IBM Business Process Manager
Versiune 8 Ediţie 0

Privire generală asupra IBM Business Process Manager

Cărţile PDF şi centrul de informare

Cărţile PDF sunt furnizate pentru tipărire şi pentru lectură online. Pentru cele mai recente informaţii, vedeţi centrul de informare online.

Setul de cărţi PDF include acelaşi conţinut ca şi centrul de informare. Unele legături din cărţile PDF au fost ajustate pentru folosirea în centrele de informare şi este posibil să nu funcţioneze corect.

Documentaţia PDF este disponibilă cam într-un trimestru după apariţia unei ediţii importante a centrului de informare, cum ar fi Versiunea 7.0 sau Versiunea 7.5.

Documentaţia PDF este actualizată mai puţin frecvent decât centrul de informare, dar mai frecvent decât manualele Redbooks. În general, cărţile PDF sunt actualizate când s-au strâns suficiente modificări pentru carte.

Cuprins

Capitolul 1. Iniţiere în IBM Business Process Manager

Înţelegeţi ce capabilităţi oferă IBM® Business Process Manager pentru gestionarea proceselor operaţionale şi relaţiile dintre diferitele faze ale gestionării proceselor operaţionale, cum ar fi crearea şi implementarea aplicaţiilor de proces.

Aplicaţia de proces reprezintă containerul fundamental pentru procese şi componentele lor în IBM Business Process Manager. Proiectanţii procesului creează aplicaţii de proces în mediile de creaţie şi ar putea include servicii, taskuri şi artefacte necesare pentru a suporta execuţia.

Serviciile Advanced Integration sunt implementate în IBM Integration Designer şi asociate cu aplicaţii de proces. Din Process Center, aplicaţiile de proces sunt implementate în Process Server, care reprezintă mediul runtime al proceselor pentru IBM Business Process Manager.

Similar, procesele automatizate create în Integration Designer pot utiliza fluxuri de activitate umană care au fost dezvoltate în IBM Process Designer.

Sumarul ediţiei

Familiarizaţi-vă cu ceea ce e nou în IBM Business Process Manager V8.0.1 şi accesaţi resurse valoroase care vă ajută să porniţi în diferite arii ale produsului.

Ce este nou

IBM BPM V8.0.1 oferă numeroase caracteristici produs noi şi diferite îmbunătăţiri aduce capabilităţilor actuale. Consultaţi lista completă a noilor caracteristici produs şi capabilităţi îmbunătăţite pentru IBM BPM V8.0.1.

“Ce este nou în IBM Business Process Manager V8.0.1” la pagina 11

Caracteristici depreciate

IBM BPMextinde capabilităţile găsite în versiunile anterioare WebSphere® Integration Developer, WebSphere Lombardi Edition, IBM Business Process Manager, WebSphere Process Server, WebSphere Enterprise Service Bus, aşi ale produse de gestionare a procesului operaţional IBM.

Consultaţi un rezumat al caracteristicilor produs perimate şi înlăturate în IBM BPM V8.0.1:

caracteristici perimate şi înlăturate ale IBM Business Process Manager

Cerinţele de sistem

Revedeţi cerinţele sistem şi asiguraţi-vă că sunt întrunite înainte de instalarea fiecărui produs în suita IBM BPM V8.0.1.

IBM Business Process Manager Advanced
Cerinţe sistem detaliate ale IBM Business Process Manager Advanced

IBM Business Process Manager Standard
Cerinţe sistem detaliate ale IBM Business Process Manager Standard

IBM Business Process Manager Express
Cerinţe sistem detaliate ale IBM Business Process Manager Express

IBM Business Process Manager Tools & Add-Ons
Cerinţe sistem detaliate ale add-on-urilorIBM Business Process Manager Tools

IBM Integration Designer
Cerinţe sistem detaliate ale IBM Integration Designer

IBM Business Monitor
Cerinţe sistem detaliate ale IBM Business Monitor şi WebSphere

Note de ediţie

Verificaţi notele de ediţie de pe site-ul web de suport pentru limitări şi metode de evitare.

IBM Business Process Manager Advanced
http://www.ibm.com/support/search.wss?q=ibpma801relnotes

IBM Business Process Manager Standard
http://www.ibm.com/support/search.wss?q=ibpms801relnotes

IBM Business Process Manager Express
http://www.ibm.com/support/search.wss?q=ibpme801relnotes

IBM Integration Designer
http://www.ibm.com/support/search.wss?q=iid801relnotes

IBM Business Monitor
http://www.ibm.com/support/search.wss?q=mon801relnotes

Process Designer
http://www.ibm.com/support/search.wss?q=pd801relnotes

Business Space
http://www.ibm.com/support/search.wss?q=bsp801relnotes

Resurse suplimentare

Folosiţi următoarele resurse pentru a accesa legăturile de comunitate IBM BPMşi pentru a partaja cunoştinţe.

IBM Business Process Manager Community Wiki
Găsiţi şi partajaţi cunoştinţe despre IBM Business Process şi Decision Management cu alţi utilizatori. Wiki-ul IBM Business Process Manager Community este locul unde găsiţi conţinutul comunităţii WebSphere Lombardi Edition şiLombardi Teamworks.

IBM BPM Community

Schimb de eşantioane
Găsiţi şi partajaţi aplicaţii eşantion, kituri de unelte şi alte coduri pe care le puteţi folosi în soluţiile IBM Business Process and Decision Management.

Schimb de eşantioane

Ediţii de servicii

Corecţiile cumulative pentru IBM Business Process Manager V8.0 şi IBM Integration Designer V8.0 sunt disponibile de la site-ul web Suport:

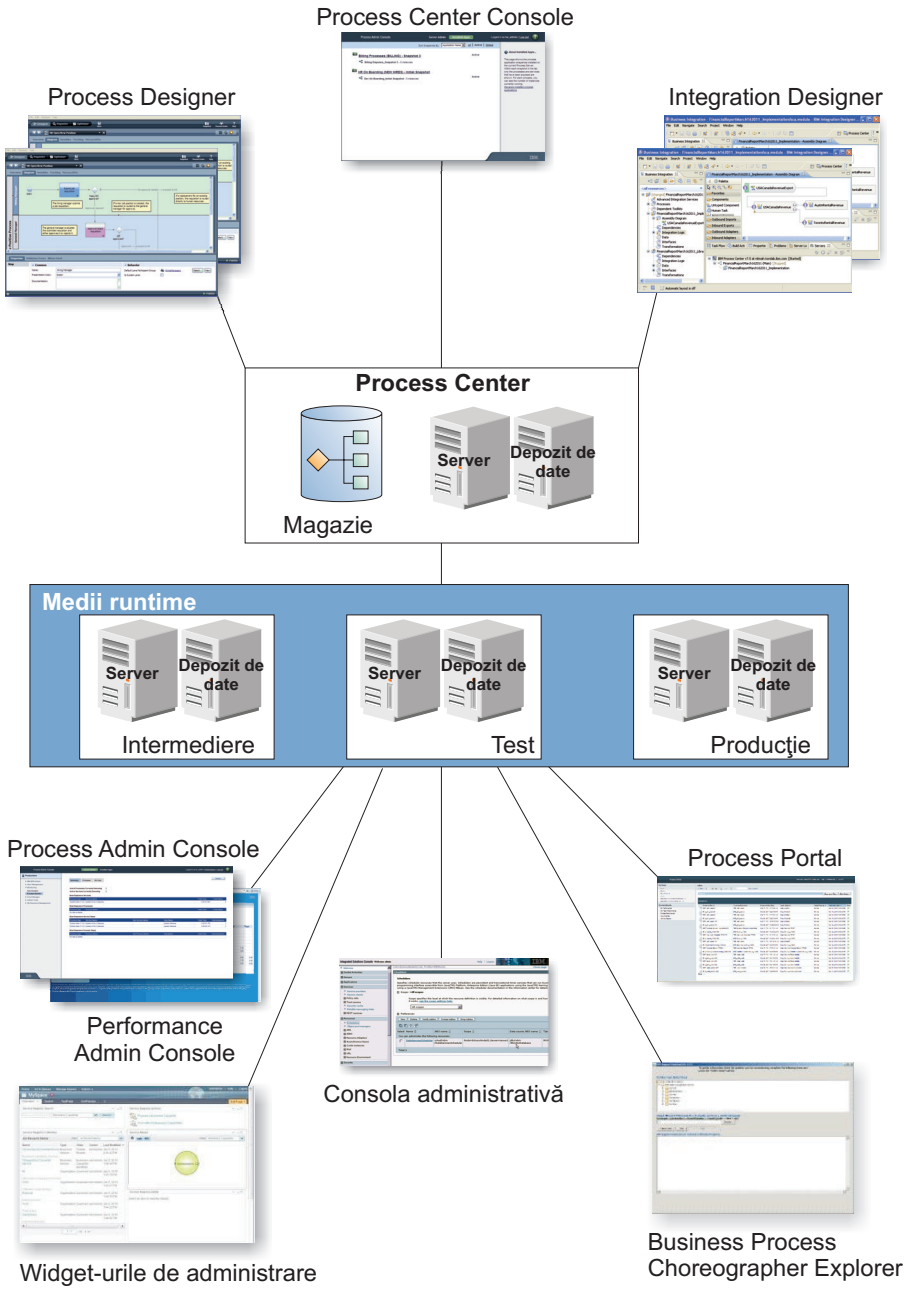
- Lista de corecţii pentru produsele IBM Business Process Manager V8.0
- Lista de corecţii pentru IBM Integration Designer Version 8.0

Examinaţi informaţiile pentru produsul dumneavoastră şi apoi descărcaţi şi instalaţi pachetele corespunzătoare.

Prezentarea produsului

IBM Business Process Manager este o platformă cuprinzătoare de gestionare a proceselor operaţionale, care asigură vizibilitatea completă şi informaţiile necesare pentru gestionarea proceselor operaţionale. Furnizează instrumente şi un mediu runtime pentru design-ul, execuţia, monitorizarea şi optimizarea proceselor, precum şi suport de bază pentru integrarea sistemelor. Produsul poate fi configurat să suporte diverse niveluri de complexitate şi implicare cu gestiunea procesului operaţional.

Componentele IBM Business Process Manager furnizează o magazie unificată BPM; unelte pentru autori, administratori şi utilizatori; şi o platformă runtime. Următoarea diagramă ilustrează o configuraţie IBM Business Process Manager obişnuită:



- Din mediile de creaţie IBM Process Designer şi IBM Integration Designer, dezvoltatorii se conectează la IBM Process Center. Din oricare dintre aceste unelte de dezvoltare aplicaţie bazate pe GUI, dezvoltatorii pot crea, testa, depana şi implementa aplicaţii operaţionale. Alegeţi o unealta sau alta, în funcţie de tipul aplicaţiei pe care o dezvoltaţi. Ar putea exista şi cazuri în care utilizarea ambelor unelte aduce avantaje semnificative.
- În mediile de creaţie Process Designer şi Integration Designer, designer-ii de procese şi servicii creează aplicaţii proces implementabile şi truse de unelte reutilizabile. Aplicaţiile proces conţin modele de procese şi implementări de servicii, inclusiv fişierele de suport care sunt necesare. Aplicaţiile proces sunt stocate în magazia Process Center astfel încât să poată fi partajate.
- Process Center include două servere, serverul Process Center şi serverul Performance Data Warehouse. Aceste servere permit dezvoltatorilor care lucrează în Process Designer să îşi ruleze aplicaţiile proces şi să îşi depoziteze datele de performanţă pentru testarea şi redarea în timpul eforturilor de dezvoltare. Performance Data Warehouse extrage datele urmărite din serverul Process Server sau Process Center la intervale regulate.

- Process Center suportă de asemenea numeroase funcţii administrative. Din Consola Process Center, administratorii instalează aplicaţii proces care sunt pregătite pentru intermediere, testare sau producţie pe serverele de proces. Administratorii pot gestiona de asemenea instanţe care rulează ale aplicaţiilor proces din medii configurate.
- Instalaţi aplicaţii proces pe un server de proces pentru intermediere, testare şi producţie. Mediile runtime suportă procese Business Process Model and Notation (BPMN) 2.0. IBM Business Process Manager Advanced suportă de asemenea procese Business Process Execution Language (BPEL).
- Din Consola de Administrare a Proceselor şi din Consola de Administrare a Performanţei, administratorii pot gestiona şi menţine toate serverele runtime. Utilizaţi Process Admin Console pentru a gestiona serverul Process Center şi serverele de proces din mediile dumneavoastră runtime. Utilizaţi Performance Admin Console pentru a identifica gâtuirile de performanţă şi pentru a captura date de instrumentare pentru analiza suplimentară.
- Utilizaţi consola administrativă pentru a crea şi gestiona obiecte precum resurse, aplicaţii şi servere. În plus, utilizaţi consola administrativă pentru a vizualiza mesaje produs.
- Utilizaţi Business Space pentru a crea spaţii de afaceri personalizate care furnizează widget-uri pentru monitorizarea sau administrarea diferitelor aspecte ale sistemului dumneavoastră. De exemplu, puteţi monitoriza activităţi de afaceri, servicii şi sănătate sistem sau administra politici de mediere şi calendare de afaceri. Puteţi crea de asemenea un spaţiu operaţional cu widget-urile Human Task Management şi îl puteţi utiliza pentru a participa la procesele operaţionale.
- Utilizând Process Portal, participanţii la proces se pot conecta la serverul Process Center sau la un Process Server din orice mediu runtime configurat, dacă un proces este dezvoltat, testat sau a fost lansat către un mediu de producţie.
- Gestionaţi instanţele de proces Business Process Execution Language (BPEL) din Business Process Choreographer Explorer sau din Business Space.

Configuraţiile IBM Business Process Manager V8.0

Configuraţiile diferite de IBM Business Process Manager se corelează cu puncte de intrare sau etape tipice din programul de gestiune a proceselor operaţionale al unei companii.

Tabela 1. Configuraţiile IBM Business Process Manager

Configurare	**Fază**
Avansat	**Transformare** Set complet de capabilităţi de gestiune procese operaţionale • Suport extins pentru automatizare proces de volum înalt • Componente SOA încorporate pentru integrare, orchestrare extensivă de servicii la nivel de întreprindere
Standard	**Program** Configurat pentru proiecte de gestiune procese operaţionale (BPM) tipice • Pentru programe de îmbunătăţire proiecte multiple, cu implicare operaţională înaltă • Suport integrare sistem de bază • Timp scurt de valorificare şi productivitate pe utilizator îmbunătăţită
Expres	**Proiect** Configurat pentru primul proiect BPM • Timp-la-valoare scurt: productivitate utilizator îmbunătăţită • Preţ mic • Instalare şi configurare uşoară

Capabilităţi de configurare IBM Business Process Manager V8.0

Înţelegeţi ce produse şi capabilităţi sunt oferite de IBM pentru gestiunea proceselor operaţionale şi alegeţi soluţia potrivită pentru întreprinderea dumneavoastră.

IBM Business Process Manager reprezintă o platformă BPM singulară care combină capabilităţile centrate pe integrare şi cele umane într-un produs unificat. Diferite configuraţii ale produsului sunt disponibile pentru diferiţi utilizatori şi îndeplinesc diferite necesităţi în întreprindere. Configuraţiile produselor pot fi combinate pentru creaţiile în colaborare şi mediile runtime implementate pe reţea.

Tabela 2. Capabilităţi de configurare IBM Business Process Manager

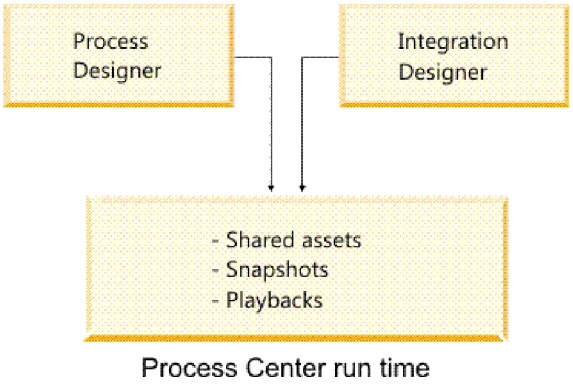
Capabilitate	**Adv.**	**Std.**	**Expres**
Execuţie compatibilă WebSphere Lombardi Edition	X	X	X
Process Designer (BPMN)	X	X	X
Editare în colaborare / Redare imediată	X	X	X
Interfeţe utilizator "coach proces" interactive	X	X	X
Reguli bazate pe procese ILOG	X	X	X
Process Portal	X	X	X
Monitorizare şi raportare în timp real	X	X	X
Optimizator şi analize de performanţă	X	X	X
Performance Data Warehouse	X	X	X
Process Center / magazie de aseturi partajate	X	X	X
Utilizatori finali şi autori de procese nelimitaţi	X	X	200 utilizatori / 3 autori
Disponibilitate înaltă: funcţionare în cluster şi nuclee nelimitate	X	X	• producţie pe 4 nuclee • dezvoltare pe 2 nuclee • Niciun cluster
Execuţie compatibilă WebSphere Process Server	X		
Integration Designer (BPEL / SOA)	X		
ESB (Enterprise Service Bus) încorporat	X		
Suport pentru tranzacţii	X		
Adaptoare de integrare	X		
Interfaţă utilizator Flexible Business Space	X		
Suport avansat pe platforme (Linux System z, IBM AIX, Solaris)	X	X	

Magazia Process Center

Process Center include o magazie pentru toate procesele, serviciile şi celelalte aseturi create în mediile de creaţie IBM Business Process Manager, Process Designer şi Integration Designer.

Process Center este o componentă software care rulează ca un server unde Process Designer şi Integration Designer împart aseturi, permiţându-le de fapt să dezvolte procese operaţionale cooperativ într-o manieră foarte interactivă. Procesele operaţionale pot utiliza puncte de monitorizare create cu Business Monitor Development Toolkit. Rezultatul este un proces operaţional care poate fi examinat în timpul rulării pentru eficacitate în condiţii reale de lucru.

În următoarea diagramă, puteţi vedea mai multe componente înrudite care vă permit împreună să construiţi procese operaţionale complexe.



Consola Process Center oferă uneltele necesare pentru menţinerea magaziei.

- Din consola Process Center, puteţi crea aplicaţii de proces şi truse de unelte şi acorda altor utilizatori acces la acele aplicaţii de proces şi truse de unelte.
- În mediile de creaţie, puteţi crea modele de proces, servicii şi alte aseturi în aplicaţiile de proces.
- Process Center include un server Process Center şi un depozit de date de performanţă, permiţând-le utilizatorilor care lucrează în mediile de creaţie să ruleze procese şi să depoziteze date de performanţă în scopuri de testare şi redare.
- Din consola Process Center, administratorii instalează aplicaţii de proces care sunt gata pentru testare sau producţie pe serverele Process în acele medii.
- Din consola Process Center, administratorii gestionează instanţele de aplicaţii de proces care rulează în mediile configurate.

Consola Process Center oferă o locaţie convenabilă în care să creaţi şi să menţineţi containere de nivel înalt, cum ar fi aplicaţii de proces şi truse de unelte. Administratorii care nu lucrează în mod activ în vizualizarea Designer pot folosi consola Process Center pentru a furniza un cadru de lucru în care analiştii şi dezvoltatorii BPM pot construi procesele lor şi implementările care stau la baza proceselor. Alt task principal pentru administratori este gestionarea accesului la magazia Process Center prin setarea autorizărilor corespunzătoare pentru utilizatori şi grupuri.

Utilizatorii cu autorizaţie corespunzătoare pot realiza anumite taskuri administrative direct în Process Designer şi Integration Designer. De exemplu, un dezvoltator cu acces de scriere la aplicaţia proces care vrea să captureze starea tuturor aseturilor proiectului într-o stare semnificativă de dezvoltare poate crea un instantaneu în timp ce lucrează în vizualizarea Designer.

Process Server şi mediile runtime

Process Server oferă un singur mediu runtime BPM care poate suporta un interval de procese operaţionale precum şi orchestrare de servicii şi capabilităţi de integrare.

În mediul dumneavoastră de creaţie, Process Server integrat în Process Center vă permite să rulaţi procesele în timp ce le construiţi. Când sunteţi pregătit, puteţi instala şi rula aceleaşi procese pe Process Server în mediul dumneavoastră runtime. Componenta Business Performance Data Warehouse colectează şi agregă datele proceselor din procesele care rulează pe serverele Process. Puteţi utiliza aceste date pentru a îmbunătăţi procesele dumneavoastră operaţionale.

Process Admin Console vă permite să gestionaţi serverele de proces în mediile dumneavoastră runtime, de exemplu, intermedierea, testarea, producţia precum şi Process Server care face parte din Process Center.

Medii de creaţie

IBM Business Process Manager Advanced oferă două medii de creaţie. Utilizaţi IBM Process Designer pentru a modela eficient procese operaţionale care implică taskuri umane. Utilizaţi IBM Integration Designer pentru a construi servicii care sunt auto-conţinute sau care invocă servicii existente cum ar fi servicii web, aplicaţii de resurse de întreprindere sau aplicaţii care rulează în CICS şi IMS.

Process Designer este disponibil în toate ediţiile produsului. IBM Business Process Manager Advanced oferă de asemenea Integration Designer cu editorii şi adaptoarele sale asociate.

Process Designer

Un procesor este unitatea de logică importantă din IBM Business Process Manager. Este containerul pentru toate componentele unei definiţii de proces, inclusiv servicii, activităţi şi gateway-uri; evenimente cronometru, mesaj şi excepţie; linii de secvenţă, reguli şi variabile. Când modelaţi un proces, creaţi un BPD (definiţie proces operaţional) reutilizabil. Atât Process Designer cât şi Integration Designer pot crea modele de procese care pot conţine taskuri umane.

Process Designer vă ajută să dezvoltaţi procese operaţionale. Cu o unealtă grafică uşor de utilizat, puteţi crea o secvenţă de acţiuni care compun un proces operaţional şi în timp puteţi reface acel proces pe măsură ce se modifică circumstanţele.

Dacă una sau mai multe activităţi necesită acces la sisteme backend mari sau la servicii care furnizează date pentru procesul operaţional (de exemplu pentru a obţine informaţii despre clienţi), puteţi satisface această cerinţă utilizând Integration Designer. Utilizând o interfaţă simplă, o activitate din Process Designer poate apela un serviciu creat în Integration Designer. Acel serviciu poate utiliza fluxuri de mediere pentru a transforma, dirija şi îmbunătăţi date şi adaptoare pentru a ajunge la multe sisteme back-end în modul standard. Pe scurt, Process Designer se concentrează pe procesul operaţional şi Integration Designer se concentrează pe servicii automatizate pentru a completa procesul operaţional. Vedeţi Iniţiere în IBM Process Designer.

Toate proiectele Process Designer sunt conţinute în aplicaţii de proces. Memoraţi acele aplicaţii de proces şi artefactele asociate în magazia Process Center. Aplicaţiile de proces pot partaja aseturi care au fost amplasate în truse de unelte.

IBM Business Process Manager furnizează mai multe interfeţe de utilizator pentru a vă permite să modelaţi, să implementaţi, să simulaţi şi să inspectaţi procese operaţionale. Creaţi şi gestionaţi aplicaţii de proces, truse de unelte, piste şi instantanee din Process Center Console. Puteţi crea modele de proces, rapoarte şi servicii simple în Process Designer. Puteţi rula şi depana procese în Inspector. Puteţi rula simulări în Optimizer.

Aplicaţiile de proces dezvoltate în Process Designer pot fi rulate în orice moment pe serverul Process Center sau salvate la un instantaneu şi implementate pe Process Server. Acelaşi lucru este adevărat despre serviciile dezvoltate în Integration Designer şi asociate cu aplicaţii de proces.

Integration Designer

Un procesor este unitatea de logică importantă din IBM Business Process Manager. Este containerul pentru toate componentele unei definiţii de proces, inclusiv servicii, activităţi şi gateway-uri; evenimente cronometru, mesaj şi excepţie; linii de secvenţă, reguli şi variabile. Când modelaţi un proces, creaţi un BPD (definiţie proces operaţional) reutilizabil. Atât Process Designer cât şi Integration Designer pot crea modele de procese care pot conţine taskuri umane.

Integration Designer furnizează editori şi ajutoare pentru a ajuta dezvoltatorii să creeze procese şi servicii automatizate complexe. Este disponibil ca o componentă din IBM Business Process Manager Advanced sau ca un set de unelte autonom pentru alte utilizări.

IBM Integration Designer a fost proiectat ca un mediu complet de dezvoltare a integrării pentru acele aplicaţii integrate de construire. Aplicaţiile integrate nu sunt simple. Ele pot apela aplicaţii pentru Enterprise Information Systems (EIS), pot implica procese operaţionale din mai multe departamente sau din întreaga companie şi pot invoca aplicaţii local sau la distanţă. Aplicaţiile pot fi scrise în diverse limbaje şi pot fi rulate în diverse sisteme de operare. Componentele sunt create şi asamblate în alte aplicaţii integrate (adică, aplicaţii create dintr-un set de componente) prin editori vizuali.

Editorii vizuali reprezintă un strat de abstracţie între componente şi implementările lor. Un dezvoltator care utilizează uneltele poate asambla o aplicaţie integrată fără cunoştinţe detaliate despre implementarea fundamentală a fiecărei componente.

Uneltele Integration Designer sunt bazate pe o arhitectură orientată spre servicii. Componentele sunt servicii şi o aplicaţie integrată care implică mai multe componente este şi ea un serviciu. Serviciile create sunt conforme cu standardele de vârf, folosite la pe scară largă în industrie. Procesele BPEL, care devin de asemenea componente, sunt create în mod similar cu unelte vizuale uşor de utilizat care sunt conforme cu standardul industrial Business Process Execution Language.

În paradigma Integration Designer, componentele sunt asamblate în module. Importurile şi exporturile sunt utilizate pentru a partaja date între module. Artefactele amplasate într-o bibliotecă pot fi partajate între module.

Modulele şi bibliotecile pot fi asociate cu o aplicaţie de proces pentru utilizare cu Process Center şi pot fi utilizate ca servicii de procesele create în Process Designer. În astfel de cazuri, ele pot fi de asemenea implementate cu aplicaţia de proces.

Alternativ, modulele şi bibliotecile pot fi implementate direct la mediul de test sau la Process Server. Puteţi utiliza module de mediere pentru a crea fluxuri de mediere, pe care le puteţi implementa în Process Server.

IBM Integration Designer furnizează de asemenea capabilitatea de creare a tipurilor de date şi a mapărilor xml care pot fi implementate pe dispozitivul WebSphere DataPower. Puteţi de asemenea să transferaţi fişiere la şi de la WebSphere DataPower.

Unelte de administrare

IBM Business Process Manager include un set de unelte de administrare pentru a vă ajuta să realizaţi taskuri care variază de la instalare şi gestionare de instantanee până la administrarea proceselor şi lucrul cu resurse din mediul dumneavoastră IT.

Unelte linie de comandă

IBM Business Process Manager oferă unelte de linie de comandă, interfeţe de scriptare şi interfeţe de programare pentru administrarea mediului runtime.

- Uneltele de linie de comandă sunt programe simple pe care le rulaţi dintr-un prompt linie de comandă al unui sistem de operare pentru a realiza taskuri specifice. Utilizând aceste unelte, puteţi porni şi opri serverele de aplicaţii, verifica starea serverului, adăuga sau înlătura noduri şi alte taskuri.
- Programul de scriptare wsadmin (WebSphere administrativ) este un mediu de interpretare a comenzii non-grafic care vă permite să rulaţi opţiuni administrative într-un limbaj de scriptare şi să lansaţi în execuţie programe de limbaje de scriptare pentru executare. Acesta suportă aceleaşi taskuri ca şi consola administrativă, precum şi multe dintre taskurile consolei Process Center. Unealta wsadmin este intenţionată pentru medii de producţie şi operaţii nesupravegheate.
- Interfeţele de programare administrative sunt un set de clase şi metode Java sub specificaţia JMX (Java Management Extensions) care oferă suport pentru administrarea SCA (Service Component Architecture) şi a obiectelor business. Fiecare interfaţă de programare include o descriere a scopului său, un exemplu care demonstrează modul de utilizare a interfeţei sau clasei şi referinţe la descrierile metodei individuale.

Process Center Console

Process Center Console oferă o locaţie comodă pentru utilizatori pentru crearea şi menţinerea elementelor de bibliotecă de nivel înalt cum ar fi aplicaţiile de proces şi trusele de unelte. Acest lucru ajută la furnizarea unui cadru în care analiştii şi dezvoltatorii BPM îşi pot construi procesele şi implementările fundamentale. În plus, Process Center Console oferă unelte pentru menţinerea magaziei, inclusiv setarea autorizaţiei corespunzătoare pentru utilizatori şi grupuri.

Accesarea consolei Process Center printr-un browser Web (de exemplu, http://host:9080/ProcessCenter).

Consolă de administrare proces

Consola Administrare proces este folosită pentru a administra serverele de proces din mediul dumneavoastră, inclusiv utilizatorii şi instantaneele instalate pentru fiecare server. În plus, aceasta oferă unelte pentru a vă ajuta să gestionaţi cozile şi memorarea în cache.

Consola Administrare proces include Inspectorul de proces, o unealtă pentru vizualizarea şi gestionarea instanţelor de proces pentru aplicaţii de proces care rulează pe un server de proces specific.

Accesarea Consolei de administrare procese printr-un browser Web (de exemplu, http://host:9080/ProcessAdmin).

Consola de administrare Business Performance

Consola de administrare Business Performance include unelte pentru gestionarea Depozitelor de date de performanţă din mediul dumneavoastră. Puteţi utiliza această unealtă pentru gestionarea cozilor de server şi monitorizarea performanţei serverului.

Accesarea Consolei de administrare Business Performance printr-un browser Web (de exemplu, http://host:9080/PerformanceAdmin).

Consola administrativă WebSphere Application Server

Consola administrativă este folosită pentru a administra aplicaţii, servicii şi alte resurse într-un domeniu de celulă, nod, server sau cluster. Puteţi utiliza consola cu servere autonome şi cu manageri de implementare care gestionează toate serverele dintr-o celulă într-un mediu de reţea.

Dacă aţi instalat un profil autonom, aveţi un singur nod în propriul său domeniu administrativ, cunoscut ca o celulă. Utilizaţi consola administrativă pentru a gestiona aplicaţii, magistrale, servere şi resurse în cadrul acelui domeniu administrativ. În mod similar, dacă aţi instalat şi configurat o celulă de implementare de reţea, aveţi un nod manager de implementare şi unul sau mai multe noduri gestionate în aceeaşi celulă. Utilizaţi consola administrativă pentru a gestiona aplicaţii, seta noduri gestionate în celulă şi monitoriza şi controla acele noduri şi resursele lor.

Accesarea acestei console printr-un browser Web (de exemplu, http://host:9043/ibm/console).

Business Process Choreographer Explorer şi Business Process Archive Explorer

În funcţie de rolul dumneavoastră de utilizator, puteţi utiliza aceste interfeţe client pentru a gestiona procesele BPEL şi taskurile umane create în IBM Integration Designer, puteţi lucra cu taskurile alocate dumneavoastră, vizualiza procesele BPEL şi taskurile umane complete care sunt într-o bază de date arhivă sau şterge procesele şi taskurile din arhivă.

Widget-urile de administrare

Widget-urile de administrare oferă o cale de a gestiona şi monitoriza anumite componente ale soluţiei dumneavoastră de gestiune proces operaţional, inclusiv module şi servicii Advanced Integration Service. Utilizaţi aceste widget-uri într-un spaţiu operaţional pentru a furniza vizibilitate în aplicaţia dumneavoastră de servicii şi module şi pentru a răspunde la întrebări precum acestea:

- Ce servicii sunt consumate sau expuse de un modul şi care sunt timpul de răspuns şi debitul pe o perioadă de timp definită pentru aceste servicii?
- Care este starea unui modul?
- Există evenimente eşuate în modul?
- Ce politici de mediere sunt asociate modulului?
- Ce procese BPEL şi taskuri umane sunt utilizate într-un modul?
- Există calendare de afaceri sau reguli operaţionale în modul?

Utilizaţi unul sau mai multe dintre widget-uri pentru a obţine un instantaneu al sănătăţii de sistem generale a soluţiei dumneavoastră de afaceri, inclusiv starea topologiei dumneavoastră (medii de implementare, cluster-e), a aplicaţiilor sistem (de exemplu, managerul de eveniment eşuat sau Business Process Choreographer), a surselor de date, a motoarelor de mesagerie şi a cozilor de mesagerie.

Managerul de Reguli de procese operaţionale

Managerul de reguli de procese operaţionale este o unealtă bazată pe web care ajută analistul de afaceri în

răsfoirea şi modificarea valorilor regulilor operaţionale. Unealta este o opţiune a IBM Process Server pe care o puteţi selecta pentru instalare în timpul creării profilului sau după instalarea serverului.

Ce este nou în IBM Business Process Manager V8.0.1

IBM Business Process Manager V8.0.1 include îmbunătăţiri pentru Coach Designer care îl fac mai uşor de utilizat, un Portal de proces îmbunătăţit care livrează o experienţă de lucru foarte colaborativă, capabilităţi îmbunătăţite de guvernanţă, integrarea cu Managerul de soluţii SAP, integrarea îmbunătăţită cu sistemele Enterprise Content Management, o vizualizare grafică a fluxului de informaţii prin procesul operaţional şi o mulţime de noi caracteristici.

Process Designer

Pentru actualizările produsului de la livrările de service, vedeţi Fix list for IBM Business Process Manager Version 8.0 products.

Coach Designer îmbunătăţit creşte productivitatea şi este mai uşor de folosit pentru autorii proceselor operaţionale

În IBM BPM V8.0.1, Coach Designer include următoarele îmbunătăţiri:

- Paginile cu proprietăţile configuraţiei îmbunătăţite care grupează proprietăţi înrudite şi care afişează mai eficient proprietăţile care sunt obiecte business şi liste.
- Suport îmbunătăţit pentru legarea componentelor interfeţei de utilizator (UI)
- Controale suplimentare de stoc
- Suport îmbunătăţit pentru validarea datelor utilizatorului
- Abilitatea de a edita fişierele .CSS care sunt adăugate ca fişiere web
- Abilitatea de a genera wrapper-e pentru serviciile umane care sunt expuse ca tablouri de bord. Portleturile JSR 286 generate pot fi încorporate în Portalul WebSphere.
- S-a îmbunătăţit performanţa de runtime pentru elementele de control stoc Filă şi Tabel. Această performanţă îmbunătăţită este valabilă pentru toate browser-ele, inclusiv Microsoft Internet Explorer 8.

Procesele operaţionale îmbogăţite reacţionează la evenimentele de ciclu de viaţă ale documentelor şi ale conţinutului de afaceri critice din sistemele ECM (Enterprise Content Management)

IBM BPM V8.0.1 furnizează capabilităţi pentru crearea de procese operaţionale sofisticate care pot iniţia procese sau reacţiona la rularea proceselor pe baza evenimentelor de conţinut care apar într-un sistem ECM, ca atunci când documentele, folderele sau proprietăţile lor, sunt create, actualizate, şterse, înregistrate la intrare sau la ieşire.

Autorii proceselor operaţionale pot realiza taskurile următoare:

- Modelarea evenimentelor de conţinut pentru detectare de către procesele IBM BPM
- Definirea grafică a proceselor care reacţionează la evenimentele de conţinut
- Crearea şi configurarea abonamentelor de eveniment care se abonează la evenimentele de conţinut

IBM BPM V8.0.1 furnizează un handler de evenimente care poate fi instalat şi configurat în IBM FileNet Content Manager V5.1 sau mai nou. IBM BPM V8.0.1 furnizează şi documentaţie despre cum să creaţi handler-e de evenimente pentru alte sisteme ECM.

Noua integrare a managerului de soluţii SAP oferă modalităţi pentru a modela şi personaliza mai uşor planurile de afaceri SAP

Capabilităţile mediului de proiectare IBM BPM V8.0.1 oferă o cale mai uşoară pentru editarea, modelarea şi personalizarea planurilor de afaceri SAP folosind IBM Process Designer. Integrarea managerului de soluţii SAP permite utilizatorilor afacerii SAP să îşi personalizeze procesele SAP într-un timp mai scurt, cu un risc mai mic de erori şi cu un efort mai mic.

Următoarele capabilităţi ale mediului de proiectare furnizează paşii spre o abordare integrată şi uşor de folosit pentru gestionarea proceselor operaţionale SAP folosind IBM BPM:

- Schimbul iterativ, bidirecţional al definiţiilor de proces selectate între IBM BPM şi managerul de soluţii SAP
- Personalizare simplificată pentru definiţiile procesului şi secvenţele de tranzacţii SAP asociate şi informaţiile asociate folosind uneltele grafice de proiectare în modelatorul de proces

Vizibilitate în fluxul datelor de procesare sporeşte productivitatea autorilor procesului operaţional

- IBM BPM V8.0.1 introduce vizualizarea fluxului de date care vă permite să vizualizaţi modul în care informaţiile trece prin procesul operaţional. Utilizaţi noua opţiune de vizualizare a datelor în IBM Process Designer pentru a crea o reprezentare vizuală pentru definiţiile proceselor operaţionale selectate (BPD - business process definition). Vizualizare este afişată într-o fereastră a noului browser web. Acesta conţine vizualizarea diagramei bazate pe web a BPD-ului. Puteţi selecta variabile de date sau grupuri de taguri pentru vizualizare.
- Capacitatea de etichetare îmbunătăţită vă permite să creaţi grupuri de taguri cu valori definite de utilizator. Grupurile de taguri furnizează modalităţi unice pentru clasificarea unui set comun de aseturi. Aseturile pot fi redate într-o fereastră care vă ajută să vizualizaţi asocierile dintre componentele procesului.

Îmbunătăţirile de legătură se integrează cu mediile externe IBM BPM

Când lucraţi cu aplicaţii de proces sau seturi de unelte, s-ar putea să fie nevoie să vă legaţi la informaţiile înrudite care se află în afara mediului IBM BPM. Acum puteţi să creaţi documente care leagă alte produse, cum ar fi IBM Rational Team Concert poate utiliza pentru a lega artefactele proiectării de proces.

IBM BPM include următoarele capabilităţi ale legăturii de referinţă îmbunătăţită:

- Definirea tipurilor de relaţii între artefactele procesului şi artefactele la care se face referire
- Definirea tipurilor, cum ar fi cerinţe şi defecte pentru legături de referinţă
- Crearea link-urilor document pe care alte produse, precum IBM Rational Team Concert, le pot lega de artefactele de proiectare ale procesului
- Extinderea capabilităţile de referinţă la ataşamentele fişierului
- Asigurarea API-urilor pentru a interoga şi a raporta cu privire la link-urile de referinţă. Pentru a extrage link-uri de referinţă în aplicaţia de proces sau trusa de unelte actuală, puteţi utiliza în aplicaţia dvs. un API pentru un link de referinţă specializat.

Integrarea îmbunătăţită a serviciilor web simplifică integrarea serviciilor web

Îmbunătăţirile de integrare între procesele operaţionale şi serviciile web fac încorporarea serviciilor web mai uşoară în IBM BPM V8.0.1.

- Nu mai este nevoie ca serviciile web să îşi actualizeze spaţiul nume ţintă de fiecare dată când este realizat un instantaneu al aplicaţiei de proces.
- Dacă utilizaţi un pas Integrare serviciu web pentru a apela un serviciu web de ieşire, pasul dvs. poate prinde şi manipula defecte.
- Îmbunătăţirile continue pentru suportul manipulării erorilor BPMN 2.0 înseamnă că defectele serviciului web service în aplicaţiile de proces sunt tratate cu uşurinţă.

Process Portal

Pentru actualizările produsului de la livrările de service, vedeţi Fix list for IBM Business Process Manager Version 8.0 products.

IBM Process Portal este îmbunătăţit astfel încât participanţii la proces pot lucra mult mai eficient şi eficace. Process Portal include acum caracteristicile următoare:

- Abilitatea de a marca o pagină pentru a înlocui pagina de start implicită a portalului de proces
- Suport pentru integrarea IBM Connections, ceea ce permite înlocuirea cărţii de vizită implicite Process Portal cu cartea de vizită Connections
- Performanţă îmbunătăţită pentru logare şi tablou de bord

Process Center

Pentru actualizările produsului de la livrările de service, vedeţi Fix list for IBM Business Process Manager Version 8.0 products.

Capabilităţi îmbunătăţite de guvernare şi ciclu de viaţă integrează IBM BPM în procesele mai largi SDLC (Systems Development Lifecycle)

Puteţi dezvolta procese de guvernare personalizate pe care le puteţi aloca aplicaţiilor de procesare individuale. Şabloanele oferă resursele de bază pentru crearea proceselor care guvernează instalarea sau oferă notificări în cazul unei modificării în starea instantaneului. Procesele de guvernare trebuie să fie dezvoltate folosind aceste şabloane. Puteţi folosi, de asemenea, serviciile de guvernare pentru a proiecta un proces care instalează în mod automat un instantaneu al aplicaţiei de proces pe server la modificarea stării.

IBM BPM include următoarele procese îmbunătăţite de guvernare şi ciclu de viaţă:

- Abilitate îmbunătăţită pentru definirea şi selectarea guvernării personalizate şi a ciclului de viaţă al procesului la nivelul aplicaţiei de proces
- Şabloane care furnizează procese de guvernare implicite
- Procese de guvernare eşantion pentru a putea porni

Îmbunătăţirile de legătură se integrează cu mediile externe IBM BPM

Când lucraţi cu aplicaţii de proces sau seturi de unelte, s-ar putea să fie nevoie să vă legaţi la informaţiile înrudite care se află în afara mediului IBM BPM. Acum puteţi să creaţi documente care leagă alte produse, cum ar fi IBM Rational Team Concert poate utiliza pentru a lega artefactele proiectării de proces.

IBM BPM include următoarele capabilităţi ale legăturii de referinţă îmbunătăţită:

- Definirea tipurilor de relaţii între artefactele procesului şi artefactele la care se face referire
- Definirea tipurilor, cum ar fi cerinţe şi defecte pentru legături de referinţă
- Crearea link-urilor document pe care alte produse, precum IBM Rational Team Concert, le pot lega de artefactele de proiectare ale procesului
- Extinderea capabilităţile de referinţă la ataşamentele fişierului
- Asigurarea API-urilor pentru a interoga şi a raporta cu privire la link-urile de referinţă. Pentru a extrage link-uri de referinţă în aplicaţia de proces sau trusa de unelte actuală, puteţi utiliza în aplicaţia dvs. un API pentru un link de referinţă specializat.

8.0.1.2+ **Extinderea abilităţii de a administra instantanee în Process Center**

Puteţi să ştergeţi instantaneele nedenumite sau arhivate ale unei aplicaţii de proces utilizând noua comandă BPMSnapshotCleanup.

Server de proces

Pentru actualizările produsului de la livrările de service, vedeţi Fix list for IBM Business Process Manager Version 8.0 products.

Comenzile noi administrative vă ajută să menţineţi integritatea mediului dvs. BPM

- Pentru a şterge datele de instanţă din definiţia procesului operaţional (BPD) pentru un instantaneu al unei aplicaţii de proces, utilizaţi comanda **BPMProcessInstancesCleanup**.

- Pentru a şterge instantaneele aplicaţiei de proces şi dependenţele acestora, utilizaţi comanda **BPMDeleteSnapshot**.

Instalare şi configurare

Pentru actualizările produsului de la livrările de service, vedeţi Fix list for IBM Business Process Manager Version 8.0 products.

Instalarea şi configurarea simplificate livrează micşorează timpul până la valorificare

IBM BPM V8.0.1 include caracteristicile următoare care îmbunătăţesc experienţa de instalare şi fac mai uşoară instalarea şi rularea proceselor operaţionale:

- Fişiere de răspuns simplificate şi îmbunătăţirea pentru instalarea IBM BPM
- A fost adăugată mai multă personalizare corespunzătoare topologiilor de producţie cu îmbunătăţiri aduse la nivelul vrăjitorului Mediu implementare şi al instalării prin script **configureNode**
- A integrat comanda **bootstrapProcessServerData** pentru mediile de implementare
- Determinarea îmbunătăţită a problemelor şi funcţionarea
- A fost adăugat z/OS în Ghidul de configurare şi instalare interactivă
- A fost adăugat suport pentru versiuni multiple pentru fişierul extensie IBM Business Process Manager for z/OS V8.0.1 pentru z/OS Profile Management Tool

8.0.1.2+ **Configurarea unei rădăcini de context personalizate într-un mediu de implementare simplifică administrarea**

Puteţi administra mai uşor mediul de implementare configurând o rădăcină de context personalizată. Acum puteţi utiliza comanda **BPMConfig** cu parametrul **-update** pentru a crea o configuraţie singulară de rădăcină de context, minimizând costurile iniţiale, reducând costurile legate de securitate şi scutind utilizatorii de problemele de gestionare a rădăcinii de context.

8.0.1.2+ **Un număr mai mic de instanţe de proces finalizate într-o bază de date reduce dimensiunea bazei de date**

Reduceţi dimensiunea bazei de date prin utilizarea noilor parametri care au fost adăugaţi în comanda **BPMProcessInstancesCleanup**, care micşorează numărul de instanţe de proces finalizate ce rulează într-o bază de date.

8.0.1.2+ **Înlăturarea mesajelor abonamentelor durabile din baza de date**

Utilizând comanda **BPMDeleteDurableMessages**, puteţi înlătura mesajele vechi de abonamente durabile din tabelul de bază de date LSW_DUR_MSG_RECEIVED. Puteţi utiliza această capabilitate pentru a reduce periodic dimensiunea tabelului.

Integration Designer

Pentru actualizările produsului de la livrările de service, vedeţi Fix list for IBM Integration Designer V8.0 Version 8.0 products.

Îmbunătăţirile de legătură se integrează cu mediile externe IBM BPM

Când lucraţi cu interfaţele obiectelor şi serviciilor operaţionale, s-ar putea să fie nevoie să vă legaţi la informaţiile înrudite care se află în afara Integration Designer. De exemplu, este posibil să doriţi să vă legaţi la un site web sau o pagină wiki. Acum puteţi include un link către o sursă externă într-o componentă care are un câmp Documentaţie.

Business Process Choreographer Explorer

Pentru actualizările produsului de la livrările de service, vedeţi Fix list for IBM Business Process Manager Version 8.0 products.

Configuraţie de securitate implicită

Implicit, Business Process Choreographer Explorer este folosit să folosească protocolul HTTPS. Orice încercare de a accesa Business Process Choreographer Explorer folosind HTTP este redirectată la HTTPS.

Business Process Archive Explorer

Pentru actualizările produsului de la livrările de service, vedeţi Fix list for IBM Business Process Manager Version 8.0 products.

Configuraţie de securitate implicită

Implicit, Business Process Archive Explorer este configurat să folosească protocolul HTTPS. Orice încercare de a accesa Business Process Archive Explorer folosind HTTP este redirectată la HTTPS.

Performance Data Warehouse

Pentru actualizările produsului de la livrările de service, vedeţi Fix list for IBM Business Process Manager Version 8.0 products.

8.0.1.2+ **Controlul dimensiunii Performance Data Warehouse**

Ştergeţi înregistrările din Performance Data Warehouse utilizând noua comandă **prune**, care este inclusă în unealta Performance Data Warehouse Tool (perfDWTool). Prin reducerea numărului de înregistrări care nu mai sunt necesare, puteţi evita creşterea excesivă a dimsniunii Performance Data Warehouse.

Pentru informaţii suplimentare despre perfDWTool, consultaţi Utilizarea uneltei de linie de comandă Performance Data Warehouse (perfDWTool).

8.0.1.2+ Pentru informaţii specifice despre comanda **prune**, consultaţi Reducerea datelor din baza de date Performance Data Warehouse.

Accesibilitatea în IBM Business Process Manager

Caracteristicile de accesibilitate ajută utilizatorii cu dizabilităţi fizice, cum ar fi mobilitatea redusă sau vederea limitată, să utilizeze cu succes produsele IT.

IBM se străduieşte să furnizeze produse care sunt accesibile pentru majoritatea persoanelor, indiferent de vârstă sau abilitate.

Pentru detalii legate de caracteristicile de accesibilitate ale acestui produs, consultaţi Caracteristici de accesibilitate în IBM Business Process Manager.

Disponibilitatea limbilor naţionale în IBM Business Process Manager

IBM Business Process Manager suportă următoarele limbi. Este posibil ca documentaţia să nu fie tradusă în întregime.

- Chineză simplificată
- Chineză tradiţională
- Cehă
- Engleză (S.U.A.)
- Franceză
- Germană

- Maghiară
- Italiană
- Japoneză
- Coreeană
- Poloneză
- Portugheză braziliană
- Rusă
- Spaniolă

IBM Business Process Manager asigură suport parţial pentru următoarele limbi. Este posibil ca documentaţia să nu fie tradusă în întregime.

- Arabă (tradusă pentru Business Process Choreographer Explorer, widget-urile IBM Business Process Manager din Process Portal şi Process Portal)
- Daneză (tradusă pentru widget-urile IBM Business Process Manager din Process Portal)
- Olandeză (tradusă pentru Process Designer, Process Center, widget-urile IBM Business Process Manager din Process Portal şi Process Portal)
- Finlandeză (tradusă pentru widget-urile IBM Business Process Manager din Process Portal)
- Greacă (tradusă pentru Process Designer, Process Center şi widget-urile IBM Business Process Manager din Process Portal)
- Ebraică (tradusă pentru Business Process Choreographer Explorer, widget-urile IBM Business Process Manager din Process Portal şi Process Portal)
- Norvegiană (tradusă pentru widget-urile IBM Business Process Manager din Process Portal)
- Portugheză-Portugalia (tradusă pentru Process Designer, Process Center şi Process Portal)
- Română (tradusă pentru operaţiile runtime)
- Slovacă (tradusă pentru widget-urile IBM Business Process Manager din Process Portal)
- Suedeză (tradusă pentru widget-urile IBM Business Process Manager din Process Portal)
- Turcă (tradusă pentru widget-urile IBM Business Process Manager din Process Portal)

  Notă: Pentru caracteristicile locate turce, trebuie să setaţi intrarea **case-insensitive-security-cache** din fişierul 60Database.xml la **false** pentru a permite numelor utilizator şi parolelor să conţină litera "**i**" (de exemolu, **tw_admin**). Fişierul 60Database.xml este localizat în directorul *install_root*\profiles*profile*\config\cells*cell*\nodes*node*\servers*server*\process-center\config\system\.

Important: Pentru Locale-ul turcă, trebuie să invocaţi Profile Management Tool autonom pentru a evita erorile. Nu invocaţi Profile Management Tool din Installation Manager.

IBM Business Process Manager furnizează suport utilizatorilor pentru a introduce şiruri bidirecţionale în mediul Process Designer şi în coach-uri, chiar dacă traducerea bidirecţională nu este suportată în acest componente. Acesta oferă API-uri JavaScript pentru manipularea bidirecţională a testului de limbă.

Privire generală asupra BPM (Business Process Management)

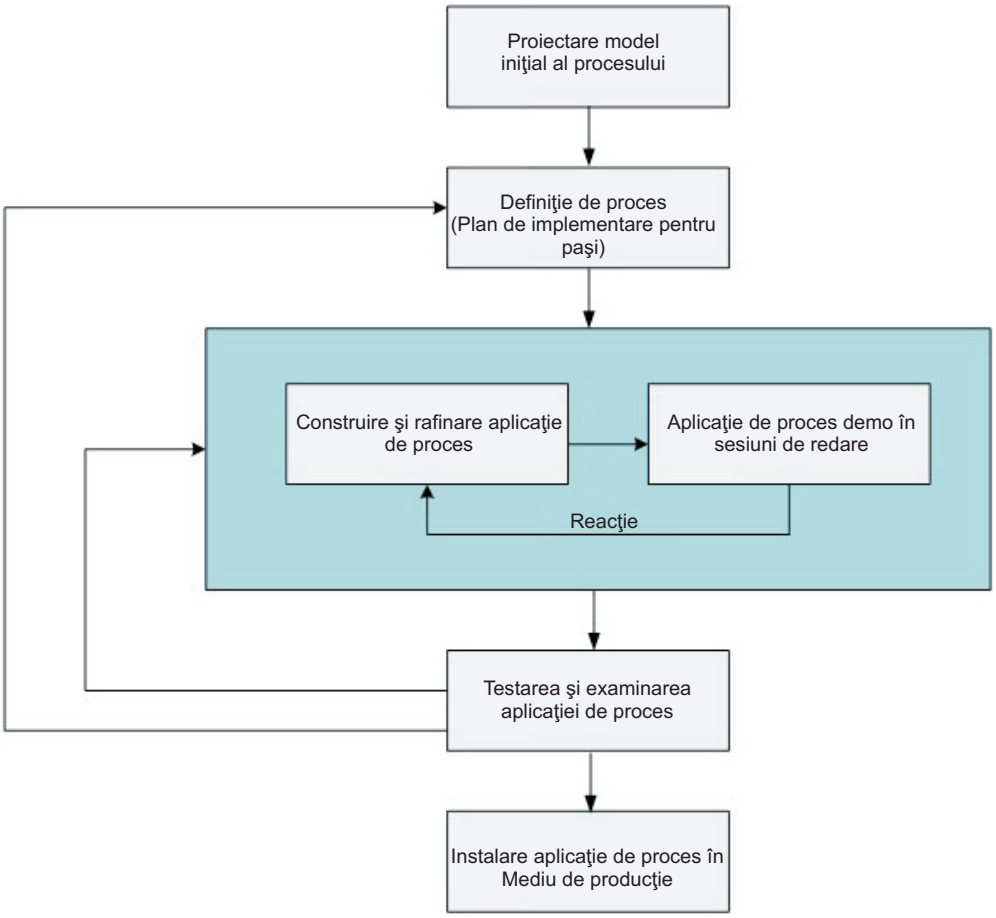
La dezvoltarea proceselor operaţionale în Process Designer, trebuie să planificaţi instalarea eventuală a aplicaţiilor dumneavoastră de proces pe servere din mediile dumneavoastră de test şi de producţie.

Process Designer poate fi găsit în IBM Business Process Manager Express, IBM Business Process Manager Standard şi IBM Business Process Manager Advanced. În această secţiune ne vom concentra pe versiunea Avansat, care este proiectată pentru automatizare de volum mare şi utilizare de servicii complexe dezvoltate în Integration Designer. Are componente SOA încorporate care pot fi utilizate pentru integrare extensivă de servicii la nivel de întreprindere. Versiunea Standard poate fi utilizată de mulţi profesionişti de afaceri cooperativ, pentru a dezvolta mai multe procese

sofisticate. Are integrare sistem de bază. Versiunea Express este pentru un număr mic de utilizatori de pe un singur server care sunt iniţiaţi în procesele operaţionale sau care nu au nevoie de acces la multe sisteme externe.

Următoarea diagramă afişează ciclul de viaţă al unui efort de dezvoltare de proces tipic. Include paşi pentru construirea şi rafinarea unui serviciu de instalare, astfel încât să vă puteţi instala aplicaţiile de proces în mediul de producţie.

După cum arată această diagramă, puteţi lucra exclusiv în mediul dumneavoastră de dezvoltare. Dar trebuie să configuraţi serverele Process Server pentru ambele medii, de test şi de producţie.



Privire generală asupra modelării de proces

Un proces este o unitate majoră de logică în IBM Business Process Manager. Este containerul pentru toate componentele unei definiţii proces, inclusiv servicii, activităţi şi gateway-uri; evenimente cronometru, mesaj şi excepţie; linii de secvenţă, reguli şi variabile. Când modelaţi un proces, creaţi un BPD (Business Process Definition) reutilizabil.

Componentele proces vă permit să definiţi fluxul de lucru al procesului pentru utilizatori finali, crearea de logică în interiorul unui proces şi integrarea cu alte aplicaţii şi surse de date. Pentru a înţelege ce se întâmplă în interiorul unui proces în timpul rulării, este important să înţelegeţi componentele care compun un proces în timpul proiectării.

Construirea proceselor în IBM BPM

Mulţi indivizi diferiţi din diverse organizaţii sunt implicaţi de obicei în dezvoltarea proceselor utilizând IBM BPM. Preocuparea de înlocuire este să vă asiguraţi că construiţi cea mai bună soluţie posibilă pentru îndeplinirea scopurilor declarate ale proiectului dumneavoastră. Pentru a asigura rezultate satisfăcătoare, membrii echipei ar trebui să lucreze

împreună pentru a îndeplini cerinţele de proces şi a dezvolta interactiv modelul şi implementările acestuia.

Reutilizarea elementelor în Process Designer

Process Designer permite dezvoltatorilor de procese să re-utilizeze elementele existente şi în aplicaţiile de proces şi peste acestea. De exemplu, dacă cunoaşteţi câteva servicii ce există deja şi care includ Antrenori şi alte elemente partajate de care dumneavoastră şi alţi dezvoltatori aveţi nevoie, puteţi accesa şi re-utiliza acele elemente incluzându-le pe acestea într-o trusă de unelte. Apoi, din aplicaţia dumneavoastră de proces, puteţi adăuga o dependenţă la trusa de unelte în care se află elementele partajate. Acest lucru vă permite să folosiţi unul din serviciile existente atunci când alegeţi implementarea pentru o activitate. Elementele din trusa de unelte pot de asemenea fi utilizate de către alţi dezvoltatori în diferite aplicaţii de proces.

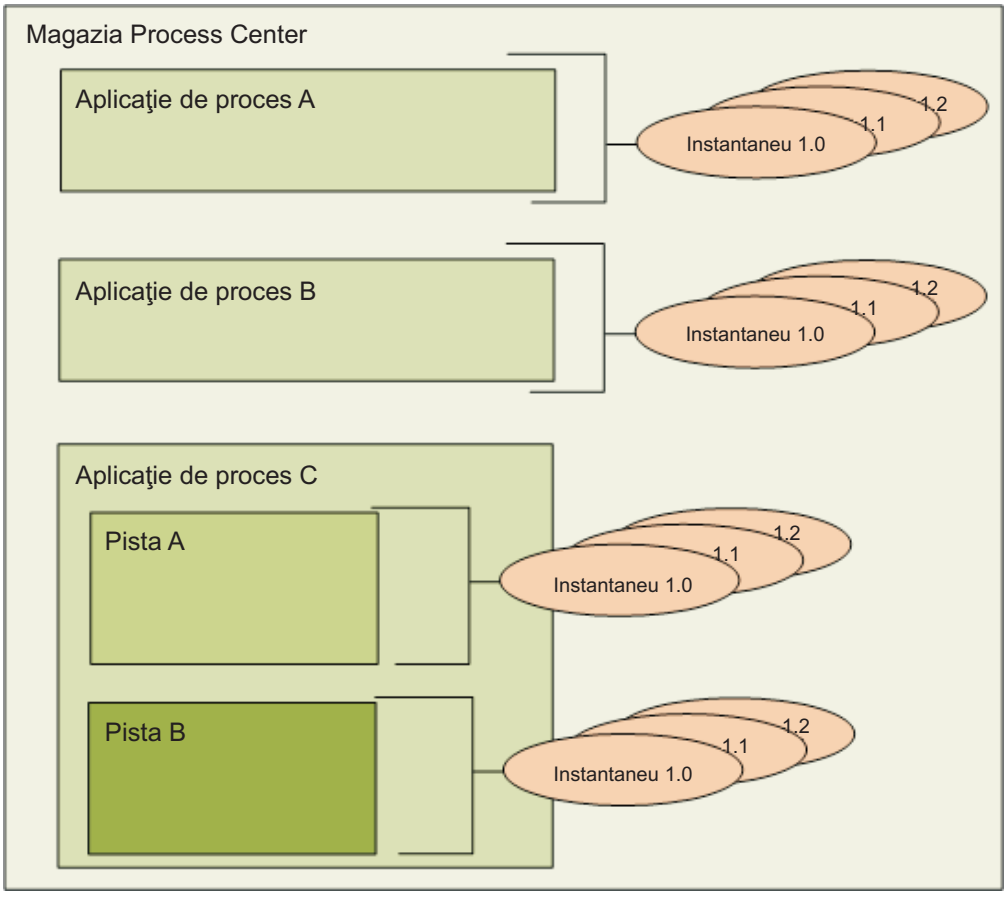
Utilizarea Designer în IBM Process Designer

Interfaţa Designer furnizează uneltele de care aveţi nevoie pentru a modela procesele dumneavoastră în IBM BPM.

Dezvoltarea procesului cu Process Center

IBM Process Center serveşte ca o magazie centrală pentru toate aseturile de proiect create în Process Designer. Atunci când mai mulţi clienţi Process Designer se conectează la Process Center, utilizatorii pot partaja elemente, cum ar fi procesele şi serviciile şi pot vedea, de asemenea, schimbările realizate de alţi utilizatori pe măsură ce se întâmplă. Process Center poate fi utilizată, de asemenea, ca un magazie pentru aseturile create în IBM Integration Designer.

Atunci când dezvoltaţi procese în Process Designer, există o ierarhie disponibilă în magazia Process Center proiectată pentru a vă ajuta să vă gestionaţi proiectele. Următoarea figură furnizează o privire generală conceptuală a ierarhiei magaziei:



După cum puteţi vedea din diagrama anterioară, magazia Process Center include următoarele artefacte:

Tip de conţinut	Descriere
Aplicaţii de Proces	Containere pentru modelele de proces şi implementările de suport pe care analiştii şi dezvoltatorii BPM le creează în Designer în IBM Process Designer.
Piste	Subdivizii opţionale într-o aplicaţie de proces bazată pe taskuri de echipă sau versiuni ale aplicaţiei de proces. Atunci când sunt activate, pistele permit realizarea dezvoltării paralele. Administratorii determină dacă pistele suplimentare sunt necesare şi, prin urmare, activate pentru fiecare aplicaţie de proces.
Instantanee	Înregistraţi starea elementelor dintr-o aplicaţie de proces sau pistă la un anumit moment de timp. De obicei, instantaneele reprezintă o bornă sau sunt utilizate pentru derulările înapoi sau instalări. Puteţi compara instantaneele şi reveni la cele anterioare. Dacă un administrator activează piste pentru o aplicaţie de proces, un instantaneu este utilizat ca bază pentru noua pistă.

Aplicaţii de proces: Privire generală

O aplicaţie de proces este un container pentru modelele de proces şi pentru implementările lor suportate; este memorată în magazie. După ce artefactele au fost create, sunt asamblate într-o aplicaţie de proces.

Aplicaţiile de proces conţin unele dintre sau toate artefactele următoare:

- Unul sau mai multe modele de proces, numite de asemenea BPD-uri (Business Process Definitions)
- Referinţe la truse de unelte
- Serviciile necesare pentru a implementa activităţi sau pentru integrarea cu alte sisteme, inclusiv Advanced Integration Services
- Una sau mai multe piste
- Modulele sau bibliotecile Service Component Architecture (SCA) (create în IBM Integration Designer)
- Un model IBM Business Monitor pentru monitorizarea performanţei afacerii
- Oricare alte articole necesare pentru a rula procesul

Sugestie pentru aplicaţia de proces, instantanee şi piste

Toate modificările pe care le faceţi la o aplicaţie proces sunt salvate dinamic la magazia Process Center la sugestie, care este versiunea curentă a aplicaţiei proces care funcţionează. Puteţi utiliza sesiuni de redare pe sugestie pentru a testa instantaneu şi a gestiona versiunea curentă care funcţionează a aplicaţiei proces.

Aplicaţia proces rămâne la nivel de sugestie până când decideţi să creaţi un instantaneu, care înregistrează starea articolelor de bibliotecă dintr-o aplicaţie proces sau pistă la un anumit moment. Tipic, realizaţi un instantaneu de fiecare dată când sunteţi pregătit să testaţi integrarea sau vreţi să instalaţi aplicaţia proces pe un server centru de proces sau un server de proces pentru testare, intermediere sau producţie.

Notă: Sugestia este un instantaneu special; este singurul tip de instantaneu în care puteţi modifica conţinuturile, însă îl puteţi rula doar pe serverul Process Center. Nu puteţi instala o sugestie pe un server de proces.

