation dar.

In diesem Beispiel gibt eine Schnittstelle an, dass die Eingabe für eine Operation aus einem Datenobjekt **Account** besteht. Diese Schnittstelle ruft eine Anwendung mit EIS-Importbindung mit dem Ziel auf, eine dynamische Eigenschaft **InteractionSpec** namens **workingSet** mit dem Wert **xyz** zu senden und zu empfangen.

Die Geschäftsgrafik oder die Geschäftsobjekte im Server enthalten ein zugrunde liegendes Geschäftsobjekt **properties**, das das Senden von protokollspezifischen Daten mit den Nutzdaten zulässt. Dieses Geschäftsobjekt **properties** ist integriert und muss beim Erstellen eines Geschäftsobjekts nicht im XML-Schema angegeben werden. Es muss lediglich erstellt und verwendet werden. Falls Sie basierend auf einem XML-Schema eigene Datentypen definiert haben, müssen Sie ein Element **properties** angeben, dass die erwarteten Name/Wert-Paare enthält.

```
BOFactory dataFactory = (BOFactory) \
serviceManager.locateService("com/ibm/websphere/bo/BOFactory");
 //Wrapper for doc-lit wrapped style interfaces,
 //skip to payload for non doc-lit
 DataObject docLitWrapper = dataFactory.createByElement /
 ("http://mytest/eis/Account", "AccountWrapper");
```

Erstellen Sie die Nutzdaten.

```
DataObject account = docLitWrapper.createDataObject(0);
  DataObject accountInfo = account.createDataObject("AccountInfo");
  //Perform your setting up of payload


  //Construct properties data for dynamic interaction

  DataObject properties = account.createDataObject("properties");
```

Legen Sie für den Namen von 'workingSet' den erwarteten Wert fest (**xyz**).

```
properties.setString("ISworkingSet", "xyz");


  //Invoke the service with argument
```

```
Service accountImport = (Service) \
serviceManager.locateService("AccountOutbound");
DataObject result = accountImport.invoke("createAccount", docLitWrapper);

//Get returned property
DataObject retProperties = result.getDataObject("properties");

String workingset = retProperties.getString("ISworkingSet");
```

Sie können die Eigenschaften der API **ConnectionSpec** für die dynamische Komponentenauthentifizierung verwenden. Hierbei gelten ebenfalls die vorgenannten Regeln, allerdings mit der Ausnahme, dass das Präfix **CS** (anstelle von **IS**) für den Eigenschaftsnamen verwendet werden muss. Die Eigenschaften der API **ConnectionSpec** sind nicht bidirektional. Ein einziges Datenobjekt **properties** kann sowohl IS- als auch CS-Eigenschaften enthalten.

Zur Verwendung der Eigenschaften für **ConnectionSpec** legen Sie **resAuth** in der Importbindung mit **Application** fest. Stellen Sie außerdem sicher, dass der Ressourcenadapter die Komponentenauthentifizierung unterstützt. Weitere Angaben finden Sie in Kapitel 8 des Dokuments J2EE Connector Architecture Specification.

Externe Clients mit EIS-Bindungen:

Der Server kann unter Verwendung von EIS-Bindungen Nachrichten an externe Clients senden bzw. von diesen empfangen.

Ein externer Client (beispielsweise ein Webportal oder ein EIS) muss eine Nachricht an ein SCA-Modul im Server senden oder muss von einer Komponente im Server aufgerufen werden.

Der Client ruft wie bei jeder anderen Anwendung den EIS-Import entweder mit der DII (Dynamic Invocation Interface - Schnittstelle für dynamischen Aufruf) oder der Java-Schnittstelle auf.

1. Der externe Client erstellt eine Instanz der API **ServiceManager** und sucht unter Verwendung des Referenznamens nach dem EIS-Import. Das Ergebnis der Suche ist eine Serviceschnittstellenimplementierung.
2. Der Client generiert ein Eingabeargument (ein generisches Datenobjekt), das dynamisch unter Verwendung des Datenobjektschemas erstellt wird. Dieser Schritt wird unter Verwendung der SDO-Schnittstellenimplementierung 'DataFactory' ausgeführt.
3. Der externe Client ruft das EIS auf und erhält die angeforderten Ergebnisse.

Alternativ kann der Client den EIS-Import unter Verwendung der Java-Schnittstelle aufrufen.

1. Der Client erstellt eine Instanz der API **ServiceManager** und sucht unter Verwendung des Referenznamens nach dem EIS-Import. Das Ergebnis der Suche ist eine Java-Schnittstelle des EIS-Imports.
2. Der Client erstellt ein Eingabeargument und ein typisiertes Datenobjekt.
3. Der Client ruft das EIS auf und erhält die angeforderten Ergebnisse.

Die EIS-Exportschnittstelle definiert die Schnittstelle der exportierten SCA-Komponente, die für die externen EIS-Anwendungen verfügbar ist. Diese Schnittstelle ist mit der Schnittstelle vergleichbar, die eine externe Anwendung (z. B. SAP oder PeopleSoft) durch die Implementierung der EIS-Exportanwendungslaufzeit aufruft.

Der Export verwendet die Schnittstelle **EISExportBinding**, um exportierte Services an die externe EIS-Anwendung zu binden. Er ermöglicht das Abonnieren einer im SCA-Modul enthaltenen Anwendung, um EIS-Serviceanforderungen zu überwachen. Die EIS-Exportbindung gibt in einer für den Ressourcenadapter verständlichen Weise (mit Schnittstellen von Java EE Connector Architecture) die Zuordnung zwischen eingehenden Ereignissen und dem Aufruf von SCA-Operationen an.

Die Schnittstelle **EISExportBinding** macht es erforderlich, dass externe EIS-Services auf den Verträgen für eingehende Daten von Java EE Connector Architecture 1.5 basieren. Die Schnittstelle **EISExportBinding** macht es ebenfalls erforderlich, dass ein Datenhandler oder eine Datenbindung entweder auf Bindungsebene oder auf Methodenebene angegeben ist.

JMS-Bindungen

Ein JMS-Provider (JMS = Java Message Service) ermöglicht den Nachrichtenaustausch basierend auf der API und dem Programmiermodell von Java Messaging Service. Er stellt JMS-Verbindungsfactorys für die Erstellung von Verbindungen für JMS-Ziele sowie zum Senden und Empfangen von Nachrichten bereit.

JMS-Bindungen können verwendet werden, wenn Sie mit der Bindung des SIB-Providers (Service Integration Bus) interagieren und mit JMS und JCA 1.5 kompatibel sind.

Mithilfe der JMS-Exportbindungen und -Importbindungen kann ein SCA-Modul externe JMS-Systeme aufrufen und Nachrichten von diesen empfangen.

Die JMS-Importbindungen und -Exportbindungen ermöglichen die Integration bei JMS-Anwendungen, die den JCA 1.5-basierten SIB-JMS-Provider verwenden, der Teil von WebSphere Application Server ist. Andere JCA 1.5-basierte JMS-Ressourcenadapter werden nicht unterstützt.

JMS-Bindungen - Übersicht:

JMS-Bindungen stellen Konnektivität zwischen der SCA-Umgebung und JMS-Systemen bereit.

JMS-Bindungen

Die Hauptkomponenten sowohl bei JMS-Importbindungen als auch bei JMS-Exportbindungen sind Folgende:

- Ressourcenadapter: Er ermöglicht die verwaltete bidirektionale Konnektivität zwischen einem SCA-Modul und externen JMS-Systemen.
- Verbindungen: Sie binden eine virtuelle Verbindung zwischen einer Client- und einer Provideranwendung ein.
- Ziele: Sie werden von einem Client verwendet, um das Ziel der von ihm erstellten Nachrichten oder die Quelle der von ihm gelesenen Nachrichten anzugeben.
- Authentifizierungsdaten: Sie werden verwendet, um den Zugriff auf die Bindung zu schützen.

Schlüsselfunktionen von JMS-Bindings

Besondere Header

Die Eigenschaften besonderer Header werden in JMS-Importen und -Exporten dazu verwendet, dem Ziel mitzuteilen, wie die Nachricht verarbeitet werden soll.

Bei Verwendung von TargetFunctionName wird eine Zuordnung von der nativen Methode zur Operationsmethode vorgenommen.

Java EE-Ressourcen

Wenn JMS-Importe und -Exporte in einer Java EE-Umgebung implementiert werden, wird eine Reihe von Java EE-Ressourcen erstellt.

`ConnectionFactory`
Wird von Clients zum Erstellen einer Verbindung zum JMS-Provider verwendet.

`ActivationSpec`
Wird von Importen zum Empfangen der Antwort auf eine Anforderung verwendet; wird von Exporten zum Konfigurieren der Nachrichtenendpunkte verwendet, die Nachrichtenlistener in ihrer Interaktion mit dem Messaging-System darstellen.

Ziele

- Sendeziel: Für einen Import ist dies das Ziel, an das die Anforderung oder Nachricht gesendet wird; für einen Export ist dies das Ziel, an das die Antwortnachricht gesendet wird, falls diese nicht durch das Headerfeld JMSReplyTo in der eingehenden Nachricht außer Kraft gesetzt wird.
- Empfangsziel: Das Ziel, an das die eingehende Nachricht gesendet werden soll; bei Importen ist es eine Antwort, bei Exporten eine Anforderung.
- Callbackziel: Das SCA-JMS-Systemziel, das zum Speichern der Korrelationsinformationen verwendet wird. Nehmen Sie an dieser Adresse keine Lese- oder Schreiboperationen vor.

Von der Installationstask werden die Verbindungsfactory (ConnectionFactory) und drei Ziele erstellt. Außerdem wird von ihr die Aktivierungsspezifikation (ActivationSpec) zum Aktivieren des Laufzeitnachrichtenlisteners zur Überwachung auf Nachrichten am Empfangsziel erstellt. Die Eigenschaften dieser Ressourcen werden in der Import- oder Exportdatei angegeben.

JMS-Integration und Ressourcenadapter:

Java Message Service (JMS) ermöglicht die Integration durch einen verfügbaren, auf JMS JCA 1.5 basierenden Ressourcenadapter. Die vollständige Unterstützung für die JMS-Integration wird für den SIB-JMS-Ressourcenadapter bereitgestellt (SIB = Service Integration Bus).

Verwenden Sie einen Ressourcenadapter für einen JMS-Provider für JCA 1.5, wenn Sie eine Integration bei einem externen JCA 1.5-konformen JMS-System wünschen. Externe Services, die mit JCA 1.5 konform sind, können mit dem SIB-JMS-Ressourcenadapter Nachrichten empfangen und senden, um sich bei den SCA-Komponenten zu integrieren.

Die Verwendung anderer providerspezifischer JCA 1.5-Ressourcenadapter wird nicht unterstützt.

JMS-Importbindungen und -Exportbindungen:

Mithilfe von JMS-Importbindungen und -Exportbindungen können Sie dafür sorgen, dass SCA-Module mit Services interagieren, die von externen JMS-Anwendungen bereitgestellt werden.

JMS-Importbindungen

Verbindungen zum zugehörigen JMS-Provider von JMS-Zielen werden unter Verwendung einer JMS-Verbindungsfactory erstellt. Mit Verwaltungsobjekten für Verbindungsfactorys können Sie JMS-Verbindungsfactorys für den Standard-Messaging-Provider verwalten.

Die Interaktion mit externen JMS-Systemen umfasst auch die Verwendung von Zielen für das Senden von Anforderungen und das Empfangen von Antworten.

Es werden zwei Typen von Einsatzszenarios für JMS-Importbindungen unterstützt, die sich nach dem Typ der aufgerufenen Operation richten:

- Unidirektional: Der JMS-Import übergibt eine Nachricht an das Sendeziel, das in der Importbindung konfiguriert ist. Das Feld **replyTo** des JMS-Headers enthält keine Angaben.
- Bidirektional (Anforderung/Antwort): Der JMS-Import übergibt eine Nachricht an das Sendeziel und behält dann die von der SCA-Komponente empfangene Antwort bei.

  Die Importbindung kann (mit dem Feld **Korrelationsschema für Antwort** in Integration Designer) so konfiguriert werden, dass das Kopieren der Korrelations-ID für die Antwortnachricht aus der Anforderungsnachrichten-ID (Standardeinstellung) oder aus der Korrelations-ID für die Anforderungsnachricht erwartet wird. Außerdem kann für die Importbindung die Verwendung eines temporären dynamischen Antwortziels konfiguriert werden, um Antworten mit Anforderungen zu korrelieren. Für jede Anforderung wird ein temporäres Ziel erstellt. Der Import verwendet dieses Ziel, um die Antwort zu empfangen.

Das Empfangsziel (**receive**) ist in der Headereigenschaft **replyTo** der abgehenden Nachricht festgelegt. Für die Überwachung des Empfangsziels wird ein Nachrichtenlistener implementiert. Sobald eine Antwort empfangen wird, übergibt der Nachrichtenlistener die Antwort zurück an die Komponente.

Sowohl bei unidirektionalen als auch bei bidirektionalen Einsatzszenarios können dynamische und statische Headereigenschaften angegeben werden. Statische Eigenschaften können über die Methodenbindung des JMS-Imports festgelegt werden. Manche dieser Eigenschaften haben in der SCA-JMS-Laufzeit eine besondere Bedeutung.

Es muss unbedingt beachtet werden, dass es sich bei JMS um eine asynchrone Bindung handelt. Falls eine aufrufende Komponente einen JMS-Import (bei einer bidirektionalen Operation) im Synchronmodus aufruft, wird die aufrufende Komponente blockiert, bis die Antwort durch den JMS-Service zurückgegeben wurde.

In Abb. 32 auf Seite 125 ist dargestellt, wie der Import mit dem externen Service verknüpft ist.

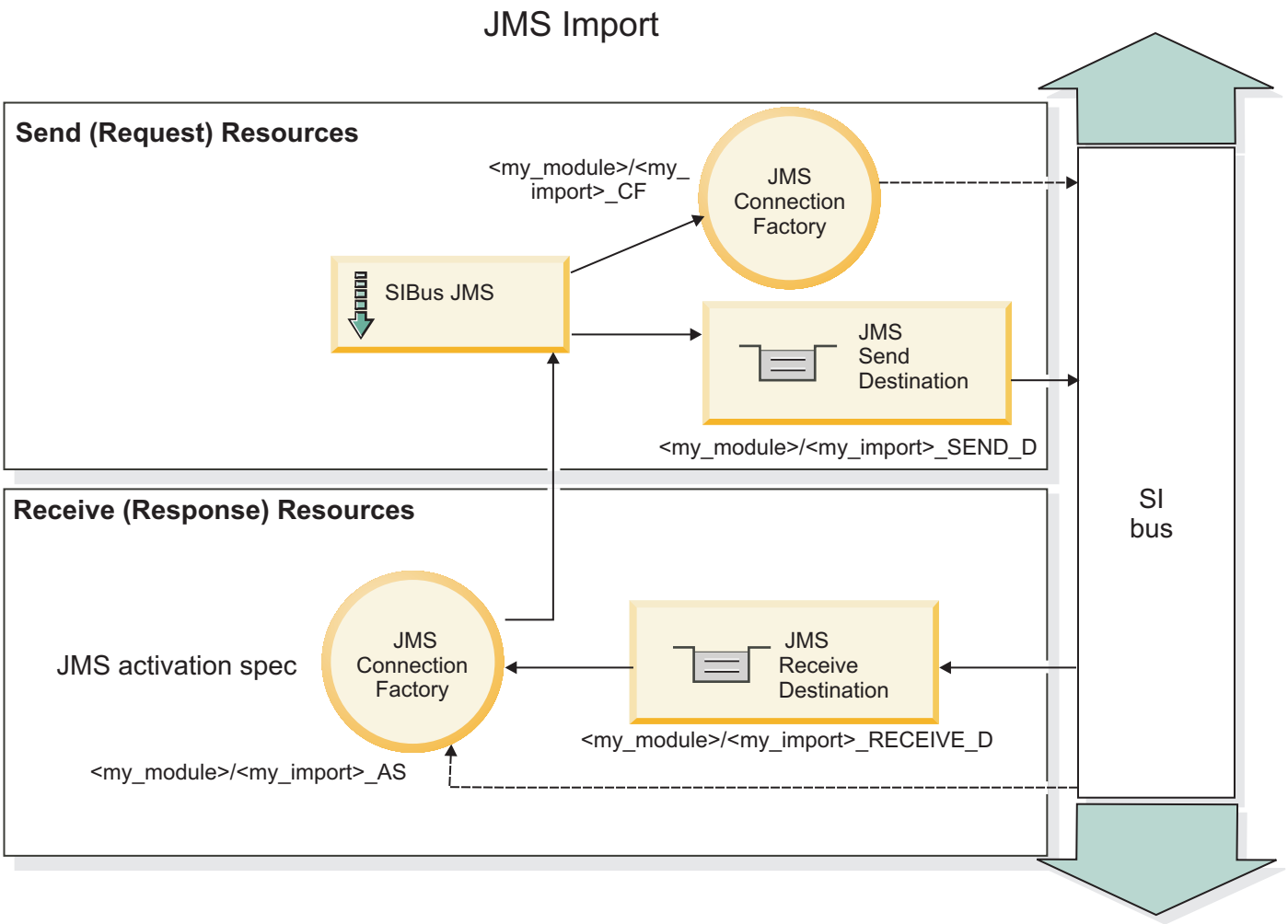


Abbildung 77. Ressourcen der JMS-Importbindung

JMS-Exportbindungen

JMS-Exportbindungen stellen SCA-Modulen ein Verfahren bereit, mit dem Services für externe JMS-Anwendungen angeboten werden können.

Bei der Verbindung, die Teil eines JMS-Exports ist, handelt es sich um eine konfigurierbare Aktivierungsspezifikation.

Ein JMS-Export besitzt Sende- und Empfangsziele.

- Das Empfangsziel ist die Position, an der die eingehende Nachricht für die Zielkomponente übergeben werden soll.
- Das Sendeziel ist die Position, an die die Antwort gesendet wird, sofern diese Angabe in der eingehenden Nachricht nicht durch die Headereigenschaft **replyTo** überschrieben wurde.

Zur Überwachung von Anforderungen, die an dem in der Exportbindung angegebenen **Empfangsziel** eingehen, wird ein Nachrichtenlistener implementiert. Das im Feld **send** angegebene Ziel wird verwendet, um die Antwort auf die eingehende Anforderung zu senden, falls die aufgerufene Komponente eine Antwort bereitstellt. Das im Feld **replyTo** der eingehenden Nachricht angegebene Ziel überschreibt das im Feld **send** angegebene Ziel.

In Abb. 33 auf Seite 126 ist dargestellt, wie der externe Anforderer mit dem Export verknüpft ist.

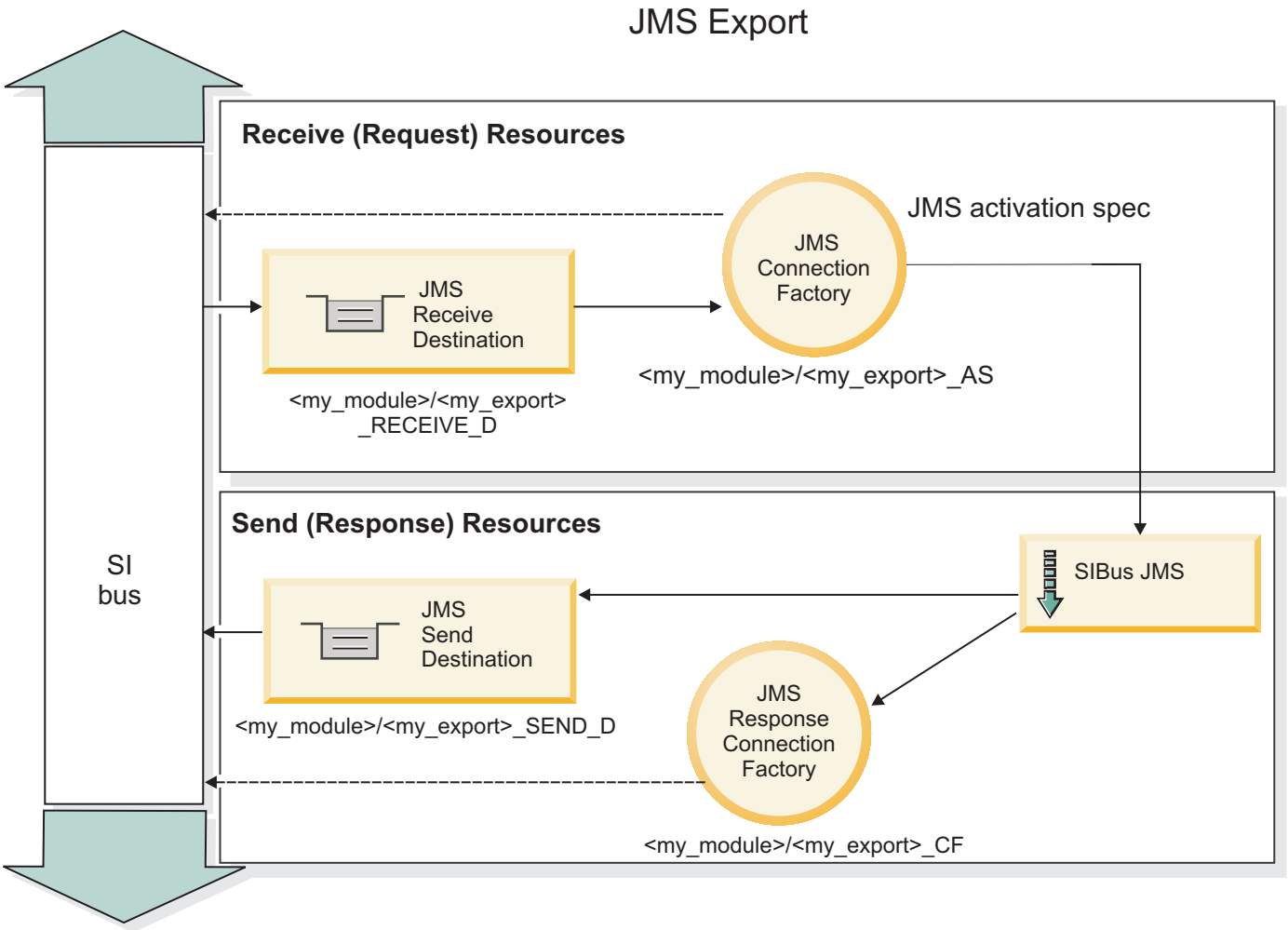


Abbildung 78. Ressourcen der JMS-Exportbindung

JMS-Header:

Eine JMS-Nachricht enthält zwei Typen von Headern, nämlich den JMS-Systemheader und mehrere JMS-Eigenschaften. Auf beide Headertypen kann entweder in einem Mediationsmodul im Servicenachrichtenobjekt (Service Message Object - SMO) oder unter Verwendung der API 'ContextService' zugegriffen werden.

JMS-Systemheader

Der JMS-Systemheader wird im Servicenachrichtenobjekt durch das Element 'JMSHeader' repräsentiert, das alle normalerweise in einem JMS-Header zu findenden Felder enthält. Obwohl diese Felder in der Mediation (oder der API 'ContextService') geändert werden können, werden einige der im Servicenachrichtenobjekt festgelegten Felder für den JMS-Systemheader nicht an die abgehende JMS-Nachricht weitergegeben, weil sie durch Systemwerte oder statische Werte überschrieben werden.

Die folgenden Schlüsselfelder im JMS-Systemheader können in einer Mediation (oder einer API 'ContextService') aktualisiert werden:

- **JMSType** und **JMSCorrelationID** - Werte der spezifischen vordefinierten Nachrichtenheadereigenschaften.
- **JMSDeliveryMode**: Werte für den Übermittlungsmodus ('persistent' oder 'nonpersistent'; Standardwert ist 'persistent').
- **JMSPriority**: Prioritätswert (0 bis 9; Standardwert ist 'JMS_Default_Priority').

JMS-Eigenschaften

JMS-Eigenschaften werden im Servicenachrichtenobjekt als Einträge in der Eigenschaftsliste ('Properties') dargestellt. Die Eigenschaften können in einer Mediation oder durch die Verwendung der API 'ContextService' hinzugefügt, aktualisiert oder gelöscht werden.

Eigenschaften können auch in der JMS-Bindung statisch festgelegt sein. Statisch festgelegte Eigenschaften überschreiben Einstellungen desselben Namens, die dynamisch festgelegt werden.

Benutzereigenschaften, die von anderen Bindungen (z. B. einer HTTP-Bindung) weitergegeben werden, werden in der JMS-Bindung als JMS-Eigenschaften ausgegeben.

Einstellungen für Headerweitergabe

Die Weitergabe des JMS-Systemheaders und der JMS-Eigenschaften entweder von der eingehenden JMS-Nachricht an nachgelagerte Komponenten oder von vorgelagerten Komponenten an die abgehende JMS-Nachricht kann durch das Attribut 'Protokollheader weitergeben' der Bindung gesteuert werden.

Wenn das Attribut 'Protokollheader weitergeben' festgelegt ist, ist die Übermittlung von Headerinformationen an die Nachricht oder die Zielkomponente, wie in der folgenden Liste beschrieben, zulässig:

- JMS-Exportanforderung

  Der in der Nachricht empfangene JMS-Header wird mittels des Kontextservice an Zielkomponenten weitergegeben. In der Nachricht empfangene JMS-Eigenschaften werden mittels des Kontextservice an Zielkomponenten weitergegeben.
- JMS-Exportantwort

  Alle im Kontextservice definierten JMS-Headerfelder werden in der abgehenden Nachricht verwendet, sofern sie nicht durch statische Eigenschaften überschrieben werden, die in der JMS-Exportbindung festgelegt sind. Alle im Kontextservice definierten Eigenschaften werden in der abgehenden Nachricht verwendet, sofern sie nicht durch statische Eigenschaften überschrieben werden, die in der JMS-Exportbindung festgelegt sind.
- JMS-Importanforderung

Alle im Kontextservice definierten JMS-Headerfelder werden in der abgehenden Nachricht verwendet, sofern sie nicht durch statische Eigenschaften überschrieben werden, die in der JMS-Importbindung festgelegt sind. Alle im Kontextservice definierten Eigenschaften werden in der abgehenden Nachricht verwendet, sofern sie nicht durch statische Eigenschaften überschrieben werden, die in der JMS-Importbindung festgelegt sind.

- JMS-Importantwort

  Der in der Nachricht empfangene JMS-Header wird mittels des Kontextservice an Zielkomponenten weitergegeben. In der Nachricht empfangene JMS-Eigenschaften werden mittels des Kontextservice an Zielkomponenten weitergegeben.

JMS-Korrelationsschema für temporäres dynamisches Antwortziel:

Das Korrelationsschema für temporäre dynamische Antwortziele bewirkt, dass für jede gesendete Anforderung eine eindeutige dynamische Warteschlange oder bzw. ein eindeutiges dynamisches Topic erstellt wird.

Anhand des im Import angegebenen statischen Antwortziels wird die Spezifik der temporären dynamischen Zielwarteschlange bzw. des Topics abgeleitet. Dies ist im Feld **ReplyTo** der Anforderung festgelegt. Der JMS-Import überwacht dieses Ziel auf Antworten. Sobald die Antwort empfangen wird, wird sie am statischen Antwortziel zur asynchronen Verarbeitung erneut in die Warteschlange eingereiht. Das Feld **CorrelationID** der Antwort wird nicht verwendet und muss nicht festgelegt sein.

Transaktionsbezogene Aspekte

Bei Verwendung eines temporären dynamischen Ziels muss die Antwort in demselben Thread wie die gesendete Antwort gelesen werden. Die Anforderung muss außerhalb der globalen Transaktion gesendet und festgeschrieben werden, bevor sie durch den Back-End-Service empfangen und eine Antwort zurückgegeben wird.

Persistenz

Temporäre dynamische Warteschlangen sind kurzlebige Entitäten, die nicht dasselbe Persistenzniveau wie eine statische Warteschlange oder ein statisches Topic gewährleisten. Eine temporäre dynamische Warteschlange bzw. ein solches Topic bleibt - wie auch die Nachrichten - bei einem Serverneustart nicht erhalten. Nachdem die Nachricht am statischen Antwortziel erneut in die Warteschlange eingereiht wurde, behält sie die in der Nachricht definierte Persistenz bei.

Zeitlimit

Der Import wartet für eine festgelegte Dauer auf eine Antwort am temporären dynamischen Antwortziel. Dieses Zeitintervall wird aus dem SCA-Qualifikationsmerkmal für den Ablauf der Antwort übernommen, falls es festgelegt wird. Andernfalls beträgt die Wartezeit standardmäßig 60 Sekunden. Wird die Wartezeit überschritten, löst der Import eine Ausnahmebedingung des Typs `ServiceTimeoutRuntimeException` aus.

Externe Clients:

Der Server kann mit JMS-Bindungen Nachrichten an externe Clients senden bzw. von diesen empfangen.

Ein externer Client (z. B. ein Webportal oder ein unternehmensweites Informationssystem) kann eine Nachricht an ein SCA-Modul im Server senden oder durch eine Komponente im Server aufgerufen werden.

Die JMS-Exportkomponenten implementieren Nachrichtenlistener, um Anforderungen zu überwachen, die bei dem in der Exportbindung definierten Empfangsziel eingehen. Das im Sendefeld angegebene Ziel

wird verwendet, um die Antwort auf die eingehende Anforderung zu senden, falls die aufgerufene Anwendung eine Antwort bereitstellt. Ein externer Client kann somit Anwendungen über die Exportbindung aufrufen.

JMS-Importe interagieren mit externen Clients, indem sie Nachrichten an JMS-Warteschlangen senden oder aus diesen abrufen.

Mit externen Clients arbeiten:

Ein externer Client, d. h. ein Client außerhalb des Servers, muss unter Umständen mit einer Anwendung interagieren, die auf dem Server installiert ist.

Stellen Sie sich ein sehr einfaches Szenario vor, bei dem ein externer Client mit einer Anwendung auf dem Server interagieren möchte. Die Abbildung stellt ein typisches einfaches Szenario dar.

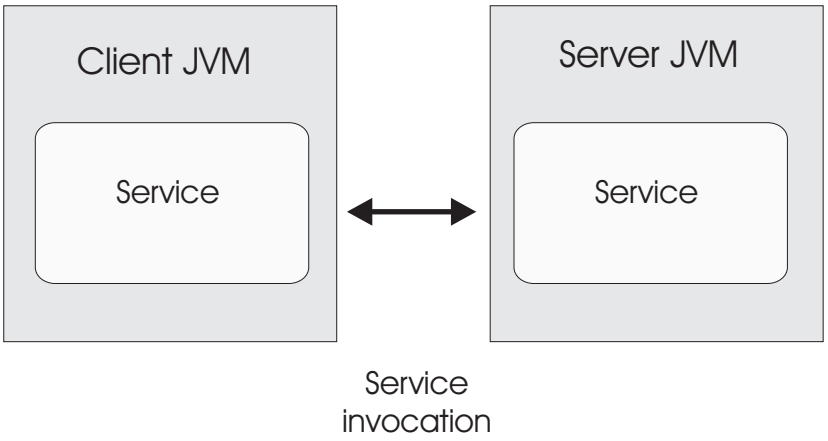


Abbildung 79. Einfaches Anwendungsfallszenario: Externer Client interagiert mit Serveranwendung

Die SCA-Anwendung beinhaltet einen Export mit einer JMS-Bindung. Hierdurch wird die Anwendung auch für externe Clients verfügbar.