Implicit, fiecare aplicaţie proces are o singură pistă, denumită Principală. Dacă vreţi să permiteţi dezvoltarea paralelă pe o aplicaţie proces, puteţi crea piste suplimentare. Aceste subdivizii opţionale din aplicaţia proces ţin modificările izolate. De exemplu, imaginaţi-vă că compania dumneavoastră este în proces de schimbare de imagine; în timpul acestei tranziţii, aplicaţiile proces curente trebuie să fie menţinute în timp ce sunt dezvoltate versiuni noi bazate pe identitatea actualizată a corporaţiei. În această situaţie, o echipă poate face modificări minore pe versiunea curentă a aplicaţiei proces (în pista Principală) în timp ce altă echipă construieşte o versiune nouă a aplicaţiei proces într-o pistă separată.

Truse de unelte pentru aplicaţii de proces

Trusele de unelte sunt containere care stochează articole de bibliotecă (de exemplu, BPD-uri) pentru reutilizare de către aplicaţiile proces sau de alte truse de unelte. Aplicaţiile de proces pot împărţi articole de bibliotecă din una sau mai multe truse de unelte, şi trusele de unelte pot împărţi articole din alte truse de unelte. Dacă aveţi acces la o trusă de

unelte, puteţi crea o dependenţă pe ea şi utiliza articolele de bibliotecă ale acelei truse de unelte în aplicaţia dumneavoastră proces.

Aplicaţii de proces şi aplicaţii la nivel operaţional

O aplicaţie de proces are o aplicaţie la nivel de afaceri BLA (business level application), care se comportă ca un container pentru aplicaţia de proces şi aseturile sale (aseturile includ lucruri precum modele de monitor, module SCA, truse de unelte şi biblioteci). Fiecare instantaneu de aplicaţie de proces are propriul său BLA. Multe dintre taskurile de administrare ale unui instantaneu (de exemplu, oprirea sau pornirea acestuia într-un server de producţie) sunt efectuate la nivelul BLA, permiţâmd administrarea mult mai rapidă şi mai simplă a instantaneului şi a tuturor aseturilor sale.

Rularea şi depanarea proceselor

Utilizând Inspectorul, dezvoltatorii individuali pot rula procese şi servicii pe Process Center Runtime-ul server sau la distanţă Process Server.

Inspectorul din IBM Process Designer este cheia pentru o abordare repetată pentru a procesa dezvoltarea. O echipă întreagă de dezvoltare poate utiliza Inspectorul pentru a demonstra proiectarea şi implementarea procesului curent în sesiunile playback. Sesiunile playback ajută capturarea informaţiilor importante de la diferiţi deţinători de acţiuni (stakeholder) într-un proces, precum management, utilizatori finali şi analişti business. Aplicarea unei abordări iterative dezvoltării procesului asigură îndeplinirea scopurile şi nevoile tuturor celor implicaţi de către aplicaţiile de proces.

Inspectorul din IBM Process Designer include mai multe unelte ce vă permit să realizaţi taskuri cum ar fi următoarele din mediile dumneavoastră configurate:

Task	Descriere
Gestionare instanţe ale proceselor	Când rulaţi un proces, puteţi vizualiza toate instanţele rulate anterior sau rulate curent pe serverele IBM Business Process Manager din mediul dumneavoastră. Puteţi gestiona instanţele care rulează prin oprirea şi repornirea lor, de exemplu. Puteţi gestiona de asemenea instanţele rulate anterior prin filtrarea sau ştergerea înregistrărilor specifice.
Executarea pas cu pas şi depanarea unui proces	Pentru o instanţă selectată, vedeţi pasul care se execută momentan şi apoi mergeţi înainte prin proces, evaluând executarea procesului pas cu pas. O afişare arbore a procesului combinată cu indicatorii numiţi jetoane într-o diagramă proces face mai uşoară înţelegerea locului în care vă aflaţi în proces. Aveţi de asemenea avantajul cunoaşterii variabilelor utilizate în fiecare pas şi valorile lor corespunzătoare (unde este aplicabil).

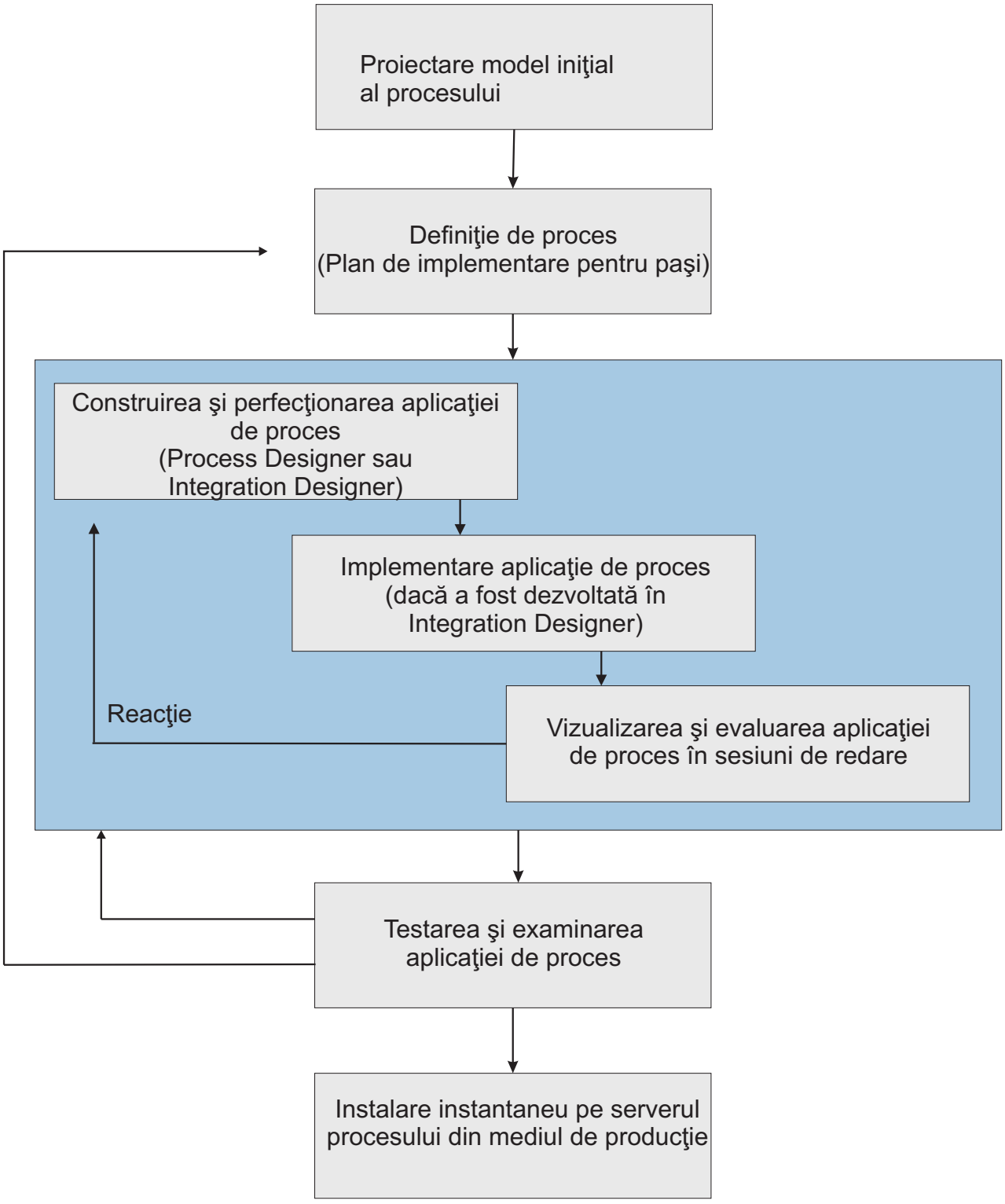
Dacă lucraţi în IBM Integration Designer, puteţi utiliza Inspectorul dacă proiectul dmneavoastră este asociat cu o aplicaţie de proces. Puteţi avea de asemenea alte unelte de depanare sau test disponibile pentru dumneavoastră. Pentru informaţii suplimentare despre aceste unelte Integration Designer, vedeţi "Module de testare" şi "Utilizarea depanatorului de integrare pentru determinarea problemei" în legăturile înrudite.

Instalarea şi gestionarea aplicaţiilor de proces

Ciclul de viaţă al aplicaţiei de proces include instalarea, administrarea şi dezimplementarea instantaneelor. Consideraţiile de versionare sunt, de asemenea, parte componentă a ciclului de viaţă.

Atunci când dezvoltaţi procese, puteţi profita de abordarea iterativă suportată de uneltele din cadrul Process Designer. Procesele evoluează în timp, iniţial de la o stare de dezvoltare şi apoi la testare şi apoi la producţie. Chiar şi în producţie, procesele dvs. îşi pot continua desfăşurarea din cauza necesităţii modificării. Este important să fiţi pregătit pentru ciclul de viaţă în curs de desfăşurare al proceselor dumneavoastră şi vă va ajuta să proiectaţi efectiv de la început.

Figura următoare ilustrează o abordare iterativă a dezvoltării procesului.



O configuraţie tipică a Managerului procesului operaţional include trei medii pentru a suporta dezvoltarea şi instalarea eventuală a proceselor dumneavoastră.

Mediu	Descriere
Dezvoltare	Construiţi şi rafinaţi aplicaţiile de proces în IBM Process Designer. Creaţi modelele de proces şi implementaţi paşii în acele modele utilizând Designer. Utilizând Inspector, demonstraţi progresul dezvoltării dumneavoastră în sesiuni de simulare, astfel încât să puteţi evalua rapid şi să vă puteţi rafina prototipul. Utilizând consola Process Center, instalaţi aplicaţiile de proces pe servere de testare sau de producţie ale procesului.
Test	Utilizând consola Process Center, instalaţi aplicaţiile de proces în Process Server din mediul dumneavoastră de testare pentru a implementa teste formale de siguranţă a calităţii. Puteţi utiliza Inspector pentru a ajuta la verificarea şi rezolvarea problemelor.
Producţie	Când toate problemele raportate la testarea formală sunt rezolvate, utilizaţi consola Process Center pentru a instala aplicaţiile de proces în Process Server din mediul dumneavoastră de producţie. Puteţi utiliza Inspector pentru a investiga şi rezolva orice probleme raportate în mediul dumneavoastră de producţie.

Dacă doriţi să testaţi, instalaţi sau să administraţi un instantaneu aplicaţie de proces care are conţinut IBM Business Process Manager Advanced sau un model IBM Business Monitor, utilizatorul sau grupul de care aparţineţi trebuie să fie alocat rolului de securitate administrativă Configurator, Operator *şi* Implementator. Dacă în prezent nu sunteţi alocat tuturor acestor roluri, faceţi clic pe **Utilizatori şi Grupuri** în consola administrativă WebSphere pentru a modifica rolurile utilizator sau grup. Consultaţi "Roluri de securitate administrative" în legăturile înrudite.

Strategii de eliberare şi instalare

Pentru a vă asigura că aplicaţiile de proces pe care le implementaţi şi instalaţi îndeplinesc standardele de calitate ale organizaţiei dumneavoastră, consideraţi definirea unei strategii de eliberare şi instalare. Când aţi identificat obiectivele şi cerinţele necesare ediţiei şi instalării noii şi actualizatei aplicaţii de proces, puteţi automatiza procesele necesare pentru a aproba şi a lansa programele.

De exemplu, poate doriţi să direcţionaţi un proces către mai mulţi manageri diferiţi care folosesc structuri diferite de raportare în cadrul organizaţiei dumneavoastră. Doar după ce fiecare manager renunţă la procesul nou sau actualizat, acesta poate fi instalat în mediul dumneavoastră de producţie şi prezentat utilizatorilor finali. Puteţi crea şi implementa paşii implicaţi într-o asemenea examinare a IBM Business Process Manager Advanced pentru a vă asigura că toate indicaţiile corporaţiei au fost îndeplinite şi că aveţi semnăturile necesare. Ultimul pas al examinării ar putea fi notificarea echipei de IT asupra faptului că aplicaţia de proces aprobată este gata pentru instalare.

Crearea, accesarea şi încorporarea serviciilor

Procesele operaţionale utilizează adesea servicii care furnizează funcţiile necesare pentru procesul operaţional. Aceste servicii apar într-o diagramă de procese operaţionale ca activitate sau pas. De exemplu, un serviciu din Process Designer poate invoca un serviciu web extern sau poate invoca un serviciu complex şi automatizat proiectat în Integration Designer.

Accesarea serviciilor externe unei aplicaţii

Acest scenariu prezintă diferite moduri de a accesa servicii ce sunt externe unei aplicaţii şi furnizează taskuri de nivel înalt pentru accesarea acestor servicii externe.

Notă: Acest scenariu este aplicabil pentru WebSphere Enterprise Service Bus şi IBM Business Process Manager. Modulele de mediere pot fi implementate la WebSphere Enterprise Service Bus şi IBM Business Process Manager. Modulele pot fi implementate la IBM Business Process Manager.

Într-o aplicaţie operaţională integrată, *serviciile operaţionale* interacţionează între ele pentru a furniza o funcţie necesară. Un serviciu operaţional realizează o funcţie sau task repetabile ce contribuie la realizarea unui scop operaţional. Dar munca de a localiza un serviciu şi conectarea la el nu este înrudită cu funcţia operaţională. Separarea funcţiei operaţionale de taskul de a gestiona conexiuni la servicii oferă flexibilitate unei soluţii.

Interacţiunea cu serviciul începe când un *solicitant de servicii* trimite o cerere unui *furnizor de servicii* să realizeze o funcţie operaţională. Această cerere este trimisă sub forma unui *mesaj*, ce defineşte funcţia ce urmează să fie realizată. Furnizorul de servicii efectuează funcţia cerută şi trimite rezultatul într-un mesaj solicitantului de servicii. În mod

normal, mesajele trebuie să fie procesate pentru a permite serviciilor să schimbe date şi să implementeze alte funcţii IT de nivel scăzut ce sunt independente de funcţiile şi datele operaţionale. De exemplu, rutarea, conversia protocoalelor, transformarea, reîncercarea unei invocări eşuate şi invocarea unui serviciu dinamic. Această procesare este cunoscută ca *mediere*.

Există două tipuri de module în IBM Integration Designer; module (sau module de Business Integration), ce sunt în primul rând proiectate să conţină logică operaţională (precum procese operaţionale, reguli operaţionale şi maşini cu stări operaţionale) şi module de mediere, ce implementează fluxuri de mediere. Deşi există o suprapunere de funcţii între cele două tipuri de module, în general, recomandăm ca logica operaţională să fie izolată în module operaţionale şi logica de mediere să fie realizată de module de mediere.

Însă nu există întotdeauna o separare clară între logica operaţională şi cea de mediere. În aceste cazuri, luaţi în considerare cantitatea de *stare* sau date din variabile ce vor trebui să fie procesate între invocări de servicii. În general, dacă este nevoie de puţină procesare de stare sau deloc, luaţi în considerare utilizarea unei componente de flux de mediere. Dacă trebuie să memoraţi starea între invocări de servicii sau dacă aveţi date ce vor trebui să fie stocate în variabile şi procesate, consideraţi utilizarea unei componente de proces operaţional. De exemplu, dacă apelaţi mai multe servicii şi înregistraţi informaţiile returnate din fiecare, astfel încât după invocarea tuturor serviciilor, doriţi să faceţi procesări mai departe cu datele returnate, utilizaţi un proces operaţional unde puteţi aloca uşor informaţiile returnate variabilelor. Cu alte cuvinte, când aveţi prea multe stări aţi traversat linia spre logica operaţională. Secţiunile următoare ajută la clarificarea acestei indicaţii.

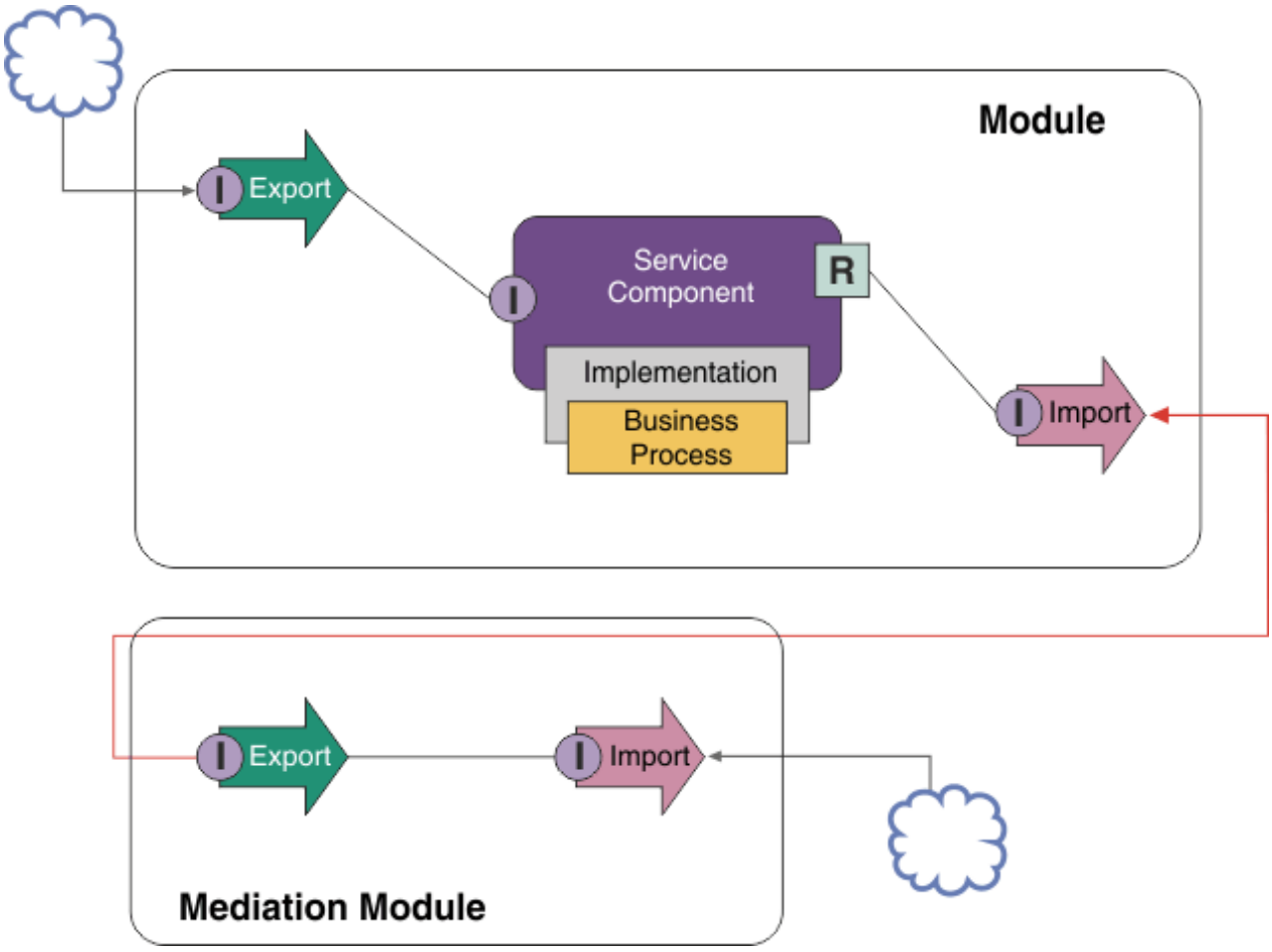
Nu există un singur scenariu de integrare şi nu există un răspuns greşit din punct de vedere tehnic. Indicaţiile discutate aici sunt un exerciţiu bun pentru a permite flexibilitatea şi reutilizarea şi sunt prezentate pentru consideraţiile

dumneavoastră. Ca de obicei, trebuie să consideraţi cu grijă beneficiile şi dezavantajele implementării acestor tipare pentru aplicaţie dumneavoastră de Business Integration. Haideţi să considerăm câteva situaţii.

Accesarea unei componente SCA

Un exemplu elementar de accesare a unui serviciu este când un import apelează altă componentă SCA, fără a necesita vreo transformare de date. Chiar şi în această situaţie, puteţi accesa serviciul extern dintr-un modul de mediere, decât să-l accesaţi direct dintr-un modul operaţional. Aceasta ar permite flexibilitate în viitor, pentru modificarea punctului final sau QoS (calitatea serviciului) sau guvernării (de exemplu, adăugarea jurnalizării) fără a afecta componentele operaţionale ce consumă serviciul. Acest tipar arhitectural este cunoscut ca "separare de interese".

Înainte să vă decideţi să implementaţi acest tipar, cântăriţi beneficiile tiparului contra potenţialelor efecte ale regiei introduse de alt modul. Dacă principala dumneavoastră cerinţă este flexibilitatea şi veţi face modificări frecvente asupra serviciilor accesate, luaţi în considerare utilizarea unui modul separat după cum este arătat aici. Dacă performanţa este cea mai importantă şi sunteţi dispuşi să actualizaţi şi să reimplementaţi logica operaţională, luaţi în considerare utilizarea unui singur modul.



Iată taskurile de nivel înalt pentru realizarea acestui exemplu.

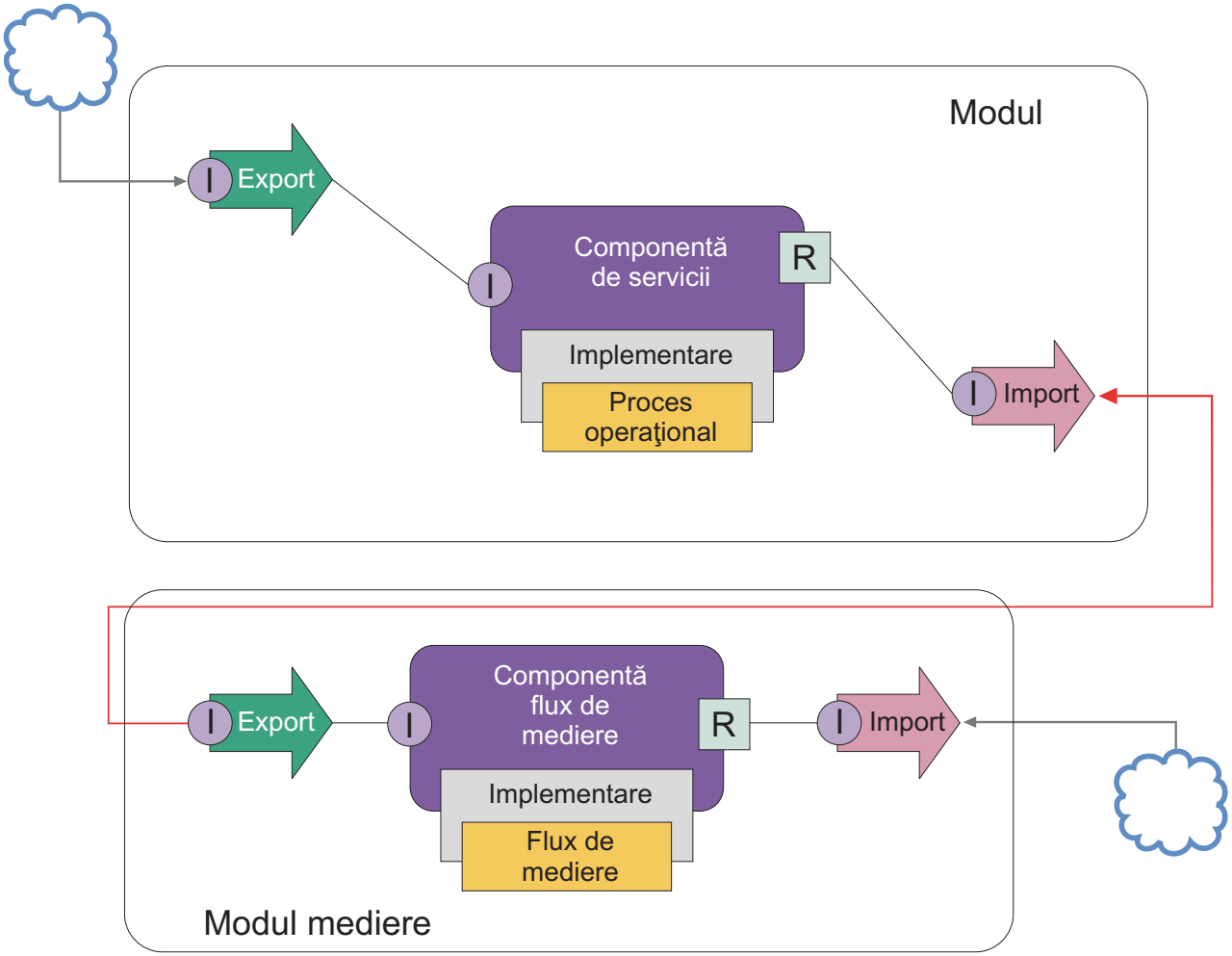
1. Crearea unui modul de mediere. Pentru instrucţiuni pas-cu-pas, vedeţi Crearea modulelor de mediere .
2. În modulul de mediere, creaţi un import cu legarea corespunzătoare pentru serviciul extern pe care doriţi să-l accesaţi. Pentru instrucţiuni pas-cu-pas, vedeţi Crearea importurilor. Pentru informaţii suplimentare despre legări, vedeţi Legări

3. Creaţi un export şi daţi-i aceeaşi interfaţă ca şi importului. Pentru instrucţiuni pas-cu-pas, vedeţi Crearea exporturilor.
4. Generarea unei legări SCA pentru export. Pentru instrucţiuni pas-cu-pas, vedeţi Generarea legărilor SCA
5. În montajul modulului de mediere, cablaţi exportul la import. Salvaţi modulul de mediere.
6. Crearea unui modul. Pentru instrucţiuni pas-cu-pas, vedeţi Crearea unui modul pentru servicii operaţionale
7. Adăugaţi un export şi o componentă.
8. În vizualizarea Business Integration, trageţi exportul pe care l-aţi creat în modulul de mediere (la pasul 4) în asamblarea modulului. Va fi creat un import cu aceeaşi legare ca exportul.
9. Cablaţi exportul de componentă şi componenta de import.
10. Adăugaţi implementarea componentei. Pentru informaţii despre tipuri de implementare, vedeţi Implementări

Mai târziu, puteţi adăuga logică de mediere precum logare sau direcţionare către modulul de mediere fără a afecta modulul operaţional.

Adăugarea medierii

Uneori nu este suficient să invocaţi pur şi simplu un serviciu extern. Uneori este nevoie să procesaţi întâi, prin adăugarea unui modul de mediere ca un intermediar între solicitantul şi furnizorul de servicii.



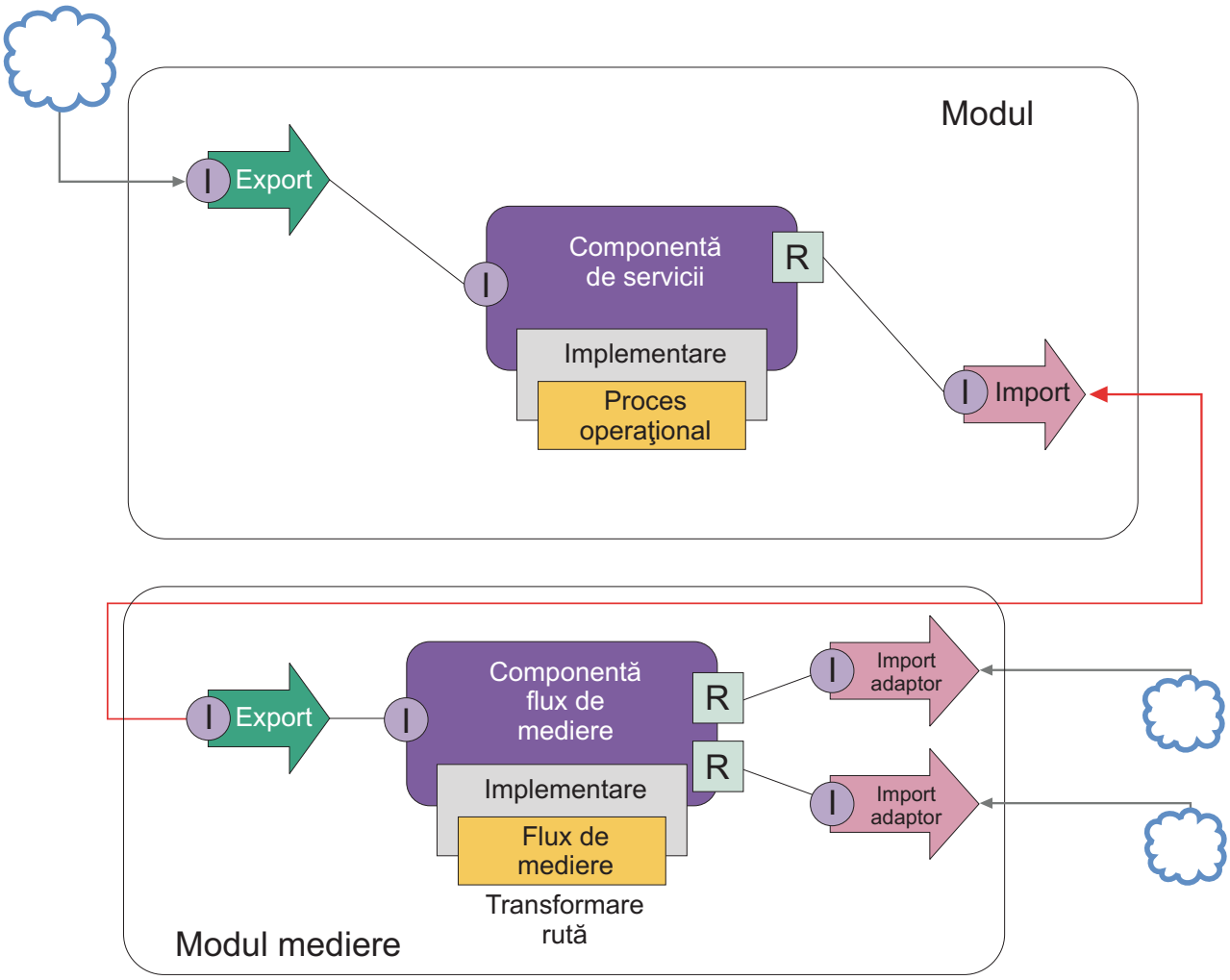
Acestea sunt câteva din funcţiile pe care le-ar efectua fluxul de mediere intermediară:

- Setarea anteturilor de protocol. Pentru informaţii suplimentare, vedeţi subiectul Conversie de protocoale din centrul de informare WebSphere Enterprise Service Bus.
- Transformarea interfeţei sau parametrilor prin utilizarea unei Mapări de obiecte business sau a unei primitive Mapping transformare. Transformarea mesajelor
- Selectarea unui anumit serviciu dintr-o listă statică, prin utilizarea unei primitive Filtre de mesaje.Filtru de mesaje
- Invocarea mai multor servicii pentru agregarea rezultatelor, prin utilizarea primitivelor Fan Out şi Fan In.Agregarea şi difuzarea mesajelor
- Tratarea eşecurilor invocării de servicii prin reîncercarea aceluiaşi serviciu sau invocarea unui serviciu diferit, prin utilizarea unei primitive Service Invoke. Reîncercarea unei invocări de serviciu eşuate
- Rutare dinamică prin alegerea cărui serviciu să folosiţi în timpul rulării, mai degrabă decât în timpul integrării, ceea ce permite serviciilor să fie cuplate mai liber şi afacerilor să reacţioneze mai rapid la modificări. Noi servicii pot fi adăugate fără a atinge modulele ce au fost implementate pe mediul runtime. Rutarea dinamică este cel mai puternică atunci când este folosită cu un registru, ceea ce necesită ca primitiva de mediere Endpoint Lookup să fie folosită. Selectarea dinamică a punctelor finale

Accesarea sistemelor EIS (Enterprise Information System)

Serviciile şi artefactele de pe sisteme externe pot fi importate în Integration Designer. Un vrăjitor descoperă aplicaţii şi date pe un EIS (Enterprise Information System) şi vă permite să generaţi servicii din aplicaţiile şi datele descoperite. Artefactele generate sunt interfeţe şi obiecte business, ce pot fi folosite de componente dintr-un modul.

Utilizarea unui modul de mediere intermediar între un modul şi un sistem gazdă îl face mai reutilizabil. În exemplul de mai jos, un flux de mediere este folosit pentru rutarea la sistemul gazdă corect şi pentru transformarea datelor în formatul cerut de sistemul gazdă.



Iată taskurile de nivel înalt pentru acest exemplu:

1. Utilizarea vrăjitorului de servicii externe pentru conectarea la sistemul gazdă. Utilizarea vrăjitorului de servicii externe pentru a accesa servicii externe urmează un tipar similar indiferent de adaptorul pe care îl folosiţi. Pentru informaţii despre cum să folosiţi vrăjitorul de servicii externe, vedeţi Tipar de accesare a serviciilor externe cu adaptoare
2. Crearea unui modul. Pentru instrucţiuni pas-cu-pas, vedeţi Crearea unui modul pentru servicii operaţionale
3. Adăugarea unui export, unei componente şi unui import cu legare SCA. Pentru informaţii suplimentare, vedeţi Apelarea serviciilor
4. Adăugarea unei interfeţe exportului şi cablarea exportului la componentă.
5. Adăugaţi implementarea componentei. În implementare, setaţi o proprietate ce indică ce serviciu gazdă va fi accesat. Pentru informaţii despre tipuri de implementare, vedeţi Implementări
6. Crearea unui modul de mediere cu un export ce are o legare SCA şi aceeaşi interfaţă ca importul modulului pe care l-aţi creat la pasul 2.
7. Cablarea exportului la o componentă de flux de mediere.
8. Crearea unui import pentru fiecare sistem gazdă pe care doriţi să-l accesaţi, folosind adaptorul de ieşire corespunzător din paleta editorului de montaj.
9. Cablarea componentei de flux de mediere la importuri.
10. Implementarea componentei de flux de mediere. Folosiţi o primitivă Message Filter pentru a alege importul pe baza unui set de proprietăţi din logica operaţională şi folosiţi o primitivă Mapping pentru fiecare import adaptor. Filtru mesaj.

11. În modul, selectaţi exportul modulului de mediere ca serviciul de importat în modul. Pentru informaţii pas-cu-pas, vedeţi Invocarea unui serviciu dintr-un alt modul.

Mai târziu, puteţi face o modificare precum adăugarea unui adaptor sau modificarea unui adaptor pentru a indica spre un sistem gazdă diferit, cu impact minim asupra logicii operaţionale.

Accesarea sistemelor de mesagerie

Pentru ca modulul dumneavoastră SCA (service component architecture) să comunice cu un client de mesagerie JMS, MQ sau MQ JMS existent, trebuie să creaţi interfeţe, obiecte business şi legări pentru importuri şi exporturi. Vedeţi Maparea unui mesaj la o interfaţă SCA.

Fluxurile de mediere folosesc mesaje, ce oferă acces la informaţii de context şi antet în plus faţă de obiectele business. Dacă doriţi acces la informaţii de antet JMS sau o proprietate JMS personalizată, folosiţi un flux de mediere. Dacă integraţi cu un sistem MQ şi doriţi să accesaţi informaţii de antet MQ, folosiţi un flux de mediere.

Crearea sau apelarea unui serviciu web

Serviciile Web sunt aplicaţii auto-conţinute ce realizează funcţii operaţionale, de la o simplă interogare la interacţiuni de procese operaţionale complexe. Puteţi apela un serviciu web existent sau puteţi dezvolta un nou serviciu web care să se potrivească cu necesităţile dumneavoastră. Acest scenariu va descrie paşii şi vă va ghida către informaţiile suplimentare.

Deşi s-ar putea să nu vă creaţi toate serviciile de la zero folosind IBM Integration Designer, unele dintre serviciile dumneavoastră vor fi într-adevăr create astfel. Când lucraţi cu editorul de montaje şi cu editorul de procese operaţionale pentru a monta servicii într-un proces operaţional, este posibil să descoperiţi că unele servicii lipsesc. De aceea, poate fi util să creaţi acele servicii lipsă folosind unelte IBM Integration Designer. Opusul este de asemenea adevărat - după ce aţi creat un nou proces, puteţi decide că ar fi util să expuneţi toate sau un subset din operaţiile de proces ca servicii pentru ca alţii să le consume.

Notă: Acest scenariu se aplică utilizatorilor de IBM Integration Designer pentru IBM Process Server şi WebSphere Enterprise Service Bus.

Există mai multe motive pentru dezvoltarea serviciilor web folosind IBM Integration Designer:

- Crearea serviciilor în IBM Integration Designer vă permite să implementaţi serviciul folosind reguli operaţionale.
- Dezvoltarea în IBM Integration Designer vă permite să dezvoltaţi un serviciu Java™ şi să îl expuneţi atât ca serviciu web, cât şi prin SCA.
- Maparea interfeţelor fără a fi nevoie să codaţi este un avantaj. Puteţi scoate toate mapările de date din codul Java lăsând un simplu program Java de cutie neagră pentru dezvoltatorul Java.
- IBM Integration Designer afişează toate serviciile şi relaţiile într-un singur loc.
- Abilitatea de a refactoriza va ajuta, de asemenea, la dezvoltarea de servicii web care utilizează IBM Integration Designer.

Reţineţi că serviciile web nu ar trebui să fie privite ca o soluţie la toate problemele dumneavoastră de integrare. Totuşi, la fel ca orice altă tehnologie sau abordare arhitecturală, există avantaje inerente la utilizarea serviciilor web în locul potrivit şi la momentul potrivit.

Exporturi, importuri şi legări

IBM Integration Designer vă permite să importaţi servicii web standard şi să utilizaţi aceste servicii în aplicaţiile dumneavoastră compuse.

În IBM Integration Designer, utilizaţi editorul de montare pentru a dezvolta servicii. Urmaţi procesul standard pentru a crea module, module de mediere, biblioteci şi componente. Apoi, puteţi utiliza exporturi, importuri şi legări pentru a partaja şi accesa acele servicii. Paşii pentru acele taskuri elementare sunt afişaţi mai jos şi legăturile conduc către informaţii mai detaliate pentru fiecare task.

Puteţi utiliza oricare dintre cele două legături de servicii web - o legare de serviciu web sau o legare HTTP. O legare de serviciu web oferă o specificaţie pentru transmiterea mesajelor către şi de la un serviciu web. Uneltele vă ajută să generaţi în mod automat o legare de serviciu web. O legare HTTP este un protocol cerere-şi-răspuns standard între clienţi şi server după cum este definit de protocolul HTTP publicat de W3C (World Wide Web consortium). Va trebui să furnizaţi câteva informaţii iniţiale de configurare a legării dacă folosiţi o legare HTTP.

1. Crearea unui export pentru a publica serviciul modulului spre a fi folosit de alte module.
2. Generarea unei legări pentru export.
   - Generarea unei legări de serviciu web pentru export.
   - Generarea unei legări de export HTTP.
3. Creaţi un import pentru a apela un serviciu existent ce nu face parte din serviciul pe care îl montaţi.
   - Generarea unei legări de serviciu web pentru import.
   - Generarea unei legări de import HTTP.

Citiţi subiectul legat pentru Invocarea serviciului web din paginile JavaServer.

Capabilităţi de dezvoltare pentru serviciile Web

La deschiderea unui editor asociat cu procesul de creaţie a serviciilor web, aţi putea vedea fereastra Confirmare activare care afişează următoarele informaţii:

```
Această acţiune necesită activare "Implementare servicii web".
Activaţi capabilitatea necesară?
```

IBM Integration Designer oferă o funcţie de filtrare cunoscută drept *capabilităţi*. În setările Preferinţelor, funcţiile şi uneltele sunt clasificate în capabilităţi şi puteţi activa sau dezactiva categorii ale capabilităţilor sau funcţiile de subset ale oricărei categorii. Vedeţi Capabilităţi pentru informaţii suplimentare.

Aflaţi mai multe despre conceptele cheie

Utilizaţi această secţiune ca un punct de plecare pentru investigarea tehnologiilor utilizate în şi de către IBM Business Process Manager.

Crearea scenariilor

Folosiţi scenarii pentru a înţelege şi lucra cu componentele şi produsele din familia de management al proceselor operaţionale.

Versionarea

Ciclul de viaţă al unei aplicaţii de proces începe cu crearea aplicaţiei de proces şi continuă cu un ciclu de actualizare, implementare, co-implementare, dezimplementare şi arhivare a aplicaţiei de proces. *Versionarea* este un mecanism folosit pentru gestionarea ciclului de viaţă al aplicaţiei de proces prin identificarea în mod unic a versiunilor individuale ale aplicaţiei de proces.

Modul în care versionarea funcţionează în IBM Business Process Manager depinde de ceea ce implementaţi –o aplicaţie de proces, implementată din magazie în IBM Process Center sau o aplicaţie de întreprindere implementată direct din IBM Integration Designer.

Aplicaţiile de proces şi trusele de unelte pe care le implementaţi într-un mediu de runtime din Process Center sunt versionate în mod implicit. Pentru aplicaţiile de întreprindere, puteţi alege să versionaţi modulele şi bibliotecile în IBM Integration Designer.

În plus, puteţi crea versiuni pentru un task uman sau pentru o stare a unei maşini, astfel încât să poate exista simultan mai multe versiuni ale taskului sau ale stării maşinii în mediul de runtime.

Versionarea aplicaţiilor de proces

Versionarea furnizează abilitatea, pentru mediul runtime, de a identifica instantaneele din ciclul de viaţă al unei aplicaţii proces şi de a fi capabil să ruleze simultan mai multe instantanee pe un server de proces.

Pentru a înţelege cum sunt versionate aplicaţiile proces, este important să vă amintiţi că o aplicaţie proces este un container care reţine diverse artefacte utilizate în sau de către aplicaţia proces (de exemplu, modele de procese sau BPD-uri, referinţe trusă de unelte, servicii, piste sau modele de monitor). Orice versionare este făcută la acest nivel de container, nu la nivelul artefactelor individuale. Pentru aplicaţii proces, aceasta înseamnă că versionarea apare când realizaţi un instantaneu.

Puteţi compara instantaneele pentru a determina diferenţele dintre versiuni. De exemplu, dacă un dezvoltator a corectat o problemă cu un serviciu şi a realizat un instantaneu al aplicaţiei de proces sau al trusei sale de unelte în acel moment, şi apoi un alt dezvoltator a făcut mai multe modificări suplimentare la acelaşi serviciu şi a realizat un nou instantaneu, managerul de proiect poate compara cele două instantanee pentru a determina ce modificări au fost făcute, unde şi de către cine. Dacă managerul de proiect a decis că modificările suplimentare la serviciu nu au valoare, managerul de proiect poate reveni la instantaneul corecţiei iniţiale.

Puteţi rula versiuni diferite (instantanee) ale unei aplicaţii de proces simultan pe un server; când instalaţi un instantaneu nou, fie înlăturaţi originalul fie lăsaţi-l să ruleze.

Context de versiune

Fiecare instantaneu are metadate unice pentru a identifica versiunea (referite ca şi context de versiune). Aţi alocat acel identificator, dar IBM recomandă folosirea unui sistem pe trei cifre în versiunea numerică în formatul <major>.<minor>.<service>. Vedeţi toate subiectele despre convenţiile de numire pentru descrieri mai detaliate ale acestei scheme de versionare.

IBM Business Process Manager asignează un spaţiu de nume global pentru fiecare aplicaţie de proces. Spaţiul de nume global este în mod specific fie sugestia aplicaţiei proces, fie un instantaneu de aplicaţie proces particular. Numele de versiune utilizat de către server nu poate fi mai lung de şapte caractere, astfel încât numele alocat este un acronim care utilizează caractere din numele instantaneului pe care l-aţi alocat. Acronimele instantaneului sunt identice cu numele instantaneului conform stilului IBM VRM recomandat şi nu sunt mai lungi de şapte caractere. De exemplu, un nume de instantaneu de 1.0.0 va avea acronimul 1.0.0 şi un nume de instantaneu of 10.3.0 va avea acronimul 10.3.0. Acronimul instantaneului va fi garantat ca şi unicitate în contextul aplicaţiei de proces din domeniul serverului Process Center. Pentru acest motiv, nu puteţi edita acronimul instantaneului.

Considerente de versionare pentru aplicaţii proces în clustere multiple

Puteţi instala aceeaşi versiune de aplicaţie proces la clustere multiple din aceeaşi celulă. Pentru a face diferenţa între aceste instalări multiple ale aceleiaşi versiuni a aplicaţie proces, creaţi un instantaneu pentru fiecare instalare şi includeţi un ID unic de celulă în numele instantaneului (de exemplu, v1.0_cell1_1 şi v1.0_cell1_2). Fiecare instantaneu este o versiune nouă a aplicaţiei proces (dintr-o perspectivă de gestiune ciclu de viaţă pură), dar conţinutul şi funcţia sunt la fel.

Când instalaţi o aplicaţie proces într-un cluster, este realizată o sincronizare automată a nodurilor.

Considerente de versionare pentru truse de unelte Process Designer

Amintiţi-vă că instantaneele de aplicaţii proces sunt realizate în mod tipic când sunteţi pregătit să testaţi sau să instalaţi. Instantaneele de truse de unelte sunt însă realizate în mod tipic când sunteţi pregătit ca acea trusă de unelte să fie utilizată de aplicaţii proces. După, dacă doriţi să actualizaţi trusa de unelte, trebuie să faceţi un alt instantaneu al "sugestiei" atunci când sunteţi pregătit şi apoi deţinătorii aplicaţiilor de proces şi ai truselor de unelte pot decide dacă doresc să treacă la noul instantaneu.

Versionarea modulelor şi bibliotecilor

Dacă un modul sau o bibliotecă este într-o aplicaţie de proces sau într-o trusă de unelte, acesta continuă ciclul de viaţă al aplicaţiei de proces sau trusei de unelte (versiuni, instantanee, urme şi aşa mai departe). Numele modulelor şi bibliotecilor trebuie să fie unice în cadrul domeniului unei aplicaţii de proces sau a unei truse de unelte.

Acest subiect descrie versionarea modulelor şi a bibliotecilor care sunt folosite împreună cu aplicaţiile de proces. Reţineţi totuşi că în cazul în care implementaţi modulele direct din IBM Integration Designer pe Process Server, puteţi continua să urmaţi procedura de asignare a numerelor de versiune modulelor în timpul implementării, aşa cum este descris în "Crearea modulelor şi bibliotecilor cu număr de versiune".

Un modul sau o bibliotecă care este asociată cu IBM Process Center trebuie să aibă bibliotecile dependente în aceeaşi aplicaţie de proces sau într-o trusă de unelte dependentă.

Următorul tabel listează selecţiile pe care le puteţi face în editorul de dependenţe din IBM Integration Designer, atunci când o bibliotecă este asociată cu o aplicaţie de proces sau o trusă de unelte:

Tabela 3. Dependenţele pentru Modul, Aplicaţie de proces sau Trusa de unelte şi bibliotecile globale

Domeniu bibliotecă	Descriere	Poate depinde de . . .
Modul	Pe server există câte o copie a acestei biblioteci pentru fiecare modul care o foloseşte.	O bibliotecă din domeniul modulului poate depinde de toate tipurile de biblioteci.
Aplicaţie de proces sau Trusă de unelte	Biblioteca este partajată între toate modulele din domeniul aplicaţiei de proces a trusei de unelte. Această setare are efect în cazul în care implementarea se realizează prin IBM Process Center. În cazul în care implementarea are loc în afara IBM Process Center, biblioteca este copiată în fiecare modul. **Notă:** Bibliotecile create în IBM Integration Designer versiunea 8 au un nivel de partajare de **Aplicaţie de proces sau Trusă de unelte** în mod implicit.	O bibliotecă de acest tip poate depinde doar de bibliotecile globale.
Global	Biblioteca este partajată între toate modulele care rulează.	O bibliotecă globală poate depinde doar de alte biblioteci globale. **Notă:** Trebuie să configuraţi o bibliotecă partajată WebSphere pentru a putea implementa biblioteca globală. Vedeţi "Dependenţele modulelor şi bibliotecilor " pentru informaţii suplimentare.

Module şi biblioteci asociate cu aplicaţii de proces sau cu truse de unelte

Nu aveţi nevoie să versionaţi module şi biblioteci asociate cu aplicaţii de proces sau cu truse de unelte.

Modulele şi bibliotecile asociate cu aplicaţii de proces sau cu truse de unelte nu au nevoie să fie versionate. De fapt, nu puteţi crea o versiune a unui modul sau a unei biblioteci asociate cu aplicaţii de proces sau cu truse de unelte în editorul de dependenţe. Modulele şi bibliotecile asociate cu aplicaţiile de proces sau cu trusele de unelte utilizează instantanee, o funcţie în Process Center, pentru a realiza acelaşi rezultat ca o versiune.

Bibliotecile, asociate cu aplicaţiile de proces sau cu trusele de unelte, nu vor avea un număr de versiune necesar în secţiunea Biblioteci a editorului de dependenţe, deoarece nu este necesară nicio versiune.

Convenţii de numire

O convenţie de numire este folosită pentru a diferenţia diversele versiuni ale unei aplicaţii de proces pe măsură ce se mută prin ciclul de viaţă de actualizare, implementarea, co-implementare, dezimplementare şi arhivare.

Această secţiune vă oferă convenţiile care sunt utilizate pentru a identifica în mod unic versiunile unei aplicaţii de proces.

Un *context de versiune* este o combinaţie de acronime care descriu în mod unic o aplicaţie de proces sau o trusă de unelte. Fiecare tip de acronim are o convenţie de numire. Acronimul este limitată la o lungime maximă de şapte caractere din setul de caractere [A-Z0-9_], cu excepţia acronimului instantaneului, care poate include un punct.

- Acronimul aplicaţiei de proces este creat odată cu aplicaţia. Acesta poate avea o lungime maximă de şapte caractere.
- Acronimul instantaneului este creat în mod automat la crearea instantaneului. Acesta poate avea o lungime maximă de şapte caractere.

  Dacă numele instantaneului îndeplineşte criteriile pentru un acronim valid de instantaneu, numele şi acronimul instantaneului vor fi identice.

  Notă: Atunci când utilizaţi funcţia de rutare dependentă de versiune pentru componenta fluxului de mediere, denumiţi-vă instantaneul, astfel încât să fie în concordanţă cu schema <*versiune*>.<*ediţie*>.<*modificare*> (de exemplu, **1.0.0**). Deoarece acronimul instantaneului este limitat la şapte caractere, valorile cifrelor sunt limitate la un maxim de cinci cifre (cinci cifre plus două perioade). Prin urmare, trebuie să aveţi grijă atunci când câmpurile cifre sunt incrementate deoarece orice depăşeşte primele şapte caractere este trunchiat.

  De exemplu, un nume de instantaneu **11.22.33** rezultă într-un acronim de instantaneu egal cu **11.22.3**.
- Acronimul de pistă este generat în mod automat din primul caracter al fiecărui cuvânt al numelui de pistă. De exemplu, o pistă nouă ce are numele **My New Track** ar rezulta în acronimul cu valoarea **MNT**.

  Numele implicite pentru urmărire şi acronim sunt **Main**. Implementarea pe un server IBM Process Center include acronimul pistei în contextul de versionare în cazul în care acesta nu este **Main**.

O definiţie de proces operaţional într-o aplicaţie de proces este identificată de obicei prin acronimul numelui procesului operaţional, acronimul instantaneului şi numele definiţiei procesului operaţional. Alegeţi nume unice pentru definiţiile procesului operaţional ori de câte ori este posibil. Atunci când există nume duplicate, este posibil să întâmpinaţi următoarele probleme:

- S-ar putea să fiţi în imposibilitatea de a expune definiţiile procesului operaţional sub formă de servicii web fără o anumită formă de mediere.
- S-ar putea să fiţi în imposibilitatea de a invoca o definiţie a procesului operaţional creat în IBM Process Designer dintr-un proces BPEL creat în IBM Integration Designer.

Contextul de versionare variază în funcţie de cum este implementată aplicaţia de proces.

Convenţii de numire pentru implementarea pe servere Process Center:

Pe serverul IBM Process Center, puteţi implementa un instantaneu al unei aplicaţii de proces, precum şi un instantaneu al unei truse de unelte. În plus, puteţi implementa sugestia unei aplicaţii de proces sau sugestia unei truse de unelte. (O *sugestie* reprezintă versiunea curentă funcţională a aplicaţiei dumneavoastră de proces sau a trusei de unelte.) Contextul de versionare variază în funcţie de tipul implementării.

Pentru aplicaţiile de proces, sugestia aplicaţiei de proces sau instantaneul specific aplicaţiei de proces este utilizat pentru a identifica versiunea în mod unic.

Trusele de unelte pot fi implementate cu una sau mai multe aplicaţii de proces, dar ciclul de viaţă al fiecărei truse de unelte este legat de ciclul de viaţă al aplicaţiei de proces. Fiecare aplicaţie de proces are propria copie a trusei de unelte dependente sau a truselor de unelte implementate pe server. O trusă de unelte implementată nu este partajată între aplicaţiile de proces.

În cazul în care o pistă asociată cu o aplicaţie de proces este numită altfel decât în mod implicit **Main**, acronimul pistei face de asemenea parte din contextul de versiune.

Pentru mai multe informaţii, vedeţi secţiunea “Exemple” la pagina 33, mai târziu în acest subiect.

Instantanee ale aplicaţiei de proces

Pentru implementarea instantaneelor aplicaţiilor de proces, contextul de versiune este o combinaţie între articolele următoare:

- Acronim pentru numele aplicaţiei de proces
- Acronim pentru pista aplicaţiei de proces (în cazul în care se foloseşte altă pistă decât **Main**)
- Acronim pentru instantaneul aplicaţiei de proces

Truse de unelte autonome

Pentru implementarea instantaneelor truselor de unelte, contextul de versionare este o combinaţie între articolele următoare:

- Acronim pentru numele trusei de unelte
- Acronim pentru pista trusei de unelte (în cazul în care se foloseşte altă pistă decât **Main**)
- Acronim pentru instantaneul trusei de unelte

Sugestii

Sugestiile pentru aplicaţia de proces sunt folosite în timpul testării în Process Designer. Acestea pot fi implementate doar pe serverele din Process Center.

Pentru implementările sugestiilor pentru aplicaţia de proces, contextul de versionare este o combinaţie între articolele următoare:

- Acronim pentru numele aplicaţiei de proces
- Acronim pentru pista aplicaţiei de proces (în cazul în care se foloseşte altă pistă decât **Main**)
- "Tip"

Sugestiile pentru trusa de unelte sunt de asemenea folosite în timpul testării iterative în Process Designer. Acestea nu sunt implementate pe un server de producţie.

Pentru implementarea sugestiilor legate de trusa de unelte, contextul de versiune este o combinaţie între articolele următoare:

- Acronim pentru numele trusei de unelte
- Acronim pentru pista trusei de unelte (în cazul în care se foloseşte altă pistă decât **Main**)
- "Tip"

Exemple

Resursele ar trebui să fie numite în mod unic şi identificate extern cu ajutorul contextului de versiune.

- Următorul tabel arată exemple de nume care sunt identificate în mod unic. În acest exemplu, o sugestie a aplicaţiei de proces foloseşte numele de pistă implicit (**Main**):

Tabela 4. Sugestie pentru aplicaţia de proces cu nume implicit pentru pistă

Tip de nume	**Exemplu**
Nume aplicaţie de proces	**Aplicaţie de proces 1**
Acronim pentru numele aplicaţiei de proces	**PA1**
Pistă aplicaţie de proces	**Main**
Acronim pentru urma aplicaţiei de proces	“” (când pista este **Main**)
Instantaneu aplicaţie de proces	
Acronim pentru instantaneul aplicaţiei de proces	**Tip**

Orice modul SCA asociat cu această sugestie legată de aplicaţia de proces include contextul de versiune, aşa cum este descris în tabelul următor:

Tabela 5. Module SCA şi fişiere EAR dependentă de versiune

Nume modul SCA	Nume dependent de versiune	EAR dependent de versiune/nume aplicaţie
M1	PA1-Tip-M1	PA1-Tip-M1.ear
M2	PA1-Tip-M2	PA1-Tip-M2.ear

- Următorul tabel arată un exemplu de sugestie pentru o aplicaţie de proces care foloseşte un nume neimplicit pentru pistă:

Tabela 6. Sugestie pentru aplicaţia de proces cu nume neimplicit pentru pistă

Tip de nume	Exemplu
Nume aplicaţie de proces	**Aplicaţie de proces 1**
Acronim pentru numele aplicaţiei de proces	**PA1**
Pistă aplicaţie de proces	**Track1**
Acronim pentru pista aplicaţiei de proces	**T1**
Instantaneu aplicaţie de proces	
Acronim pentru instantaneul aplicaţiei de proces	**Tip**

Orice modul SCA asociat cu această sugestie legată de aplicaţia de proces include contextul de versiune, aşa cum este descris în tabelul următor:

Tabela 7. Module SCA şi fişiere EAR dependentă de versiune

Nume modul SCA	Nume dependent de versiune	EAR dependent de versiune/nume aplicaţie
M1	PA1-T1-Tip-M1	PA1-T1-Tip-M1.ear
M2	PA1-T1-Tip-M2	PA1-T1-Tip-M2.ear

Convenţii de numire similare se aplică la implementările avansate de instantanee şi sugestii ale trusei de unelte. Ele se aplică şi instantaneelor avansate instalate pe Process Server.

- Următorul tabel arată exemple de nume care sunt identificate în mod unic. În acest exemplu, un instantaneu de aplicaţie de proces foloseşte numele de pistă implicit (**Main**):

Tabela 8. Instantaneu pentru aplicaţia de proces cu nume implicit pentru pistă

Tip de nume	Exemplu
Nume aplicaţie de proces	**Aplicaţie de proces 1**
Acronim pentru numele aplicaţiei de proces	**PA1**
Urmă aplicaţie de proces	**Main**
Acronim pentru urma aplicaţiei de proces	“” (când pista este **Main**)
Instantaneu aplicaţie de proces	**Process Shapshot V1**
Acronim pentru instantaneul aplicaţiei de proces	**PSV1**

Toate modulele SCA asociate cu această aplicaţie de proces includ contextul versiunii, aşa cum se arată în următorul tabel:

Tabela 9. Module SCA şi fişiere EAR dependentă de versiune

Nume modul SCA	Nume dependent de versiune	EAR dependent de versiune/nume aplicaţie
M1	PA1-PSV1-M1	PA1-PSV1-M1.ear

Tabela 9. Module SCA şi fişiere EAR dependentă de versiune (continuare)

Nume modul SCA	Nume dependent de versiune	EAR dependent de versiune/nume aplicaţie
M2	PA1-PSV1-M2	PA1-PSV1-M2.ear

- Următorul tabel arată un exemplu de instantaneu de aplicaţie de proces care foloseşte un nume de pistă neimplicit:

Tabela 10. Instantaneu pentru aplicaţia de proces cu nume neimplicit pentru pistă

Tip de nume	Exemplu
Nume aplicaţie de proces	**Aplicaţie de proces 1**
Acronim pentru numele aplicaţiei de proces	**PA1**
Urmă aplicaţie de proces	**Track1**
Acronim pentru urma aplicaţiei de proces	**T1**
Instantaneu aplicaţie de proces	**Process Snapshot V1**
Acronim pentru instantaneul aplicaţiei de proces	**PSV1**

Toate modulele SCA asociate cu această aplicaţie de proces includ co ntextul versiunii, aşa cum se arată în următorul tabel:

Tabela 11. Module SCA şi fişiere EAR dependentă de versiune

Nume modul SCA	Nume dependent de versiune	EAR dependent de versiune/nume aplicaţie
M1	PA1-T1-PSV1-M1	PA1-T1-PSV1-M1.ear
M2	PA1-T1-PSV1-M2	PA1-T1-PSV1-M2.ear

Convenţii de numire pentru implementarea Process Server:

Pe un Process Server, puteţi implementa instantaneul unei aplicaţii de proces. Acronimul instantaneului aplicaţiei de proces este folosit pentru a identifica versiunea în mod unic.

Pentru implementarea instantaneelor aplicaţiilor de proces, contextul de versionare este o combinaţie între articolele următoare:

- Acronim pentru numele aplicaţiei de proces
- Acronim instantaneu pentru aplicaţia de proces

Resursele ar trebui să fie numite în mod unic şi identificate extern cu ajutorul contextului de versionare. Următorul tabel arată exemple de nume care sunt identificate în mod unic:

Tabela 12. Exemple de nume şi acronime

Tip de nume	Exemplu
Nume aplicaţie de proces	**Aplicaţie de proces 1**
Acronim pentru numele aplicaţiei de proces	**PA1**
Instantaneu aplicaţie de proces	**1.0.0**
Acronim instantaneu pentru aplicaţia de proces	**1.0.0**

Pentru o resursă, precum un modul sau o bibliotecă, versiunea este parte componentă din identitatea sa.

Tabelul de mai jos prezintă un exemplu de două module şi modul în care fişierele EAR asociate includ contextul de versiune:

Tabela 13. Module SCA şi fişiere EAR dependente de versiune

Nume modul SCA	Nume dependent de versiune	EAR dependent de versiune/nume aplicaţie
M1	PA1-1.0.0-M1	PA1-1.0.0-M1.ear
M2	PA1-1.0.0-M2	PA1-1.0.0-M2.ear

Tabelul de mai jos prezintă un exemplu de două biblioteci din domeniul aplicaţiei de proces şi modul în care fişierele JAR asociate includ contextul versiunii:

Tabela 14. Biblioteci din domeniul aplicaţiei de proces şi fişiere JAR dependente de versiune

Nume bibliotecă din domeniul aplicaţiei de proces SCA	Nume dependent de versiune	Nume JAR dependent de versiune
Lib1	PA1-1.0.0-Lib1	PA1-1.0.0-Lib1.jar
Lib2	PA1-1.0.0-Lib2	PA1-1.0.0-Lib2.jar

Legări dependente de versiune

Aplicaţiile de proces pot conţine module SCA care includ legări de import şi export. Atunci când co-implementaţi aplicaţii, legarea pentru fiecare versiune a aplicaţiei trebuie să fie unică. Unele legări sunt actualizate automat în timpul implementării pentru a se asigura unicitatea dintre versiuni. În alte cazuri, va trebui să actualizaţi legarea după implementare pentru a-i asigura unicitatea.

O legare *dependentă de versiune* este pusă în domeniul unei anumite versiuni ale aplicaţiei de proces, lucru care garantează unicitatea între aplicaţiile de proces. Următoarele secţiuni descriu legările care sunt actualizate automat pentru a fi dependente de versiune, precum şi orice acţiune pe care aveţi nevoie să o efectuaţi la momentul rulării atunci când o legare nu este dependentă de versiune. Pentru mai multe informaţii despre lucrurile care se iau în considerare atunci când creaţi module, vedeţi “Considerente atunci când se utilizează legări”.

SCA

Ţinta unei legări SCA este redenumită în mod automat pentru a fi dependentă de versiune în timpul implementării în cazul în care legările de import şi export ale modulului sunt definite în acelaşi domeniu al aplicaţiei de proces.

În cazul în care legările nu sunt definite în acelaşi domeniu al aplicaţiei de proces, se înregistrează un mesaj de informare. Trebuie să modificaţi legarea de import după implementare pentru ca adresa ţintă a punctului final să se modifice. Puteţi utiliza consola administrativă pentru a modifica adresa ţintă a punctului final.

Serviciu Web (JAX-WS sau JAX-RPC)

Adresa ţintă a punctului final a unui serviciu web este redenumită automat pentru a fi dependentă de versiune în timpul implementării în cazul în care următoarele considerente sunt adevărate:

- Aţi urmat convenţia implicită de numire pentru adresa:
  http://*ip*:*port*/*ModuleName*Web/sca/*ExportName*
- Punctul final al adresei este SOAP/HTTP.
- Legările de import şi export ale modulului sunt definite în acelaşi domeniu al aplicaţiei de proces.

Dacă aceste condiţii nu sunt adevărate, este înregistrat un mesaj de informare. Acţiunea pe care o întreprindeţi apoi depinde de modul în care vă implementaţi aplicaţia de proces:

- Dacă vă co-implementaţi aplicaţia de proces, trebuie să redenumiţi manual URL-ul punctului final pentru SOAP/HTTP sau coada de destinaţie SOAP/JMS, astfel încât să fie unic între versiunile aplicaţiei de proces. Puteţi utiliza consola administrativă după implementare pentru a modifica adresa ţintă a punctului final.
- Dacă implementaţi doar o singură versiune a aplicaţiei de proces, aveţi posibilitatea să ignoraţi acest mesaj

Pentru co-implementarea instantaneului legării serviciului Web SOAP/ JMS, acţiunea pe care o întreprindeţi depinde de modul în care vă implementaţi aplicaţia de proces:

- În cazul în care importul şi exportul ţintă sunt în aceeaşi aplicaţie de proces, realizaţi paşii următori înainte de a publica aplicaţia de proces pe Process Center şi de a crea instantaneul:
  1. Modificaţi URL-ul punctului final pentru export. Asiguraţi-vă că destinaţia şi fabrica de conexiuni sunt unice.
  2. Modificaţi URL-ul punctului final pentru import, astfel încât să fie acelaşi cu cel pe care l-aţi pecificat pentru export la pasul anterior.
- În cazul în care importul şi exportul ţintă sunt în aplicaţii de proces diferite, realizaţi paşii următori:
  1. Modificaţi URL-ul punctului final pentru export. Asiguraţi-vă că destinaţia şi fabrica de conexiuni sunt unice.
  2. Publicaţi aplicaţia de proces la Process Center.
  3. Creaţi instantaneul.
  4. Implementaţi aplicaţia de proces pe Process Server.
  5. Utilizaţi consola administrativă WebSphere pentru a modifica URL-ul punctului final pentru importul corespunzător, astfel încât să fie acelaşi cu cel specificat pentru export.

HTTP

Adresa URL a punctului final al unei legări HTTP este redenumită automat în timpul implementării ca să ţină cont de versiune, în cazul în care toate condiţiile următoare sunt adevărate:

- Aţi urmat convenţia implicită de numire pentru adresa:

  http(s)://*ip*:*port*/*ModuleName*Web/*contextPathinExport*
- Legările de import şi export ale modulului sunt definite în acelaşi domeniu al aplicaţiei de proces.

Dacă aceste condiţii nu sunt adevărate, este înregistrat un mesaj de informare. Acţiunea pe care o întreprindeţi apoi depinde de modul în care vă implementaţi aplicaţia de proces:

- Dacă co-implementaţi aplicaţia de proces, trebuie să redenumiţi manual URL-ul punctului final, astfel încât să fie unic între versiunile aplicaţiei de proces. Puteţi utiliza consola administrativă după implementare pentru a modifica adresa ţintă a punctului final.
- Dacă implementaţi doar o singură versiune a aplicaţiei de proces, aveţi posibilitatea să ignoraţi acest mesaj

JMS şi JMS generic

Legările JMS generate de sistem şi legările generice JMS sunt conştiente de versiune în mod automat.

Notă: Pentru legările JMS definite de utilizator şi legările JMS generice, nu are loc redenumirea automată în timpul implementării pentru ca legările să devină conştiente de versiune. În cazul în care legarea este definită de utilizator, trebuie să redenumiţi următoarele atribute, astfel încât să fie unice între versiunile aplicaţiilor de proces:

- Configurarea punctului final
- Coadă destinaţie de recepţie
- Nume port ascultător (dacă este definit)

Setaţi destinaţia Trimitere corespunzătoare în cazul în care modificaţi punctul final al modulului ţintă.

MQ/JMS şi MQ

Nu apare redenumirea automată în timpul implementării pentru a permite legăturilor de tip MQ/JMS sau MQ să fie dependente de versiune.

Trebuie să redenumiţi atributele următoare, astfel încât să fie unice între versiunile aplicaţiilor de proces:

- Configurarea punctului final
- Coadă destinaţie de recepţie

Setaţi destinaţia Trimitere corespunzătoare în cazul în care modificaţi punctul final al modulului ţintă.

EJB

Nu apare redenumirea automată în timpul implementării pentru a permite legăturilor de tip EJB să fie dependente de versiune.

Trebuie să redenumiţi atributul nume JNDI, astfel încât acestea să fie unice între versiunile aplicaţiilor de proces.

Reţineţi că aplicaţiile client trebuiesc de asemenea actualizate pentru a utiliza noile nume JNDI.

EIS

Un adaptor de resurse este redenumit în timpul implementării în mod automat pentru a fi dependent de versiune, atâta timp cât numele implicit al resursei (***ModuleName*App:Descriere Adaptor**) nu a fost modificat.

În cazul în care numele implicit al resursei a fost modificat, numele adaptorului resursei trebuie să fie unic între versiunile aplicaţiei de proces.

Dacă numele adaptorului de resurse nu sunt unice, este înregistrat în timpul implementării un mesaj de informare pentru a vă atenţiona. Aveţi posibilitatea să redenumiţi manual adaptoarele de resurse după implementare folosind consola administrativă.

Invocarea dinamică dependentă de versiune

Aveţi posibilitatea să configuraţi componentele fluxului de mediere, astfel încât să ruteze mesajele către punctele finale care sunt stabilite în mod dinamic în momentul rulării. Atunci când creaţi modulul de mediere, configuraţi căutarea punctelor finale, astfel încât să folosească rutarea dependentă de versiune.

În cazul în care folosiţi stilul IBM_VRM (<*versiune*>.<*ediţie*>.<*modificare*) pentru instantaneu, puteţi exporta fişierul EAR al aplicaţiei de proces în WSRR (WebSphere Service Registry and Repository). Atunci când creaţi modulul de mediere, configuraţi atunci şi căutarea punctelor finale, astfel încât să folosească rutarea dependentă de versiune. De exemplu, selectaţi **Returnare punct final care se potriveşte cu cea mai recentă versiune compatibilă pentru serviciile bazate pe modulul SCA** în câmpul **Politică potrivire**, apoi selectaţi **SCA** pentru **Tip legare**.

Versiunile viitoare ale aplicaţiei de proces sunt implementate pe server şi publicate la WSRR, iar căutarea punctului final al modulului de mediere invocă în mod dinamic cea mai recentă versiune compatibilă a punctului final al serviciului.

Reţineţi ca o alternativă, că puteţi seta ţinta în SMOHeader, iar valoarea poate fi purtată de către mesajul de cerere.

Implementarea aplicaţiilor de proces cu module şi proiecte Java

Aplicaţiile de proces pot conţine module Java EE şi proiecte Java personalizate. Atunci când co-implementaţi aplicaţii, modulul Java EE personalizat pentru fiecare versiune a aplicaţiei trebuie să fie unic.

Reţineţi că modulele Java EE şi proiectele Java personalizate sunt implementate pe un server în cazul în care acestea sunt implementate cu un modul SCA care are o dependenţă declarată în acestea. Dacă nu selectaţi **Implementare cu modul** (care este implicit) atunci când declaraţi dependenţa, trebuie să implementaţi modulul sau proiectul în mod manual.

Implementarea aplicaţiilor de proces cu ajutorul regulilor operaţionale şi a selectoarelor

În cazul în care implementaţi mai multe versiuni ale unei aplicaţii de proces care include o regulă operaţională sau o componentă de tip selector, fiţi atenţi la modul în care versiunile folosesc metadatele asociate.

Metadatele dinamice pentru o regulă operaţională sau pentru o componentă de tip selector sunt definite în momentul rulării prin numele componentei, numele spaţiului pentru componenta ţintă şi tipul componentei. În cazul în care două

sau mai multe versiuni ale aplicaţiei de proces care conţin o regulă operaţională sau un selector sunt implementate în acelaşi mediu de runtime, acestea vor partaja aceeaşi logică pentru reguli (regulă operaţională) sau aceleaşi metadate (selector) pentru rutare.

Pentru a permite fiecărei versiuni a regulii operaţionale sau a componentei de tip selector ce aparţine de aplicaţia de proces să îşi utilizeze propriile metadate dinamice (logică pentru reguli sau rutare), modificaţi codul (refactorizaţi) pentru spaţiul de nume ţintă, astfel încât acesta să fie unic pentru fiecare versiune a aplicaţiei de proces.

Arhitectura de implementare

Arhitectura de implementare IBM Business Process Manager conţine procese software numite servere, unităţi topologice referite ca noduri şi celule şi magazia de configurare utilizată pentru memorarea informaţiilor de configurare.

Celulele

În IBM Business Process Manager, *celulele* sunt grupări logice de unul sau mai multe noduri dintr-o reţea distribuită.

O celulă este un concept de configurare, o cale pentru administratori de a asocia logic nodurile unul cu celălalt. Administratorii definesc nodurile care alcătuiesc o celulă în funcţie de criteriile specifice care au sens în mediile lor organizaţionale.

Datele de configurare administrative sunt memorate în fişiere XML. O celulă reţine fişiere de configurare master pentru fiecare server din fiecare nod din celulă. Fiecare nod şi server are, de asemenea, propriile fişiere locale de configurare. Modificările fişierului local de configurare a unui nod sau a unui server sunt temporare dacă serverul aparţine celulei. Când sunt efective, modificările locale înlocuiesc configuraţiile celulei. Modificările fişierelor de configurare ale serverului master şi ale nodului master făcute la nivelul celulei înlocuiesc orice modificare temporară făcută asupra nodului când documentele configuraţiei celulei sunt sincronizate cu nodurile. Sincronizarea apare la evenimente desemnate, cum ar fi pornirea unui server.

Serverele

Serverele furnizează funcţionalitatea de bază a IBM Business Process Manager. Servere de proces extind sau măresc abilitatea unui server de aplicaţii de a manipula module SCA (Service Component Architecture). Alte servere (manageri de implementare şi agenţi de nod) sunt utilizate pentru gestionarea serverelor Process.

Un Process Server poate fi un *server autonom* sau un *server gestionat*. Un server gestionat poate fi eventual membru unui *cluster*. O colecţie de servere gestionate, cluster-e de servere şi alte middleware-uri se numeşte un *mediu de implementare*. Într-un mediu de implementare, fiecare server sau cluster gestionat este configurat pentru o anumită funcţie în mediul de implementare (de exemplu, gazdă destinaţie, gazdă modul de aplicaţie sau server Common Event Infrastructure). Un server autonom este configurat să furnizeze toate funcţiile cerute.

Serverele asigură mediul runtime pentru modulele SCA, pentru resursele care sunt utilizate de către acele module (surse de date, specificări de activare şi destinaţii JMS) şi pentru resurse livrate de IBM (destinaţii de mesaje, containere Business Process Choreographer şi servere Common Event Infrastructure).

Un *agent de nod* este un agent administrativ ce reprezintă un nod din sistemul dvs. şi gestionează serverele acelui nod. Agenţii de nod monitorizează serverele de pe un sistem de gazdă şi rutează cererile administrative către servere. Agentul de nod este creat când nodul este federalizat unui manager de implementare.

Un *manager de implementare* este un agent administrativ care furnizează o vizualizare centralizată de gestionare pentru mai multe servere şi cluster-e.

Un server autonom este definit de un profil autonom; un manager de implementare este definit de un profil corespunzător; serverele gestionate sunt create într-un *nod gestionat*, ce este definit de un profil personalizat.

Serverele autonome:

Un server autonom asigură un mediu pentru implementarea modulelor SCA într-un proces server. Acest proces server include, dar nu este limitat la o consolă administrativă, o ţintă de implementare, suportul de mesaje, managerul de reguli de proces operaţional şi un server Infrastructură eveniment comun.

Un server autonom este simplu de setat şi are o consolă Primii paşi din care puteţi porni şi opri serverul şi puteţi deschide galeria de eşantioane şi consola administrativă. Dacă instalaţi eşantioanele IBM Business Process Manager şi apoi deschideţi galeria de eşantioane, o soluţie exemplu este implementată în serverul autonom. Puteţi explora resursele utilizate pentru acest exemplu din consola administrativă.

Puteţi implementa propriile dumneavoastră soluţii într-un server autonom, dar un server autonom nu poate furniza capacitatea, scalabilitatea sau robusteţea care este cerută de un mediu de producţie. Pentru mediul dumneavoastră de producţie, este mai bine să utilizaţi un mediu Network Deployment.

Este posibil să porniţi cu un server autonom şi mai târziu să îl includeţi într-un mediu Network Deployment, federalizându-l la o celulă a managerului de implementare, *asigurat că nici un alt nod nu a fost federalizat la acea celulă*. Nu este posibil să federalizaţi mai multe servere autonome într-o celulă. Pentru a federaliza serverul autonom, utilizaţi consola administrativă a managerului de implementare sau comanda**addNode**. Serverul autonom nu trebuie să ruleze când îl federalizaţi utilizând comanda **addNode**.

Un server autonom este definit de către un profil server autonom.

Cluster-ele:

Cluster-ele sunt grupuri de servere care sunt gestionate împreună şi care participă la gestiunea încărcării de lucru.

Un cluster poate conţine noduri sau servere individuale de aplicaţii. Un nod este de obicei un calculator fizic cu o adresă IP a gazdei distinctă care rulează unul sau mai multe servere de aplicaţii. Cluster-ele pot fi grupate sub configuraţia unei celule care asociază logic multe servere şi cluster-e cu configuraţii şi aplicaţii diferite unul cu celălalt în funcţie de discreţia administratorului şi de ceea ce are sens în mediile lor organizaţionale.

Cluster-ele sunt responsabile pentru echilibrarea încărcării de lucru în servere. Serverele care sunt parte componentă a unui cluster sunt numite membri ai cluster-ului. La instalarea unei aplicaţii sau a unui cluster, aplicaţia este instalată automat în fiecare membru al cluster-ului.

Deoarece fiecare membru al cluster-ului conţine aceleaşi aplicaţii, puteţi distribui taskuri client în funcţie de capacităţile diferitelor maşini prin asignarea de ponderi fiecărui server.

Asignarea de ponderi serverelor dintr-un cluster îmbunătăţeşte performanţa şi preluarea la defect. Taskurile sunt asignate serverelor care au capacitatea de a realiza operaţiile taskului. Dacă un server nu este disponibil pentru a realiza taskul, acesta este asignat altui membru al cluster-ului. Această capabilitate de reasignare are avantaje evidente la rularea unui singur server de aplicaţii care poate deveni supraîncărcat dacă sunt făcute prea multe cereri.

Profilurile

Un profil defineşte un mediu runtime unic, cu fişiere de comandă, fişiere de configurare şi fişiere istoric separate. Profilurile definesc trei tipuri diferite de medii în sistemele IBM Business Process Manager: server autonom, manager de implementare şi nod gestionat.

Utilizând profiluri, puteţi avea mai multe medii runtime într-un sistem, fără a trebui să instalaţi copii multiple ale fişierelor binare IBM Business Process Manager.

Utilizaţi Profile Management Tool sau utilitarul **manageprofiles** al liniei de comandă pentru a crea profiluri.

Notă: În platformele distribuite, fiecare profil are un nume unic. În platforma z/OS, toate profilurile sunt numite “implicit”.

Directorul profilului

Fiecare profil din sistem are propriul său director care conţine toate fişierele sale. Specificaţi locaţia directorului de profil la crearea profilului. În mod implicit, este în directorul de profiluri din directorul în care este instalat IBM Business Process Manager. De exemplu, profilul Dmgr01 este în C:\Program Files\IBM\WebSphere\ProcServer\profiles\Dmgr01.

Consola Primii paşi

Fiecare profil din sistem are o consolă Primii paşi. Puteţi utiliza această interfaţă pentru a vă familiariza cu serverul autonom, cu managerul de implementare sau cu nodul gestionat.

Profilul implicit

Primul profil pe care îl puteţi crea într-o instalare a IBM Business Process Manager este *profilul implicit*. Profilul implicit este destinaţia implicită pentru comenzi lansate din directorul bin din directorul în care a fost instalat IBM Business Process Manager. Dacă există doar un profil într-un sistem, fiecare comandă operează în acel profil. Dacă creaţi un alt profil, puteţi să îl faceţi implicit.

Notă: Profilul implicit nu este neapărat un profil al cărui nume este "implicit".

Augmentare profiluri

Dacă aveţi deja un profil manager de implementare, un profil personalizat sau un profil server autonom creat pentru WebSphere Application Server Network Deployment sau WebSphere ESB, îl puteţi *completa* să suporte IBM Business Process Manager în plus faţă de funcţiile existente. Pentru a completa un profil, instalaţi mai întâi IBM Business Process Manager. Apoi utilizaţi Profile Management Tool sau utilitarul **manageprofiles** al liniei de comandă.

Restricţie: Nu puteţi completa un profil dacă acesta defineşte un nod gestionat care este deja federalizat la un manager de implementare.

Managerii de implementare

Un manager de implementare este un server care gestionează operaţiile pentru un grup sau celulă logică a altor servere. Managerul de implementare este locaţia centrală pentru administrarea serverelor şi a cluster-elor.

La crearea unui mediu de implementare, profilul managerului de implementare este primul profil pe care îl creaţi. Managerul de implementare are o consolă Primii paşi, de la care puteţi porni şi opri managerul de implementare şi porniţi consola administrativă. Folosiţi consola administrativă a managerului de implementare pentru a gestiona serverele şi cluster-ele din celulă. Aceasta include configurarea serverelor şi a cluster-elor, adăugarea serverelor la cluster-e, pornirea şi oprirea serverelor şi a cluster-elor şi implementarea modulelor SCA.

Deşi managerul de implementare este un tip de server, nu puteţi implementa module pe însuşi managerul de implementare.

Nodurile

Un *nod* este o grupare logică de servere gestionate.

De obicei, un nod corespunde unui sistem informatic logic sau fizic cu o adresă IP gazdă distinctă. Nodurile nu se pot extinde pe mai multe calculatoare. De obicei, numele nodului sunt identice cu numele gazdă pentru calculator.

Nodurile din topologia Network Deployment pot fi gestionate sau negestionate. Un nod gestionat are un proces al agentului de nod care gestionează configuraţia sa şi serverele. Nodurile negestionate nu au un agent de nod.

Nodurile gestionate:

Un *nod gestionat* este un nod care este federalizat către un manager de implementare şi care conţine agent nod şi poate conţine servere gestionate. Într-un nod gestionat, puteţi configura şi rula serverele gestionate.

Serverele gestionate care sunt configurate pe un nod formează resursele mediului dumneavoastră de implementare. Aceste servere sunt create, configurate, pornite, oprite, gestionate şi şterse folosind consola administrativă a managerului de implementare.

Un nod gestionat are un agent nod care gestionează toate serverele de pe un nod.

Când un nod este federalizat, este creat automat un proces agent nod. Acest agent nod trebuie să ruleze pentru a putea gestiona configuraţia profilului. De exemplu, când realizaţi următoarele taskuri:

- Porniţi şi opriţi procesele serverului.
- Sincronizaţi datele de configurare de pe managerul de implementare cu copia de pe nod.

Totuşi, agentul nod nu trebuie să ruleze pentru ca aplicaţiile să poată rula sau configura resursele din nod.

Un nod gestionat poate conţine unul sau mai multe servere, care sunt gestionate de un manager de implementare. Puteţi implementa soluţii pentru servere într-un nod gestionat, dar nodul gestionat nu conţine o galerie de eşantioane de aplicaţii. Nodul gestionat este definit de un profil personalizat şi are o consolă Primii paşi.

Nodurile negestionate:

Un nod negestionat nu are un agent de nod care să îi gestioneze serverele.

Nodurile negestionate din topologia Network Deployment pot avea definiţii de servere cum ar fi servere web, dar nu definiţii de servere de aplicaţii. Nodurile negestionate nu pot fi niciodată federalizate. Prin urmare, un agent de nod nu poate fi adăugat niciodată la un nod negestionat. Alt tip de nod negestionat este un server autonom. Managerul de implementare nu poate gestiona acest server autonom deoarece nu este cunoscut de celulă. Un server autonom poate fi federalizat. Când este federalizat, un agent de nod este creat în mod automat. Nodul devine un nod gestionat în celulă.

Agenţii de nod

Agenţii de nod sunt agenţi administrativi care rutează cereri administrative la servere.

Un agent de nod este un server care rulează pe fiecare calculator gazdă care participă la configuraţia Network Deployment. Este pur şi simplu un agent administrativ şi nu este implicat în funcţii care deservesc aplicaţii. Un agent de nod găzduieşte, de asemenea, alte funcţii administrative importante precum servicii de transfer de fişiere, sincronizare de configurare şi monitor de performanţă.

Considerente privind numirea pentru profiluri, noduri, server, gazde şi celule

Acest subiect discută termeni rezervaţi şi probleme pe care trebuie să le luaţi în considerare la numirea profilului, nodului, serverului, gazdei şi celulei dumneavoastră (dacă se poate aplica). Acest subiect se aplică platformelor distribuite.

Considerente privind numirea profilului

Numele profilului poate fi orice nume unic cu următoarele restricţii. Nu folosiţi niciunul din următoarele caractere când vă denumiţi profilul:

- Spaţii
- Caractere speciale care nu sunt permise în cadrul unui director al sistemului dumneavoastră de operare, precum *, & sau ?.
- Semne / sau \

Sunt permise caractere pe doi octeţi.

Windows **Considerente privind calea directorului:** Calea către directorul instalării trebuie să fie mai mică decât sau egală cu 60 caractere. Numărul de caractere din directorul *cale_director_profiluri\nume_profil* trebuie să fie mai mic decât sau egal cu 80 caractere.

Notă: Folosiţi o convenţie de numire cale scurtă când creaţi un profil într-un mediu Windows pentru a evita limitarea Windows de lungime de cale de 255 caractere.

Considerente privind numele de nod, server, gazdă şi celulă

Nume rezervate: Evitaţi utilizarea numelor rezervate ca valori ale câmpului. Utilizarea numelor rezervate poate duce la rezultate neprevăzute. Sunt rezervate următoarele cuvinte:

- cells
- nodes
- servers
- clusters
- applications
- deployments

Descrieri de câmpuri pe paginile Nod şi nume de gazde şi nod, Gazdă şi Nume celule: Utilizaţi indicaţiile corespunzătoare pentru a denumi atunci când creaţi profile.

- Profilurile serverului autonom
- Profilurile managerului de implementare
- Profiluri personalizate

Tabela 15. Indicaţii de denumire pentru profiluri de server autonome

Nume câmp	Valoare implicită	Constrângeri	Descriere
Nume nod	Linux UNIX Windows *shortHostName* Node *NodeNumber* unde: • *shortHost Name* este numele scurt al gazdei. • *NodeNumber* este un număr secvenţial care porneşte de la 01.	Evitaţi folosirea numelor rezervate.	Selectaţi orice nume doriţi. Pentru a ajuta organizarea instalaţiei dumneavoastră, folosiţi un nume unic dacă aveţi de gând să creaţi mai mult de un server pe sistem.
Nume server	Linux UNIX Windows server1	Folosiţi un nume unic pentru server.	Numele logic pentru server.
Nume de gazdă	Linux UNIX Windows Forma lungă a numelui serverului de nume domeniu (DNS - domain name server).	Numele gazdă trebuie să fie adresabil prin reţeaua dumneavoastră. Dacă doriţi să folosiţi Business Space, utilizaţi un nume de gazdă complet calificat.	Folosiţi numele DNS actual sau adresa IP a staţiei dumneavoastră de lucru pentru a permite comunicaţia cu ea. Vedeţi informaţii suplimentare despre numele gazdă urmând această tabelă.

Tabela 16. Indicaţii de denumire pentru profiluri de manager de implementare

Nume câmp	Valoare implicită	Constrângeri	Descriere
Nume nod	Linux UNIX Windows *shortHostName* Cell Manager*Node Number* unde: • *shortHost Name* este numele scurt al gazdei. • *NodeNumber* este un număr secvenţial care porneşte de la 01.	Folosiţi un nume unic pentru managerul de implementare. Evitaţi folosirea numelor rezervate.	Numele este folosit pentru administrarea în cadrul celulei manager de implementare.
Nume de gazdă	Linux UNIX Windows Forma lungă a numelui serverului de nume domeniu (DNS - domain name server).	Numele gazdă trebuie să fie adresabil prin reţeaua dumneavoastră. Evitaţi folosirea numelor rezervate. Dacă doriţi să folosiţi Business Space, utilizaţi un nume de gazdă complet calificat.	Folosiţi numele DNS actual sau adresa IP a staţiei dumneavoastră de lucru pentru a permite comunicaţia cu ea. Vedeţi informaţii suplimentare despre numele gazdă urmând această tabelă.
Nume celulă	Linux UNIX Windows *shortHostName* Cell *CellNumber* unde: • *shortHost Name* este numele scurt al gazdei. • *CellNumber* este un număr secvenţial care porneşte de la 01.	Folosiţi un nume unic pentru celula manager de implementare. Un nume de celulă trebuie să fie unic în orice circumstanţe în care rulează produsul pe aceeaşi staţie de lucru fizică sau cluster de staţii de lucru, precum un Sysplex. În plus, un nume de celulă trebuie să fie unic în orice circumstanţe în care conectarea la reţea între entităţi este necesară fie între celule, fie de la un client care trebuie să comunice cu fiecare dintre celule. Numele de celulă trebuie de asemenea să fie unice dacă spaţiile lor de nume vor fi federalizate. Altfel, aţi putea întâlni excepţii precum javax.naming.Name NotFoundException, caz în care trebuie să creaţi celule unic numite.	Toate nodurile federalizate devin membri ai celulei manager de implementare, pe care aţi numit-o în pagina Nume de noduri, gazde şi celule a Profile Management Tool.

Tabela 17. Indicaţii de denumire pentru profiluri personalizate

Nume câmp	Valoare implicită	Constrângeri	Descriere
Nume nod	Linux UNIX Windows *shortHostName* Node *NodeNumber* unde: • *shortHost Name* este numele scurt al gazdei. • *NodeNumber* este un număr secvenţial care porneşte de la 01.	Evitaţi folosirea numelor rezervate. Folosiţi un nume unic în cadrul celulei manager de implementare.	Numele este folosit pentru administrarea în cadrul celulei manager de implementare la care este adăugat profilul personalizat. Folosiţi un nume unic în cadrul celulei manager de implementare.

Tabela 17. Indicaţii de denumire pentru profiluri personalizate (continuare)

Nume câmp	Valoare implicită	Constrângeri	Descriere
Nume de gazdă	Linux UNIX Windows Forma lungă a numelui serverului de nume domeniu (DNS - domain name server).	Numele gazdă trebuie să fie adresabil prin reţeaua dumneavoastră. Dacă doriţi să folosiţi Business Space, utilizaţi un nume de gazdă complet calificat.	Folosiţi numele DNS actual sau adresa IP a staţiei dumneavoastră de lucru pentru a permite comunicaţia cu ea. Vedeţi informaţii suplimentare despre numele gazdă urmând această tabelă.

Considerente privind numele gazdă:

Numele gazdă este numele de reţea pentru staţia de lucru fizică pe care este instalat nodul. Numele gazdă trebuie să rezolve un nod reţea fizică pe server. Când există mai multe plăci de reţea în server, numele gazdă sau adresa IP trebuie să fie rezolvată la una din plăcile de reţea. Nodurile la distanţă utilizează numele gazdă pentru a se conecta şi a comunica cu acest nod.

IBM Business Process Manager este compatibil cu ambele versiuni ale Internet Protocol versiunea 4 (IPv4) şi versiunea 6 (IPv6). Oricând puteţi introduce adresele IP în consola administrativă sau în altă parte, puteţi face aceasta în orice format. Reţineţi că dacă IPv6 este implementat pe sistemul dumneavoastră, trebuie să introduceţi adresa IP în formatul IPv6 şi, invers dacă IPv6 nu vă este încă disponibilă, introduceţi adresele IP în formatul IPv4. Pentru mai multe informaţii despre IPv6, referiţi-vă la descrierea următoare: IPv6.

Următoarele indicaţii vă pot ajuta la determinarea numelui de gazdă corespunzător pentru staţia dumneavoastră de lucru:

- Selectaţi un nume de gazdă la care pot ajunge alte staţii de lucru din reţeaua dumneavoastră.
- Nu folosiţi identificatorul generic, localhost, pentru această valoare.
- Nu încercaţi să instalaţi produse IBM Business Process Manager pe un server cu un nume de gazdă care foloseşte caractere din setul de caractere pe doi octeţi (DBCS - double-byte character set). Caracterele DBCS nu sunt suportate când sunt folosite în numele gazdă.
- Evitaţi folosirea caracterului liniuţă de subliniere (_) în numele de server. Standardele Internet obligă ca numele de domenii să se conformeze cerinţelor numelor gazdă descrise în Internet Official Protocol Standards RFC 952 şi RFC 1123. Numele de domenii trebuie să conţină doar litere (majuscule sau minuscule) şi cifre. Numele de domenii pot conţine de asemenea caractere liniuţă (-), atâta timp cât liniuţele nu se află la sfârşitul numelui. Caracterele liniuţă de subliniere (_) nu sunt suportate în numele gazdă. Dacă aţi instalat IBM Business Process Manager pe un server cu un caracter liniuţă de subliniere în numele său, accesaţi serverul prin adresa sa IP până îl redenumiţi.

Dacă definiţi noduri coexistente pe acelaşi calculator cu adrese IP unice, definiţi fiecare adresă IP într-o tabelă de căutare a unui server de nume domeniu (DNS - domain name server). Fişierele de configurare pentru servere nu furnizează rezolvarea numelui domeniului pentru adresele IP multiple de pe o staţie de lucru cu o singură adresă de reţea.

Valoarea pe care o specificaţi pentru numele gazdă este folosită ca valoare pentru proprietatea hostName din documentele de configurare. Specificaţi valoarea numelui gazdei într-unul din următoarele formate:

- Şirul numelui gazdă DNS complet calificat, precum xmachine.manhattan.ibm.com
- Şirul numelui gazdă DNS scurt implicit, precum xmachine
- Adrese ID numerice, precum 127.1.255.3

Numele gazdă DNS complet calificat prezintă avantajele de a fi neambiguu şi flexibil. Aveţi flexibilitatea de a schimba adresa IP actuală pentru sistemul gazdă, fără a schimba configuraţia serverului. Această valoare pentru numele gazdă este folositoare în special dacă doriţi să schimbaţi frecvent adresa IP când folosiţi Dynamic Host Configuration Protocol (DHCP) pentru a aloca adresele IP. Un dezavantaj al acestui format este acela de a fi dependent de DNS. Dacă DNS nu este disponibil, atunci conectivitatea este compromisă.

Numele scurt al gazdei este de asemenea rezolvabil în mod dinamic. Un format de nume scurt are abilitatea de a fi redefinit pe fişierul gazdelor locale, pentru ca sistemul să poată rula serverul chiar când este deconectat de la reţea. Definiţi numele scurt la 127.0.0.1 (loopback local) în fişierul gazdelor pentru a rula în mod deconectat. Un dezavantaj al formatului de nume scurt este acela de a fi dependent de DNS pentru accesul la distanţă. Dacă DNS nu este disponibil, atunci conectivitatea este compromisă.

O adresă IP numerică prezintă avantajul de a nu necesita rezolvarea numelui prin intermediul DNS. Un nod la distanţă se poate conecta la nodul pe care îl numiţi cu o adresă IP numerică, fără ca DNS să fie disponibil. Un dezavantaj al acestui format este acela că adresa IP numerică este fixată. Trebuie să modificaţi setările proprietăţii hostName în documentele de configurare de fiecare dată când modificaţi adresa IP a staţiei de lucru. Prin urmare, nu folosiţi o adresă IP numerică dacă utilizaţi DHCP sau dacă modificaţi regulat adresele IP. Un alt dezavantaj al acestui format este acela că nu puteţi folosi nodul dacă gazda este deconectată de la reţea.

BPMN 2.0

Definiţiile de proces operaţional IBM Business Process Manager suportă subclasa Common Executable a clasei de conformitate Modelare proces BPMN 2.0 care lucrează cu modele pe care le puteţi rula.

BPMN (Business Process Model and Notation) este standardul de bază pentru procese din IBM Process Designer şi IBM Process Center. Diagramele BPD (Business process definition) sunt bazate pe specificaţia BPMN. Acest subiect introduce unele dintre modalităţile în care BPMN 2.0 este aplicat în IBM Business Process Manager. Pentru informaţii detaliate despre BPMN, vedeţi pagina Specificaţii BPMN la http://www.bpmn.org/.

IBM Business Process Manager suportă următoarele tipuri de taskuri BPMN 2.0:

- None (task abstract în specificaţia BPMN 2.0)
- Task de sistem (task al serviciului în specificaţia BPMN 2.0)
- Task utilizator
- Script
- Task de decizie (task de reguli operaţionale în specificaţia BPMN 2.0)

Evenimentele mesaj intermediar IBM BPM oferă funcţii asemănătoare cu taskul de trimitere BPMN şi cu taskul de primire.

Notare BPMN 2.0

Începând cu V7.5.1, Process Designer pictogramele de taskuri BPMN 2.0 din diagramele BPD sunt colectate într-o paletă simplificată şi afişate în diagrame de proces. Pictogramele afişează dacă activitatea dumneavoastră este un task de sistem, un task utilizator, un task de decizie, script sau proces legat. Activităţile din modelele care au fost create în versiuni anterioare afişează, de asemenea, tipuri de taskuri BPMN 2.0 corespunzătoare şi pictograme de taskuri la vizualizarea lor în versiunea 7.5.1 sau mai târziu.

Activităţi şi taskuri

Există câteva modificări de terminologie de la versiunile anterioare a Process Designer. Un număr din aceste modificări implică tipuri de activităţi care au fost redenumite.

- Activităţile serviciu (automate) sunt acum taskuri sistem.
- Activităţile serviciu (task) într-un culoar non-sistem sunt acum taskuri utilizator.
- Activităţile serviciu (task) într-un culoar sistem sunt acum taskuri de decizie dacă facă referire la un serviciu de decizie.
- Activităţile serviciu (task) într-un culoar sistem sunt acum taskuri sistem dacă fac referire la orice tip de serviciu altul decât serviciu de decizie.
- Activităţile Javascript sunt acum taskuri script.
- Activităţile proces imbricat sunt acum procese legate.

- Activităţile externe de la versiunile anterioare ale Process Designer sunt disponibile ca implementări externe pentru taskurile utilizator sau pentru taskurile sistem.

Gateway-uri

Nu există modificări de notaţie în gateway-urile din versiunile anterioare. Totuşi, există trei modificări de terminologie. Gateway-ul de decizie este acum *gateway exclusiv*, gateway-ul simplu divizat sau unit este acum *gateway paralel*, iar gateway-ul condiţional separat sau unit este acum *gateway inclusiv*.

De asemenea, există un nou tip de gateway, *gateway de eveniment*. Un gateway de eveniment reprezintă un punct de ramificare într-un proces unde căile alternative care urmează gateway-ul se bazează pe evenimente care apar mai degrabă decât pe evaluarea expresiilor folosind datele de proces (precum cu un gateway exclusiv sau inclusiv). Un eveniment specific, de obicei, primirea unui mesaj, determină calea care va fi urmată.

Evenimente care nu întrerup

BPMN 2.0 a adăugat o notare pentru evenimente care nu întrerup. În mod implicit, un eveniment graniţă întrerupe activitatea de care este ataşată. Atunci când evenimentul este declanşat, activitatea se opreşte iar jetonul continuă secvenţa iieşire a evenimentului. Dacă evenimentul este setat ca non-întrerupere atunci când evenimentul este declanşat activitatea ataşată continuă în paralel şi un nou jeton este generat şi este transmis în fluxul secvenţă de ieşire al evenimentului. Graniţa evenimentului se modifică într-o linie întreruptă pentru evenimente care nu întrerup.

Evenimentele intermediare care sunt ataşate activităţilor sunt evenimente intermediare de întrerupere dacă închid activităţile ataşate lor sau evenimente intermediare non-întrerupere dacă nu închid activităţile ataşate lor.

Eveniment de pornire

Specificaţia BPMN permite modelor de proces pentru să omită simbolurile de evenimente de pornire şi oprire. Process Designer necesită ca modelele de proces să folosească evenimente de pornire şi oprire.

Există diverse tipuri de evenimente de pornire disponibile în Process Designer:

procese

- fără
- mesaj
- ad hoc

subprocese

- fără

subprocese eveniment

- eroare
- mesaj
- cronometru

Puteţi modifica tipul unui eveniment de pornire prin editarea proprietăţilor evenimentului. Puteţi avea numeroase evenimente pornire mesaj într-un proces, dar puteţi folosi doar un eveniment fără pornire.

Oprire evenimente

Sunt disponibile patru tipuri de evenimente: *mesaj*, *terminare*, *eroare* şi *none*. Puteţi modifica tipul unui eveniment de oprire.

Când un proces părinte apelează un proces copil şi procesul copil o acţiune de terminare eveniment, procesul copil se opreşte şi procesul părinte continuă apoi cu următorii paşi.

Subprocese

Specificaţia BPMN defineşte două tipuri de subprocese, înglobate şi refolosibile. Puteţi crea ambele tipuri în Process Designer. Subprocesele înglobate sunt numite doar *subprocese* în Process Designer şi sunt noi în versiunea 7.5.1. Subprocesul reutilizabil BPMN este numit *proces legat* în Process Designer.

Un subproces există în cadrul procesului care îl conţine şi este o modalitate de grupare a paşilor procesului pentru a reduce complexitatea diagramei şi confuzia. Subprocesele restrâng mai mulţi paşi într-o singură activitate. Subprocesul poate fi văzut doar de către procesul în care este definit. Un subproces există în domeniul apelantului său şi are acces la toate variabilele din acel mediu. Nu există nici un parametru care să fie transmis în şi în afara subprocesului înglobat.

Separat de subproces şi de procesul legat, Process Designer are un subproces eveniment care este un subproces specializat care este utilizat pentru manipularea evenimentului. Nu este conectat la alte activităţi prin flux de secvenţe, şi apare doar dacă evenimentul de pornire este declanşat.

Procese legate

Un subproces BPMN reutilizabil este numit un *proces legat* în Process Designer. Acesta este un proces creat în afara procesului actual care poate fi apelat de către procesul actual. Acesta este reutilizabil deoarece alte definiţii de proces pot de asemenea apela acest proces. Procesul legat defineşte parametrii de intrare şi de ieşire şi nu are niciun acces la domeniul sau mediul apelantului. Procesul legat este similar cu procesul imbricat disponibil în versiunile anterioare; nu există nici o modificare în comportamentul activităţii. Procesele anterioare imbricate sunt migrate la procese legate. Procesul legat arată ca un subproces cu o limită compactă şi este evidenţiat în fereastra Inspector.

Bucle

BPMN oferă noţiunea de activitate care poate fi repetată. Activitatea poate fi atomică, însemnând că activitatea se repetă, sau poate fi un subproces, încapsulând o serie de paşi care se repetă. Dacă expandaţi activitatea repetată, vedeţi activităţile conţinute care urmează să fie rulate în mod repetat. Condiţia este evaluată întotdeauna la începutul fiecărei iteraţii buclă. Nu există nici o abilitate de a evalua la sfârşitul fiecărei iteraţii buclă.

IBM Business Process Manager are o *buclă cu mai multe instanţe* care este rulată de un număr finit de ori cu activităţile conţinute în cadrul rulării secvenţiale sau în paralel.

Import procese non-BPMN

Puteţi importa modele care au fost create în IBM WebSphere Business Modeler şi le puteţi utiliza în Process Designer. Pentru informaţii despre importul BPMN 2.0, vedeţi elementele Mapare IBM WebSphere Business Modeler în construcţiile IBM Business Process Manager. De asemenea, puteţi importa modele BPMN 2.0 care au fost create în IBM WebSphere Business Compass, Rational Software Architect sau alte medii de modelare.

Definiţii de proces operaţional (BPD-uri)

Pentru a modela un proces în IBM Process Designer, trebuie să creaţi o definiţie a procesului operaţional (BPD). Business Process Definition poate fi bazată pe un model BPMN importat.

Un BPD este un model reutilizabil al unui proces, definind ceea ce este comun tuturor instanţelor runtime ale acelui model de proces. Un BPD trebuie să conţină un eveniment de pornire, un eveniment sfârşit, cel puţin un strat, şi una sau mai multe activităţi. Consultaţi "Convenţiile de numire IBM Process Designer" în legăturile înrudite pentru detalii despre limitarea caracterului care se aplică BPD-urilor.

Business Process Definition (BPD) trebuie să includă un strat pentru fiecare sistem sau grup de utilizatori care participă într-un proces. Un strat poate fi un strat participant sau un strat sistem. Totuşi puteţi crea un BPD care grupează activităţile unui grup şi a unui sistem într-un singur strat dacă acea este preferinţa dumneavoastră. Consultaţi "Crearea unei definiţii a procesului operaţional (BPD)" în legăturile înrudite pentru informaţii despre cum se creează un BPD.

Puteţi desemna orice persoană sau grup specific pentru a fi responsabil pentru activităţi într-un strat participant. Fiecare strat creat este alocat grupului de participanţi Toţi utilizatorii în mod implicit. Puteţi folosi acest grup implicit de participanţi pentru rularea şi testarea BPD în Inspector. Grupul Toţi utilizatorii include toţi utilizatorii care sunt membrii ai grupului de securitate tw_allusers, care este un grup de securitate special care include în mod automat toţi utilizatorii din sistem.

Un strat sistem conţine activităţi manipulate de un sistem IBM Process Center specific. Fiecare activitate necesită o implementare, care defineşte activitatea şi seturile de proprietăţi pentru task. În timpul implementării, un dezvoltator creează un serviciu sau scrie JavaScript necesar pentru a finaliza activităţile din stratul sistem. Consultaţi "Înţelegerea tipurilor de serviciu" în legăturile înrudite pentru informaţii despre servicii.

Pentru fiecare BPD creat, trebuie să declaraţi variabile pentru a captura datele de afacere care sunt transmise de la o activitate la altă activitate în procesul dumneavoastră. Consultaţi "Gestionarea şi maparea variabilelor" în legăturile înrudite pentru a învăţa despre implementarea variabilelor.

Puteţi de asemenea să adăugaţi evenimente unui BPD. Evenimentele IBM BPM pot fi determinate de trecerea unei date scadente, de sosirea unei excepţii sau a unui mesaj. Declanşatorul dorit determină tipul de eveniment pe care îl alegeţi pentru a implementa. Pentru informaţii detaliate despre tipurile de evenimente disponibile şi despre declanşatorii lor, consultaţi "Modelarea evenimentelor".

Legările

La nucleul unei arhitecturi orientată-pe-servicii este conceptul unui *serviciu*, o unitate de funcţionalitate realizată de o interacţiune între dispozitive de calcul. Un *export* defineşte interfaţa externă (sau punctul de acces) a unui modul, astfel încât componentele SCA (Service Component Architecture) din modul să-şi poată oferi serviciile clienţilor externi. Un *import* defineşte o interfaţă către serviciile din afara modulului, astfel încât serviciile să poată fi apelate din modul. Utilizaţi *legări* specifice-protocolului cu exporturi şi importuri pentru a specifica mijloacele de a transporta datele în şi din modul.

Exporturile

Clienţii externi pot invoca componente SCA într-un modul de integrare peste o varietate de protocoale (cum ar ca HTTP, JMS, MQ şi RMI/IIOP) cu date într-o varietate de formate (cum ar fi XML, CSV, COBOL şi JavaBeans). Exporturile sunt componente ce primesc aceste cereri de la surse externe şi apoi invocă componente IBM Business Process Manager folosind modelul de programare SCA.

De exemplu, în următoarea figură, un export primeşte o cerere prin protocolul HTTP de la o aplicaţie client. Datele sunt transformate într-un obiect business, formatul folosit de componenta SCA. Componenta este apoi invocată cu acel obiect de date.

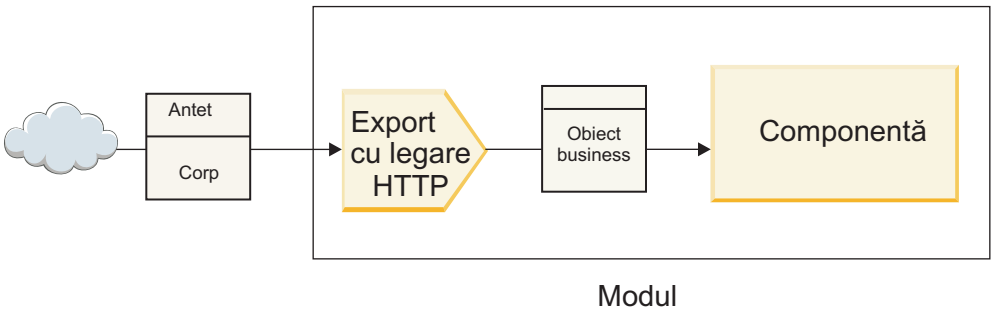


Figura 1. Un export cu legare HTTP

Importuri

O componentă SCA ar putea dori să invoce un serviciu extern non-SCA ce aşteaptă date într-un format diferit. Un import este utilizat de o componentă SCA pentru a invoca serviciul extern folosind modelul de programare SCA. Importul invocă apoi serviciul ţintă în modul aşteptat de serviciu.

De exemplu, în următoarea figură, o cerere de la o componentă SCA este trimisă, de către import, unui serviciu extern. Obiectul business, care este formatul folosit de componenta SCA, este transformat în formatul aşteptat de serviciu şi serviciul este invocat.

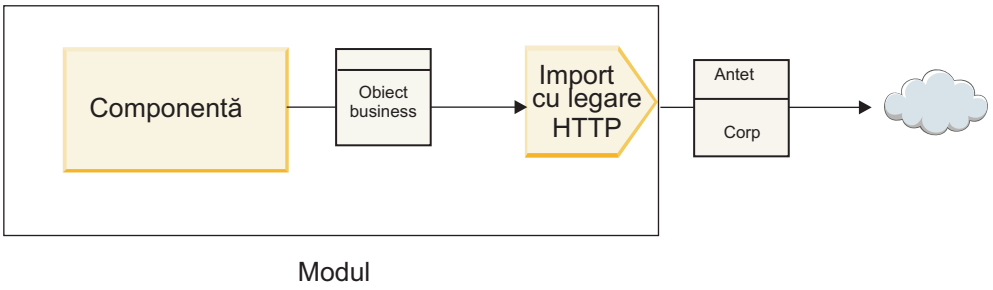


Figura 2. Un import cu legare HTTP

Lista de legări

Utilizaţi Integration Designer pentru a genera o legare pentru un import sau un export şi pentru a configura legarea. Tipurile de legări ce sunt disponibile sunt descrise în următoarea listă.

- SCA

  Legarea SCA, care este cea implicită, dă voie serviciului dumneavoastră să comunice cu servicii din alte module SCA. Utilizaţi un import cu legare SCA pentru a accesa un serviciu dintr-un alt modul SCA. Utilizaţi un export cu legare SCA pentru a oferi un serviciu altor module SCA.
- Serviciu Web

  O legare de serviciu web vă permite să accesaţi un serviciu extern folosind mesajele SOAP interoperabile şi calităţile de serviciu. Puteţi folosi, de asemenea, legăturile serviciului web pentru a include ataşamente ca parte componentă într-un mesaj SOAP.

  Legarea serviciului web poate folosi ca protocol de transport fie SOAP/HTTP (SOAP peste HTTP), fie SOAP/JMS (SOAP peste JMS). Indiferent de transport (HTTP sau JMS) utilizat pentru a pentru a transmite mesajele SOAP, legările serviciului manipulează întotdeauna interacţiunile cerere/răspuns în mod sincron.
- HTTP

  Legarea HTTP vă lasă să accesaţi un serviciu extern folosind protocolul HTTP, unde sunt folosite mesaje non-SOAP sau unde este necesar acces HTTP direct. Legarea este utilizată atunci când lucraţi cu servicii web care se bazează pe modelul HTTP (adică, servicii care folosesc operaţii bine cunoscute din interfaţa HTTP precum GET, PUT, DELETE, şi aşa mai departe).
- Enterprise JavaBeans (EJB)

  Legările EJB permit componentelor SCA să interacţioneze cu servicii furnizate de logica operaţională a Java EE care rulează pe un server Java EE.
- EIS

  Legarea EIS (enterprise information system), când este folosită cu un adaptor de resurse JCA, vă lasă să accesaţi servicii de pe un sistem de informaţii de întreprindere sau să vă faceţi serviciile disponibile EIS-ului.
- Legări JMS

  Legările Java Message Service (JMS), JMS generic şi WebSphere MQ JMS (MQ JMS) sunt folosite pentru interacţiuni cu sisteme de mesagerie, unde comunicarea asincronă prin cozi de mesaje este critică pentru fiabilitate.

Un export cu una din legările JMS urmăreşte coada pentru sosirea unui mesaj şi trimite asincron răspunsul, dacă există, cozii de răspunsuri. Un import cu una din legările JMS construieşte şi trimite un mesaj unei cozi JMS şi urmăreşte o coadă pentru sosirea răspunsului, dacă există.

– JMS

  Legarea JMS vă lasă să accesaţi furnizorul JMS încorporat în WebSphere.

– Generic JMS

  Legarea JMS generic vă lasă să accesaţi un sistem de mesagerie al unui vendor non-IBM.

– MQ JMS

  Legarea MQ JMS vă lasă să accesaţi subsetul JMS al unui sistem de mesagerie WebSphere MQ. Veţi folosi această legare când subsetul JMS de funcţii este suficient pentru aplicaţia dumneavoastră.

- MQ

  Legarea WebSphere MQ vă lasă să comunicaţi cu aplicaţii native MQ, aducându-le în cadrul de lucru al arhitecturii orientate spre servicii şi furnizând acces la informaţii de antet specifice-MQ. Veţi folosi această legare când aveţi nevoie să folosiţi funcţii native MQ.

Privire generală asupra legării de export şi import

Un export vă permite să faceţi serviciile dintr-un modul de integrare disponibile clienţilor externi, iar un import permite componentelor dumneavoastră SCA dintr-un modul de integrare să apeleze servicii externe. Legarea asociată cu exportul sau importul specifică relaţia dintre mesajele de protocol şi obiectele business. De asemenea specifică modul în care sunt selectate operaţiile şi defectele.

Fluxul de informaţii printr-un export

Un export primeşte o cerere, care este intenţionată pentru componenta la care este cablat exportul, peste un anumit transport determinat de legarea asociată (de exemplu, HTTP).

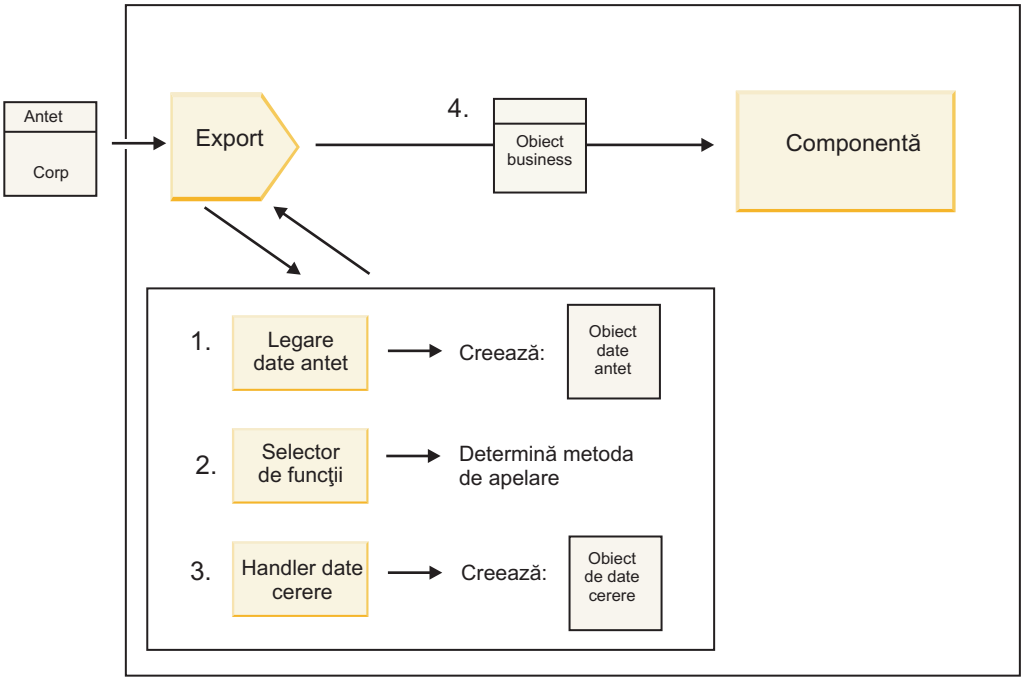


Figura 3. Fluxul unei cereri prin export către o componentă

Când exportul primeşte cererea, are loc următoarea secvenţă de evenimente:

1. Doar pentru legări WebSphere MQ, legarea datelor de antet transformă antetul protocolului într-un obiect de tip date de antet.

2. Selectorul de funcţii determină numele metodei native din mesajul de protocol. Numele metodei native este mapat de către configuraţia exportului de numele unei operaţii de pe interfaţa exportului.
3. Handler-ul de date sau legarea de date de cerere de pe metodă transformă cererea într-un obiect business cerere.
4. Exportul invocă metoda componentei cu obiectul business de cerere.
   - Legarea de export HTTP, legarea de export a serviciului web şi legarea de export EJB invocă componenta SCA în mod sincron.
   - Legările de export JMS, Generic JMS, MQ JMS şi WebSphere MQ invocă componenta SCA asincron.

Notaţi că un export poate propaga anteturile şi proprietăţile de utilizator pe care le primeşte prin protocol, dacă propagarea contextului este activată. Componentele cablate de export pot accesa apoi aceste anteturi şi proprietăţi de utilizator. Vedeţi subiectul “Propagare” din centrul de informare WebSphere Integration Developer pentru informaţii suplimentare.

Dacă aceasta este o operaţie pe două direcţii, componenta returnează un răspuns.

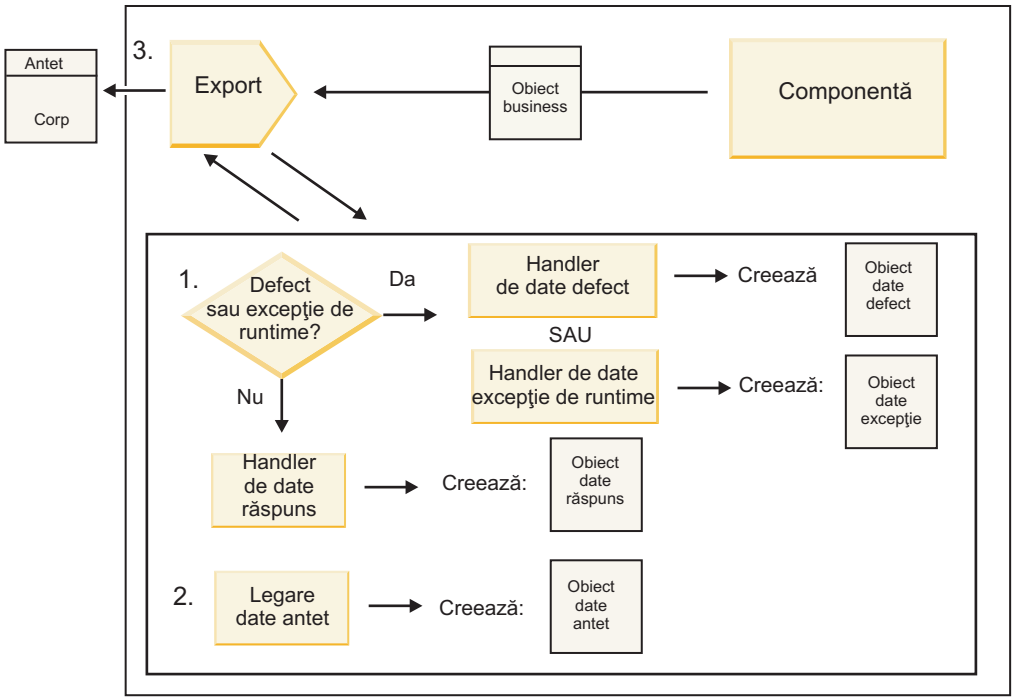


Figura 4. Fluxul unui răspuns înapoi prin export

Are loc următoarea secvenţă de paşi:

1. Dacă este primit un mesaj normal de răspuns de către legarea de export, handler-ul de date sau legarea de date de răspuns de pe metodă transformă obiectul business într-un răspuns.

   Dacă răspunsul este un defect, handler-ul de date sau legarea de date de defecte de pe metodă transformă defectul într-un răspuns defect.

   Doar pentru legări de export HTTP,dacă răspunsul este o excepţie din timpul rulării, handler-ul de date pentru excepţii din timpul rulării, dacă este configurat, este apelat.
2. Doar pentru legări WebSphere MQ, legarea de date de antet transformă obiectele de date de antet în anteturi de protocol.
3. Exportul trimite răspunsul serviciului prin transport.

Fluxul informaţiilor printr-un import

Componentele trimit cereri serviciilor din afara modulului folosind un import. Cererea este trimisă, printr-un anumit transport determinat de legarea asociată.

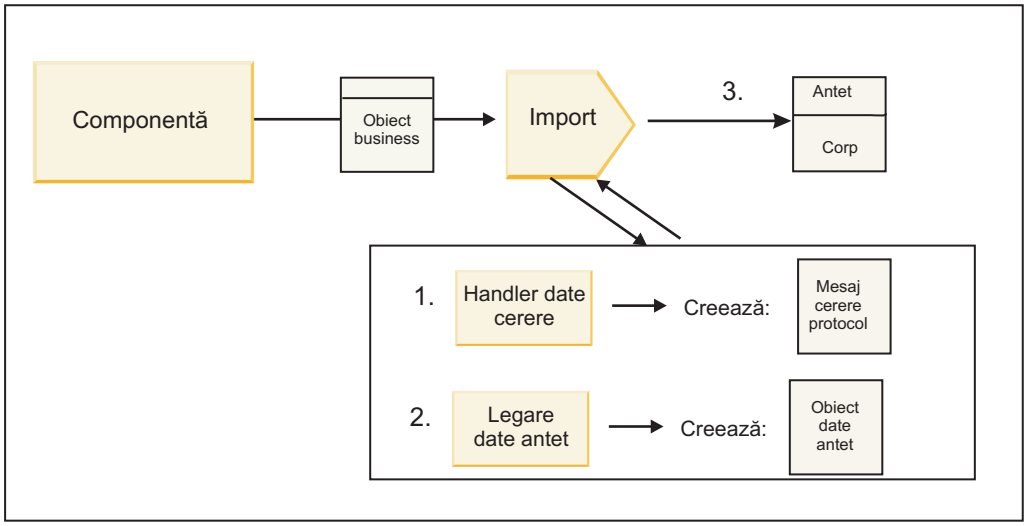


Figura 5. Fluxul de la o componentă prin import către un serviciu

Componenta invocă importul cu un obiect business de cerere.

Notă:

- Legarea de import HTTP, legarea de import a serviciului web şi legarea de import EJB ar trebui să fie invocate în mod sincron de componenta apelantă.
- Legarea de import JMS, Generic JMS, MQ JMS şi WebSphere MQ trebuie invocate asincron.

După ce componenta invocă importul, are loc următoarea secvenţă de evenimente:

1. Handler-ul de date sau legarea de date de cerere de pe metodă transformă obiectul business de cerere într-un mesaj de cerere de protocol.
2. Doar pentru legări WebSphere MQ, legarea de date de antet de pe metodă setează obiectul business de antet în antetul protocolului.
3. Importul invocă serviciul cu cererea de serviciu prin transport.

Dacă aceasta este o operaţie pe două-căi, serviciul returnează un răspuns şi are loc următoarea secvenţă de paşi.

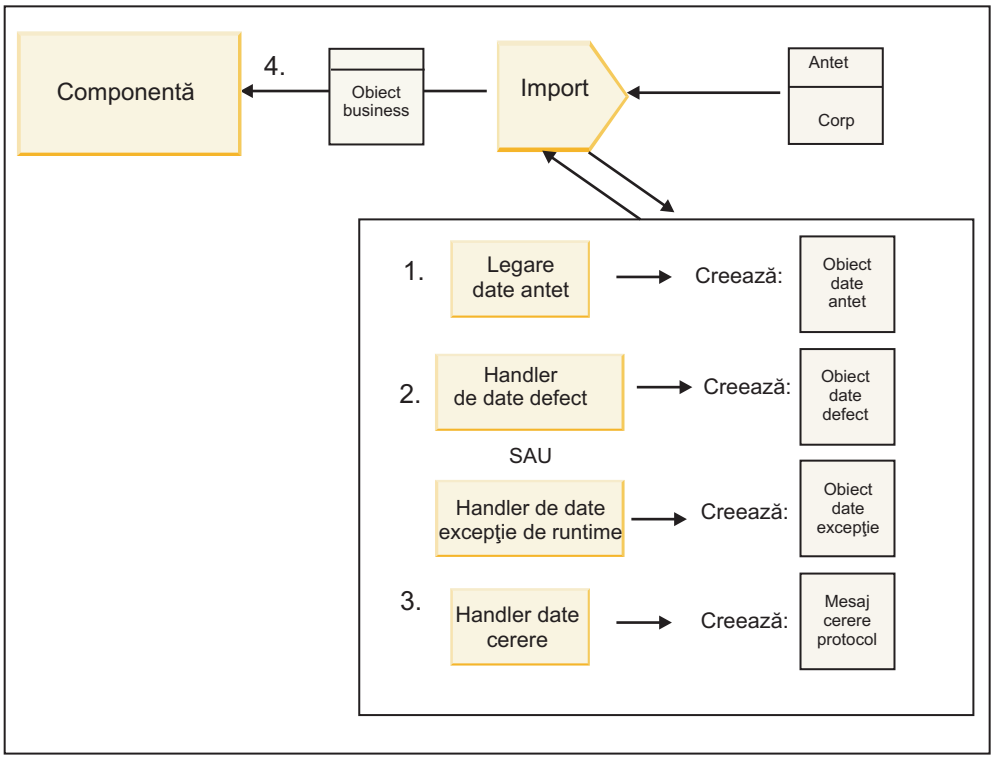


Figura 6. Fluxul unui răspuns înapoi prin import

1. Doar pentru legări WebSphere MQ, legarea datelor de antet transformă antetul protocolului într-un obiect de tip date de antet.
2. Este făcută o determinare dacă răspunsul este un defect sau nu.
   - Dacă răspunsul este un defect, selectorul de defecte inspectează defectul pentru a determina către ce defect WSDL mapează. Handler-ul de date de defect de pe metodă transformă atunci defectul într-un răspuns defect.
   - Dacă răspunsul este o excepţie din timpul rulării, handler-ul de date pentru excepţii din timpul rulării, dacă este configurat, este apelat.
3. Handler-ul de date sau legarea de răspuns de pe metodă transformă răspunsul într-un obiect business de răspuns.
4. Importul returnează obiectul business de răspuns componentei.

Configurarea legărilor de export şi import

Unul din aspectele cheie ale legărilor de export şi import este transformarea formatelor de date, ce indică cum sunt mapate (deserializate) datele de la un format de fire nativ la un obiect business sau cum sunt mapate (serializate) de la un obiect business la un format de fire nativ. Pentru legări asociate cu exporturi, puteţi de asemenea specifica un selector de funcţii care să indice ce operaţie ar trebui efectuată pe date. Pentru legări asociate cu exporturi sau importuri, puteţi indica cum ar trebui tratate defectele ce au loc în timpul procesării.

În plus, specificaţi informaţii specifice transportului pe legări. De exemplu, pentru o legare HTTP, specificaţi URL-ul punctului final. Pentru legarea HTTP, informaţiile specifice transportului sunt descrise în subiectele “Generarea unei legări de import HTTP” şi “Generarea unei legări de export HTTP”. Puteţi de asemenea găsi informaţii despre alte legări în centrul de informare.

Transformarea formatului de date în importuri şi exporturi:

La configurarea unei legări export sau import în IBM Integration Designer, una dintre proprietăţile de configurare pe care le specificaţi este formatul de date utilizat de legare.

- Pentru legări de export, unde o aplicaţie client trimite cereri către şi primeşte răspunsuri de la o componentă SCA, indicaţi formatul datelor native. În funcţie de format, sistemul selectează handler-ul de date sau legarea de date

potrivită pentru a transforma datele native într-un obiect business (care este folosit de componenta SCA) şi invers pentru a transforma obiectul business în date native (care este răspunsul către aplicaţia client).

- Pentru legări de import, unde o componentă SCA trimite cereri către şi primeşte răspunsuri de la un serviciu din afara modulului, indicaţi formatul de date al datelor native. În funcţie de format, sistemul selectează handler-ul de date sau legarea de date potrivită pentru a transforma obiectul business în date native şi viceversa.

IBM Business Process Manager oferă un set de formate de date predefinite şi handler-e de date sau legări de date corespunzătoare ce suportă formatele. Vă puteţi de asemenea crea propriile handler-e de date personalizate şi înregistra formatul de date pentru acele handler-e de date. Pentru informaţii suplimentare, vedeţi subiectul "Dezvoltare handler-e de date" din Centrul de informare IBM Integration Designer.