Wenn Sie über einen externen Client in einer Java Virtual Machine (JVM) verfügen, der sich nicht auf Ihrem Server befindet, so müssen Sie mehrere Schritte ausführen, um eine Verbindung herzustellen und mit einem JMS-Export zu interagieren. Der Client bezieht einen Ausgangskontext (InitialContext) mit den korrekten Werten und sucht dann die Ressourcen über JNDI. Der Client greift unter Verwendung des JMS 1.1-Spezifikationsclients auf das Ziel zu und sendet bzw. empfängt Nachrichten auf dem Ziel.

Die JNDI-Standardnamen der Ressourcen, die automatisch von der Laufzeit erstellt werden, sind in diesem Abschnitt unter 'Konfiguration' aufgelistet. Wenn Sie jedoch im Vorfeld Ressourcen erstellt haben, verwenden Sie diese JNDI-Namen.

1. Konfigurieren Sie JMS-Ziele und die Verbindungsfactory für das Senden der Nachricht.
2. Stellen Sie sicher, dass der JNDI-Kontext, der Port für den SIB-Ressourcenadapter und der Messaging-Bootstrapping-Port korrekt sind.

   Der Server verwendet einige Standardports, aber wenn auf diesem System weitere Server installiert sind, werden zum Zeitpunkt der Installation alternative Ports erstellt, um eventuelle Konflikte mit anderen Serverinstanzen zu vermeiden. Über die Administrationskonsole können bestimmen, welche Ports Ihr Server verwendet. Gehen Sie zu **Server** > **Anwendungsserver** > ***Name_Ihres_Servers*** > **Konfiguration** und klicken Sie unter **Kommunikation** auf **Ports**. Dann können Sie den Port bearbeiten, der verwendet wird.
3. Der Client bezieht einen Ausgangskontext mit den korrekten Werten und sucht dann die Ressourcen über JNDI.

4. Der Client greift unter Verwendung von JMS 1.1-Spezifikationen auf die Ziele und auf die Sende- und Empfangsnachrichten auf den Zielen.

Fehlerbehebung für JMS-Bindungen:

Sie können Probleme im Zusammenhang mit JMS-Bindungen diagnostizieren und beheben.

Ausnahmebedingungen bei der Implementierung

Als Reaktion auf diverse Fehlerbedingungen kann die JMS-Implementierung für Importe und Exporte zwei Typen von Ausnahmebedingungen zurückgeben:

- Geschäftsausnahmebedingung für Service: Diese Ausnahmebedingung wird zurückgegeben, wenn der für die Geschäftsschnittstelle für den Service (Typ WSDL-Port) definierte Fehler aufgetreten ist.
- Laufzeitausnahmebedingung für den Service: Diese Ausnahmebedingung wird in allen übrigen Fällen ausgelöst. In den meisten Fällen enthält die Ausnahmebedingung vom Typ cause (Ursache) die ursprüngliche Ausnahmebedingung (JMSException).

  Ein Import erwartet zum Beispiel nur eine einzige Antwortnachricht für jede Anforderungsnachricht. Wenn mehr als eine Antwort eingeht oder wenn eine verspätete Antwort eingeht (d. h. eine Antwort, für die die definierte SCA-Antwortablauffrist verstrichen ist), so wird eine Laufzeitausnahmebedingung für den Service ausgelöst. Für die Transaktion wird ein Rollback ausgeführt und die Antwortnachricht wird aus der Warteschlange entfernt oder vom Failed Event Manager verarbeitet.

Primäre Fehlerbedingungen

Die primären Fehlerbedingungen von JMS-Bindungen werden durch Transaktionssemantik, durch die JMS-Providerkonfiguration oder durch Verweise auf bestehendes Verhalten in anderen Komponenten bestimmt. Zu den primären Fehlerbedingungen zählen die Folgenden:

- Die Herstellung der Verbindung zum JMS-Provider oder Ziel ist fehlgeschlagen.

  Das Fehlschlagen der Verbindungsherstellung zum JMS-Provider für den Empfang von Nachrichten hat zur Folge, dass der Nachrichtenlistener nicht gestartet werden kann. Dieser Zustand wird im WebSphere Application Server-Protokoll aufgezeichnet. Persistente Nachrichten verbleiben so lange au f dem Ziel, bis sie erfolgreich abgerufen wurden (oder abgelaufen sind).

  Das Fehlschlagen der Verbindungsherstellung zum JMS-Provider für den Versand abgehender Nachrichten bewirkt ein Rollback der Transaktion, die die Sendeaktion steuert.
- Die Ausführung der Syntaxanalyse für eine eingehende Nachricht oder die Erstellung einer abgehenden Nachricht ist fehlgeschlagen.

  Ein Fehler in der Datenbindung oder im Datenhandler bewirkt ein Rollback der Transaktion, die die Arbeit steuert.
- Das Senden der abgehenden Nachricht ist fehlgeschlagen.

  Das Fehlschlagen des Versands einer Nachricht bewirkt ein Rollback der relevanten Transaktion.
- Mehrere oder nicht erwartete verspätete Antwortnachrichten.

  Der Import erwartet für jede Anforderungsnachricht jeweils nur eine Antwortnachricht. Der Zeitraum, in dem der Empfang einer Antwort gültig ist, wird durch das SCA-Qualifikationsmerkmal für den Ablauf der Antwort bestimmt, das für die Anforderung festgelegt ist. Wenn eine Antwort eintrifft oder die Verfallszeit überschritten wird, so wird der Korrelationsdatensatz gelöscht. Treffen Antwortnachrichten unerwartet oder verspätet ein, so wird eine Laufzeitausnahmebedingung für den Service ausgelöst.
- Laufzeitausnahmebedingung durch Servicelimit, verursacht durch verspätete Antwort bei Verwendung des Korrelationsschemas für temporäre dynamische Antwortziele.

  Der JMS-Import überschreitet nach Ablauf einer durch das SCA-Qualifikationsmerkmal für den Ablauf der Antwort bestimmten Zeitspanne das Zeitlimit. Ist kein Zeitlimit definiert, gilt als Zeitlimit standardmäßig eine Zeitspanne von 60 Sekunden.

JMS-basierte SCA-Nachrichten werden im Failed Event Manager nicht angezeigt

Wenn die SCA-Nachrichten ihren Ursprung in einem Fehler oder einer Störung in der JMS-Interaktion haben, würden Sie erwarten, diese Nachrichten im Failed Event Manager vorzufinden. Werden keine derartigen Nachrichten im Failed Event Manager angezeigt, stellen Sie sicher, dass für das dem JMS-Ziel zugrunde liegende SIB-Ziel ein Wert größer als **1** für die maximale Anzahl fehlgeschlagener Zustellungen definiert ist. Durch Festlegen von **2** oder höher für diesen Wert wird eine Interaktion mit dem Failed Event Manager während SCA-Aufrufen für die JMS-Bindungen ermöglicht.

Ausnahmebedingungen verarbeiten:

Die Art und Weise, in der die Bindung konfiguriert ist, bestimmt, wie Ausnahmebedingungen verarbeitet werden, die von Datenhandlern oder Datenbindungen ausgelöst werden. Außerdem gibt die Spezifik des Mediationsablaufs das Verhalten des Systems vor, wenn eine solche Ausnahmebedingung ausgelöst wird.

Wenn ein Datenhandler oder eine Datenbindung durch eine Bindung aufgerufen wird, können verschiedene Probleme auftreten. Beispielsweise kann es sein, dass ein Datenhandler eine Nachricht mit beschädigten Nutzdaten empfängt oder er versucht, eine Nachricht mit einem falschen Format zu lesen.

Das Verfahren, mit dem die Bindung eine solche Ausnahmebedingung behandelt, wird dadurch bestimmt, wie Sie den Datenhandler oder die Datenbindung implementieren. Das empfohlene Verhalten besteht darin, die Datenbindung so zu konfigurieren, dass sie eine Ausnahmebedingung des Typs **DataBindingException** auslöst.

Wenn eine Laufzeitausnahmebedingung (inklusive **DataBindingException**) ausgelöst wird, geschieht Folgendes:

- Falls der Mediationsablauf transaktionsorientiert konfiguriert ist, wird die JMS-Nachricht standardmäßig in Failed Event Manager gespeichert, damit sie manuell wiedergegeben oder gelöscht werden kann.

  Anmerkung: Sie können den Fehlerbehebungsmodus für die Bindung so ändern, dass die Nachricht zurückgesetzt und nicht in Failed Event Manager gespeichert wird.
- Ist der Mediationsablauf nicht transaktionsorientiert, wird die Ausnahmebedingung protokolliert und die Nachricht ist nicht mehr vorhanden.

Bei einem Datenhandler ist die Situation ähnlich. Da der Datenhandler durch die Datenbindung aufgerufen wird, wird jede Ausnahmebedingung des Datenhandlers in eine Ausnahmebedingung der Datenbindung eingeschlossen. Daher wird eine Ausnahmebindung des Typs **DataHandlerException** als Typ **DataBindingException** gemeldet.

Generische JMS-Bindungen

Die generische JMS-Bindung stellt Konnektivität für JMS 1.1-konforme Provider von Drittherstellern bereit. Der Operation der generischen JMS-Bindung ähnelt der Operation von JMS-Bindungen.

Mit dem durch eine JMS-Bindung bereitgestellten Service kann ein SCA-Modul Aufrufe an externe Systeme absetzen oder Nachrichten von diesen Systemen empfangen. Bei dem System kann es sich um ein externes JMS-System handeln.

Die generische JMS-Bindung ermöglicht die Integration bei nicht mit JCA 1.5-konformen JMS-Providern, die JMS 1.1 unterstützen und die optionale JMS-Anwendungsserverfunktion implementieren. Die generische JMS-Bindung unterstützt diejenigen JMS-Provider (einschließlich Oracle AQ, TIBCO, SonicMQ, WebMethods und BEA WebLogic), die JCA 1.5 nicht unterstützen, jedoch mit einer Unterstützung der Anwendungsserverfunktion gemäß der JMS 1.1-Spezifikation ausgestattet sind. Der in WebSphere integrierte JMS-Provider (SIBJMS), bei dem es sich um einen JCA 1.5-JMS-Provider handelt, wird durch diese Bindung nicht unterstützt. Bei Verwendung dieses Providers verwenden Sie die „JMS-Bindungen" auf Seite 122.

Verwenden Sie diese generische Bindung zur Integration bei einem nicht mit JCA 1.5 konformen JMS-basierten System in einer SCA-Umgebung. Die externen Zielanwendungen können dann zur Integration bei einer SCA-Komponente Nachrichten empfangen und senden.

Generische JMS-Bindungen - Übersicht:

Generische JMS-Bindungen sind JMS-Bindungen ohne JCA, die Konnektivität zwischen der SCA-Umgebung und JMS-Systemen bereitstellen, die mit JMS 1.1 konform sind und die optionale Anwendungsserverfunktion implementieren.

Generische JMS-Bindungen

Die Hauptaspekte von generischen JMS-Importbindungen und -Exportbindungen sind Folgende:

- Listener-Port: Er ermöglicht nicht auf JCA basierenden JMS-Providern den Empfang von Nachrichten und deren Zuteilung zu einer nachrichtengesteuerten Bean (Message Driven Bean - MDB).
- Verbindungen: Sie binden eine virtuelle Verbindung zwischen einer Client- und einer Provideranwendung ein.
- Ziele: Sie werden von einem Client verwendet, um das Ziel der von ihm erstellten Nachrichten oder die Quelle der von ihm gelesenen Nachrichten anzugeben.
- Authentifizierungsdaten: Sie werden verwendet, um den Zugriff auf die Bindung zu schützen.

Generische JMS-Importbindungen

Generische JMS-Importbindungen ermöglichen den Komponenten in einem SCA-Modul die Kommunikation mit Services, die durch externe und nicht mit JCA 1.5 konforme JMS-Provider bereitgestellt werden.

Der Verbindungsteil eines JMS-Imports ist eine Verbindungsfactory. Eine Verbindungsfactory, also das Objekt, mit dem der Client eine Verbindung zu einem Provider herstellt, bindet eine Reihe von Verbindungskonfigurationsparametern ein, die durch einen Administrator definiert werden. Jede Verbindungsfactory ist eine Instanz der Schnittstelle **ConnectionFactory**, **QueueConnectionFactory** oder **TopicConnectionFactory**.

Die Interaktion mit externen JMS-Systemen umfasst auch die Verwendung von Zielen für das Senden von Anforderungen und das Empfangen von Antworten.

Es werden zwei Typen von Einsatzszenarios für generische JMS-Importbindungen unterstützt, die sich nach dem Typ der aufgerufenen Operation richten:

- Unidirektional: Der generische JMS-Import übergibt eine Nachricht an das Sendeziel, das in der Importbindung konfiguriert ist. An das Feld **replyTo** des JMS-Headers werden keine Daten gesendet.
- Bidirektional (Anforderung/Antwort): Der generische JMS-Import übergibt eine Nachricht an das Sendeziel und behält dann die von der SCA-Komponente empfangene Antwort bei.

  Das Empfangsziel (**receive**) ist in der Headereigenschaft **replyTo** der abgehenden Nachricht festgelegt. Für die Überwachung des Empfangsziels wird eine nachrichtengesteuerte Bean implementiert. Sobald eine Antwort empfangen wird, übergibt die nachrichtengesteuerte Bean die Antwort zurück an die Komponente.

  Die Importbindung kann (mit dem Feld **Korrelationsschema für Antwort** in Integration Designer) so konfiguriert werden, dass das Kopieren der Korrelations-ID für die Antwortnachricht aus der Anforderungsnachrichten-ID (Standardeinstellung) oder aus der Korrelations-ID für die Anforderungsnachricht erwartet wird.

Sowohl bei unidirektionalen als auch bei bidirektionalen Einsatzszenarios können dynamische und statische Headereigenschaften angegeben werden. Statische Eigenschaften können über die Methodenbindung des generischen JMS-Imports festgelegt werden. Manche dieser Eigenschaften haben in der SCA-JMS-Laufzeit eine besondere Bedeutung.

Es muss unbedingt beachtet werden, dass es sich bei der generischen JMS-Bindung um eine asynchrone Bindung handelt. Falls eine aufrufende Komponente einen generischen JMS-Import (bei einer bidirektionalen Operation) im Synchronmodus aufruft, wird die aufrufende Komponente blockiert, bis die Antwort durch den JMS-Service zurückgegeben wurde.

In Abb. 35 auf Seite 133 ist dargestellt, wie der Import mit dem externen Service verknüpft ist.

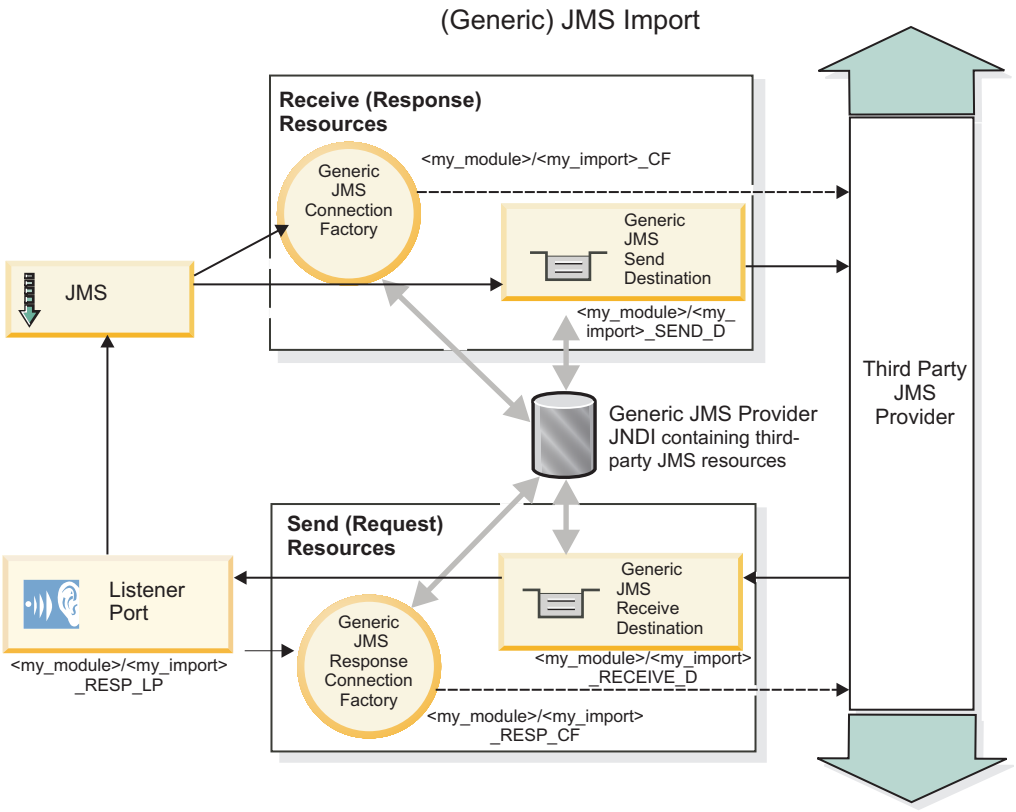


Abbildung 80. Ressourcen der generischen JMS-Importbindung

Generische JMS-Exportbindungen

Von generischen JMS-Exportbindungen wird SCA-Modulen ein Verfahren bereitgestellt, mit dem Services für externe JMS-Anwendungen bereitgestellt werden können.

Der Verbindungsteil eines JMS-Exports besteht aus einer Verbindungsfactory (ConnectionFactory) und einem Listener-Port (ListenerPort).

Ein generischer JMS-Export besitzt Sende- und Empfangsziele.

- Das Ziel receive ist die Position, an der die eingehende Nachricht für die Zielkomponente übergeben werden soll.
- Das Ziel send ist die Position, an die die Antwort gesendet wird, sofern diese Angabe in der eingehenden Nachricht nicht durch die Headereigenschaft replyTo überschrieben wurde.

Zur Überwachung von Anforderungen, die an dem in der Exportbindung angegebenen Ziel receive eingehen, wird eine nachrichtengesteuerte Bean implementiert.

- Das im Feld send angegebene Ziel wird verwendet, um die Antwort auf die eingehende Anforderung zu senden, falls die aufgerufene Komponente eine Antwort bereitstellt.
- Das im Feld replyTo der eingehenden Nachricht angegebene Ziel überschreibt das im Feld send angegebene Ziel.

- Bei Anforderungs-/Antwortszenarios kann die Importbindung (über das Feld **Korrelationsschema für Antwort** in Integration Designer) so konfiguriert werden, dass das Kopieren der Nachrichten-ID aus der Anforderung in das Feld **correlation ID** der Antwortnachricht (Standardverhalten) erwartet wird, oder die Antwort kann die Korrelations-ID der Anforderung in das Feld **correlation ID** der Antwortnachricht kopieren.

In Abb. 36 auf Seite 134 ist dargestellt, wie der externe Anforderer mit dem Export verknüpft ist.

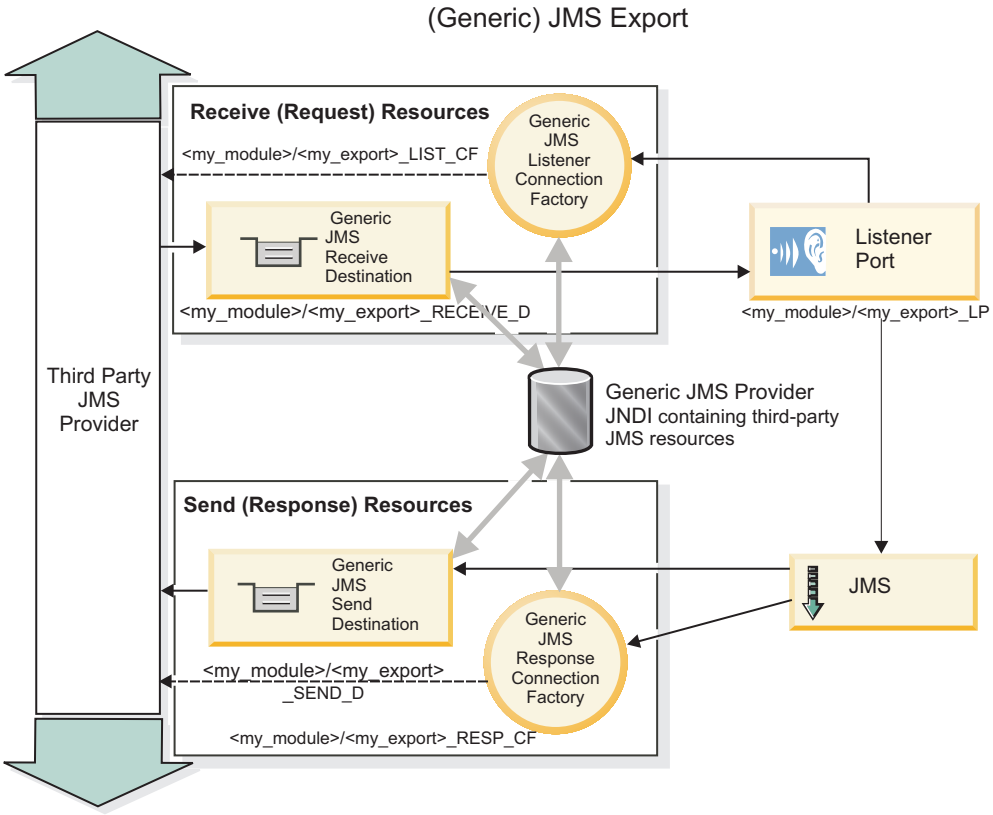


Abbildung 81. Ressourcen der generischen JMS-Exportbindung

Schlüsselfunktionen generischer JMS-Bindings:

Die Funktionen von generischen JMS-Importbindings und -Exportbindings sind mit den Funktionen der eingebetteten WebSphere JMS- und MQ-JMS-Importbindings konsistent. Zu den Schlüsselfunktionen gehören Headerdefinitionen und der Zugriff auf vorhandene Java EE-Ressourcen. Aufgrund ihrer generischen Natur verfügen sie jedoch nicht über Verbindungsoptionen, die für einen bestimmten JMS-Provider spezifisch sind und die Möglichkeiten eines solchen Bindings zum Generieren von Ressourcen im Rahmen der Implementierung und Installation sind eingeschränkt.

Generische Importe

Wie eine MQ-JMS-Importanwendung ist auch eine generische JMS-Implementierung asynchron und unterstützt drei Aufrufe: unidirektional, bidirektional (Request-Response) und Callback.

Wenn ein JMS-Import implementiert wird, wird eine von der Laufzeitumgebung bereitgestellte Message-driven Bean (MDB) implementiert. Von der MDB wird überwacht, ob Antworten auf die Anforderungsnachricht gesendet werden. Die MDB ist dem Ziel zugeordnet (überwacht es), das mit der Anforderung im Headerfeld replyTo der JMS-Nachricht gesendet wurde.

Generische Exporte

Generische JMS-Exportbindings unterscheiden sich von EIS-Exportbindings im Umgang mit der Rückgabe des Ergebnisses. Von einem generischen JMS-Export wird die Antwort explizit an das Ziel replyTo gesendet, das in der eingehenden Nachricht angegeben ist. Wenn keines angegeben ist, wird das Sendeziel verwendet.

Wenn ein generischer JMS-Export implementiert wird, wird eine Message-driven Bean (eine andere MDB als die für die generischen JMS-Importe) implementiert. Von ihr werden die eingehenden Anforderungen am Empfangsziel überwacht und anschließend die Anforderungen zur Verarbeitung durch die SCA-Laufzeit zugeteilt.

Besondere Header

Die Eigenschaften besonderer Header werden in generischen JMS-Importen und -Exporten dazu verwendet, dem Zielbinding mitzuteilen, wie die Nachricht verarbeitet werden soll.

Beispiel: Die Eigenschaft 'TargetFunctionName' wird standardmäßig vom Funktionsselektor dazu verwendet, den Namen der Operation in der Schnittstelle export zu ermitteln, die aufgerufen wird.

Anmerkung: Das Importbinding kann so konfiguriert werden, dass der Header 'TargetFunctionName' als Operationsname für jede Operation eingestellt wird.

Java EE-Ressourcen

Wenn ein JMS-Binding in einer Java EE-Umgebung implementiert wird, werden Java EE-Ressourcen erstellt.

- Listener-Port zum Überwachen am Empfangsziel (Antwort, nur bidirektional) für Importe und am Empfangsziel (Anforderung) für Exporte
- Generische JMS-Verbindungsfactory für abgehende Verbindung (Import) und eingehende Verbindung (Export)
- Generisches JMS-Ziel für Sendeziel (Import) und Empfangsziel (Export, nur bidirektional)
- Generische JMS-Verbindungsfactory für Antwortverbindung (bidirektional und optional; andernfalls wird die abgehende Verbindung für Exporte verwendet)
- Generisches JMS-Ziel für Empfangsziel (Import) und Sendeziel (Export, nur bidirektional)
- Callback-JMS-Ziel für Standard-Messaging-Provider für Zugriff auf Warteschlangenziel für SIB-Callback (nur bidirektional)
- Callback-JMS-Verbindungsfactory für Standard-Messaging-Provider für Zugriff auf Callback-JMS-Ziel (nur bidirektional)
- Warteschlangenziel für SIB-Callback zum Speichern der Anforderungsnachrichtendaten für die Antwortverarbeitung (nur bidirektional)

Von der Installationstask werden die Verbindungsfactory (ConnectionFactory), die drei Ziele und die Aktivierungsspezifikation (ActivationSpec) aus den Informationen in den Import- und Exportdateien erstellt.

Generische JMS-Header:

Generische JMS-Header sind Servicedatenobjekte (Service Data Objects - SDO), die alle Eigenschaften der generischen JMS-Nachrichteneigenschaften enthalten. Diese Eigenschaften können aus der eingehenden Nachricht stammen. Es kann sich aber auch um Eigenschaften handeln, die auf die abgehende Nachricht angewendet werden.

Eine JMS-Nachricht enthält zwei Typen von Headern, nämlich den JMS-Systemheader und mehrere JMS-Eigenschaften. Auf beide Headertypen kann entweder in einem Mediationsmodul im Servicenachrichtenobjekt (Service Message Object - SMO) oder unter Verwendung der API 'ContextService' zugegriffen werden.

Die folgenden Eigenschaften werden in der Methodenbindung (methodBinding) statisch festgelegt:

- JMSType
- JMSCorrelationID
- JMSDeliveryMode
- JMSPriority

Genauso wie die JMS- und die MQ-JMS-Bindung unterstützt die generische JMS-Bindung ebenfalls die dynamische Änderung von JMS-Headern und -Eigenschaften.

Einige generische JMS-Provider schränken die Eigenschaften und deren Kombination ein, die durch die Anwendung festgelegt werden können. Weitere Informationen entnehmen Sie der Dokumentation für das Produkt des Drittherstellers. Es wurde jedoch für 'methodBinding' die zusätzliche Eigenschaft 'ignoreInvalidOutboundJMSProperties' hinzugefügt, die eine Weitergabe aller Ausnahmebedingungen zulässt.

Die generischen JMS-Header und -Nachrichteneigenschaften werden nur dann verwendet, wenn der SCDL-Bindungsswitch der Basisservicekomponentenarchitektur aktiviert ist. Wenn der Switch aktiviert ist, werden Kontextinformationen weitergegeben. Dieser Switch ist standardmäßig aktiviert. Um die Weitergabe von Kontextinformationen zu verhindern, ändern Sie den Wert in **false**.

Bei aktivierter Kontextweitergabe können Headerinformationen an die Nachricht oder die Zielkomponente fließen. Um die Kontextweitergabe zu aktivieren oder zu inaktivieren, geben Sie den Wert **true** bzw. **false** für das Attribut 'contextPropagationEnabled' der Import- und Exportbindungen an. Beispiel:

```
<esbBinding xsi:type="eis:JMSImportBinding" contextProgagationEnabled="true">
```

Der Standardwert ist **true**.

Fehlerbehebung für generische JMS-Bindungen:

Sie können Probleme im Zusammenhang mit generischen JMS-Bindungen diagnostizieren und beheben.

Ausnahmebedingungen bei der Implementierung

Als Reaktion auf diverse Fehlerbedingungen kann die generische JMS-Implementierung für Importe und Exporte zwei Typen von Ausnahmebedingungen zurückgeben:

- Geschäftsausnahmebedingung für Service: Diese Ausnahmebedingung wird zurückgegeben, wenn der für die Geschäftsschnittstelle für den Service (Typ WSDL-Port) definierte Fehler aufgetreten ist.
- Laufzeitausnahmebedingung für den Service: Diese Ausnahmebedingung wird in allen übrigen Fällen ausgelöst. In den meisten Fällen enthält die Ausnahmebedingung vom Typ cause (Ursache) die ursprüngliche Ausnahmebedingung (JMSException).

Fehlerbehebung für den Ablauf generischer JMS-Nachrichten

Eine Anforderungsnachricht vom JMS-Provider kann ablaufen.

Die Bezeichnung *Anforderungsablauf* bezieht sich auf den Verfall der Gültigkeit oder das Ablaufen einer Anforderungsnachricht vom JMS-Provider, wenn die für JMSExpiration festgelegte Zeit für die Anforderungsnachricht erreicht wird. Wie bei anderen JMS-Bindungen handhabt die generische JMS-Bindung das Ablaufen der Anforderung, indem für die abgehende Anforderung denselben Wert für den Ablauf festlegt, wie für die Callback-Nachricht vom Import festgelegt wurde. Eine Benachrichtigung bezüglich des Ablaufs der Callback-Nachricht weist darauf hin, dass die Anforderungsnachricht abgelaufen ist und der Client über eine Geschäftsausnahmebedingung informiert werden sollte.

Wird das Callback-Ziel jedoch zu einem anderen Anbieter verlagert, wird diese Art von Anforderungsablauf (Gültigkeitsverfall von Anforderungen) nicht unterstützt.

Die Bezeichnung *Antwortablauf* bezieht sich auf den Verfall der Gültigkeit oder das Ablaufen einer Antwortnachricht vom JMS-Provider, wenn die für JMSExpiration festgelegte Zeit für die Antwortnachricht erreicht wird.

Die Funktion des Antwortablaufs für generische JMS-Bindungen wird nicht unterstützt, da das genaue Verhalten eines anderen JMS-Providers bezüglich des Verfalls der Gültigkeit nicht definiert ist. Sie können jedoch prüfen, ob die Antwort nicht abgelaufen ist, falls bzw. wenn sie empfangen wird.

Bei abgehenden Anforderungsnachrichten wird der Wert für JMSExpiration anhand der Wartezeit und der in asyncHeader mitgeführten Werte für requestExpiration berechnet, sofern diese festgesetzt wurden.

Fehlerbehebung für generische JMS-Verbindungsfactoryfehler

Wenn Sie bestimmte Typen von Verbindungsfactorys in Ihrem generischen JMS-Provider definieren, wird unter Umständen bei dem Versuch, eine Anwendung zu starten, eine Fehlernachricht angezeigt. Sie können die externe Verbindungsfactory solcherart ändern, dass dieses Problem vermieden wird.

Beim Starten einer Anwendung erhalten Sie möglicherweise eine Fehlernachricht ähnlich der folgenden:

```
Typ von 'JMSConnectionFactory' für MDB-Listener-Port stimmt nicht mit Typ von 'JMSDestination' überein
```

Dieses Problem kann beim Definieren externer Verbindungsfactorys auftreten. Die Ausnahmebedingung kann insbesondere dann ausgelöst werden, wenn Sie eine JMS 1.0.2-Topic-Verbindungsfactory anstelle einer (einheitlichen) JMS 1.1-Verbindungsfactory erstellen (d. h. anstelle einer Verbindungsfactory, die in der Lage ist, sowohl Punkt-zu-Punkt- als auch Publish/Subscribe-Kommunikation zu unterstützen).

Führen Sie die folgenden Schritte aus, um dieses Problem zu beheben:

1. Greifen Sie auf den generischen JMS-Provider zu, den Sie verwenden.
2. Ersetzen Sie die definierte JMS 1.0.2-Topic-Verbindungsfactory durch eine (einheitliche) JMS 1.1-Verbindungsfactory.

Wenn Sie die Anwendung nun mit der neu definierten JMS 1.1-Verbindungsfactory starten, dürfte Sie eigentlich keine Fehlernachricht mehr erhalten.

Generische JMS-basierte SCA-Nachrichten werden im Failed Event Manager nicht angezeigt

Wenn die SCA-Nachrichten ihren Ursprung in einem Fehler oder einer Störung in der generischen JMS-Interaktion haben, würden Sie erwarten, diese Nachrichten im Failed Event Manager vorzufinden. Werden keine derartigen Nachrichten im Failed Event Manager angezeigt, stellen Sie sicher, dass für den zugrunde liegenden Listener-Port ein Wert größer als 1 für die Eigenschaft der maximalen Anzahl von Wiederholungsversuchen definiert ist. Durch Festlegen von 2 oder höher für diesen Wert wird eine Interaktion mit dem Failed Event Manager während SCA-Aufrufen für die generischen JMS-Bindungen ermöglicht.

Ausnahmebedingungen verarbeiten:

Die Art und Weise, in der die Bindung konfiguriert ist, bestimmt, wie Ausnahmebedingungen verarbeitet werden, die von Datenhandlern oder Datenbindungen ausgelöst werden. Außerdem gibt die Spezifik des Mediationsablaufs das Verhalten des Systems vor, wenn eine solche Ausnahmebedingung ausgelöst wird.

Wenn ein Datenhandler oder eine Datenbindung durch eine Bindung aufgerufen wird, können verschiedene Probleme auftreten. Beispielsweise kann es sein, dass ein Datenhandler eine Nachricht mit beschädigten Nutzdaten empfängt oder er versucht, eine Nachricht mit einem falschen Format zu lesen.

Das Verfahren, mit dem die Bindung eine solche Ausnahmebedingung behandelt, wird dadurch bestimmt, wie Sie den Datenhandler oder die Datenbindung implementieren. Das empfohlene Verhalten besteht darin, die Datenbindung so zu konfigurieren, dass sie eine Ausnahmebedingung des Typs **DataBindingException** auslöst.

Bei einem Datenhandler ist die Situation ähnlich. Da der Datenhandler durch die Datenbindung aufgerufen wird, wird jede Ausnahmebedingung des Datenhandlers in eine Ausnahmebedingung der Datenbindung eingeschlossen. Daher wird eine Ausnahmebindung des Typs **DataHandlerException** als Typ **DataBindingException** gemeldet.

Wenn eine Laufzeitausnahmebedingung (inklusive **DataBindingException**) ausgelöst wird, geschieht Folgendes:

- Falls der Mediationsablauf transaktionsorientiert konfiguriert ist, wird die JMS-Nachricht standardmäßig in Failed Event Manager gespeichert, damit sie manuell wiedergegeben oder gelöscht werden kann.

  Anmerkung: Sie können den Fehlerbehebungsmodus für die Bindung so ändern, dass die Nachricht zurückgesetzt und nicht in failed event manager gespeichert wird.
- Ist der Mediationsablauf nicht transaktionsorientiert, wird die Ausnahmebedingung protokolliert und die Nachricht ist nicht mehr vorhanden.

Bei einem Datenhandler ist die Situation ähnlich. Da der Datenhandler durch die Datenbindung aufgerufen wird, wird jede Ausnahmebedingung des Datenhandlers in eine Ausnahmebedingung der Datenbindung eingeschlossen. Daher wird eine Ausnahmebindung des Typs **DataHandlerException** als Typ **DataBindingException** gemeldet.

WebSphere MQ-JMS-Bindungen

Die WebSphere MQ-JMS-Bindung ermöglicht die Integration bei externen Anwendungen, die einen JMS-basierten WebSphere MQ-Provider verwenden.

Mit den WebSphere MQ-JMS-Exportbindungen und -Importbindungen können Sie eine direkte Integration bei externen JMS- oder MQ-JMS-Systemen aus Ihrer Serverumgebung heraus erreichen. Die Verwendung der Funktionen für die MQ- oder Clientverbindung von Service Integration Bus ist dann nicht mehr erforderlich.

Wenn eine Komponente mit einem JMS-basierten WebSphere MQ-Service über einen Import interagiert, verwendet die WebSphere MQ-JMS-Importbindung ein Ziel, an das Daten gesendet werden, und ein Ziel, an dem die Antwort empfangen werden kann. Die Konvertierung der Daten in eine und aus einer JMS-Nachricht wird durch die Ersterkennungskomponente des JMS-Datenhandlers oder der JMS-Datenbindung ausgeführt.

Wenn ein SCA-Modul einen Service für WebSphere MQ-JMS-Clients bereitstellt, verwendet die WebSphere MQ-JMS-Exportbindung ein Ziel, an dem die Anforderung empfangen und an das die Antwort gesendet werden kann. Die Konvertierung der Daten in eine und aus einer JMS-Nachricht wird durch den JMS-Datenhandler oder die JMS-Datenbindung vorgenommen.

Der Funktionsselektor stellt eine Zuordnung zu der Operation in der aufzurufenden Zielkomponente bereit.

WebSphere MQ-JMS-Bindungen - Übersicht:

Die WebSphere MQ-JMS-Bindung ermöglicht die Integration bei externen Anwendungen, die den WebSphere MQ-JMS-Provider verwenden.

WebSphere MQ-Verwaltungstasks

Der WebSphere MQ-Systemadministrator muss den zugrunde liegenden WebSphere MQ Queue Manager, den die WebSphere MQ-JMS-Bindungen verwenden, erstellen, bevor eine Anwendung ausgeführt wird, die diese Bindungen enthält.

WebSphere MQ-JMS-Importbindungen

Durch den WebSphere MQ-JMS-Import können Komponenten in Ihrem SCA-Modul mit Services kommunizieren, die von JMS-basierten WebSphere MQ-Providern bereitgestellt werden. Sie müssen eine unterstützte Version von WebSphere MQ verwenden. Ausführliche Angaben über die Hardware- und Softwarevoraussetzungen finden Sie auf den Seiten von IBM Support.

Es werden zwei Typen von Einsatzszenarios für WebSphere MQ-JMS-Importbindungen unterstützt, die sich nach dem Typ der aufgerufenen Operation richten:

- Unidirektional: Der WebSphere MQ-JMS-Import übergibt eine Nachricht an das Sendeziel, das in der Importbindung konfiguriert ist. An das Feld **replyTo** des JMS-Headers werden keine Daten gesendet.
- Bidirektional (Anforderung-Antwort): Der WebSphere MQ-JMS-Import übergibt eine Nachricht an das Sendeziel.

  Das Empfangsziel (**receive**) ist im Headerfeld **replyTo** festgelegt. Für die Überwachung des Empfangsziels wird eine nachrichtengesteuerte Bean implementiert. Sobald eine Antwort empfangen wird, übergibt die nachrichtengesteuerte Bean die Antwort zurück an die Komponente.

  Die Importbindung kann (mit dem Feld **Korrelationsschema für Antwort** in Integration Designer) so konfiguriert werden, dass das Kopieren der Korrelations-ID für die Antwortnachricht aus der Anforderungsnachrichten-ID (Standardeinstellung) oder aus der Korrelations-ID für die Anforderungsnachricht erwartet wird.

Sowohl bei unidirektionalen als auch bei bidirektionalen Einsatzszenarios können dynamische und statische Headereigenschaften angegeben werden. Statische Eigenschaften können über die Methodenbindung des JMS-Imports festgelegt werden. Manche dieser Eigenschaften haben in der SCA-JMS-Laufzeit eine besondere Bedeutung.

Es muss unbedingt beachtet werden, dass es sich bei WebSphere MQ-JMS um eine asynchrone Bindung handelt. Falls eine aufrufende Komponente einen WebSphere-JMS-Import (bei einer bidirektionalen Operation) im Synchronmodus aufruft, wird die aufrufende Komponente blockiert, bis die Antwort durch den JMS-Service zurückgegeben wurde.

In Abb. 37 auf Seite 140 ist dargestellt, wie der Import mit dem externen Service verknüpft ist.

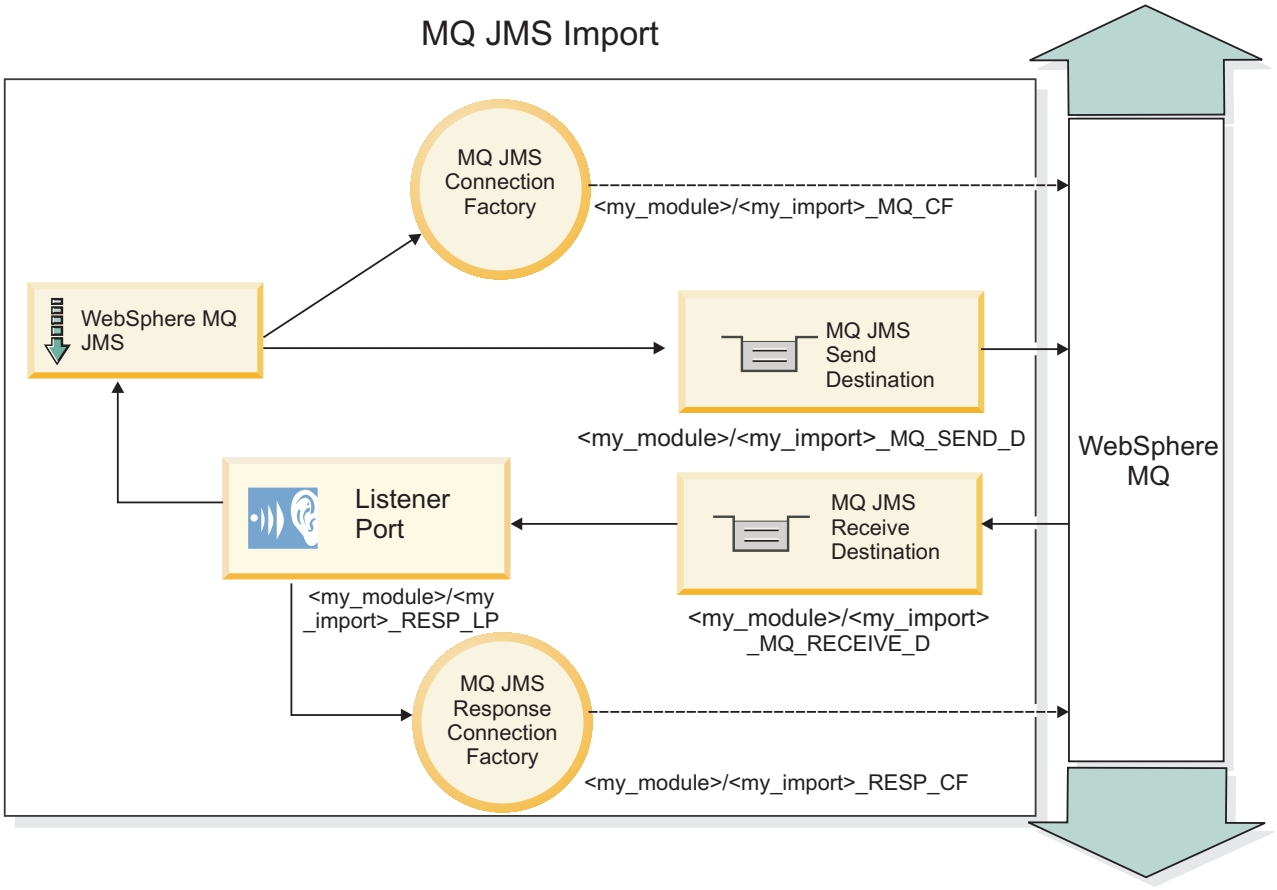


Abbildung 82. Ressourcen der WebSphere MQ-JMS-Importbindung

WebSphere MQ-JMS-Exportbindungen

Die WebSphere MQ-JMS-Exportbindung stellt für SCA-Module ein Verfahren bereit, mit dem Services für externe Anwendungen beim WebSphere MQ-basierten JMS-Provider angeboten werden können.

Zur Überwachung von Anforderungen, die an dem in der Exportbindung angegebenen Empfangsziel eingehen, wird eine nachrichtengesteuerte Bean implementiert. Das im Feld **send** angegebene Ziel wird verwendet, um die Antwort auf die eingehende Anforderung zu senden, falls die aufgerufene Komponente eine Antwort bereitstellt. Das im Feld **replyTo** der Antwortnachricht angegebene Ziel überschreibt das im Feld **send** angegebene Ziel.

In Abb. 38 auf Seite 141 ist dargestellt, wie der externe Anforderer mit dem Export verknüpft ist.

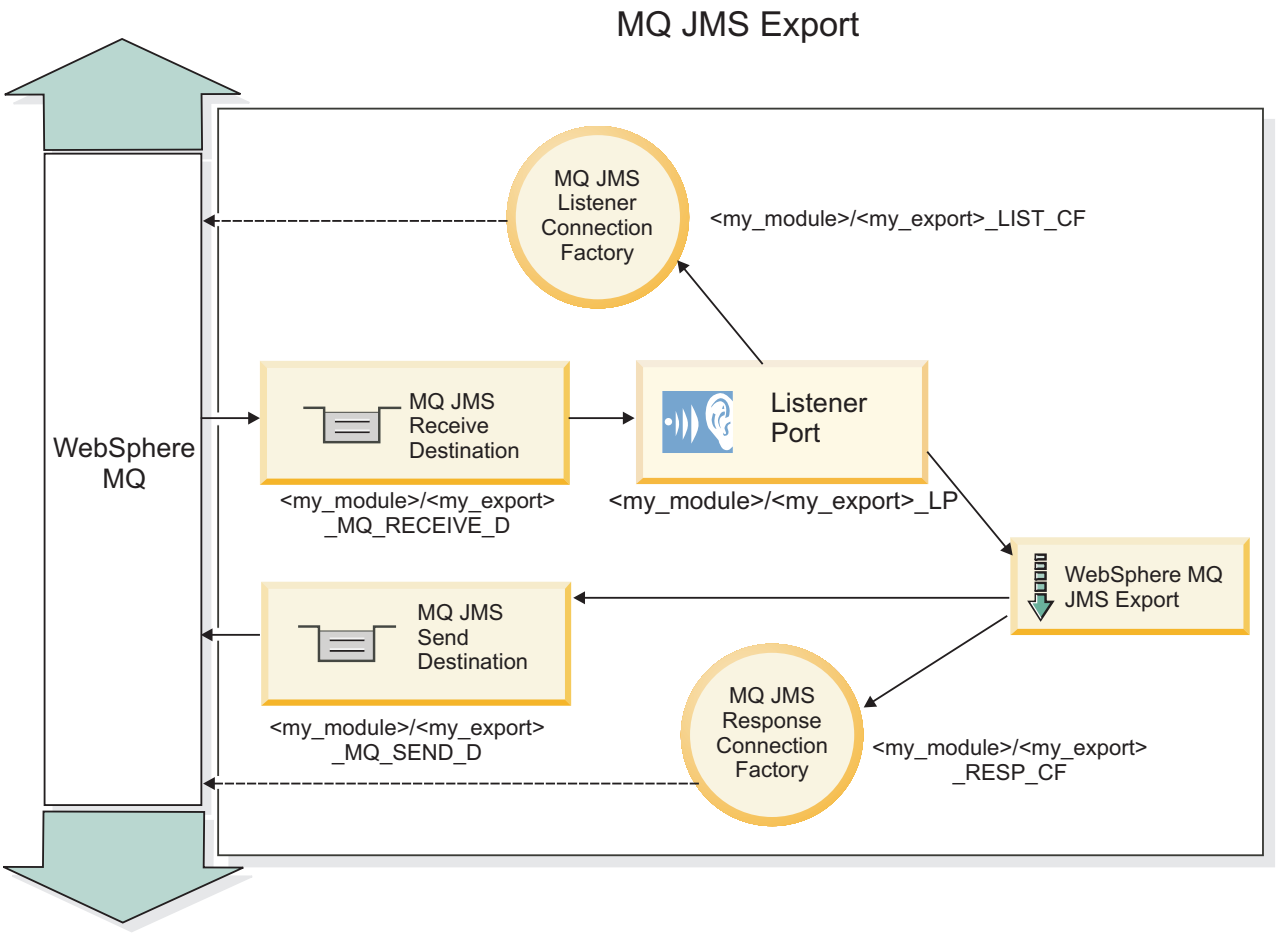


Abbildung 83. Ressourcen der WebSphere MQ-JMS-Exportbindung

Anmerkung: Abb. 37 auf Seite 140 und Abb. 38 auf Seite 141 veranschaulichen, wie eine Anwendung aus einer früheren Version von IBM Business Process Manager mit einem externen Service verbunden wird. Bei Anwendungen, die für IBM Business Process Manager Version 7.0 entwickelt werden, wird anstelle des Listener-Ports und der Verbindungsfactory die Aktivierungsspezifikation verwendet.

Schlüsselfunktionen von WebSphere MQ-JMS-Bindings:

Die Schlüsselfunktionen von WebSphere MQ JMS-Bindings umfassen Header, Java EE-Artefakte und erstellte Java EE-Ressourcen.

Header

Ein JMS-Nachrichtenheader enthält eine Reihe vordefinierter Felder, in denen Werte enthalten sind, die sowohl von den Clients als auch den Providern zum Angeben und Weiterleiten von Nachrichten verwendet werden. Sie können diese Header anhand der Bindingeigenschaften mit festen Werten konfigurieren, die Header können aber auch dynamisch zur Laufzeit angegeben werden.

`JMSCorrelationID`
Stellt eine Verbindung zur zugehörigen Nachricht her. In der Regel wird in diesem Feld die Zeichenfolge der Nachrichten-ID für die Nachricht eingestellt, auf die geantwortet wird.

`TargetFunctionName`
Dieser Header wird von einem der angegebenen Funktionsselektoren zum Angeben der Operation verwendet, die aufgerufen wird. Wenn die JMS-Headereigenschaft 'TargetFunctionName' in Nachrichten eingestellt wird, die an einen JMS-Export gesendet werden, kann dieser Funktionsselektor ver-

wendet werden. Die Eigenschaft kann direkt in den JMS-Clientanwendungen eingestellt werden oder wenn eine Verbindung zwischen einem Import mit einem JMS-Binding und einem solchen Export hergestellt wird. In diesem Fall muss das JMS-Importbinding so konfiguriert sein, dass der Header 'TargetFunctionName' für jede Operation in der Schnittstelle als Name der Operation festgelegt wird.

Korrelationsschemas

Von den WebSphere MQ-JMS-Bindings werden verschiedene Korrelationsschemas bereitgestellt, die dazu verwendet werden, festzustellen, wie Anforderungsnachrichten und Antwortnachrichten miteinander korrelieren.

RequestMsgIDToCorrelID
Die JMSMessageID wird in das Feld JMSCorrelationID kopiert. Dies ist die Standardeinstellung.

RequestCorrelIDToCorrelID
Die JMSCorrelationID wird in das Feld JMSCorrelationID kopiert.

Java EE-Ressourcen

Wenn ein MQ-JMS-Import in einer Java EE-Umgebung implementiert wird, werden Java EE-Ressourcen erstellt.

Parameter

MQ-Verbindungsfactory
Wird von Clients zum Erstellen einer Verbindung zum MQ-JMS-Provider verwendet.

Antwortverbindungsfactory
Wird von der SCA-MQ-JMS-Laufzeit verwendet, wenn sich das Sendeziel in einem anderen Queue Manager als das Empfangsziel befindet.

Aktivierungsspezifikation
Eine MQ-JMS-Aktivierungsspezifikation ist mindestens einer Message-driven Bean zugeordnet und stellt die Konfiguration bereit, die zum Empfangen von Nachrichten erforderlich ist.

Ziele

- Sendeziel:
  - Importe: Das Ziel, an das die Anforderung oder abgehende Nachricht gesendet wird.
  - Exporte: Das Ziel, an das die Antwortnachricht gesendet wird, falls diese nicht durch das Headerfeld JMSReplyTo der eingehenden Nachricht außer Kraft gesetzt wird.
- Empfangsziel:
  - Importe: Das Ziel, an das die Antwort oder eingehende Nachricht gesendet werden soll.
  - Exporte: Das Ziel, an das die eingehende oder Anforderungsnachricht gesendet werden soll.

JMS-Header:

Eine JMS-Nachricht enthält zwei Typen von Headern, nämlich den JMS-Systemheader und mehrere JMS-Eigenschaften. Auf beide Headertypen kann entweder in einem Mediationsmodul im Servicenachrichtenobjekt (Service Message Object - SMO) oder unter Verwendung der API 'ContextService' zugegriffen werden.

JMS-Systemheader

Der JMS-Systemheader wird im Servicenachrichtenobjekt durch das Element 'JMSHeader' repräsentiert, das alle normalerweise in einem JMS-Header zu findenden Felder enthält. Obwohl diese Felder in der Mediation (oder der API 'ContextService') geändert werden können, werden einige der im Servicenachrichtenobjekt festgelegten Felder für den JMS-Systemheader nicht an die abgehende JMS-Nachricht weitergegeben, weil sie durch Systemwerte oder statische Werte überschrieben werden.

Die folgenden Schlüsselfelder im JMS-Systemheader können in einer Mediation (oder einer API 'Context-Service') aktualisiert werden:

- **JMSType** und **JMSCorrelationID** - Werte der spezifischen vordefinierten Nachrichtenheadereigenschaften.
- **JMSDeliveryMode**: Werte für den Übermittlungsmodus ('persistent' oder 'nonpersistent'; Standardwert ist 'persistent').
- **JMSPriority**: Prioritätswert (0 bis 9; Standardwert ist 'JMS_Default_Priority').

JMS-Eigenschaften

JMS-Eigenschaften werden im Servicenachrichtenobjekt als Einträge in der Eigenschaftsliste ('Properties') dargestellt. Die Eigenschaften können in einer Mediation oder durch die Verwendung der API 'Context-Service' hinzugefügt, aktualisiert oder gelöscht werden.

Eigenschaften können auch in der JMS-Bindung statisch festgelegt sein. Statisch festgelegte Eigenschaften überschreiben Einstellungen desselben Namens, die dynamisch festgelegt werden.

Benutzereigenschaften, die von anderen Bindungen (z. B. einer HTTP-Bindung) weitergegeben werden, werden in der JMS-Bindung als JMS-Eigenschaften ausgegeben.

Einstellungen für Headerweitergabe

Die Weitergabe des JMS-Systemheaders und der JMS-Eigenschaften entweder von der eingehenden JMS-Nachricht an nachgelagerte Komponenten oder von vorgelagerten Komponenten an die abgehende JMS-Nachricht kann durch das Attribut 'Protokollheader weitergeben' der Bindung gesteuert werden.

Wenn das Attribut 'Protokollheader weitergeben' festgelegt ist, ist die Übermittlung von Headerinformationen an die Nachricht oder die Zielkomponente, wie in der folgenden Liste beschrieben, zulässig:

- JMS-Exportanforderung

  Der in der Nachricht empfangene JMS-Header wird mittels des Kontextservice an Zielkomponenten weitergegeben. In der Nachricht empfangene JMS-Eigenschaften werden mittels des Kontextservice an Zielkomponenten weitergegeben.
- JMS-Exportantwort

  Alle im Kontextservice definierten JMS-Headerfelder werden in der abgehenden Nachricht verwendet, sofern sie nicht durch statische Eigenschaften überschrieben werden, die in der JMS-Exportbindung festgelegt sind. Alle im Kontextservice definierten Eigenschaften werden in der abgehenden Nachricht verwendet, sofern sie nicht durch statische Eigenschaften überschrieben werden, die in der JMS-Exportbindung festgelegt sind.
- JMS-Importanforderung

  Alle im Kontextservice definierten JMS-Headerfelder werden in der abgehenden Nachricht verwendet, sofern sie nicht durch statische Eigenschaften überschrieben werden, die in der JMS-Importbindung festgelegt sind. Alle im Kontextservice definierten Eigenschaften werden in der abgehenden Nachricht verwendet, sofern sie nicht durch statische Eigenschaften überschrieben werden, die in der JMS-Importbindung festgelegt sind.
- JMS-Importantwort

  Der in der Nachricht empfangene JMS-Header wird mittels des Kontextservice an Zielkomponenten weitergegeben. In der Nachricht empfangene JMS-Eigenschaften werden mittels des Kontextservice an Zielkomponenten weitergegeben.

Externe Clients:

Der Server kann mit WebSphere MQ-JMS-Bindungen Nachrichten an externe Clients senden bzw. von diesen empfangen.

Ein externer Client (z. B. ein Webportal oder ein unternehmensweites Informationssystem) kann eine Nachricht an eine SCA-Komponente in der Anwendung durch einen Export senden oder von einer SCA-Komponente in der Anwendung durch einen Import aufgerufen werden.

Die WebSphere MQ-JMS-Exportbindung implementiert nachrichtengesteuerte Beans (Message driven bean - MDB), um Anforderungen zu überwachen, die bei dem in der Exportbindung definierten Empfangsziel eingehen. Das im Feld **send** angegebene Ziel wird verwendet, um die Antwort auf die eingehende Anforderung zu senden, falls die aufgerufene Anwendung eine Antwort bereitstellt. Ein externer Client kann somit Anwendungen über die Exportbindung aufrufen.

WebSphere MQ-JMS-Importe stellen eine Bindung zu externen Clients her und können Nachrichten an diese Clients übermitteln. Diese Nachrichten machen möglicherweise, jedoch nicht zwangsläufig, eine Antwort an den externen Client erforderlich.

Weitere Informationen zur Interaktion mit externen Clients unter Verwendung von WebSphere MQ finden Sie im Information Center von WebSphere MQ.

Fehlerbehebung für WebSphere MQ-JMS-Bindungen:

Sie können Probleme im Zusammenhang mit WebSphere MQ-JMS-Bindungen diagnostizieren und beheben.

Ausnahmebedingungen bei der Implementierung

Als Reaktion auf diverse Fehlerbedingungen kann die MQ-JMS-Implementierung für Importe und Exporte zwei Typen von Ausnahmebedingungen zurückgeben:

- Geschäftsausnahmebedingung für Service: Diese Ausnahmebedingung wird zurückgegeben, wenn der für die Geschäftsschnittstelle für den Service (Typ WSDL-Port) definierte Fehler aufgetreten ist.
- Laufzeitausnahmebedingung für den Service: Diese Ausnahmebedingung wird in allen übrigen Fällen ausgelöst. In den meisten Fällen enthält die Ausnahmebedingung vom Typ cause (Ursache) die ursprüngliche Ausnahmebedingung (JMSException).

  Ein Import erwartet zum Beispiel nur eine einzige Antwortnachricht für jede Anforderungsnachricht. Wenn mehr als eine Antwort eingeht oder wenn eine verspätete Antwort eingeht (d. h. eine Antwort, für die die definierte SCA-Antwortablauffrist verstrichen ist), so wird eine Laufzeitausnahmebedingung für den Service ausgelöst. Für die Transaktion wird ein Rollback ausgeführt und die Antwortnachricht wird aus der Warteschlange entfernt oder vom Failed Event Manager verarbeitet.

WebSphere MQ-JMS-basierte SCA-Nachrichten werden im Failed Event Manager nicht angezeigt

Wenn die SCA-Nachrichten ihren Ursprung in einem Fehler oder einer Störung in der WebSphere MQ-JMS-Interaktion haben, würden Sie erwarten, diese Nachrichten im Failed Event Manager vorzufinden. Werden keine derartigen Nachrichten im Failed Event Manager angezeigt, stellen Sie sicher, dass für den zugrunde liegenden Listener-Port ein Wert größer als **1** für die Eigenschaft der maximalen Anzahl von Wiederholungsversuchen definiert ist. Durch Festlegen von **2** oder höher für diesen Wert wird eine Interaktion mit dem Failed Event Manager während SCA-Aufrufen für die MQ-JMS-Bindungen ermöglicht.

Szenario falscher Verwendungen: Vergleich mit WebSphere MQ-Bindungen

Die WebSphere MQ-JMS-Bindung wurde dazu konzipiert, mit JMS-Anwendungen zu interagieren, die auf WebSphere MQ implementiert wurden, was eine Offenlegung von Nachrichten gemäß dem JMS-Nachrichtenmodell bewirkt. Import und Export von WebSphere MQ wurden jedoch in erster Linie entwickelt, um mit nativen WebSphere MQ-Anwendungen zu interagieren und den gesamten Inhalt des WebSphere MQ-Nachrichtenhauptteils gegenüber Mediationen verfügbar zu machen.

Die folgenden Szenarien sollten unter Verwendung der WebSphere MQ-JMS-Bindung erstellt werden und nicht mit der WebSphere MQ-Bindung:

- Aufrufen einer JMS-Message-driven Bean (MDB) von einem SCA-Modul, wenn die MDB auf dem WebSphere MQ-JMS-Provider implementiert ist. Verwenden Sie in diesem Fall einen WebSphere MQ-JMS-Import.
- Zulassen des Aufrufs des SCA-Moduls von einem Java EE-Komponentenservlet oder einer EJB anhand eines JMS. Verwenden Sie in diesem Fall einen WebSphere MQ-JMS-Export.
- Mediation des Inhalts einer JMS-Zuordnungsnachricht beim Durchqueren von WebSphere MQ betreiben. Verwenden Sie einen WebSphere MQ-JMS-Export und -Import in Verbindung mit dem entsprechenden Datenhandler oder der entsprechenden Datenbindung.

In gibt Fälle, in denen eine Interaktion zwischen WebSphere MQ-Bindung und WebSphere MQ-JMS-Bindung gegebenenfalls erwartet wird. Besonders bei Überbrückungen zwischen Java EE- und Nicht-Java-EE-Anwendungen von WebSphere MQ sollten Sie einen WebSphere MQ-Export und einen WebSphere MQ-JMS-Import (oder umgekehrt) in Verbindung mit den geeigneten Datenbindungen oder Mediationsmodulen (oder beiden) verwenden.