- *Handler-ele de date* sunt neutre din punct de vedere al protocolului şi transformă datele dintr-un format în altul. În IBM Business Process Manager, handler-ele de date transformă în mod tipic datele native (precum XML, CSV şi COBOL) într-un obiect business şi un obiect business în date native. Deoarece sunt neutre din punct de vedere al protocolului, puteţi reutiliza acelaşi handler de date cu o varietate de legări de export şi import. De exemplu, puteţi folosi acelaşi handler de date XML cu o legare de export sau import HTTP sau cu o legare de export sau import JMS.
- *Legările de date* de asemenea transformă date native într-un obiect business (şi viceversa), dar sunt specifice-protocolului. De exemplu, o legare de date HTTP poate fi folosită doar cu o legare de export sau import HTTP. Spre deosebire de handler-ele de date, o legare de date HTTP nu poate fi refolosită cu o legare de export sau import MQ.

  Notă: Trei legări de date HTTP (HTTPStreamDataBindingSOAP, HTTPStreamDataBindingXML şi HTTPServiceGatewayDataBinding) sunt depreciate începând cu IBM Business Process Manager Versiunea 7.0. Folosiţi handler-e de date oricând este posibil.

După cum s-a menţionat mai devreme, puteţi crea handler-e de date personalizate, dacă este necesar. Puteţi de asemenea crea legări de date personalizate; totuşi, este recomandat să creaţi handler-e de date personalizate deoarece pot fi folosite peste multiple legări.

Handler-e de date:

Handler-ele de date sunt configurate funcţie de legările de export şi import pentru a transforma datele dintr-un format în altul într-un stil neutru protocolului. Mai multe handler-e de date sunt furnizate ca parte din produs, dar vă puteţi de asemenea crea propriul handler de date, dacă este necesar. Puteţi asocia un handler de date cu o legare de export sau import la unul din două niveluri: îl puteţi asocia cu toate operaţiile din interfaţa exportului sau importului sau îl puteţi asocia cu o anumită operaţie pentru cerere sau răspuns.

Handler-e de date predefinite

Utilizaţi IBM Integration Designer pentru a specifica handler-ul de date pe care vreţi să îl utilizaţi.

Handler-ele de date ce sunt predefinite pentru folosul dumneavoastră sunt menţionate în următorul tabel, ce de asemenea descrie cum fiecare handler de date transformă datele de intrare sau de ieşire.

Notă: Cu excepţia notificărilor, aceste handler-e de date pot fi utilizate cu legări JMS, Generic JMS, MQ JMS, WebSphere MQ şi HTTP.
Vedeţi subiectul "Handler-e de date" din centrul de informare Integration Designer pentru informaţii mai detaliate.

Tabela 18. Handler-e de date predefinite

Handler de date	Date native la obiect business	Obiect business la date native
ATOM	Parsează alimentări ATOM într-un obiect business de alimentare ATOM.	Serializează un obiect business de alimentare ATOM în alimentări ATOM.
Delimitat	Parsează date delimitate într-un obiect business.	Serializează un obiect business în date delimitate, inclusiv CSV.

Tabela 18. Handler-e de date predefinite (continuare)

Handler de date	Date native la obiect business	Obiect business la date native
Lăţime fixată	Parsează date cu lăţime fixată într-un obiect business.	Serializează un obiect business în date cu lăţime fixată.
Tratat de WTX	Delegă transformarea formatului datelor către WebSphere Transformation Extender (WTX). Numele mapării WTX este derivat de handler-ul de date.	Delegă transformarea formatului datelor către WebSphere Transformation Extender (WTX). Numele mapării WTX este derivat de handler-ul de date.
Tratat de WTX Invoker	Delegă transformarea formatului datelor către WebSphere Transformation Extender (WTX). Numele mapării WTX este furnizat de utilizator.	Delegă transformarea formatului datelor către WebSphere Transformation Extender (WTX). Numele mapării WTX este furnizat de utilizator.
JAXB	Serializează bean-urile Java într-un obiect business folosind regulile de mapare definite de specificaţia Java Architecture for XML Binding (JAXB).	Deserializează un obiect de business în bean-uri Java folosind regulile de mapare definite de specificaţia JAXB.
JAXWS **Notă:** Handler-ul de date JAXWS poate fi folosit doar cu legarea EJB.	Utilizat de o legare EJB pentru a transforma obiect Java răspuns sau obiect Java excepţie într-un obiect business răspuns folosind regulile de mapare definite de specificaţia Java API for XML Web Services (JAX-WS).	Utilizat de o legare EJB pentru a transforma un obiect business în parametri Java de ieşire folosind regulile de mapare definite de specificaţia JAX-WS.
JSON	Parsează date JSON într-un obiect business.	Serializează un obiect business în date JSON.
Corp nativ	Parsează octeţii, textul, maparea, fluxul sau obiectul nativ în unul din cinci obiecte business de bază (text, octeţi, mapare, flux sau obiect).	Transformă cele cinci obiecte business de bază în octet, text, mapare, flux sau obiect.
SOAP	Parsează mesajul SOAP (şi antetul) într-un obiect business.	Serializează un obiect business într-un mesaj SOAP.
XML	Parsează date XML într-un obiect business.	Serializează un obiect business în date XML.
UTF8XMLDataHandler	Parsează date XML codificate UTF-8 într-un obiect business.	Serializează un obiect business în date XML codificate UTF-8 când se trimite un mesaj.

Crearea unui handler de date

Informaţii detaliate despre crearea unui handler de date pot fi găsite în subiectul “Dezvoltarea handler-elor de date” din centrul de informare Integration Designer.

Legări de date:

Legările de date sunt configurate contra legărilor de export şi import pentru a transforma datele dintr-un format în altul. Legările de date sunt specifice unui protocol. Mai multe legări de date sunt oferite ca parte din produs, dar vă puteţi de asemenea crea propria legare de date, dacă este necesar. Puteţi asocia o legare de date cu o legare de export sau import la unul din două niveluri–o puteţi asocia cu toate operaţiile din interfaţa exportului sau importului sau o puteţi asocia cu o anumită operaţie pentru cerere sau răspuns.

Utilizaţi IBM Integration Designer pentru a specifica legarea de date pe care vreţi să o utilizaţi sau pentru a crea propria dumneavoastră legare de date. O discuţie despre crearea legăturilor de date poate fi găsită în secţiunea “Privire generală asupra legărilor JMS, MQ JMS şi JMS generic” a Centrului de informare IBM Integration Designer.

Legări JMS

Următorul tabel listează legările de date ce pot fi folosite cu:

- Legări JMS
- Legări Generic JMS
- Legări WebSphere MQ JMS

Tabelul include, de asemenea, o descriere a taskurilor realizate de legările de date.

Tabela 19. Legări de date predefinite pentru legări JMS

Legare de date	Date native la obiect business	Obiect business la date native
Obiect Java serializat	Transformă obiectul serializat Java într-un obiect business (care este mapat ca tipul de intrare sau ieşire în WSDL).	Serializează un obiect business în obiectul serializat Java în mesajul obiect JMS.
Octeţi înfăşuraţi	Extrage octeţii din mesajul de octeţi JMS de intrare şi îi înfăşoară în obiectul business JMSBytesBody.	Extrage octeţii din obiectul business JMSBytesBody şi îi înfăşoară în mesajul de octeţi JMS de ieşire
Intrare mapare înfăşurată	Extrage informaţiile despre nume, valoare şi tip pentru fiecare intrare din mesajul de mapare JMS de intrare şi creează o listă cu obiecte business MapEntry. Înfăşoară apoi lista în obiectul business JMSMapBody	Extrage informaţiile despre nume, valoare şi tip din lista MapEntry din obiectul business JMSMapBody şi creează intrările corespunzătoare în mesajul de mapare JMS de ieşire.
Obiect înfăşurat	Extrage obiectul din mesajul obiect JMS de intrare şi îl înfăşoară în obiectul business JMSObjectBody.	Extrage obiectul din obiectul business JMSObjectBody şi îl înfăşoară în mesajul obiect JMS de ieşire.
Text înfăşurat	Extrage textul din mesajul text JMS de intrare şi îl înfăşoară în obiectul business JMSTextBody.	Extrage textul din obiectul business JMSTextBody şi îl înfăşoară în mesajul text JMS de ieşire.

Legări WebSphere MQ

Următorul tabel listează legările de date ce pot fi folosite cu WebSphere MQ şi descrie taskurile realizate de legările de date.

Tabela 20. Legări de date predefinite pentru legări WebSphere MQ

Legare de date	Date native la obiect business	Obiect business la date native
Obiect Java serializat	Transformă obiectul serializat Java din mesajul de intrare într-un obiect business (care este mapat ca tipul de intrare sau ieşire în WSDL).	Transformă un obiect business în obiectul serializat Java din mesajul de ieşire
Octeţi înfăşuraţi	Extrage octeţii din mesajul de octeţi MQ nestructurat şi îi înfăşoară în obiectul business JMSBytesBody.	Extrage octeţii dintr-un obiect business JMSBytesBody şi înfăşoară octeţii în mesajul de octeţi MQ nestructurat de ieşire.
Text înfăşurat	Extrage textul dintr-un mesaj text MQ nestructurat şi îl înfăşoară într-un obiect business JMSTextBody.	Extrage textul dintr-un obiect business JMSTextBody şi îl înfăşoară într-un mesaj text MQ nestructurat.
Intrare flux înfăşurată	Extrage informaţiile despre nume şi tip pentru fiecare intrare din mesajul flux JMS de intrare şi creează o listă cu obiectele business StreamEntry. Înfăşoară apoi lista în obiectul business JMSStreamBody.	Extrage informaţiile despre nume şi tip din lista StreamEntry din obiectul business JMSStreamBody şi creează intrări corespunzătoare în JMSStreamMessage de ieşire.

În plus faţă de legările de date menţionate în Tabela 20 la pagina 57, WebSphere MQ foloseşte de asemenea legări de date de antet. Vedeţi Centrul de informare IBM Integration Designer pentru detalii.

Legări HTTP

Următorul table listează legările de date ce pot fi folosite cu HTTP şi descrie taskurile realizate de legările de date.

Tabela 21. Legări de date predefinite pentru legări HTTP

Legare de date	Date native la obiect business	Obiect business la date native
Octeţi înfăşuraţi	Extrage octeţii din corpul mesajului HTTP de intrare şi îi înfăşoară în obiectul business HTTPBytes.	Extrage octeţii din obiectul business HTTPBytes şi îi adaugă la corpul mesajului HTTP de ieşire.
Text înfăşurat	Extrage textul din corpul mesajului HTTP de intrare şi îl înfăşoară în obiectul business HTTPText.	Extrage textul din obiectul business HTTPText şi îl adaugă la corpul mesajului HTTP de ieşire.

Selectori de funcţii în legări de export:

Un selector de funcţii este folosit pentru a indica ce operaţie ar trebui efectuată pe date pentru un mesaj cerere. Selectorii de funcţii sunt configuraţi ca parte dintr-o legare de export.

Consideraţi un export SCA ce expune o interfaţă. Interfaţa conţine două operaţii–Creare şi Actualizare. Exportul are o legare JMS ce citeşte dintr-o coadă.

Când un mesaj ajunge în coadă, exportului îi sunt transmise datele asociate, dar ce operaţie din interfaţa exportului ar trebui invocată pe componenta cablată? Operaţia este determinată de selectorul de funcţii şi configuraţia legării de export.

Selectorul de funcţii returnează numele funcţiei native (numele funcţiei din sistemul client ce a trimis mesajul). Numele funcţiei native este apoi mapat de numele operaţiei sau funcţiei de pe interfaţa asociată cu exportul. De exemplu, în următoarea figură, selectorul de funcţii returnează numele funcţiei native (CRT) din mesajul de intrare, numele funcţiei native este mapat de operaţia Creare şi obiectul business este trimis componentei SCA cu operaţia Creare.

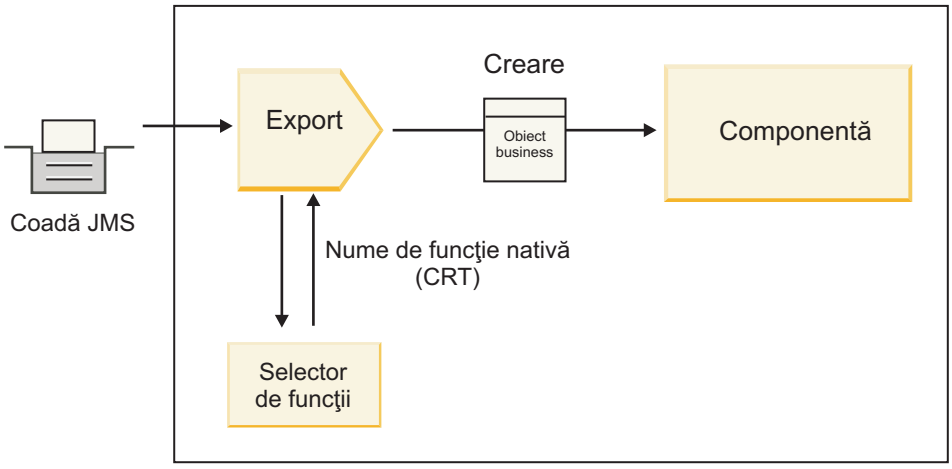


Figura 7. Selectorul de funcţii

Dacă interfaţa are o singură operaţie, nu este nevoie să specificaţi un selector de funcţii.

Mai mulţi selectori de funcţii preambalaţi sunt disponibili şi sunt menţionaţi în secţiunile ce urmează.

Legări JMS

Următorul tabel listează selectorii de funcţii ce pot fi folosiţi cu:

- Legări JMS
- Legări JMS generice
- Legări WebSphere MQ JMS

Tabela 22. Selectori de funcţii predefiniţi pentru legări JMS

Selector de funcţii	Descriere
Selector de funcţii JMS pentru legări simple de date JMS	Foloseşte proprietatea JMSType a mesajului pentru a selecta numele operaţiei.
Selector de funcţii pentru proprietatea antetului JMS	Returnează valoarea Proprietăţii şir JSM, TargetFunctionName, din antet.
Selector funcţie gateway de servicii JMS	Determină dacă cererea este o operaţie cu un singur sens sau cu sens dublu prin examinarea proprietăţii JMSReplyTo setată de client.

Legări WebSphere MQ

Următorul tabel listează selectorii de funcţii ce pot fi utilizaţi cu legări WebSphere MQ.

Tabela 23. Selectorii de funcţii predefiniţi pentru legări WebSphere MQ

Selector de funcţii	Descriere
Selector de funcţii MQ handleMessage	Returnează handleMessage ca o valoare ce este mapată folosind legările metodei de export de numele unei operaţii de pe interfaţă.
MQ foloseşte selector de funcţii implicit JMS	Citeşte operaţia nativă din proprietatea TargetFunctionName din folderul unui antet MQRFH2.
MQ foloseşte formatul corpului mesajului ca funcţie nativă	Găseşte câmpul Format al ultimului antet şi returnează acel câmp ca un Şir.
Selector de funcţii de tip MQ	Creează o metodă în legarea dumneavoastră de export prin extragerea unui URL ce conţine proprietăţile Msd, Set, Type şi Format aflate în antetul MQRFH2.
Selector funcţie gateway de servicii MQ	Foloseşte proprietatea MsgType din antetul MQMD pentru a determina numele proprietăţii.

Legări HTTP

Următorul tabel listează selectorii de funcţii ce pot fi utilizaţi cu legări HTTP.

Tabela 24. Selectori de funcţii predefiniţi pentru legări HTTP

Selector de funcţii	Descriere
Selector de funcţii HTTP bazat pe antetul TargetFunctionName	Utilizează proprietatea antetului HTTP TargetFunctionName de la client pentru a determina ce operaţie să invoce în momentul rulării din export.
Selector de funcţii HTTP bazat pe metoda URL şi HTTP	Foloseşte calea relativă de la URL adăugată la sfârşitul metodei HTTP de la client pentru a determina operaţia nativă definită pe export.
Selector funcţie gateway de servicii HTTP bazat pe URL cu un nume de operaţie	Determină metoda de invocat în funcţie de URL dacă s-a adăugat "operationMode = oneway" la URL-ul cererii.

Notă: Vă puteţi de asemenea crea propriul selector de funcţii, folosind IBM Integration Designer. Informaţii despre crearea unui selector de funcţii sunt furnizate în centrul de informare IBM Integration Designer. De exemplu, o descriere a creării unui selector de funcţii pentru legări WebSphere MQ poate fi găsită în "Privire generală asupra selectorilor de funcţii MQ".

Tratarea defectelor:

Vă puteţi configura legările de import şi export să trateze defecte (de exemplu, excepţii operaţionale) ce au loc în timpul procesării prin specificarea handler-elor de date ale defectelor. Puteţi configura un handler de date al defectelor la trei niveluri–puteţi asocia un handler de date al defectelor cu un defect, cu o operaţie sau pentru toate operaţiile cu o legare.

Un handler de date al defectelor procesează date ale defectelor şi le transformă în formatul corect pentru a fi trimise de legarea de export sau import.

- Pentru o legare de export, handler-ul de date al defectelor transformă obiectul business al excepţiei trimis de la componentă într-un mesaj răspuns ce poate fi folosit de aplicaţia client.
- Pentru o legare de import, handler-ul de date al defectelor transformă datele defectelor sau mesajul răspuns trimis de la un serviciu într-un obiect business al excepţiei ce poate fi folosit de componenta SCA.

Pentru legări de import, legarea apelează selectorul de defecte, ce determină dacă mesajul răspuns este un răspuns normal, un defect operaţional sau o excepţie runtime.

Puteţi specifica un handler de date ale defectelor pentru un anumit defect, pentru o operaţie şi pentru toate operaţiile cu o legare.

- Dacă handler-ul de date ale defectelor este setat la toate cele trei niveluri, handler-ul de date asociat cu un anumit defect este apelat.
- Dacă handler-ele de date ale defectelor sunt setate la nivelurile de operaţie şi legare, este apelat handler-ul de date asociat cu operaţia.

Sunt folosite două editoare în IBM Integration Designer pentru specificarea tratării defectelor. Editorul de interfeţe este folosit pentru a indica dacă va fi un defect pe o operaţie. După ce este generată o legare cu această interfaţă, editorul din vizualizarea proprietăţilor vă dă voie să configuraţi cum va fi tratat defectul. Pentru informaţii suplimentare, vedeţi subiectul "Selectori de defecte" din centrul de informare IBM Integration Designer.

Cum sunt tratate defectele în legările de export:

Când apare un defect în timpul procesării cererii de la o aplicaţie client, legarea de export poate returna informaţiile despre defect clientului. Configuraţi legarea de export să specifice cum ar trebui procesat defectul şi returnat clientului.

Configuraţi legarea de export folosind IBM Integration Designer.

În timpul procesării cererii, un client invocă un export cu o cerere şi exportul invocă componenta SCA. În timpul procesării cererii, componenta SCA poate fie returna un răspuns operaţional fie poate arunca o excepţie operaţională de serviciu sau o excepţie runtime de serviciu. Când apare aceasta, legarea de export transformă excepţia într-un mesaj de defect şi îl trimite clientului, după cum este arătat în figura următoare şi descris în secţiunile ce urmează.

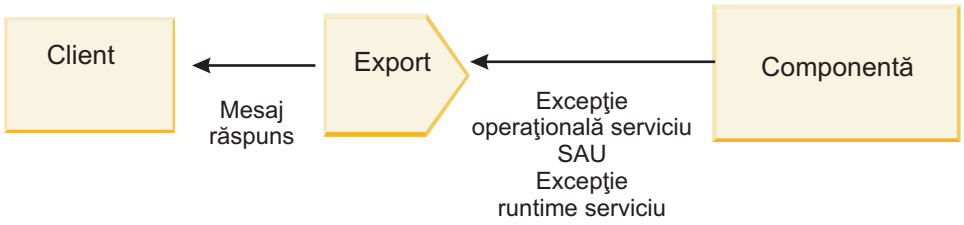


Figura 8. Cum sunt trimise informaţiile despre defect de la componentă prin legarea de export clientului

Puteţi crea un handler de date sau o legare de date personalizată pentru a trata defectele.

Defecte operaţionale

Defectele operaţionale sunt erori sau excepţii operaţionale ce au loc în timpul procesării.

Consideraţi următoarea interfaţă, ce are o operaţie createCustomer pe ea. Această operaţie are două defecte operaţionale definite: CustomerAlreadyExists şi MissingCustomerId.

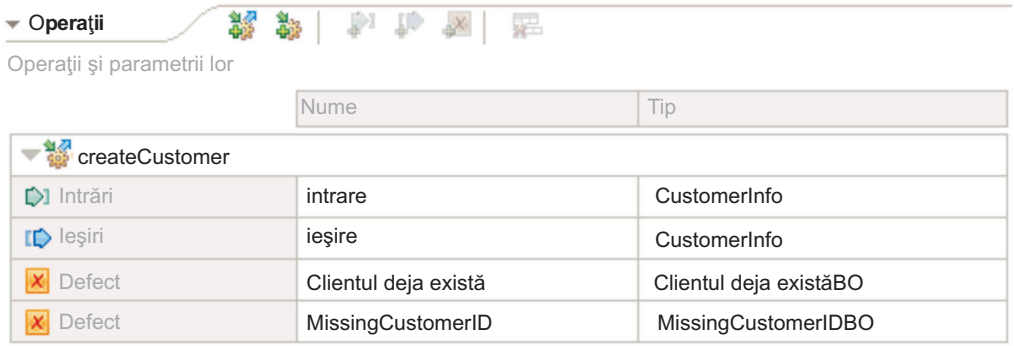


Figura 9. Interfaţă cu două defecte

În acest exemplu, dacă un client trimite o cerere pentru crearea unui client (acestei componente SCA) şi acel client există deja, componenta aruncă un defect CustomerAlreadyExists exportului. Exportul trebuie să propage acest defect operaţional înapoi la clientul apelant. Pentru a face aceasta, utilizează handler-ul de date de defect configurat pe legarea de export.

Când un defect operaţional este primit de legarea de export, are loc următoarea procesare:

1. Legarea determină ce handler de date de defect să invoce pentru tratarea defectului. Dacă excepţia operaţională de servicii conţine numele defectului, este apelat handler-ul de date ce este configurat pe defect. Dacă excepţia operaţională de servicii nu conţine numele defectului, numele defectului este derivat prin potrivirea cu tipurile de defecte.
2. Legarea apelează handler-ul de date de defecte cu obiectul de date din excepţia operaţională de servicii.
3. Handler-ul de date de defecte transformă obiectul de date de defect într-un mesaj răspuns şi îl returnează legării de export.
4. Exportul returnează mesajul răspuns clientului.

Dacă excepţia operaţională de servicii conţine numele defectului, este apelat handler-ul de date ce este configurat pe defect. Dacă excepţia operaţională de servicii nu conţine numele defectului, numele defectului este derivat prin potrivirea cu tipurile de defecte.

Excepţii în timpul rulării

O excepţie runtime este o excepţie ce are loc în aplicaţia SCA în timpul procesării unei cereri ce nu corespunde unui defect operaţional. Spre deosebire de defectele operaţionale, excepţiile runtime nu sunt definite pe interfaţă.

În anumite scenarii, aţi putea dori să propagaţi aceste excepţii runtime către aplicaţia client, astfel încât aplicaţia client să poată lua măsurile corespunzătoare.

De exemplu, dacă un client trimite o cerere (componentei SCA) pentru a crea un client şi are loc o eroare de autorizare în timpul procesării cererii, componenta aruncă o excepţie runtime. Această excepţie runtime trebuie să fie propagată înapoi către clientul apelant, astfel încât să poată lua măsurile necesare referitor la autorizaţie. Aceasta se obţine prin handler-ul de date al excepţiei runtime configurat pe legarea de export.

Notă: Puteţi configura un handler de date al excepţiei runtime doar pe legări HTTP.

Procesarea unei excepţii runtime este similară cu procesarea unui defect operaţional. Dacă a fost configurat un handler de date al excepţiei runtime, are loc următoarea procesare:

1. Legarea de export apelează handler-ul de date corespunzător cu excepţia runtime de servicii.
2. Handler-ul de date transformă obiectul de date al defectului într-un mesaj răspuns şi îl returnează legării de export.
3. Exportul returnează mesajul răspuns clientului.

Tratarea defectelor şi tratarea excepţiilor runtime sunt opţionale. Dacă nu doriţi să propagaţi defecte sau excepţii runtime către clientul apelant, nu configuraţi handler-ul de date al defectelor sau handler-ul de date al excepţiei runtime.

Modul în care defectele sunt tratate în legările de import:

O componentă foloseşte un import pentru a trimite o cerere către un serviciu care se află în afara modulului. Atunci când apare un defect în timpul procesării cererii, serviciul returnează defectul către legarea de import. În momentul în care configuraţi legarea de import puteţi specifica modul în care defectul ar trebui procesat şi returnat către componentă.

Configuraţi legarea de import folosind IBM Integration Designer. Puteţi specifica un handler pentru datele ce conţin defecte (sau o legare de date), şi, de asemenea, puteţi specifica un selector pentru defect.

Handler-e pentru datele ce conţin defecte

Serviciul care procesează cererea trimite, către legarea de import, informaţii despre defect sub forma unei excepţii sau a unui mesaj de răspuns care conţine datele de defect.

Legarea de import transformă excepţia serviciului sau mesajul de răspuns într-un excepţie a serviciului operaţional sau o excepţie de runtime serviciu, după cum se arată în figura de mai jos şi cum este descris în secţiunile care urmează.

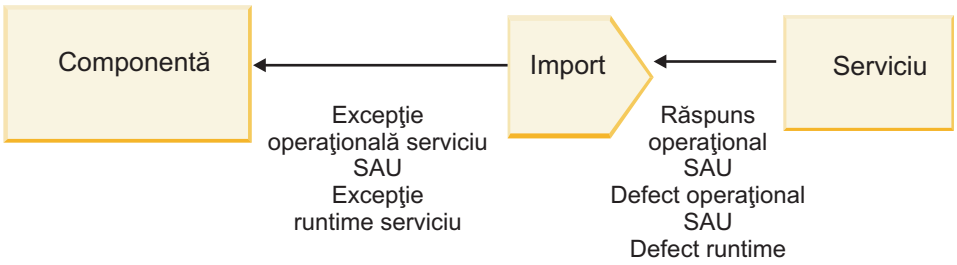


Figura 10. Modul în care informaţiile despre eroare sunt transmise cu ajutorul importului de la serviciu către componentă

Puteţi crea un handler de date personalizat sau o legare de date pentru a trata defectele.

Selectori de defect

Atunci când configuraţi o legare de import, aveţi posibilitatea să specificaţi un selector de defect. Selectorul de defect determină dacă răspunsul de import este un răspuns real, o excepţie operaţională sau un defect apărut în timpul rulării. De asemenea, determină din corpul sau antetul răspunsului, numele nativ al defectului, care este mapat prin configuraţia legării de numele defectului din interfaţa asociată.

Două tipuri de selectoare pentru defecte împachetate sunt disponibile pentru utilizarea cu importurile JMS, MQ JMS, Generic JMS, WebSphere MQ şi HTTP:

Tabela 25. Selectoare pentru defecte împachetate

Tip selector defect	Descriere
Bazat pe antet	Stabileşte dacă un mesaj de răspuns este un defect operaţional, o excepţie apărută în timpul rulării sau un mesaj normal bazat pe anteturile din mesajul de răspuns de intrare.
SOAP	Determină dacă mesajul SOAP de răspuns este un răspuns normal, o defectare a afacerii sau o excepţie în timpul rulării.

În continuare sunt arătate exemple de selectoare pentru anteturi bazate pe defecte şi pentru selectorul de defect SOAP.

- Selector de defect bazat pe antet

  Dacă o aplicaţie doreşte să indice că mesajul primit este un defect operaţional, atunci trebuie să existe două anteturi în mesajul de intrare pentru defectele operaţionale, aşa cum este prezentat în continuare:

  ```
  Header name = FaultType, Header value = Business
  Header name = FaultName, Header value = <user defined native fault name>
  ```

  Dacă o aplicaţie doreşte să indice că mesajul primit este o excepţie apărută în timpul rulării, atunci trebuie să existe un antet în mesajul de intrare, aşa cum este prezentat în continuare:

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

- Selector de defect SOAP

  Un defect operaţional poate fi trimis ca parte componentă a mesajului SOAP care are în compoziţie următorul antet SOAP personalizat. "CustomerAlreadyExists" este numele defectării în acest caz.

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

Selectorul defectării este opţional. Dacă nu specificaţi un selector pentru defectare, legarea de import nu poate determina tipul răspunsului. Prin urmare, legarea îl tratează ca un răspuns operaţional (business) şi apelează handler-ul datelor de răspuns sau legarea de date.

Puteţi crea un selector personalizat pentru defect. Paşii pentru crearea unui selector personalizat pentru defect sunt furnizate în subiectul “Dezvoltarea unui selector personalizat pentru defect” din Centrul de informare IBM Integration Designer.

Defecte operaţionale

O eroare operaţională poate să apară atunci când există o eroare în procesarea unei cereri. De exemplu, dacă trimiteţi o cerere pentru a crea un client, iar acel client există deja, serviciul trimite o excepţie operaţională către legarea de import.

Atunci când legarea primeşte o excepţie operaţională, etapele de prelucrare depind de faptul dacă un selector de defect a fost setat pentru legare.

- Dacă nu a fost setat nici un selector de defect, legarea apelează handler-ul de date pentru răspuns sau legarea de date.
- Dacă a fost setat un selector de defect, are loc următoarea prelucrare:
  1. Legarea de import apelează selectorul de defect pentru a determina dacă răspunsul este un defect operaţional (business) sau un defect apărut în timpul rulării.

2. În cazul în care răspunsul este un defect operaţional, legarea de import apelează selectorul de defect pentru a furniza numele nativ al defectului.
3. Legarea de import determină defectul WSDL corespunzător numelui nativ al defectului returnat de către selectorul de defect.
4. Legarea de import determină handler-ul da date pentru defect care este configurat pentru acest defect WSDL.
5. Legarea de import apelează acest handler-ul da date pentru defect cu datele defectului.
6. Handler-ul da date pentru defect transformă datele defectului într-un obiect de date şi îl returnează către legarea de import.
7. Legarea de import construieşte un obiect de tip excepţie pentru serviciul operaţional cu obiectul de date şi numele defectului.
8. Importul returnează obiectul de tip excepţie pentru serviciul operaţional către componentă.

Excepţii din timpul rulării

O excepţie de runtime poate apărea atunci când există o problemă în comunicarea cu serviciul. Procesarea unei excepţii apărute în timpul rulării este similară cu cea a unei excepţii operaţionale (business). Dacă a fost setat un selector de defect, are loc următoarea prelucrare:

1. Legarea de import apelează handler-ul corespunzător pentru datele excepţiei runtime cu datele excepţiei.
2. Handler-ul da date pentru excepţia runtime transformă datele excepţiei într-un obiect de tip excepţie runtime serviciu şi îl returnează către legarea de import.
3. Importul returnează obiectul excepţie runtime serviciu către componentă.

Interoperabilitatea între modulele SCA şi serviciile Open SCA

IBM WebSphere Application Server V7.0 Feature Pack for Service Component Architecture (SCA) oferă un model de programare simplu, dar foarte puternic pentru construirea aplicaţiilor bazate pe specificaţiile Open SCA. Modulele SCA din IBM Business Process Manager folosesc legările de import şi export pentru a interopera cu serviciile Open SCA dezvoltate într-un mediu Rational Application Developer şi găzduite de WebSphere Application Server Feature Pack for Service Component Architecture.

O aplicaţie SCA invocă o aplicaţie Open SCA prin intermediul unei legări de import. O aplicaţie SCA primeşte un apel de la aplicaţie Open SCA prin intermediul unei legări de export. O listă de legări acceptate este prezentată în “Invocarea serviciilor peste legările interoperabile” la pagina 66.

Invocarea serviciilor Open SCA din module SCA

Aplicaţiile SCA dezvoltate în IBM Integration Designer pot invoca aplicaţiile Open SCA dezvoltate într-un mediu Rational Application Developer. Această secţiune oferă un exemplu de invocare a un serviciu Open SCA dintr-un modul SCA utilizând o legare de import SCA.

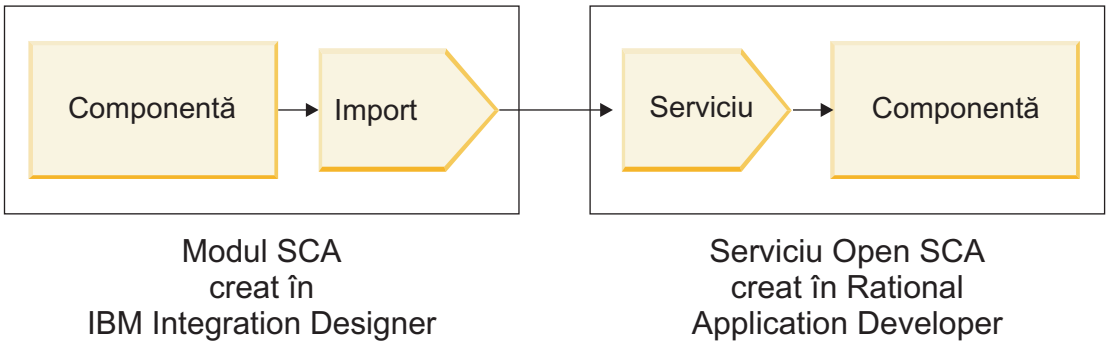


Figura 11. Componenta din modulul SCA invocă un serviciu Open SCA

Nu este nevoie de nici o configuraţie specială pentru a invoca un serviciu Open SCA.

Pentru a vă conecta la un serviciu Open SCA prin intermediul unei legări de import SCA, furnizaţi numele componentei şi numele serviciului pentru serviciul Open SCA în legarea de import.

1. Pentru a obţine numele componentei ţintă şi serviciul din Open SCA compus, efectuaţi următorii paşi:
   a. Asiguraţi-vă că fila **Proprietăţi** este deschisă făcând clic pe **Fereastră > Afişare vizualizare > Proprietăţi**.
   b. Deschideţi editorul pentru compoziţie făcând dublu-clic pe diagrama compusă care conţine componenta şi serviciul. De exemplu, pentru o componentă ce are numele **customer**, diagrama compusă este **customer.composite_diagram**.
   c. Faceţi clic pe componenta ţintă.
   d. În câmpul **Nume** din fila **Proprietăţi**, notaţi numele componentei ţintă.
   e. Faceţi clic pe pictograma serviciului asociată cu componenta.
   f. În câmpul **Nume** din fila **Proprietăţi**, notaţi numele serviciului.
2. Pentru a configura importul IBM Business Process Manager, astfel încât să se conecteze la serviciul Open SCA, efectuaţi paşii următori:
   a. În IBM Integration Designer, mergeţi la fila **Proprietăţi** din importul SCA pe care vreţi să îl conectaţi la serviciul Open SCA.
   b. În câmpul **Nume modul**, introduceţi numele componentei de la pasul 1d.
   c. În câmpul **Nume export**, introduceţi numele serviciului de la pasul 1f.
   d. Salvaţi-vă munca apăsând Ctrl+S.

Invocarea modulelor SCA din serviciile Open SCA

Deschideţi aplicaţiile SCA dezvoltate într-un mediu Rational Application Developer. Acest mediu poate invoca aplicaţiile SCA dezvoltate cu IBM Integration Designer. Această secţiune oferă un exemplu de invocare a unui modul SCA (prin intermediul unei legări de export SCA) dintr-un serviciu Open SCA.

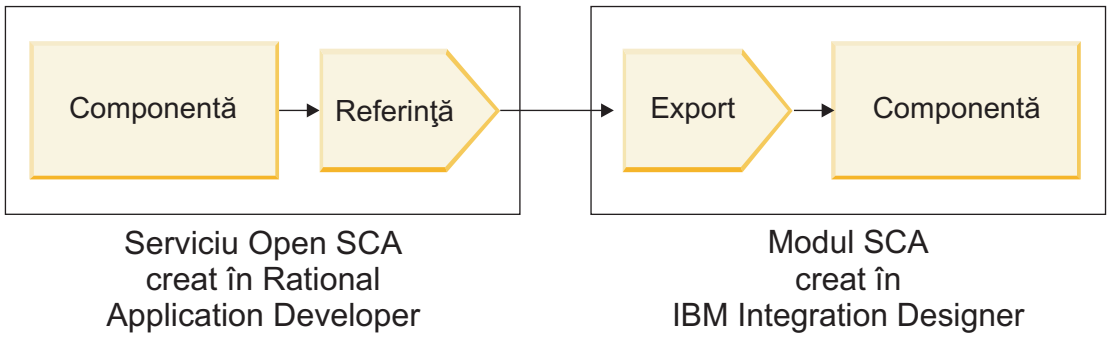


Figura 12. Deschiderea serviciului SCA prin invocarea componentei în modulul SCA

Pentru a vă conecta la o componentă SCA printr-o legare de referinţă a unui Open SCA, furnizaţi numele modulului şi numele de export.

1. Pentru a obţine numele modulului şi exportului ţintă, efectuaţi următorii paşi:
   a. În IBM Integration Designer, deschideţi modulul în editorul de asamblare făcând dublu-clic pe modul.
   b. Faceţi clic pe export.
   c. În câmpul **Nume** din fila **Proprietăţi**, notaţi numele exportului.
2. Configuraţi referinţa Open SCA pe care doriţi să o conectaţi la modulul IBM Business Process Manager şi exportaţi:
   a. În Rational Application Developer, deschideţi editorul pentru compoziţie făcând dublu-clic pe diagrama compusă care conţine componenta şi serviciul.

b. Faceţi clic pe pictograma referinţei componentei pentru a afişa aceste proprietăţi de referinţă în fila **Proprietăţi**.
c. Faceţi clic pe fila **Legare** din partea stânga a paginii.
d. Faceţi clic pe **Legări**, iar apoi faceţi clic pe **Adăugare**.
e. Selectaţi legarea **SCA**.
f. În câmpul **Uri**, introduceţi numele modulului IBM Business Process Manager, urmat de o bară oblică (“/”), urmat de numele de export (pe care îl stabiliţi la pasul 1c la pagina 65).
g. Apăsaţi **OK**.
h. Salvaţi-vă munca apăsând Ctrl+S.

Invocarea serviciilor peste legările interoperabile

Sunt acceptate următoarele legări pentru interoperabilitatea cu un serviciu Open SCA.

- Legare SCA

  În IBM Business Process Manager, atunci când un modul SCA invocă un serviciu Open SCA prin intermediul unei legături de import SCA, sunt acceptate următoarele stiluri de invocare:
  - Asincron (într-un singur sens)
  - Sincron (cerere/răspuns)

  Interfaţa de import SCA şi interfaţa serviciului Open SCA trebuie să utilizeze o interfaţă WSDL conformă WS-I.

  Reţineţi că legarea SCA suportă propagarea contextului de tranzacţie şi de securitate.
- Legare (JAX-WS) a serviciului Web fie cu protocol SOAP1.1/HTTP, fie cu protocol SOAP1.2/HTTP

  Interfaţa de import SCA şi interfaţa serviciului Open SCA trebuie să utilizeze o interfaţă WSDL conformă WS-I.

  În plus, sunt suportate următoarele calităţi ale serviciului:
  - Tranzacţia atomică a serviciilor Web
  - Securitatea serviciilor Web
- Legare EJB

  Pentru a defini interacţiunea dintre un modul SCA şi un serviciu Open SCA se foloseşte o interfaţă Java atunci când este utilizată legarea EJB.

  Reţineţi că legarea EJB suportă propagarea contextului de tranzacţie şi de securitate.
- Legări JMS

  Interfaţa de import SCA şi interfaţa serviciului Open SCA trebuie să utilizeze o interfaţă WSDL conformă WS-I.

  Sunt suportaţi următorii furnizori JMS:
  - Platforma de mesagerie WebSphere (Legarea JMS)
  - WebSphere MQ (Legarea MQ JMS)

Notă: Graficele operaţionale nu sunt interoperabile pe orice legări SCA şi, prin urmare, nu sunt acceptate în interfeţele utilizate pentru a interopera cu WebSphere Application Server Feature Pack for Service Component Architecture.

Tipuri de legări

Folosiţi *legări* specifice-protocolului cu importuri şi exporturi pentru a specifica mijloacele de transportare a datelor în sau dintr-un modul.

Selectarea legărilor corespunzătoare:

Atunci când creaţi o aplicaţie, trebuie să ştiţi cum să selectaţi legarea care corespunde cel mai bine necesităţilor aplicaţiei dumneavoastră.

Legările disponibile din IBM Integration Designer furnizează o gamă de alegeri. În această listă puteţi determina care tip de legătură este mai potrivită nevoilor aplicaţiei dumneavoastră.

Luaţi în considerare o legătură *Service Component Architecture (SCA)* atunci când se pot aplica aceşti factori:

- Toate serviciile sunt conţinute în module; ceea ce înseamnă că nu există servicii externe.
- Vreţi să separaţi funcţia în module SCA diferite care interacţionează direct unul cu celălalt.
- Modulele sunt cuplate strâns.

Luaţi în considerare o legare *serviciu web* atunci când se aplică aceşti factori:
- Trebuie să accesaţi un serviciu extern de pe Internet sau să furnizaţi un serviciu pe Internet.
- Serviciile sunt cuplate slab.
- Comunicaţia sincronă este preferabilă; adică o cerere de la un serviciu poate aştepta pentru un răspuns de la altul.
- Protocolul serviciului extern pe care îl accesaţi sau al serviciului pe care vreţi să îl furnizaţi este SOAP/HTTP sau SOAP/JMS.

Consideraţi o legare *HTTP* când aceşti factori sunt aplicabili:
- Este nevoie să accesaţi un serviciu extern de pe Internet sau să furnizaţi un serviciu de pe Internet şi lucraţi cu alte servicii web cum ar fi GET, PUT şi DELETE.
- Serviciile sunt cuplate slab.
- Comunicaţia sincronă este preferabilă; adică o cerere de la un serviciu poate aştepta pentru un răspuns de la altul.

Luaţi în considerare o legătură *Enterprise JavaBeans (EJB)* atunci când se pot aplica aceşti factori:
- Legarea este pentru un serviciu importat care este el însuşi un EJB sau care trebuie să fie accesat de clienţi EJB.
- Serviciul importat este cuplat slab.
- Interacţiunile EJB stateful nu sunt necesare.
- Comunicaţia sincronă este preferabilă; adică o cerere de la un serviciu poate aştepta pentru un răspuns de la altul.

Luaţi în considerare o legătură *Enterprise Information Szstems(EIS)* atunci când se pot aplica aceşti factori:
- Trebuie să accesaţi un serviciu de pe un sistem EIS utilizând un adaptor de resurse.
- Transmisia de date sincronă este preferată în locul celei asincrone.

Luaţi în considerare o legătură *Java Message Service (JMS)* atunci când se pot aplica aceşti factori:

Important: Există mai multe tipuri de legări JMS. Dacă vă aşteptaţi să schimbaţi mesajele SOAP folosind JMS, luaţi în considerare legarea de servicii web cu protocolul SOAP/JMS. Vedeţi “Legări de serviciu Web” la pagina 68.
- Trebuie să accesaţi un sistem de mesagerie.
- Serviciile sunt cuplate slab.
- Transmisia de date asincron este preferată în locul celei sincron.

Luaţi în considerare o legătură *Generic Java Message Service (JMS)* atunci când se pot aplica aceşti factori:
- Trebuie să accesaţi sistem de mesagerie a unui furnizor non-IBM.
- Serviciile sunt cuplate slab.
- Fiabilitatea este mai importantă decât performanţa; adică, transmisia de date asincronă este preferată în locul celei sincrone.

Luaţi în considerare o legătură *Message Queue (MQ)* atunci când se pot aplica aceşti factori:
- Trebuie să accesaţi un sistem de mesagerie WebSphere MQ şi să utilizaţi funcţiile native MQ.
- Serviciile sunt cuplate slab.
- Fiabilitatea este mai importantă decât performanţa; adică, transmisia de date asincronă este preferată în locul celei sincrone.

Luaţi în considerare o legare *MQ JMS* când se aplică aceşti factori:
- Trebuie să accesaţi un sistem de mesagerie WebSphere MQ dar puteţi face astfel într-un context JMS; adică, subsetul JMS al funcţiilor este suficient pentru aplicaţia dumneavoastră.

- Serviciile sunt cuplate slab.
- Fiabilitatea este mai importantă decât performanţa; adică, transmisia de date asincronă este preferată în locul celei sincrone.

Legări SCA:

O legarea SCA (Service Component Architecture) permite unui serviciu să comunice cu alte servicii din alte module. Un import cu o legare SCA vă permite să accesaţi un serviciu dintr-un alt modul SCA. Un export cu o legare SCA vă permite să oferiţi un serviciu către alte module.

Utilizaţi IBM Integration Designer pentru a genera şi configura legături SCA în importuri şi exporturi din module SCA.

În cazul în care modulele rulează pe acelaşi server sau sunt implementate în acelaşi cluster, o legare SCA este cel mai uşor şi cel mai rapid tip de legare ce poate fi utilizat.

După ce modulul care conţine legarea SCA este implementat pe server, puteţi utiliza consola administrativă pentru a vizualiza informaţii despre legare, sau în cazul unei legări de import, pentru a modifica proprietăţile selectate ale legării.

Legări de serviciu Web:

Legarea unui serviciu web este mijlocul de transmitere a mesajelor de la o componentă SCA (Service Component Architecture) către un serviciu web (şi vice versa).

Privire generală asupra legărilor de serviciu Web:

O legare de import a serviciului web vă permite să apelaţi un serviciu web extern din componentele dvs SCA (Service Component Architecture). O legare de export a serviciului web vă permite să expuneţi componentele SCA clienţilor sub formă de servicii web.

Cu o legare de serviciu web, puteţi accesa serviciile externe folosind mesaje SOAP interoperabile şi calităţile de serviciu (QoS).

Utilizaţi Integration Designer pentru a genera şi configura legările serviciului web la importuri şi exporturi în modeule SCA. Sunt disponibile următoarele tipuri de legături de servicii web:

- SOAP1.2/HTTP şi SOAP1.1/HTTP

  Aceste b se bazează pe API-ul Java API pentru serviciile web XML (JAX-WS), un API de programare Java pentru crearea serviciilor web.

  – Utilizaţi SOAP1.2/HTTP dacă serviciul dvs. web corespunde specificaţiei SOAP 1.2.
  – Utilizaţi SOAP1.1/HTTP dacă serviciul dvs. web corespunde specificaţiei SOAP 1.1.

  Important: Când implementaţi o aplicaţie cu o legare de serviciu web (JAX-WS), serverul ţintă nu trebuie să aibă selectată opţiunea **Pornire componente după cum este necesar**. Pentru detalii, vedeţi “Verificarea configuraţiei serverului” la pagina 76.

  Când selectaţi una din aceste legări, puteţi trimite ataşamente cu mesajele dvs. SOAP.

  Legările serviciului web funcţionează cu mesaje SOAP standard. Folosind una dintre legările serviciului web JAX-WS, totuşi, aveţi posibilitatea să personalizaţi modul în care sunt parsate sau scrise mesajele SOAP. De exemplu, puteţi manipula elementele care nu sunt standard în mesajele SOAP sau puteţi aplica prelucrări suplimentare mesajelor SOAP. Când configuraţi legarea specificaţi un handler personalizat pentru date, care efectuează această prelucrare într-un mesaj SOAP.

  Puteţi folosi seturi de politici cu o legare a serviciului web (JAX-WS). Un set de politici este o colecţie de tipuri de politici, fiecare dintre ele oferind o calitate a serviciilor (QoS). De exemplu, setul de politici WSAddressing oferă o cale de transport neutră pentru adresarea uniformă a serviciilor web şi a mesajelor. Utilizaţi Integration Designer pentru a selecta setul de politici pentru legare.

Notă: Dacă doriţi să utilizaţi un set de politici de tip SAML (Security Assertion Markup Language), trebuie să realizaţi câteva configuraţii suplimentare, aşa cum este descris în “Importul seturilor de politici” la pagina 74.

- SOAP1.1/HTTP

  Utilizaţi această legare dacă doriţi să creaţi servicii web care folosesc un mesaj codat SOAP care se bazează pe JAX-RPC (Java API for XML-based RPC).

- SOAP1.1/JMS

  Utilizaţi această legare pentru a trimite sau pentru a primi mesaje SOAP folosind o destinaţie JMS (Java Message Service).

Indiferent de transportul (HTTP sau JMS) utilizat pentru transmiterea mesajelor SOAP, legările serviciului manipulează întotdeauna interacţiunile cerere/răspuns în mod sincron. Firul de execuţie care face invocarea către furnizorul de servicii este blocat până când este primit un răspuns de la furnizor. Vedeţi “Invocarea sincronă” pentru informaţii suplimentare despre acest stil de invocare.

Important: Următoarele combinaţii de legături de servicii web nu pot fi utilizate pentru exporturile către acelaşi modul. Dacă aveţi nevoie să expuneţi componente folosind mai mult de una dintre aceste legări de export, este nevoie ca fiecare dintre ele să fie într-un modul separat, iar apoi să conectaţi acele module la componentele dumneavoastră folosind legarea SCA:

- SOAP 1.1/JMS şi SOAP 1.1/HTTP folosind JAX-RPC
- SOAP 1.1/HTTP folosind JAX-RPC şi SOAP 1.1/HTTP folosind JAX-WS
- SOAP 1.1/HTTP folosind JAX-RPC şi SOAP 1.2/HTTP folosind JAX-WS

După ce modulul SCA care conţine legarea serviciului web este implementată pe server, aveţi posibilitatea să utilizaţi consola administrativă pentru a vizualiza informaţii legate de legare sau să modificaţi proprietăţile selectate a legăturii.

Notă: Serviciile Web permit aplicaţiilor să interopereze prin utilizarea descrierilor standard ale servicii şi formatelor standard pentru mesajele pe care le schimbă între ei. De exemplu, legările de import şi export ale serviciului web pot interopera cu servicii care sunt implementate folosind WSE (Web Services Enhancements) Versiunea 3.5 şi WCF (Windows Communication Foundation) Versiunea 3.5 pentru Microsoft .NET. Atunci când interacţionaţi cu astfel de servicii, trebuie să vă asiguraţi că:

- Fişierul WSDL care este folosit pentru accesarea unui export de serviciu web include o valoare validă pentru acţiune pentru fiecare operaţie din interfaţă.
- Clientul serviciului web setează fie antetul SOAPAction, fie antetul wsa:Action atunci când trimite un mesaj către exportul serviciului web.

Propagarea antetului SOAP:

În momentul în care manipulaţi mesaje SOAP, este posibil să aveţi nevoie să accesaţi informaţii din anumite anteturi SOAP din mesaje ce sunt primite, ca să vă asiguraţi că mesajele cu anteturi SOAP sunt trimise cu anumite valori sau ca să permiteţi anteturilor SOAP să parcurgă un modul.

Când configuraţi legarea unui serviciu web în Integration Designer, puteţi indica faptul că doriţi ca anteturile SOAP să fie propagate.

- Atunci când cererile sunt primite într-un export sau răspunsurile sunt primite la un import, informaţiile din antetul SOAP pot fi accesate, permiţând ca logica din modul să se bazeze pe valorile antetului şi permiţând acestor anteturi să poată fi modificate.
- Atunci când cererile sunt trimise dintr-un export sau răspunsurile sunt trimise dintr-un import, anteturile SOAP pot fi incluse în aceste mesaje.

Forma şi prezenţa anteturilor SOAP transmise pot fi afectate de seturile de politici configurate la import sau export, aşa cum este explicat în Tabela 26 la pagina 71.

Pentru a configura transmiterea anteturilor SOAP pentru un import sau export, selectaţi (din modul de vizualizare Proprietăţi al Integration Designer) fila **Antet Protocol de Transmitere** şi selectaţi opţiunile de care aveţi nevoie.

Anteturi de Adresare WS

Antetul WS-Addressing poate fi propagat de legarea serviciului web (JAX-WS).

Atunci când transmiteţi antetul de adresare WS, fiţi atenţi la următoarele informaţii:

- Dacă activaţi transmiterea pentru antetul de adresare WS, antetul va fi transmis prin modul în următoarele circumstanţe:
  - Atunci când cererile sunt primite la un export
  - Atunci când răspunsurile sunt primite la un import
- Antetul WS-Addressing nu este transmis în mesajele de ieşire de la IBM Business Process Manager (ceea ce înseamnă că, antetul nu este transmis atunci când cererile sunt trimise de la un import sau atunci când răspunsurile sunt trimise de la un export).

Antetul de securitate WS

Antetul WS-Security poate fi propagat atât prin legarea serviciului web (JAX-WS), cât şi prin legarea (JAX-RPC) a serviciului web.

Specificaţiile SecuritateWS ale serviciilor web descriu îmbunătăţirile aduse mesajelor SOAP pentru a oferi calitatea protecţiei prin integritatea mesajelor, confidenţialitatea mesajelor şi şi autentificarea unică a mesajului. Aceste mecanisme pot fi folosite pentru a acomoda o mare varietate de modele de securitate şi tehnologii de criptare.

Atunci când transmiteţi antetul WS-Security, fiţi atenţi la următoarele informaţii:

- Dacă activaţi transmiterea pentru antetul de securitate WS, antetul va fi transmis prin modul în următoarele circumstanţe:
  - Atunci când cererile sunt primite la un export
  - Atunci când cererile sunt trimise de la un import
  - Atunci când răspunsurile sunt primite la un import
- Implicit, antetul *nu* va fi propagat atunci când răspunsurile sunt trimise de la export. Totuşi, dacă setaţi proprietatea JVM **WSSECURITY.ECHO.ENABLED** cu **true**, antetul va fi transmis atunci când răspunsurile sunt trimise de la export. În acest caz, dacă antetul WS-Security din calea cererii nu este modificat, anteturile WS-Security ar putea fi trimise înapoi sub formă de răspunsuri.
- Forma exactă a mesajului SOAP trimis de la un import pentru o cerere sau dintr-un export pentru un răspuns ar putea să nu se potrivească exact cu mesajul SOAP primit iniţial. Din acest motiv, se presupune că orice semnătură digitală devine nevalidă. În cazul în care este necesară o semnătură digitală în mesajele trimise, aceasta trebuie să fie stabilită prin utilizarea setului de politici de utilizare corespunzător, iar anteturile WS-Security înrudite cu semnătura digitală din mesajele primite ar trebui să fie eliminate din cadrul modulului.

Pentru a propaga antetul WS-Security, trebuie să includeţi schema WS-Security în modulul aplicaţiei. Vedeţi “Includerea schemei WS-Security într-un modul de aplicaţie” la pagina 71 pentru modul de includere al schemei în procedură.

Modul în care anteturile sunt propagate

Modul în care sunt propagate anteturile depinde de politica de securitate setată în legarea de import sau export, aşa cum este arătat în Tabela 26 la pagina 71:

Tabela 26. Modul în care anteturile de securitate sunt transmise

	Legare de import fără politică de securitate	**Legare de import cu politică de securitate**
Legare de import fără politică de securitate	Anteturile de securitate sunt transmise aşa cum sunt prin modul. Ele nu sunt decriptate. Anteturile sunt trimise către ieşire în aceeaşi formă în care au fost primite. Semnătura digitală ar putea deveni nevalidă.	Anteturile de securitate sunt decriptate şi transmise prin modul cu verificarea semnăturii şi autentificare. Anteturile decriptate sunt trimise către ieşire. Semnătura digitală ar putea deveni nevalidă.
Legare de import cu politică de securitate	Anteturile de securitate sunt transmise aşa cum sunt prin modul. Ele nu sunt decriptate. Anteturile nu ar trebui să fie propagate către import. În caz contrar, din cauza dublării apare o eroare.	Anteturile de securitate sunt decriptate şi transmise prin modul cu verificarea semnăturii şi autentificare. Anteturile nu ar trebui să fie propagate către import. În caz contrar, din cauza dublării apare o eroare.

Configuraţi seturile de politici corespunzătoare pentru legările de export şi import, deoarece acestea izolează solicitantul serviciului de modificările aduse configuraţiei sau cerinţelor QoS ale furnizorului de servicii. Anteturile SOAP standard vizibile într-un modul pot fi utilizate pentru a influenţa prelucrarea (de exemplu, înregistrarea în istoric şi urmărirea) în modulul. Propagarea anteturilor SOAP printr-un modul ce aparţin de un mesaj recepţionat către un mesaj trimis înseamnă că beneficiile izolării prin modul sunt reduse.

Anteturilor standard, precum anteturilor WS-Security, nu ar trebui să fie propagate la cererea unui import sau la răspunsul unui export atunci când importul sau exportul au asociate un set de politici care în mod normal rezultă în generarea acestor anteturi. Altfel, va apărea o eroare din cauza duplicării anteturilor. În schimb, anteturile ar trebui să fie eliminate în mod explicit sau legarea de import sau export ar trebui să fie configurat în aşa fel încât să prevină propagarea anteturilor protocolului.

Accesarea anteturilor SOAP

Atunci când este primit un mesaj care conţine anteturi SOAP printr-un import sau export de serviciu web, anteturile sunt puse în secţiunea anteturi a obiectului mesaj al serviciului (SMO - service message object). Puteţi accesa informaţiile din antet, aşa cum este descris în “Accesarea informaţiilor din antetul SOAP în SMOO”.

Includerea schemei WS-Security într-un modul de aplicaţie

Procedura următoare defineşte paşii pentru includerea schemei în modulul aplicaţiei:

- În cazul în care computerul pe care rulează Integration Designer are acces la Internet, efectuaţi următorii paşi:
  1. În perspectiva Business Integration, selectaţi **Dependenţe** pentru proiectul dumneavoastră.
  2. Extindeţi **Resurse predefinite** şi selectaţi fie **Fişiere schemă WS-Security 1.0**, fie **Fişiere schemă WS-Security 1.1** pentru a importa schema în modulul dumneavoastră.
  3. Curăţaţi şi reconstruiţi proiectul.
- În cazul în care un computer pe care rulează Integration Designer nu are acces la Internet, puteţi descărca schema pe un al doilea computer care are acces la Internet. Apoi o puteţi copia pe computerul pe care rulează Integration Designer.
  1. De pe computerul care are acces la Internet, descărcaţi schema de la distanţă:
     a. Faceţi clic pe **Fişier > Import > Business Integration > WSDL şi XSD**.
     b. Selectaţi **WSDL la distanţă** sau **Fişier XSD**.
     c. Importaţi următoarele scheme:

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. Copiaţi schemele pe un computer care nu are acces la Internet.
3. De pe computerul care nu are acces la Internet, importaţi schema:
   a. Faceţi clic pe **Fişier > Import > Business Integration > WSDL şi XSD**.
   b. Selectaţi **WSDL Local** sau **Fişier XSD**.
4. Modificaţi locaţiile schemei pentru oasis-wss-wssecurity_secext-1.1.xsd:
   a. Deschideţi schema în *locaţie_spaţiu_de_lucru*/*nume_modul*/StandardImportFilesGen/oasis-wss-wssecurity-secext-1.1.xsd.
   b. Modificaţi:

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

      în:

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

   c. Modificaţi:

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

      în:

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

5. Modificaţi locaţia schemei pentru oasis-200401-wss-wssecurity-secext-1.0.xsd:
   a. Deschideţi schema în *locaţie_spaţiu_de_lucru*/*nume_modul*/StandardImportFilesGen/oasis-200401-wss-wssecurity-secext-1.0.xsd.
   b. Modificaţi:

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

      în:

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

6. Curăţaţi şi reconstruiţi proiectul.

Propagarea antetului Transport:

În momentul în care manipulaţi mesaje SOAP, este posibil să aveţi nevoie să accesaţi informaţii din anumite anteturi de transport din mesaje ce sunt primite, asiguraţi-vă că mesajele cu anteturi de transport sunt trimise cu anumite valori, sau permiteţi anteturilor de transport să parcurgă un modul.

Când configuraţi legarea unui serviciu web în Integration Designer, puteţi indica faptul că doriţi ca anteturile transport să fie propagate.

- Atunci când cererile sunt primite într-un export sau răspunsurile sunt primite la un import, informaţiile din antetul de transport pot fi accesate, permiţând ca logica din modul să se bazeze pe valorile antetului şi permiţând acestor anteturi să poată fi modificate.
- Atunci când răspunsurile sunt trimise dintr-un export sau cererile sunt trimise dintr-un import, anteturile de transport pot fi incluse în aceste mesaje.

Specificarea propagării anteturilor

Pentru a configura propagarea anteturilor de transport pentru un import sau export, realizaţi paşii următori:

1. Din vizualizarea Proprietăţi Integration Designer, selectaţi **Legare > Propagare**.

2. Setaţi opţiunea de propagare a antetului de transport de care aveţi nevoie.

Notă: Propagarea anteturilor de transport este dezactivată în mod implicit şi poate fi implementată doar în mediu de runtime Versiunea 7.0.0.3 (sau mai recentă). De asemenea, reţineţi faptul că, pentru versiunea 7.0.0.3, propagarea antetului de transport se limitează doar la anteturile de transportul HTTP.

Dacă activaţi propagarea anteturilor de transport, anteturile din mesajele primite vor fi propagate printr-un modul, iar dacă nu le eliminaţi în mod explicit aceste vor fi utilizate în invocaţiile ulterioare prin acelaşi fir de execuţie.

Notă: Anteturile de transport nu pot fi propagate atunci când folosiţi o legare pentru serviciul web (JAX-RPC).

Accesarea informaţiilor din antet

Atunci când propagarea antetului de transport este activată pentru mesajele primite, toate anteturile de transport (inclusiv cele definite de client) sunt vizibile în SMO (service message objec). Puteţi seta anteturile cu valori diferite sau puteţi crea altele noi. Cu toate acestea, reţineţi că nu există nicio verificare sau validare a valorilor setate de dvs., şi orice antet necorespunzător sau incorect poate cauza probleme în timpul execuţiei serviciului web.

Luaţi în considerare următoarele informaţii cu privire la setarea anteturilor HTTP:

- Orice modificare adusă anteturilor rezervate pentru motorul de servicii web nu vor fi onorate în mesajul de ieşire. De exemplu, versiunea sau metoda HTTP, anteturile Content-Type, Content-Length and SOAPAction sunt rezervate pentru motorul serviciului web.
- În cazul în care valoarea antetului este un număr, numărul (mai degrabă decât şirul) ar trebui stabilit în mod direct. De exemplu, utilizaţi **Max-Forwards = 5** (în loc de **Max-Forwards = Max-Forwards: 5**) şi **Age = 300** (în loc de **Age = Age: 300**).
- Dacă mesajul de solicitare are o dimensiune mai mică de 32 KB, motorul de servicii web înlătură antetul Transfer-Encoding şi setează în loc antetul Content-Length pentru dimensiune fixată a mesajului.
- Conţinutul de limbă este resetat de WAS.channel.http pe calea de răspuns.
- O setare invalidă pentru Modernizare rezultă într-o eroare 500.
- Anteturile următoare adaugă la sfârşit valorile rezervate de motorul serviciilor web la setările clientului:
  - User-Agent
  - Cache-Control
  - Pragma
  - Accept
  - Conexiune

Puteţi accesa informaţiile din antet într-unul din următoarele moduri:

- Utilizarea unei primitive de mediere pentru accesarea structurilor de tip SMO

  Vedeţi linkurile “Informaţii înrudite” pentru a găsi informaţii despre utilizarea primitivelor de mediere.
- Utilizarea serviciului de context SPI

  Următorul exemplu de cod citeşte anteturile de transport HTTP din serviciul de context:

```
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();
      }
}
```

Depanarea

În cazul în care întâlniţi probleme în momentul în care trimiteţi anteturi revizuite, puteţi intercepta mesajul TCP/IP folosind unelte precum TCP/IP Monitor din Integration Designer. Puteţi accesa TCP/IP Monitor selectând **Rulare/Depanare > TCP/IP Monitor** în pagina Preferinţe.

De asemenea puteţi vedea valorile antetului folosind urma motorului JAX-WS: **org.apache.axis2.*=all: com.ibm.ws.websvcs.*=all:**

Lucrul cu legări de servicii web (JAX-WS):

Atunci când utilizaţi legări de servicii web (JAX-WS) cu aplicaţiile dvs., aveţi posibilitatea să adăugaţi o calitate SAML (Security Assertion Markup Language) a serviciului (QOS) la legare. În primul rând trebuie să utilizaţi consola administrativă pentru a importa setul de politici. De asemenea, puteţi utiliza consola administrativă pentru a vă asigura că serverul este configurat în mod corespunzător pentru utilizarea legării (JAX-WS) a serviciului web.

Importul seturilor de politici:

Security Assertion Markup Language (SAML) este un standard OASIS bazat pe XML pentru schimbarea identităţii şi atributelor de securitate ale informaţiei. Atunci când configuraţi un serviciu web (JAX-WS) legat cu Integration Designer, puteţi specifica un set de politici SAML. Se foloseşte mai întâi consola administrativă IBM Business Process Manager pentru a face seturile de politici SAML disponibile, astfel încât ele să poată fi importate în Integration Designer.

Seturile de politici SAML sunt localizate în mod normal în directorul de configurare al profilului:

rădăcină_profil/config/templates/PolicySets

Înainte de a începe această procedură, se verifică dacă următoarele directoare (ce conţin seturile de politici) sunt localizate în directorul de configurare al profilului:

- 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
- Username WSHTTPS default

Dacă directoarele nu sunt în directorul de configurare al profilului, copiaţi-le în acel director din următoarea locaţie:

rădăcină_app_server/profileTemplates/default/documents/config/templates/PolicySets

Importaţi seturile de politici în consola administrativă, selectaţi una pe care doriţi să o faceţi disponibilă pentru Integration Designer şi apoi salvaţi un fişier .zip pentru fiecare dintre aceste seturi de politici într-o locaţie care este accesibilă prin Integration Designer.

1. Se vor importa seturile de politici prin urmărirea paşilor următori:
   a. Din consola administrativă, se face clic pe **Servicii > Seturi de politici > Seturi de politici aplicaţie**.

b. Se face clic pe **Import > Din magazia implicită**.
c. Se selectează seturile de politici implicite SAML şi se face clic pe **OK**.

2. Se vor exporta seturile de politici, astfel încât ele să poată fi utilizate de către Integration Designer:
   a. Din pagina Seturi de politici aplicaţie, se selectează setul de politici SAML care se vrea a fi exportat şi se face clic pe **Export**.

      Notă: Dacă pagina Seturi de politici aplicaţie nu este momentan afişată, se face clic pe **Servicii > Seturi de politici > Seturi de politici aplicaţie** din consola administrativă.
   b. Pe pagina următoare, se face clic pe legătura la fişierului .zip pentru setul de politici.
   c. În fereastra Descărcare fişier, se face clic pe **Salvare** şi se indică o locaţie ce este accesibilă prin Integration Designer.
   d. Se face clic pe **Înapoi**.
   e. Finalizaţi paşii de la 2a la 2d pentru fiecare set de politici care se vrea a fi exportat.

Seturile de politici SAML sunt salvate în fişierul .zip şi sunt gata să fie importate în Integration Designer.

Se vor importa seturile de politici în Integration Designer, aşa cum este descris în subiectul “Seturi de politici”.

Invocarea serviciilor web ce necesită autentificare de bază HTTP:

Autentificarea de bază HTTP are un nume şi parolă de utilizator pentru a autentifica un serviciu cleint la un punct final sigur. Puteţi seta autentificare de bază HTTP la trimiterea sau primirea cererilor de serviciu web.

Setaţi autentificarea de bază HTTP pentru recepţionarea cererilor de servicii web prin configurarea Java API pentru XML Web Services (JAX-WS) legături de export, aşa cum este descris în Crearea şi asignarea rolurilor de securitate pentru exportări de servicii web.

Autentificarea de bază HTTP poate fi activată pentru cererile de serviciu web ce sunt trimise de către o legare de import JAX-WS în unul sau două modalităţi:

- La configurarea legării de import într-un modul SCA, puteţi selecta setul politicii de autentificare HTTP furnizat denumit BPMHTTPBasicAuthentication (ce este furnizat prin intermediul legării de import a serviciului web (JAX-WS)) sau orice alt set de politică ce include politica HTTPTransport.
- La construirea modulului SCA, puteţi utiliza capabilităţi de mediere a fluxului pentru a crea dinamic un nou antet de autentificate HTTP şi specifica informaţii cu numele şi parola de utilizator în antet.

Notă: Setul de politică are precedenţă peste valoarea specificată în antet. Dacă doriţi să utilizaţi setul de valori în antetul de autentificare HTTP la momentul rulării, nu potriviţi un set de politică ce include politica HTTPTransport. Specific, nu utilizaţi setul de politică BPMHTTPBasicAuthentication şi dacă aveţi un set de politică definit, asiguraţi-vă că include politica HTTPTransport.

Pentru informaţii suplimentare despre seturile de politici pentru servicii web şi legături de politici şi cum sunt utilizate acestea, vedeţi Seturi de politici de servicii web din WebSphere Application Server Information Center.

- Pentru a utiliza setul de politică furnizat, realizaţi următorii paşi:
  1. Opţional: În consola administrativă, creaţi o legătură de politică generală sau editaţi una existentă care include politica HTTPTransport cu valorile ID-ului de utilizator şi parolei necesare.
  2. În IBM Integration Designer, generaţi o legare de import a serviciului web (JAX-WS) şi ataşaţi setul de politică BPMHTTPBasicAuthentication.
  3. Realizaţi *unul* din următorii paşi:
     – În IBM Integration Designer, în proprietăţile de legare de import al serviciului web (JAX-WS), specificaţi numele unei legări a politicii existente de client generale ce include politica HTTPTransport.

– După implementarea modulului SCA, utilizaţi consola administrativă fie pentru a selecta o legătură de politică pentru un client existent, fie creaţi o nouă legătură de politică de client şi apoi o asociaţi cu legarea de import.

4. Opţional: În consola administrativă a serverului Process editaţi legarea setului de politici selectat pentru a specifica ID-ul şi parola necesare.

- Pentru a specifica numele şi parola de utilizator în antetul de autentificare HTTP, realizaţi unul din următorul set de paşi:
  – Utilizaţi medierea simplă HTTP Header Setter în IBM Integration Designer pentru a crea antetul de autentificare HTTP şi specificaţi numele şi parola de utilizator.
  – Dacă o logică suplimentară este necesară, utilizaţi codul Java într-o mediere simplă personalizată (asemenea următorului exemplu) pentru a:
    1. Crea un antet de autentificare HTTP.
    2. Specifica informaţii privind numele şi parola de utilizator.
    3. Adăuga un nou antet de autentificare HTTP la HTTPControl.
    4. Seta HTTPControl actualizat înapoi în serviciul Context.

```
//Obţineţi HeaderInfoType din contextService
ContextService contextService = (ContextService) ServiceManager.INSTANCE
.locateService("com/ibm/bpm/context/ContextService");
HeaderInfoType headers = contextService.getHeaderInfo();
if(headers == null){
    headers = ContextObjectFactory.eINSTANCE.createHeaderInfoType();
}
//Get the HTTP header and HTTP Control from HeaderInfoType
HTTPHeaderType httpHeaderType = headers.getHTTPHeader();
HTTPControl cp = httpHeaderType.getControl();
HeadersFactory factory = HeadersFactory.eINSTANCE;
if(cp == null){
    cp = factory.createHTTPControl();
}
//Creaţi HTTPAuthentication nou şi setaţi 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);
// Setaţi informaţii de antet înapoi în contextul curent de execuţie.
contextService.setHeaderInfo(headers);
```

Verificarea configuraţiei serverului:

Atunci când implementaţi o aplicaţie cu legarea unui serviciu web (JAX-WS), trebuie să vă asiguraţi că serverul pe care este implementată aplicaţia nu are selectată opţiunea **Pornire componente după cum este necesar**.

Se poate verifica pentru a vedea dacă această opţiune este selectată prin realizarea următorilor paşi din consola administrativă:

1. Se face clic pe **Servere > Tipuri de servere > Servere de aplicaţii WebSphere**.
2. Se face clic pe nume server.
3. Din fişa Configuraţie, se determină dacă este selectat **Pornire componente după cum este necesar**.
4. Se vor realiza următorii paşi:
   - Dacă **Pornire componente după cum este necesar** este selectat, se va şterge caseta de bifare şi apoi se va face clic pe **Aplicare**.
   - Dacă **Pornire componente după cum este necesar** nu este selectat, se va faceţi clic pe **Anulare**.

Ataşamente în mesajele SOAP:

Aveţi posibilitatea să trimiteţi şi să primiţi mesaje SOAP care includ date binare (precum fişiere PDF sau imagini JPEG) sub formă de ataşamente. Ataşamentele pot fi cu *referinţă* (adică sunt reprezentate în mod explicit ca părţi componente ale mesajului în interfaţa serviciului) sau *fără referinţă* (în care pot fi incluse numere arbitrare şi tipuri de ataşamente).

Un ataşament la care se face referire poate fi reprezentat în unul dintre următoarele moduri:

- Ataşamentele MTOM folosesc codarea specifică SOAP Message Transmission Optimization Mechanism (http://www.w3.org/TR/soap12-mtom/). Ataşamentele MTOM sunt activate prin opţiuni de configurare în legările de import şi de export şi reprezintă modul recomandat de codare a ataşamentelor pentru noi aplicaţii.
- Sub forma unui element de tip wsi:swaRef-typed în schema mesajului

  Ataşamentele definite folosind tipul wsi:swaRef conform WS-I (Web Services Interoperability Organization), *Attachments Profile Version 1.0* (http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html), care defineşte modul în care elementele mesajului sunt legate de părţile componente MIME.
- Ca o parte componentă de nivel înalt a mesajului, folosiţi un tip de scheme binare

  Ataşamentele reprezentate sub formă de părţi componente de nivel înalt ale mesajului în concordanţă cu specificaţiile *Mesaje SOAP cu Ataşamente* (http://www.w3.org/TR/SOAP-attachments).

  Ataşamentele reprezentate sub formă de părţi componente de nivel înalt pot fi de asemenea configurate pentru a vă asigura că mesajele şi documentele WSDL produse de legare în concordanţă cu WS-I *Attachments Profile Version 1.0* şi WS-I *Basic Profile Version 1.1* (http://www.ws-i.org/Profiles/BasicProfile-1.1.html).

Un ataşament către care nu se face referinţă este purtat într-un mesaj SOAP fără nici o reprezentare în schema mesajului.

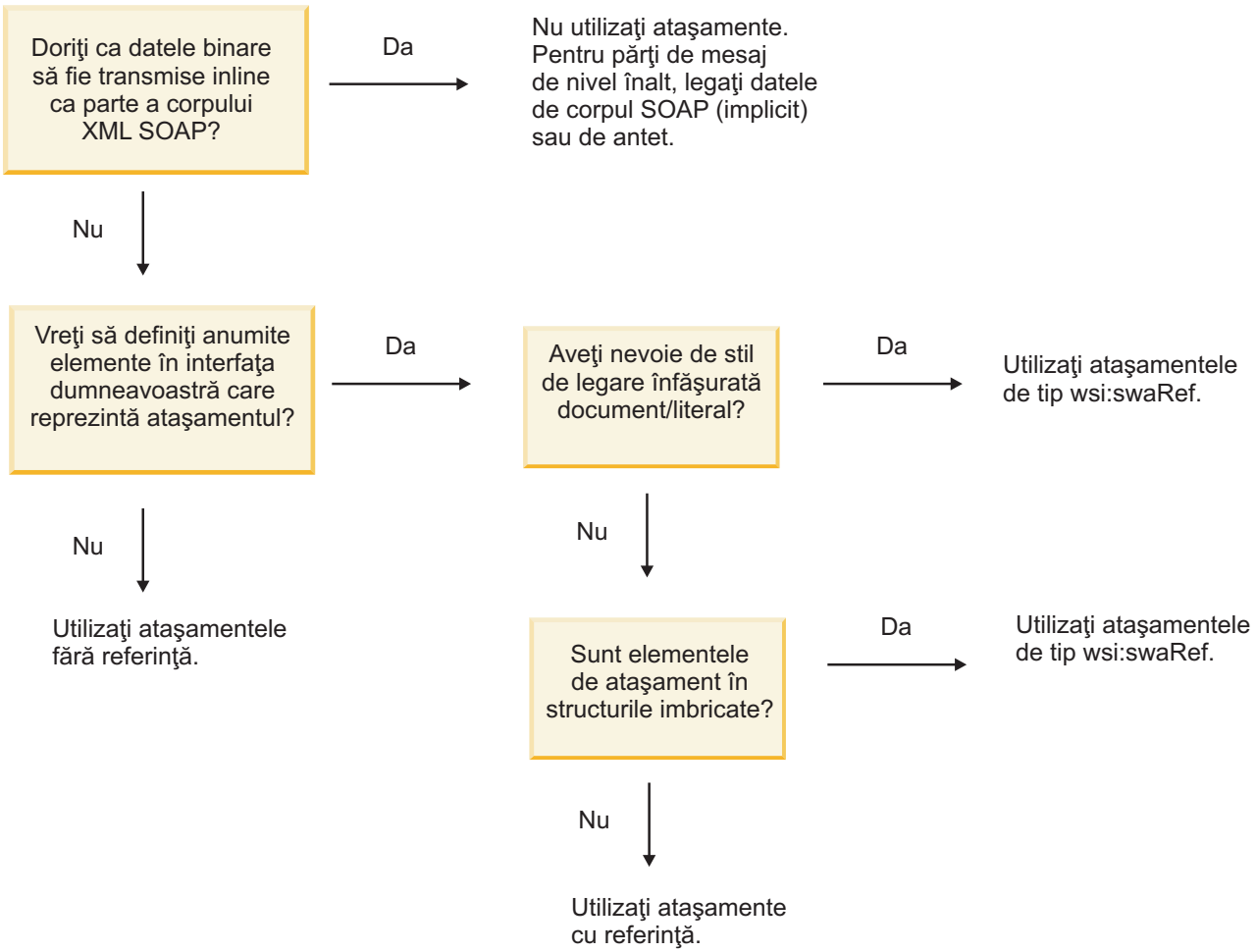
În toate cazurile, ataşamentele MTOM, legarea WDSL SOAP ar trebui să includă legarea MIME pentru ataşamente care vor fi folosite, şi dimensiunea maximă a ataşamentelor nu ar trebui să depăşească 20 MB.

Notă: Pentru a trimite sau a primi mesaje SOAP cu ataşamente, trebuie să folosiţi una dintre legările serviciului web care se bazează pe JAX-WS (Java API for XML Web Services).

Cum alegeţi stilul corespunzător de ataşamente:

Când proiectaţi o interfaţă nouă pentru serviciu care include date binare, luaţi în considerare modul în care datele binare sunt transportate în mesajele SOAP care sunt trimise şi primite de către serviciu.

Message Transmission Optimization Mechanism (MTOM) ar trebui folosit pentru ataşamente dacă aplicaţia serviciului web conectat suportă acest lucru. Dacă nu, diagrama următoare arată modul în care sunt alese alte stiluri de ataşamente. Utilizaţi următorul set de întrebări pentru a determina stilul corespunzător de ataşamente:



Ataşamente MTOM: părţi de mesaj la nivel de top:

Puteţi trimite şi primi mesaje de servicii web care includ ataşamente MTOM (Message Transmission Optimization Mechanism - Mecanism de optimizare a transmisiei mesajului) SOAP. Într-un mesaj SOAP cu mai multe părţi MIME, corpul SOAP este prima parte a mesajului, iar ataşamentul sau ataşamentele sunt în părţi ulterioare.

Trimiţând sau primind un ataşament cu referinţă într-un mesaj SOAP, datele binare care alcătuiesc ataşamentul (care deseori este destul de mare) sunt păstrate separat de corpul mesajului SOAP, pentru a nu fi necesară parsarea lor ca XML. Acest lucru rezultă în mai multe procesări eficiente decât dacă datele binare ar fi fost ţinute într-un element XML.

Iată un eşantion de mesaj MTOM SOAP:

```
... 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>

... aici sunt datele binare ...
--MIMEBoundaryurn_uuid_0FE43E4D025F0BF3DC11582467646812--
```

Reţineţi că în eşantionul MTOM, tipul de conţinut pentru plicul SOAP este **application/xop+xml** şi datele binare sunt înlocuite de un element **xop:Include** ca mai jos:

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

Procesarea la intrare a ataşamentelor cu referinţă

Atunci când un client trece un mesaj SOAP cu un ataşament către o componentă Service Component Architecture (SCA), legarea de export (JAX-WS) a serviciului web înlătură mai întâi ataşamentul. Apoi transmite partea de tip SOAP a mesajului şi creează un obiect business. În sfârşit, legarea setează binarele ataşamentului în obiectul business.

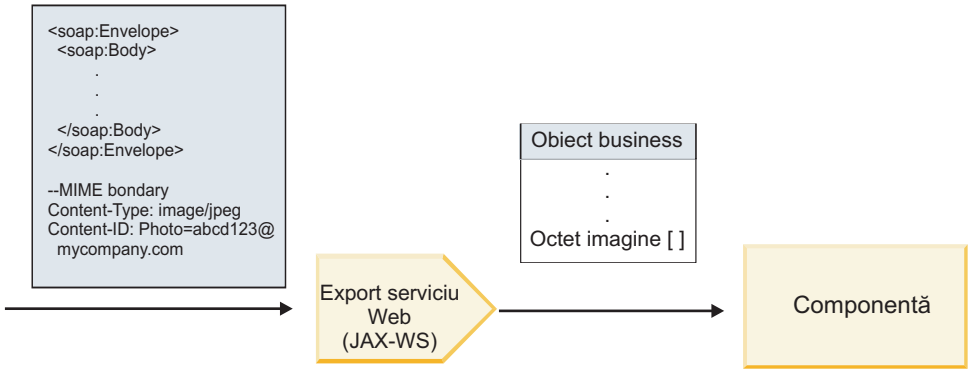


Figura 13. Cum procesează legarea export (JAX-WS) a serviciului web un mesaj SOAP cu un ataşament referit

Atribute ataşament MTOM

- MTOM poate suporta elemente ataşament în structuri imbricate.
- MTOM este disponibil numai pentru tipul base64Binary.
- MTOM poate suporta elemente ataşament în structuri imbricate, ceea ce înseamnă că **bodyPath** pentru ataşamentele MTOM este locaţia **xpath** pentru elementul unde este ţinut ataşamentul MTOM. Logica de calcul pentru **bodyPath** urmează strict schema de generare a locaţiei **xpath** aşa cum se arată în exemplele de mai jos:
  - Pentru un tip non-matrice (**maxOccurs** este 1): /sendImage/input/imageData
  - Pentru un tip matrice (**maxOccurs** > 1): /sendImage/input/imageData[1]
- Tipurile de ataşament amestecate nu sunt suportate, ceea ce înseamnă că, dacă MTOM este activat pe legarea de import, se va genera ataşamentul MTOM. Dacă MTOM este dezactivat sau dacă valoarea de configurare MTOM este lăsată la valoarea implicită pe legarea de export, mesajul MTOM de intrare nu este suportat.

Ataşamente către care se face referinţă: ataşamente de tip swaRef:

Aveţi posibilitatea să trimiteţi şi să primiţi mesaje SOAP care includ ataşamente care sunt reprezentate în interfaţa serviciului sub formă de elemente de tip swaRef.

Un element de tip swaRef este definit în WS-I (Web Services Interoperability Organization) *Attachments Profile* Versiunea 1.0 (http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html), în care este definit modul în care elementele mesajului sunt înrudite cu părţile componente MIME.

În mesajul SOAP, corpul conţine un element de tip swaRef care identifică ID-ul conţinutului ataşamentului.

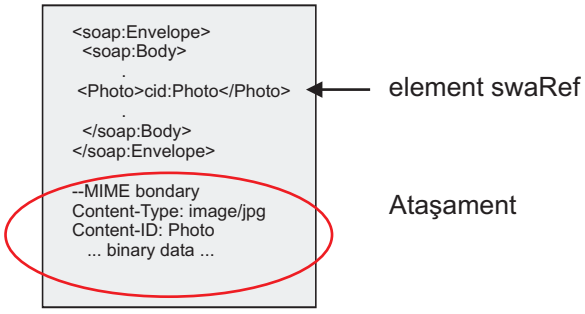


Figura 14. Un mesaj SOAP cu un element swaRef

WSDL-ul pentru acest mesaj SOAP conţine un element de tip swaRef într-o parte componentă a mesajului care identifică ataşamentul.

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

WSDL-ul ar trebui să conţină de asemenea o legare MIME care să indice mesajele MIME compuse din mai multe părţi ce urmează să fie utilizate.

Notă: WSDL *nu* include o legare MIME pentru elementul de mesaj de tip swaRef specific, deoarece legările MIME se aplică doar asupra părţilor componente de nivel înalt ale mesajului.

Ataşamentele reprezentate sub formă de elemente de tip swaRef pot fi transmise doar peste componentele fluxului de mediere. În cazul în care un ataşament trebuie să fie accesat sau transmis către o componentă de un alt tip, folosiţi o componentă a fluxului de mediere pentru a muta ataşamentul la o locaţie care este accesibilă acelei componente.

Procesarea la intrare a ataşamentelor

Folosiţi Integration Designer pentru a configura o legare de export, astfel încât să recepţioneze ataşamentul. Creaţi un modul şi interfaţa şi operaţiile sale asociate, incluzând un element de tip swaRef. Puteţi crea apoi o legare (JAX-WS) pentru serviciul web.

Notă: Vedeţi subiectul “Lucrul cu ataşamente” din Centrul de informare Integration Designer pentru informaţii mai detaliate.

Atunci când un client transmite un mesaj SOAP cu un ataşament swaRef către o componentă SCA (Service Component Architecture), legarea de export (JAX-WS) a serviciului web înlătură mai întâi ataşamentul. Apoi transmite partea de tip SOAP a mesajului şi creează un obiect business. În sfârşit, legarea setează Id-ul conţinutului ataşamentului în obiectul business.

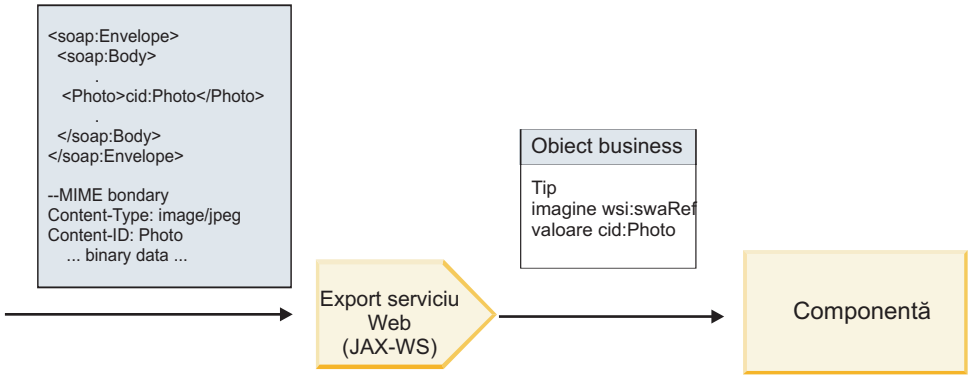


Figura 15. Modul în care o legare externă (JAX-WS) a serviciului web procesează un mesaj SOAP cu un ataşament swaRef

Accesarea metadatelor ataşamentului într-o componentă a fluxului de mediere

Aşa cum se arată în Figura 16, atunci când ataşamentele de tip swaRef sunt accesate de componente, identificatorul conţinutului ataşamentului apare sub forma unui element de tip swaRef.

Fiecare ataşament al unui mesaj SOAP are, de asemenea, un element **attachments** corespunzător în SMO. Atunci când este folosit tipul swaRef WS-I, elementul **attachments** include tipul conţinutul ataşamentului şi ID-ul conţinutului, precum şi datele binare reale ale ataşamentului.

Pentru a putea obţine valoarea unui ataşament de tip swaRef, este necesară obţinerea valorii elementului de tip swaRef, iar apoi localizarea elementului **attachments** cu valoarea **contentID** corespunzătoare. Reţineţi că valoarea **contentID** are de obicei prefixul **cid:** înlăturat din valoarea swaRef.

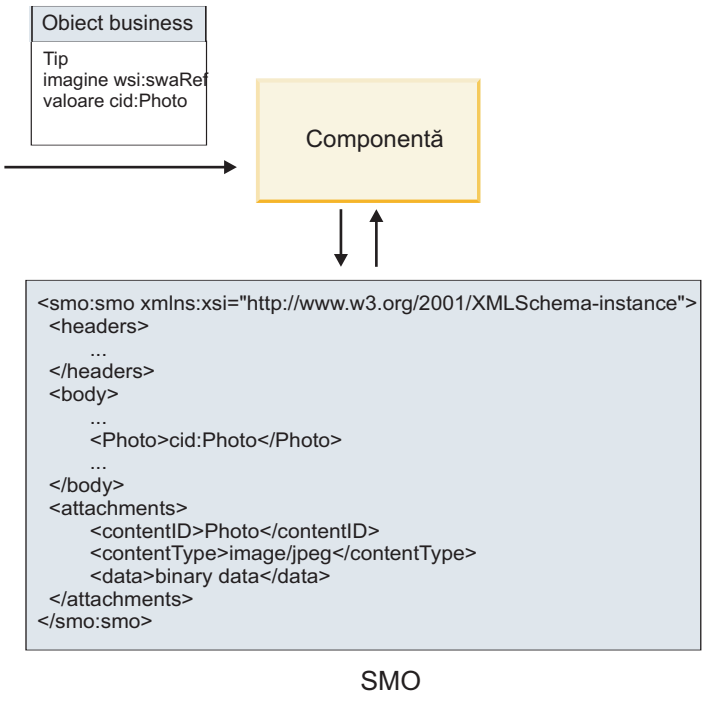


Figura 16. Modul în care ataşamentele swaRef apar în SMO

Procesarea la ieşire

Utilizaţi Integration Designer pentru a configura legarea de import (JAX-WS) a serviciului web pentru a invoca un serviciu web extern. Legarea de import este configurată cu un document WSDL care descrie serviciul web ce trebuie invocat şi defineşte ataşamentul care va fi trimis către serviciul web.

Atunci când legarea unui serviciu web (JAX-WS) primeşte un mesaj SCA, elementele de tip swaRef sunt trimise ca ataşament în cazul în care importul este legat la componenta unui flux de mediere şi elementul de tip swaRef are un element **attachments** corespunzător.

Pentru procesarea la ieşire, elementele de tip swaRef sunt trimise întotdeauna cu valorile lor de ID de conţinut; totuşi, modulul de mediere trebuie să se asigure că există un element **attachments** corespunzător valorii **contentID**.

Notă: Pentru a fi în concordanţă cu WS-I Attachments Profile, valoarea **content ID** ar trebui să fie conformă "codării id-ului de conţinut," aşa cum este descrisă în secţiunea 3.8 din WS-I *Attachments Profile* 1.0.

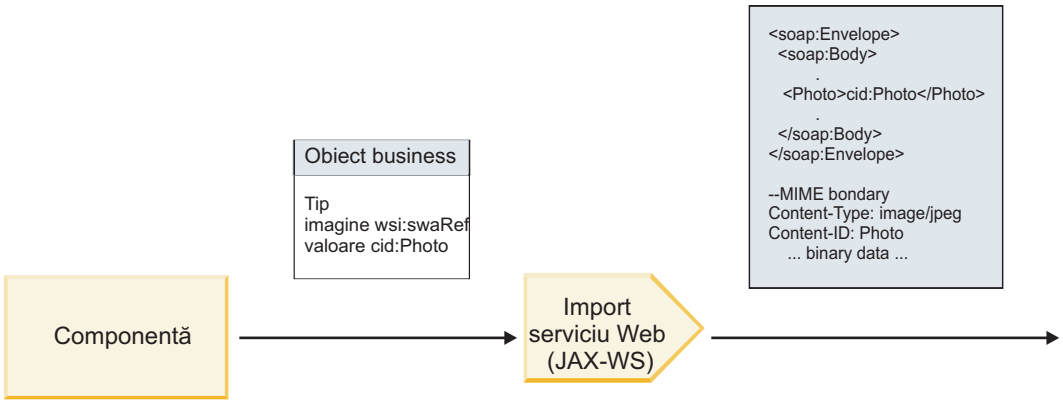


Figura 17. Modul în care o legare de import (JAX-WS) a serviciului web generează un mesaj SOAP cu un ataşament swaRef

Setarea metadatelor ataşamentului într-o componentă a fluxului de mediere

În cazul în care, în SMO există o valoarea pentru un element de tip swaRef şi un element **attachments**, legarea pregăteşte mesajul SOAP (cu ataşament) şi îl trimite către un destinatar.

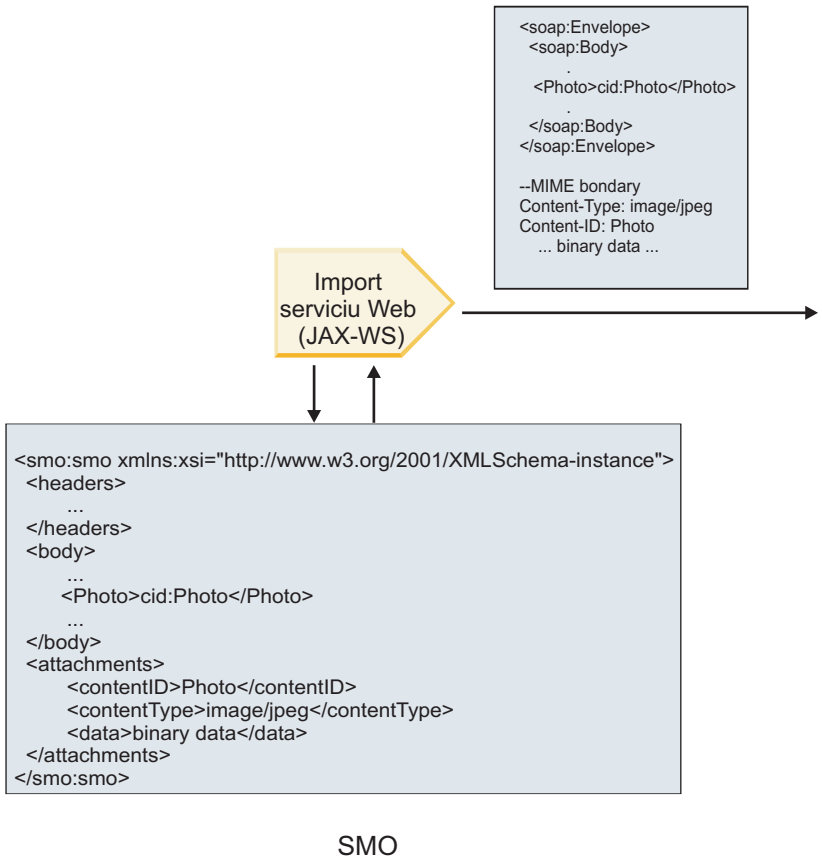


Figura 18. Modul în care un ataşament de tip swaRef din SMO este accesat pentru a crea un mesaj de tip SOAP

Elementul **attachments** este prezent în SMO doar dacă o componentă a fluxului de mediere este conectată direct la import sau la export; aceasta nu a trecut prin alte tipuri de componente. În cazul în care valorile sunt necesare într-un modul care conţine alte tipuri de componente, ar trebui să fie utilizată o componentă a fluxului de mediere pentru a copia valorile într-o locaţie în care pot fi accesate din modul, şi o altă componentă a fluxului de mediere utilizată pentru a seta valorile corecte înainte de o invocare de ieşire prin intermediul unui serviciu web de import.

Important: Aşa cum este descris în “Reprezentare XML a SMO,” primitiva de mediere Mapping transformă mesajele utilizând o transformare XSLT 1.0. Transformarea operează asupra unei serializări XML a SMO. Primitiva de mediere Mapping permite specificarea rădăcinii de serializare şi elementul rădăcină a documentului XML reflectă această rădăcină.

Atunci când trimiteţi mesaje SOAP cu ataşamente, elementul rădăcină pe care îl alegeţi determină modul în care ataşamentele sunt propagate.

- Dacă folosiţi “/body” drept rădăcină a mapării XML, toate ataşamentele sunt propagate implicit de-a lungul mapării.
- Dacă folosiţi “/” drept rădăcină pentru mapare, puteţi controla propagarea ataşamentelor.

Ataşamente la care se face referire: părţi componente de nivel înalt ale mesajului:

Aveţi posibilitatea să trimiteţi şi să primiţi mesaje SOAP care includ ataşamente binare ce sunt declarate ca părţi componente în interfaţa serviciului dumneavoastră.

Într-un mesaj SOAP cu mai multe părţi MIME, corpul SOAP este prima parte a mesajului, iar ataşamentul sau ataşamentele se află în părţile componente următoare.

Care este avantajul de a trimite sau de a primi un ataşament cu referinţă într-un mesaj SOAP? Datele binare care alcătuiesc ataşamentul (care este adesea destul de mare), sunt ţinute separat de corpul mesajului SOAP, astfel încât acesta să nu trebuiască să fie parsat ca XML. Acest lucru rezultă în mai multe procesări eficiente decât dacă datele binare ar fi fost ţinute într-un element XML.

Tipuri de mesaje SOAP cu ataşamente către care se face referinţă

Începând cu versiunea 7.0.0.3 a IBM Business Process Manager, aveţi posibilitatea de a alege modul în care mesajul SOAP este generat:

- **Mesaje conforme cu WS-I**

  Runtime-ul poate genera mesaje SOAP care sunt conforme cu WS-I *Attachments Profile Version 1.0* şi cu WS-I *Basic Profile Version 1.1*. Într-un mesaj SOAP, care este în conformitate cu aceste profiluri, doar o singură parte este legată de corpul SOAP; pentru cele care sunt legate ca ataşamente, se utilizează codarea părţii content-id (aşa cum este descrisă în WS-I *Attachments Profile Version 1.0*) pentru a asocia ataşamentul la partea componentă a mesajului.
- **Mesaje care nu sunt conforme cu WS-I**

  Runtime-ul poate genera mesaje SOAP care sunt conforme cu profilurile WS-I, dar care sunt compatible cu mesajele generate în Versiunea 7.0 sau 7.0.0.2 a IBM Business Process Manager. Mesajele SOAP folosesc elemente de nivel înalt numite după partea mesajului care are un atribut **href** care reţine ataşamentul **content-id**, dar codarea părţii content-id nu este folosită (aşa cum este descris în WS-I *Attachments Profile Version 1.0*).

Selectarea compatibilităţii WS-I pentru exporturile de servicii web

Folosiţi Integration Designer pentru a configura o legare de export. Creaţi un modul şi interfaţa şi operaţiile sale asociate. Puteţi crea apoi o legare (JAX-WS) pentru serviciul web. Pagina Ataşamente cu referinţă afişează toate părţile componente binare care aparţin de operaţiile create, apoi selectaţi care părţi vor fi ataşate. Apoi specificaţi, în pagina Specificare compatibilitatea WS-I AP 1.0 pentru Integration Designer, una dintre următoarele alegeri:

- **Utilizare WS-I AP 1.0 compatibil cu mesajul SOAP**

  Dacă selectaţi această opţiune, specificaţi de asemenea şi care parte a mesajului ar trebui să fie legată de corpul SOAP.

  Notă: Această opţiune poate fi utilizată doar atunci când fişierul WSDL corespunzător este de asemenea conforme cu WS-I .

  Un fişier WSDL care este generat de Integration Designer Versiunea 7.0.0.3 este în conformitate cu WS-I. Totuşi, dacă importaţi un fişier WSDL care nu este în conformitate cu WS-I, nu puteţi selecta această opţiune.
- **Utilizare mesaje SOAP care nu sunt conforme cu WS-I AP 1.0**

  Dacă selectaţi această opţiune, care este implicită, prima parte a mesajului este legată de corpul SOAP.

Notă: Doar părţile de nival înalt ale mesajului (adică, elementele definite în portType WSDL drept părţi componente din cadrul mesajului de intrare sau ieşire) care au un tip binar (fie base64Binary, fie hexBinary) pot fi trimise sau primite sub formă de ataşamente cu referinţă.
Vedeţi subiectul “Lucrul cu ataşamente” din Centrul de informare Integration Designer pentru detalii suplimentare.

Pentru mesajele conforme WS-I, content-ID-ul care este generat în mesajul SOAP este o concatenare a următoarelor elemente:

- Valoarea atributului **nume** din elementul **wsdl:part** către care se face referinţă prin **mime:content**
- Caracterul **=**
- O valoare unică globală, precum un UUID
- Caracterul **@**
- Un nume de domeniu valid

Procesarea la intrare a ataşamentelor cu referinţă

Atunci când un client transmite un mesaj SOAP cu un ataşament către o componentă SCA (Service Component Architecture), legarea de export (JAX-WS) a serviciului web înlătură mai întâi ataşamentul. Apoi transmite partea de tip SOAP a mesajului şi creează un obiect business. În sfârşit, legarea setează binarele ataşamentului în obiectul business.

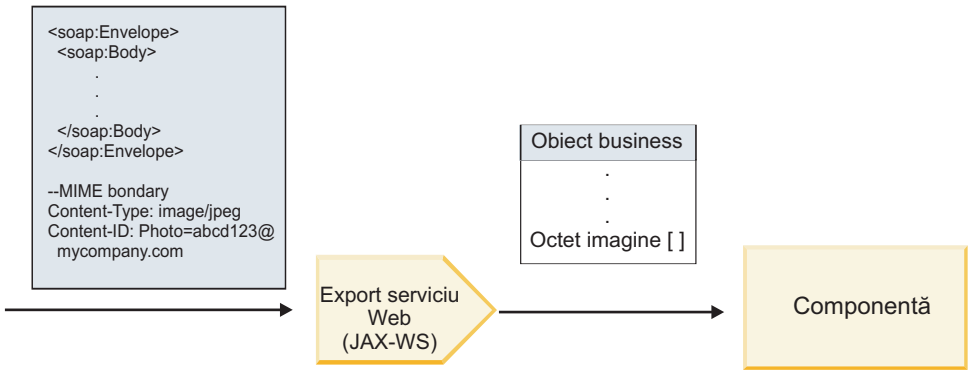


Figura 19. Modul în care o legare externă (JAX-WS) a serviciului web procesează un mesaj SOAP conform WS-I cu un ataşament către care se face o referinţă

Accesarea metadatelor ataşamentului într-o componentă a fluxului de mediere

Aşa cum se arată în Figura 19, atunci când ataşamentele cu referinţă sunt accesate de componente, datele ataşamentului apar sub forma unei matrice cu date de tip octet.

Fiecare ataşament cu referinţă al unui mesaj SOAP are, de asemenea, un element **attachments** corespunzător în SMO. Elementul **attachments** include tipul conţinutul ataşamentului şi calea către corpul mesajului în care este păstrat ataşamentul.

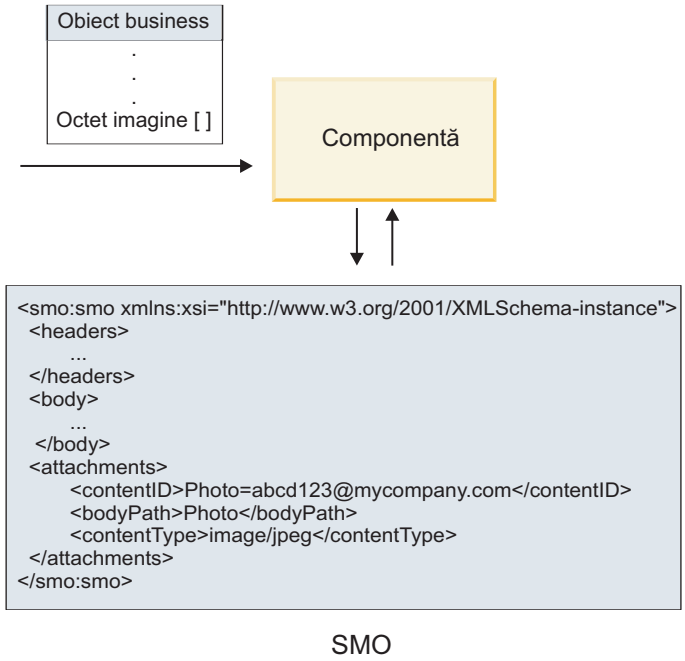


Figura 20. Modul în care ataşamentele către care se face referinţă apar în SMO

Important: Calea către elementul corp al mesajului nu este actualizată automat în cazul în care mesajul este transformat şi ataşamentul mutat. Puteţi utiliza fluxul de mediere pentru a actualiza elementul **attachments** cu noua cale (de exemplu, ca parte a transformării sau a utilizării unui setter separat pentru elementul mesaj).

Modul în care sunt construite mesajele SOAP de ieşire

Utilizaţi Integration Designer pentru a configura legarea de import (JAX-WS) a serviciului web pentru a invoca un serviciu web extern. Legarea de import este configurată cu un document WSDL care descrie serviciul web ce trebuie invocat şi defineşte ce părţi din mesaj ar trebui să fie trimise ca ataşamente. De asemenea, puteţi indicata, în pagina Specificare compatibilităţii cu WS-I AP 1.0 a Integration Designer, una dintre următoarele alegeri:

- **Utilizare WS-I AP 1.0 compatibil cu mesajul SOAP**

  Dacă selectaţi această opţiune, specificaţi de asemenea şi care parte a mesajului ar trebui să fie legată de corpul SOAP; toate celelalte sunt legate de ataşamente sau anteturi. Mesajele trimise de legare nu includ elemente în corpul SOAP care se referă la echipamente; relaţia este exprimată prin intermediul ID-ului de conţinut al ataşamentului, inclusiv numele părţii mesajului.
- **Utilizare mesaje SOAP care nu sunt conforme cu WS-I AP 1.0**

  Dacă selectaţi această opţiune, care este implicită, prima parte a mesajului este legată de corpul SOAP; toate celelalte sunt legate de ataşamente sau anteturi. Mesajele trimise de legare nu includ unul sau mai multe elemente în corpul SOAP care se referă la ataşamente prin intermediul unui atribut **href**.

Notă: Partea care reprezintă un ataşament, aşa cum este definit în WSDL, trebuie să fie de un tip simplu (fie base64Binary, fie hexBinary). În cazul în care o parte este definită de un complexType, acea parte nu poate fi legată ca un ataşament.

Procesarea la intrare a ataşamentelor cu referinţă

Legarea de import foloseşte informaţiile din SMO pentru a determina modul în care părţile componente de nivel înalt ale mesajului sunt trimise ca ataşamente.

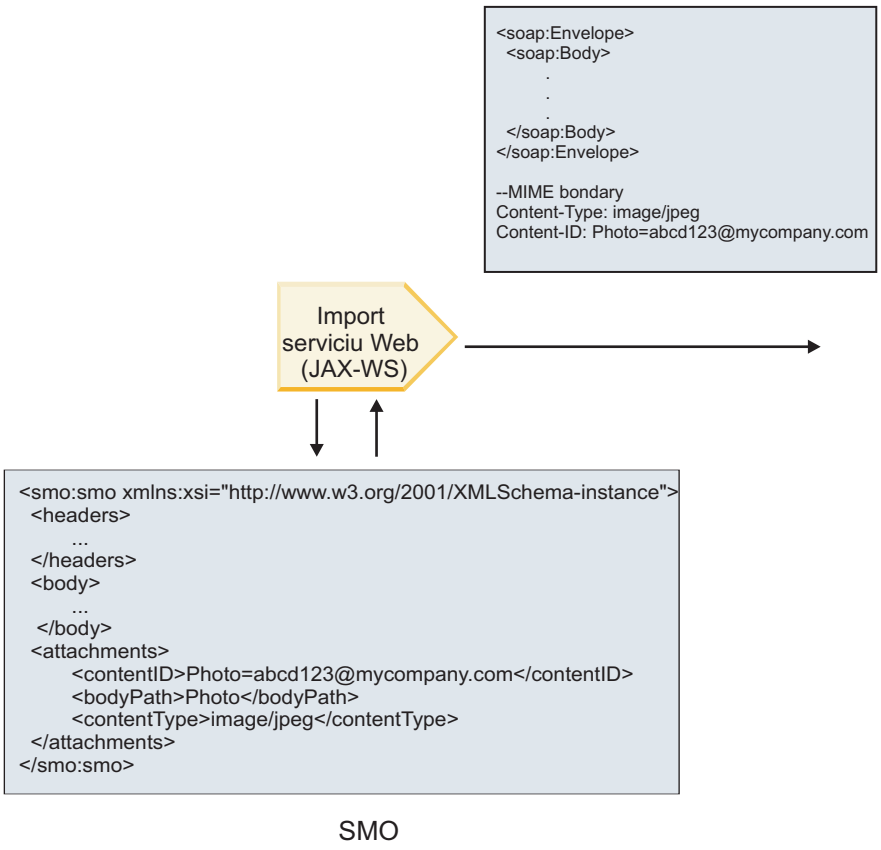


Figura 21. Modul în care ataşamentul cu referinţă din SMO este accesat pentru a crea mesajul SOAP

Elementul **attachments** este prezent în SMO doar dacă o componentă a fluxului de mediere este conectată direct la import sau la export; aceasta nu a trecut prin alte tipuri de componente. În cazul în care valorile sunt necesare într-un modul care conţine alte tipuri de componente, ar trebui să fie utilizată o componentă a fluxului de mediere pentru a copia valorile într-o locaţie în care pot fi accesate din modul, şi o altă componentă a fluxului de mediere utilizată pentru a seta valorile corecte înainte de o invocare de ieşire prin intermediul unui serviciu web de import.

Legarea foloseşte o combinaţie între următoarele condiţii pentru a determina modul în care (sau dacă) este trimis mesajul:

- În cazul în care există o legare MIME WSDL pentru partea binară de nivel înalt a mesajului, iar dacă există, modul în care este definit tipul conţinutului
- În cazul în care există un element **attachments** în SMO a cărui valoare **bodyPath** face referinţă către o parte binară de nivel înalt

Modul în care sunt create ataşamentele atunci când există un element attachment în SMO

Tabelul următor arată modul în care este creat şi trimis un ataşament în cazul în care SMO conţine un element **atachment** cu un **bodyPath** care se potriveşte cu o parte a numelui mesajului:

Tabela 27. Modul în care este generat ataşamentul

Starea legării WSDL MIME pentru partea componentă binară de nivel înalt a mesajului	**Modul în care mesajul este creat şi transmis**
Prezent cu una dintre următoarele: • Nu este definit un tip de conţinut pentru partea componentă a mesajului • Sunt definite mai multe tipuri de conţinut • Este definit tipul conţinutului metacaracterului	Partea componentă a mesajului este trimisă ca un ataşament. **Content-Id** este setat cu o valoare în elementul attachments, în cazul în care acesta este prezent; altfel, se generează una. **Content-Type** este setat cu o valoare în elementul attachments în cazul în care acesta este prezent; altfel, este setat cu **application/octet-stream**.
Prezent cu conţinut unic, fără metacaractere pentru partea componentă a mesajului	Partea componentă a mesajului este trimisă ca un ataşament. **Content-Id** este setat cu o valoare în elementul attachments, în cazul în care acesta este prezent; altfel, se generează una. **Content-Type** este setat cu o valoare în elementul attachments, în cazul în care acesta este prezent; altfel, este setat cu tipul definit în elementul de conţinut WSDL MIME.
Nu este prezent	Partea componentă a mesajului este trimisă ca un ataşament. **Content-Id** este setat cu o valoare în elementul attachments, în cazul în care acesta este prezent; altfel, se generează una. **Content-Type** este setat cu o valoare în elementul attachments în cazul în care acesta este prezent; altfel, este setat cu **application/octet-stream**. **Notă:** Trimiţând părţi componente ale sub formă de ataşamente atunci când nu sunt definite ca atare în WSDL poate întrerupe compatibilitatea cu WS-I Attachments Profile 1.0, iar acest lucru ar trebui evitat pe cât posibil.

Modul în care sunt create ataşamentele atunci când nu există nici un element attachment în SMO

Tabelul următor arată modul în care este creat şi trimis un ataşament în cazul în care SMO nu conţine un element **atachment** cu un **bodyPath** care se potriveşte cu o parte din numele mesajului:

Tabela 28. Modul în care este generat ataşamentul

Starea legării WSDL MIME pentru partea componentă binară de nivel înalt a mesajului	**Modul în care mesajul este creat şi transmis**
Prezent cu una dintre următoarele: • Nu este definit un tip de conţinut pentru partea componentă a mesajului • Sunt definite mai multe tipuri de conţinut • Este definit tipul conţinutului metacaracterului	Partea componentă a mesajului este trimisă ca un ataşament. **Content-Id** este generat. **Content-Type** este setat cu **application/octet-stream**.
Prezent cu conţinut unic, fără metacaractere pentru partea componentă a mesajului	Partea componentă a mesajului este trimisă ca un ataşament. **Content-Id** este generat. **Content-Type** este setat cu tipul definit în elementul conţinut WSDL MIME.
Nu este prezent	Partea componentă a mesajului nu este trimisă ca un ataşament.

Important: Aşa cum este descris în “Reprezentare XML a SMO,” primitiva de mediere Mapping transformă mesajele utilizând o transformare XSLT 1.0. Transformarea operează asupra unei serializări XML a SMO. Primitiva de mediere Mapping permite specificarea rădăcinii de serializare şi elementul rădăcină a documentului XML reflectă această rădăcină.

Atunci când trimiteţi mesaje SOAP cu ataşamente, elementul rădăcină pe care îl alegeţi determină modul în care ataşamentele sunt propagate.

- Dacă folosiţi “/body” drept rădăcină a mapării XML, toate ataşamentele sunt propagate implicit de-a lungul mapării.
- Dacă folosiţi “/” drept rădăcină pentru mapare, puteţi controla propagarea ataşamentelor.

Ataşamente fără referinţă:

Aveţi posibilitatea să trimiteţi şi să primiţi ataşamente *fără referinţă* care nu sunt declarate în interfaţa serviciului.

Într-un mesaj SOAP cu mai multe părţi MIME, corpul SOAP este prima parte a mesajului, iar ataşamentele se află în părţile următoare. Nu este inclusă nici o referinţă către ataşament în corpul SOAP.

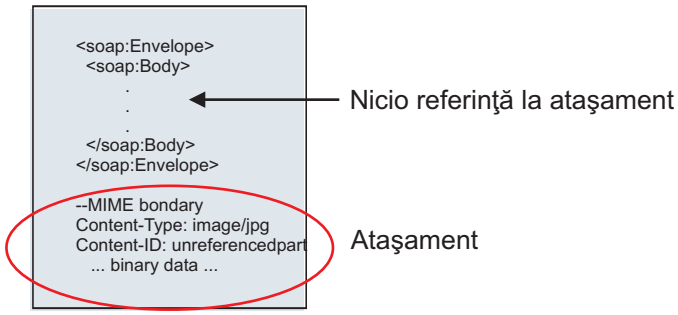


Figura 22. Un mesaj SOAP cu un ataşament către care nu se face referinţă

Aveţi posibilitatea să trimiteţi un mesaj SOAP cu un ataşament fără referinţă printr-un export de serviciu web către importul unui serviciu web. Mesajul de ieşire, care este trimis la serviciul web ţintă, conţine ataşamentul.

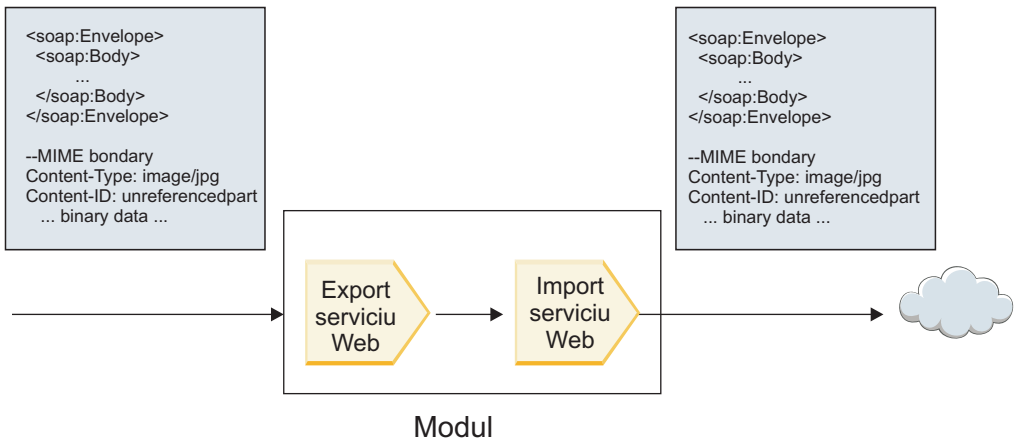


Figura 23. Un ataşament care trece printr-un modul SCA

În Figura 23, mesajul SOAP, cu ataşament, este transmis fără modificări.

De asemenea, puteţi modifica mesajul SOAP prin utilizarea unei componente de tip flux de mediere. De exemplu, aveţi posibilitatea să utilizaţi componenta de tip flux de mediere pentru a extrage datele din mesajul SOAP (în acest caz, date binare din corpul mesajului) şi pentru a crea un mesaj SOAP cu ataşamente. Datele sunt prelucrate ca parte componentă

a elementului ce conţine ataşamentele dintr-un SMO (service message object).

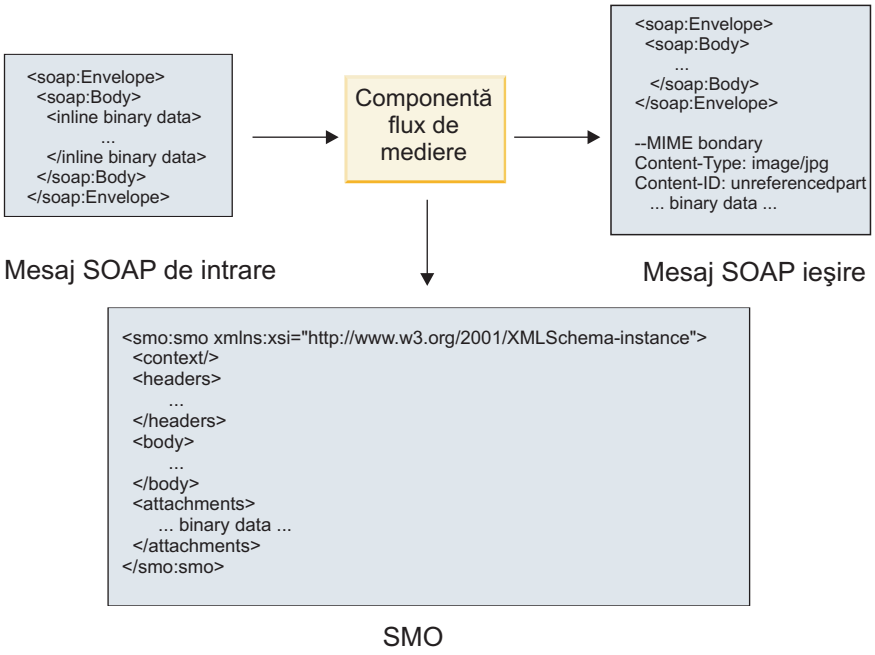


Figura 24. Un mesaj procesat de o componentă a fluxului de mediere

În schimb, componenta de fluxul de mediere se poate transforma mesajul de intrare prin extragerea şi codificarea ataşamentului, iar apoi transmite mesajul fără nici un ataşament.

În loc de extragerea datelor dintr-un mesaj SOAP de intrare pentru a forma un mesaj SOAP cu ataşamente, puteţi obţine datele ataşamentului la o sursă externă,⌂cum ar fi o bază de date.

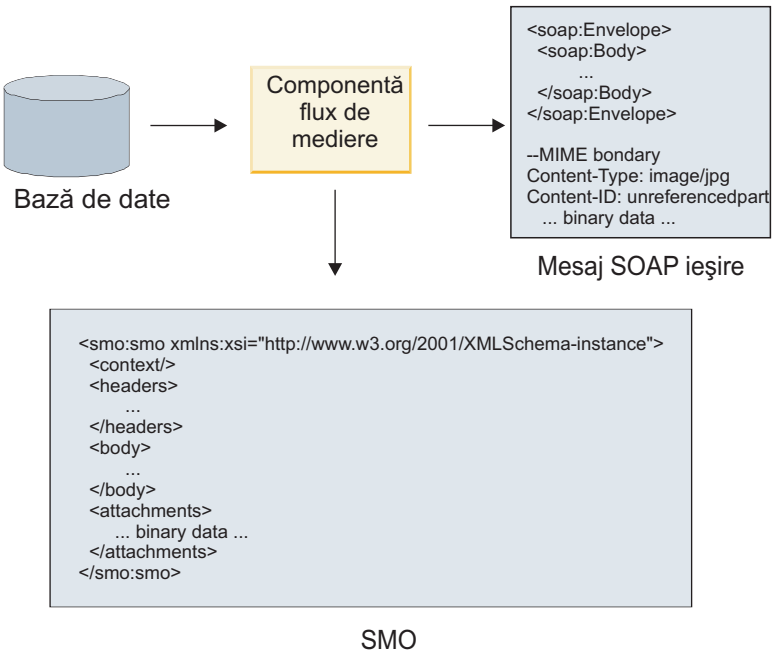


Figura 25. Un ataşament obţinut dintr-o bază de date şi adăugat la mesajul SOAP

În schimb, componenta fluxului de mediere poate extrage ataşamentul dintr-un mesaj SOAP de intrare şi poate procesa mesajul (de exemplu, păstrează ataşamentul într-o bază de date).

Ataşamentele fără referinţă pot fi transmise doar peste componentele fluxului de mediere. În cazul în care un ataşament trebuie să fie accesat sau transmis către o componentă de un alt tip, folosiţi o componentă a fluxului de mediere pentru a muta ataşamentul la o locaţie care este accesibilă acelei componente.

Important: Aşa cum este descris în "Reprezentare XML a SMO," primitivă de mediere Mapping transformă mesajele utilizând o transformare XSLT 1.0. Transformarea operează asupra unei serializări XML a SMO. Primitiva de mediere Mapping permite specificarea rădăcinii de serializare şi elementul rădăcină a documentului XML reflectă această rădăcină.

Atunci când trimiteţi mesaje SOAP cu ataşamente, elementul rădăcină pe care îl alegeţi determină modul în care ataşamentele sunt propagate.

- Dacă folosiţi "/body" drept rădăcină a mapării XML, toate ataşamentele sunt propagate implicit de-a lungul mapării.
- Dacă folosiţi "/" drept rădăcină pentru mapare, puteţi controla propagarea ataşamentelor.

Utilizarea legării stilului documentului WSDL cu mesaje multiple:

Organizaţia WS-I (Web Services Interoperability Organization) a definit un set de reguli legate de modul în care serviciile web ar trebui să fi descrise prin intermediul unui WSDL şi modul în care mesajele SOAP corespunzătoare ar trebui să fie formate, în scopul asigurării interoperabilităţii.

Aceste reguli sunt specificate în WS-I *Basic Profile Version 1.1* (http://www.ws-i.org/Profiles/BasicProfile-1.1.html). În special, WS-I Basic Profile 1.1 R2712 declară: "O legare literală de document TREBUIE să fie serializată ca PLIC cu un soap:Body al cărui element copil este o instanţă a declaraţiei elementului global referită de partea componentă wsdl:message corespunzătoare."

Aceasta înseamnă că, atunci când utilizaţi o legare SOAP de stil document pentru o operaţie cu mesaje (intrare, ieşire sau greşeală) care sunt definite cu părţi componente multiple, doar una dintre acele părţi ar trebui să fie legată de corpul SOAP pentru a fi compatibilă cu WS-I Basic Profile 1.1.

Suplimentar, WS-I Attachments Profile 1.0 R2941 declară: "O wsdl:binding dintr-o DESCRIERE AR TREBUI să lege fiecare wsdl:part a unui wsdl:message din wsdl:portType la care se referă la unul dintre soapbind:body, soapbind:header, soapbind:fault, soapbind:headerfault sau mime:content.".

Aceasta înseamnă că, atunci când utilizaţi o legare SOAP de stil document pentru o operaţie cu mesaje (intrare, ieşire sau greşeală) care sunt definite cu părţi componente multiple, toate părţile componente în afară de cea selectată care vor fi legate la corpul SOAP trebuie să fie legate ca ataşamente sau anteturi.

Următoarea abordare este utilizată atunci când descrierile WSDL sunt generate pentru exporturi cu legări de servicii web (JAX-WS şi JAX-RPC) în acest caz:

- Puteţi alege care parte a mesajului să fie legată la corpul SOAP dacă există mai multe elemente de tip non-binar. Dacă există un singur element de tip non-binar, acel element este legat automat la corpul SOAP.
- Pentru legarea JAX-WS, toate celelalte părţi componente ale mesajului de tip "hexBinary" sau "base64Binary" sunt legate ca ataşamente referite. Vedeţi "Ataşamente la care se face referire: părţi componente de nivel înalt ale mesajului" la pagina 83.
- Toate celelalte părţi componente ale mesajului sunt legate ca anteturi SOAP.

Legările de import JAX-RPC şi JAX-WS respectă legarea SOAP într-un document WSDL existent cu mesaje compuse de tip document chiar dacă nu leagă mai multe părţi la corpul SOAP; totuşi, nu sunteţi în măsură să generaţi clienţi pentru serviciile web pentru astfel de documente WSDL în Rational Application Developer.

Notă: Legarea JAX-RPC nu suportă ataşamente.

Tiparul recomandat când utilizaţi mesaje multiple cu o operaţie care are legare SOAP de stil document este prin urmare:

1. Utilizaţi stilul înfăşurat document/literal. În acest caz, mesajele au mereu o sngură parte componentă; totuşi, ataşamentele trebuie să fie nereferite (după cum este descris în “Ataşamente fără referinţă” la pagina 89) sau swaRef-typed (după cum este descris în “Ataşamente către care se face referinţă: ataşamente de tip swaRef” la pagina 79) în acest caz.
2. Utilizaţi stilul RPC/literal. În acest caz, nu există restricţii asupra legării WSDL în ceea ce priveşte numărul de părţi componente legate de corpul SOAP; mesajul SOAP care rezultă are întotdeauna un singur copil care reprezintă operaţia care este invocată, cu părţile componente ale mesajului fiind copiii acelui element.
3. Pentru legarea JAX-WS, trebuie să aveţi cel mult o parte componentă mesaj care nu este de tip "hexBinary" sau "base64Binary", doar dacă nu este acceptabil să legaţi celelalte părţi componente non-binare la anteturi SOAP.
4. Orice alte cazuri sunt supuse comportamentului descris.

Notă: Există restricţii suplimentare când utilizaţi mesaje SOAP care nu sunt compatibile cu WS-I *Basic Profile Version 1.1*.

- Prima parte componentă a mesajului ar trebui să fie non-binară.
- Când recepţionaţi mesaje SOAP stil document multiple cu ataşamente referite, legarea JAX-WS aşteaptă ca fiecare ataşament referit să fie reprezentat de un element copil al corpului SOAP cu o valoare atribut href care identifică ataşamentul după ID-ul conţinutului său. Legarea JAX-WS trimite ataşamente referite pentru astfel de mesaje în acelaşi mod. Acest comportament nu este compatibil cu WS-I Basic Profile.

  Pentru a vă asigura că mesajele dumneavoastră sunt compatibile cu Basic Profile, urmaţi abordarea 1 sau 2 din lista anterioară sau evitaţi utilizarea ataşamentelor referite pentru astfel de mesaje şi utilizaţi în schimb ataşamente nereferite sau de tip swaRef.

Legări HTTP:

Legarea HTTP este proiectată să furnizeze conectivitate SCA (Service Component) la HTTP. În consecinţă, aplicaţii HTTP existente sau nou-dezvoltate pot participa în medii SOA (Service Oriented Architecture).

HTTP (Hypertext Transfer Protocol) este un protocol utilizat pe scară largă pentru transferul de informaţiilor pe web. Când lucraţi cu o aplicaţie externă ce foloseşte protocolul HTTP, este necesară o legare HTTP. Legarea HTTP transformă datele transmise ca un mesaj într-un format nativ unui obiect operaţional într-o aplicaţie SCA. Legarea HTTP poate de asemenea transforma datele transmise în afară ca un obiect operaţional în formatul nativ aşteptat de aplicaţia externă.

Notă: Dacă doriţi să interacţionaţi cu clienţii şi serviciile care utilizează protocolul SOAP/HTTP pentru serviciile web, luaţi în considerare utilizarea uneia dintre legările serviciilor web, care oferă funcţionalităţi suplimentare cu privire la manipulare standardului calităţii serviciilor pentru serviciile web.