Ausnahmebedingungen verarbeiten:

Die Art und Weise, in der die Bindung konfiguriert ist, bestimmt, wie Ausnahmebedingungen verarbeitet werden, die von Datenhandlern oder Datenbindungen ausgelöst werden. Außerdem gibt die Spezifik des Mediationsablaufs das Verhalten des Systems vor, wenn eine solche Ausnahmebedingung ausgelöst wird.

Wenn ein Datenhandler oder eine Datenbindung durch eine Bindung aufgerufen wird, können verschiedene Probleme auftreten. Beispielsweise kann es sein, dass ein Datenhandler eine Nachricht mit beschädigten Nutzdaten empfängt oder er versucht, eine Nachricht mit einem falschen Format zu lesen.

Das Verfahren, mit dem die Bindung eine solche Ausnahmebedingung behandelt, wird dadurch bestimmt, wie Sie den Datenhandler oder die Datenbindung implementieren. Das empfohlene Verhalten besteht darin, die Datenbindung so zu konfigurieren, dass sie eine Ausnahmebedingung des Typs **DataBindingException** auslöst.

Bei einem Datenhandler ist die Situation ähnlich. Da der Datenhandler durch die Datenbindung aufgerufen wird, wird jede Ausnahmebedingung des Datenhandlers in eine Ausnahmebedingung der Datenbindung eingeschlossen. Daher wird eine Ausnahmebindung des Typs **DataHandlerException** als Typ **DataBindingException** gemeldet.

Wenn eine Laufzeitausnahmebedingung (inklusive **DataBindingException**) ausgelöst wird, geschieht Folgendes:

- Falls der Mediationsablauf transaktionsorientiert konfiguriert ist, wird die JMS-Nachricht standardmäßig in Failed Event Manager gespeichert, damit sie manuell wiedergegeben oder gelöscht werden kann.

  Anmerkung: Sie können den Fehlerbehebungsmodus für die Bindung so ändern, dass die Nachricht zurückgesetzt und nicht in failed event manager gespeichert wird.
- Ist der Mediationsablauf nicht transaktionsorientiert, wird die Ausnahmebedingung protokolliert und die Nachricht ist nicht mehr vorhanden.

Bei einem Datenhandler ist die Situation ähnlich. Da der Datenhandler durch die Datenbindung aufgerufen wird, wird jede Ausnahmebedingung des Datenhandlers in eine Ausnahmebedingung der Datenbindung eingeschlossen. Daher wird eine Ausnahmebindung des Typs **DataHandlerException** als Typ **DataBindingException** gemeldet.

WebSphere MQ-Bindungen

Die WebSphere MQ-Bindung ist so konzipiert, dass sie SCA-Konnektivität mit WebSphere MQ-Anwendungen bereitstellt.

Mit den WebSphere MQ-Exportbindungen und -Importbindungen können Sie eine direkte Integration bei einem WebSphere MQ-basierten System aus Ihrer Serverumgebung heraus erreichen. Die Verwendung der Funktionen für die MQ- oder Clientverbindung von Service Integration Bus ist dann nicht mehr erforderlich.

Wenn eine Komponente mit einem WebSphere MQ-Service über einen Import interagiert, verwendet die WebSphere MQ-Importbindung eine Warteschlange, an die Daten gesendet werden, und eine Warteschlange, in der die Antwort empfangen werden kann.

Wenn ein SCA-Modul einen Service für WebSphere MQ-Clients bereitstellt, verwendet die WebSphere MQ-Exportbindung eine Warteschlange, in der Anforderung empfangen und an die die Antwort gesendet werden kann. Der Funktionsselektor stellt eine Zuordnung zu der Operation in der aufzurufenden Zielkomponente bereit.

Die Konvertierung der Nutzdaten in eine und aus einer MQ-Nachricht wird durch den Datenhandler oder die Datenbindung des MQ-Hauptteils vorgenommen. Die Konvertierung der Headerdaten in eine und aus einer MQ-Nachricht wird die durch Datenbindung des MQ-Headers vorgenommen.

Informationen zu den unterstützten WebSphere MQ-Versionen finden Sie auf der Webseite detailed system requirements.

WebSphere MQ-Bindungen - Übersicht:

Die WebSphere MQ-Bindung ermöglicht die Integration bei nativen MQ-basierten Anwendungen.

WebSphere MQ-Verwaltungstasks

Der WebSphere MQ-Systemadministrator muss den zugrunde liegenden WebSphere MQ Queue Manager, den die WebSphere MQ-Bindungen verwenden, erstellen, bevor eine Anwendung ausgeführt wird, die diese Bindungen enthält.

WebSphere-Verwaltungstasks

Sie müssen die Eigenschaft für den **Pfad der nativen Bibliothek** des MQ-Ressourcenadapters in WebSphere mit der vom Server unterstützten WebSphere MQ-Version festlegen und den Server erneut starten. Dies stellt sicher, dass die Bibliotheken einer unterstützten Version von WebSphere MQ verwendet werden. Detaillierte Informationen zu Hardware- und Softwarevoraussetzung finden Sie auf den IBM-Unterstützungsseiten.

WebSphere MQ-Importbindungen

Durch die WebSphere MQ-Importbindung können Komponenten in Ihrem SCA-Modul mit Services kommunizieren, die von externen WebSphere MQ-basierten Anwendungen bereitgestellt werden. Sie müssen eine unterstützte Version von WebSphere MQ verwenden. Detaillierte Informationen zu Hardware- und Softwarevoraussetzung finden Sie auf den IBM-Unterstützungsseiten.

Die Interaktion mit externen WebSphere-Systemen umfasst auch die Verwendung von Warteschlangen für das Senden von Anforderungen und das Empfangen von Antworten.

Es werden zwei Typen von Einsatzszenarios für die WebSphere MQ-Importbindungen unterstützt, die sich nach dem Typ der aufgerufenen Operation richten:

- Unidirektional: Der WebSphere MQ-Import stellt eine Nachricht in die Warteschlange, die im Feld für die **Sendezielwarteschlange** der Importbindung konfiguriert ist. An das Feld **replyTo** des MQMD-Headers werden keine Daten gesendet.
- Bidirektional: Der WebSphere MQ-Import stellt eine Nachricht in die Warteschlange, die im Feld für die **Sendezielwarteschlange** konfiguriert ist.

Die Empfangswarteschlange (**receive**) ist im MQMD-Headerfeld **replyTo** festgelegt. Für die Überwachung der Empfangswarteschlange wird eine nachrichtengesteuerte Bean implementiert. Sobald eine Antwort empfangen wird, übergibt die nachrichtengesteuerte Bean die Antwort zurück an die Komponente.

Die Importbindung kann (mit dem Feld **Korrelationsschema für Antwort**) so konfiguriert werden, dass das Kopieren der Korrelations-ID für die Antwortnachricht aus der Anforderungsnachrichten-ID (Standardeinstellung) oder aus der Korrelations-ID für die Anforderungsnachricht erwartet wird.

Es muss unbedingt beachtet werden, dass es sich bei WebSphere MQ um eine asynchrone Bindung handelt. Falls eine aufrufende Komponente einen WebSphere MQ-Import (bei einer bidirektionalen Operation) im Synchronmodus aufruft, wird die aufrufende Komponente blockiert, bis die Antwort durch den WebSphere MQ-Service zurückgegeben wurde.

In Abb. 39 auf Seite 147 ist dargestellt, wie der Import mit dem externen Service verknüpft ist.

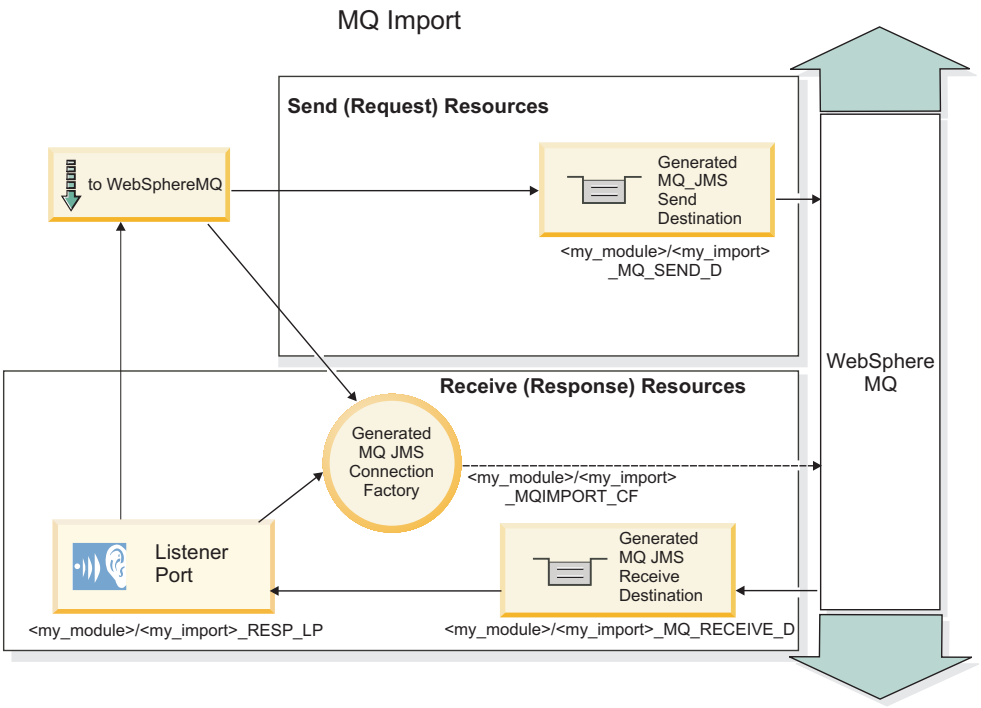


Abbildung 84. Ressourcen der WebSphere MQ-Importbindung

WebSphere MQ-Exportbindungen

Die WebSphere MQ-Exportbindung stellt SCA-Modulen ein Verfahren bereit, mit dem Services für externe WebSphere MQ-basierte Anwendungen angeboten werden können.

Zur Überwachung von Anforderungen, die in der in der Exportbindung angegebenen **Empfangszielwartschlange** eingehen, wird eine nachrichtengesteuerte Bean implementiert. Die im Feld für die **Sendezielwarteschlange** angegebene Warteschlange wird verwendet, um die Antwort auf die eingehende Anforderung zu senden, falls die aufgerufene Komponente eine Antwort bereitstellt. Die im Feld **replyTo** der Antwortnachricht angegebene Warteschlange überschreibt die im Feld für die **Sendezielwarteschlange** angegebene Warteschlange.

In Abb. 40 auf Seite 148 ist dargestellt, wie der externe Anforderer mit dem Export verknüpft ist.

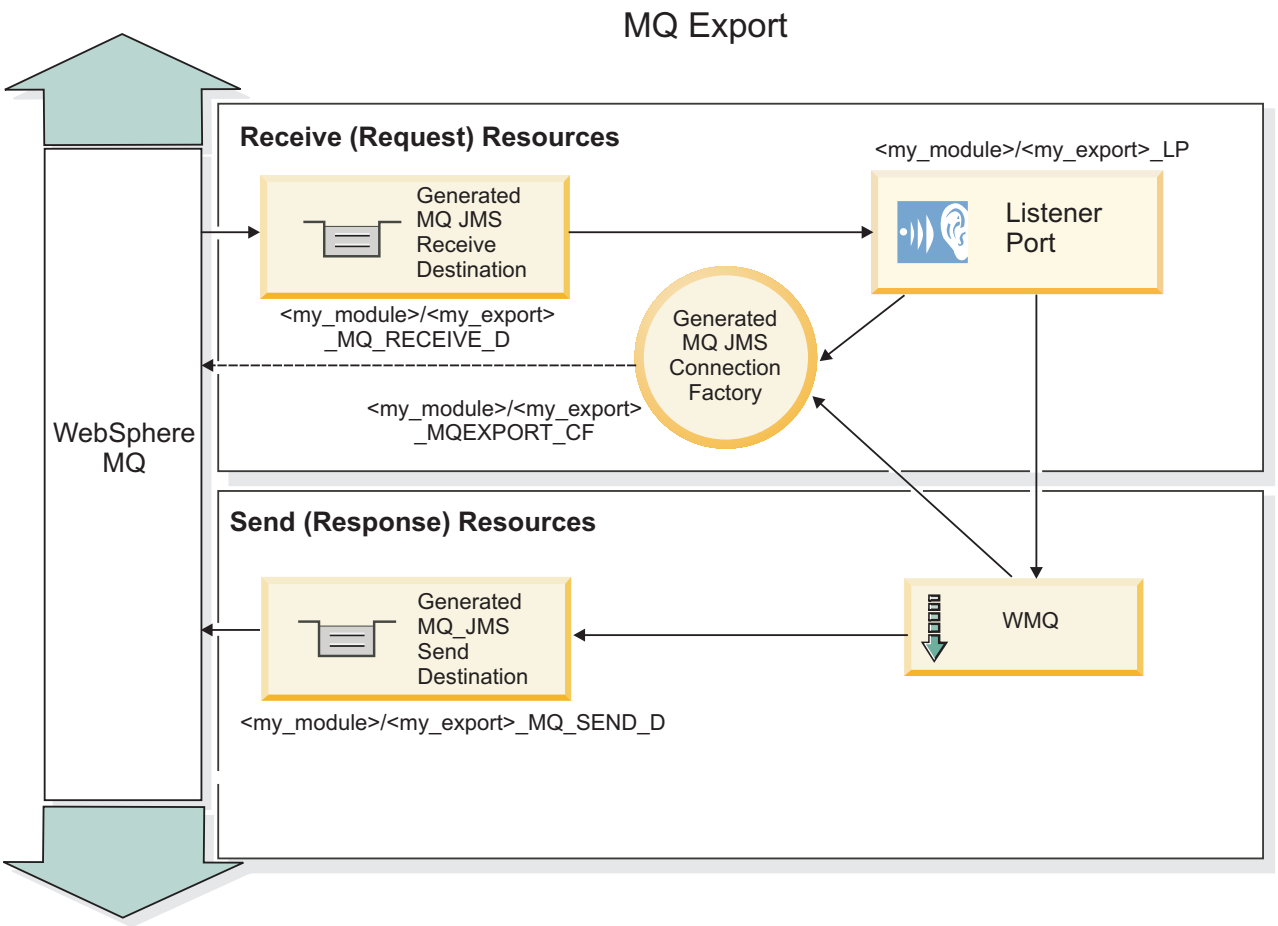


Abbildung 85. Ressourcen der WebSphere MQ-Exportbindung

Anmerkung: Abb. 39 auf Seite 147 und Abb. 40 auf Seite 148 veranschaulichen, wie eine Anwendung aus einer früheren Version von IBM Business Process Manager mit einem externen Service verbunden wird. Bei Anwendungen, die für IBM Business Process Manager Version 7.x oder aktuellere Versionen entwickelt werden, wird anstelle des Listener-Ports und der Verbindungsfactory die Aktivierungsspezifikation verwendet.

Schlüsselfunktionen von WebSphere MQ-Bindings:

Die Schlüsselfunktionen von WebSphere MQ-Bindings umfassen Header, Java EE-Artefakte und erstellte Java EE-Ressourcen.

Korrelationsschemas

Von einer WebSphere MQ-Anforderungs-/Antwortanwendung kann ein Verfahren aus einer Reihe verschiedener Verfahren zum Korrelieren von Antwortnachrichten mit Anforderungen verwendet werden; für die Erstellung werden die MQMD-Felder MessageID und CorrelID verwendet. Im Großteil der Fälle lässt der Anforderer den Warteschlangenmanager eine MessageID auswählen und erwartet, dass die antwortende Anwendung diese in die CorrelID der Antwort kopiert. In den meisten Fällen wissen der Anforderer und die antwortende Anwendung implizit, welches Korrelationsverfahren verwendet wird. In manchen Fällen berücksichtigt die antwortende Anwendung verschiedene Markierungen im Feld Report der Anforderung, aus denen hervorgeht, wie mit diesen Feldern umgegangen werden soll.

Exportbindings für WebSphere MQ-Nachrichten können mit den folgenden Optionen konfiguriert werden:

Optionen für Nachrichten-ID der Antwort:

Neue Nachrichten-ID
Bei Auswahl dieser Option kann der Warteschlangenmanager eine eindeutige MsgId für die Antwort auswählen (Standardeinstellung).

Aus Nachrichten-ID der Anforderung kopieren
Kopiert die Angabe im Feld MsgId aus dem Feld MsgId in die Anforderung.

Aus SCA-Nachricht kopieren
Legt fest, dass die MsgId in den WebSphere MQ-Headern in die SCA-Antwortnachricht übertragen werden soll oder lässt den Warteschlangenmanager eine neue ID definieren, wenn kein Wert vorhanden ist.

Wie bei Berichtsoptionen
Überprüft das Feld Report von MQMD in der Anforderung auf einen Hinweis zum Umgang mit der MsgId. Die Optionen MQRO_NEW_MSG_ID und MQRO_PASS_MSG_ID werden unterstützt, ihr Verhalten entspricht New MsgId bzw. Copy from Request MsgID.

Optionen für Korrelations-ID der Antwort:

Aus Nachrichten-ID der Anforderung kopieren
Kopiert die Angabe im Feld CorrelId aus dem Feld MsgId in die Anforderung (Standardeinstellung).

Korrelations-ID-Kopie aus Anforderungskorrelations-ID
Kopiert die Angabe im Feld CorrelId aus dem Feld CorrelId in die Anforderung.

Aus SCA-Nachricht kopieren
Legt fest, dass die CorrelId in den WebSphere MQ-Headern in die SCA-Antwortnachricht übertragen werden soll oder lässt das Feld leer, wenn kein Wert vorhanden ist.

Wie bei Berichtsoptionen
Überprüft das Feld Report von MQMD in der Anforderung auf einen Hinweis zum Umgang mit der CorrelId. Die Optionen MQRO_COPY_MSG_ID_TO_CORREL_ID und MQRO_PASS_CORREL_ID werden unterstützt, ihr Verhalten entspricht dem von Copy from Request MsgID bzw. Copy from Request CorrelID.

Importbindings für WebSphere MQ-Nachrichten können mit den folgenden Optionen konfiguriert werden:

Optionen für Nachrichten-ID der Anforderung:

Neue Nachrichten-ID
Bei Auswahl dieser Option kann der Warteschlangenmanager eine eindeutige MsgId für die Anforderung auswählen (Standardeinstellung).

Aus SCA-Nachricht kopieren
Legt fest, dass die MsgId in den WebSphere MQ-Headern in die SCA-Anforderungsnachricht übertragen werden soll oder lässt den Warteschlangenmanager eine neue ID definieren, wenn kein Wert vorhanden ist.

Optionen für Korrelations-ID der Antwort:

Response has CorrelID copied from MsgId
Erwartet, dass in der Antwortnachricht ein Feld mit der Bezeichnung CorrelId eingestellt ist und dafür die MsgId der Anforderung verwendet wird (Standardeinstellung).

Response has MsgID copied from MsgId
Erwartet, dass in der Antwortnachricht ein Feld mit der Bezeichnung MsgId eingestellt ist und dafür die MsgId der Anforderung verwendet wird.

Response has CorrelID copied from CorrelId
Erwartet, dass in der Antwortnachricht ein Feld mit der Bezeichnung CorrelId eingestellt ist und dafür die CorrelId der Anforderung verwendet wird.

Java EE-Ressourcen

Wenn ein WebSphere MQ-Binding in einer Java EE-Umgebung implementiert wird, werden Java EE-Ressourcen erstellt.

Parameter

MQ-Verbindungsfactory
Wird von Clients zum Erstellen einer Verbindung zum WebSphere MQ-Provider verwendet.

Antwortverbindungsfactory
Wird von der SCA-MQ-Laufzeit verwendet, wenn sich das Sendeziel in einem anderen Queue Manager als das Empfangsziel befindet.

Aktivierungsspezifikation
Eine MQ-JMS-Aktivierungsspezifikation ist mindestens einer Message-driven Bean zugeordnet und stellt die Konfiguration bereit, die zum Empfangen von Nachrichten erforderlich ist.

Ziele

- Sendeziel: Das Ziel, an das die Anforderung oder Nachricht gesendet wird; das Ziel, an das die Antwortnachricht gesendet (exportiert) wird, falls diese nicht durch das MQMD-Headerfeld ReplyTo in der eingehenden Nachricht außer Kraft gesetzt wird.
- Empfangsziel: Das Ziel, an das die Anforderung bzw. Antwort oder eingehende Nachricht gesendet werden soll.

WebSphere MQ-Header:

WebSphere MQ-Header berücksichtigen bestimmte Konventionen für die Konvertierung in SCA-Nachrichten.

WebSphere MQ-Nachrichten bestehen aus einem Systemheader (MQMD), aus null oder mehr MQ-Headern (Systemheader oder benutzerdefinierte Header) sowie einem Nachrichtenhauptteil. Falls die Nachricht mehrere Nachrichtenheader enthält, ist die Reihenfolge der Header von Bedeutung.

Jeder Header enthält Informationen, die die Struktur des nachfolgenden Headers beschreiben. Der Header 'MQMD' beschreibt den ersten Header.

Syntaxanalyse bei MQ-Headern

Zur Syntaxanalyse von MQ-Headern wird eine MQ-Headerdatenbindung verwendet. Die folgenden Header werden automatisch unterstützt:

- MQRFH
- MQRFH2
- MQCIH
- MQIIH

Header, die mit den Zeichen **MQH** beginnen, werden anders gehandhabt. Bestimmte Felder des Headers werden nicht syntaktisch analysiert, sondern behalten das Format von nicht analysierten Byte bei.

Bei anderen MQ-Headern können Sie benutzerdefinierte MQ-Headerdatenbindungen schreiben, um diese Header syntaktisch zu analysieren.

Zugriff auf MQ-Header

Für den Zugriff auf MQ-Header gibt es im Produkt zwei mögliche Verfahren:
- Verwendung des Servicenachrichtenobjekts in einer Mediation
- Verwendung der API 'ContextService'

MQ-Header werden intern mit dem Element 'MQHeader' des Servicenachrichtenobjekts dargestellt. Das Element 'MQHeader' ist ein Container für Headerdaten, der 'MQControl' erweitert, jedoch ein Wertelement mit dem Typ 'anyType' enthält. Er enthält den Header 'MQMD', das Element 'MQControl' (Steuerinformationen für den MQ-Nachrichtenhauptteil) und eine Liste weiterer MQ-Header.
- Der Header 'MQMD' stellt den Inhalt der WebSphere MQ-Nachrichtenbeschreibung dar. Ausgenommen sind Informationen, die die Struktur und die Codierung des Hauptteils bestimmen.
- Das Element 'MQControl' enthält Informationen, die die Struktur und die Codierung eines Nachrichtenhauptteils bestimmen.
- Das Element 'MQHeaders' enthält eine Liste von Objekten des Typs 'MQHeader'.

Die MQ-Headerkette ist nicht verbunden. Daher enthält jeder MQ-Header im Servicenachrichtenobjekt seine eigenen Steuerinformationen (ID des codierten Zeichensatzes, Codierung und Format). Header können ohne großen Aufwand hinzugefügt oder gelöscht werden, ohne andere Headerdaten zu ändern.

Felder im Header 'MQMD' festlegen

Sie können den Header 'MQMD' unter Verwendung der Kontext-API oder über das Servicenachrichtenobjekt in einer Mediation aktualisieren. Die folgenden Felder werden automatisch an die abgehende MQ-Nachricht weitergegeben:
- Encoding
- CodedCharacterSet
- Format
- Report
- Expiry
- Feedback
- Priority
- Persistence
- CorrelId
- MsgFlags

Konfigurieren Sie die MQ-Bindung für einen Import oder Export, um die folgenden Eigenschaften in der abgehenden MQ-Nachricht weiterzugeben:

MsgID
Legen Sie für die **Anforderungsnachrichten-ID** die Einstellung für das `Kopieren aus der SCA-Nachricht` fest.

MsgType
Wählen Sie das Markierungsfeld für das **Festlegen des Nachrichtentyps auf MQMT_DATAGRAM oder MQMT_REQUEST für Anforderungs-/Antwortoperation** ab.

ReplyToQ
Wählen Sie das Markierungsfeld für das **Überschreiben der Empfangswarteschlange für Antworten auf die Anforderungsnachricht** ab.

ReplyToQMgr
Wählen Sie das Markierungsfeld für das **Überschreiben der Empfangswarteschlange für Antworten auf die Anforderungsnachricht** ab.

Ab Version 7.0 können Kontextfelder mithilfe einer benutzerdefinierten Eigenschaft für die JNDI-Zieldefinition überschrieben werden. Legen Sie die benutzerdefinierte Eigenschaft MDCTX mit dem Wert `SET_IDENTITY_CONTEXT` für das Sendeziel fest, damit die folgenden Felder an die abgehende MQ-Nachricht weitergegeben werden:

- UserIdentifier
- AppIdentityData

Legen Sie die benutzerdefinierte Eigenschaft MDCTX mit dem Wert `SET_ALL_CONTEXT` für das Sendeziel fest, damit die folgenden Eigenschaften an die abgehende MQ-Nachricht weitergegeben werden:

- UserIdentifier
- AppIdentityData
- PutApplType
- PutApplName
- ApplOriginData

Einige Felder werden nicht an die abgehende MQ-Nachricht weitergegeben. Die folgenden Felder werden beim Senden der Nachricht überschrieben:

- BackoutCount
- AccountingToken
- PutDate
- PutTime
- Offset
- OriginalLength

MQCIH statisch in einer WebSphere MQ-Bindung hinzufügen:

IBM Business Process Manager unterstützt das statische Hinzufügen von MQCIH-Headerinformationen ohne Verwendung eines Mediationsmoduls.

Es gibt mehrere Möglichkeiten, MQCIH-Headerinformationen zu einer Nachricht hinzuzufügen (z. B. durch Verwendung des Mediationsbasiselements 'Header-Setter'). Es könnte zweckmäßig sein, diese Headerinformationen statisch, ohne Verwendung eines zusätzlichen Mediationsmoduls, hinzuzufügen. Statische Headerinformationen, z. B. der CICS-Programmname, die Transaktions-ID und weitere Details zu Datenformatheadern, können im Rahmen der WebSphere MQ-Bindung definiert und hinzugefügt werden.

WebSphere MQ, MQ CICS Bridge und CICS müssen für das statische Hinzufügen von MQCIH-Headerinformationen konfiguriert werden.

Sie können Integration Designer verwenden, um den WebSphere MQ-Import mit den statischen Werten zu konfigurieren, die für die MQCIH-Headerinformationen erforderlich sind.

Wenn eine Nachricht empfangen und vom WebSphere MQ-Import verarbeitet wird, wird überprüft, ob die MQCIH-Headerinformationen in der Nachricht bereits vorhanden sind. Wenn der MQCIH vorhanden ist, werden die dynamischen Werte in der Nachricht mithilfe der im WebSphere MQ-Import definierten statischen Werte überschrieben. Wenn der MQCIH nicht vorhanden ist, wird er in der Nachricht erstellt und die im WebSphere MQ-Import definierten statischen Werte werden hinzugefügt.

Die im WebSphere MQ-Import definierten statischen Werte beziehen sich auf eine bestimmte Methode. Für verschiedene Methoden innerhalb eines WebSphere MQ-Imports können unterschiedliche statische MQCIH-Werte angegeben werden.

Über diese Funktion werden keine Standardwerte bereitgestellt, wenn der MQCIH keine spezifischen Headerinformationen enthält, da ein im WebSphere MQ-Import definierter statischer Wert den entsprechenden Wert in der eingehenden Nachricht überschreibt.

Externe Clients:

IBM Business Process Manager kann mit WebSphere MQ-Bindungen Nachrichten an externe Clients senden bzw. von diesen empfangen.

Ein externer Client (z. B. ein Webportal oder ein unternehmensweites Informationssystem) kann eine Nachricht an eine SCA-Komponente in der Anwendung durch einen Export senden oder von einer SCA-Komponente in der Anwendung durch einen Import aufgerufen werden.

Die WebSphere MQ-Exportbindung implementiert nachrichtengesteuerte Beans (Message driven bean - MDB), um Anforderungen zu überwachen, die bei dem in der Exportbindung definierten Empfangsziel eingehen. Das im Feld **send** angegebene Ziel wird verwendet, um die Antwort auf die eingehende Anforderung zu senden, falls die aufgerufene Anwendung eine Antwort bereitstellt. Ein externer Client kann somit Anwendungen über die Exportbindung aufrufen.

WebSphere MQ-Importe stellen eine Bindung zu externen Clients her und können Nachrichten an diese Clients übermitteln. Diese Nachrichten machen möglicherweise, jedoch nicht zwangsläufig, eine Antwort an den externen Client erforderlich.

Weitere Informationen zur Interaktion mit externen Clients unter Verwendung von WebSphere MQ finden Sie im Information Center von WebSphere MQ.

Fehlerbehebung für WebSphere MQ-Bindungen:

Sie können Fehler und Fehlerbedingungen, die in Verbindung mit WebSphere MQ-Bindungen auftreten, diagnostizieren und beheben.

Primäre Fehlerbedingungen

Die primären Fehlerbedingungen von WebSphere MQ-Bindungen werden durch Transaktionssemantik, durch die WebSphere MQ-Konfiguration oder durch Verweise auf bestehendes Verhalten in anderen Komponenten bestimmt. Zu den primären Fehlerbedingungen zählen die Folgenden:

- Die Herstellung der Verbindung zum Warteschlangenmanager oder zur Warteschlange von WebSphere MQ schlägt fehl.

  Das Fehlschlagen der Verbindungsherstellung zum WebSphere MQ für den Empfang von Nachrichten hat zur Folge, dass der MDB-Listener-Port nicht gestartet werden kann. Dieser Zustand wird im WebSphere Application Server-Protokoll aufgezeichnet. Persistente Nachrichten verbleiben so lange in der WebSphere MQ-Warteschlange, bis sie erfolgreich abgerufen wurden (oder gemäß WebSphere MQ abgelaufen sind).

  Das Fehlschlagen der Verbindungsherstellung zu WebSphere MQ für den Versand abgehender Nachrichten bewirkt ein Rollback der Transaktion, die die Sendeaktion steuert.
- Die Ausführung der Syntaxanalyse für eine eingehende Nachricht oder die Erstellung einer abgehenden Nachricht ist fehlgeschlagen.

  Ein Fehler in der Datenbindung bewirkt ein Rollback der Transaktion, die die Arbeit steuert.
- Das Senden der abgehenden Nachricht ist fehlgeschlagen.

  Das Fehlschlagen des Versands einer Nachricht bewirkt ein Rollback der relevanten Transaktion.
- Mehrere oder nicht erwartete Antwortnachrichten.

  Der Import erwartet für jede Anforderungsnachricht jeweils nur eine Antwortnachricht. Wenn mehr als eine Antwort eingeht oder wenn eine verspätete Antwort eingeht (d. h. eine Antwort, für die die definierte SCA-Antwortablauffrist verstrichen ist), so wird eine Laufzeitausnahmebedingung für den Ser-

vice ausgelöst. Für die Transaktion wird ein Rollback ausgeführt und die Antwortnachricht wird aus der Warteschlange entfernt oder vom Failed Event Manager verarbeitet.

Szenario falscher Verwendungen: Vergleich mit WebSphere MQ-JMS-Bindungen

Import und Export von WebSphere MQ wurden in erster Linie entwickelt, um mit nativen WebSphere MQ-Anwendungen zu interagieren und den gesamten Inhalt des WebSphere MQ-Nachrichtenhauptteils gegenüber Mediationen verfügbar zu machen. Die WebSphere MQ-JMS-Bindung wurde jedoch dazu konzipiert, mit JMS-Anwendungen zu interagieren, die auf WebSphere MQ implementiert wurden, was eine Offenlegung von Nachrichten gemäß dem JMS-Nachrichtenmodell bewirkt.

Die folgenden Szenarien sollten unter Verwendung der WebSphere MQ-JMS-Bindung erstellt werden und nicht mit der WebSphere MQ-Bindung:

- Aufrufen einer JMS-Message-driven Bean (MDB) von einem SCA-Modul, wenn die MDB auf dem WebSphere MQ-JMS-Provider implementiert ist. Verwenden Sie in diesem Fall einen WebSphere MQ-JMS-Import.
- Zulassen des Aufrufs des SCA-Moduls von einem Java EE-Komponentenservlet oder einer EJB anhand eines JMS. Verwenden Sie in diesem Fall einen WebSphere MQ-JMS-Export.
- Mediation des Inhalts einer JMS-Zuordnungsnachricht beim Durchqueren von WebSphere MQ betreiben. Verwenden Sie einen WebSphere MQ-JMS-Export und -Import in Verbindung mit der entsprechenden Datenbindung.

In gibt Fälle, in denen eine Interaktion zwischen WebSphere MQ-Bindung und WebSphere MQ-JMS-Bindung gegebenenfalls erwartet wird. Besonders bei Überbrückungen zwischen Java EE- und Nicht-Java-EE-Anwendungen von WebSphere MQ sollten Sie einen WebSphere MQ-Export und einen WebSphere MQ-JMS-Import (oder umgekehrt) in Verbindung mit den geeigneten Datenbindungen oder Mediationsmodulen (oder beiden) verwenden.

Nicht zugestellte Nachrichten

Wenn WebSphere MQ eine Nachricht ihrem Ziel nicht zustellen kann (beispielsweise wegen Konfigurationsfehlern), sendet es diese Nachrichten stattdessen an eine nominierte Warteschlange für nicht zustellbare Nachrichten.

Dabei wird dem Anfang des Nachrichtenhauptteils ein Header für nicht zustellbare Nachrichten hinzugefügt. Neben anderen Informationen enthält dieser Header die Gründe für das Fehlschlagen und das ursprüngliche Ziel.

MQ-basierte SCA-Nachrichten werden im Failed Event Manager nicht angezeigt

Wenn SCA-Nachrichten ihren Ursprung in einem Fehler oder einer Störung in der WebSphere MQ-Interaktion haben, würden Sie erwarten, diese Nachrichten im Failed Event Manager vorzufinden. Werden diese Nachrichten nicht im Failed Event Manager angezeigt, stellen Sie sicher, dass für das zugrunde liegende WebSphere MQ-Ziel ein Wert größer als 1 für die maximale Anzahl fehlgeschlagener Zustellungen definiert ist. Durch Festlegen von 2 oder höher für diesen Wert wird eine Interaktion mit dem Failed Event Manager während SCA-Aufrufen für die WebSphere MQ-Bindungen ermöglicht.

In MQ fehlgeschlagene Ereignisse werden im falschen Warteschlangenmanager wiedergegeben

Wenn eine vordefinierte Verbindungsfactory für abgehende Verbindungen verwendet werden soll, müssen die Verbindungseigenschaften mit denen übereinstimmen, die in der für abgehende Verbindungen verwendeten Aktivierungsspezifikation definiert sind.

Anhand der vordefinierten Verbindungsfactory wird bei der Wiedergabe eines fehlgeschlagenen Ereignisses eine Verbindung erstellt. Aus diesem Grund muss sie so konfiguriert sein, dass derselbe Warteschlan-

genmanager verwendet wird, von dem die Nachricht ursprünglich empfangen wurde.

Ausnahmebedingungen verarbeiten:

Die Art und Weise, in der die Bindung konfiguriert ist, bestimmt, wie Ausnahmebedingungen verarbeitet werden, die von Datenhandlern oder Datenbindungen ausgelöst werden. Außerdem gibt die Spezifik des Mediationsablaufs das Verhalten des Systems vor, wenn eine solche Ausnahmebedingung ausgelöst wird.

Wenn ein Datenhandler oder eine Datenbindung durch eine Bindung aufgerufen wird, können verschiedene Probleme auftreten. Beispielsweise kann es sein, dass ein Datenhandler eine Nachricht mit beschädigten Nutzdaten empfängt oder er versucht, eine Nachricht mit einem falschen Format zu lesen.

Das Verfahren, mit dem die Bindung eine solche Ausnahmebedingung behandelt, wird dadurch bestimmt, wie Sie den Datenhandler oder die Datenbindung implementieren. Das empfohlene Verhalten besteht darin, die Datenbindung so zu konfigurieren, dass sie eine Ausnahmebedingung des Typs **DataBindingException** auslöst.

Bei einem Datenhandler ist die Situation ähnlich. Da der Datenhandler durch die Datenbindung aufgerufen wird, wird jede Ausnahmebedingung des Datenhandlers in eine Ausnahmebedingung der Datenbindung eingeschlossen. Daher wird eine Ausnahmebindung des Typs **DataHandlerException** als Typ **DataBindingException** gemeldet.

Wenn eine Laufzeitausnahmebedingung (inklusive **DataBindingException**) ausgelöst wird, geschieht Folgendes:

- Falls der Mediationsablauf transaktionsorientiert konfiguriert ist, wird die JMS-Nachricht standardmäßig in Failed Event Manager gespeichert, damit sie manuell wiedergegeben oder gelöscht werden kann.

  Anmerkung: Sie können den Fehlerbehebungsmodus für die Bindung so ändern, dass die Nachricht zurückgesetzt und nicht in failed event manager gespeichert wird.
- Ist der Mediationsablauf nicht transaktionsorientiert, wird die Ausnahmebedingung protokolliert und die Nachricht ist nicht mehr vorhanden.

Bei einem Datenhandler ist die Situation ähnlich. Da der Datenhandler durch die Datenbindung aufgerufen wird, wird jede Ausnahmebedingung des Datenhandlers in eine Ausnahmebedingung der Datenbindung eingeschlossen. Daher wird eine Ausnahmebindung des Typs **DataHandlerException** als Typ **DataBindingException** gemeldet.

Einschränkungen bei Bindungen

Für die Verwendung von Bindungen gelten einige Einschränkungen, die in diesem Thema aufgeführt sind.

Einschränkungen für MQ-Bindungen:

Für die Verwendung von MQ-Bindungen gelten einige Einschränkungen, die in diesem Thema aufgeführt sind.

Keine Nachrichtenverteilung mit Publish/Subscribe-Verfahren

Das Publish/Subscribe-Verfahren für die Verteilung von Nachrichten wird von der MQ-Bindung gegenwärtig nicht unterstützt, obwohl WebSphere MQ selbst das Publish/Subscribe-Verfahren unterstützt. Von der MQ-JMS-Bindung wird dieses Verteilungsverfahren jedoch unterstützt.

Gemeinsam genutzte Empfangswarteschlangen

Mehrere WebSphere MQ-Exportbindungen und Importbindungen erwarten, dass alle Nachrichten in ihrer konfigurierten Empfangswarteschlange für diesen Export bzw. Import bestimmt sind. Beim Konfigurieren von Import- und Exportbindungen sollten die folgenden Aspekte berücksichtigt werden:

- Jeder MQ-Import muss eine eigene separate Empfangswarteschlange besitzen, da die MQ-Importbindung davon ausgeht, dass alle Nachrichten in der Empfangswarteschlange Antworten auf die von ihr gesendeten Anforderungen sind. Wird die Empfangswarteschlange von mehreren Importen gemeinsam genutzt, könnten Antworten durch den falschen Import empfangen werden, was dazu führt, dass die Korrelation mit der ursprünglichen Anforderungsnachricht fehlschlägt.
- Jeder MQ-Export sollte eine eigene separate Empfangswarteschlange besitzen, da andernfalls nicht vorhergesagt werden kann, welcher Export eine bestimmte Anforderungsnachricht empfängt.
- MQ-Importe und -Exporte können auf dieselbe Sendewarteschlange verweisen.

Einschränkungen für JMS-, MQ-JMS- und generische JMS-Bindungen:

Die JMS- und MQ-JMS-Bindungen unterliegen einigen Einschränkungen.

Auswirkungen der Generierung von Standardbindungen

In den folgenden Abschnitten sind die Einschränkungen für die Verwendung von JMS-, MQ-JMS- und generischen JMS-Bindungen beschrieben:

- Auswirkungen der Generierung von Standardbindungen
- Antwortkorrelationsschema
- Bidirektionale Unterstützung

Wenn Sie eine Bindung generieren, werden mehrere Felder automatisch mit Standardwerten gefüllt, falls Sie die Werte nicht selbst eingeben. Beispielsweise wird automatisch ein Name für die Verbindungsfactory erstellt. Falls Sie wissen, dass Sie Ihre Anwendung auf einen Server stellen und mit einem Client über Fernzugriff auf sie zugreifen werden, sollten Sie bei der Bindungserstellung JNDI-Namen eingeben, statt die Standardwerte zu übernehmen, da diese Werte wahrscheinlich zur Ausführungszeit über die Administrationskonsole gesteuert werden müssen.

Wenn Sie jedoch die Standardwerte akzeptiert haben und später feststellen, dass Sie nicht von einem fernen Client aus auf Ihre Anwendung zugreifen können, können Sie den Wert für die Verbindungsfactory über die Administrationskonsole explizit festlegen. Suchen Sie in den Einstellungen für die Verbindungsfactory nach dem Feld für die Providerendpunkte und fügen Sie einen Wert wie '<serverhostname>:7276' hinzu (falls die Standardportnummer verwendet wird).

Antwortkorrelationsschema

Wenn Sie das Antwortkorrelationsschema 'CorrelationId To CorrelationId' verwenden, mit dem Nachrichten in einer Anforderungs-/Antwortoperation korreliert wurden, muss die Nachricht eine dynamische Korrelations-ID enthalten.

Wenn Sie in einem Mediationsmodul mithilfe des Mediationsablaufeditors eine dynamische Korrelations-ID erstellen möchten, fügen Sie vor dem Import mit der JMS-Bindung ein Mediationsbasiselement für Zuordnung hinzu. Öffnen Sie den Zuordnungseditor. In der Zielnachricht sind die bekannten SCA-Header verfügbar. Ziehen Sie aus der Quellennachricht ein Feld, in dem eine eindeutige ID enthalten ist, in die Korrelations-ID im JMS-Header in der Zielnachricht.

Bidirektionale Unterstützung

Für JNDI-Namen (JNDI = Java Naming and Directory Interface) werden zur Laufzeit nur ASCII-Zeichen unterstützt.

Gemeinsam genutzte Empfangswarteschlangen

Mehrere Export- und Importbindungen erwarten, dass alle Nachrichten in ihrer konfigurierten Empfangswarteschlange für diesen Export bzw. Import bestimmt sind. Beim Konfigurieren von Import- und Exportbindungen sollten die folgenden Aspekte berücksichtigt werden:

- Jede Importbindung muss eine eigene separate Empfangswarteschlange besitzen, da die Importbindung davon ausgeht, dass alle Nachrichten in der Empfangswarteschlange Antworten auf die von ihr gesendeten Anforderungen sind. Wird die Empfangswarteschlange von mehreren Importen gemeinsam genutzt, könnten Antworten durch den falschen Import empfangen werden, was dazu führt, dass die Korrelation mit der ursprünglichen Anforderungsnachricht fehlschlägt.
- Jeder Export sollte eine eigene separate Empfangswarteschlange besitzen, da andernfalls nicht vorhergesagt werden kann, welcher Export eine bestimmte Anforderungsnachricht empfängt.
- Importe und Exporte können auf dieselbe Sendewarteschlange verweisen.

Geschäftsobjekte

Die Softwarebranche hat verschiedene Programmiermodelle und -frameworks entwickelt, bei denen *Geschäftsobjekte* eine einfach zu handhabende Darstellung der Geschäftsdaten für die Anwendungsverarbeitung liefern.

Für diese Geschäftsobjekte gilt im Wesentlichen Folgendes:

- Sie werden über Branchenstandards definiert.
- Sie ordnen Daten transparent zu Datenbanktabellen bzw. zu unternehmensweiten Informationssystemen zu.
- Sie unterstützen ferne Aufrufprotokolle.
- Sie liefern die Basis für die Datenprogrammiermodelle, die für die Anwendungsprogrammierung benötigt werden.

Process Designer und Integration Designer stellen Entwicklern ein auf diesen Merkmalen basierendes allgemeines Geschäftsobjektmodell bereit, das für die Darstellung von Geschäftsentitäten unterschiedlicher Arten aus verschiedenen Domänen ausgelegt ist.

In Process Designer sind die Geschäftsobjekte auf eine Datentypendarstellung ausgerichtet. Basis-Geschäftsobjekte (Variablentypen) werden in System-Toolkits bereitgestellt. Sie können auch angepasste Variablentypen erstellen, die dann als "angepasste Geschäftsobjekte" bezeichnet werden. Weitere Informationen hierzu finden Sie in Geschäftsobjekte und Variablen.

In Integration Designer, das nur zusammen mit IBM Business Process Manager Advanced erhältlich ist, können Geschäftsobjekte auch komplexere XSD-Konstrukte darstellen. In Integration Designer weisen die Geschäftsobjekte eine enge Affinität zu XML-Schemata auf. Informationen dazu, was bei der Integration von Geschäftsobjekten zu beachten ist, die in Integration Designer mit Geschäftsobjekten definiert sind, welche in Process Designer definiert sind, finden Sie in Spiegeln der Bibliothek und XML-Konstrukte nicht unterstützt.

Zur Entwicklungszeit in Integration Designer ermöglicht es das Geschäftsobjektmodell den Entwicklern, Geschäftsobjekte als XML-Schemadefinitionen zu definieren. Zur Laufzeit werden die von den XML-Schemadefinitionen definierten Geschäftsdaten als Java-Geschäftsobjekte dargestellt. Bei diesem Modell sind Geschäftsobjekte flexibel an in einem frühen Stadium erstellten Entwürfen der SDO-Spezifikation (SDO = Service Data Object) ausgerichtet und stellen die gesamte Palette der Anwendungsschnittstellen für Programmiermodelle zur Verfügung, die zum Bearbeiten von Geschäftsdaten erforderlich sind. Die nachfolgenden Unterabschnitte geben weitere Beschreibungen, wie Geschäftsobjekte mit Integration Designer verwendet werden können.

Geschäftsobjekte definieren

Zum Definieren von Geschäftsobjekten verwenden Sie den Geschäftsobjekteditor in Integration Designer. Der Geschäftsobjekteditor speichert die Geschäftsobjekte als XML-Schemadefinitionen.

Die Verwendung des XML-Schemas zum Definieren von Geschäftsobjekten bietet mehrere Vorteile:

- Das XML-Schema bietet ein standardisiertes Datendefinitionsmodell und eine Grundlage für die Interoperabilität zwischen unterschiedlich und voneinander unabhängigen heterogenen Systemen und Anwendungen. XML-Schemas werden zusammen mit WSDL (Web Services Description Language) verwendet, um standardisierte Schnittstellenverträge zwischen Komponenten, Anwendungen und Systemen bereitzustellen.
- XML-Schemas definieren ein funktionsreiches Datendefinitionsmodell für die Darstellung von Geschäftsdaten. Dieses Modell schließt - neben anderen Funktionen - komplexe Typen, einfache Typen, benutzerdefinierte Datentypen, die Typübernahme und die Kardinalität ein.
- Geschäftsobjekte können durch Geschäftsschnittstellen und -daten definiert werden, die in WSDL (Web Services Description Language) definiert sind, sowie durch XML-Schemas von Industriestandardorganisationen oder von anderen Systemen und Anwendungen. Integration Designer kann diese Geschäftsobjekte direkt importieren.

Integration Designer unterstützt außerdem die Erkennung von Geschäftsdaten in Datenbanken und unternehmensweiten Informationssystemen und die anschließende Generierung der Geschäftsobjektdefinition als standardisiertem XML-Schema aus diesen Geschäftsdaten. Auf diese Weise generierte Geschäftsobjekte werden häufig als *anwendungsspezifische Geschäftsobjekte* bezeichnet, weil sie die Struktur der Geschäftsdaten abbilden, die im unternehmensweiten Informationssystem definiert ist.

Wenn ein Prozess Daten aus vielen verschiedenen Informationssystemen bearbeitet, kann es von Nutzen sein, die unterschiedlichen Darstellungen der Geschäftsdaten (z. B. 'CustomerEIS1' und 'CustomerEIS2' oder 'OrderEIS1' und 'OrderEIS2') in eine einzige kanonische Darstellung (z. B. 'Customer' oder 'Order') zu transformieren. Die kanonische Darstellung wird häufig als *generisches Geschäftsobjekt* bezeichnet.

Geschäftsobjektdefinitionen werden, insbesondere für generische Geschäftsobjekte, häufig von mehr als einer Anwendung verwendet. Zur Unterstützung dieser Wiederverwendung ermöglicht Integration Designer die Erstellung von Geschäftsobjekten in Bibliotheken, die anschließend mehreren Anwendungsmodulen zugeordnet werden können.

Anhand der Web Services Description Language (WSDL) werden die Verträge für die Services definiert, die über ein SCA-Anwendungsmodul bereitgestellt und verwendet werden, sowie die Verträge, die zum Erstellen der Komponenten in einem Anwendungsmodul verwendet werden. In einem Vertrag kann WSDL sowohl die Operationen als auch die Geschäftsobjekte darstellen (die durch das XML-Schema zur Darstellung der Geschäftsdaten definiert sind).

Mit Geschäftsobjekten arbeiten

SCA (Service Component Architecture - Servicekomponentenarchitektur) stellt das Framework für die Definition eines Anwendungsmoduls, der von ihm bereitgestellten Services, der von ihm verwendeten Services und der Zusammenstellung von Komponenten bereit, die die Geschäftslogik des Anwendungsmoduls zur Verfügung stellen. Geschäftsobjekte spielen in der Anwendung eine wichtige Rolle, denn sie definieren die Geschäftsdaten, mit denen die Service- und Komponentenverträge beschrieben werden, sowie die Geschäftsdaten, die von den Komponenten verarbeitet werden.

Im folgenden Diagramm, in dem ein SCA-Anwendungsmodul dargestellt ist, werden viele der Stellen erkennbar, an denen ein Entwickler mit Geschäftsobjekten arbeitet.

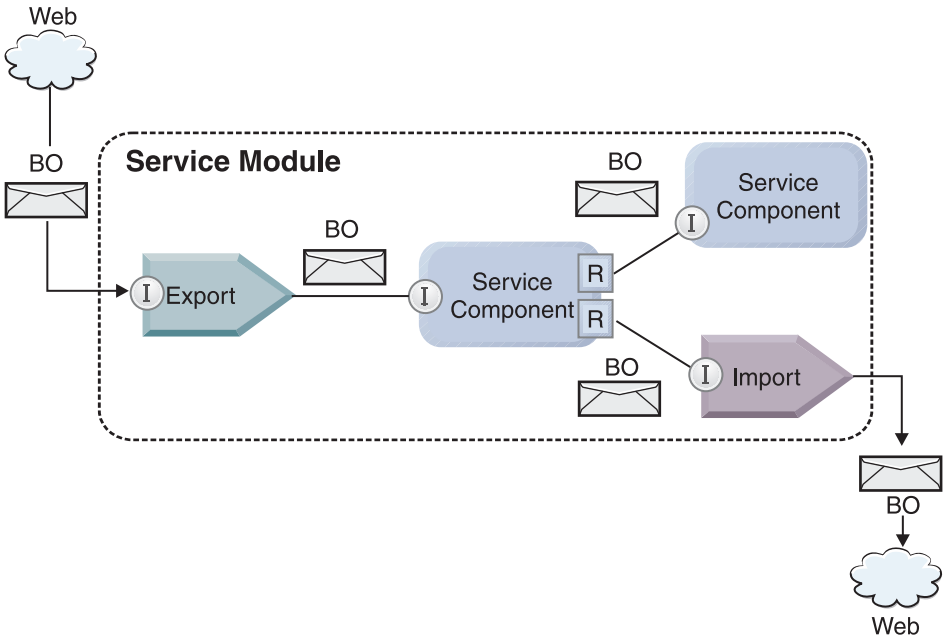


Abbildung 86. Geschäftsobjekte stellen die Daten dar, die in einer Anwendung zwischen Services fließen.

Anmerkung: In diesem Thema ist beschrieben, wie Geschäftsobjekte von SCA-Anwendungsmodulen verwendet werden. Falls Sie Java-Schnittstellen einsetzen, können die SCA-Anwendungsmodule auch Java-Objekte verarbeiten.

Programmiermodell für Geschäftsobjekte

Das Programmiermodell für Geschäftsobjekte besteht aus einer Reihe von Java-Schnittstellen, die Folgendes darstellen:

- Geschäftsobjektdefinition und Instanzdaten
- Gruppe von Services, die die Operationen für die Geschäftsobjekte unterstützen

Definitionen von Geschäftsobjekttypen werden durch die Schnittstellen 'commonj.sdo.Type' und 'commonj.sdo.Property' dargestellt. Das Programmiermodell für Geschäftsobjekte bietet eine Reihe von Regeln für die Zuordnung von XML-Schemainformationen für komplexe Typen zur Typschnittstelle und für die Zuordnung der einzelnen Elemente in der Definition des komplexen Typs zur Eigenschaftenschnittstelle.

Geschäftsobjektinstanzen werden durch die Schnittstelle 'commonj.sdo.DataObject' dargestellt. Das Programmiermodell für Geschäftsobjekte ist nicht typisiert. Dies bedeutet, dass dieselbe Schnittstelle 'commonj.sdo.DataObject' für die Darstellung verschiedener Geschäftsobjektdefinitionen (z. B. 'Customer' und 'Order') verwendet werden kann. Die Definition, welche Eigenschaften in jedem Geschäftsobjekt festgelegt und aus diesem abgerufen werden können, werden durch Typinformationen bestimmt, die in dem jedem Geschäftsobjekt zugeordneten XML-Schema definiert sind.

Das Verhalten des Programmiermodells für Geschäftsobjekte basiert auf der Spezifikation von 'Service Data Object 2.1'. Zusätzliche Informationen enthalten die Spezifikation, die Lernprogramme und Javadocs für 'SDO 2.1 for Java', die Sie im Internet unter der Adresse http://www.oasis-opencsa.org/ finden.

Geschäftsobjektservices unterstützen viele Lebenszyklusoperationen (z. B. Erstellung, Gleichheit, Parsing und Serialisierung) für Geschäftsobjekte.

Spezifikationen zum Programmiermodell für Geschäftsobjekte finden Sie unter Programmieren mit Geschäftsobjektservices und unter Generated API and SPI documentation on business objects.

Bindungen, Datenbindungen und Datenhandler

Wie in Abb. 41 auf Seite 159 gezeigt, werden Geschäftsdaten, mit denen von SCA-Anwendungsmodulen bereitgestellte Services aufgerufen werden, in Geschäftsobjekte transformiert, damit die SCA-Komponenten die Geschäftsdaten bearbeiten können. Die von SCA-Komponenten bearbeiteten Geschäftsobjekte werden analog in das Datenformat konvertiert, das von den externen Services benötigt wird.

In manchen Fällen (beispielsweise bei der Web-Service-Bindung) transformiert die Bindung, die zum Exportieren und Importieren von Services verwendet wird, die Daten automatisch in das geeignete Format. In anderen Fällen (beispielsweise bei der JMS-Bindung) können Entwickler eine Datenbindung oder einen Datenhandler bereitstellen, von der/dem nicht native Format in Geschäftsobjekte transformiert werden, die durch die Datenobjektschnittstelle (DataObject) dargestellt werden.

Weitere Informationen zum Entwickeln von Datenbindungen und Datenhandlern enthalten die Themen „Datenhandler" auf Seite 60 und „Datenbindungen" auf Seite 62.

Komponenten

SCA-Komponenten definieren ihre Bereitstellungs- und Verarbeitungsserviceverträge mit einer Kombination aus WSDL (Web Services Description Language) und XML-Schema. Die von SCA zwischen Komponenten übergebenen Geschäftsdaten werden unter Verwendung der Schnittstelle 'DataObject' als Geschäftsobjekte dargestellt. SCA prüft, ob diese Geschäftsobjekttypen mit dem Schnittstellenvertrag kompatibel sind, der durch die aufzurufende Komponente definiert ist.

Die Programmiermodellabstraktionen für die Bearbeitung von Geschäftsobjekten variieren je nach Komponente. Die POJO-Komponente und das Basiselement 'Angepasst' der Mediationsablaufkomponente ermöglichen eine direkte Bearbeitung der Geschäftsobjekte, da die Java-Programmierung durch die Verwendung der Programmierschnittstelle und Services für Geschäftsobjekte direkt möglich ist. Die meisten Komponenten bieten für die Bearbeitung von Geschäftsobjekten Abstraktionen höherer Ebene, aber auch Java-Code-Snippets, um angepasstes Verhalten in den Schnittstellen und Services von Geschäftsobjekten zu definieren.

Geschäftsobjekte können entweder mit einer Kombination aus den Komponenten für die Schnittstellenablaufmediation und die Geschäftsobjektzuordnung oder aus der Mediationsablaufkomponente und deren Basiselement 'XML-Zuordnung' transformiert werden. Diese Funktionen für die Geschäftsobjekttransformation sind zur Konvertierung von anwendungsspezifischen Geschäftsobjekten in generische Geschäftsobjekte (und umgekehrt) von Nutzen.

Spezielle Geschäftsobjekte

Servicenachrichtenobjekte und Geschäftsgrafiken sind zwei spezielle Typen von Geschäftsobjekten, die für bestimmte Anwendungszwecke eingesetzt werden.

Servicenachrichtenobjekt

Ein Servicenachrichtenobjekt ist ein spezialisiertes Geschäftsobjekt, mit dem in Mediationsablaufkomponenten die Erfassung der Daten dargestellt wird, die einem Serviceaufruf zugeordnet sind.

Ein Servicenachrichtenobjekt hat eine festgelegte Struktur der höchsten Ebene, die aus Headern, Kontext, Hauptteil und Anhängen (sofern vorhanden) besteht.

- Header übertragen Informationen im Zusammenhang mit dem Serviceaufruf über ein bestimmtes Protokoll bzw. eine bestimmte Bindung. Beispiele sind SOAP-Header und JMS-Header.
- Kontextdaten übertragen zusätzliche logische Informationen, die dem Aufruf zugeordnet sind, während dieser von der Mediationsablaufkomponente verarbeitet wird. Diese Informationen gehören normalerweise nicht zu den Anwendungsdaten, die von Clients gesendet oder empfangen werden.

- Der Hauptteil des Servicenachrichtenobjekts überträgt die Geschäftsnutzdaten, die die Kernanwendungsnachrichten- oder Aufrufdaten in Form eines Standardgeschäftsobjekts darstellen.

Das Servicenachrichtenobjekt kann außerdem Anhangsdaten für Web-Service-Aufrufe übertragen, die SOAP mit Anhängen verwenden.

Mediationsabläufe führen solche Tasks als Anforderungsweiterleitung und Datentransformation aus. Das Servicenachrichtenobjekt ermöglicht die kombinierte Sicht von Header- und Nutzdateninhalt in einer einzigen einheitlichen Struktur.

Geschäftsgrafik

Eine Geschäftsgrafik ist ein spezielles Geschäftsobjekt, mit dem die Unterstützung für die Datensynchronisation in Integrationsszenarios bereitgestellt wird.

Dies soll am Beispiel von zwei unternehmensweiten Informationssystemen erläutert werden, die eine Darstellung für einen bestimmten Auftrag enthalten. Wenn der Auftrag in einem System geändert wird, kann an das andere System eine Nachricht gesendet werden, um die Auftragsdaten zu synchronisieren. Geschäftsgrafiken unterstützen ein Konzept, bei dem lediglich der geänderte Teil des Auftrags an das andere System gesendet und mit Informationen zur Änderungsübersicht versehen ist, um den Typ der Änderung zu definieren.

In diesem Beispiel würde eine Auftragsgeschäftsgrafik dem anderen System mitteilen, dass eine der Artikelpositionen im Auftrag gelöscht und die Auftragseigenschaft für das voraussichtliche Lieferdatum aktualisiert wurde.

Geschäftsgrafiken können in Integration Designer ohne großen Aufwand zu Geschäftsobjekten hinzugefügt werden. Sie sind am häufigsten in Szenarios zu finden, in denen WebSphere-Adapter verwendet werden oder die Migration von WebSphere InterChange Server-Anwendungen unterstützt werden muss.

Parsingmodus für Geschäftsobjekte

Integration Designer bietet eine Eigenschaft für Module und Bibliotheken, mit der Sie für Geschäftsobjekte den XML-Parsingmodus 'Vollständig' oder 'Bedarfsorientiert' definieren können.

- Falls die Option auf *Vollständig* gesetzt ist, werden XML-Byteströme zur Erstellung des Geschäftsobjekts vollständig syntaktisch analysiert.
- Ist die Option auf *Bedarfsorientiert* gesetzt, wird das Geschäftsobjekt normal erstellt, das eigentliche Parsing des XML-Bytestroms wird jedoch verzögert und der Strom wird nur partiell analysiert, wenn auf die Eigenschaften des Geschäftsobjekts zugegriffen wird.

Daten, die keine XML-Daten sind, werden in beiden XML-Parsingmodi zur Erstellung des Geschäftsobjekts vollständig analysiert.

Hinweise zur Auswahl des Parsingmodus für Geschäftsobjekte

Der Parsingmodus für Geschäftsobjekte legt fest, wie XML-Daten während der Laufzeit syntaktisch analysiert werden. Ein Parsingmodus für Geschäftsobjekte wird für ein Modul oder eine Bibliothek bei der Erstellung dieses Moduls bzw. dieser Bibliothek definiert. Sie können den Parsingmodus des Moduls oder der Bibliothek ändern. Jedoch sind hierbei gewisse Auswirkungen zu beachten.

Der Parsingmodus für ein Geschäftsobjekt wird auf der Modul- und Bibliotheksebene festgelegt. Module, die in einer Version von IBM Integration Designer vor Version 7 erstellt wurden, werden im vollständigen Parsingmodus ausgeführt, ohne dass Änderungen erforderlich sind. Standardmäßig werden Module und Bibliotheken, die in IBM Integration Designer ab Version 7 erstellt werden, mit dem geeignetsten Parsingmodus abhängig von einer Reihe von Faktoren definiert. Solche Faktoren sind zum Beispiel der Parsingmodus der vorhandenen Projekte in Ihrem Arbeitsbereich oder der Parsingmodus abhängiger Projekte

oder anderer Projekte in derselben Lösung usw. Sie können den Parsingmodus für Geschäftsobjekte eines Moduls oder einer Bibliothek an Ihre Implementierung anpassen, wobei jedoch die folgenden Punkte zu beachten sind.