Câteva scenarii comune pentru utilizarea legării HTTP sunt descrise în următoarea listă:

- Serviciile găzduite de SCA pot invoca aplicaţii HTTP folosind un import HTTP.
- Serviciile găzduite de SCA se pot expune ca aplicaţii cu HTTP activat, pentru a putea fi folosite de clienţi HTTP, folosind un export HTTP.
- IBM Business Process Manager şi Process Server pot comunica între ei peste o infrastructură HTTP, în consecinţă utilizatorii îşi pot gestiona comunicările conform standardelor de corporaţie.
- IBM Business Process Manager şi Process Server pot acţiona ca mediatori de comunicaţii HTTP, transformând şi direcţionând mesaje, ceea ce îmbunătăţeşte integrarea aplicaţiilor folosind o reţea HTTP.
- IBM Business Process Manager şi Process Server pot fi folosite ca o punte între HTTP şi alte protocoale, precum servicii Web SOAP/HTTP, adaptoare de resurse bazate pe JCA (Java Connector Architecture), JMS, şi aşa mai departe.

Informaţii detaliate despre crearea legărilor de import şi export HTTP pot fi găsite în centrul de informare Integration Designer. Vedeţi subiectele **Dezvoltarea aplicaţiilor de integrare > Accesarea serviciilor externe cu HTTP>**.

Privire generală asupra legărilor HTTP:

Legarea HTTP oferă conectivitate la aplicaţii găzduite-HTTP. Mediază comunicarea dintre aplicaţii HTTP şi permite aplicaţiilor bazate pe HTTP existente să fie apelate dintr-un modul.

Legări de import HTTP

Legarea de import HTTP oferă conectivitate de ieşire de la aplicaţii SCA (Service Component Architecture) la un server sau aplicaţii HTTP.

Importul invocă un URL de punct final HTTP. URL-ul poate fi specificat în unul din trei moduri:

- URL-ul poate fi setat dinamic în anteturile HTTP prin URL-ul de înlocuire dinamic.
- URL-ul poate fi setat dinamic în elementul de adresă ţintă SMO.
- URL-ul poate fi specificat ca o proprietate de configurare pe import.

Această invocare este întotdeauna sincronă în natură.

Deşi invocările HTTP sunt întotdeauna cerere-răspuns, importul HTTP suportă şi operaţii cu sens unic şi cu sens dublu şi ignoră răspunsul în cazul unei operaţii cu sens unic.

Legări de export HTTP

Legarea de export HTTP oferă conectivitate de intrare de la aplicaţii HTTP la o aplicaţie SCA.

Un URL este definit pe exportul HTTP. Aplicaţiile HTTP ce vor să trimită mesaje cerere exportului folosesc acest URL pentru a invoca exportul.

Exportul HTTP de asemenea suportă ping-uri.

Legări HTTP la momentul rulării

Un import cu o legare HTTP în momentul rulării trimite o cerere cu sau fără date în corpul mesajului din aplicaţia SCA către serviciul web extern. Cererea se face din aplicaţia SCA către serviciul web extern, după cum se arată în Figura 26.

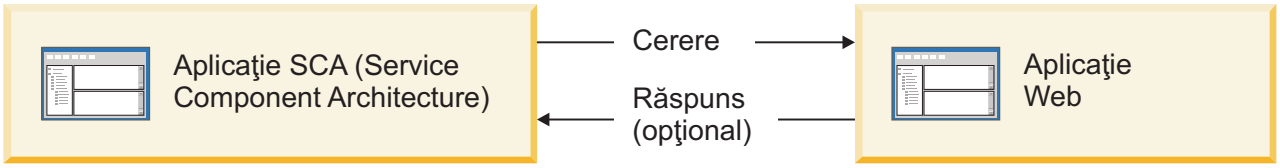


Figura 26. Fluxul unei cereri de la aplicaţia SCA către aplicaţia web

Op⌂ional, importul cu legarea HTTP poate primi date înapoi de la aplicaţia web într-un răspuns la cerere.

Cu un export, cererea se face de către o aplicaţie client către un serviciu web, după cum se arată în Figura 27.

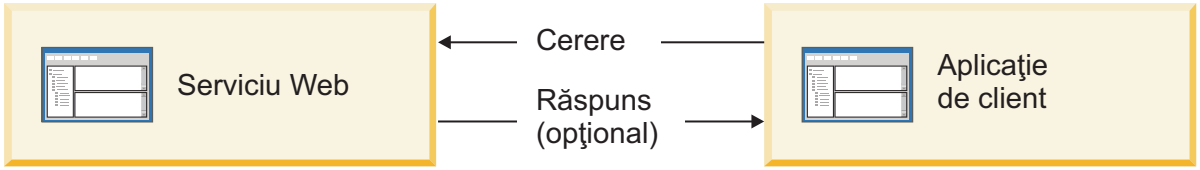


Figura 27. Fluxul unei cereri de la aplicaţia client la serviciul web.

Serviciul web este o aplicaţie web care rulează pe server. Exportul este implementat în aplicaţia web ca un servlet, astfel încât clientul trimite cererea către o adresă URL. Servletul transmite cererea aplicaţiei SCA în timpul rulării.

Opţional, exportul poate trimite date aplicaţiei client ca răspuns la cerere.

Anteturi HTTP:

Legările de import şi export HTTP permit ca configurarea anteturilor HTTP şi a valorilor lor să fie folosită pentru mesaje de ieşire. Importul HTTP foloseşte aceste anteturi pentru cereri şi exportul HTTP le foloseşte pentru răspunsuri.

Anteturile configurate static şi informaţiile de control prevalează asupra valorilor setate dinamic în timpul rulări. Totuşi, valorile de control ale URL-ului de înlocuire dinamic, Versiunii şi Metodei înlocuiesc valorile statice, ce sunt altfel considerate implicite.

Legarea suportă natura dinamică a URL-ului importului HTTP prin determinarea valorii URL-ului ţintă HTTP, Versiunii şi Metodei la momentul rulării. Aceste valori sunt determinate prin extragerea valorii Referinţei de punct final, URL-ului de înlocuire dinamic, Versiunii şi Metodei.

- Pentru Referinţa punctului final, folosiţi API-uri com.ibm.websphere.sca.addressing.EndpointReference APIs sau setaţi câmpul /headers/SMOHeader/Target/address în antetul SMO.
- Pentru URL de înlocuire dinamic, Versiune şi Metodă, folosiţi secţiunea parametrilor de control HTTP a mesajului SCA (Service Component Architecture). Reţineţi că URL-ul de înlocuire dinamic are precedenţă în faţa Referinţei de punct final ţintă; totuşi, Referinţa punctului final se aplică peste legări, deci este abordarea preferată şi ar trebui folosită unde este posibil.

Informaţiile de control şi antet pentru mesaje de ieşire sub legările de export şi import HTTP sunt procesate în următoarea ordine:

1. Informaţii de antet şi control excluzând URL de înlocuire dinamic HTTP, Versiune şi Metodă din Mesajul SCA (cea mai scăzută prioritate)
2. Modificări de la consola administrativă la nivelul de export/import
3. Modificări de la consola administrativă la nivelul metodei al exportului sau importului
4. Adresa ţintă specificată prin intermediul Referinţei punctului final sau antetului SMO
5. URL de înlocuire dinamic, Versiune şi Metodă din mesajul SCA
6. Informaţii de antet şi control de la handler-ul de date sau legarea de date (cea mai înaltă prioritate)

Exportul şi importul HTTP vor popula anteturile de direcţie de intrare şi parametri de control cu date din mesajul de intrare (HTTPExportRequest şi HTTPImportResponse) doar dacă propagarea antetului de protocol este setată la **Adevărat**. Invers, exportul şi importul HTTP vor citi şi procesa anteturile de ieşire şi parametri de controls (HTTPExportResponse şi HTTPImportRequest) doar dacă propagarea antetului de protocol este setată la **Adevărat**.

Notă: Modificările handler-ului de date sau legării de date asupra anteturilor sau parametrilor de control din răspunsul de import sau cererea de export nu vor altera instrucţiunile de procesare ale mesajului din interiorul legării de import sau export şi ar trebui folosite doar pentru propagarea valorilor modificate către componentele SCA din aval.

Serviciul de context este responsabil pentru propagarea contextului (inclusiv anteturile de protocol, precum antetul HTTP şi contextul de utilizator, precum ID cont) de-a lungul unei căi de invocare SCA. În timpul dezvoltării în IBM Integration Designer, puteţi controla propagarea contextului prin proprietăţile de import şi export. Pentru detalii suplimentare, vedeţi informaţiile legărilor de import şi export din centrul de informare al IBM Integration Designer.

Structuri furnizate de anteturi HTTP şi suport

Tabela 29 detaliază parametri de cerere/răspuns pentru cereri şi răspunsuri Import HTTP şi Export HTTP.

Tabela 29. Informaţii de antet HTTP furnizate

Nume control	Cerere Import HTTP	Răspuns Import HTTP	Cerere Export HTTP	Răspuns Export HTTP
URL	Ignorat	Nesetat	Citit din mesajul de cerere. **Notă:** Şirul de interogare face de asemenea parte din parametrul de control URL.	Ignorat
Versiune (valori posibile: 1.0, 1.1; cea implicită este 1.1)	Ignorat	Nesetat	Citit din mesajul de cerere	Ignorat
Metodă	Ignorat	Nesetat	Citit din mesajul de cerere	Ignorat
URL de înlocuire dinamic	Dacă este setat în handler-ul de date sau legarea de date, înlocuieşte URL-ul de import HTTP. Scris în mesaj în linia de cerere. **Notă:** Şirul de interogare face de asemenea parte din parametrul de control URL.	Nesetat	Nesetat	Ignorat
Versiune de înlocuire dinamică	Dacă este setat, înlocuieşte Versiunea de import HTTP. Scris în mesaj în linia de cerere.	Nesetat	Nesetat	Ignorat
Metodă de înlocuire dinamică	Dacă este setat, înlocuieşte Metoda de import HTTP. Scris în mesaj în linia de cerere.	Nesetat	Nesetat	Ignorat
Tip media (Acest parametru de control transmite parte din valoarea antetului HTTP Tip-conţinut.)	Dacă este prezent, este scris în mesaj ca parte din antetul Tip-conţinut. **Notă:** Această valoare a elementului de control ar trebui furnizată de handler-ul de date sau legarea de date.	Citit din mesajul răspuns, antetul Tip-conţinut	Citit din mesajul cerere, antetul Tip-conţinut	Dacă este prezent, scris în mesaj ca parte din antetul Tip-conţinut. **Notă:** Această valoare a elementului de control ar trebui furnizată de handler-ul de date sau legarea de date.
Set de caractere (implicit: UTF-8)	Dacă este prezent, scris în mesaj ca parte din antetul Tip-conţinut. **Notă:** Această valoare a elementului de control ar trebui furnizată de legarea de date.	Citit din mesajul răspuns, antetul Tip-conţinut	Citit din mesajul cerere, antetul Tip-conţinut	Suportat; scris în mesaj ca parte din antetul Tip-conţinut. **Notă:** Această valoare a elementului de control ar trebui furnizată de legarea de date.

Tabela 29. Informaţii de antet HTTP furnizate (continuare)

Nume control	Cerere Import HTTP	Răspuns Import HTTP	Cerere Export HTTP	Răspuns Export HTTP
Codare de transfer (Valori posibile: chunked, identity; implicit este identity)	Dacă este prezent, scris în mesaj ca un antet şi controlează cum este codată transformarea mesajului.	Citit din mesajul răspuns	Citit din mesajul de cerere	Dacă este prezent, scris în mesaj ca un antet şi controlează cum este codată transformarea mesajului.
Codare de conţinut (Valori posibile: gzip, x-gzip, deflate, identity; implicit este identity)	Dacă este prezent, scris în mesaj ca un antet şi controlează cum sunt codate datele utile (payload).	Citit din mesajul răspuns	Citit din mesajul de cerere	Dacă este prezent, scris în mesaj ca un antet şi controlează cum sunt codate datele utile (payload).
Lungime-Conţinut	Ignorat	Citit din mesajul răspuns	Citit din mesajul de cerere	Ignorat
StatusCode (implicit: 200)	Nesuportat	Citit din mesajul răspuns	Nesuportat	Dacă este prezent, scris în mesaj în linia de răspuns
ReasonPhrase (implicit: OK)	Nesuportat	Citit din mesajul răspuns	Nesuportat	Valoare de control ignorată. Valoarea liniei răspunsului mesajului este generată din StatusCode.
Autentificare (conţine proprietăţi multiple)	Dacă este prezent, folosit pentru a construi antetul Autentificare de bază. **Notă:** Valoarea pentru acest antet va fi codată doar pe protocolul HTTP. În SCA, va fi decodată şi transmisă ca text în clar.	Nu se aplică	Citit din antetul Autentificare de bază al mesajului răspuns. Prezenţa acestui antet nu indică faptul că utilizatorul a fost autentificat. Autentificarea ar trebui controlată în configurarea servletului. **Notă:** Valoarea pentru acest antet va fi codată doar pe protocolul HTTP. În SCA, va fi decodată şi transmisă ca text în clar.	Nu se aplică
Proxy (conţine proprietăţi multiple: Gazdă, Port, Autentificare)	Dacă este prezent, este folosit pentru a stabili conexiunea prin proxy.	Nu se aplică	Nu se aplică	Nu se aplică
SSL (conţine proprietăţi multiple: Keystore, Keystore Password, Trustore, Trustore Password, ClientAuth)	Dacă este populat şi url-ul destinaţie este HTTPS, este folosit pentru a stabili o conexiune prin SSL.	Nu se aplică	Nu se aplică	Nu se aplică

Legări de date HTTP:

Pentru fiecare mapare de date între un mesaj SCA (Service Component Architecture) şi un mesaj de protocol HTTP, trebuie configurate un handler de date sau o legare de date HTTP. Handler-ele de date oferă o interfaţă neutră-legării ce permite reutilizarea peste legări de transport şi reprezintă abordarea recomandată; legările de date sunt specifice unei anumite legări de transport. Sunt furnizate clase de legări de date specifice-HTTP; puteţi de asemenea scrie handler-e de date sau legări de date personalizate.

Notă: Cele trei clase de legări de date HTTP descrise în acest subiect (HTTPStreamDataBindingSOAP, HTTPStreamDataBindingXML şi HTTPServiceGatewayDataBinding) sunt depreciate începând cu IBM Business Process Manager Versiunea 7.0. În loc de a folosi legările de date descrise în acest subiect, consideraţi următoarele handler-e de date:

- Folosiţi SOAPDataHandler în loc de HTTPStreamDataBindingSOAP.
- Folosiţi UTF8XMLDataHandler în loc de HTTPStreamDataBindingXML
- Folosiţi GatewayTextDataHandler în loc de HTTPServiceGatewayDataBinding

Legările de date sunt furnizate pentru utilizare cu importuri HTTP şi exporturi HTTP: legare de date binare, legare de date XML şi legare de date SOAP. O legare de date răspuns nu este necesară pentru operaţii cu sens unic. O legare de date este reprezentată de numele unei clase Java ale cărei instanţe pot converti atât de la HTTP la ServiceDataObject şi vice-versa. Un selector de funcţii trebuie folosit pe un export care, în conjuncţie cu legări de metode, poate determina ce legare de date este folosită şi ce operaţie este invocată. Legările de date livrate sunt:

- Legări de date binare, ce tratează corpul ca date binare nestructurate. Implementarea schemei XSD a legării de date binare este după cum urmează:

```
<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>
```

- Legările de date XML, care suportă corpul ca date XML. Implementarea legării de date XML este similară cu legarea de date XML JMS şi nu are restricţii pe schema interfeţei.
- Legări de date SOAP, ce suportă corpul ca date SOAP. Implementarea legării de date SOAP nu are restricţii pe schema interfeţei.

Implementarea legărilor de date HTTP personalizate

Această secţiune descrie cum se implementează o legare de date HTTP personalizată.

Notă: Abordarea recomandată este să implementaţi un handler de date personalizat deoarece poate fi reutilizat peste legări de transport.

HTTPStreamDataBinding este principala interfaţă pentru tratarea mesajelor HTTP personalizate. Interfaţa este proiectată să permită tratarea datelor utile mari. Totuşi, pentru ca astfel de implementări să funcţioneze, această legare de date trebuie să returneze informaţiile de control şi anteturile înainte de a scrie mesajul în flux.

Metodele şi ordinea lor de execuţie, listate mai jos, trebuie implementate de legarea de date personalizată.

Pentru a personaliza o legare de date, scrieţi o clasă ce implementează HTTPStreamDataBinding. Legarea de date ar trebui să aibă patru proprietăţi private:

- private DataObject pDataObject
- private HTTPControl pCtrl
- private HTTPHeaders pHeaders

- private yourNativeDataType nativeData

Legarea HTTP va invoca legarea de date personalizată în următoarea ordine:

- Procesare de ieşire (DataObject în format Nativ):
  1. setDataObject(...)
  2. setHeaders(...)
  3. setControlParameters(...)
  4. setBusinessException(...)
  5. convertToNativeData()
  6. getControlParameters()
  7. getHeaders()
  8. write(...)
- Procesare de intrare (format Nativ în DataObject):
  1. setControlParameters(...)
  2. setHeaders(...)
  3. convertFromNativeData(...)
  4. isBusinessException()
  5. getDataObject()
  6. getControlParameters()
  7. getHeaders()

Trebuie să invocaţi setDataObject(...) în convertFromNativeData(...) pentru a seta valoarea lui dataObject, ce este convertit din date native în proprietatea privată "pDataObject".

```
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;
}
/*
* Adăugaţi antetul http "IsBusinessException" în pHeaders.
* Doi paşi:
* 1.Înlăturaţi toate anteturile cu numele IsBusinessException (nesensibil la majuscule) întâi.
*   Aceasta este pentru a vă asigura că doar un singur antet este prezent.
* 2.Adăugaţi noul antet "IsBusinessException"
*/
public void setBusinessException(boolean isBusinessException) {
  //înlăturaţi toate anteturile cu numele IsBusinessException (nesensibil la majuscule) întâi.
  //Aceasta este pentru a vă asigura că doar un singur antet este prezent.
  //adăugaţi noul antet "IsBusinessException", exemplu de cod:
  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;
}
/*
* Obţineţi antetul "IsBusinessException" din pHeaders, returnaţi-i valoarea booleană
*/
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){
  //Metodă dezvoltată de client pentru a
  //Citi date din HTTPInputStream
  //Convertirea lor la DataObject
  DataObject dataobject=realConvertWorkFromNativeDataToSDO(arg0);
  setDataObject(dataobject);
}
public void write(HTTPOutputStream output) throws IOException {
  if (nativeData != null)
  output.write(nativeData);
}
```

Legări EJB:

Legările de import EJB (Enterprise JavaBeans) permit componentelor SCA (Service Component Architecture) să invoce serviciile furnizate de o logică operaţională Java EE ce rulează pe un server Java EE. Legările de export EJB permit componentelor SCA să fie afişate sub formă de Enterprise JavaBeans, astfel încât logica operaţională Java EE să poată invoca componentele SCA, care altfel sunt indisponibile pentru ele.

Legările de import EJB:

Legările de import EJB permit unui modul SCA să apeleze implementările EJB prin specificarea modului în care modulul de consum este legat de EJB-ul extern. Importul serviciilor dintr-o implementare EJB externă permite utilizatorilor să îşi conecteze logica operaţională în mediul IBM Business Process Manager şi să participe într-un proces operaţional.

Folosiţi Integration Designer pentru a crea legări de import EJB. Puteţi utiliza oricare dintre următoarele proceduri pentru a genera legările:

- Creare import EJB folosind vrăjitorul de servicii externe

  Puteţi folosi vrăjitorul pentru servicii externe în Integration Designer pentru a construi un import EJB bazat pe implementarea existentă. Vrăjitorul pentru servicii externe creează servicii pe criteriile furnizate de dumneavoastră. Apoi generează obiecte business, interfeţe şi importă fişiere în baza serviciilor descoperite.
- Creare legări de import EJB folosind editorul de asamblare

  Puteţi să creaţi un import EJB într-o diagramă de asamblare utilizând editorul de asamblare Integration Designer. Din paletă, puteţi folosi fie un Import, fie o clasă Java pentru a crea o legare EJB.

Importul generat are legări de date care fac conexiunea Java-WSDL în loc de a cere o componentă punte Java. Aveţi posibilitatea să legaţi în mod direct o componentă cu o referinţă WSDL (Web Services Description Language) la importul EJB care comunică cu un serviciu bazat pe EJB folosind o interfaţa Java.

Importul EJB poate interacţiona cu logica operaţională Java EE folosind fie modelul de programare EJB 2.1, fie modelul de programare EJB 3.0.

Invocarea logicii operaţionale Java EE poate fi locală (doar pentru EJB 3.0) sau de la distanţă.

- Invocarea locală este utilizată atunci când doriţi să apelaţi logica operaţională Java EE care se află pe acelaşi server ca şi importul.

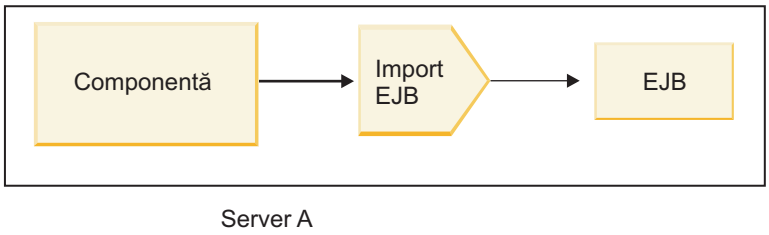


Figura 28. Invocarea locală a unui EJB (doar EJB 3.0)

- Invocarea de la distanţă este utilizată atunci când doriţi să apelaţi logica operaţională Java EE care nu se află pe acelaşi server ca şi importul.

  De exemplu, în figura de mai jos, un import EJB utilizează RMI/IIOP (Remote Method Invocation over Internet InterORB Protocol) pentru a invoca o metodă EJB de pe un alt server.

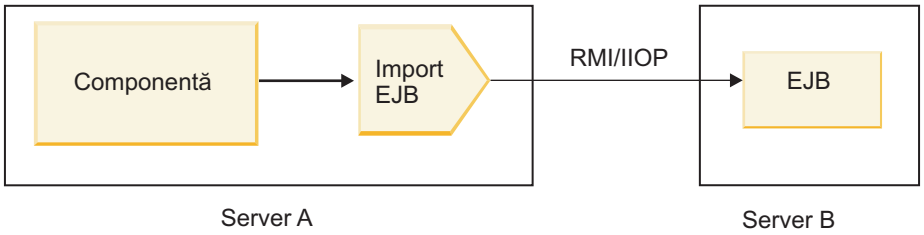


Figura 29. Invocare de la distanţă a unui EJB

Atunci când configurează legarea EJB, Integration Designer foloseşte numele JNDI pentru a determina nivelul modelului de programare EJB şi tipul de invocare (local sau de la distanţă).

Legările de export EJB conţin următoarele componente majore:
- Handler date JAX-WS
- Selector de defectare EJB
- Selector pentru funcţia de import EJB

Dacă scenariul dvs. de utilizator nu se bazează pe maparea JAX-WS, s-ar putea să aveţi nevoie de un handler de date personalizat, un selectorul de funcţie şi de un selectorul de defect pentru a realiza taskurile care altfel ar fi fost finalizate de către componentele care fac parte din legările de import EJB. Acest lucru include maparea care ar fi finalizată în mod normal de către algoritmul personalizat de mapare.

Legările de export EJB:

Aplicaţiile Java EE externe pot invoca o componentă SCA prin intermediul unei legări de export EJB. Folosind un export EJB puteţi expune componente SCA, astfel încât aplicaţiile Java EE externe pot invoca aceste componente folosind modelul de programare EJB.

Notă: Exportul EJB este un bean stateless.

Folosiţi Integration Designer pentru a crea legări EJB. Puteţi utiliza oricare dintre următoarele proceduri pentru a genera legările:
- Crearea legărilor de export EJB folosind vrăjitorul pentru servicii externe

Puteţi folosi vrăjitorul pentru servicii externe în Integration Designer pentru a construi un serviciu de export EJB bazat pe implementarea existentă. Vrăjitorul pentru servicii externe creează servicii pe criteriile furnizate de dumneavoastră. Apoi acesta generează obiecte business, interfeţe şi exportă fişiere pe baza serviciilor descoperite.

- Crearea legărilor de export EJB folosind editorul de asamblare

  Puteţi să creaţi un export EJB folosind editorul de asamblare Integration Designer.

Important: Un client Java 2 Platform, Standard Edition (J2SE) nu poate invoca clientul de export EJB care este generat în Integration Designer.

Puteţi genera legarea dintr-o componentă SCA existentă, sau puteţi genera un export cu o legare EJB pentru o interfaţă Java.

- Atunci când generaţi un export dintr-o componentă SCA existentă care are o interfaţă WSDL existentă, exportului i se alocă o interfaţă Java.
- Atunci când generaţi un export dintr-o interfaţă Java, puteţi selecta fie un WSDL, fie o interfaţă Java pentru export.

  Notă: O interfaţă Java folosită pentru a crea un export EJB are următoarele limitări în ceea ce priveşte obiectele (parametrii de ieşire şi intrare şi excepţii) transmise ca parametrii la un apel de la distanţă:
  - Trebuie să fie de un tip concret (în loc de un tip interfaţă sau abstract).
  - Acestea trebuie să fie în concordanţă cu specificaţia Enterprise JavaBeans. Trebuie să fie serializabili şi să aibă un constructor implicit fără argumente, iar toate proprietăţile trebuie să fie accesibile prin intermediul metodelor getter si setter.

    Consultaţi site-ul web Sun Microsystems, Inc., http://java.sun.com pentru informaţii legate de specificaţiile JavaBeans Enterprise.

  În plus, excepţia trebuie să fie o excepţie verificată, moştenită din java.lang.Exception, şi trebuie să fie unică (adică, nu suportă să arunce mai multe tipuri de excepţii verificate).

  De asemenea, reţineţi că interfaţa afacerii pentru un EnterpriseBean Java este o interfaţă Java simplă şi nu trebuie să extindă javax.ejb.EJBObject sau javax.ejb.EJBLocalObject. Metodele interfeţei de afaceri nu ar trebui să arunce java.rmi.Remote.Exception.

Legările de export EJB pot interacţiona cu logica operaţională Java EE fie folosind modelul de programare EJB 2.1, fie modelul de programare EJB 3.0.

Invocarea poate fi locală (doar pentru EJB 3.0) sau de la distanţă.

- Invocarea locală este utilizată atunci când logica operaţională Java EE apelează o componentă SCA care se află pe acelaşi server ca şi exportul.
- Invocarea de la distanţă este utilizată atunci când logica operaţională the Java EE business logic nu se află pe acelaşi server ca şi exportul.

  De exemplu, în figura următoare, un EJB foloseşte RMI/IIOP pentru a apela o componentă SCA aflată pe un server diferit.

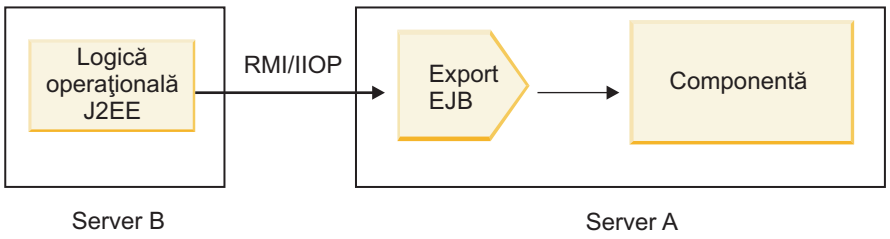


Figura 30. Un apel la distanţă de la un client către o componentă SCA prin intermediul unui export EJB

Atunci când configurează legarea EJB, Integration Designer foloseşte numele JNDI pentru a determina nivelul modelului de programare EJB şi tipul de invocare (local sau de la distanţă).

Legările de export EJB conţin următoarele componente majore:

- Handler date JAX-WS
- Selector pentru funcţia de export EJB

Dacă scenariul dvs. de utilizator nu se bazează pe maparea JAX-WS, s-ar putea să aveţi nevoie de un handler de date personalizat şi de un selectorul de funcţie pentru a realiza taskurile care altfel ar fi finalizate de către componentele care fac parte din legările export EJB. Acest lucru include maparea care ar fi finalizată în mod normal de către algoritmul personalizat de mapare.

Proprietăţile legărilor EJB:

Legările pentru import EJB îşi folosesc numele JNDI configurate pentru a determina nivelul modelului de programare EJB şi tipul de invocare (local sau de la distanţă). Legările de import şi export EJB utilizează handler-ul de date JAX-WS pentru transformarea datelor. Legarea de import EJB foloseşte un selector pentru funcţia de import EJB şi un selector pentru defectul EJB, iar legarea de export EJB foloseşte un selector pentru funcţia de export EJB.

Numele JNDI şi legările de import EJB:

Atunci când configurează legarea EJB pentru un import, Integration Designer foloseşte numele JNDI pentru a determina nivelul modelului de programare EJB şi tipul de invocare (local sau de la distanţă).

În cazul în care nu este specificat nici un nume JNDI, se foloseşte legarea interfeţei EJB implicite. Numele implicite care sunt create depind de faptul că invocaţi EJB 2.1 JavaBeans sau EJB 3.0 JavaBeans.

Notă: Consultaţi subiectul "Privire generală asupra legărilor de aplicaţie EJB 3.0" din Centrul de informare WebSphere Application Server pentru informaţii suplimentare detaliate despre convenţiile de numire.

- EJB 2.1 JavaBeans

  Numele JNDI implicit preselectat de către Integration Designer este legarea EJB 2.1 implicită, care ia formularul **ejb/** plus interfaţa home, separate prin linii oblice.

  De exemplu, pentru interfaţa home a EJB 2.1 JavaBeans pentru com.mycompany.myremotebusinesshome, legarea implicită este:

  ```
  ejb/com/mycompany/myremotebusinesshome
  ```

  Pentr EJB 2.1, este suportată doar invocarea EJB de la distanţă.

- EJB 3.0 JavaBeans

  Numele JNDI implicit preselectat de Integration Designer pentru JNDI-ul local este numele clasei complet calificate al interfeţei locale precedate de **ejblocal:**. De exemplu, pentru interfaţa complet calificată a interfeţei com.mycompany.mylocalbusiness locale, EJB-ul 3.0 JNDI preselectat este:

  ```
  ejblocal:com.mycompany.mylocalbusiness
  ```

  Pentru interfaţa com.mycompany.myremotebusiness de la distanţă, EJB-ul 3.0 JNDI preselectat este interfaţa complet calificată:

  ```
  com.mycompany.myremotebusiness
  ```

  Legările aplicaţiei implicite EJB 3.0 sunt descrise la următoarea locaţie: Privire generală asupra legărilor de aplicaţie EJB 3.0.

  Integration Designer va folosi numele "scurt" drept locaţia JNDI implicită pentru EJB-uri folosind versiunea 3.0 a modelului de programare.

  Notă: În cazul în care referinţa JNDI implementată a ţintei EJB este diferită de locaţia legării JNDI implicite deoarece a fost utilizată sau configurată o mapare personalizată, numele JNDI-ului ţintă trebuie să fie specificat în

mod corespunzător. Aveţi posibilitatea să specificaţi numele în Integration Designer înainte de implementare, sau , pentru legarea de import, puteţi modifica numele în consola administrativă (după implementare) pentru a potrivi numele JNDI pentru EJB ţintă.

Pentru informaţii suplimentare despre crearea legărilor EJB, vedeţi secţiunea dedicată pentru Lucrul cu legări EJB din Centrul de informare Integration Designer.

Handler date JAX-WS:

Legarea EJB (Enterprise JavaBeans) de import utilizează handler-ul de date JAX-WS pentru a transforma obiecte business cerere în parametri de obiect Java şi pentru a transforma valoarea returnată a obiectului Java în obiecte business răspuns. Legarea EJB de import foloseşte handler-ul de date JAX-WS pentru a transforma EJB-urile cerere în obiecte business cerere şi pentru a transforma obiectul business răspuns într-o valoare returnată.

Acest handler de date mapează datele de la interfaţa WSDL specificată în SCA cu interfaţa Java EJB ţintă (şi vice versa) folosind specificaţiile pentru JAX-WS (Java API for XML Web Services) şi specificaţiile pentru JAXB (Java Architecture for XML Binding).

Notă: Suportul actual este limitat la specificaţiile pentru JAX-WS 2.1.1 şi JAXB 2.1.3.

Handler-ul de date specificat la nivelul legării EJB este folosit pentru a realiza procesarea cererilor, răspunsurilor, defectelor şi excepţiilor apărute în timpul rulării.

Notă: Pentru defecte, poate fi specificat un anumit handler de date pentru fiecare dintre ele prin specificarea proprietăţii de configurare faultBindingType. Acesta înlocuieşte valoarea specificată la nivelul legării EJB.

Handler-ul de date JAX-WS este folosit în mod implicit atunci când legarea EJB are o interfaţă WSDL. Acest handler-ul de date nu poate fi folosit pentru a transforma un mesaj SOAP care reprezintă o invocare JAX-WS către un obiect de date.

Legarea de import EJB foloseşte un handler de date pentru a transforma un obiect de date într-o matrice cu elemente de tip Object Java (Object[]). În timpul comunicaţiilor ce au loc la ieşire în, are loc următoarea procesare:

1. Legarea EJB setează tipul aşteptat, elementul aşteptat şi numele metodei ţintă în BindingContext pentru a se potrivi cu cele specificate în WSDL.
2. Legarea EJB invocă metoda de transformare pentru obiectul de date care are nevoie de transformarea datelor.
3. Handler-ul de date returnează un Object[] care în care sunt reprezentaţi parametrii metodei (în ordinea definirii lor în cadrul metodei).
4. Legarea EJB foloseşte Object[] pentru a invoca metoda în interfaţa EJB ţintă.

Legarea pregăteşte de asemenea un Object[] pentru a procesa răspunsul din invocarea EJB.

- Primul element din Object[] este valoarea returnată din invocarea metodei Java.
- Valorile următoare reprezintă parametrii de intrare pentru metodă.

Aceasta este necesară pentru a suporta parametrii de tip In/Out şi Out.

Pentru parametrii de tip Out, valorile trebuie să fie returnate în obiectul datelor de răspuns.

Handler-ul de date procesează şi transformă valorile găsite în Object[], iar apoi returnează un răspuns către obiectul de date.

Handler-ul de date suportă xs:AnyType, xs:AnySimpleType şi xs:Any împreună cu alte tipuri de date XSD. Pentru a activa suportul pentru xs:Any, utilizaţi **@XmlAnyElement (lax=true)** pentru proprietatea JavaBeans în codul Java aşa cum este afişat în exemplul următor:

```
public class TestType {
 private Object[] object;

 @XmlAnyElement (lax=true)
 public Object[] getObject() {
  return object;
 )

 public void setObject (Object[] object) {
  this.object=object;
 )
)
```

Acest lucru face obiectul proprietate din TestType un câmp xs:any. Valoarea clasei Java folosite în câmpul xs:any ar trebui să aibă adnotarea **@XmlAnyElement**. De exemplu, dacă clasa Java folosită pentru a popula obiectul de tip matrice este Address, atunci această clasă ar trebui să aibă adnotarea **@XmlRootElement**.

Notă: Pentru a personaliza maparea de la tipul XSD la tipurile Java definite prin specificaţia JAX-WS, modificaţi adnotările JAXB, astfel încât să se potrivească nevoilor afacerii dvs. Handler-ul de date JAX-WS suportă xs:any, xs:anyType şi xs:anySimpleType.

Pentru handler-ul de date JAX-WS se aplică următoarele restricţii:

- Handler-ul de date nu include suport pentru adnotarea **@WebParam** din atributul antetului.
- Spaţiul de nume pentru fişierele schemei obiectului business (fişiere XSD) nu include maparea implicită din numele pachetului Java. Adnotarea **@XMLSchema** din package-info.java, de asemenea, nu funcţionează. Singura modalitate de a crea un XSD cu un spaţiu de nume este de a utiliza adnotările **@XmlType** şi **@XmlRootElement**. **@XmlRootElement** defineşte spaţiul nume ţintă pentru elementul global în tipuri JavaBeans.
- Vrăjitorul pentru importul EJB nu creează fişiere XSD pentru clasele fără legătură. Versiunea 2.0 nu suportă adnotare **@XmlSeeAlso**, aşa că, dacă nu se face referire în mod direct de la clasa părinte către clasa copil, nu se creează un XSD. Soluţia la această problemă este să rulaţi SchemaGen pentru astfel de clase copil.

  SchemaGen este un utilitar al liniei de comandă (localizat în directorul *WPS_Install_Home*/bin) furnizat pentru a crea fişiere XSD pentru un bean dat. Aceste XSD-uri trebuie să fie copiate manual în modulul pentru ca soluţia să funcţioneze.

Selector defect EJB:

Selectorul defectului EJB determină dacă o invocare EJB a rezultat într-un defect, o excepţie de runtime sau într-un răspuns cu succes.

În cazul în care este detectat un defect, selectorul de defect EJB returnează numele defectului nativ către runtime-ul legării, astfel încât handler-ul datelor JAX-WS pot converti obiectul de tip excepţie într-un obiect business defect.

Într-un răspuns cu succes (fără defect), legarea de import EJB asamblează un obiect Java de tip matrice (Object[]) pentru a returna valorile.

- Primul element din Object[] este valoarea returnată din invocarea metodei Java.
- Valorile următoare reprezintă parametrii de intrare pentru metodă.

Aceasta este necesar pentru a suporta parametrii de tip In/Out şi Out.

Pentru scenariile de excepţie, legarea asamblează un Object[], iar primul element reprezintă excepţia aruncată de metodă.

Selectorul de defect poate returna oricare dintre următoarele valori:

Tabela 30. Valori returnate

Tip	Valoare returnată	Descriere
Defect	**ResponseType.FAULT**	Returnată atunci când Object[] transmis conţine un obiect de tip excepţie.
Excepţie runtime	**ResponseType.RUNTIME**	Returnată în cazul în care obiectul de tip excepţie nu se potriveşte cu nici unul dintre tipurile de excepţii declarate în cadrul metodei.
Răspuns normal	**ResponseType.RESPONSE**	Returnată în toate celelalte cazuri.

În cazul în care selectorul de defect returnează o valoare **ResponseType.FAULT**, atunci este returnat numele defectului nativ. Acest nume nativ de defect este folosit de legare pentru a determina numele defectului WSDL corespunzător din model şi pentru a invoca handler-ul corect pentru datele defecte.

Selector funcţie EJB:

Legările EJB folosesc un selector pentru funcţia de import (pentru procesarea la ieşire) sau un selector pentru funcţia de export (pentru procesarea la intrare) pentru a determina ce metodă EJB să apeleze.

Selector pentru funcţia de import

Pentru procesarea la ieşire, selectorul pentru funcţia de import derivă tipul metodei EJB în funcţie de numele operaţiei invocate de componenta SCA care este legată de importul EJB. Selectorul funcţiei caută adnotarea @WebMethod în clasa Java adnotată JAX-WS generată de Integration Designer pentru a determina numele operaţiei ţintă asociate.

- În cazul în care adnotarea @WebMethod există, selectorul funcţiei foloseşte această adnotare pentru a determina maparea corectă a metodei Java pentru metoda WSDL.
- În cazul în care adnotarea @WebMethod lipseşte, selectorul funcţiei presupune că numele metodei Java este acelaşi cu cel al operaţiei invocate.

Notă: Acest selector de funcţie este valid doar pentru o interfaţă de tip WSDL într-un import EJB, şi nu pentru o interfaţă de tip Java dintr-un import EJB.

Selectorul funcţiei returnează un obiect java.lang.reflect.Method care reprezintă metoda interfeţei EJB.

Selectorul funcţiei foloseşte un Obiect Java de tip matrice (Object[]) pentru a reţine răspunsul de la metoda ţintă. Primul element din Object[] este o metodă Java care are numele WSDL, iar al doilea element din Object[] este obiectul business de intrare.

Selector pentru funcţia de export

Pentru procesarea la intrare, selectorul pentru funcţia de export derivă metoda ţintă, astfel încât să fie invocată din metoda Java.

Selectorul funcţiei de export mapează numele operaţiei Java invocate de clientul EJB în numele operaţiei din interfaţa componentei ţintă. Numele metodei este returnat sub forma unui şir de caractere şi este rezolvat de runtime-ul SCA în funcţie de tipul interfeţei al componentei ţintă.

Legări EIS:

Legările EIS (Enterprise information system) asigură conectivitatea între componente SCA şi un EIS extern. Această comunicaţie este realizată folosind exporturile EIS şi importurile EIS care suportă adaptoarele de resurse JCA 1.5 şi WebSphere Adapters.

Componentele dumneavoastră SCA ar putea impune ca datele să fie transferate către sau de la un EIS extern. Atunci când creaţi un modul SCA care necesită o astfel de conectivitate, veţi include (în plus faţă de componenta dvs. SCA) un import sau un export cu o legare EIS pentru comunicaţia cu un anumit EIS extern.

Adaptoarele de resurse din IBM Integration Designer sunt utilizate în contextul unui import sau export. Dezvoltaţi un import sau un export cu ajutorul vrăjitorului de servicii externe, iar în timpul dezvoltării includeţi adaptorul de resurse. Un import EIS care permite aplicaţiei dumneavoastră să invoce un serviciu într-un sistem EIS sau un export EIS care permite unei aplicaţii dintr-un sistem EIS să invoce un serviciu dezvoltat în IBM Integration Designer sunt create cu un adaptor de resurse. De exemplu, veţi crea un import cu adaptorul JD Edwards pentru a invoca un serviciu în sistemul JD Edwards.

Atunci când utilizaţi vrăjitorul pentru servicii externe, informaţiile legate de legarea EIS sunt create pentru dvs. De asemenea, puteţi utiliza o altă unealtă, editorul de asamblare, pentru a adăuga sau modifica informaţiile legate de legare. Vedeţi Accesarea serviciilor externe cu adaptoare pentru informaţii suplimentare.

După ce modulul care conţine legarea EIS este implementat pe server, puteţi utiliza consola administrativă pentru a vizualiza informaţii despre legare sau pentru a configura legarea.

Privire generală asupra legărilor EIS:

Legarea EIS (enterprise information system), când este folosită cu un adaptor de resurse JCA, vă lasă să accesaţi servicii de pe un sistem de informaţii de întreprindere sau să vă faceţi serviciile disponibile EIS-ului.

Următorul exemplu arată cum un modul SCA numit ContactSyncModule sincronizează informaţii de contact între un sistem Siebel şi un sistem SAP.

1. Componenta SCA numită ContactSync ascultă (prin intermediul unui export de aplicaţie EIS numit Contact Siebel) modificări asupra contactelor Siebel.
2. Însăşi componenta SCA ContactSync foloseşte o aplicaţie SAP (printr-un import de aplicaţie EIS) pentru a actualiza informaţiile de contact SAP corespunzător.

Deoarece structurile de date folosite pentru memorarea contactelor sunt diferite în sistemele Siebel şi SAP, componenta SCA ContactSync trebuie să ofere mapare.

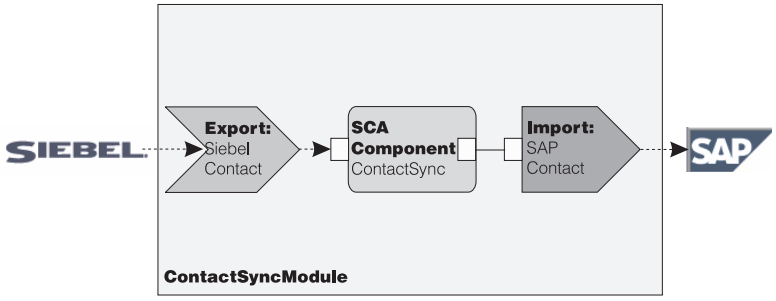


Figura 31. Flux de la un sistem Siebel la un sistem SAP

Exportul Contact Siebel şi importul Contact SAP au configurate adaptoarele de resurse corespunzătoare.

Caracteristici cheie ale legărilor EIS:

Un import EIS este un import SCA ce permite componentelor din modulul SCA să utilizeze aplicaţii EIS definite în afara modulului. Un import EIS este utilizat pentru a transfera date de la componenta SCA la un EIS extern; un export EIS este utilizat pentru a transfera date de la un EIS extern la modulul SCA.

Importuri

Rolul importului EIS este de a umple golul dintre componentele SCA şi sistemele EIS externe. Aplicaţiile externe pot fi tratate ca un import EIS. În acest caz, importul EIS trimite date către EIS extern şi primeşte opţional date în răspuns.

Importul EIS furnizează componentelor SCA cu o vizualizare uniformă a aplicaţilor externe modulului. Acest lucru permite componentelor să comunice cu un EIS extern, cum ar fi SAP, Siebel, sau PeopleSoft, utilizând un model consistent SCA.

Pe partea clientului al importului, există o interfaţă, expusă de aplicaţia de import EIS, cu una sau mai multe metode, fiecare luând obiecte de date ca argumente şi returnează valori. Pe partea de implementare, există un CCI implementat de un adaptor de resurse.

Implementarea runtime a importului EIS conectează interfaţa de pe partea clientului şi CCI-ul. Importul mapează invocarea metodei pe interfaţă cu invocarea de pe CCI.

Legările sunt create la trei niveluri: legarea de interfaţă, ce apoi utilizează legările de metode conţinute, ce mai apoi utilizează legările de date.

Legarea de interfaţă leagă interfaţa importului cu conexiunea la sistemul EIS ce furnizează aplicaţia. Acest lucru reflectă faptul că setul de aplicaţii, reprezentat de interfaţă, este furnizat de instanţa specifică a EIS, iar conexiunea furnizează accesul la această instanţă. Elementul de legare conţine proprietăţi cu destule informaţii pentru a crea conexiunea (aceste proprietăţi fac parte din instanţa javax.resource.spi.ManagedConnectionFactory).

Legarea de metodă asociază metoda cu interacţiunea specifică cu sistemul EIS. Pentru JCA, interacţiunea este caracterizată de setul de proprietăţi al implementării de interfaţă javax.resource.cci.InteractionSpec. Elementul de interacţiune al legării de metodă conţine aceste proprietăţi, împreună cu numele clasei, astfel furnizând destule informaţii pentru a realiza interacţiunea. Legarea de metodă utilizează legări de date ce descriu maparea argumentului şi rezultatului metodei de interfaţă la reprezentarea EIS.

Scenariul runtime pentru un import EIS este după cum urmează:

1. Metoda de pe interfaţa de import este invocată utilizând modelul de programare SCA.
2. Cererea, ce ajunge la importul EIS, conţine numele metodei şi argumentele acesteia.
3. Importul întâi creează o implementare legare de interfaţă; apoi, utilizând date din legarea de import, acesta creează o ConnectionFactory şi le asociază pe cele două. Adică, importul apelează setConnectionFactory pe legarea de interfaţă.
4. este creată implementarea legării de metodă ce se potriveşte cu metoda invocată.
5. Instanţa javax.resource.cci.InteractionSpec este creată şi populată; apoi, legările de date sunt utilizate pentru a lega argumentele metodei la un format înţeles de adaptorul de resurse.
6. Interfaţa CCI este utilizată pentru a realiza interacţiunea.
7. Atunci când apelul este returnat, legarea de date este utilizată pentru a crea rezultatul invocării, iar acesta este returnat apelantului.

Exporturile

Rolul exportului EIS este de a face o punte între o componentă SCA şi un EIS extern. Aplicaţiile externe pot fi tratate ca un export EIS. În acest caz, aplicaţia externă trimite datele sale în formă de notificări periodice. Un export EIS poate fi gândit ca o aplicaţie de abonare ce ascultă o cerere externă de la EIS. Componenta SCA ce utilizează exportul EIS o vizualizează ca o aplicaţie locală.

Exportul EIS furnizează componentelor SCA o vizualizare uniformă a aplicaţiilor externe modulului. Acest lucru le permite componentelor să comunice cu un EIS, cum ar fi SAP, Siebel, sau PeopleSoft, utilizând un model SCA consistent.

Exportul prezintă o implementare ascultător ce primeşte cereri de la EIS. Ascultătorul implementează o interfaţă ascultător specifică adaptorului de resurse. Exportul conţine de asemenea o interfaţă ce implementează componente, expusă la EIS prin export.

Implementarea runtime a unui export EIS conectează ascultătorul cu interfaţa ce implementează componente. Exportul mapează cererea EIS cu invocarea operaţiilor corespunzătoare de pe componente. Legările sunt create la trei niveluri: o legare ascultător, ce apoi utilizează o metodă nativă conţinută , ce apoi utilizează o legare de date .

Legarea ascultător leagă ascultătorul ce recepţionează cererile cu componenta expusă prin export. Definiţia de export conţine numele componentei; runtime-ul o localizează şi înaintează cererile la aceasta.

Legarea de metodă nativă asociază metoda nativă sau tipul de eveniment recepţionat de către ascultător cu operaţia implementată de componenta expusă prin calea exportului. Nu există nicio relaţie între metoda invocată pe ascultător şi tipul de eveniment; toate evenimentele ajung prin una sau mai multe metode ale ascultătorului. Legarea de metodă nativă utilizează selectorul de funcţii definit în export pentru a extrage numele metodei netive din datele de intrare şi legările de date pentru a lega formatul de date al EIS cu un format înţeles de componentă.

Scenariul runtime pentru un export EIS este după cum urmează:

1. Cererea EIS declanşează invocarea metodei pe implementarea ascultătorului.
2. Ascultătorul localizează şi invocă exportul, pasându-i toate argumentele de invocare.
3. Exportul creează implementarea legării ascultătorului.
4. Exportul instanţiază selectorul de funcţii şi îl setează pe legarea ascultătorului.
5. Exportul iniţializează legările metodelor native şi le adaugă la legarea ascultătorului. Pentru fiecare legare de metodă nativă, legările de date sunt de asemenea iniţializate.
6. Exportul invocă legarea ascultătorului.
7. Legarea ascultătorului localizează componentele exportate şi utilizează selectorul de funcţii pentru a extrage numele metodei native.
8. Acest nume este utilizat pentru a localiza legarea metodei native, ce apoi invocă componente destinaţie.

Stilul de interacţiune al adaptorului permite legării de export EIS să invoce componenta destinaţie fie asincron (implicit) fie sincron.

Adaptoare de resurse

Dezvoltaţi un import sau un export cu vrăjitorul de servicii externe şi, în dezvoltarea acestuia, includeţi un adaptor de resurse. Adaptoarele ce vin cu IBM Integration Designer folosite pentru a accesa sisteme CICS, IMS, JD Edwards, PeopleSoft, SAP şi Siebel sunt intenţionate doar pentru scopuri de dezvoltare şi testare. Acest lucru presupune că le utilizaţi pentru a dezvolta şi a testa aplicaţiile dumneavoastră.

Odată ce aţi implementat aplicaţia, veţi avea nevoie de adaptoare runtime licenţiate pentru a o rula. Totuşi, atunci când construiţi serviciul dumneavoastră, puteţi îngloba adaptorul împreună cu serviciul. Licenţierea adaptorului vă poate permite să utilizaţi adaptorul înglobat ca adaptor runtime licenţiat. Aceste adaptoare se potrivesc cu Arhitectura Java EE Connector (JCA 1.5). JCA, un standard deschis, este standardul Java EE pentru conectivitatea EIS. JCA furnizează un cadru de lucru gestionat; care este, Quality of Service (QoS) este furnizat de un server de aplicaţii, care oferă gestiunea şi securitatea ciclului de viaţă pentru tranzacţii. Acestea se potrivesc de asemenea cu specificaţiile Enterprise Metadata Discovery cu excepţia adaptorului de resurse IBM CICS ECI şi a conectorului IBM IMS pentru Java.

Adaptoarele WebSphere Business Integration, un set mai vechi de adaptoare, sunt de asemenea suportate de către vrăjitor.

Resursele Java EE

Modulul EIS, un modul SCA ce urmează tiparul modulului EIS, poate fi implementat pe platforma Java EE.

Implementarea modulului EIS pe platforma Java EE are ca rezultat o aplicaţie care este gata de pornire, împachetată ca fişier EAr şi implementată pe server. Toate artefactele Java EE şi resursele sunt create; aplicaţia este configurată şi gata de rulare.

Proprietăţi dinamice JCA InteractionSpec şi ConnectionSpec:

Legarea EIS poate accepta intrare pentru InteractionSpec şi ConnectionSpec specificate prin utilizarea unui obiect bine definit de date copil ce acompaniază datele utile. Aceasta permite interacţiuni cerere-răspuns dinamice cu un adaptor de resurse prin InteractionSpec şi autentificare de componentă prin ConnectionSpec.

javax.cci.InteractionSpec conţine informaţii despre cum ar trebui tratată cererea de interacţiune cu adaptorul de resurse. Poate de asemenea conţine informaţii despre cum a fost obţinută interacţiunea după cerere. Aceste comunicări pe două căi prin interacţiuni sunt uneori referite ca *conversaţii*.

Legarea EIS aşteaptă datele utile ce vor fi un argument pentru adaptorul de resurse să conţină un obiect de date copil numit **proprietăţi**. Acest obiect de date proprietăţi va conţine perechi nume/valoare, cu numele proprietăţilor InteractionSpec într-un anumit format. Regulile de formatare sunt:

- Numele trebuie să înceapă cu prefixul **IS**, urmat de numele proprietăţii. De exemplu, un o specificaţie de interacţiune cu o proprietate JavaBeans numită **InteractionId** ar specifica numele proprietăţii ca **ISInteractionId**.
- Perechea nume/valoare reprezintă numele şi valoarea tipului simplu al proprietăţii InteractionSpec.

În acest exemplu, o interfaţă specifică faptul că intrarea unei operaţii este un obiect de date **Account**. Această interfaţă invocă o aplicaţie de legare de import EIS cu intenţia de a trimite şi recepţiona o proprietate InteractionSpec dinamică numită **workingSet** cu valoarea **xyz**.

Graficul operaţional sau obiectele business de pe server conţin un obiect business **properties** de bază ce permite trimiterea datelor specifice protocolului cu datele utile (payload). Acest obiect business **properties** este încorporat şi nu trebuie specificat în schema XML când construiţi un obiect business. Trebuie doar creat şi utilizat. Dacă aveţi definite propriile tipuri de date bazat pe o schemă XML, trebuie să specificaţi un element **properties** ce vă conţine perechile nume/valoare aşteptate.

```
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");
```

Creaţi datele utile.

```
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");
```