Hinweise

- Der bedarfsorientierte Parsingmodus für Geschäftsobjekte verarbeitet XML-Daten schneller. Allerdings bestehen Kompatibilitätsunterschiede zwischen dem vollständigen Parsingmodus und dem bedarfsorientierten Modus, die zu beachten sind, bevor die Konfiguration eines Moduls oder einer Bibliothek geändert wird. Diese Unterschiede haben Auswirkungen auf das Laufzeitverhalten der Module. Informationen dazu, welcher Parsingmodus sich für Ihre Anwendung optimal eignet, finden Sie im Abschnitt über die Vorteile des bedarfsorientierten Parsingmodus im Vergleich zum vollständigen Parsingmodus in den zugehörigen Links.
- Ein Modul kann nur zur Ausführung in einem Parsingmodus konfiguriert werden. Bibliotheken können zur Unterstützung eines der beiden Parsingmodi oder zur Unterstützung beider Parsingmodi konfiguriert werden. Eine Bibliothek, die zur Unterstützung beider Parsingmodi konfiguriert ist, kann sowohl von einem Modul mit dem vollständigen Parsingmodus als auch von einem Modul mit dem bedarfsorientierten Parsingmodus referenziert werden. Der Parsingmodus einer Bibliothek zur Laufzeit wird durch die Module festgelegt, die die Bibliothek referenzieren. Zur Laufzeit deklariert ein Modul seinen Parsingmodus und dieser Parsingmodus wird von dem Modul und allen Bibliotheken verwendet, auf die das Modul zugreift.
- Module und Bibliotheken, die für verschiedene Parsingmodi konfiguriert sind, sind in den folgenden Fällen kompatibel:
  - Module und Bibliotheken, die mit dem bedarfsorientierten Parsingmodus konfiguriert sind, sind mit Bibliotheken kompatibel, die entweder den bedarfsorientierten Parsingmodus verwenden oder sowohl den vollständigen als auch den bedarfsorientierten Parsingmodus verwenden.
  - Module und Bibliotheken, die mit dem vollständigen Parsingmodus konfiguriert sind, sind mit Bibliotheken kompatibel, die entweder den vollständigen Parsingmodus verwenden oder sowohl den vollständigen als auch den bedarfsorientierten Parsingmodus verwenden.
  - Bibliotheken, die mit dem bedarfsorientierten und dem vollständigen Parsingmodus konfiguriert sind, sind nur mit Bibliotheken kompatibel, die sowohl den bedarfsorientierten Parsingmodus als auch den vollständigen Parsingmodus verwenden.
- Verwenden Sie denselben Parsingmodus für interagierende Module, die mithilfe der SCA-Bindung kommunizieren. Wenn Module mit verschiedenen Parsingmodi kommunizieren, kann es zu Leistungsproblemen kommen.

Zugehörige Konzepte:

„Vorteile des Parsingmodus 'Bedarfsorientiert' (Lazy) gegenüber dem Modus 'Vollständig' (Eager)” auf Seite 162
Einige Anwendungen profitieren vom XML-Parsingmodus 'Bedarfsorientiert', während sich für andere Anwendungen beim Parsingmodus 'Vollständig' ein besseres Leistungsverhalten ergibt. Es empfiehlt sich, für Ihre Anwendungen einen Vergleichspunkt in beiden Parsingmodi zu ermitteln, um den für die speziellen Merkmale der Anwendung am besten geeigneten Modus festzustellen.

Vorteile des Parsingmodus 'Bedarfsorientiert' (Lazy) gegenüber dem Modus 'Vollständig' (Eager)

Einige Anwendungen profitieren vom XML-Parsingmodus 'Bedarfsorientiert', während sich für andere Anwendungen beim Parsingmodus 'Vollständig' ein besseres Leistungsverhalten ergibt. Es empfiehlt sich, für Ihre Anwendungen einen Vergleichspunkt in beiden Parsingmodi zu ermitteln, um den für die speziellen Merkmale der Anwendung am besten geeigneten Modus festzustellen.

Anwendungen, die umfangreiche XML-Datenströme syntaktisch analysieren (= parsen), bieten bei Verwendung des XML-Parsingmodus 'Bedarfsorientiert' wahrscheinlich eine gesteigerte Leistung. Mit steigender Größe des XML-Bytestroms und mit abnehmendem Volumen der Daten aus dem Bytestrom, auf die durch die Anwendung zugegriffen wird, nimmt die Leistungsverbesserung zu.

Die folgenden Anwendungen zeigen im Parsingmodus 'Vollständig' wahrscheinlich ein besseres Leistungsverhalten:

- Anwendungen, die Datenströme syntaktisch analysieren, die keine XML-Datenströme sind
- Anwendungen, die Nachrichten verwenden, die mit dem Service 'BOFactory' erstellt wurden
- Anwendungen, die sehr kleine XML-Nachrichten syntaktisch analysieren

Zugehörige Verweise:

„Hinweise zur Auswahl des Parsingmodus für Geschäftsobjekte" auf Seite 161
Der Parsingmodus für Geschäftsobjekte legt fest, wie XML-Daten während der Laufzeit syntaktisch analysiert werden. Ein Parsingmodus für Geschäftsobjekte wird für ein Modul oder eine Bibliothek bei der Erstellung dieses Moduls bzw. dieser Bibliothek definiert. Sie können den Parsingmodus des Moduls oder der Bibliothek ändern. Jedoch sind hierbei gewisse Auswirkungen zu beachten.

Hinweise zur Anwendungsmigration und -entwicklung

Falls Sie eine Anwendung, die ursprünglich für die Verwendung des Parsingmodus 'Vollständig' entwickelt wurde, so konfigurieren, dass künftig der Parsingmodus 'Bedarfsorientiert' verwendet werden soll, oder falls Sie beabsichtigen, bei einer Anwendung den Wechsel zwischen den Modi 'Vollständig' und 'Bedarfsorientiert' zu ermöglichen, müssen Sie die Unterschiede zwischen den Modi sowie die Hinweise für den Moduswechsel berücksichtigen.

Fehlerbehandlung

Falls der syntaktisch analysierte XML-Bytestrom falsch formatiert ist, treten Parsingausnahmebedingungen auf.

- Im XML-Parsingmodus 'Vollständig' treten diese Ausnahmebedingungen auf, sobald das Geschäftsobjekt aus dem eingehenden XML-Datenstrom syntaktisch analysiert wird.
- Ist der XML-Parsingmodus 'Bedarfsorientiert' konfiguriert, treten die Parsingausnahmebedingungen latent auf, wenn der Zugriff auf die Geschäftsobjekteigenschaften erfolgt und der falsch formatierte XML-Teil syntaktisch analysiert wird.

Zur Behandlung von falsch formatierten XML-Daten haben Sie die folgenden Möglichkeiten:

- Enterprise Service Bus für die Auswertung von eingehenden XML-Daten bei der Ersterkennung implementieren
- Logik für die verzögerte Fehlererkennung am Zugriffspunkt für Geschäftsobjekteigenschaften schreiben

Ausnahmebedingungsstacks und -nachrichten

Da den XML-Parsingmodi 'Vollständig' und 'Bedarfsorientiert' verschiedene Implementierungen zugrunde liegen, haben die durch die Programmierschnittstellen für Geschäftsobjekte ausgelösten Stack-Traces und Services denselben Ausnahmebedingungsklassennamen, enthalten jedoch möglicherweise nicht dieselbe Ausnahmebedingungsnachricht oder eingeschlossene Gruppe von implementierungsspezifischen Ausnahmebedingungsklassen.

XML-Serialisierungsformat

Der XML-Parsingmodus 'Bedarfsorientiert' bietet eine Leistungsoptimierung, die versucht, nicht geänderte XML-Datei aus dem eingehenden Bytestrom bei der Serialisierung in den ausgehenden Bytestrom zu kopieren. Dies hat eine Leistungssteigerung zur Folge, aber das Format des abgehenden XML-Bytestroms kann unterschiedlich sein, falls das gesamte Geschäftsobjekt im XML-Parsingmodus 'Bedarfsorientiert' aktualisiert oder im XML-Parsingmodus 'Vollständig' ausgeführt wurde.

Obwohl das XML-Serialisierungsformat unter Umständen syntaktisch nicht gänzlich äquivalent ist, ist der vom Geschäftsobjekt bereitgestellte semantische Wert ungeachtet der Parsingmodi äquivalent. Daher können XML-Daten ohne Weiteres mit semantischer Äquivalenz zwischen Anwendungen übergeben werden, die in unterschiedlichen Parsingmodi ausgeführt werden.

Prüfprogramm für Geschäftsobjektinstanz

Das Prüfprogramm für die Geschäftsobjektinstanz im XML-Parsingmodus 'Bedarfsorientiert' bietet eine Prüfung mit größerer Formattreue für Geschäftsobjekte, insbesondere in Bezug auf die Facettenprüfung bei Eigenschaftswerten. Aufgrund dieser Verbesserungen erfasst das Instanzprüfprogramm für den Parsingmodus 'Bedarfsorientiert' zusätzliche Aspekte, die im Parsingmodus 'Vollständig' nicht berücksichtigt werden, und bietet so genauere Fehlernachrichten.

XML-Zuordnungen in Version 602

Mediationsabläufe, die ursprünglich vor WebSphere Integration Developer Version 6.1 entwickelt wurden, enthalten möglicherweise Basiselemente vom Typ 'Zuordnung' mit einer Zuordnung oder einem Style-Sheet, die bzw. der nicht direkt im XML-Parsingmodus 'Bedarfsorientiert' ausgeführt werden kann. Wenn eine Anwendung für die Verwendung im XML-Parsingmodus 'Bedarfsorientiert' migriert wird, können die Zuordnungsdateien, die zu den Basiselementen vom Typ 'Zuordnung' gehören, vom Migrationsassistenten automatisch so aktualisiert werden, dass sie im neuen Modus ausgeführt werden. Wenn ein Basiselement vom Typ 'Zuordnung' jedoch direkt ein Style-Sheet referenziert, das manuell bearbeitet wurde, wird das Style-Sheet nicht migriert und kann nicht im XML-Parsingmodus 'Bedarfsorientiert' ausgeführt werden.

Private unveröffentlichte APIs

Falls eine Anwendung unveröffentlichte private implementierungsspezifische Programmierschnittstellen für Geschäftsobjekte nutzt, schlägt die Kompilierung der Anwendung bei einem Wechsel des Parsingmodus wahrscheinlich fehl. Beim Parsingmodus 'Vollständig' sind diese privaten Schnittstellen normalerweise Implementierungsklassen für Geschäftsobjekte, die durch EMF (Eclipse Modeling Framework) definiert sind.

In allen Fällen wird dringend empfohlen, private APIs aus der Anwendung zu entfernen.

EMF-APIs für Servicenachrichtenobjekte

Eine Mediationskomponente in IBM Integration Designer bietet die Möglichkeit, Nachrichteninhalte mithilfe der Java-Klassen und -Schnittstellen aus dem Paket 'com.ibm.websphere.sibx.smobo' zu manipulieren. Im XML-Parsingmodus 'Bedarfsorientiert' können weiterhin die Java-Schnittstellen aus dem Paket 'com.ibm.websphere.sibx.smobo' verwendet werden. Methoden, die EMF-Klassen und -Schnittstellen direkt referenzieren oder die von EMF-Schnittstellen geerbt wurden, schlagen jedoch wahrscheinlich fehl.

Das Servicenachrichtenobjekt und sein Inhalt können im XML-Parsingmodus 'Bedarfsorientiert' nicht in EMF-Objekte umgesetzt werden.

Service 'BOMode'

Mit dem Service 'BOMode' wird ermittelt, ob gegenwärtig der XML-Parsingmodus 'Vollständig' oder 'Bedarfsorientiert' ausgeführt wird.

Migration

Alle Anwendungen vor Version 7.0.0.0 werden im XML-Parsingmodus 'Vollständig' ausgeführt. Werden sie unter Verwendung der BPM-Laufzeitmigrationstools einer Laufzeitmigration unterzogen, erfolgt die Ausführung weiterhin im XML-Parsingmodus 'Vollständig'.

Damit eine Anwendung einer älteren Version als Version 7.0.0.0 für die Verwendung des XML-Parsingmodus 'Bedarfsorientiert' konfiguriert werden kann, müssen Sie zunächst die Artefakt der Anwendung mit Integration Designer migrieren. Nach der Migration konfigurieren Sie die Anwendung für das XML-Parsing 'Bedarfsorientiert'.

Das Thema Quellenartefakte migrieren enthält Informationen zum Migrieren von Artefakten in Integration Designer. Unter Parsingmodus des Geschäftsobjekts von Modulen und Bibliotheken konfigurieren finden Sie Angaben über das Festlegen des Parsingmodus.

Beziehungen

Eine Beziehung ist eine Zuordnung zwischen mindestens zwei Datenentitäten, die in der Regel Business-Objekte sind. In IBM Business Process Manager Advanced können Beziehungen zum Transformieren von Daten verwendet werden, die in Geschäftsobjekten oder anderen Daten gleich sind, aber unterschiedlich dargestellt werden; oder sie werden für die Erstellung von Zuordnungen zwischen verschiedenen Objekten in unterschiedlichen Anwendungen verwendet. Sie können über mehrere Anwendungen, Lösungen oder sogar Produkte hinweg gemeinsam genutzt werden.

Der Relationship Service in IBM Business Process Manager Advanced stellt die Infrastruktur und Operationen für die Verwaltung von Beziehungen bereit. Da er die Bearbeitung von Business-Objekten unabhängig von der jeweiligen Position ermöglicht, bietet er eine ganzheitliche Sicht auf alle Anwendungen in einem Unternehmen und dient zudem als Baustein für BPM-Lösungen. Da Beziehungen erweiterbar und einfach zu verwalten sind, können sie in komplexen Integrationslösungen verwendet werden.

Was sind Beziehungen?

Eine Beziehung ist eine Zuordnung zwischen Business-Objekten. Jedes Business-Objekt in einer Beziehung wird als *Teilnehmer* in der Beziehung bezeichnet. Die Teilnehmer in der Beziehung unterscheiden sich durch die Funktion oder *Rolle*, die sie in der Beziehung einnehmen. Eine Beziehung enthält eine Liste von Rollen.

Die *Beziehungsdefinition* beschreibt die einzelnen Rollen und gibt an, wie diese Rollen zueinander in Beziehung stehen. Ferner wird die 'Gesamtform' der Beziehung beschrieben. Es wäre beispielsweise möglich, dass eine Rolle nur einen Teilnehmer, eine andere Rolle dagegen so viele Teilnehmer wie nötig haben kann. Sie könnten eine *Auto-Eigentümer*-Beziehung definieren, in der ein Eigentümer möglicherweise mehrere Autos besitzt. Eine Instanz könnte beispielsweise die folgenden Teilnehmer für jede dieser Rollen aufweisen:

- Auto (Ferrari)
- Eigentümer (John)

Die Beziehungsdefinition ist eine Schablone für die *Beziehungsinstanz*. Die Instanz ist die Laufzeitinstanzierung der Beziehung. Im *Auto-Eigentümer*-Beispiel kann eine Instanz die folgenden Zuordnungen beschreiben:

- John ist Eigentümer von Ferrari.
- Sara ist Eigentümer von Mazda.
- Bob ist Eigentümer von Ferrari.

Bei Verwendung von Beziehungen ist die benutzerdefinierte Erstellung von Persistenz zur Beziehungsverfolgung innerhalb der Geschäftslogik nicht mehr erforderlich. In bestimmten Szenarios übernimmt der Relationship Service alle erforderlichen Arbeitsschritte. Siehe dazu das Beispiel im Abschnitt zu Identitätsbeziehungen.

Szenarios

Im Folgenden ist ein typisches Beispiel für eine Situation aufgeführt, in der eine Integrationslösung Beziehungen verwenden könnte. Ein Großunternehmen kauft mehrere Unternehmen oder Business-Units. Jede Business-Unit verwendet zur Überwachung von Personal und Notebooks unterschiedliche Software. Das Unternehmen sucht nach einer Möglichkeit zur Überwachung der Mitarbeiter und deren Notebooks. An die Lösung werden die folgenden Anforderungen gestellt:

- Alle Mitarbeiter in den verschiedenen Unternehmensbereichen sollen so angezeigt werden, als befänden sie sich in derselben Datenbank.
- Eine Gesamtansicht aller Notebooks soll möglich sein.
- Die Mitarbeiter sollen sich am System anmelden und ein Notebook erwerben können.
- Die verschiedenen Unternehmensanwendungssysteme sollen in die verschiedenen Geschäftsbereiche aufgenommen werden.

Um diese Ziele zu erreichen, muss das Unternehmen beispielsweise sicherstellen, dass John Smith und John A. Smith in verschiedenen Anwendungen als derselbe Mitarbeiter angezeigt wird. Es muss zum Beispiel eine Möglichkeit geben, eine einzelne Entität über den Geltungsbereich mehrerer Anwendungen hinweg zu konsolidieren.

Zu den komplexeren Beziehungsszenarios gehört die Erstellung von BPEL-Prozessen, die Beziehungen zwischen verschiedenen Objekten in mehreren Anwendungen herstellen. Bei komplexen Beziehungsszenarios befinden sich die Business-Objekte in der Integrationslösung und nicht in den Anwendungen. Der Relationship Service stellt eine Plattform zur persistenten Verwaltung von Beziehungen bereit. Vor dem Relationship Service muss jedoch ein eigener Persistenzservice für Objekte erstellt werden. Zwei Beispiele für komplexe Szenarios:

- Sie verfügen über das Business-Objekt **Auto** mit einer VIN-Nummer in einer SAP-Anwendung und möchten den Umstand protokollieren, dass dieses Auto einer anderen Person gehört. Die Eigentumsbeziehung besteht jedoch mit einer Person in einer PeopleSoft-Anwendung. In diesem Beziehungsmuster gibt es zwei Lösungen und Sie müssen eine Brücke zwischen diesen beiden Lösungen bauen.
- Ein großes Einzelhandelsunternehmen möchte die Waren überwachen, die gegen Rückerstattung oder Kredit zurückgegeben werden. Daran sind zwei unterschiedliche Anwendungen beteiligt: ein Auftragsbearbeitungssystem für Einkäufe und ein Retourenbearbeitungssystem für Retouren. Die Business-Objekte befinden sich in mehreren Anwendungen und Sie suchen nach einer Möglichkeit, die Beziehungen zwischen diesen Objekten anzuzeigen.

Allgemeine Verwendungsmuster

Die häufigsten Beziehungsmuster sind *Äquivalenzmuster*. Diese basieren auf der Arbeit mit Querverweisen oder Korrelationen. Es gibt zwei Typen von Beziehungen, die zu diesem Muster passen: *Nicht-Identitätsbeziehungen* und *Identitätsbeziehungen*.

- **Nicht-Identitätsbeziehungen** stellen Beziehungen zwischen Business-Objekten oder anderen Daten auf einer Eins-zu-viele- oder Viele-zu-viele-Basis her. Für jede Beziehungsinstanz können eine oder mehrere Instanzen jedes Teilnehmers existieren. Ein Typ von Nicht-Identitätsbeziehung ist eine statische Referenzbeziehung. Ein Beispiel für diesen Beziehungstyp ist eine Beziehung, bei der sich der Eintrag **CA** in einer SAP-Anwendung auf den Eintrag **California** in einer Siebel-Anwendung bezieht.
- **Identitätsbeziehungen** stellen Beziehungen zwischen Business-Objekten oder anderen Daten auf einer Eins-zu-eins-Basis her. Für jede Beziehungsinstanz kann nur eine Instanz jedes Teilnehmers existieren. Identitätsbeziehungen erfassen Querverweise zwischen Business-Objekten, die semantisch äquivalent sind, aber in verschiedenen Anwendungen unterschiedlich erkannt werden. Jeder Teilnehmer in der Beziehung ist einem Business-Objekt zugeordnet, das über einen Wert (oder eine Kombination von Werten) verfügt, der das Objekt eindeutig identifiziert. Identitätsbeziehungen transformieren normalerweise die Schlüsselattribute von Business-Objekten wie ID-Nummern und Produktcodes.

  Wenn beispielsweise Business-Objekte vom Typ **Auto** in SAP-, PeopleSoft- und Siebel-Anwendungen enthalten sind und eine Lösung erstellt werden soll, die diese synchronisiert, müssen Sie normalerweise eine manuelle Beziehungssynchronisationslogik in sechs Zuordnungen einführen:

  SAP -> generisch

  generisch -> SAP

  PeopleSoft -> generisch

  generisch -> PeopleSoft

Siebel -> generisch

generisch -> Siebel

Wenn Sie jedoch Beziehungen in Ihrer Lösung verwenden, stellt der Relationship Service vordefinierte Musterimplementierungen bereit, die diese Beziehungsinstanzen für Sie verwalten.

Tools zum Arbeiten mit Beziehungen

Der *Relationship Editor* in Integration Designer ist das Tool zum Modellieren und Entwerfen von Business-Integrationsbeziehungen und Rollen. Detaillierte Hintergrund- und Taskinformationen zur Erstellung von Beziehungen und zur Verwendung des Relationship Editors finden Sie unter Beziehungen erstellen.

Der *Relationship Service* ist ein Infrastrukturservice in IBM Business Process Manager, der Beziehungen und Rollen im System verwaltet und Operationen für die Beziehungs- und Rollenverwaltung bereitstellt.

Der *Relationship Manager* ist die Verwaltungsschnittstelle zur Verwaltung von Beziehungen. Der Zugriff erfolgt über die Relationship Manager-Seiten der Administrationskonsole.

Beziehungen können programmgesteuert über Relationship Service-APIs aufgerufen werden.

Relationship Service

Der Relationship Service speichert Beziehungsdaten in Beziehungstabellen, über die anwendungsspezifische Werte aus verschiedenen Anwendungen oder Lösungen überwacht werden. Der Relationship Service stellt Operationen für die Beziehungs- und Rollenverwaltung bereit.

Funktionsweise von Beziehungen

Beziehungen und Rollen werden über die grafische Schnittstelle des Relationship Editors in Integration Designer definiert. Der Relationship Service speichert die Korrelationsdaten in Tabellen in der Beziehungsdatenbank in der Standarddatenquelle, die Sie beim Konfigurieren des Relationship Service angeben. In einer separaten Tabelle (manchmal auch als Teilnehmertabelle bezeichnet) werden Informationen zu jedem Teilnehmer in der Beziehung gespeichert. Der Relationship Service verwendet diese Beziehungstabellen zur Überwachung der zugehörigen anwendungsspezifischen Werte und zur Weitergabe aktualisierter Informationen an alle Lösungen.

Beziehungen sind Geschäftsartefakte, die innerhalb eines Projekts oder in einer gemeinsam genutzten Bibliothek implementiert werden. Bei der erstmaligen Implementierung füllt der Relationship Service die Daten.

Wenn Zuordnungen oder andere IBM Business Process Manager-Komponenten zur Laufzeit eine Beziehungsinstanz benötigen, werden die Instanzen der Beziehung je nach Szenario entweder aktualisiert oder abgerufen.

Zur Bearbeitung von Beziehungs- und Rolleninstanzdaten gibt es drei Möglichkeiten:

- Java-Snippet-Aufrufe der Relationship Service-APIs der IBM Business Process Manager-Komponente
- Beziehungsumsetzungen im Business-Objektzuordnungsservice von IBM Business Process Manager
- Relationship Manager-Tool

Detaillierte Hintergrund- und Taskinformationen zum Erstellen von Beziehungen, zum Angeben von Beziehungstypen und zur Verwendung des Relationship Editors finden Sie im Abschnitt Beziehungen erstellen.

Relationship Manager

Relationship Manager ist die Verwaltungsschnittstelle zur Verwaltung von Beziehungen. Der Zugriff erfolgt über die Relationship Manager-Seiten der Administrationskonsole.

Relationship Manager enthält eine grafische Benutzerschnittstelle für die Erstellung und Bearbeitung von Beziehungs- und Rollendaten zur Laufzeit. Beziehungsentitäten können auf allen Ebenen verwaltet werden: Beziehungsinstanzen, Rolleninstanzen, Attributdaten und Eigenschaftsdaten. Mit Relationship Manager können die folgenden Aktionen ausgeführt werden:

- Anzeigen einer Liste der Beziehungen im System sowie detaillierter Informationen für einzelne Beziehungen
- Verwaltung von Beziehungsinstanzen:
  - Abfragen von Beziehungsdaten zum Anzeigen von Teilmengen der Instanzdaten
  - Abfragen von Beziehungsdaten zum Anzeigen von Teilmengen der Instanzdaten mithilfe von Datenbanksichten
  - Anzeigen einer Liste von Beziehungsinstanzen, die mit einer einer Beziehungsabfrage übereinstimmen, sowie detaillierter Informationen zu einer Instanz
  - Ändern der Eigenschaftswerte für eine Beziehungsinstanz
  - Erstellen und Löschen von Beziehungsinstanzen
- Verwalten von Rollen und Rolleninstanzen
  - Anzeigen von Details zu einer Rolle oder Rolleninstanz
  - Bearbeiten der Eigenschaften von Rolleninstanzen
  - Erstellen und Löschen von Rolleninstanzen für eine Beziehung
  - Durchführen von Rollback-Operationen für Beziehungsinstanzdaten bis zu einem Zeitpunkt, zu dem die Daten garantiert zuverlässig sind
- Importieren von Daten aus einer vorhandenen statischen Beziehung in das System oder Exportieren von Daten aus einer vorhandenen statischen Beziehung in eine RI- oder CSV-Datei
- Entfernen von Beziehungsschemas und -daten aus dem Repository, wenn die darauf zugreifende Anwendung deinstalliert wird

Beziehungen in Network Deployment-Umgebungen

Beziehungen können ohne zusätzlichen Konfigurationsaufwand in Network Deployment-Umgebungen verwendet werden.

In Network Deployment-Umgebungen werden Beziehungen in einem Anwendungscluster installiert. Beziehungen sind dann innerhalb des Clusters sichtbar, und alle Server im Cluster haben Zugriff auf die Instanzdaten, die in der Beziehungsdatenbank gespeichert sind. Die Möglichkeit der Ausführung des Relationship Service in einer Network Deployment-Umgebung macht diese skalierbar und hoch verfügbar.

Mit Relationship Manager können Beziehungen in verschiedenen Clustern über eine zentrale Verwaltungsschnittstelle verwaltet werden. Die Verbindung zwischen Relationship Manager und einem Server in einem Cluster erfolgt über die Auswahl der zugehörigen Beziehungs-MBean.

Relationship Service-APIs

Beziehungen können innerhalb oder außerhalb von Business-Objektzuordnungen programmgesteuert über Relationship Service-APIs aufgerufen werden.

Es stehen drei API-Typen zur Verfügung:

- APIs zur Bearbeitung von Beziehungsinstanzen (einschließlich der direkten Erstellung, Aktualisierung und Löschung von Instanzdaten)
- APIs zur Unterstützung von Beziehungsmustern (einschließlich correlate(), correlateforeignKeyLookup)
- Muster für die Beziehungssuche (Such-APIs)

Enterprise Service Bus in IBM Business Process Manager

IBM Business Process Manager unterstützt die Integration von Anwendungsservices und enthält dieselben Funktionen wie WebSphere Enterprise Service Bus.

Services über einen Enterprise Service Bus verbinden

Mit einem Enterprise Service Bus (ESB) kann die Flexibilität einer SOA maximiert werden. Die Teilnehmer einer Serviceinteraktion werden mit dem ESB und nicht direkt miteinander verbunden.

Wenn der Serviceanforderer eine Verbindung zum ESB herstellt, übernimmt der ESB die Verantwortung für die Zustellung der entsprechenden Anforderungen. Diese Anforderungen werden mithilfe von Nachrichten einem Serviceanbieter zugestellt, der die erforderliche Funktion und Servicequalität bereitstellt. Der ESB erleichtert die Interaktionen zwischen Anforderer und Anbieter und adressiert abweichende Protokolle, Interaktionsmuster oder Servicekompetenzen. Ferner kann der ESB die Überwachung und Verwaltung aktivieren oder erweitern. Der ESB stellt Virtualisierungs- und Verwaltungsfunktionen bereit, die die zentralen Leistungsmerkmale der SOA implementieren und erweitern.

Der ESB zeichnet sich durch die folgenden Funktionen aus:

Position und Identität
: Die Teilnehmer müssen die Position oder Identität der anderen Teilnehmer nicht kennen. Ein Anforderer muss beispielsweise nicht wissen, dass eine Anforderung von mehreren Anbietern verarbeitet werden könnte; Serviceanbieter können ohne Unterbrechung hinzugefügt oder entfernt werden.

Interaktionsprotokoll
: Die Teilnehmer müssen nicht dasselbe Kommunikationsprotokoll und auch nicht denselben Interaktionsstil gemeinsam nutzen. Eine mit SOAP over HTTP gesendete Anforderung kann beispielsweise von einem Anbieter verarbeitet werden, der nur SOAP over JMS (Java Message Service) versteht.

Schnittstelle
: Anforderer und Anbieter müssen keine gemeinsame Schnittstelle verwenden. Der ESB gleicht Unterschiede durch das Transformieren von Anforderungs- und Antwortnachrichten in ein vom Anbieter erwartetes Format aus.

Servicequalität (Interaktionen)
: Teilnehmer oder Systemadministratoren deklarieren ihre Anforderungen an die Servicequalität, einschließlich der Autorisierung von Anforderungen, der Verschlüsselung und Entschlüsselung von Nachrichteninhalten, der automatischen Prüfung von Serviceinteraktionen, und geben an, wie die Anforderungen weitergeleitet werden sollen (z. B. Optimierung im Hinblick auf Geschwindigkeit oder Kosten).

Durch das Einfügen eines ESB zwischen die Teilnehmer kann die Interaktion zwischen den Teilnehmern über ein logisches Konstrukt namens *Mediation* moduliert werden. Mediationen arbeiten mit Nachrichten, die zwischen Anforderern und Anbietern noch nicht vollständig verarbeitet wurden. Mediationen können beispielsweise zum Suchen von Services mit bestimmten Merkmalen, die von einem Anforderer benötigt werden, oder zum Auflösen von Schnittstellendifferenzen zwischen Anforderern und Anbietern verwendet werden. Für komplexe Interaktionen können Mediationen sequenziell verkettet werden.

Unter Verwendung von Mediationen führt ein Enterprise Service Bus die folgenden Aktionen zwischen Anforderer und Service aus:

- *Weiterleitung* von Nachrichten zwischen Services. Ein Enterprise Service Bus bietet eine gemeinsame Kommunikationsinfrastruktur, die zum Verbinden von Services und daher auch zum Verbinden der von diesen Services dargestellten Geschäftsfunktionen verwendet werden kann, ohne dass der Programmierer komplexe Konnektivitätslogik entwickeln und verwalten muss.

- *Konvertieren* von Transportprotokollen zwischen Anforderer und Service. Ein Enterprise Service Bus bietet eine konsistente, standardisierte Möglichkeit zur Integration von Geschäftsfunktionen, die verschiedene IT-Standards verwenden. Dies ermöglicht die Integration von Geschäftsfunktionen, die normalerweise nicht kommunizieren könnten, z. B. das Verbinden von Anwendungen in Abteilungssilos oder das Aktivieren von Anwendungen in unterschiedlichen Unternehmen für die Teilnahme an Serviceinteraktionen.
- *Transformieren* von Nachrichtenformaten zwischen Anforderer und Service. Mit einem Enterprise Service Bus können Geschäftsfunktionen Informationen in unterschiedlichen Formaten austauschen, wobei der Bus sicherstellt, dass die an eine Geschäftsfunktion zugestellten Informationen das von der Anwendung geforderte Format aufweisen.
- *Verarbeiten* von Business-Ereignissen aus unterschiedlichen Quellen. Der Enterprise Service Bus unterstützt zur Bearbeitung von Serviceanforderungen zusätzlich zum Nachrichtenaustausch ereignisgesteuerte Interaktionen.

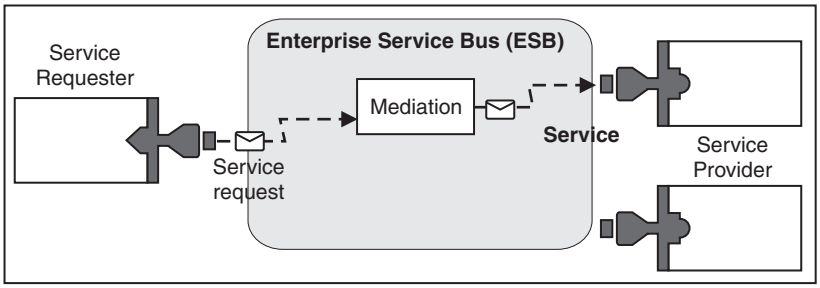


Abbildung 87. Ein Enterprise Service Bus. Der Enterprise Service Bus leitet Nachrichten zwischen Anwendungen weiter, die Anforderer oder Anbieter von Services sind. Der Bus konvertiert Transportprotokolle und transformiert Nachrichtenformate zwischen Anforderern und Anbietern. In dieser Abbildung verwendet jede Anwendung ein anderes Protokoll (dargestellt durch die verschiedenen geometrischen Formen der Connectors) und unterschiedliche Nachrichtenformate.

Wenn Sie einen Enterprise Service Bus verwenden, können Sie sich auf das Kerngeschäft konzentrieren, während die Computersysteme in den Hintergrund treten. Services können nach Bedarf geändert oder erweitert werden, um beispielsweise auf geänderte Geschäftsanforderungen zu reagieren oder zusätzliche Servicekapazität bzw. neue Funktionen hinzuzufügen. Sie können die erforderlichen Änderungen vornehmen, indem Sie den Bus neu konfigurieren, was wenig oder keine Auswirkungen auf die vorhandenen Services und Anwendungen hat, die den Bus verwenden.

Messaging-Infrastruktur für Enterprise Service Bus

IBM Business Process Manager enthält Enterprise Service Bus-Funktionen. IBM Business Process Manager unterstützt die Integration serviceorientierter, nachrichtenorientierter und ereignisgesteuerter Technologien zur Bereitstellung einer standardisierten Messaging-Infrastruktur in einem integrierten Enterprise Service Bus.

Die Enterprise Service-Funktionen, die Sie für Unternehmensanwendungen nutzen können, bieten eine Transportschicht und darüber hinaus Mediationsunterstützung zur Vereinfachung von Serviceinteraktionen. Der Enterprise Service Bus wurde für offene Standards und die serviceorientierte Architektur (Service-Oriented Architecture, SOA) entwickelt. Er basiert auf der leistungsfähigen Java EE-Infrastruktur und den zugehörigen Plattformservices, die durch IBM WebSphere Application Server Network Deployment bereitgestellt werden.

IBM Business Process Manager basiert auf derselben Technologie wie IBM WebSphere Enterprise Service Bus. Diese Funktionen sind Teil der zugrunde liegenden Funktionalität von IBM Business Process Manager; zur Nutzung dieser Funktionen ist keine zusätzliche Lizenz für WebSphere Enterprise Service Bus erforderlich.

Sie können jedoch zusätzliche Einzellizenzen für WebSphere Enterprise Service Bus in Ihrem Unternehmen implementieren, um die Konnektivität der auf IBM Business Process Manager basierenden Prozessintegrationslösungen zu erhöhen. Zum Beispiel könnte WebSphere Enterprise Service Bus näher bei einer SAP-Anwendung installiert werden, um einen IBM WebSphere Adapter for SAP aufzunehmen und SAP-Nachrichten umzusetzen, bevor diese Informationen über das Netz an einen von IBM Business Process Manager koordinierten Business-Prozess gesendet werden.

Messaging- oder Warteschlangenzielhosts

Ein Host für Messaging oder Warteschlangenziele stellt die Nachrichtenübertragungsfunktion innerhalb eines Servers bereit. Ein Server wird zum Messaging-Zielhost, wenn Sie ihn als Messaging-Ziel konfigurieren.

Eine Messaging-Steuerkomponente wird auf einem Server ausgeführt. Die Messaging-Steuerkomponente stellt Messaging-Funktionen und einen Verbindungspunkt bereit, an dem Anwendungen eine Verbindung zum Bus herstellen. Die asynchrone SCA-Kommunikation, JMS-Importe und -Exporte sowie die asynchrone interne Verarbeitung verwenden Nachrichtenwarteschlangen.

Die Implementierungsumgebung verbindet die Nachrichtenquelle im Zuge der Implementierung der Anwendungsmodule über den Bus mit dem Nachrichtenziel. Wenn Nachrichtenquelle und -ziel bekannt sind, können Sie leichter bestimmen, welcher Typ von Implementierungsumgebung erforderlich ist.

Anwendungen können persistente Daten in einem Datenspeicher speichern. Hierbei handelt es sich um eine Gruppe von Tabellen in einer Datenbank oder einem Schema oder aber in einem Dateispeicher. Die Messaging-Steuerkomponente verwendet eine Instanz einer JDBC-Datenquelle, um mit dieser Datenbank zu interagieren.

Konfigurieren Sie den Messaging-Zielhost, wenn Sie Ihre Implementierungsumgebung über die Option **Server** in der Administrationskonsole definieren oder den Server bei der Softwareinstallation als Zielhost definieren.

Datenspeicher:

Jede Messaging-Steuerkomponente kann einen Datenspeicher verwenden. Hierbei handelt es sich um eine Gruppe von Tabellen in einer Datenbank oder einem Schema, in denen persistente Daten gespeichert werden.

Alle Tabellen im Datenspeicher befinden sich im selben Datenbankschema. Jeder Datenspeicher kann in einer separaten Datenbank erstellt werden. Alternativ können Sie mehrere Datenspeicher in derselben Datenbank erstellen, wobei jeder Datenspeicher ein anderes Schema verwendet.

Eine Messaging-Steuerkomponente verwendet eine Instanz einer JDBC-Datenquelle, um mit der Datenbank zu interagieren, die den Datenspeicher für diese Messaging-Steuerkomponente enthält.

JDBC-Provider

Sie können JDBC-Provider für die Interaktion von Anwendungen mit relationalen Datenbanken verwenden.

JDBC-Provider werden von Anwendungen verwendet, damit diese mit relationalen Datenbanken interagieren können. Ein JDBC-Provider stellt die Implementierungsklasse eines JDBC-Treibers bereit, um den Zugriff auf einen bestimmten Datenbanktyp zu ermöglichen. Sie ordnen dem JDBC-Provider eine Datenquelle zu, um einen Verbindungspool für die verwendete Datenbank zu erstellen. Die Kombination aus JDBC-Provider und Datenquellenobjekten entspricht funktional der JCA-Verbindungsfactory (JCA = Java EE Connector Architecture), die Verbindungen zu nicht relationalen Datenbanken bereitstellt.

Weitere Informationen hierzu finden Sie in den Beispielen für die typische Konfiguration einer eigenständigen Umgebung und einer Implementierungsumgebung im vorherigen Abschnitt.

Weitere Informationen zu JDBC-Providern enthält das Information Center von WebSphere Application Server unter dem Stichwort 'JDBC-Provider'.

Service Integration Bus (SIBus) für IBM Business Process Manager

Ein SIBus ist ein verwalteter Kommunikationsmechanismus, der die Serviceintegration durch synchrones und asynchrones Messaging unterstützt. Ein Bus besteht aus verbundenen Messaging-Steuerkomponenten, die Busressourcen verwalten. Es handelt sich hierbei um eine der WebSphere Application Server-Technologien, auf denen IBM Business Process Manager basiert.

Ein einzelner SIBus und eine einzelne Messaging-Steuerkomponente verwenden standardmäßig dasselbe Datenbankschema wie die Produktdatenbank. Jeder Implementierungsumgebung verfügt über ihren eigenen Bus. Ein einzelner Bus hat den Namen **BPM.*deployment_environment_name*.Bus**.

Das Ziel eines Busses ist eine logische Adresse, der Anwendungen als Produzent und/oder als Konsument zugeordnet werden können. Das Ziel einer Warteschlange ist das Ziel eines Busses, das für das Punkt-zu-Punkt-Messaging verwendet wird.

Jeder Bus kann über ein oder mehrere Bus-Member verfügen, bei denen es sich entweder um Server oder Cluster handelt.

Als *Bustopologie* wird die physische Anordnung der Anwendungsserver, Messaging-Steuerkomponenten und WebSphere MQ-Warteschlangenmanager sowie das Muster der Busverbindungen und Links zwischen diesen Komponenten bezeichnet, das den Enterprise Service Bus ergibt.

Serviceanwendungen und -module

Ein Servicemodul ist ein SCA-Modul, das zur Laufzeit Services bereitstellt. Bei der Implementierung eines Servicemoduls in IBM Business Process Manager erstellen Sie eine zugeordnete Serviceanwendung, die als EAR-Datei gepackt wird.

Servicemodule sind die Basiseinheiten der Implementierung und können Komponenten, Bibliotheken und Staging-Module enthalten, die von der zugeordneten Serviceanwendung verwendet werden. Servicemodule verfügen über Exporte und optional auch über Importe, um die Beziehungen zwischen Modulen sowie Serviceanforderern und -anbietern zu definieren. WebSphere Process Server unterstützt Module für Business-Services und Mediationsmodule. Module und Mediationsmodule sind Typen von SCA-Modulen. Ein Mediationsmodul ermöglicht die Kommunikation zwischen Anwendungen durch das Transformieren des Serviceaufrufs in ein für das Ziel verständliches Format, wobei die Anforderung an das Ziel übergeben und das Ergebnis an den Absender zurückgegeben wird. Ein Modul für einen Business-Service implementiert die Logik eines Business-Prozesses. Ein Modul kann jedoch auch die Mediationslogik enthalten, die in ein Mediationsmodul gepackt werden kann.

Serviceanwendung implementieren

Der Prozess der Implementierung einer EAR-Datei, die eine Serviceanwendung enthält, entspricht dem Prozess der Implementierung jeder beliebigen EAR-Datei. Zur Implementierungszeit können Werte für Mediationsparameter geändert werden. Nach der Implementierung einer EAR-Datei mit einem SCA-Modul können Details zur Serviceanwendung und dem zugeordneten Modul angezeigt werden. Sie können feststellen, wie ein Servicemodul mit Serviceanforderern (über Exporte) und mit Serviceanbietern (über Importe) verbunden ist.

SCA-Moduldetails anzeigen

Die Servicemoduldetails, die Sie anzeigen können, hängen vom jeweiligen SCA-Modul ab. Sie umfassen die folgenden Attribute:

- Name des SCA-Moduls
- Beschreibung des SCA-Moduls

- Zugeordneter Anwendungsname
- Versionsangaben zum SCA-Modul, sofern das Modul versioniert ist
- SCA-Modulimporte:
  - Importschnittstellen sind abstrakte Definitionen, die beschreiben, wie ein SCA-Modul auf einen Service zugreift.
  - Importbindungen sind konkrete Definitionen, die den physischen Mechanismus angeben, mit dessen Hilfe ein SCA-Modul auf einen Service zugreift. Beispiel: SOAP/HTTP.
- SCA-Modulexporte:
  - Exportschnittstellen sind abstrakte Definitionen, die beschreiben, wie ein Serviceanforderer auf ein SCA-Modul zugreift.
  - Exportbindungen sind konkrete Definitionen, die den physischen Mechanismus angeben, mit dessen Hilfe ein Serviceanforderer auf ein SCA-Modul und indirekt auf einen Service zugreift.
- Eigenschaften des SCA-Moduls

Importe und Importbindungen

Importe definieren Interaktionen zwischen SCA-Modulen und Serviceanbietern. SCA-Module verwenden Importe, um Komponenten den Zugriff auf externe Services (Services außerhalb des SCA-Moduls) mit einer lokalen Darstellung zu ermöglichen. Importbindungen definieren die Art und Weise, wie der Zugriff auf einen externen Service erfolgt.

Wenn SCA-Module nicht auf externe Services zugreifen müssen, müssen sie über keine Importe verfügen. Mediationsmodule verfügen in der Regel über mindestens einen Import, der zum Übergeben von Nachrichten oder Anforderungen an die gewünschten Ziele verwendet wird.

Schnittstellen und Bindungen

Ein SCA-Modulimport benötigt mindestens eine Schnittstelle und verfügt über eine einzelne Bindung.

- Importschnittstellen sind abstrakte Definitionen, die eine Reihe von Operationen unter Verwendung von WSDL (Web Services Description Language) definieren. WSDL ist eine XML-Sprache zur Beschreibung von Web-Services. Ein SCA-Modul kann viele Importschnittstellen aufweisen.
- Importbindungen sind konkrete Definitionen, die den physischen Mechanismus angeben, der von SCA-Modulen für den Zugriff auf einen externen Service verwendet werden.

Unterstützte Importbindungen

IBM Business Process Manager unterstützt die folgenden Importbindungen:

- SCA-Bindungen verbinden SCA-Module mit anderen SCA-Modulen. SCA-Bindungen werden auch als Standardbindungen bezeichnet.
- Web-Service-Bindungen ermöglichen Komponenten das Aufrufen von Web-Services. Die unterstützten Protokolle sind SOAP1.1/HTTP, SOAP1.2/HTTP und SOAP1.1/JMS.

  Sie können eine SOAP1.1/HTTP- oder SOAP1.2/HTTP-Bindung basierend auf JAX-WS (Java API for XML Web Services) verwenden, die Interaktionen mit Services ermöglicht, die mit Dokument- oder literalen RPC-Bindungen arbeiten, und Aufrufe mithilfe von JAX-WS-Handlern anpasst. Für die Interaktion mit Services, die eine RPC-codierte Bindung verwenden, und für Fälle, in denen JAX-RPC-Handler zum Anpassen von Aufrufen verwendet werden müssen, wird eine separate SOAP1.1/HTTP-Bindung bereitgestellt.
- Mit HTTP-Bindungen kann über das HTTP-Protokoll auf Anwendungen zugegriffen werden.
- EJB-Importbindungen (EJB - Enterprise JavaBeans) ermöglichen SCA-Komponenten das Aufrufen von Services, die von der Java EE-Geschäftslogik auf einem Java EE-Server bereitgestellt werden.
- EIS-Bindungen stellen Konnektivität zwischen SCA-Komponenten und einem externen EIS (Enterprise Information System, unternehmensweites Informationssystem) bereit. Diese Kommunikation wird durch die Verwendung von Ressourcenadaptern ermöglicht.

- JMS 1.1-Bindungen (JMS - Java Message Service) ermöglichen Interoperabilität mit dem Standard-Messaging-Provider von WebSphere Application Server. JMS kann verschiedene Transporttypen nutzen, einschließlich TCP/IP und HTTP oder HTTPS. Die JMS-Nachrichtenklasse und die fünf zugehörigen Subtypen (Text, Bytes, Object, Stream und Map) werden automatisch unterstützt.
- Generische JMS-Bindungen ermöglichen Interoperabilität mit JMS-Providern anderer Anbieter, die über JMS-ASF (Application Server Facility) mit WebSphere Application Server integriert werden.
- WebSphere MQ-JMS-Bindungen ermöglichen Interoperabilität mit WebSphere MQ-basierten JMS-Providern. Die JMS-Nachrichtenklasse und die fünf zugehörigen Subtypen (Text, Bytes, Object, Stream und Map) werden automatisch unterstützt. Wenn Sie WebSphere MQ als JMS-Provider nutzen möchten, müssen Sie WebSphere MQ-JMS-Bindungen verwenden.
- WebSphere MQ-Bindungen ermöglichen Interoperabilität mit WebSphere MQ. WebSphere MQ-Bindungen können nur mit fernen Warteschlangenmanagern über eine WebSphere MQ-Clientverbindung verwendet werden; die Verwendung mit lokalen Warteschlangenmanagern ist nicht möglich. Verwenden Sie WebSphere MQ-Bindungen, wenn Sie mit nativen WebSphere MQ-Anwendungen kommunizieren möchten.

Dynamischer Aufruf von Services

Services können über eine unterstützte Importbindung aufgerufen werden. Ein Service befindet sich normalerweise an einem Endpunkt, der im Import angegeben ist. Dieser Endpunkt wird als statischer Endpunkt bezeichnet. Durch das Überschreiben des statischen Endpunkts kann ein anderer Service aufgerufen werden. Durch das dynamische Überschreiben von statischen Endpunkten können über unterstützte Importbindungen Services an anderen Endpunkten aufgerufen werden. Ferner kann durch das dynamische Aufrufen von Services auch ein Service aufgerufen werden, für den die unterstützte Importbindung keinen statischen Endpunkt aufweist.

Zum Angeben des Protokolls und der entsprechenden Konfiguration für einen dynamischen Aufruf wird ein Import mit einer zugeordneten Bindung verwendet. Der für den dynamischen Aufruf verwendete Import kann mit der aufrufenden Komponente verknüpft oder zur Laufzeit dynamisch ausgewählt werden.

Bei Web-Service- und SCA-Aufrufen ist es außerdem möglich, einen dynamischen Aufruf ohne einen Import auszuführen, wobei das zugehörige Protokoll und die Konfiguration aus der Endpunkt-URL abgeleitet werden. Der Zieltyp des Aufrufs wird über die Endpunkt-URL ermittelt. Bei Verwendung eines Imports muss die URL mit dem Protokoll der Importbindung kompatibel sein.

- Eine SCA-URL weist auf den Aufruf eines anderen SCA-Moduls hin.
- Eine HTTP- oder JMS-URL weist standardmäßig auf den Aufruf eines Web-Service hin; für diese URLs kann ein zusätzlicher Wert für den Bindungstyp angegeben werden, der darauf hinweist, dass die URL einen Aufruf mittels einer HTTP- oder JMS-Bindung darstellt.
- Für eine Web-Service-HTTP-URL wird standardmäßig SOAP 1.1 verwendet; es kann auch ein Bindungstypwert angegeben werden, der auf die Verwendung von SOAP 1.2 hinweist.

Exporte und Exportbindungen

Exporte definieren Interaktionen zwischen SCA-Modulen und Serviceanforderern. SCA-Module verwenden Exporte, um Services bereitzustellen. Exportbindungen definieren die Art und Weise, wie der Zugriff auf ein SCA-Modul durch einen Serviceanforderer erfolgt.

Schnittstellen und Bindungen

Ein SCA-Modulexport benötigt mindestens eine Schnittstelle.

- Exportschnittstellen sind abstrakte Definitionen, die eine Reihe von Operationen unter Verwendung von WSDL (Web Services Description Language) definieren. WSDL ist eine XML-Sprache zur Beschreibung von Web-Services. Ein SCA-Modul kann viele Exportschnittstellen aufweisen.

- Exportbindungen sind konkrete Definitionen, die den physischen Mechanismus angeben, den der Serviceanforderer für den Zugriff auf einen Service verwendet. Normalerweise ist für einen SCA-Modulexport eine Bindung angegeben. Ein Export ohne angegebene Bindung wird von der Laufzeit als Export mit einer SCA-Bindung interpretiert.

Unterstützte Exportbindungen

IBM Business Process Manager unterstützt die folgenden Exportbindungen:

- SCA-Bindungen verbinden SCA-Module mit anderen SCA-Modulen. SCA-Bindungen werden auch als Standardbindungen bezeichnet.
- Mit Web-Service-Bindungen können Exporte als Web-Services aufgerufen werden. Die unterstützten Protokolle sind SOAP1.1/HTTP, SOAP1.2/HTTP und SOAP1.1/JMS.

  Sie können eine SOAP1.1/HTTP- oder SOAP1.2/HTTP-Bindung basierend auf JAX-WS (Java API for XML Web Services) verwenden, die Interaktionen mit Services ermöglicht, die mit Dokument- oder literalen RPC-Bindungen arbeiten, und Aufrufe mithilfe von JAX-WS-Handlern anpasst. Für die Interaktion mit Services, die eine RPC-codierte Bindung verwenden, und für Fälle, in denen JAX-RPC-Handler zum Anpassen von Aufrufen verwendet werden müssen, wird eine separate SOAP1.1/HTTP-Bindung bereitgestellt.
- Über HTTP-Bindungen kann mit dem HTTP-Protokoll auf Exporte zugegriffen werden.
- Durch EJB-Exportbindungen (EJB - JavaBeans) können SCA-Komponenten als EJBs zugänglich gemacht werden, sodass die Java EE-Geschäftslogik SCA-Komponenten aufrufen kann, die normalerweise nicht verfügbar wären.
- EIS-Bindungen stellen Konnektivität zwischen SCA-Komponenten und einem externen EIS (Enterprise Information System, unternehmensweites Informationssystem) bereit. Diese Kommunikation wird durch die Verwendung von Ressourcenadaptern ermöglicht.
- JMS 1.1-Bindungen (JMS - Java Message Service) ermöglichen Interoperabilität mit dem Standard-Messaging-Provider von WebSphere Application Server. JMS kann verschiedene Transporttypen nutzen, einschließlich TCP/IP und HTTP oder HTTPS. Die JMS-Nachrichtenklasse und die fünf zugehörigen Subtypen (Text, Bytes, Object, Stream und Map) werden automatisch unterstützt.
- Generische JMS-Bindungen ermöglichen Interoperabilität mit JMS-Providern anderer Anbieter, die über JMS-ASF (Application Server Facility) mit WebSphere Application Server integriert werden.
- WebSphere MQ-JMS-Bindungen ermöglichen Interoperabilität mit WebSphere MQ-basierten JMS-Providern. Die JMS-Nachrichtenklasse und die fünf zugehörigen Subtypen (Text, Bytes, Object, Stream und Map) werden automatisch unterstützt. Wenn Sie WebSphere MQ als JMS-Provider nutzen möchten, müssen Sie WebSphere MQ-JMS-Bindungen verwenden.
- WebSphere MQ-Bindungen ermöglichen Interoperabilität mit WebSphere MQ. Für die Verbindung zu einem MQ-Warteschlangenmanager auf einem fernen System wird eine ferne Verbindung (Clientverbindung) verwendet. Eine lokale Verbindung (Bindungsverbindung) ist eine Direktverbindung zu WebSphere MQ. Diese kann nur bei einer Verbindung zu einem MQ-Warteschlangenmanager auf demselben System verwendet werden. WebSphere MQ lässt beide Verbindungstypen zu, MQ-Bindungen unterstützen jedoch nur ferne Verbindungen (Clientverbindungen).

Mediationsmodule

Mediationsmodule sind SCA-Module, die das Format, den Inhalt oder das Ziel von Serviceanforderungen ändern können.

Mediationsmodule arbeiten mit Nachrichten, die zwischen Serviceanforderern und Serviceanbietern noch nicht vollständig verarbeitet wurden. Sie können Nachrichten an unterschiedliche Serviceanbieter weiterleiten und den Nachrichteninhalt oder die Nachrichtenstruktur korrigieren. Von Mediationsmodulen können Funktionen wie beispielsweise Nachrichtenprotokollierung und Fehlerverarbeitung bereitgestellt werden, die auf Ihre Bedürfnisse zugeschnitten sind.

Bestimmte Aspekte von Mediationsmodulen können über die Administrationskonsole geändert werden, ohne das Modul erneut implementieren zu müssen.

Komponenten von Mediationsmodulen

Mediationsmodule enthalten die folgenden Elemente:

- Importe, die Interaktionen zwischen SCA-Modulen und Serviceanbietern definieren. Mithilfe dieser Importe können SCA-Module externe Services wie lokale Services aufrufen. Importe von Mediationsmodulen können angezeigt und die Bindung geändert werden.
- Exporte, die Interaktionen zwischen SCA-Modulen und Serviceanforderern definieren. Sie ermöglichen SCA-Modulen das Anbieten von Services und definieren die externen Schnittstellen (Zugriffspunkte) eines SCA-Moduls. Exporte von Mediationsmodulen können angezeigt werden.
- SCA-Komponenten, die Bausteine für SCA-Module (z. B. Mediationsmodule) sind. Mit Integration Designer können SCA-Module und Komponenten grafisch erstellt und angepasst werden. Nach der Implementierung können über die Administrationskonsole bestimmte Aspekte eines Mediationsmoduls angepasst werden, ohne das Modul erneut implementieren zu müssen.

  Normalerweise enthalten Mediationsmodule einen bestimmten Typ von SCA-Komponente, der als *Mediationsablaufkomponente* bezeichnet wird. Mediationsablaufkomponenten definieren Mediationsabläufe.

  Eine Mediationsablaufkomponente kann kein, ein oder mehrere Mediationsbasiselemente enthalten. IBM Business Process Manager unterstützt eine Gruppe von Mediationsbasiselementen, die Funktionalität für Nachrichtenweiterleitung und -umsetzung bereitstellen. Wird zusätzliche Flexibilität für Mediationsbasiselemente benötigt, können Sie mit dem angepassten Mediationsbasiselement angepasste Logik aufrufen.

  Mediationsmodule, die keine Mediationsablaufkomponenten enthalten, sind dafür vorgesehen, Serviceanforderungen aus einem Protokoll in einen anderes zu transformieren. Es könnte beispielsweise eine Serviceanforderung mit SOAP/JMS erstellt werden, die vor der Weiterleitung jedoch in SOAP/HTTP transformiert werden muss.

  Anmerkung: In IBM Business Process Manager können Mediationsmodule angezeigt und gewisse Änderungen vorgenommen werden. Die SCA-Komponenten in einem Modul können über IBM Business Process Manager jedoch nicht angezeigt oder geändert werden. Verwenden Sie Integration Designer zum Anpassen von SCA-Komponenten.

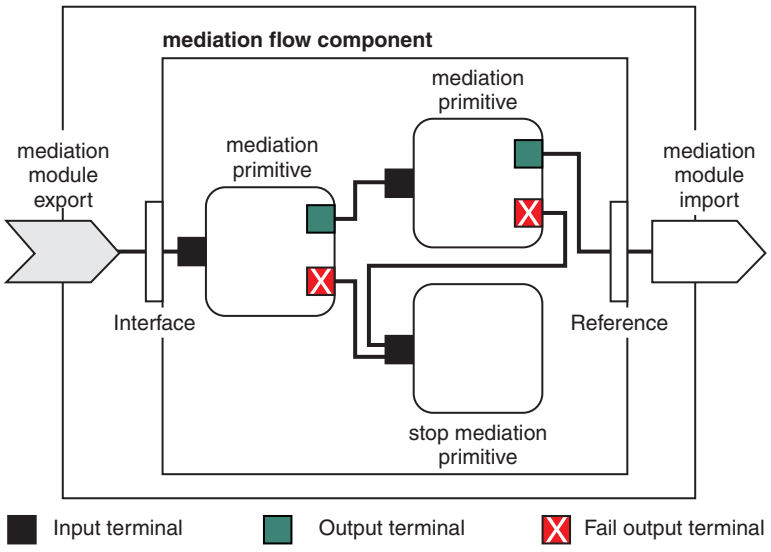


Abbildung 88. Vereinfachtes Beispiel eines Mediationsmoduls. Das Mediationsmodul enthält eine Mediationsablaufkomponente, die Mediationsbasiselemente enthält.

- Eigenschaften

Mediationsbasiselemente verfügen über Eigenschaften; einige dieser Eigenschaften können in der Administrationskonsole als zusätzliche Eigenschaften eines SCA-Moduls angezeigt werden.

Der Integrationsentwickler muss die Eigenschaften hochstufen, damit sie in der Administrationskonsole von IBM Business Process Manager angezeigt werden. Bestimmte Eigenschaften eignen sich für eine Verwaltungskonfiguration; Integration Designer bezeichnet diese Eigenschaften als hochstufbare Eigenschaften, da sie vom Integrationszyklus auf den Administrationszyklus hochgestuft werden können. Andere Eigenschaften sind nicht für die Verwaltungskonfiguration geeignet, da sich eine Änderung dieser Eigenschaften so auf den Mediationsablauf auswirken kann, dass das Mediationsmodul erneut implementiert werden muss. In Integration Designer sind die Eigenschaften aufgeführt, die unter den hochgestuften Eigenschaften eines Mediationsbasiselement hochgestuft werden können.

Werte von hochgestufte Eigenschaften können über die Administrationskonsole von IBM Business Process Manager geändert werden, ohne das Mediationsmodul erneut implementieren oder den Server/das Modul erneut starten zu müssen.

Im Allgemeinen werden Eigenschaftsänderungen von Mediationsabläufen umgehend verwendet. Wenn die Eigenschaftsänderungen jedoch in einer Deployment Manager-Zelle erfolgen, werden diese Änderungen bei der Synchronisation eines Knotens auf diesem Knoten wirksam. Mediationsabläufe, die noch ausgeführt werden, verwenden weiterhin die vorherigen Werte.

Anmerkung: Über die Administrationskonsole können nur Eigenschaftswerte und keine Eigenschaftsgruppen, -namen oder -typen geändert werden. Zur Änderung von Eigenschaftsgruppen, -namen oder -typen muss Integration Designer verwendet werden.

- Ein Mediationsmodul oder eine abhängige Bibliothek kann auch Unterabläufe definieren. In einen Unterablauf ist eine Gruppe aus Mediationsbasiselementen eingebunden, die als wieder verwendbarer Teil der Integrationslogik miteinander verknüpft sind. Einem Mediationsablauf kann ein Basiselement hinzugefügt werden, um einen Unterablauf aufzurufen.

Mediationsmodule implementieren

Mediationsmodule werden mit Integration Designer erstellt und im Allgemeinen in Form einer EAR-Datei in IBM Business Process Manager implementiert.

Zur Implementierungszeit können die Werte von hochgestuften Eigenschaften geändert werden.

Sie können ein Mediationsmodul aus Integration Designer exportieren und Integration Designer dazu veranlassen, das Mediationsmodul in eine JAR-Datei und die JAR-Datei in eine EAR-Datei zu packen. Anschließend kann die EAR-Datei durch Installation einer neuen Anwendung über die Administrationskonsole implementiert werden.

Mediationsmodule können als eine Entität betrachtet werden. SCA-Module werden jedoch durch eine Reihe von XML-Dateien definiert, die in einer JAR-Datei gespeichert sind.

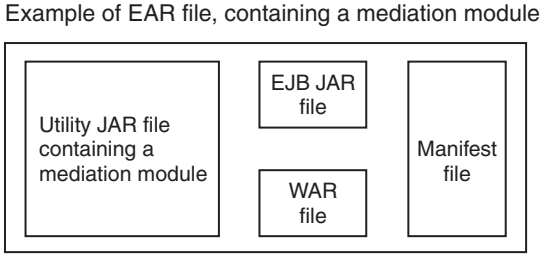


Abbildung 89. Vereinfachtes Beispiel einer EAR-Datei mit einem Mediationsmodul. Die EAR-Datei enthält JAR-Dateien. Die Dienstprogramm-JAR-Datei enthält ein Mediationsmodul.

Mediationsbasiselemente

Mediationsablaufkomponenten arbeiten mit Nachrichtenflüssen zwischen Servicekomponenten. Die Funktionalität einer Mediationskomponente wird durch *Mediationsbasiselemente* implementiert, die Standard-Serviceimplementierungstypen implementieren.

Eine Mediationsablaufkomponente verfügt über mindestens einen Ablauf (z. B. einen Ablauf für Anforderungen und einen Ablauf für Antworten).

IBM Business Process Manager unterstützt eine Gruppe von bereitgestellten Mediationsbasiselementen, die Standardmediationsfunktionen für Mediationsmodule oder Module in IBM Business Process Manager implementieren. Wenn Sie besondere Mediationsfunktionen benötigen, können Sie eigene angepasste Mediationsbasiselemente entwickeln.

Ein Mediationsbasiselement definiert eine eingehende Operation, die Nachrichten verarbeitet oder ausführt, die durch Servicenachrichtenobjekte (Service Message Objects, SMOs) dargestellt werden. Ein Mediationsbasiselement kann auch abgehende Operationen definieren, die Nachrichten an andere Komponenten oder Module senden.

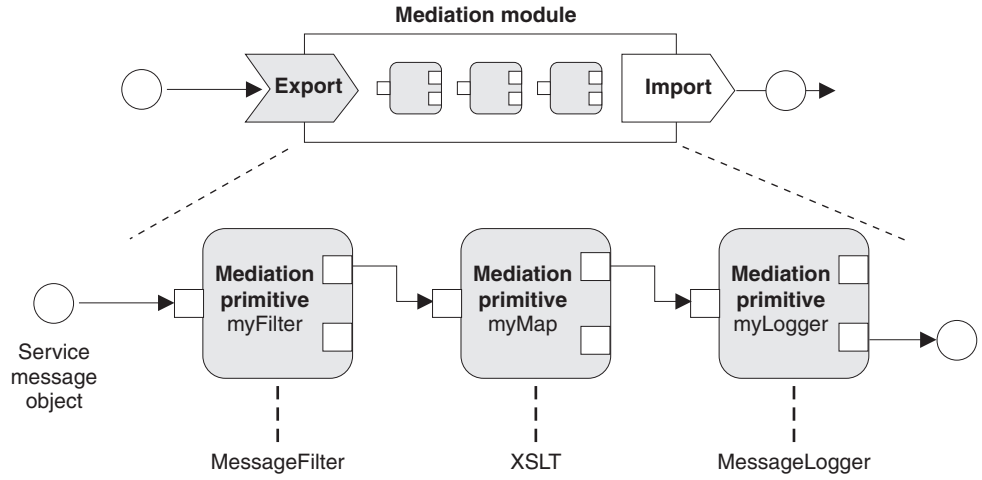


Abbildung 90. Mediationsmodul mit drei Mediationsbasiselementen

Mit Integration Designer können Mediationsbasiselemente konfiguriert und die zugehörigen Eigenschaften definiert werden. Einige dieser Eigenschaften können durch Hochstufen für den Laufzeitadministrator sichtbar gemacht werden. Hochstufbare Eigenschaften von Mediationsbasiselementen können auch dynamische Eigenschaften sein. Eine dynamische Eigenschaft kann zur Laufzeit mithilfe einer Richtliniendatei überschrieben werden.

Integration Designer ermöglicht ferner das grafische Modellieren und Assemblieren von Mediationsablaufkomponenten aus Mediationsbasiselementen sowie das Assemblieren von Mediationsmodulen oder Modulen aus Mediationsablaufkomponenten. Die Administrationskonsole bezeichnet Mediationsmodule und Module als SCA-Module.

Darüber hinaus ermöglicht Integration Designer das Definieren von Unterabläufen in Modulen oder den zugehörigen Bibliotheken. Ein Unterablauf kann alle Mediationsbasiselemente mit Ausnahme des Mediationsbasiselements für die Richtlinienauflösung enthalten. Ein Unterablauf wird über einen Anforderungs- oder Antwortablauf oder einen anderen Unterablauf mithilfe des Mediationsbasiselements für Unterabläufe aufgerufen. Eigenschaften, die aus Mediationsbasiselementen in einem Unterablauf hochgestuft wurden, werden in den Mediationsbasiselementen für Unterabläufe als Eigenschaften verfügbar gemacht. Diese können dann erneut hochgestuft werden, bis sie die Modulstufe erreichen, auf der sie vom Laufzeitadministrator geändert werden können.

Unterstützte Mediationsbasiselemente

Die folgenden Mediationsbasiselemente werden von IBM Business Process Manager unterstützt:

Business-Objektzuordnung
Transformiert Nachrichten.

- Definiert Nachrichtenumsetzungen mithilfe einer Business-Objektzuordnung, die wieder verwendet werden kann.
- Ermöglicht die grafische Definition von Nachrichtenumsetzungen unter Verwendung des Editors für Business-Objektzuordnungen.
- Kann den Inhalt einer Nachricht ändern.
- Kann einen Eingabenachrichtentyp in einen anderen Ausgabenachrichtentyp transformieren.

Angepasste Mediation
Ermöglicht die Implementierung eigener Mediationslogik in Java-Code. Das angepasste Mediationsbasiselement kombiniert die Flexibilität benutzerdefinierter Mediationsbasiselemente mit der Einfachheit vordefinierter Mediationsbasiselemente. Durch die folgenden Aktionen können komplexe Transformationen und Weiterleitungsmuster erstellt werden:

- Erstellen von Java-Code.
- Erstellen eigener Eigenschaften.
- Hinzufügen neuer Terminals.

Sie können zwar einen Service über das angepasste Mediationsbasiselement aufrufen, zum Aufrufen von Services wurde jedoch das Mediationsbasiselement für Serviceaufrufe (Service Invoke) konzipiert, das zusätzliche Funktionalität (z. B. Wiederholungen) bereitstellt.

Datenhandler
Ermöglicht die Transformation von Nachrichtenteilen. Der Datenhandler wird zum Konvertieren eines Nachrichtenelements aus einem physischen Format in eine logische Struktur oder aus einer logischen Struktur in ein physisches Format verwendet. Die primäre Funktion des Basiselements besteht darin, ein physisches Format, z. B. eine Textzeichenfolge in einem JMS-Textnachrichtenobjekt, in eine logische Business-Objektstruktur zu konvertieren und anschließend zu rekonvertieren. Diese Mediation wird häufig für die folgenden Aktionen verwendet:

- Transformation eines Abschnitts der Eingabenachricht aus einer definierten Struktur in eine andere. Beispiel: Ein SMO enthält einen durch Kommas begrenzten Zeichenfolgewert, und Sie möchten diesen Wert in ein bestimmtes Business-Objekt (BO) parsen.
- Ändern des Nachrichtentyps. Beispiel: Ein JMS-Export wurde für die Verwendung einer JMS-Datenbindung (Basistyp) konfiguriert und innerhalb des Mediationsmoduls legt der Integrationsentwickler fest, dass der Inhalt um eine bestimmte BO-Struktur erweitert werden soll.

Datenbanksuche
Ändert Nachrichten anhand von Informationen aus einer vom Benutzer bereitgestellten Datenbank.

- Sie müssen eine Datenbank, eine Datenquelle und die Einstellungen zur Serverauthentifizierung definieren, die das Mediationsbasiselement für die Datenbanksuche (Database Lookup) verwenden kann. Die Administrationskonsole unterstützt Sie bei dieser Aufgabe.
- Das Mediationsbasiselement für die Datenbanksuche kann nur aus einer Tabelle lesen.
- Die angegebene Schlüsselspalte muss einen eindeutigen Wert enthalten.
- Die Daten in den Wertspalten müssen entweder einen einfachen XML-Schematyp oder einen XML-Schematyp aufweisen, der einen einfachen XML-Schematyp erweitert.

Endpunktsuche
Ermöglicht die dynamische Weiterleitung von Anforderungen durch die Suche nach Serviceendpunkten in einem Repository.

- Serviceendpunktinformationen werden aus WebSphere Service Registry and Repository (WSRR) abgerufen. Die WSRR-Registry kann lokal oder fern sein.
- Registry-Änderungen können über die WSRR-Administrationskonsole durchgeführt werden.
- IBM Business Process Manager muss wissen, welche Registry verwendet werden soll. Deshalb müssen über die IBM Business Process Manager-Administrationskonsole WSRR-Zugriffsdefinitionen erstellt werden.

Ereignisemitter

Erweitert die Überwachung durch die Möglichkeit, Ereignisse aus einer Mediationsablaufkomponente heraus zu senden.

- Die Mediationsaktion kann durch Inaktivieren des Kontrollkästchens ausgesetzt werden.
- Ereignisse des Ereignisemitters können mit dem CBE-Browser (CBE - Common Base Events) in IBM Business Process Manager angezeigt werden.
- Aus Leistungsgründen sollten Ereignisse nur an signifikanten Punkten in einem Mediationsablauf gesendet werden.
- Sie können die Teile der Nachricht definieren, die im Ereignis enthalten sind.
- Die Ereignisse werden in Form von Common Base Events an einen Common Event Infrastructure-Server gesendet.
- Ereigniskonsumenten müssen die Struktur der Common Base Events verstehen, um die Informationen des Ereignisemitters vollständig nutzen zu können. Die Common Base Events verfügen über ein Gesamtschema, das jedoch die anwendungsspezifischen Daten nicht modelliert, die in den erweiterten Datenelementen enthalten sind. Zur Modellierung der erweiterten Datenelemente generieren die Integration Designer-Tools für jedes konfigurierte Mediationsbasiselement 'Ereignisemitter' eine Definitionsdatei für den Common Event Infrastructure-Ereigniskatalog. Die Definitionsdateien für den Ereigniskatalog sind Exportartefakte, die zur Ihrer Unterstützung dienen; sie werden von Integration Designer oder der IBM Business Process Manager-Laufzeit nicht verwendet. Ziehen Sie die Definitionsdateien für den Ereigniskatalog zu Rate, wenn Sie Anwendungen zur Verarbeitung von Ereignissen des Ereignisemitters erstellen.
- In IBM Business Process Manager können Sie weitere Überwachungsoptionen angeben. Sie können beispielsweise Ereignisse überwachen, die von Importen und Exporten ausgegeben werden.

Fehlgeschlagen

Stoppt einen bestimmten Pfad im Ablauf und generiert eine Ausnahmebedingung.

Eingabefächerung

Hilft beim Aggregieren (Zusammenfassen) von Nachrichten.

- Kann nur in Verbindung mit dem Mediationsbasiselement 'Ausgabefächerung' verwendet werden.
- Zusammen ermöglichen die Mediationsbasiselemente für die Eingabe- und Ausgabefächerung das Aggregieren von Daten in einer Ausgabenachricht.
- Das Mediationsbasiselement 'Eingabefächerung' empfängt Nachrichten, bis ein Entscheidungspunkt erreicht wird; dann wird eine Nachricht ausgegeben.
- Der gemeinsam genutzte Kontext sollte für die Aggregationsdaten verwendet werden.

Ausgabefächerung

Hilft beim Aufteilen und Aggregieren (Zusammenfassen) von Nachrichten.

- Zusammen ermöglichen die Mediationsbasiselemente für die Eingabe- und Ausgabefächerung das Aggregieren von Daten in einer Ausgabenachricht.
- Im Iterationsmodus kann mit dem Mediationsbasiselement 'Ausgabefächerung' eine einzelne Eingabenachricht durchlaufen werden, die ein sich wiederholendes Element enthält. Für jedes Vorkommen des sich wiederholenden Elements wird eine Nachricht gesendet.
- Der gemeinsam genutzte Kontext sollte für die Aggregationsdaten verwendet werden.

HTTP-Header-Setter

Stellt einen Mechanismus zur Verwaltung von Headern in HTTP-Nachrichten bereit.

- Kann HTTP-Nachrichtenheader erstellen, definieren, kopieren oder löschen.
- Kann mehrere Aktionen zur Änderung mehrerer HTTP-Header definieren.

Zuordnung

Transformiert Nachrichten.

- Ermöglicht die Ausführung von XSL-Transformationen (XSL - Extensible Stylesheet Language) oder Transformationen von Geschäftsobjektzuordnungen.
- Nachrichten werden mithilfe einer XSLT 1.0- oder XSLT 2.0-Transformation oder einer Geschäftsobjektzuordnungstransformation transformiert. Die XSL-Transformationen arbeiten mit einer XML-Serialisierung der Nachricht, während die Transformation von Geschäftsobjektzuordnungen mit Service Data Objects (SDOs) arbeitet.

Nachrichtenelementsetter

Stellt einen einfachen Mechanismus zum Definieren von Nachrichteninhalten bereit.

- Kann Nachrichtenelemente ändern, hinzufügen oder löschen.
- Der Nachrichtentyp wird nicht geändert.
- Die Daten in den Wertspalten müssen entweder einen einfachen XML-Schematyp oder einen XML-Schematyp aufweisen, der einen einfachen XML-Schematyp erweitert.

Nachrichtenfilter

Leitet Nachrichten basierend auf dem Nachrichteninhalt an verschiedene Pfade weiter.

- Die Mediationsaktion kann durch Inaktivieren des Kontrollkästchens ausgesetzt werden.

Nachrichtenprotokollfunktion

Protokolliert Nachrichten in einer relationalen Datenbank oder über eine eigene angepasste Protokollfunktion. Die Nachrichten werden in XML gespeichert, sodass eine Nachverarbeitung der Daten durch XML-konforme Anwendungen möglich ist.

- Die Mediationsaktion kann durch Inaktivieren des Kontrollkästchens ausgesetzt werden.
- Das Schema der rationalen Datenbank (Tabellenstruktur) wird durch IBM definiert.
- Das Mediationsbasiselement für die Nachrichtenprotokollfunktion verwendet standardmäßig die Common-Datenbank. Die Laufzeitumgebung ordnet die Datenquelle unter **jdbc/mediation/messageLog** der Common-Datenbank zu.
- Zur Anpassung des Verhaltens der angepassten Protokollfunktion können Sie Implementierungsklassen für Handler definieren. Optional können Implementierungsklassen für Formatierungsprogramme und/oder Filter angegeben werden, um das Verhalten der angepassten Protokollfunktion anzupassen.

MQ-Header-Setter

Stellt einen Mechanismus zur Verwaltung von Headern in MQ-Nachrichten bereit.

- Kann MQ-Nachrichtenheader erstellen, definieren, kopieren oder löschen.
- Kann mehrere Aktionen zur Änderung mehrerer MQ-Header definieren.

Richtlinienauflösung

Ermöglicht die dynamische Konfiguration von Anforderungen durch die Suche nach Serviceendpunkten und zugehörigen Richtliniendateien in einem Repository.

- Mit einer Richtliniendatei können die hochgestuften Eigenschaften anderer Mediationsbasiselemente dynamisch überschrieben werden.
- Serviceendpunkt- und Richtlinieninformationen werden aus WebSphere Service Registry and Repository (WSRR) abgerufen. Die WSRR-Registry kann lokal oder fern sein.
- Registry-Änderungen können über die WSRR-Administrationskonsole durchgeführt werden.

- IBM Business Process Manager muss wissen, welche Registry verwendet werden soll. Deshalb müssen über die IBM Business Process Manager-Administrationskonsole WSRR-Zugriffsdefinitionen erstellt werden.

Serviceaufruf
Ruft einen Service aus einem Mediationsablauf heraus auf, anstatt bis zum Ende des Mediationsablaufs zu warten und den Callout-Mechanismus zu verwenden.

- Wenn der Service einen Fehler zurückgibt, kann der Aufruf für diesen Service wiederholt oder ein anderer Service aufgerufen werden.
- Das Mediationsbasiselement 'Serviceaufruf' ist ein leistungsstarkes Mediationsbasiselement, das für einfache Serviceaufrufe allein oder für komplexe Mediationen in Kombination mit anderen Mediationsbasiselementen verwendet werden kann.

Nachrichtentyp festlegen
Mit diesem Element können während der Integrationsentwicklung schwach typisierte Nachrichtenfelder wie stark typisierte Felder behandelt werden. Ein Feld ist schwach typisiert, wenn es mehr als einen Datentyp enthalten kann. Ein Feld ist stark typisiert, wenn der Typ und die interne Struktur des Felds bekannt sind.

- Zur Laufzeit kann mit dem Mediationsbasiselement 'Nachrichtentyp festlegen' überprüft werden, ob der Inhalt einer Nachricht mit dem erwarteten Datentypen übereinstimmt.

SOAP-Header-Setter
Stellt einen Mechanismus zur Verwaltung von Headern in SOAP-Nachrichten bereit.

- Kann SOAP-Nachrichtenheader erstellen, definieren, kopieren oder löschen.
- Kann mehrere Aktionen zur Änderung mehrerer SOAP-Header definieren.

Stoppen
Stoppt einen bestimmten Pfad im Ablauf und generiert keine Ausnahmebedingung.

Typfilter
Ermöglicht die typabhängige Weiterleitung von Nachrichten an verschiedene Pfade in einem Ablauf.

WebSphere eXtreme Scale - Abrufen
Sie können Informationen aus der Umgebung eines eXtreme Scale-Server-Cache abrufen.

- Sie können Werte im Cache suchen und diese Werte mithilfe eines Schlüssels als Elemente in der Nachricht speichern.
- Durch Kombinieren der eXtreme Scale-Mediationsbasiselemente für Speichern (Store) und Abrufen (Retrieve) können Sie die Antwort von einem Back-End-System in den Cache stellen. Zukünftige Anforderungen benötigen keinen Zugriff auf das betreffende Back-End-System.
- Sie müssen eXtreme Scale-Definitionen mithilfe der WebSphere ESB-Administrationskonsole erstellen, um angeben zu können, welcher eXtreme Scale-Server verwendet werden soll.

WebSphere eXtreme Scale - Speichern
Sie können Informationen in der Umgebung eines eXtreme Scale-Server-Cache speichern.

- Sie können Informationen mithilfe eines Schlüssels und Objekts in einem eXtreme Scale-Cache speichern.
- Durch Kombinieren der eXtreme Scale-Mediationsbasiselement für Speichern (Store) und Abrufen (Retrieve) können Sie mithilfe des Mediationsbasiselements 'Speichern' Daten in den Cache stellen und mithilfe des Mediationsbasiselements 'Abrufen' zuvor in den Cache gestellte Daten abrufen.
- Sie müssen eXtreme Scale-Definitionen mithilfe der WebSphere ESB-Administrationskonsole erstellen, um angeben zu können, welcher eXtreme Scale-Server verwendet werden soll.

Dynamische Weiterleitung

Zur Weiterleitung von Nachrichten gibt es verschiedene Möglichkeiten. Dazu werden Endpunkte verwendet, die zur Integrationszeit oder dynamisch zur Laufzeit definiert werden.

Die dynamische Weiterleitung bezieht sich auf zwei Fälle der Nachrichtenweiterleitung:

- Nachrichtenweiterleitung, bei der der Ablauf dynamisch ist, alle möglichen Endpunkte aber in einem SCA-Modul vordefiniert sind.
- Nachrichtenweiterleitung, bei der sowohl der Ablauf als auch die Endpunktauswahl dynamisch sind. Die Serviceendpunkte werden zur Laufzeit aus einer externen Quelle ausgewählt.

Dynamische Endpunktauswahl

Die Laufzeit ist in der Lage, Anforderungs- und Antwortnachrichten an eine Endpunktadresse weiterzuleiten, die durch ein Nachrichtenheaderelement identifiziert wird. Dieses Nachrichtenheaderelement kann durch Mediationsbasiselemente in einem Mediationsablauf aktualisiert werden. Die Endpunktadresse kann mit Informationen aus einer Registry, einer Datenbank oder der Nachricht selbst aktualisiert werden. Antwortnachrichten werden nur dann weitergeleitet, wenn die Nachricht von einem Web-Service-Export (JAX-WS) gesendet wird.

Für das SCA-Modul muss die Eigenschaft `Dynamischen Endpunkt verwenden, falls im Nachrichtenheader festgelegt` definiert sein, damit die dynamische Weiterleitung von der Laufzeit für eine Anforderung oder eine Antwort implementiert werden kann. Integrationsentwickler können die Eigenschaft `Dynamischen Endpunkt verwenden, falls im Nachrichtenheader festgelegt` definieren oder hochstufen (zur Laufzeit sichtbar machen), damit sie vom Laufzeitadministrator definiert werden kann. Moduleigenschaften können im Fenster für die **Moduleigenschaften** angezeigt werden. Klicken Sie zum Anzeigen des Fensters auf **Anwendungen** > **SCA-Module** > **Moduleigenschaften**. Der Integrationsentwickler ordnet hochgestuften Eigenschaften Aliasnamen zu und diese Namen werden in der Administrationskonsole angezeigt.

Registry

Mit IBM WebSphere Service Registry and Repository (WSRR) können Sie Serviceendpunktinformationen speichern und dann SCA-Module erstellen, um Endpunkte aus der WSRR-Registry abzurufen.

Bei der Entwicklung von SCA-Modulen wird das Mediationsbasiselement für die Endpunktsuche verwendet, um einem Mediationsablauf das Abfragen einer WSRR-Registry bezüglich eines Serviceendpunkts oder einer Gruppe von Serviceendpunkten zu ermöglichen. Wenn ein SCA-Modul eine Gruppe von Endpunkten abruft, muss das Modul ein anderes Mediationsbasiselement verwenden, um den gewünschten Endpunkt auszuwählen.

Mediationsrichtliniensteuerung für Serviceanforderungen

Mithilfe von Mediationsrichtlinien können Mediationsabläufe zwischen Serviceanforderern und Serviceanbietern gesteuert werden.

Mediationsabläufe können mithilfe von Mediationsrichtlinien gesteuert werden, die in IBM WebSphere Service Registry and Repository (WSRR) gespeichert sind. Die Implementierung der Servicerichtlinienverwaltung in WSRR basiert auf dem Web Services Policy Framework (WS-Policy).

Zur Steuerung von Serviceanforderungen mit Mediationsrichtlinien müssen in der WSRR-Registry geeignete SCA-Module und Mediationsrichtliniendokumente vorhanden sein.

Mediationsrichtlinie an eine Serviceanforderung anhängen

Bei der Entwicklung eines SCA-Moduls, das eine Mediationsrichtlinie verwenden soll, muss in den Mediationsablauf ein Mediationsbasiselement für die Richtlinienauflösung eingefügt werden. Während der Ausführung ruft das Mediationsbasiselement für die Richtlinienauflösung Mediationsrichtlinieninformationen aus der Registry ab. Deshalb muss das SCA-Modul eine Mediationsablaufkomponente enthalten, damit die Steuerung von Serviceanforderungen durch Mediationsrichtlinien unterstützt wird.

In der Registry kann mindestens eine Mediationsrichtlinie an ein SCA-Modul oder an einen vom SCA-Modul verwendeten Zielservice angehängt werden. Angehängte Mediationsrichtlinien können für alle von diesem SCA-Modul verarbeiteten Servicenachrichten verwendet werden (befinden sich innerhalb des Geltungsbereich). Die Mediationsrichtlinien können über Richtlinienanhänge verfügen, die Bedingungen definieren. Über Mediationsrichtlinienbedingungen können unterschiedliche Mediationsrichtlinien in verschiedenen Kontexten angewendet werden. Darüber hinaus können Mediationsrichtlinien über Klassifizierungen zum Angeben eines Governance-Zustands verfügen.

WebSphere Service Registry and Repository

Mit WebSphere Service Registry and Repository (WSRR) können Sie Informationen zu Serviceendpunkten und Mediationsrichtlinien speichern und verwalten und auf diese Informationen zugreifen. Verwenden Sie WSRR, um Serviceanwendungen dynamischer zu gestalten und besser an sich ändernde Geschäftsbedingungen anzupassen.

Einführung

Mediationsabläufe können WSRR als Mechanismus für die dynamische Suche verwenden, der Informationen zu Serviceendpunkten oder Mediationsrichtlinien bereitstellt.

Zur Konfiguration des Zugriffs auf WSRR müssen über die Administrationskonsole WSRR-Definitionsdokumente erstellt werden. Alternativ dazu können Sie die WSRR-Verwaltungsbefehle über den Scripting-Client 'wsadmin' verwenden. Die WSRR-Definitionen und die zugehörigen Verbindungseigenschaften sind der Mechanismus, der für die Verbindung zu einer Registry-Instanz und zum Abrufen eines Serviceendpunkts oder einer Mediationsrichtlinie verwendet wird.

Serviceendpunkte

Mit WSRR können Informationen zu den Services gespeichert werden, die Sie bereits verwenden, in Zukunft verwenden möchten oder über die Sie Informationen benötigen. Diese Services können sich in Ihren oder in anderen Systemen befinden. Eine Anwendung könnte WSRR beispielsweise verwenden, um den Service zu suchen, der zur Erfüllung der jeweiligen Funktions- und Leistungsanforderungen am besten geeignet ist.

Bei der Entwicklung eines SCA-Moduls, das über WSRR auf Serviceendpunkte zugreifen soll, muss in den Mediationsablauf ein Mediationsbasiselement für die Endpunktsuche eingefügt werden. Zur Laufzeit ruft das Mediationsbasiselement für die Endpunktsuche Serviceendpunkte aus der Registry ab.

Mediationsrichtlinien

Sie können WSRR auch zum Speichern von Mediationsrichtlinieninformationen verwenden. Mediationsrichtlinien können Ihnen durch das dynamische Überschreiben von Moduleigenschaften bei der Steuerung von Serviceanforderungen helfen. Wenn WSRR Mediationsrichtlinien enthält, die an ein Objekt angehängt sind, das entweder das SCA-Modul oder den Zielservice darstellt, können die Moduleigenschaften von diesen Mediationsrichtlinien überschrieben werden. Durch die Erstellung von Mediationsrichtlinienbedingungen können in verschiedenen Kontexten unterschiedliche Mediationsrichtlinien verwendet werden.

Anmerkung: Mediationsrichtlinien befassen sich mit der Steuerung von Mediationsabläufen, aber nicht mit der Sicherheit.

Bei der Entwicklung eines SCA-Moduls, das eine Mediationsrichtlinie verwenden soll, muss in den Mediationsablauf ein Mediationsbasiselement für die Richtlinienauflösung eingefügt werden. Während der Ausführung ruft das Mediationsbasiselement für die Richtlinienauflösung Mediationsrichtlinieninformationen aus der Registry ab.

WebSphere eXtreme Scale

Durch Verwendung des Produkts WebSphere eXtreme Scale (eXtreme Scale) können Sie ein Cachingsystem bereitstellen, das mit einer IBM Business Process Manager-Anwendung integriert werden kann. Die Verwendung von eXtreme Scale mit IBM Business Process Manager kann die Serviceantwortzeiten und -zuverlässigkeit verbessern und ermöglicht die Bereitstellung zusätzlicher Integrationsfunktionen.

eXtreme Scale fungiert als flexibles, skalierbares, speicherinternes Datengrid. Das Datengrid ermöglicht ein dynamisches Zwischenspeichern, Partitionieren, Replizieren und Verwalten von Anwendungsdaten und Geschäftslogik auf mehreren Servern. Mit eXtreme Scale können Sie auch für Servicequalität sorgen, wie beispielsweise für Transaktionsintegrität, hohe Verfügbarkeit und vorhersehbare Antwortzeiten.

Sie können Mediationsabläufe verwenden, um auf die Cachingfunktion von eXtreme Scale zuzugreifen, indem Sie die WebSphere eXtreme Scale-Mediationsbasiselemente in Ihren Ablauf integrieren. Bei der Entwicklung eines SCA-Moduls (Service Component Architecture), das Informationen in einem eXtreme Scale-Cache speichern muss, ist es erforderlich, das WebSphere eXtreme Scale-Mediationsbasiselement für Speichern (Store) in den Mediationsablauf zu integrieren. Wenn Sie Informationen aus einem eXtreme Scale-Cache abrufen wollen, müssen Sie das WebSphere eXtreme Scale-Mediationsbasiselement für Abrufen (Retrieve) integrieren. Durch Kombinieren der beiden Mediationsbasiselemente in einem Mediationsablauf können Sie die Antwort von einem Back-End-System in den Cache stellen, sodass zukünftige Anforderungen diese Antwort aus dem Cache abrufen können.

Um den Zugriff auf eXtreme Scale zu konfigurieren, müssen Sie mithilfe der Administrationskonsole eine WebSphere eXtreme Scale-Definition erstellen. Alternativ dazu können Sie auch die WebSphere eXtreme Scale-Verwaltungsbefehle über den Scripting-Client 'wsadmin' verwenden. Bei einer eXtreme Scale-Definition handelt es sich um den Mechanismus, der von den WebSphere eXtreme Scale-Mediationsbasiselementen für Abrufen (Retrieve) und Speichern (Store) verwendet wird, um eine Verbindung zu einem eXtreme Scale-Server herzustellen.

Message Service Clients

Message Service Clients stehen für C/C++ und .NET zur Verfügung, um Nicht-Java-Anwendungen eine Verbindung zum Enterprise Service Bus zu ermöglichen.

Message Service Clients for C/C++ and .NET stellen eine API namens XMS bereit, die über dieselbe Gruppe von Schnittstellen wie die JMS-API (JMS - Java Message Service) verfügt. Message Service Client for C/C++ enthält zwei Implementierungen von XMS, eine für C-Anwendungen und eine weitere für C++-Anwendungen. Message Service Client for .NET enthält eine vollständig verwaltete Implementierung von XMS, die von allen .NET-konformen Sprachen verwendet werden kann.

Message Service Client for .NET erhalten Sie unter http://www-01.ibm.com/support/docview.wss?rs=0&q1=IA9H&uid=swg24011756&loc=en_US&cs=utf-8&cc=us&lang=en.

Message Service Client for C/C++ erhalten Sie unter http://www-01.ibm.com/support/docview.wss?rs=0&q1=ia94&uid=swg24007092&loc=en_US&cs=utf-8&cc=us&lang=en.

Sie können auch die Java EE-Clientunterstützung von WebSphere Application Server Network Deployment installieren und verwenden, einschließlich Web-Service-Client, EJB-Client und JMS-Client.