Pentru numele workingSet, setaţi valoarea aşteptată (**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");
```

Puteţi folosi proprietăţi ConnectionSpec pentru autentificarea componentelor dinamice. Se aplică aceleaşi reguli ca mai sus, exceptând faptul că prefixul numelui proprietăţii trebiue să fie **CS** (în loc de **IS**). Proprietăţile ConnectionSpec nu sunt cu sens dublu. Acelaşi obiect de date de **proprietăţi** poate conţine atât proprietăţi IS cât şi CS.

Pentru a utiliza proprietăţi ConnectionSpec, setaţi **resAuth** specificat pe legarea de import la **Application**. De asemenea, asiguraţi-vă că adaptorul de resurse suportă autorizarea componentelor. Vedeţi capitolul 8 din J2EE Connector Architecture Specification pentru detalii suplimentare.

Clienţi externi cu legări EIS:

Serverul poate trimite mesaje către, sau primi mesaje de la, clienţi externi folosind legări EIS.

Un client extern, de exemplu un portal web sau un EIS, trebuie să trimită un mesaj către un modul SCA din server sau trebuie să fie invocat de o componentă din cadrul serverului.

Clientul invocă importul EIS la fel ca orice altă aplicaţie, folosind fie DII (Dynamic Invocation Interface) fie interfaţa Java.

1. Clientul extern creează o instanţă a ServiceManager şi caută importul EIS folosindu-i numele de referinţă. Rezultatul căutării este o implementare a interfeţei serviciului.
2. Clientul creează un argument de intrare, un obiect de date generic, creat dinamic folosind schema obiectului de date. Acest pas este realizat folosind implementarea interfeţei Service Data Object DataFactory.
3. Clientul extern invocă EIS-ul şi obţine rezultatele necesare.

Alternativ, clientul poate invoca importul EIS folosind interfaţa Java.

1. Clientul creează o instanţă a ServiceManager şi caută importul EIS folosindu-i numele de referinţă. Rezultatul căutării este o interfaţă Java a importului EIS.
2. Clientul creează un argument de intrare şi un obiect de date tastat.
3. Clientul invocă EIS şi obţine rezultatele necesare.

Interfaţa exportului EIS defineşte interfaţa componentei SCA exportate ce este disponibilă aplicaţiilor EIS externe. Vă puteţi gândi la această interfaţă ca la interfaţa pe care o va invoca o aplicaţie externă (precum SAP sau PeopleSoft) prin implementarea runtime-ului aplicaţiei exportului EIS.

Exportul foloseşte EISExportBinding pentru a lega servicii exportate de aplicaţia EIS externă. Vă permite să abonaţi o aplicaţie conţinută în modulul dumneavoastră SCA să asculte cereri de servicii EIS. Legarea de export EIS specifică maparea dintre definiţia evenimentelor de intrare aşa cum este înţeleasă de adaptorul de resurse (folosind interfeţe Java EE Connector Architecture) şi invocarea operaţiilor SCA.

EISExportBinding necesită ca serviciile EIS externe să fie bazate pe contracte de intrare Java EE Connector Architecture 1.5. EISExportBinding necesită ca un handler de date sau o legare de date să fie specificate, fie la nivel de legare, fie la nivel de metodă.

Legări JMS:

Un furnizor JMS (Java Message Service) permite mesageria bazată pe API-ul Java Messaging Service şi pe modelul de programare. Acesta asigură fabrici de conexiune JMS pentru a crea conexiuni pentru destinaţiile JMS şi pentru a trimite şi primi mesaje.

Legările JMS pot fi folosite atunci când interacţionaţi cu legarea furnizorului SIB (Service Integration Bus) şi sunt conforme cu JMS şi JCA 1.5.

Puteţi utiliza legările de export şi import JMS ale unui modul SCA (Service Component Architecture) pentru a apela şi primi mesaje de la sisteme JMS externe.

Legările de import şi export JMS asigură integrarea în aplicaţiile JMS folosind furnizorul SIB JMS bazat pe JCA 1.5 care face parte din WebSphere Application Server. Alte adaptoare pentru resurse JMS bazate pe JCA 1.5 nu sunt suportate

Privire generală asupra legărilor JMS:

Legările JMS asigură conectivitatea între mediul SCA (Service Component Architecture) şi sistemele JMS.

Legări JMS

Componentele majore ale ambelor legări JMS pentru import şi export sunt:

- Adaptor resursă: activează conectivitatea bidirecţională, gestionată între un modul SCA şi sistemele JMS externe
- Conexiuni: încapsulează o conexiune virtuală între un client şi o aplicaţie furnizor
- Destinaţii: sunt folosite de un client pentru a specifica ţinta mesajelor pe care le produce sau sursa mesajelor pe care le primeşte
- Date de autentificare: sunt folosite pentru a securiza accesul la legare

Caracteristicile cheie ale legărilor JMS

Anteturi speciale

Proprietăţile anteturilor speciale sunt utilizate în importurile şi exporturile JMS pentru a-i spune destinaţiei cum să trateze mesajul.

De exemplu, TargetFunctionName mapează de la metoda nativă la metoda de funcţionare.

Resurse Java EE

Un număr de resurse Java EE este creat la implementarea importurilor şi exporturilor JMS într-un mediu Java EE.

ConnectionFactory
Utilizată de clienţi pentru a crea o conexiune la furnizorul JMS.

ActivationSpec
Importurile utilizează acest lucru pentru a recepţiona răspunsul la o cerere; exporturile o utilizează la configurarea punctelor finale ale mesajelor ce reprezintă ascultători de mesaje în interacţiunea acestora cu sistemul de mesagerie.

Destinaţii

- Destinaţie de trimitere: pe un import, aceasta este unde cererea sau mesajul de ieşire este trimis; pe un export, aceasta este destinaţia unde mesajul de răspuns va fi trimis, dacă nu este depreciată de câmpul antetului JMSReplyTo din mesajul de intrare.
- Destinaţie de recepţionare: unde mesajul de intrare va fi plasat; cu importuri, acesta este un răspuns; cu exporturi, este o cerere.
- Destinaţie de callback: destinaţia sistemului SCA JMS utilizată pentru a stoca informaţii de corelare. Nu citiţi sau scrieţi la această destinaţie.

Operaţia de instalare creează ConnectionFactory şi trei destinaţii. De asemenea creează ActivationSpec pentru a permite ascultătorului de mesaje runtime să asculte pentru replici pe destinaţia de recepţionare. Proprietăţile ale acestor resurse sunt specificate în fişierul de import sau de export.

Integrarea JMS şi adaptoarele de resurse:

JMS (Java Message Service) asigură integrare printr-un adaptor disponibil de resurse bazat pe JMS JCA 1.5. Suportul complet pentru integrarea JMS este furnizat pentru adaptorul de resurse JMS SIB (Service Integration Bus).

Utilizaţi un furnizor JMS pentru adaptorul de resurse JCA 1.5 atunci când doriţi integrarea într-un sistem extern JMS conform cu JCA 1.5. Serviciile externe conforme cu JCA 1.5 pot primi şi trimite mesaje pentru a se integra în componentele SCA-ului (service component architecture) dumneavoastră folosind adaptorul de resurse JMS SIB.

Utilizarea altor adaptoare de resurse JCA 1.5 specifice furnizorului nu este acceptată.

Legări de import şi export JMS:

Puteţi face ca modulele SCA să interacţioneze cu serviciile furnizate de aplicaţiile JMS externe utilizând legături de import şi export JMS.

Legări de import JMS

Conexiunile către furnizorul JMS asociat destinaţiilor JMS sunt create folosind o fabrică de conexiuni JMS. Utilizaţi obiecte administrative pentru fabrica de conexiuni pentru a gestiona fabricile de conexiuni JMS pentru furnizorul implicit de mesaje.

Interacţiunea cu sistemele JMS externe include utilizarea destinaţiilor pentru trimiterea cererilor sau primirea răspunsurilor.

Se oferă suport pentru două tipuri de scenarii de utilizare pentru legările de import JMS, în funcţie de tipul operaţiei invocate:

- Primul tip: Importul JMS pune un mesaj în destinaţia de trimitere configurată în legarea de import. Nu este setat nimic în câmpul replyTo din antetul JMS.
- Al doilea tip (cerere-răspuns): Importul JMS pune un mesaj în destinaţia de trimitere, iar apoi persistă răspunsul pe care îl primeşte de la componenta SCA.

  Legarea de import poate fi configurată (folosind câmpul **chema de corelare a răspunsului** în Integration Designer), astfel încât să aştepte ID-ul de corelare al mesajului pentru răspuns ce a fost copiat din ID-ul mesajului de cerere (cel implicit) sau din ID-ul de corelare al mesajului de cerere. Legarea de import poate fi de asemenea configurată astfel încât să utilizez o destinaţie de răspuns dinamică temporară pentru a corela răspunsurile cu cererile. Pentru fiecare cerere se creează o destinaţie temporară, iar importul foloseşte această destinaţie pentru a primi răspunsul.

  Destinaţia receive este setată în proprietatea antetului replyTo din mesajul de ieşire. Un ascultător de mesaje este implementat pentru a asculta la destinaţia de recepţionare, iar în momentul în care este primit un răspuns, acesta transminte răspunsul înapoi către componentă.

Pentru ambele scenarii de utilizare pot fi specificate atât proprietăţile dinamice, cât şi cele statice ale antetului. Proprietăţile statice pot fi setate din legarea metodei de import JMS. Unele dintre aceste proprietăţi au semnificaţii speciale pentru runtime-ul JMS SCA.

Este important de menţionat că JMS este o legare asincronă. Dacă o componentă apelantă invocă un import JMS în mod sincron (pentru o operaţie bidirecţională), atunci aceasta este blocată până când răspunsul este returnat de serviciul JMS.

Figura 32 la pagina 113 ilustrează modul în care importul este legat de serviciul extern.

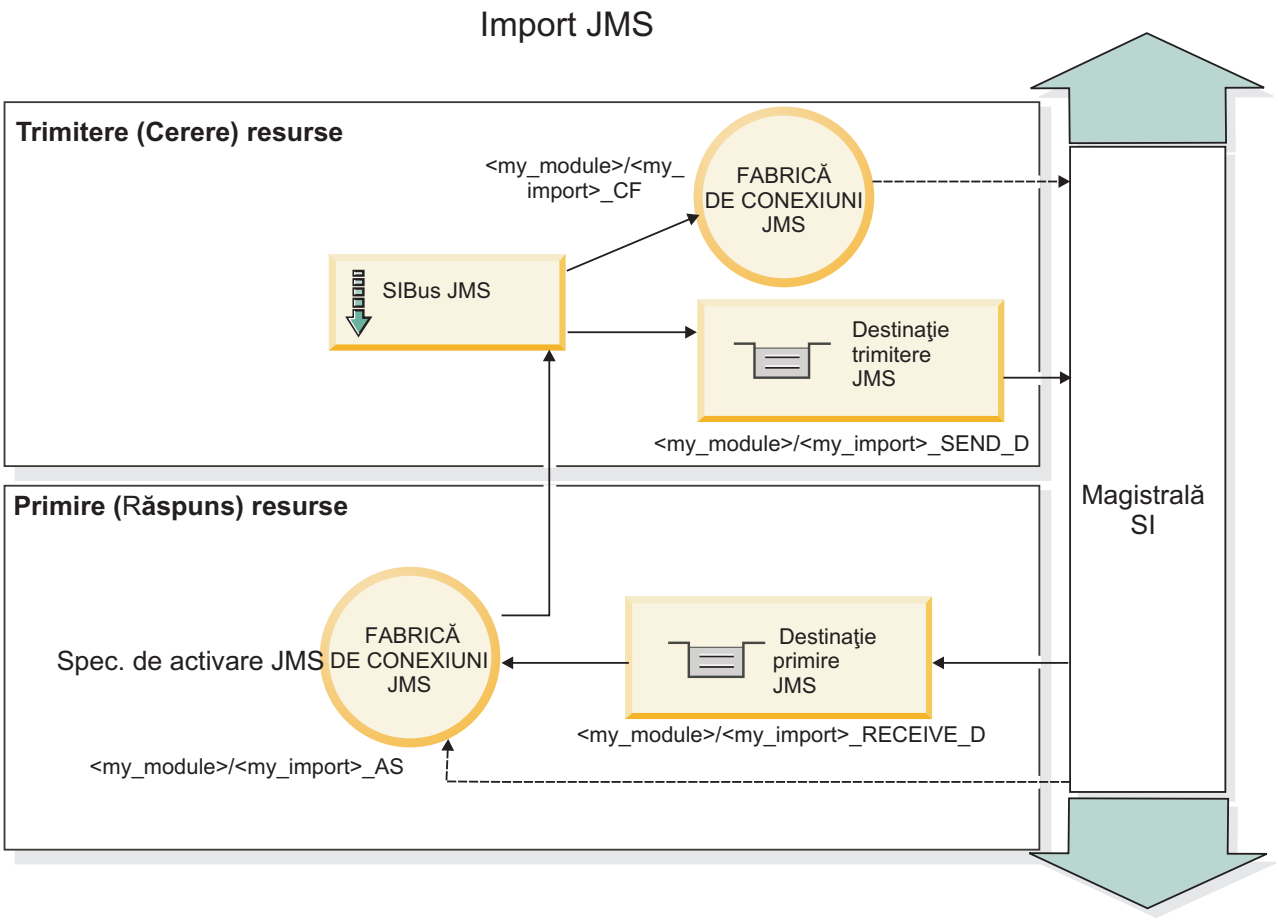


Figura 32. Resurse pentru legarea de import JMS

Legări de export JMS

Legări de export JMS pun la dispoziţie mijloacele prin care modulele SCA asigură servicii către aplicaţiile JMS externe.

Conexiunea care face parte dintr-un export JMS este o specificaţie de activare configurabilă.

Un export JMS a trimis şi a primit destinaţiile.

- Destinaţia receive este acolo unde ar trebui să fie pus mesajul de intrare pentru componenta ţintă.
- Destinaţia send este acolo unde va fi trimis răspunsul, cu excepţia cazului în mesajul de intrare a înlocuit-o folosind proprietatea antetului replyTo.

Un ascultător de mesaje este implementat pentru a asculta cererile ce intră prin destinaţia primire specificată în legarea de export. Destinaţia specificată în câmpul send este utilizat pentru a trimite răspunsul către cererea de intrare în cazul în care componenta invocată furnizează un răspuns. Destinaţia specificată în câmpul replyTo din mesajul de intrare înlocuieşte destinaţia specificată în send.

Figura 33 la pagina 114 ilustrează modul în care solicitantul extern este legat de export.

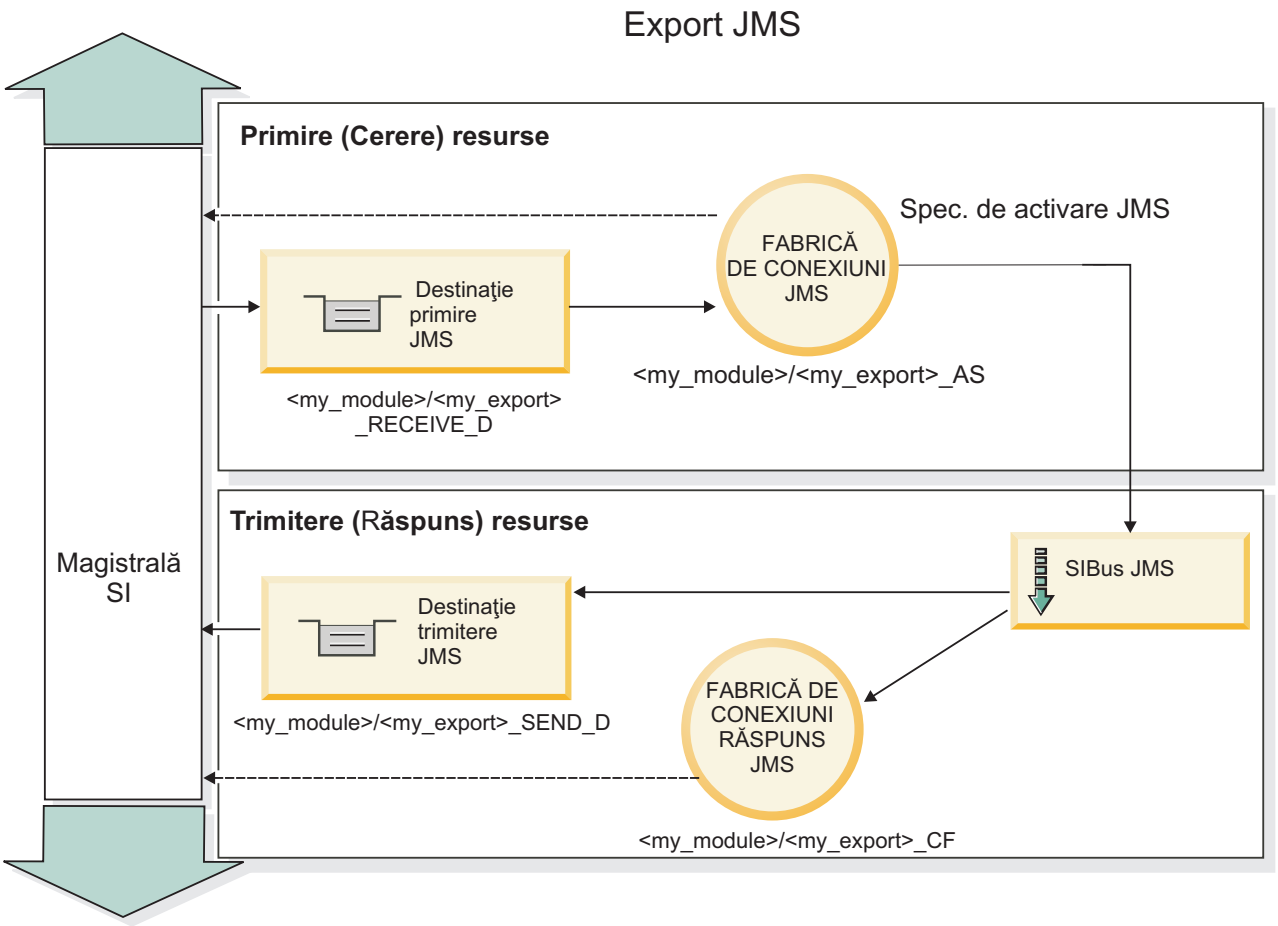


Figura 33. Resurse pentru legarea de export JMS

Anteturi JMS:

Un mesaj JMS conţine două tipuri de anteturi- antetul sistemului JMS şi mai multe proprietăţi JMS. Ambele tipuri de anteturi pot fi accesate fie într-un modul de mediere din SMO (Service Message Object) fie folosind API-ul ContextService.

Antetul sistemului JMS

Antetul sistemului JMS este reprezentat în SMO prin elementul JMSHeader care conţine toate câmpurile care se găsesc de obicei într-un antet JMS. Deşi acestea pot fi modificate în modul de mediere (sau ContextService), unele câmpuri din antetul sistemului JMS setate în SMO nu vor fi trimise în mesajul JMS de ieşire pe măsură ce acestea sunt înlocuite de sistem sau valori statice.

Câmpurile cheie din antetul sistemului JMS care pot fi actualizate într-un modul mediere (sau ContextService) sunt:

- **JMSType** şi **JMSCorrelationID** – valorile proprietăţilor specifice antetului mesajului predefinit
- **JMSDeliveryMode** – valori pentru modul de livrare (persistent sau nepersistent; valoarea implicită este persistent)
- **JMSPriority** – valoare prioritate (de la 0 la 9; valoarea implicită este JMS_Default_Priority)

Proprietăţi JMS

Proprietăţile JMS sunt reprezentate în SMO sub formă de intrări în lista Proprietăţi. Proprietăţile pot fi adăugate, actualizate sau şterse într-o mediere sau folosind API-ul ContextService.

De asemenea, proprietăţile pot fi setate în legarea JMS. Proprietăţile care sunt setate în mod static înlocuiesc setările (cu acelaşi nume) care sunt setate în mod dinamic.

Proprietăţile utilizatorului propagate din alte legări (de exemplu, o legare HTTP) va fi pusă în legarea JMS sub formă de proprietăţi JMS.

Setările propagării anteturilor

Propagarea antetului şi proprietăţilor sistemului JMS fie de la mesajul JMS de intrare către componentele următoare, fie de la componentele anterioare către mesajul JMS de ieşire poate fi controlată prin steguleţul Propagate Protocol Header din legare.

Atunci când Propagate Protocol Header este setat, informaţiilor de antet le este permis să circule către mesaj sau către componenta ţintă, aşa cum este descris în următoarea listă:

- Cerere de export JMS

  Antetul JMS primit în mesaj va fi propagat către o componentă ţintă prin intermediul serviciului de context. Proprietăţile JMS primite în mesaj vor fi propagate către o componentă ţintă prin intermediul serviciului de context.
- Răspuns la exportul JMS

  Oricare dintre câmpurile din antetul JMS din serviciul de contextul va fi utilizat în mesajul de ieşire, dacă nu este înlocuită de proprietăţile statice stabilite în legarea de export JMS. Oricare dintre proprietăţile setate în serviciul de context va fi utilizată în mesajul de ieşire, dacă nu este înlocuită de proprietăţile statice stabilite în legarea de export JMS.
- Cerere de import JMS

  Oricare dintre câmpurile din antetul JMS din serviciul de context va fi utilizat în mesajul de ieşire, dacă nu este înlocuit de proprietăţile statice stabilite în legarea de import JMS. Oricare dintre proprietăţile setate în serviciul de context va fi utilizată în mesajul de ieşire, dacă nu este înlocuită de proprietăţile statice stabilite în legarea de import JMS.
- Răspuns import JMS

  Antetul JMS primit în mesaj va fi propagat către o componentă ţintă prin intermediul serviciului de context. Proprietăţile JMS primite în mesaj vor fi propagate către o componentă ţintă prin intermediul serviciului de context.

Schema de corelare a destinaţiilor de răspuns dinamice temporare JMS:

Schema de corelare a destinaţiilor de răspuns dinamice temporare determină o coadă dinamică unică sau un subiect care urmează să fie creat pentru fiecare cerere trimisă.

Destinaţia statică de răspuns menţionată în import este utilizată pentru a obţine natura cozii dinamice temporare de destinaţie sau subiectul. Acest lucru este setat în câmpul **ReplyTo** al cererii, iar importul JMS ascultă pentru răspunsuri pe această destinaţie. În cazul în care răspunsul este primit este cerut de destinaţia statică a răspunsului pentru prelucrarea asincronă. Câmpul **CorrelationID** din răspuns nu este utilizat şi nu este nevoie să fie setat.

Probleme legate de Tranzacţii

Atunci când este folosită o destinaţie dinamică temporară, cererea trebuie să fie prelucrată în acelaşi fir de execuţie ca răspunsul trimis. Cererea trebuie să fie expediată în afara tranzacţiei globale, şi trebuie să fie comisă înainte să fie primită de serviciul de back-end, şi ca un răspuns să fie returnat.

Persistenţă

Cozile temporare dinamice sunt entităţi de scurtă durată şi nu garantează acelaşi nivel de persistenţă asociat cu o coadă sau subiect static. O coadă dinamică temporară sau subiect nu va mai exista după repornirea serverului şi nici mesajele. După ce mesajul a fost cerut la destinaţia statică de răspuns, acesta îşi păstrează persistenţa definită în mesaj.

Timeout

Importul aşteaptă să primească răspunsul la destinaţia temporară pentru răspunsuri dinamice pentru o durată fixă de timp. Acest interval de timp va fi extras din calificativul de tip Expirare răspuns SCA dacă este setat, iar, în caz contrar, timpul prestabilit este de 60 de secunde. În cazul în care timpul de aşteptare este depăşit, importul aruncă o excepţie de tip ServiceTimeoutRuntimeException.

Clienţi externi:

Serverul poate trimite mesaje către, sau primi mesaje de la, clienţi externi folosind legări JMS.

Un client extern (de exemplu, un portal web sau un sistem de informaţii pentru întreprindere) poate trimite un mesaj către un modul SCA din server, sau poate fi invocat de o componentă din cadrul serverului.

Componentele exportului JMS dezvoltă ascultători de mesaje pentru a asculta cereri ce intră în destinaţia de primire specificată în legarea de export. Destinaţia specificată în câmpul de trimitere este folosită pentru a trimite răspunsul către cererea de intrare dacă aplicaţia invocată furnizează un răspuns. Astfel, un client extern este capabil să invoce aplicaţii cu legarea de export.

Importurile JMS interacţionează cu clienţi externi prin trimiterea de mesaje şi primirea de mesaje de la cozi JMS.

Lucrul cu clienţi externi:

Un client extern (ce este în afara serverului) ar putea avea nevoie să interacţioneze cu o aplicaţie instalată pe server.

Consideraţi un scenariu foarte simplu în care un client extern vrea să interacţioneze cu o aplicaţie de pe server. Figura descrie un scenariu tipic simplu.

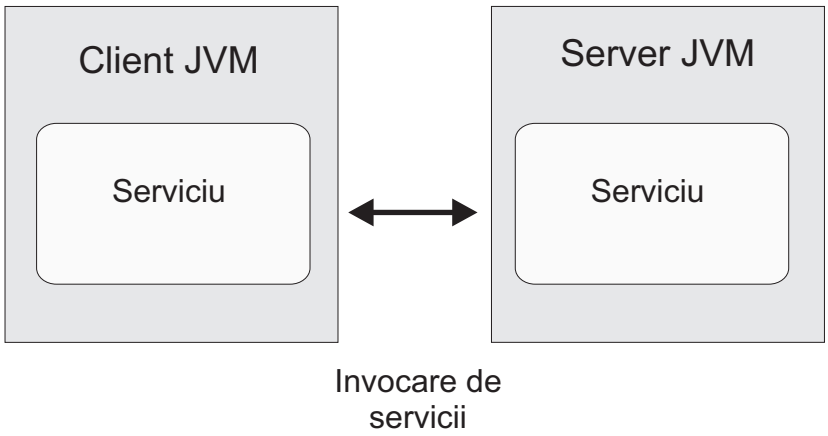


Figura 34. Scenariu simplu utilizare-caz: client extern interacţionează cu aplicaţia serverului

Aplicaţia SCA include un export cu o legare JMS; aceasta face aplicaţia disponibilă pentru clienţi externi.

Atunci când aveţi un client extern în maşina virtuală Java (JVM) separat de serverul dumneavoastră, există mai mult paşi pe care trebuie să-i realizaţi pentru a realiza o conexiune şi pentru a interacţiona cu un export JMS. Clientul obţine un InitialContext cu valorile corecte şi apoi caută resursele prin JNDI. Apoi clientul utilizează clientul specificaţie JMS 1.1 pentru a accesa destinaţiile şi mesajele de trimitere şi primire a destinaţiilor.

Numele implicite JNDI ale resurselor create automat prin runtime sunt listate în subiectul de configurare al acestei secţiuni. Totuşi, dacă aveţi resurse create anterior, utilizaţi acele nume JNDI.

1. Configuraţi destinaţiile JMS şi fabrica de conexiuni pentru a trimite mesajul.

2. Asiguraţi-vă de faptul că contextul JNDI, portul pentru adaptorul de resurse SIB şi portul bootstrapping al mesageriei sunt corecte.

   Serverul utilizează câteva porturi implicite, dar dacă sunt mai multe server instalate pe acel sistem, porturi alternative sunt create la timpul instalării pentru a evita conflicte cu alte instanţe de server. Puteţi utiliza consola administrativă pentru a determina ce porturi ale serverului dumneavoastră sunt implementate. Mergeţi la **Servere > servere de aplicaţii > *nume_server* > Configurare** şi apăsaţi pe **Porturi** din **Comunicaţie**. Puteţi edita portul ce este utilizat.
3. Clientul obţine un context iniţial cu valorile corecte şi apoi caută resurse prin JNDI.
4. Utilizând specificaţiile JMS 1.1, clientul accesează destinaţiile şi mesajele de trimitere şi recepţionare de pe destinaţii.

Depanare legări JMS:

Puteţi diagnostica şi repara problemele cu legările JMS.

Excepţii de implementare

Ca răspuns la condiţii de eroare, implementarea JMS import şi export poate returna una din cele două tipuri de excepţii:

- Service Business Exception: această excepţie este returnată dacă fault specificată pe interfaţa serviciului business (WSDL port type) a survenit.
- Service Runtime Exception: apărută în toate celelalte cazuri. În cele mai multe cazuri, excepţia cause va conţine excepţia originală (JMSException).

  De exemplu, un import aşteaptă numai un mesaj de răspuns pentru fiecare mesaj de cerere. Dacă mai mult de un răspuns soseşte, sau dacă un răspuns întârziat (unul pentru care expirarea răspunsului SCA are loc) soseşte, este aruncată o excepţie Service Runtime. Tranzacţia este repetată, şi mesajul de răspuns este retras din coadă sau este manevrat de către managerul de eveniment.

Condiţii de eşec primar

Condiţiile de eşec primar ale legărilor JMS sunt determinate de către sensurile tranzacţionale, de către configuraţia furnizorului JMS sau prin referire la comportamente existente în alte componente. Condiţiile de eşec primar includ:

- Eşec la conexiunea la furnizorul sau destinaţia JMS.

  Un eşec în conexiunea la furnizorul JMS pentru a primi mesaje va avea ca rezultat începutul căderii ascultătorului de mesaje. Această condiţie va fi jurnalizată în istoricul WebSphere Application Server. Mesajele persistente vor rămâne la destinaţie până când vor fi retrase cu succes (sau expirate).

  Un eşec la conectarea la furnizorul JMS pentru a trimite mesaje în exterior va cauza derularea înapoi a tranzacţiei ce controlează trimiterea.
- Eşecul în analiza unui mesaj de intrare sau în construirea unui mesaj de ieşire.

  Un eşec în legarea de date sau în handler-ul de date cauzează derularea înapoi a tranzacţiei care controlează lucrul.
- Eşec la trimiterea mesajului de ieşire.

  Un eşec la trimiterea unui mesaj cauzează derularea înapoi a tranzacţiei relevante.
- Mesaje de răspuns multiple sau întârziate neaşteptat.

  Importul aşteaptă numai un mesaj de răspuns pentru fiecare mesaj de cerere. De asemenea perioada de timp validă în care un răspuns poate fi primit este determinată de către calificativul SCA Response Expiration de pe cerere. Când un răspuns soseşte sau timpul de expirare este depăşit, înregistrarea de corelare este ştearsă. Dacă mesajele de răspuns ajung neaşteptat sau sosesc târziu, este aruncată o excepţie Service Runtime.
- Excepţie runtime timeout serviciu cauzată de către răspunsul întârziat când se foloseşte schema de corelare a destinaţiei de răspuns dinamic temporară.

  Importul JMS va genera timeout după o perioadă de timp determinată de către calficativul expirare răspuns SCA sau dacă acesta nu este setat va avea valoarea implicită de 60 de secunde.

Mesajele SCA bazate pe JMS ce nu apar în managerul de evenimente eşuate

Dacă mesajele SCA îşi au originea într-o cădere a JMS, ar trebui să vă aşteptaţi să găsiţi aceste mesaje în managerul de evenimente eşuate. Dacă astfel de mesaje nu apar în managerul de evenimente eşuate, asiguraţi-vă că destinaţia de bază SIB a destinaţiei JMS are o valoare maximă de livrări nereuşite mai mare decât **1**. Setarea acestei valori la **2** sau mai mult activează interacţiunea cu managerul de evenimente eşuate în timpul invocărilor SCA pentru legările JMS.

Tratarea excepţiilor:

Modul în care este configurată legarea determină cum excepţiile ce sunt ridicate de handler-e de date sau legări de date sunt tratate. În plus, natura fluxului de mediere dictează comportamentul sistemului când este aruncată o astfel de excepţie.

Poate apărea o varietate de probleme când un handler de date sau o legare de date este apelat(ă) de legarea dumneavoastră. De exemplu, este posibil ca handler-ul de date să primească un mesaj care are date utile corupte, sau este posibil să încerce să citească un mesaj care are un format incorect.

Modul în care legarea dumneavoastră tratează o astfel de excepţie este determinat de modul în care implementaţi handler-ul de date sau legarea de date. Comportamentul recomandat este să vă proiectaţi legarea de date să arunce o **DataBindingException**.

Când orice excepţie runtime, inclusiv o **DataBindingException**, este aruncată:

- Dacă fluxul de mediere este configurat să fie tranzacţional, mesajul JMS , implicit, este memorat în Managerul de evenimente eşuate pentru reluare sau ştergere manuală.

  Notă: Puteţi modifica modul de recuperare de pe legare, astfel încât mesajul să fie derulat înapoi în loc să fie stocat în Managerul de evenimente eşuate.
- Dacă fluxul de mediere nu este tranzacţional, excepţia este înregistrată în istoric şi mesajul este pierdut.

Situaţia este similară pentru un handler de date. Din moment ce handler-ul de date este invocat de legarea de date, orice excepţie a handler-ului de date este înfăşurată într-o excepţie a legării de date. Prin urmare o **DataHandlerException** vă este raportată ca o **DataBindingException**.

Legările Generic JMS:

Conectivitatea furnizorilor de legări Generic JMS cu furnizorii terţă parte care sunt conformi cu JMS 1.1. Operarea legărilor Generic JMS este similară cu cea a legărilor JMS.

Serviciul pus la dispoziţie prin intermediul unei legări JMS permite unui modul SCA (Service Component Architecture) să apeleze sau să primească mesaje de la sisteme externe. Sistemul poate fi un sistem extern JMS.

Legarea Generic JMS asigură integrarea cu furnizorii JMS care nu sunt compatibili cu JCA 1.5 şi care suportă JMS 1.1 şi implementează facilitatea opţională JMS Application Server. Legarea JMS Generic suportă acei furnizori JMS (inclusiv Oracle AQ, TIBCO, SonicMQ, WebMethods şi BEA WebLogic) care nu suportă JCA 1.5, dar suportă Application Server Facility a specificaţiei JMS 1.1. Furnizorul de SIBJMS (WebSphere embedded JMS), care este un furnizor JCA 1.5 JMS, nu este suportat de această legare; atunci când se utilizează acest furnizor, utilizaţi “Legări JMS” la pagina 110.

Utilizaţi această legare generică la integrarea unui sistem JMS conform cu care nu este de tip JCA 1.5 în cadrul unui mediu SCA. Aplicaţiile externe ţintă pot primi şi trimite apoi mesaje pentru a se integra într-o componentă SCA.

Privire generală asupra legărilor JMS generice:

Legările JMS generice sunt legări non-JCA JMS care asigură conectivitate între SCA mediul (Service Component Architecture) şi sistemele JMS care sunt în conformitate cu JMS 1.1 şi care implementează facilitatea opţională a serverului de aplicaţii JMS.

Legări JMS generice

Aspectele majore ale legărilor generice pentru import şi export JMS includ următoarele:

- Port ascultător: permite furnizorilor JMS ce nu se bazează pe JCA să primească mesaje şi să le trimită către un MDB (Message Driven Bean)
- Conexiuni: încapsulează o conexiune virtuală între un client şi o aplicaţie furnizor
- Destinaţii: sunt folosite de un client pentru a specifica ţinta mesajelor pe care le produce sau sursa mesajelor pe care le primeşte
- Date de autentificare: sunt folosite pentru a securiza accesul la legare

Legări de import JMS generice

Legările generice de import JMS permit componentelor din modulul dumneavoastră SCA să comunice cu serviciile oferite de furnizorii externi conformi non-JCA 1.5 JMS.

Partea de conexiuni a importului JMS este o fabrică de conexiuni. O fabrică de conexiuni, obiectul folosit de un client pentru a crea o conexiune către un furnizor, încapsulează un set de parametri de configurare definiţi de către un administrator. Fiecare fabrică de conexiuni este o instanţă a uneia dintre interfeţele ConnectionFactory, QueueConnectionFactory sau TopicConnectionFactory.

Interacţiunea cu sistemele JMS externe include utilizarea destinaţiilor pentru trimiterea cererilor sau primirea răspunsurilor.

Se oferă suport pentru două tipuri de scenarii de utilizare pentru legările generice de import JMS, în funcţie de tipul operaţiei invocate:

- Primul tip: Importul generic JMS pune un mesaj în destinaţia de trimitere configurată în legarea de import. Nu se trimite nimic către câmpul replyTo din antetul JMS.
- Al doilea tip (cerere-răspuns): Importul JMS generic pune un mesaj în destinaţia de trimitere, iar apoi persistă răspunsul pe care îl primeşte de la componenta SCA.

  Destinaţia receive este setată în proprietatea antetului replyTo din mesajul de ieşire. Un MDB (bean controlat de mesaj) este implementat pentru a asculta la destinaţia de recepţionare, iar în momentul în care este primit un răspuns, acesta transmite răspunsul înapoi către componentă.

  Legarea de import poate fi configurată (folosind câmpul **Schema de corelare a răspunsulu** în Integration Designer), astfel încât să aştepte ID-ul de corelare al mesajului pentru răspuns ce a fost copiat din ID-ul mesajului de cerere (cel implicit) sau din ID-ul de corelare al mesajului de cerere.

Pentru ambele scenarii de utilizare pot fi specificate atât proprietăţile dinamice, cât şi cele statice ale antetului. Proprietăţile statice pot fi setate din legarea metodei de import generic JMS. Unele dintre aceste proprietăţi au semnificaţii speciale pentru runtime-ul JMS SCA.

Este important de menţionat faptul că JMS este o legare asincronă. Dacă o componentă apelantă invocă un import JMS generic în mod sincron (pentru o operaţie bidirecţională), atunci aceasta este blocată până când răspunsul este returnat de serviciul JMS.

Figura 35 la pagina 120 ilustrează modul în care importul este legat de serviciul extern.

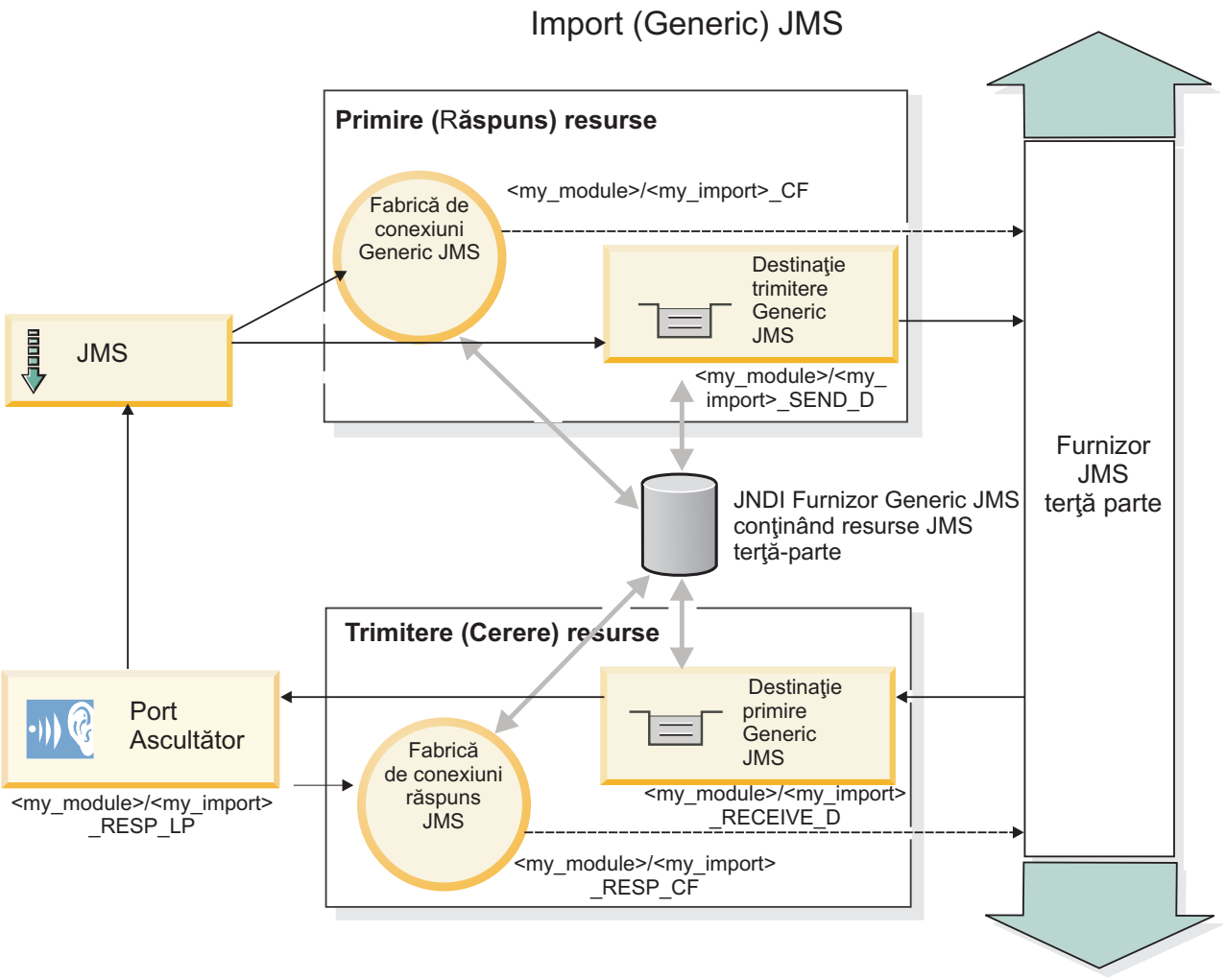


Figura 35. Resurse pentru legarea generică de import JMS

Legări de export JMS generice

Legările de export JMS generice pun la dispoziţie mijloacele prin care modulele SCA asigură servicii către aplicaţiile JMS externe.

Partea de conexiuni a unui export JMS este compus dintr-un ConnectionFactory şi un ListenerPort.

Un export generic JMS a trimis şi a primit destinaţiile.

- Destinaţia primire este acolo unde ar trebui să fie pus mesajul de intrare pentru componenta ţintă.
- Destinaţia trimitere este acolo unde va fi trimis răspunsul, cu excepţia cazului în mesajul de intrare a înlocuit-o folosind proprietatea antetului replyTo.

Este implementat un MDB pentru a asculta cererile ce intră prin destinaţia primire specificată în legarea de export.

- Destinaţia specificată în câmpul trimitere este folosită pentru a trimite răspunsul către cererea de intrare în cazul în care aplicaţia invocată oferă un răspuns.
- Destinaţia specificată în câmpul replyTo din mesajul de intrare înlocuieşte destinaţia specificată în câmpul trimitere.
- Pentru scenariile cerere/răspuns, legarea de import poate fi configurată (folosind câmpul **Schema de corelare a răspunsului** din Integration Designer), astfel încât să se aştepte ca răspunsul să copieze ID-ul mesajului de cerere către câmpul correlation ID al mesajului de răspuns (implicit) sau răspunsul poate copia correlation ID al cererii în câmpul correlation ID din mesajul de răspuns.

Figura 36 ilustrează modul în care solicitantul extern este legat de export.

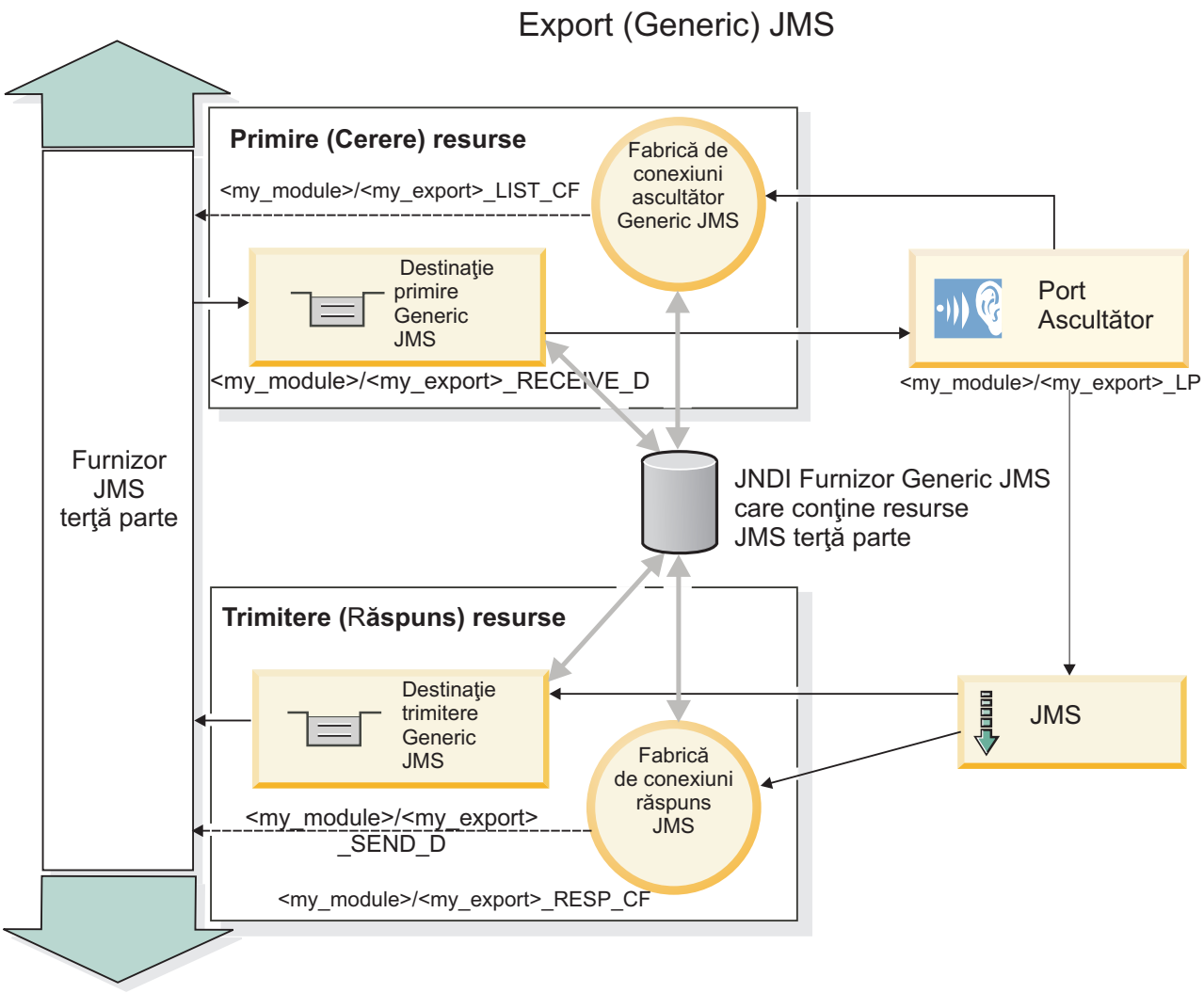


Figura 36. Resurse pentru legarea generică pentru exportul JMS

Caracteristici cheie ale legărilor JMS generic:

Caracteristicile legărilor de import şi export JMS generic sunt consistente cu acelea ale legărilor de import MQ JMS şi cele JMS imbricate WebSphere. Caracteristicile cheie includ definiţii de antet şi acces la resurse Java EE existente. Totuşi, din cauza naturii sale generice, nu există nicio opţiune de conectivitate specifică furnizorilor JMS, iar această legare are capacitate limitată pentru a genera resurse la implementare şi instalare.

Importuri generice

La fel ca aplicaţia de import MQ JMS, implementarea Generic JMS este asincronă şi suportă trei invocări: o cale, două căi (de asemenea cunoscută ca cerere-răspuns), şi callback.

Atunci când importul JMS este implementat, un MDB furnizat de mediul de runtime este implementat. MDB-ul ascultă pentru replici la mesajul de cerere. MDB- ul este asociat cu (ascultă pe) destinaţia trimisă cu cerere în câmpul de antetreplyTo al mesajului JMS.

Exporturi generice

Legările de exporturi JMS generice diferă de legările de export EIS în cazul tratării returnării rezultatului. Un export Generic JMS trimite explicit răspunsul la destinaţia replyTo specificată în mesajul de intrare. Dacă niciuna nu este specificată, destinaţia de trimitere este utilizată.

Atunci când este implementat exportul Generic JMS, un MDB (diferit de cel utilizat pentru importurile Generic JMS) este implementat. Acesta ascultă pentru cereri de intrare pe destinaţia de recepţionare şi apoi dispecerizează cererile pentru a fi procesate de runtime-ul SCA.

Anteturi speciale

Proprietăţile anteturilor speciale sunt utilizate în importurile şi exporturile Generic JMS pentru a informa legarea destinaţie cum să fie tratat mesajul.

De exemplu, proprietatea TargetFunctionName este utilizată de către selectorul de funcţii implicit pentru a identifica numele operaţiei din interfaţa export ce este invocată.

Notă: Legarea de import poate fi configurată pentru a seta antetul TargetFunctionName la numele de operaţie al fiecărei operaţii.

Resurse Java EE

Un număr de resurse Java EE sunt create atunci când o legare JMS este implementată într-un mediu Java EE.

- Portul ascultător pentru ascultarea pe destinaţia de recepţionare (răspuns) (doar două căi) pentru importuri şi pe destinaţia de recepţionare (cerere) pentru exporturi
- Fabrica de conexiuni JMS generic pentru outboundConnection (import) şi inboundConnection (export)
- Destinaţia JMS generic pentru destinaţiile de trimitere (import) şi recepţionare (export; doar două căi)
- Fabrica de conexiuni JMS generic pentru responseConnection (doar două căi şi opţional; altfel, outboundConnection este utilizat pentru importuri, iar inboundConnection este utilizat pentru exporturi)
- Destinaţia JMS generic pentru destinaţiile de recepţionare (import) şi trimitere (export) (doar două căi)
- Destinaţia JMS de callback a furnizorului de mesagerie implicit utilizată pentru a accesa destinaţia coadă de callback SIB (doar două căi)
- Fabrica de conexiuni JMS de callback a furnizorului de mesagerie implicit utilizată pentru a accesa destinaţia JMS de callback (doar două căi)
- Destinaţia coadă de callback SIB utilizată pentru a memora informaţiile despre mesajul de cerere pentru utilizarea în timpul procesării răspunsului (doar două căi)

Taskul de instalare creeazăConnectionFactory, cele trei destinaţii, şi ActivationSpec din informaţiile din fişierele de import şi export.

Anteturi JMS Generic:

Anteturile JMS Generic sunt SDO-uri (Service Data Objects) ce conţin toate proprietăţile proprietăţilor mesajului JMS Generic. Aceste proprietăţi pot fi de la mesajul de intrare sau pot fi proprietăţile ce vor fi aplicate mesajului de ieşire.

Un mesaj JMS conţine două tipuri de anteturi – antetul de sistem JSM şi mai multe proprietăţi JMS. Ambele tipuri de anteturi pot fi accesate fie într-un modul de mediere din SMO (Service Message Object), fie folosind API-ul ContextService.

Următoarele proprietăţi sunt setate static pe methodBinding:

- JMSType
- JMSCorrelationID
- JMSDeliveryMode
- JMSPriority

Legarea JMS Generic suportă modificarea dinamică a anteturilor şi proprietăţilor JMS în acelaşi mod ca legările JMS şi MQ JMS.

Unii furnizori de JMS Generic pun restricţii pe ce proprietăţi pot fi setate de aplicaţie şi în ce combinaţii. Trebuie să vă consultaţi documentaţia produsului de terţă parte pentru informaţii suplimentare. Totuşi, o proprietate suplimentară a fost adăugată la methodBinding, ignoreInvalidOutboundJMSProperties, ce permite să fie propagate orice excepţii.

Anteturile şi proprietăţile mesajelor JMS Generic sunt folosite doar când comutatorul de bază al legării SCDL a arhitecturii componentei de servicii este pornit. Când comutatorul este pornit, informaţiile de context sunt propagate. Implicit, comutatorul este pornit. Pentru a preveni propagarea informaţiilor de context, modificaţi valoarea în **fals**.

Când propagarea contextelor este activată, informaţiile de antet sunt permise să curgă către mesaj sau către componenta ţintă. Pentru a porni şi opri propagarea contextelor, specificaţi **adevărat** sau **fals** pentru atributul contextPropagationEnabled al legărilor de import şi export. De exemplu:

```
<esbBinding xsi:type="eis:JMSImportBinding" contextProgagationEnabled="true">
```

Valoarea implicită este **adevărat**.

Depanarea legăturilor JMS generic:

Puteţi diagnostica şi rezolva probleme cu legarea JMS generic.

Excepţii de implementare

Ca răspuns la diversele condiţii de eroare, implementarea Generic JMS import şi export poate returna una din cele două tipuri de excepţii:

- Service Business Exception: această excepţie este returnată dacă fault specificată pe interfaţa serviciului business (WSDL port type) a survenit.
- Service Runtime Exception: apărută în toate celelalte cazuri. În cele mai multe cazuri, excepţia cause va conţine excepţia (JMSException) originală.

Depanarea mesajului de expirare Generic JMS

Un mesaj de cerere de către furnizorul JMS este subiect de expirare.

Expirare cerere face referire la expirarea mesajului de cerere de către furnizorul JMS atunci când timpul JMSExpiration din mesajul de cerere este atins. Ca şi în cazul altor legări JMS, legarea Generic JMS manevrează expirarea prin setarea expirării la mesajul de apel invers plasat prin import pentru a fi asemănător cu cererea de ieşire. Notificarea expirării mesajului de apel invers va indica faptul că mesajul de cerere a expirat şi clientul ar trebui notificat printr-o excepţie operaţională.

Dacă destinaţia de apel invers este mutată pe un furnizor terţă parte, totuşi, acest tip de cerere de expirare nu este suportat.

Expirare răspuns face referire la expirarea mesajului de răspuns de către furnizorul JMS atunci când timpul JMSExpiration din mesajul de răspuns este atins.

Expirarea răspunsului pentru legarea generică JMS nu este suportat din cauza faptului că expirarea comportamentului exact al furnizorului JMS nu este definită. Totuşi, puteţi verifica dacă răspunsul este expirat atunci când este primit.

Pentru mesaje de cerere de ieşire, valoarea JMSExpiration va fi calculată în timpul aşteptat şi din valorile requestExpiration efectuate în asyncHeader, dacă este setat.

Depanarea erorilor fabricii de conexiuni Generic JMS

Atunci când definiţia anumite tipuri de fabrici de conexiuni în furnizorul dumneavoastră Generic JMS, s-ar putea să primiţi un mesaj de eroare atunci când încercaţi să porniţi aplicaţia. Puteţi modifica fabrica de conexiuni externă pentru a evita această problemă.

Atunci când lansaţi o aplicaţie, este posibil să primiţi următorul mesaj de eroare:

```
Tipul MDB Listener Port JMSConnectionFactory nu corespunde
tipului JMSDestination.
```

Această problemă poate apărea atunci când definiţi fabrici de conexiuni externe. Specific, excepţia poate fi aruncată atunci când creaţi o fabrică de conexiuni subiect JMS 1.0.2, în locul unei fabrici de conexiuni (unificate) JMS 1.1 (ce este una capabilă să suporte atât comunicarea punct-la-punct, cât şi cea de publicare/anonare).

Pentru a rezolva această problemă, realizaţi următorii paşi:

1. Accesaţi furnizorul Generic JMS pe care îl utilizaţi.
2. Înlocuiţi fabrica de conexiuni subiect JMS 1.0.2 pe care aţi definit-o cu fabrica de conexiuni (unificate) JMS 1.1.

Când lansaţi aplicaţia cu fabrica de conexiuni JMS 1.1 definită mai nou, ar trebui să nu mai primiţi un mesaj de eroare.

Mesajele SCA bazate pe Generic JMS nu apar în managerul de evenimente eşuate

Dacă mesajele SCA au originea într-o eşuare a interacţiunii JMS, v-aţi aştepta să găsiţi aceste mesaje în managerul de evenimente eşuate. Dacă asemenea mesaje nu apar în managerul de evenimente eşuate, asiguraţi-vă că valoarea proprietăţii de maxim de reîncercări din ascultătorul ce stă la bază este mai mare sau egală cu 1. Setarea aceste valori la 1 sau mai mult, permite interacţiunea cu managerul de evenimente eşuate în timpul invocării SCA pentru legările Generic JMS.

Tratarea excepţiilor:

Modul în care este configurată legarea determină cum sunt tratate excepţiile ridicate de handler-ele de date sau legările de date. În plus, natura fluxului de mediere dictează comportamentul sistemului când este aruncată o astfel de excepţie.

Poate apărea o varietate de probleme când un handler de date sau o legare de date este apelat(ă) de legarea dumneavoastră. De exemplu, este posibil ca handler-ul de date să primească un mesaj care are date utile corupte, sau este posibil să încerce să citească un mesaj care are un format incorect.

Modul în care legarea dumneavoastră tratează o astfel de excepţie este determinat de cum implementaţi handler-ul de date sau legarea de date. Comportamentul recomandat este să vă proiectaţi legarea de date să arunce o **DataBindingException**.

Situaţia este similară pentru un handler de date. Din moment ce handler-ul de date este invocat de legarea de date, orice excepţie a handler-ului de date este înfăşurată într-o excepţie a legării de date. Prin urmare o **DataHandlerException** vă este raportată ca o **DataBindingException**.

Când orice excepţie runtime, inclusiv o excepţie **DataBindingException**, este aruncată:

- Dacă fluxul de mediere este configurat să fie tranzacţional, mesajul JMS este memorat în Managerul de evenimente eşuate implicit pentru reluare sau ştergere manuală.

  Notă: Puteţi modifica modul de recuperare de pe legare, astfel încât mesajul să fie derulat înapoi în loc să fie stocat în managerul de evenimente eşuate.
- Dacă fluxul de mediere nu este tranzacţional, excepţia este înregistrată în istoric şi mesajul este pierdut.

Situaţia este similară pentru un handler de date. Deoarece handler-ul de date este apelat de legarea de date, o excepţie a handler-ului de date este produsă înăuntrul unei excepţii a legării de date. Prin urmare, o **DataHandlerException** vă este raportată ca o **DataBindingException**.

Legări JMS WebSphere MQ:

Legarea WebSphere MQ JMS asigură integrarea în aplicaţiile externe care folosesc un furnizor bazat pe WebSphere MQ JMS.

Utilizaţi WebSphere MQ JMS legările de export şi import pentru a integra direct în sistemele JMS sau MQ JMS externe din mediul serverului dumneavoastră. Acest lucru elimină nevoia de a utiliza caracteristicile MQ Link sau Client Link ale Magistralei de Integrale a Serviciului.

Atunci când o componentă interacţionează cu un serviciu bazat pe WebSphere MQ JMS prin intermediul unui import, legarea de import WebSphere MQ JMS utilizează o destinaţie către care datele vor fi trimise şi o destinaţie unde va fi primit răspunsul. Conversia datelor către şi de la un mesaj JMS se realizează prin intermediul componentei edge de legare handler sau date pentru datele JMS.

Atunci când un modul SCA oferă un serviciu clienţilor WebSphere MQ JMS, legarea de export WebSphere MQ JMS utilizează o destinaţie la care vor fi recepţionate datele şi către care poate fi trimis răspunsul. Conversia datelor către şi de la un mesaj JMS se realizează prin intermediul handler-ului de date JMS sau prin legarea datelor.

Selectorul funcţiei asigură o mapare operaţiei în cadrul componentei ţintă care va fi invocată.

Privire generală asupra legărilor WebSphere MQ JMS:

Legarea WebSphere MQ JMS asigură integrarea în aplicaţiile externe care folosesc furnizorul WebSphere MQ JMS.

Taskuri administrative WebSphere MQ

Se aşteaptă ca administratorul de sistem pentru WebSphere MQ să creeze WebSphere MQ Queue Manager de bază; acesta va fi folosit de legările WebSphere MQ JMS înainte de a rula o aplicaţie care conţine aceste legări.

Legări de import WebSphere MQ JMS

WebSphere MQ JMS de import permit componentelor din modulul dumneavoastră SCA să comunice cu serviciile oferite de furnizorii bazaţi pe WebSphere MQ JMS. Trebuie să utilizaţi o versiune suportată de WebSphere MQ. Cerinţe detaliate despre hardware şi software se pot găsi pe paginile de suport IBM.

Se oferă suport pentru două tipuri de scenarii de utilizare pentru legările de import WebSphere MQ JMS, în funcţie de tipul de operaţie care este invocat:

- Primul tip: Importul WebSphere MQ JMS pune un mesaj în destinaţia de trimitere configurată în legarea de import. Nu se trimite nimic către câmpul replyTo din antetul JMS.
- Al doilea tip (cerere-răspuns): Importul WebSphere MQ JMS pune un mesaj în destinaţia de trimitere.

  Destinaţia receive este setată în câmpul replyTo din antet. Un MDB (message-driven bean) este implementat pentru a asculta la destinaţia receive (primire), iar în momentul în care este primit un răspuns, acesta transmite răspunsul înapoi către componentă.

  Legarea de import poate fi configurată (folosind câmpul **Schema de corelare a răspunsului** în Integration Designer), astfel încât să aştepte ID-ul de corelare al mesajului pentru răspuns ce a fost copiat din ID-ul mesajului de cerere (cel implicit) sau din ID-ul de corelare al mesajului de cerere.

Pentru ambele scenarii de utilizare pot fi specificate atât proprietăţile dinamice, cât şi cele statice ale antetului. Proprietăţile statice pot fi setate din legarea metodei de import JMS. Unele dintre aceste proprietăţi au semnificaţii speciale pentru runtime-ul JMS SCA.

Este important de menţionat faptul că WebSphere MQ JMS este o legare asincronă. Dacă o componentă apelantă invocă un import WebSphere MQ JMS în mod sincron (pentru o operaţie bidirecţională), atunci aceasta este blocată până când răspunsul este returnat de serviciul JMS.

Figura 37 ilustrează modul în care importul este legat de serviciul extern.

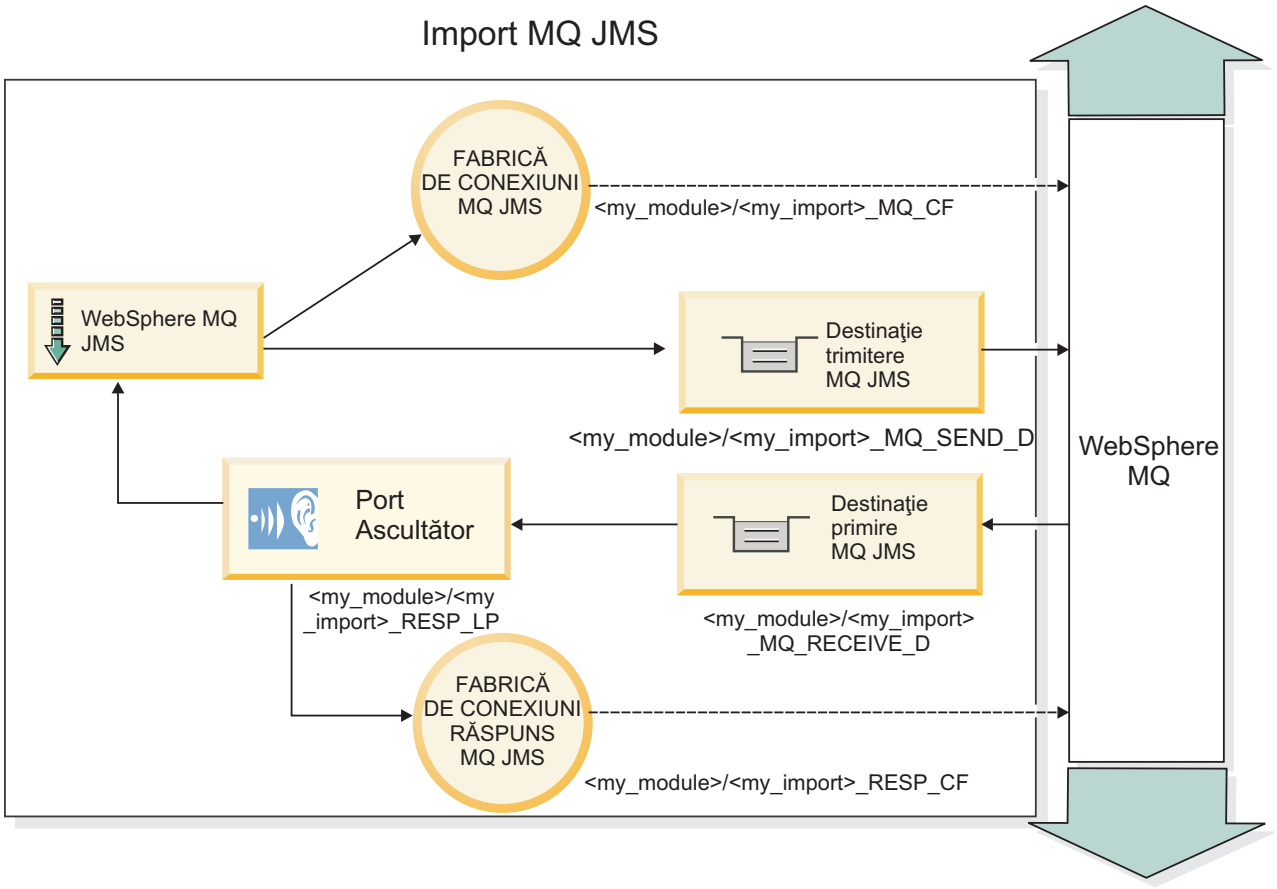


Figura 37. Resurse pentru legarea de import WebSphere MQ JMS

Legări pentru export WebSphere MQ JMS

Legarea de export WebSphere MQ JMS pune la dispoziţie mijloacele prin care modulele SCA asigură servicii către aplicaţiile externe bazate pe WebSphere MQ.

Este implementat un MDB pentru a asculta cererile ce intră prin destinaţia primire specificată în legarea de export. Destinaţia specificată în câmpul trimite este utilizat pentru a trimite răspunsul către cererea de intrare în cazul în care componenta invocată furnizează un răspuns. Destinaţia specificată în câmpul replyTo din mesajul de răspuns înlocuieşte destinaţia specificată în câmpul trimite.

Figura 38 la pagina 127 ilustrează modul în care solicitantul extern este legat de export.

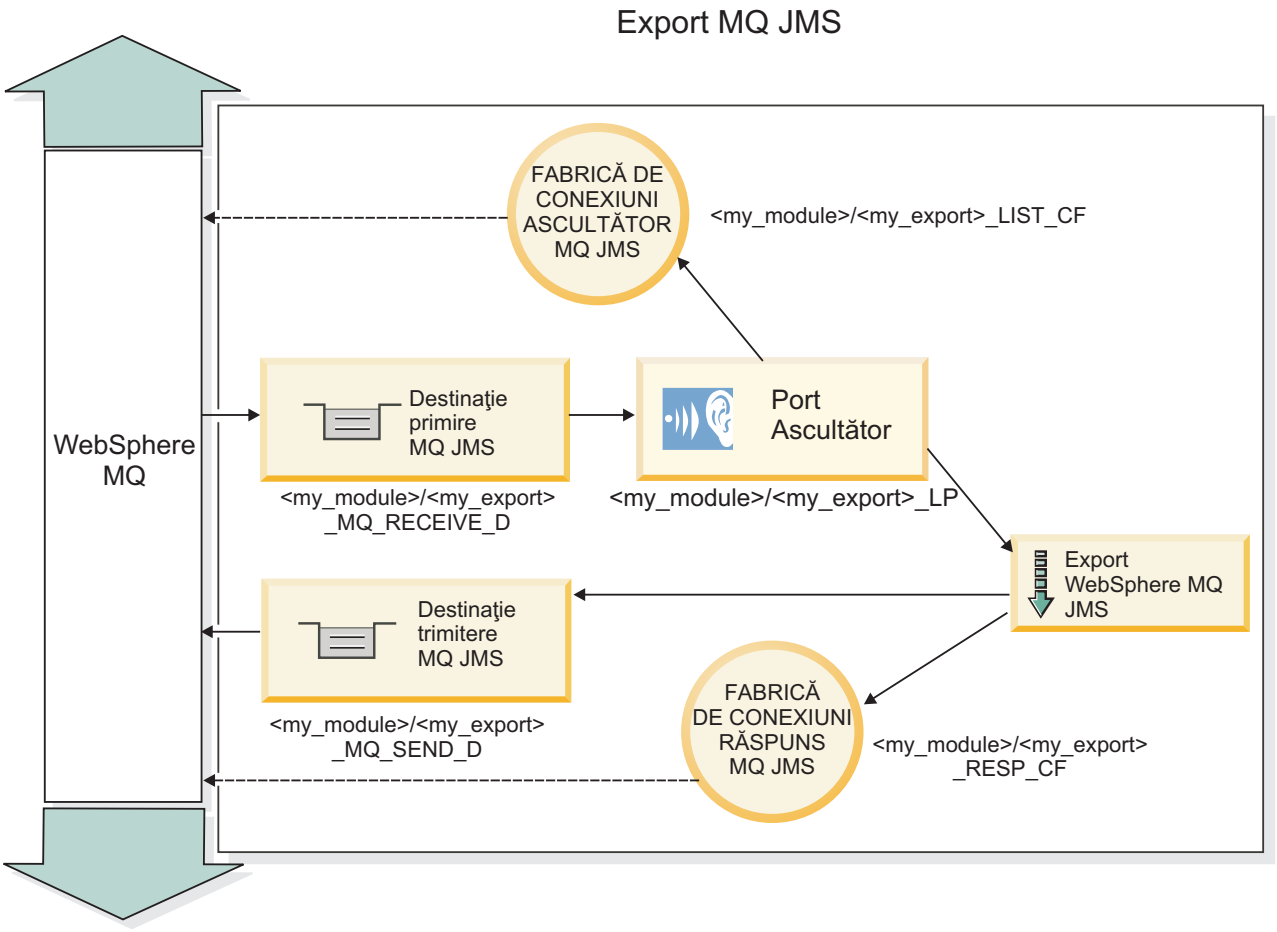


Figura 38. Resurse pentru legarea de export WebSphere MQ JMS

Notă: Figura 37 la pagina 126 şi Figura 38 ilustrează modul în care o aplicaţie dintr-o versiune anterioară a IBM Business Process Manager este legată de un serviciu extern. Pentru aplicaţiile dezvoltate pentru IBM Business Process Manager Versiunea 7.0, se foloseşte Specificare activare în loc de Port ascultător şi Fabrică de conexiuni.

Caracteristici cheie ale legărilor WebSphere MQ JMS:

Caracteristicile cheie ale legărilor WebSphere MQ JMS includ anteturi, artefacte Java EE, şi resurse create Java EE.

Anteturile

Un antet de mesaj JMS conţine un număr de câmpuri predefinite ce conţin valori utilizate de şi de clienţi şi de furnizori pentru a identifica şi a ruta mesajele. Puteţi folosi proprietăţi de legătură pentru a configura aceste antete cu valori fixe, sau antetele pot fi specificate dinamic la rulare.

JMSCorrelationID
Legări la un mesaj înrudit. În mod tipic, acest câmp este setat la şirul identificatorului mesajului căuria îi este dat un răspuns.

TargetFunctionName
Acest antet este utilizat de unul din selectorii de funcţii furnizaţi pentru a identifica operaţia ce este invocată. Setarea proprietăţii de antet JMS TargetFunctionName în mesajele trimise la un export JMS permite acestui selector de funcţii să fie utilizat. Proprietatea poate fi setată direct în aplicaţiile clientului JMS sau la conectarea unui import cu o legare JMS la un export. În acest caz, legarea de import JMS ar trebui configurată pentru a seta antetul TargetFunctionName pentru fiecare operaţie din interfaţă la numele acesteia.

Scheme de corelare

Legările JMS WebSphere MQ furnizează scheme de corelare variate ce sunt utilizate pentru a determina cum vor fi corelate mesajele de crere cu mesajele de răspuns.

RequestMsgIDToCorrelID
JMSMessageID este copiat la câmpul JMSCorrelationID . Aceasta este setarea implicită.

RequestCorrelIDToCorrelID
JMSCorrelationID este copiat la câmpul JMSCorrelationID .

Resurse Java EE

Un număr de resurse Java EE sunt create atunci când un import MQ JMS este implementat la un mediu Java EE.

Parametri

Fabrica de conexiuni MQ
Utilizată de clienţi pentru a crea o conexiune la furnizorul MQ JMS.

Fabrica de conexiuni răspuns
Folosit de runtime-ul SCA MQ JMS atunci când destinaţia de trimitere este pe un Manager coadă diferit de cel al destinaţiei de primire.

Specificaţii de activare
O specificaţie de activare MQ JMS este asociată cu una sau mai multe bean-uri bazate pe mesaje şi furnizează configuraţia necesară pentru a recepţiona mesajele.

Destinaţii

- Trimitere destinaţie:
  - Importuri: Unde sunt trimise mesajele de cerere sau de ieşire.
  - Exporturi: Unde vor fi trimise mesajele de răspuns dacă nu este înlocuit de câmpul de antet JMSReplyTo al mesajului de intrare.
- Destinaţie primire:
  - Importuri: Unde vor fi plasate mesajele de intrare şi de răspuns.
  - Exporturi: Unde vor fi plasate mesajele de intrare şi de cerere.

Anteturi JMS:

Un mesaj JMS conţine două tipuri de anteturi- antetul sistemului JMS şi mai multe proprietăţi JMS. Ambele tipuri de anteturi pot fi accesate fie într-un modul de mediere din SMO (Service Message Object) fie folosind API-ul ContextService.

Antetul sistemului JMS

Antetul sistemului JMS este reprezentat în SMO prin elementul JMSHeader care conţine toate câmpurile care se găsesc de obicei într-un antet JMS. Deşi acestea pot fi modificate în modul de mediere (sau ContextService), unele câmpuri din antetul sistemului JMS setate în SMO nu vor fi trimise în mesajul JMS de ieşire pe măsură ce acestea sunt înlocuite de sistem sau valori statice.

Câmpurile cheie din antetul sistemului JMS care pot fi actualizate într-un modul mediere (sau ContextService) sunt:

- **JMSType** şi **JMSCorrelationID** – valorile proprietăţilor specifice antetului mesajului predefinit
- **JMSDeliveryMode** – valori pentru modul de livrare (persistent sau nepersistent; valoarea implicită este persistent)
- **JMSPriority** – valoare prioritate (de la 0 la 9; valoarea implicită este JMS_Default_Priority)

Proprietăţi JMS

Proprietăţile JMS sunt reprezentate în SMO sub formă de intrări în lista Proprietăţi. Proprietăţile pot fi adăugate, actualizate sau şterse într-o mediere sau folosind API-ul ContextService.

De asemenea, proprietăţile pot fi setate în legarea JMS. Proprietăţile care sunt setate în mod static înlocuiesc setările (cu acelaşi nume) care sunt setate în mod dinamic.

Proprietăţile utilizatorului propagate din alte legări (de exemplu, o legare HTTP) va fi pusă în legarea JMS sub formă de proprietăţi JMS.

Setările propagării anteturilor

Propagarea antetului şi proprietăţilor sistemului JMS fie de la mesajul JMS de intrare către componentele următoare, fie de la componentele anterioare către mesajul JMS de ieşire poate fi controlată prin steguleţul Propagate Protocol Header din legare.

Atunci când Propagate Protocol Header este setat, informaţiilor de antet le este permis să circule către mesaj sau către componenta ţintă, aşa cum este descris în următoarea listă:

- Cerere de export JMS

  Antetul JMS primit în mesaj va fi propagat către o componentă ţintă prin intermediul serviciului de context. Proprietăţile JMS primite în mesaj vor fi propagate către o componentă ţintă prin intermediul serviciului de context.
- Răspuns la exportul JMS

  Oricare dintre câmpurile din antetul JMS din serviciul de contextul va fi utilizat în mesajul de ieşire, dacă nu este înlocuită de proprietăţile statice stabilite în legarea de export JMS. Oricare dintre proprietăţile setate în serviciul de context va fi utilizată în mesajul de ieşire, dacă nu este înlocuită de proprietăţile statice stabilite în legarea de export JMS.
- Cerere de import JMS

  Oricare dintre câmpurile din antetul JMS din serviciul de context va fi utilizat în mesajul de ieşire, dacă nu este înlocuit de proprietăţile statice stabilite în legarea de import JMS. Oricare dintre proprietăţile setate în serviciul de context va fi utilizată în mesajul de ieşire, dacă nu este înlocuită de proprietăţile statice stabilite în legarea de import JMS.
- Răspuns import JMS

  Antetul JMS primit în mesaj va fi propagat către o componentă ţintă prin intermediul serviciului de context. Proprietăţile JMS primite în mesaj vor fi propagate către o componentă ţintă prin intermediul serviciului de context.

Clienţi externi:

Serverul poate trimite mesaje către, sau să primească mesaje de la, clienţi externi folosind legări JMS WebSphere MQ.

Un client extern (de exemplu, un portal web sau un sistem de informaţii pentru întreprindere) poate trimite un mesaj către o componentă SCA din aplicaţie prin intermediul unui export sau poate fi invocat de o componentă SCA din aplicaţie prin intermediul unui import.

Legarea de export WebSphere MQ JMS implementează MDB-uri (bean-uri controlate de mesaj) cu scopul de a asculta cererile ce intră prin destinaţia receive specificată în legarea de export. Destinaţia specificată în câmpul send este folosită pentru a trimite răspunsul către cererea de intrare în cazul în care aplicaţia invocată oferă un răspuns. Cu toate acestea, un client extern poate invoca aplicaţii prin intermediul legării de export.

WebSphere MQ JMS importă legarea în, şi poate trimite un mesaj către, clienţii externi. Acest mesaj ar putea sau nu să solicite un răspuns de la clientul extern.

Informaţii suplimentare despre modul de interacţionare cu clienţii externi folosind WebSphere MQ pot fi găsite în Centrul de informare WebSphere MQ.

Depanarea legărilor WebSphere MQ JMS:

Puteţi diagnostica şi rezolva probleme cu legările WebSphere MQ JMS.

Excepţii de implementare

Ca răspuns la diferite condiţii de eroare, implementarea MQ JMS import şi export poate returna unul din cele doua tipuri de excepţii:

- Service Business Exception: această excepţie este returnată dacă fault specificată pe interfaţa serviciului business (WSDL port type) a survenit.
- Service Runtime Exception: apărută în toate celelalte cazuri. În cele mai multe cazuri, cause exception va conţine excepţia originală (JMSException).

  De exemplu, un import aşteaptă numai un mesaj de răspuns pentru fiecare mesaj de cerere. Dacă mai mult de un răspuns soseşte, sau dacă un răspuns întârziat (unul pentru care expirarea răspunsului SCA are loc) soseşte, o excepţie Service Runtime Exception este aruncată. Tranzacţia este repetată, şi mesajul de răspuns este retras din coadă sau este manevrat de către managerul de eveniment.

Mesajele SCA bazate pe WebSphere MQ JMS care nu apar în managerul de evenimente eşuate.

Dacă mesajele SCA îşi au originea într-un eşec de interacţiune WebSphere MQ JMS, ar trebui să vă aşteptaţi să găsiţi aceste mesaje în managerul de evenimente eşuate. Dacă astfel de mesaje nu apar în managerul de evenimente eşuate, asiguraţi-vă că valoarea proprietăţii maximului de reîncercări de pe portul ascultător de bază este egală sau mai mare decât **1**. Setarea acestei valori la **1** sau mai mult activează interacţiunea cu managerul de evenimente eşuate în timpul invocărilor SCA pentru legările MQ JMS.

Scenarii de folosire greşită: comparaţie cu legările WebSphere MQ

Legările WebSphere MQ JMS sunt proiectate să interacţioneze cu aplicaţiile JMS implementate în comparaţie cu WebSphere MQ, care expune mesajele în concordanţă cu modelul de mesaje JMS. Totuşi, importul şi exportul WebSphere MQ , sunt în principal proiectate să interacţioneze cu aplicaţiile native WebSphere MQ şi să expună întregul conţinut al corpului mesajului WebSphere MQ la mediere.

Următoarele scenarii ar trebui construite folosind legarea WebSphere MQ JMS, şi nu legarea WebSphere MQ:

- Invocarea bean-ului JMS controlat de mesaj (MDB) dintr-un modul SCA, unde MDB este implementat împotriva furnizorului WebSphere MQ JMS. Se utilizează un import WebSphere MQ JMS.
- Permisiunea modulului SCA de a fi apelat dintr-un servlet componentă Java EE sau EJB prin intermediul căii JMS. Se va folosi un export WebSphere MQ JMS.
- Medierea conţinutului unui JMS MapMessage, la traversare de-a lungul WebSphere MQ. Se va folosi un export sau import WebSphere MQ JMS împreună cu handler-ul de date sau legarea de date potrivită.

Există situaţii în care legarea WebSphere MQ şi legarea WebSphere MQ JMS se aşteaptă să interacţioneze. În particular, când se face o punte între aplicaţii Java EE şi non-Java EE WebSphere MQ, se va folosi export WebSphere MQ şi import WebSphere MQ JMS (sau invers) în concordanţă cu legările de date potrivite sau modulele de mediere potrivite (sau ambele).

Tratarea excepţiilor:

Modul în care este configurată legarea determină cum sunt tratate excepţiile ridicate de handler-ele de date sau legările de date. În plus, natura fluxului de mediere dictează comportamentul sistemului când este aruncată o astfel de excepţie.

Poate apărea o varietate de probleme când un handler de date sau o legare de date este apelat(ă) de legarea dumneavoastră. De exemplu, este posibil ca handler-ul de date să primească un mesaj care are date utile corupte, sau este posibil să încerce să citească un mesaj care are un format incorect.

Modul în care legarea dumneavoastră tratează o astfel de excepţie este determinat de cum implementaţi handler-ul de date sau legarea de date. Comportamentul recomandat este să vă proiectaţi legarea de date să arunce o **DataBindingException**.

Situaţia este similară pentru un handler de date. Din moment ce handler-ul de date este invocat de legarea de date, orice excepţie a handler-ului de date este înfăşurată într-o excepţie a legării de date. Prin urmare o **DataHandlerException** vă este raportată ca o **DataBindingException**.

Când orice excepţie runtime, inclusiv o excepţie **DataBindingException**, este aruncată:

- Dacă fluxul de mediere este configurat să fie tranzacţional, mesajul JMS este memorat în Managerul de evenimente eşuate implicit pentru reluare sau ştergere manuală.

  Notă: Puteţi modifica modul de recuperare de pe legare, astfel încât mesajul să fie derulat înapoi în loc să fie stocat în managerul de evenimente eşuate.
- Dacă fluxul de mediere nu este tranzacţional, excepţia este înregistrată în istoric şi mesajul este pierdut.

Situaţia este similară pentru un handler de date. Deoarece handler-ul de date este apelat de legarea de date, o excepţie a handler-ului de date este produsă înăuntrul unei excepţii a legării de date. Prin urmare, o **DataHandlerException** vă este raportată ca o **DataBindingException**.

Legări WebSphere MQ:

Legarea WebSphere MQ asigură conectivitatea Service Component Architecture (SCA) cu aplicaţiile WebSphere MQ.

Utilizaţi legările de export şi de import WebSphere MQ pentru integrare directă cu un sistem bazat pe WebSphere MQ din mediul dumneavoastră de server. Acest lucru elimină nevoia de a utiliza caracteristicile MQ Link sau Client Link ale Magistralei de Integrale a Serviciului.

Atunci când o componentă interacţionează cu un serviciu WebSphere MQ prin intermediul unui import, legarea de import WebSphere MQ utilizează o coadă către care vor fi trimise datele şi o coadă unde poate fi primit răspunsul.

Atunci când un modul SCA oferă un serviciu clienţilor WebSphere MQ, legarea de export WebSphere MQ foloseşte o coadă în care pot fi primite cererile şi către care pot fi trimise răspunsurile. Selectorul funcţiei asigură o mapare operaţiei în cadrul componentei ţintă care va fi invocată.

Conversia datelor de sarcină utilă către şi de la un mesaj MQ se realizează prin intermediul handler-ului de date al corpului MQ sau prin legarea datelor. Conversia datelor din antet către şi de la un mesaj MQ se realizează prin intermediul legării datelor din antetul MQ.

Pentru informaţii despre versiunile WebSphere MQ suportate, vedeţi pagina web cu cerinţe detaliate de sistem.

Privire generală asupra legărilor WebSphere MQ:

Legarea WebSphere MQ asigură integrarea în aplicaţiile native bazate pe MQ.

Taskuri administrative WebSphere MQ

Se aşteaptă ca administratorul de sistem al WebSphere MQ să creeze WebSphere MQ Queue Manager de bază; acesta va fi folosit de legările WebSphere MQ înainte de a rula o aplicaţie care conţine aceste legări.

Taskuri administrative WebSphere

Trebuie să setaţi proprietatea **Cale bibliotecă nativă** din adaptorul pentru resurse MQ în WebSphere cu versiunea WebSphere MQ suportată de server şi să reporniţi serverul. Acest lucru asigură faptul că sunt utilizate bibliotecile unei versiuni acceptate a WebSphere MQ. Cerinţe detaliate despre hardware şi software se pot găsi pe Paginile de suport IBM .

Legări de import WebSphere MQ

Legarea de import WebSphere MQ permite componentelor din modulul dumneavoastră SCA să comunice cu serviciile oferite de aplicaţiile externe bazate pe WebSphere MQ. Trebuie să utilizaţi o versiune suportată de WebSphere MQ. Cerinţe detaliate despre hardware şi software se pot găsi pe Paginile de suport IBM .

Interacţiunea cu sistemele WebSphere MQ externe include utilizarea cozilor pentru trimiterea cererilor sau primirea răspunsurilor.

Se oferă suport pentru două tipuri de scenarii de utilizare pentru legările de import WebSphere MQ, în funcţie de tipul de operaţie invocat:

- Într-o direcţie: Importul WebSphere MQ pune un mesaj pe coada configurată în câmpul **Coadă destinaţie trimitere** al legării de import. Nu se trimite nimic către câmpul replyTo din antetul MQMD.
- În ambele sensuri (cerere-răspuns): Importul WebSphere MQ pune un mesaj în coada configurată în câmpul **Coadă destinaţie de trimitere**

  Coada de primire este setată în câmpul replyTo din antetul MQMD. Un MDB (bean controlat de mesaj) este implementat pentru a asculta la coada de primire, iar în momentul în care este primit un răspuns, acesta transmite răspunsul înapoi către componentă.

  Legarea de import poate fi configurată (folosind câmpul **Schema de corelare a răspunsului**), astfel încât să aştepte ID-ul de corelare al mesajului pentru răspuns ce a fost copiat din ID-ul mesajului de cerere (cel implicit) sau din ID-ul de corelare al mesajului de cerere.

Este important de menţionat faptul că WebSphere MQ este o legare asincronă. Dacă o componentă apelantă invocă un import în mod sincron WebSphere MQ (pentru o operaţie bidirecţională), atunci aceasta este blocată până când răspunsul este returnat de serviciul WebSphere MQ.

Figura 39 ilustrează modul în care importul este legat de serviciul extern.

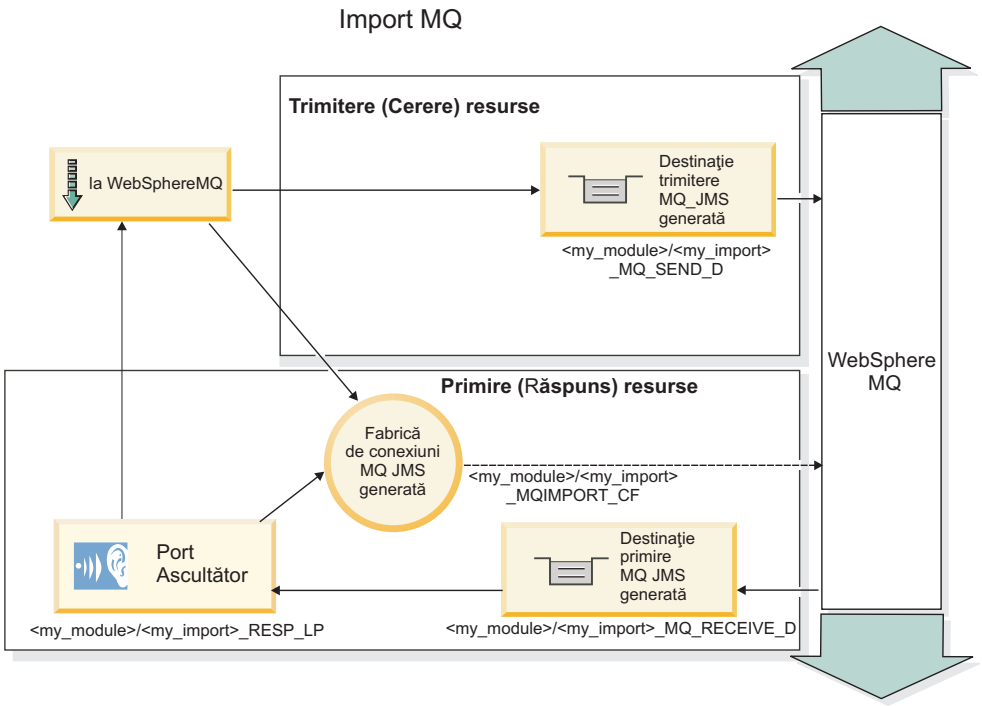


Figura 39. Resurse pentru legarea de import WebSphere MQ

Legări pentru export WebSphere MQ

Legarea de export WebSphere MQ pune la dispoziţie mijloacele prin care modulele SCA asigură servicii către aplicaţiile externe bazate pe WebSphere MQ.

Este implementat un MDB pentru a asculta cererile de intrare în **Coadă destinaţii de primire** specificată în legarea de export. Coada specificată în câmpul **Coadă destinaţii de trimitere** este folosit pentru a trimite răspunsul către cererea de intrare în cazul în care componenta invocată oferă un răspuns. Coada specificată în câmpul replyTo din mesajul de răspuns înlocuieşte coada specificată în câmpul **Coadă destinaţii de trimitere** field.

Figura 40 ilustrează modul în care solicitantul extern este legat de export.

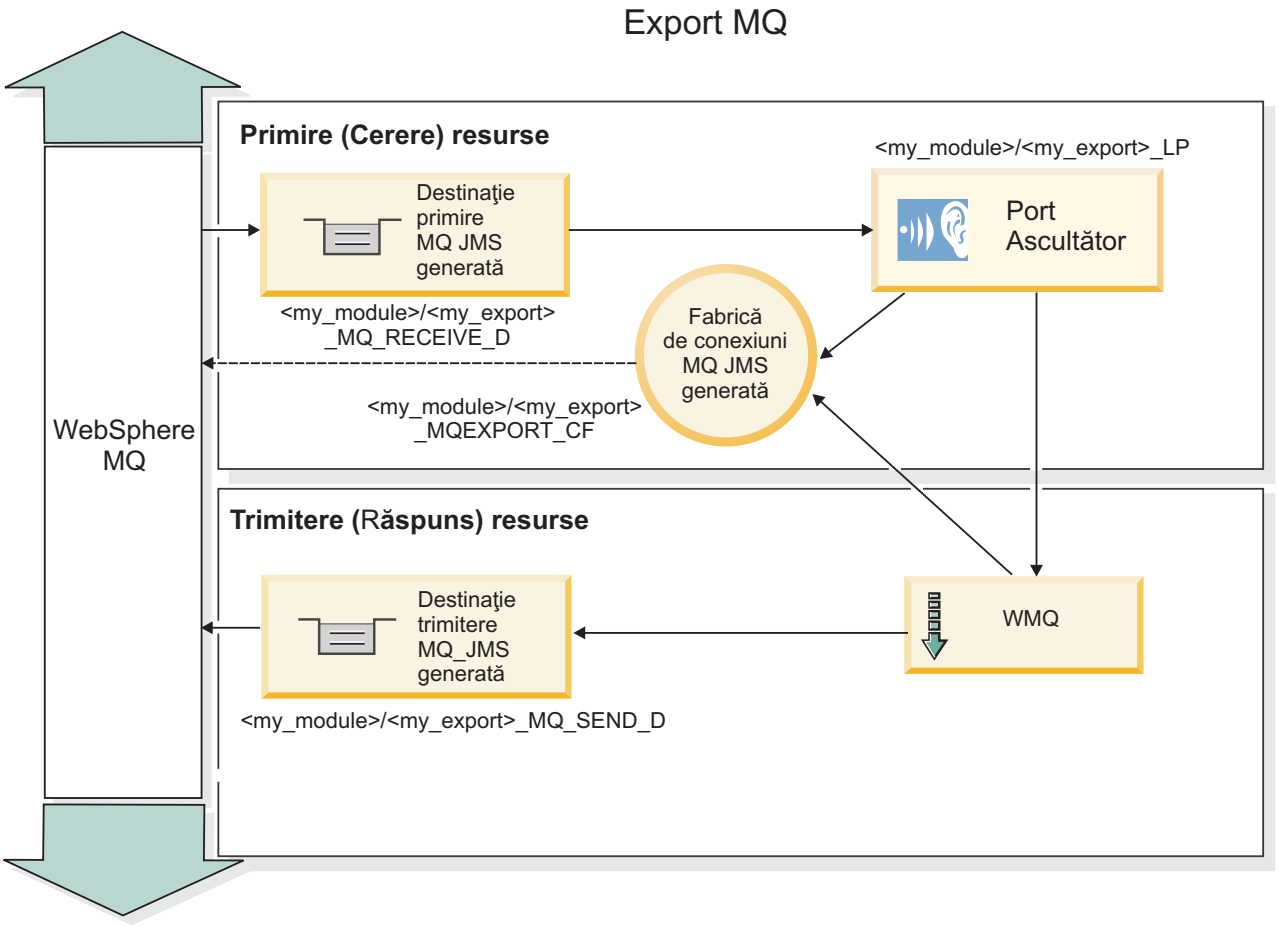


Figura 40. Resurse pentru legarea de export WebSphere MQ

Notă: Figura 39 la pagina 132 şi Figura 40 ilustrează modul în care o aplicaţie dintr-o versiune anterioară a IBM Business Process Manager este legată de un serviciu extern. Pentru aplicaţiile dezvoltate pentru IBM Business Process Manager Versiunea 7, se foloseşte Specificare activare în loc de Port ascultător şi Fabrică de conexiuni.

Caracteristici cheie ale unei legări WebSphere MQ:

Caracteristicile cheie ale unei legări WebSphere MQ includ anteturi, artefacte Java EE, şi resurse create Java EE.

Scheme de corelare

O aplicaţie de cerere/răspuns WebSphere MQ poate utiliza un număr de tehnici pentru a corela mesajele de răspuns cu cererile, construite în jurul câmpurilor MQMD MessageID şi CorrelID. În marea majoritate a cazurilor, solicitantul lasă

managerul de cozi să selecteze un MessageID şi aşteaptă ca aplicaţia de răspuns să copieze aceasta în CorrelID al răspunsului. În majoritatea cazurilor, solicitantul şi aplicaţia de răspuns ştiu implicit ce tehnică de corelare este în uz. Ocazional aplicaţia de răspuns va onora diverse steguleţe în câmpul Report al cererii ce descriu cum vor fi tratate aceste câmpuri.

Legările de export pentru mesajele WebSphere MQ pot fi configurate cu următoarele opţiuni:

Opţiuni răspuns MsgId:

MsgID nou
Permite managerului de coadă să selecteze un MsgId unic pentru răspuns (implicit).

Copiere de la MsgID cerere
Copiază câmpul MsgId de la câmpul MsgId din cerere.

Copiere de la mesaj SCA
Setează MsgId pentru a fi transportat în anteturile WebSphere MQ din mesajul de răspuns SCA, sau permite managerului de coadă să definească un nou Id dacă valoarea nu există.

Ca opţiuni de raport
Inspectează câmpul Raport al MQMD din cerere pentru o sugestie despre cum să fie tratat MsgId. Opţiunile MQRO_NEW_MSG_ID şi MQRO_PASS_MSG_ID sunt suportate şi se comportă ca New MsgId şi Copy from Request MsgID.

Opţiuni CorrelId răspuns:

Copy from Request MsgID
Copiază câmpul CorrelId de la câmpul MsgId din cerere (implicit).

Copy from Request CorrelID
Copiază câmpul CorrelId de la câmpul CorrelId din cerere.

Copy from SCA message
Setează CorrelId pentru a fi transportat în anteturile WebSphere MQ din mesajele de răspuns SCA sau îl lasă gol dacă valoarea nu există.

As Report Options
Inspectează câmpul Report al MQMD din cerere pentru o sugestie despre cum să fie tratat CorrelId. Opţiunile MQRO_COPY_MSG_ID_TO_CORREL_ID şi MQRO_PASS_CORREL_ID sunt suportate şi se comportă ca Copy fromRequest MsgID şi Copy from Request CorrelID.

Legările de import pentru mesajele WebSphere MQ pot fi configurate cu următoarele opţiuni:

Opţiuni MsgId cerere:

New MsgID
Permite managerului cozii să selecteze un MsgId unic pentru cerere (implicit).

Copy from SCA message
Setează MsgId pentru a fi transportat în anteturile WebSphere MQ din mesajul de cerere SCA sau permite managerului de coadă să definească un nou Id dacă valoarea nu există.

Opţiuni corelare răspuns:

Response has CorrelID copied from MsgId
Se aşteaptă ca mesajul de răspuns să aibă un câmp CorrelId setat, per MsgId al cererii (implicit).

Response has MsgID copied from MsgId
Se aşteaptă ca mesajul de răspuns să aibă un câmp MsgId setat, per MsgId al cererii.

Response has CorrelID copied from CorrelId
Se aşteaptă ca mesajul de răspuns să aibă un câmp CorrelId setat, per CorrelId al cererii.

Resurse Java EE

Un număr de resurse Java EE sunt create atunci când o legare WebSphere MQ este implementată într-un mediu Java EE.

Parametri

Fabrica de conexiuni MQ
Utilizată de clienţi pentru a crea o conexiune la furnizorul WebSphere MQ.

Fabrica de conexiuni răspuns
Utilizată de runtime-ul SCA MQ atunci când destinaţia de trimitere este pe un Manager de coadă diferit ca destinaţia de recepţionare.

Specificaţii de activare
O specificaţie de activare MQ JMS este asociată cu una sau mai multe bean-uri bazate pe mesaje şi furnizează configuraţia necesară pentru a recepţiona mesajele.

Destinaţii
- Destinaţie de trimitere: unde cererea sau mesajul de ieşire este trimis (import); unde mesajul de răspuns va fi trimis (export), dacă nu este înlocuit de câmpul de antet MQMD ReplyTo din mesajul de intrare.
- Destinaţie de primire: unde răspunsul/cererea sau mesajul de intrare ar trebui plasat.

Anteturi WebSphere MQ:

Anteturile WebSphere MQ încorporează anumite convenţii pentru conversie la mesaje SCA (Service Component Architecture).

Mesajele WebSphere MQ conţin un antet sistem (MQMD), zero sau mai multe alte anteturi MQ (sistem sau personalizate) şi un corp mesaj. Dacă există mai multe anteturi în mesaj, ordinea anteturilor este semnificativă.

Fiecare antet conţine informaţii care descriu structura antetului următor. MQMD-ul descrie primul antet.

Cum sunt parsate anteturile MQ

O legare de date de Antet MQ este utilizată pentru a parsa anteturi MQ. Următoarele anteturi sunt suportate automat:
- MQRFH
- MQRFH2
- MQCIH
- MQIIH

Anteturile care încep cu **MQH** sunt manipulate diferit. Anumite câmpuri ale antetului nu sunt parsate; ele rămân ca octeţi neparsaţi.

Pentru alte anteturi MQ, puteţi scrie legări de date antet MQ pentru a parsa acele anteturi.

Cum sunt accesate anteturile MQ

Anteturile MQ pot fi accesate din produs într-unul din două moduri:
- Prin SMO (Service Message Object) într-o mediere
- Prin API-ul ContextService

Anteturile MQ sunt reprezentate intern cu elementul SMO MQHeader. MQHeader este un container de date antet care extinde MQControl dar conţine un element valoare de orice tip. Conţine MQMD, MQControl (informaţii de control corp mesaj MQ) şi o listă de alte anteturi MQ.

- MQMD reprezintă conţinuturile descrierii mesajului WebSphere MQ, cu excepţia informaţiilor care determină structura şi codarea corpului.
- MQControl conţine informaţii care determină structura şi codarea unui corp de mesaj.
- MQHeaders conţine o listă de obiecte MQHeader.

Lanţul antet MQ este desfăcut astfel încât, în interiorul SMO, fiecare antet MQ are propriile sale informaţii de control (CCSID, Codare şi Format). Anteturile pot fi adăugate sau şterse cu uşurinţă, fără a modifica alte date ale antetului.

Setarea câmpurilor în MQMD

Puteţi actualiza MQMD-ul utilizând API-ul Context sau prin SMO (Service Message Object) într-o mediere. Câmpurile următoare sunt propagate automat la mesajul MQ de ieşire:

- Encoding
- CodedCharacterSet
- Format
- Report
- Expiry
- Feedback
- Priority
- Persistence
- CorrelId
- MsgFlags

Configuraţi legarea MQ pe un Import sau Export pentru a propaga următoarele proprietăţi la mesajul MQ de ieşire:

MsgID
Setaţi **ID mesaj de cerere** la copiere din mesaj SCA.

MsgType
Curăţaţi caseta de bifare **Setare tip mesaj la MQMT_DATAGRAM sau MQMT_REQUEST pentru operaţii cerere-răspuns**.

ReplyToQ
Curăţaţi caseta de bifare **Înlocuire răspuns la coadă de mesaje de cerere**.

ReplyToQMgr
Curăţaţi caseta de bifare **Înlocuire răspuns la coadă de mesaje de cerere**.

De la versiunea 7.0 înainte, câmpurile context pot fi înlocuite utilizând o proprietate personalizată de pe definiţia destinaţiei JNDI. Setaţi proprietatea personalizată MDCTX cu valoarea SET_IDENTITY_CONTEXT pe destinaţia de trimitere pentru a propaga câmpurile următoare la mesajul MQ de ieşire:

- UserIdentifier
- AppIdentityData

Setaţi proprietatea personalizată MDCTX cu valoarea SET_ALL_CONTEXT pe destinaţia de trimitere pentru a propaga proprietăţile următoare la mesajul MQ de ieşire:

- UserIdentifier
- AppIdentityData
- PutApplType
- PutApplName
- ApplOriginData

Unele câmpuri nu sunt la mesajul MQ de ieşire. Câmpurile următoare sunt înlocuite în timpul trimiterii mesajului:

- BackoutCount
- AccountingToken
- PutDate
- PutTime
- Offset
- OriginalLength

Adăugarea statică a MQCIH într-o îmbinare WebSphere MQ:

IBM Business Process Manager suportă adăugarea informaţiilor de antet MQCIH static, fără folosirea unui modul de mediere.

Există diferite moduri de a adăuga informaţii de antet MQCIH unui mesaj (de exemplu, folosind primitiva de mediere Header Setter). Poate fi util să adăugaţi aceste informaţii de antet static, fără folosirea unui modul de mediere suplimentar. Informaţii de antet statice, inclusiv numele de program CICS, ID-ul tranzacţiei şi alte detalii de antet de format de date, pot fi definite ca şi parte a legării WebSphere MQ.

WebSphere MQ, MQ CICS Bridge şi CICS trebuie să fie configurat pentru ca informaţiile de antet MQCIH să fie adăugate static.

Puteţi folosi Integration Designer pentru a configura importul WebSphere MQ cu valori statice care sunt necesare pentru informaţiile de antet MQCIH.

Atunci când un mesaj ajunge şi este procesat de către importul WebSphere MQ, este efectuată o verificare pentru vedeţi dacă informaţia de antet MQCIH este deja prezentă în mesaj. Dacă MQCIH este prezent, valorile statice definite în importul WebSphere MQ sunt folosite pentru a înlocui valorile dinamice corespunzătoare din mesaj. Dacă MQCIH nu este prezent, este creat unul în mesaj şi valorile statice definite în importul WebSphere MQ sunt adăugate.

Valorile statice definite în importul WebSphere MQ sunt specifice unei metode. Puteţi specifica valori statice MQCIH diferite pentru diferite metode din acelaşi import WebSphere MQ.

Această facilitate nu este folosită pentru a furniza valori implicite dacă MQCIH nu conţine informaţii de antet specifice deoarece o valoare statică definită în importul WebSphere MQ va înlocui o valoare corespunzătoare furnizată în mesajul de intrare.

Clienţi externi:

IBM Business Process Manager poate trimite mesaje către, sau să primească mesaje de la, clienţi externi folosind legări WebSphere MQ.

Un client extern (de exemplu, un portal web sau un sistem de informaţii pentru întreprindere) poate trimite un mesaj către o componentă SCA din aplicaţie prin intermediul unui export sau poate fi invocat de o componentă SCA din aplicaţie prin intermediul unui import.

Legarea de export WebSphere MQ implementează MDB-uri (bean-uri controlate de mesaj) cu scopul de a asculta cererile ce intră prin destinaţia receive specificată în legarea de export. Destinaţia specificată în câmpul trimite este folosită pentru a trimite răspunsul către cererea de intrare în cazul în care aplicaţia invocată oferă un răspuns. Cu toate acestea, un client extern poate invoca aplicaţii prin intermediul legării de export.

WebSphere MQ importă legarea la, şi poate trimite un mesaj către, clienţii externi. Acest mesaj ar putea sau nu să solicite un răspuns de la clientul extern.

Informaţii suplimentare despre modul de interacţionare cu clienţii externi folosind WebSphere MQ pot fi găsite în Centrul de informare WebSphere MQ.

Depanarea legărilor WebSphere MQ:

Puteţi diagnostica şi rezolva condiţiile de eşec ce apar în legările WebSphere MQ.

Condiţii de eşec primar

Condiţiile de eşec primar ale legărilor WebSphere MQ sunt determinate de către sensurile tranzacţionale, de către configuraţia WebSphere MQ sau prin referire la comportamente existente în alte componente. Condiţiile de eşec primar includ:

- Eşec în conexiunea la managerul de coadă sau la coada WebSphere MQ.

  Un eşec la conectarea la WebSphere MQ pentru a primi mesaje va avea ca rezultat eşecul în pornirea portului MDB Listener. Această condiţie va fi jurnalizată în istoricul WebSphere Application Server. Mesaje persistente vor rămâne pe coada WebSphere MQ până în momentul în care sunt extrase cu succes (sau expirate de către WebSphere MQ).

  Un eşec la conectarea la WebSphere MQ pentru a trimite mesaje de ieşire va duce la derularea înapoi a operaţiei de control a trimiterii.
- Eşecul în analiza unui mesaj de intrare sau în construirea unui mesaj de ieşire.

  Un eşec la legarea de date sau la handler-ul de date cauzează derularea înapoi a tranzacţiei care controlează lucrul.
- Eşec la trimiterea mesajului de ieşire.

  Un eşec la trimiterea unui mesaj cauzează derularea înapoi a tranzacţiei relevante.
- Mesaje de răspuns multiple sau neaşteptate.

  Importul aşteaptă numai un mesaj de răspuns pentru fiecare mesaj de cerere. Dacă mai mult de un răspuns soseşte, sau dacă un răspuns întârziat (unul pentru care expirarea răspunsului SCA are loc) soseşte, este aruncată o excepţie Service Runtime. Tranzacţia este repetată, şi mesajul de răspuns este retras din coadă sau este manevrat de către managerul de eveniment.

Utilizarea greşită a scenariilor: comparaţie cu legările WebSphere MQ JMS

WebSphere MQ import şi export sunt în principal proiectate să interacţioneze cu aplicaţiile native WebSphere MQ şi să expună întregul conţinut al corpului mesajului WebSphere MQ la medieri. Legările WebSphere MQ JMS sunt proiectate să interacţioneze cu aplicaţiile JMS implementate în comparaţie cu WebSphere MQ, care expune mesajele în concordanţă cu modelul de mesaje JMS.

Următoarele scenarii ar trebui construite folosind legarea WebSphere MQ JMS, şi nu legarea WebSphere MQ:

- Invocarea bean-ului JMS controlat de mesaj (MDB) dintr-un modul SCA, unde MDB este implementat împotriva furnizorului WebSphere MQ JMS. Se utilizează un import WebSphere MQ JMS.
- Permisiunea modulului SCA de a fi apelat dintr-un servlet componentă Java EE sau EJB prin intermediul căii JMS. Utilizaţi un export WebSphere MQ JMS.
- Medierea conţinutului unui JMS MapMessage, la traversare de-a lungul WebSphere MQ. Utilizaţi un export şi import WebSphere MQ JMS împreună cu legarea de date potrivită.

Există situaţii în care legarea WebSphere MQ şi legarea WebSphere MQ JMS se aşteaptă să interacţioneze. În particular, când se face o punte între aplicaţii Java EE şi non-Java EE WebSphere MQ, utilizaţi export WebSphere MQ şi import WebSphere MQ JMS (sau invers) în concordanţă cu legările de date corespunzătoare sau modulele de mediere potrivite (sau ambele).

Mesaje netrimise

Dacă WebSphere MQ nu poate livra un mesaj la destinaţia sa intenţionată (de exemplu, din cauza erorilor de configuraţie), trimite mesaje în schimb la o coadă de scrisori moarte.

Pentru realizarea acestui lucru, are nevoie de un antet cu mesaj nelivrat la începutul corpului mesajului. Acest antet conţine motivele de eşec, destinaţia originală şi alte informaţii.

Mesajele SCA bazate pe MQ nu apar în managerul de evenimente eşuate

Dacă mesajele SCA îşi au originea într-un eşec de interacţiune WebSphere, ar trebui să vă aşteptaţi să găsiţi aceste mesaje în managerul de evenimente eşuate. Dacă astfel de mesaje nu apar în managerul de evenimente eşuate, asiguraţi-vă că destinaţia WebSphere MQ ce stă la bază are valoarea maximă de livrări eşuate mai mare decât 1. Setarea acestei valori la 2 sau mai mult permite interacţiunea cu managerul de evenimente eşuate în timpul invocărilor SCA pentru legările WebSphere MQ.

Evenimentele eşuate MQ sunt reluate pe managerul de coadă greşit

Atunci când o fabrică de conexiuni predefinită trebuie să fie utilizată pentru conexiuni de ieşire, proprietăţile de conexiune trebuie să se potrivească cu acelea definite în specificaţia de activare utilizată pentru conexiunile de intrare.

Fabrica de conexiuni predefinită este utilizată pentru a crea o conexiune la reluarea unui eveniment eşuat şi, prin urmare, trebuie configurată pentru a utiliza acelaşi manager de coadă ca cel din care a fost primit iniţial mesajul.

Tratarea excepţiilor:

Modul în care este configurată legarea determină cum sunt tratate excepţiile ridicate de handler-ele de date sau legările de date. În plus, natura fluxului de mediere dictează comportamentul sistemului când este aruncată o astfel de excepţie.

Poate apărea o varietate de probleme când un handler de date sau o legare de date este apelat(ă) de legarea dumneavoastră. De exemplu, este posibil ca handler-ul de date să primească un mesaj care are date utile corupte, sau este posibil să încerce să citească un mesaj care are un format incorect.

Modul în care legarea dumneavoastră tratează o astfel de excepţie este determinat de cum implementaţi handler-ul de date sau legarea de date. Comportamentul recomandat este să vă proiectaţi legarea de date să arunce o **DataBindingException**.

Situaţia este similară pentru un handler de date. Din moment ce handler-ul de date este invocat de legarea de date, orice excepţie a handler-ului de date este înfăşurată într-o excepţie a legării de date. Prin urmare o **DataHandlerException** vă este raportată ca o **DataBindingException**.

Când orice excepţie runtime, inclusiv o excepţie **DataBindingException**, este aruncată:

- Dacă fluxul de mediere este configurat să fie tranzacţional, mesajul JMS este memorat în Managerul de evenimente eşuate implicit pentru reluare sau ştergere manuală.

  Notă: Puteţi modifica modul de recuperare de pe legare, astfel încât mesajul să fie derulat înapoi în loc să fie stocat în managerul de evenimente eşuate.
- Dacă fluxul de mediere nu este tranzacţional, excepţia este înregistrată în istoric şi mesajul este pierdut.

Situaţia este similară pentru un handler de date. Deoarece handler-ul de date este apelat de legarea de date, o excepţie a handler-ului de date este produsă înăuntrul unei excepţii a legării de date. Prin urmare, o **DataHandlerException** vă este raportată ca o **DataBindingException**.

Limitări ale legărilor:

Legările au unele limitări în utilizarea lor, care sunt menţionate aici.

Limitările legării MQ:

Legarea MQ are unele limitări în utilizarea sa. Acestea sunt menţionate aici.

Distribuţia mesajelor din abonamentele nepublicate

Metoda de abonare fără publicare a mesajelor distribuite nu este suportată în prezent de legarea MQ prin intermediul propriului WMQ. Totuşi, legarea MQ JMS nu suportă această metodă de distribuţie.

Cozi de primire partajate

Mai multe legări de export şi import WebSphere MQ aşteaptă ca orice mesaj prezent în coada lor configurată pentru recepţie să fie intenţionate pentru acel export sau import. Legările pentru importuri şi exporturi ar trebui să fie configurate ţinând cont de următoarele considerente:

- Fiecare import MQ trebuie să aibă o coadă diferită de recepţionare deoarece legarea de import MQ presupune că toate mesajele din coada de recepţionare sunt răspunsuri la cererile pe care le trimite. În cazul în care coada de recepţionare este partajată între mai multe importuri, răspunsurile ar putea fi recepţionate de un import greşit şi nu vor putea fi corelate cu mesajul de cerere iniţial.
- Fiecare export MQ ar trebui să aibă o coadă diferită de recepţionare, deoarece altfel nu se poate anticipa care dintre exporturi va primi oricare dintre mesajele de cerere individuale.
- Importurile şi exporturile MQ pot face referire către aceiaşi coadă de trimitere.

Limitările legărilor JMS, MQ JMS şi JMS generice:

Legările JMS şi MQ JMS au unele limitări.

Implicaţiile generării legărilor implicite

Limitările utilizării legărilor JMS, MQ JMS şi JMS sunt discutate în următoarele secţiuni:

- Implicaţiile generării legărilor implicite
- Schema de corelare a răspunsurilor
- Suport bidirecţional

Atunci când generaţi o legare, vor fi completate în mod implicit mai multe câmpuri în cazul în care nu alegeţi să introduceţi singur valorile. De exemplu, numele unui fabrici de conexiuni va fi creat pentru dvs. Dacă ştiţi că veţi pune aplicaţia dumneavoastră pe un server şi o veţi accesa de la distanţă cu un client, ar trebui ca în momentul creării legării să introduceţi numele JNDI în locul celor implicite, deoarece probabil veţi dori să controlaţi aceste valori prin intermediul consolei administrative în momentul rulării.

Totuşi, dacă aţi acceptat valorile implicite, iar apoi aţi aflat mai târziu că nu vă puteţi accesa aplicaţia de la un client aflat la distanţă, aveţi posibilitatea să utilizaţi consola administrativă pentru a seta în mod explicit valoarea fabricii de conexiuni. Localizaţi câmpul pentru punctele finale care aparţin de furnizor în setările fabricii de conexiuni şi adăugaţi valoarea în forma <nume_server>:7276 (dacă se utilizează numărul de port implicit).

Schema de corelare a răspunsurilor

În cazul în care folosiţi schema de corelare a răspunsurilor CorrelationId To CorrelationId, care este folosită pentru a corela mesaje într-o operaţie de tip răspuns-răspuns, trebuie să aveţi un Id de corelare dinamic în mesaj.

Pentru a crea un ID de corelare dinamic într-un modul de mediere folosind editorul de fluxul de mediere, adăugaţi o primitivă de mediere Mapping înainte de importul cu legarea JMS. Deschideţi editorul de mapare. Anteturile arhitecturii componentei serviciului cunoscut vor fi disponibile în mesajul ţintă. Trageţi un câmp care conţine un ID unic în mesajul sursă în ID-ul de corelare din antetul JMS din mesajul ţintă.

Suportul bidirecţional

Doar caracterele ASCII sunt acceptate pentru numele JNDI (Java Naming and Directory Interface) la momentul execuţiei.

Cozi de primire partajate

Mai multe legări de export şi import aşteaptă ca orice mesaj prezent în coada lor configurată pentru recepţie să fie intenţionate pentru acel export sau import. Legările pentru importuri şi exporturi ar trebui să fie configurate ţinând cont de următoarele considerente:

- Fiecare legare pentru import trebuie să aibă o coadă diferită de recepţionare deoarece aceasta presupune că toate mesajele din coada de recepţionare sunt răspunsuri la cererile pe care le trimite. În cazul în care coada de recepţionare este partajată între mai multe importuri, răspunsurile ar putea fi recepţionate de un import greşit şi nu vor putea fi corelate cu mesajul de cerere iniţial.
- Fiecare legare de export ar trebui să aibă o coadă diferită de primire, deoarece altfel nu se poate anticipa care dintre exporturi va primi oricare dintre mesajele de cerere individuale.
- Importurile şi exporturile pot face referire la aceeaşi coadă de trimitere.

Obiectele business

Industria software-ului pentru computer a dezvoltat mai multe modele de programare şi cadre de lucru în care *obiectele business* furnizează o reprezentare naturală a datelor operaţionale pentru procesarea aplicaţiei.

În general, aceste obiecte business:

- Sunt definite utilizând standarde de industrie
- Mapează transparent date la tabele de baze de date sau sisteme de informaţii de întreprindere
- Suportă protocoale de invocare la distanţă
- Furnizează fundaţia modelului de programare de date pentru programare aplicaţie

Process Designer şi Integration Designer le furnizează dezvoltatorilor un astfel de model de obiect business comun pentru reprezentarea diferitelor tipuri de entităţi de afaceri din diferite domenii. În timpul dezvoltării, acest model permite dezvoltatorilor să definească obiecte business ca definiţii de scheme XML.

În timpul rulării, datele operaţionale definite de definiţiile de scheme XML sunt reprezentate ca obiecte business Java. În acest model, obiectele business sunt bazate pe ciorne anterioare ale specificaţiei SDO (Service Data Object) şi furnizează setul complet de interfeţe aplicaţie model de programare necesare pentru manipularea datelor operaţionale.

Definirea obiectelor business

Definiţi obiecte business utilizând editorul de obiecte business din Integration Designer. Editorul de obiecte business memorează obiectele business ca definiţii schemă XML.

Utilizarea schemei XML pentru a defini obiecte business furnizează mai multe avantaje:

- Schema XML furnizează un model de definiţie de date bazat pe standarde şi o fundaţie pentru interoperabilitatea dintre aplicaţii şi sisteme eterogene incompatibile. Schemele XML sunt utilizate împreună cu WSDL (Web Services Description Language) pentru a furniza contracte de interfaţă bazate pe standarde de-a lungul componentelor, aplicaţiilor şi sistemelor.
- Schemele XML definesc un model de definiţie date bogat pentru reprezentarea datelor operaţionale. Acest model include tipuri complexe, tipuri simple, tipuri definite de utilizator, moştenire tip şi cardinalitate printre alte caracteristici.
- Obiectele business pot fi definite de interfeţe operaţionale şi date definite din Web Services Description Language, precum şi de schema XML din organizaţii de standarde industriale sau din alte sisteme şi aplicaţii. Integration Designer poate importa aceste obiecte business direct.

Integration Designer furnizează de asemenea suport pentru descoperirea datelor operaţionale din baze de date şi sisteme de informaţii de întreprindere şi generarea definiţiei obiectului business al schemei XML bazată pe standarde a acelor date operaţionale. Obiectele business generate în acest fel sunt adesea adresate ca *obiecte business specifice aplicaţiei* deoarece imită structura datelor operaţionale definite în sistemul de informaţii de întreprindere.

Când un proces manipulează date din mai multe sisteme de informaţii diferite, poate fi preţios să transformaţi reprezentarea disparată a datelor operaţionale (de exemplu, CustomerEIS1 şi CustomerEIS2 sau OrderEIS1 şi OrderEIS2) într-o singură reprezentare canonică (de exemplu, Customer sau Order). Reprezentarea canonică este adesea adresată ca *obiectul business generic*.

Definiţiile de obiecte business, în special pentru obiecte business generice, sunt utilizate frecvent de mai multe aplicaţii. Pentru a susţine această reutilizare, Integration Designer permite ca obiectele business să fie create în biblioteci care pot fi apoi asociate cu module de aplicaţii multiple.

Web Services Description Language (WSDL) defineşte contractele pentru serviciile furnizate şi consumate de un modul de aplicaţii SCA (Service Component Architecture) precum şi contractele utilizate pentru a crea componentele dintr-un modul de aplicaţii. Într-un contract, un WSDL poate reprezenta atât operaţii cât şi obiecte operaţionale (care sunt definite de scheme XML pentru a reprezenta datele operaţionale).

Lucrul cu obiecte business

SCA (Service Component Architecture) furnizează cadrul de lucru pentru definirea unui modul de aplicaţii, serviciile pe care le furnizează, serviciile pe care le consumă şi alcătuirea componentelor care furnizează logica operaţională a modulului de aplicaţii. Obiectele business joacă un rol important în aplicaţie, definind datele operaţionale care sunt utilizate pentru a descrie contractele componentei şi datele operaţionale pe care le manipulează componentele.

Diagrama următoare descrie un modul de aplicaţii SCA şi ilustrează multe dintre locurile în care dezvoltatorul lucrează cu obiecte business.

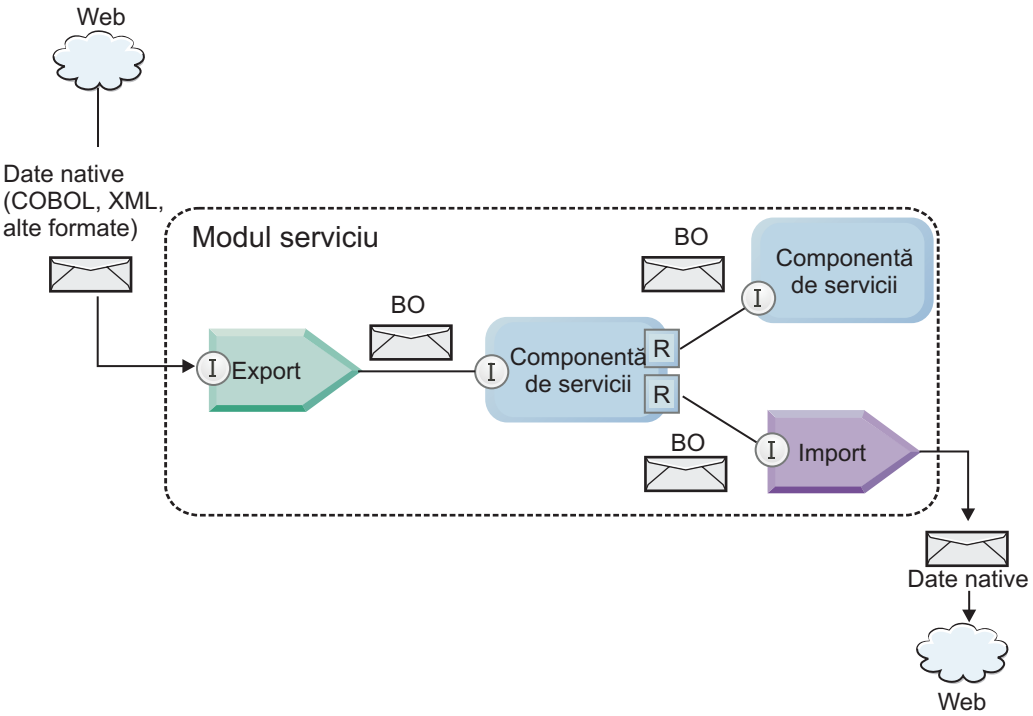


Figura 41. Obiectele business reprezintă datele care curg între servicii într-o aplicaţie

Notă: Acest subiect descrie cum sunt utilizate obiectele business de către modulele de aplicaţii SCA. Dacă utilizaţi interfeţe Java, modulele de aplicaţii SCA pot procesa de asemenea obiecte Java.

Model de programare obiect business

Modelul de programare obiect business conţine un set de interfeţe Java care reprezintă:

- Definiţia obiectului business şi date instanţă

- Un set de servicii care suportă operaţiile de pe obiectele business

Definiţiile tipului de obiect business sunt reprezentate de interfeţele commonj.sdo.Type şi commonj.sdo.Property. Modelul de programare obiect business furnizează un set de reguli pentru maparea informaţiilor de tip complex ale schemei XML la interfaţa Tip şi fiecare dintre elementele din definiţia tipului complex la interfaţa Proprietate.

Instanţele obiectului business sunt reprezentate de interfaţa commonj.sdo.DataObject. Modelul de programare obiect business nu are tip, ceea ce înseamnă că aceeaşi interfaţă commonj.sdo.DataObject poate fi utilizată pentru a reprezenta diferite definiţii de obiect business, cum ar fi Client sau Comandă. Definiţia căror proprietăţi pot fi setate şi extrase din fiecare obiect business este determinată de informaţiile de tip definite în schema XML asociată cu fiecare obiect business.

Comportamentul modelului de programare obiect business este bazat pe specificaţia Service Data Object 2.1. Pentru informaţii suplimentare, consultaţi SDO 2.1 pentru specificaţiile Java, îndrumare şi javadocs pe web: http://osoa.org/display/Main/Service+Data+Objects+Specifications.

Serviciile obiectului business suportă diverse operaţii de ciclu de viaţă (cum ar fi creare, egalitate, parsare şi serializare) pe obiecte business.

Pentru lucruri specifice despre modelul de programare al obiectului business, vedeţi Programare utilizând servicii de obiecte business şi Documentaţie API şi SPI generată despre obiecte business.

Legări, legări de date şi handler-e de date

După cum a fost afişat în Figura 41 la pagina 142, datele operaţionale care sunt utilizate pentru a invoca servicii furnizate de module de aplicaţii SCA sunt transformate în obiecte business, astfel încât componentele SCA să poate manipula datele operaţionale. În mod similar, obiectele business manipulate de componente SCA sunt convertite în formatul de date cerut de serviciile externe.

În unele cazuri, cum ar fi legarea de servicii Web, legarea utilizată pentru a exporta şi importa servicii transformă automat datele în formatul corespunzător. În alte cazuri, cum ar fi legarea JMS, dezvoltatorii pot furniza o legare de date sau handler de date care converteşte formatele nenative în obiecte business reprezentate de interfaţa DataObject.

Pentru informaţii suplimentare despre dezvoltarea legărilor de date şi handler-elor de date, consultaţi “Handler-e de date” la pagina 55 şi “Legări de date” la pagina 56.

Componente

Componentele SCA definesc contractele lor de servicii de provizionare şi consum utilizând o combinaţie de Web Services Description Language şi schemă XML. Datele operaţionale pe care SCA le transmite între componente sunt reprezentate ca obiecte business utilizând interfaţa DataObject. SCA verifică dacă aceste tipuri de obiecte business sunt compatibile cu contractul interfeţei definit de componenta care va fi invocată.

Teoriile modelului de programare pentru manipularea obiectelor business variază de la componentă la componentă. Componenta POJO şi primitiva personalizată a componentei flux de mediere furnizează manipulare directă a obiectelor business prin activarea programării Java direct utilizând interfeţele şi serviciile de programare obiect business. Majoritatea componentelor furnizează teorii de nivel superior pentru manipularea obiectelor business, dar furnizează de asemenea fragmente de cod Java pentru definirea comportamentului personalizat din interfeţele şi serviciile obiectului business.

Obiectele business pot fi transformate utilizând fie combinaţia componentei Interface Flow Mediation şi Business Object Map, fie componenta flux de mediere şi primitiva sa XML Map. Aceste capabilităţi de transformare obiecte business sunt utile pentru convertirea obiectelor business specifice aplicaţiei la şi de la obiecte business generice.

Obiecte business speciale

Obiectele de mesaje de servicii şi graficele operaţionale sunt două tipuri specializate de obiecte business care sunt utilizate în scopuri de aplicaţii specifice.

Obiect mesaj serviciu

Un SMO (Service Message Object) este un obiect business specializat care este utilizat de componentele fluxului de mediere pentru a reprezenta colecţia de date asociată cu o invocare de servicii.

Un SMO are o structură de nivel înalt corectată alcătuită din anteturi, context, corp şi ataşamente (dacă sunt prezente).

- Anteturile transmit informaţii înrudite cu invocarea de servicii peste un anumit protocol sau legare. Exemple sunt anteturile SOAP şi anteturile JMS.
- Datele de context transmit informaţii logice suplimentare asociate cu invocarea în timp ce sunt procesate de componenta fluxului de mediere. Aceste informaţii nu fac parte în mod tipic din datele de aplicaţie trimise sau recepţionate de clienţi.
- Corpul SMO-ului transmite datele operaţionale ale datelor utile, care reprezintă mesajul de aplicaţie nucleu sau datele de invocare sub forma unui obiect business standard.

SMO poate transporta de asemenea date în ataşament pentru invocările serviciilor web folosind SOAP cu ataşamente.

Fluxurile de mediere realizează astfel de taskuri precum dirijare cerere şi transformare de date, şi SMO-ul furnizează vizualizarea combinată de conţinut antet şi date utile (payload) într-o singură structură unificată.

Grafic operaţional

Un grafic operaţional este un obiect business special utilizat pentru a furniza suport pentru sincronizare date în scenarii de integrare.

Consideraţi un exemplu în care două sisteme de informaţii de întreprindere au o reprezentare a unei anumite comenzi. Când se modifică comanda dintr-un sistem, poate fi trimis un mesaj la celălalt sistem pentru a sincroniza datele comenzii. Graficele operaţionale suportă noţiunea de trimitere doar a porţiunii de comandă care s-a modificat la celălalt sistem şi adnotarea ei cu informaţii de modificare sumar pentru a defini tipul de modificare.

În acest exemplu, un Grafic operaţional de comenzi va transmite celuilalt sistem că unul dintre articolele de linie din comandă a fost şters şi că proprietatea dată de livrare proiectată a comenzii a fost actualizată.

Graficele operaţionale pot fi adăugate cu uşurinţă la obiecte business existente din Integration Designer. Sunt găsite de cele mai multe ori în scenarii în care sunt utilizate adaptoarele WebSphere şi pentru a ajuta migrarea aplicaţiilor WebSphere InterChange Server.

Mod de parsare obiect business

Integration Designer furnizează o proprietate pe module şi biblioteci pe care le puteţi vedea pentru a configura modul de parsare XML pentru obiecte business fie la vioi, fie la leneş.

- Dacă opţiunea este setată la *vioi*, fluxurile de octeţi XML sunt parsate cu nerăbdare pentru a crea obiectul business.
- Dacă opţiunea este setată la *leneş*, obiectul business este creat normal, dar parsarea reală a fluxului de octeţi XML este amânată şi parsată parţial doar când sunt accesate proprietăţile obiectului business.

În oricare dintre modurile de parsare XML, datele non-XML sunt întotdeauna parsate vioi pentru a crea obiectul business.

Considerente la alegerea modului de parsare a obiectului business:

Modul de parsare a obiectului business determină modul în care sunt parsate datele XML în timpul rulării. Un mod de parsare de obiect business este definit pe un modul sau o bibliotecă atunci când este creat(ă). Puteţi modifica modul de parsare pentru modul sau bibliotecă, totuşi ar trebui să fiţi conştient de implicaţii.

Modul de parsare pentru obiectul business este setat la nivelul modulului sau al bibliotecii. Modulele care au fost create într-o versiune de IBM Integration Designer anterioară versiunii 7 vor rula în modul de parsare rapidă fără modificări necesare. Implicit, modulele şi bibliotecile care sunt create în IBM Integration Designer versiunea 7 şi ulterioare vor primi cel mai adecvat mod de parsare în funcţie de numărul de factori, cum ar fi modul de parsare al proiectelor existente în spaţiul dumneavoastră de lucru sau modul de parsare al proiectelor dependente sau al altor proiecte din aceeaşi soluţie şi aşa mai departe. Puteţi modifica modul de parsare de obiect business al unui mod sau bibliotecă pentru a se potrivi cu implementarea dumneavoastră, totuşi ar trebui să fiţi conştienţi de următoarele considerente.

Considerente

- Modul de parsare obiect business leneş procesează datele XML mai repede; totuşi există diferenţe de compatibilitate între modul vioi şi modul leneş de care trebuie să fiţi conştient înainte de modificarea configuraţiei unui modul sau a unei biblioteci. Aceste diferenţe vor afecta comportamentul din timpul rulării modulelor. Pentru informaţii despre modul optim de parsare pentru aplicaţia dumneavoastră, vedeţi "Beneficii ale utilizării modului de parsare lent versus modul de parsare vioi" în legăturile înrudite.
- Un modul poate fi configurat doar pentru rularea într-un mod de parsare. Bibliotecile pot fi configurate fie pentru suportul nodurilor de parsare, fie a ambelor moduri de parsare. O bibliotecă configurată pentru suportul ambelor moduri de parsare ar putea fi referită de ambele module, utilizând modul de parsare vioi, şi de un modul utilizând modul de parsare leneş. Modul de parsare al unei biblioteci la momentul rulării este determinat de modulele care fac referire la bibliotecă. La momentul execuţiei, un modul îşi declară modul de parsare, iar acel mod de parsare este utilizat de modul şi de orice bibliotecă folosită de modul.
- Modulele şi bibliotecile care sunt configurate pentru moduri de parsare diferite sunt compatibile în următoarele cazuri:
  - Modulele şi bibliotecile configurate cu modul de parsare leneş sunt compatibile cu bibliotecile care utilizează fie modul de parsare leneş, fie ambele moduri de parsare, leneş şi vioi.
  - Modulele şi bibliotecile configurate cu modul de parsare vioi sunt compatibile cu bibliotecile care utilizează fie modul de parsare vioi, fie ambele moduri de parsare, leneş şi vioi.
  - Bibliotecile configurate cu modurile de parsare leneş şi vioi sunt compatibile doar cu bibliotecile care utilizează ambele moduri de parsare, leneş şi vioi.
- Utilizaţi acelaşi mod de parsare pentru modulele interactive care comunică utilizând legarea SCA. Dacă modulele comunică utilizând diferite moduri de parsare, ar putea rezulta probleme de performanţă.

Concepte înrudite:

"Beneficii ale utilizării modului de parsare leneş faţă de cel vioi"
Unele aplicaţii beneficiază de modul leneş de parsare XML în timp ce altele văd o performanţă îmbunătăţită cu mod de parsarea vioi. Este recomandat să măsuraţi aplicaţia dumneavoastră în ambele moduri de parsare pentru a determina ce mod se potriveşte cel mai bine caracteristicilor specifice ale aplicaţiei dumneavoastră.

Beneficii ale utilizării modului de parsare leneş faţă de cel vioi:

Unele aplicaţii beneficiază de modul leneş de parsare XML în timp ce altele văd o performanţă îmbunătăţită cu mod de parsarea vioi. Este recomandat să măsuraţi aplicaţia dumneavoastră în ambele moduri de parsare pentru a determina ce mod se potriveşte cel mai bine caracteristicilor specifice ale aplicaţiei dumneavoastră.

Este posibil ca aplicaţiile care parsează fluxuri de date XML mari să vadă îmbunătăţiri de performanţă când este utilizat modul de parsare XML leneş. Beneficiile performanţei cresc pe măsură ce dimensiunea şirului de octeţi XML creşteţi cantitatea de date de la şirul de octeţi care este accesat de către aplicaţie scade.

Este posibil ca aplicaţiile următoare să funcţioneze mai bine folosind un mod de parsarea vioi:

- Aplicaţii care parsează fluxuri de date non-XML
- Aplicaţii care folosesc mesajele care sunt create folosind serviciul BOFactory
- Aplicaţii care parsează mesaje XML foarte mici

Referinţe înrudite:

“Considerente la alegerea modului de parsare a obiectului business” la pagina 144
Modul de parsare a obiectului business determină modul în care sunt parsate datele XML în timpul rulării. Un mod de parsare de obiect business este definit pe un modul sau o bibliotecă atunci când este creat(ă). Puteţi modifica modul de parsare pentru modul sau bibliotecă, totuşi ar trebui să fiţi conştient de implicaţii.

Consideraţii de migrare şi de dezvoltare de aplicaţii:

Dacă configuraţi o aplicaţie care a fost dezvoltată iniţial utilizând un mod de parasare vioi pentru a utiliza acum un mod de parsare leneş, sau dacă plănuiţi să comutaţi o aplicaţie între modul de parsare leneş şi cel vioi, fiţi conştient de diferenţele dintre moduri şi de considerente când comutaţi moduri.

Tratarea erorilor

Dacă fluxul de octeţi XML care este parsat este format greşit, apar excepţii de parsare.

- În modul de parsare XML vioi, acele excepţii apar de îndată ce este parsat obiectul business din fluxul XML de intrare.
- Dacă este configurat modul de parsare XML leneş, excepţiile de parsare apar cu întârziere când sunt accesate proprietăţile obiectului business şi este parsată porţiunea XML-ulu icare este format greşit.

Pentru a vă ocupa de XML format greşit, selectaţi una dintre următoarele opţiuni:

- Implementarea unei magistrale de servicii de întreprindere pe margini pentru a valida XML de intrare
- Logica de detecţie eroare autor în punctul în care sunt accesate proprietăţile de obiect business

Mesaje şi stive de excepţii

Deoarece modurile de parsare XML leneş şi vioi au implementări fundamentale diferite, urmele de stivă aruncate de interfeţele şi serviciile de programare ale obiectului business au acelaşi nume de clasă de excepţii,, dar ar putea să nu conţină acelaşi mesaj de excepţie sau set înfăşurat de clase de excepţii specifice implementării.

Format serializare XML

Modul de parsare XML leneş furnizează o optimizare de performanţă care încearcă să copieze XML nemodificat de la fluxul de octeţi de intrare la fluxul de octeţi de ieşire pe serializare. Rezultatul este performanţă crescută, dar formatul de serializare al şirului de octeţi XML de ieşire poate fi diferit dacă întregul obiect business a fost actualizat în mod de parsare XML leneş sau dacă rulează în mod de parsare XML vioi.

Deşi formatul de serializare XML ar putea să nu fie întocmai echivalent din punct de vedere sintactic, valoarea semantică furnizată de obiectul business este echivalentă independent de modurile de parsare şi XML poate fi transmis în siguranţă între aplicaţii care rulează în moduri de parsare diferite cu echivalenţă semantică.

Validator instanţă obiect business

Validatorul instanţei modului obiectului business de parsare XML leneş furnizează o validare de fidelitate mai înaltă a obiectelor business, în special validare faţetă de valori proprietate. Din cauza acestor îmbunătăţiri, validatorul instanţei modului de parsare leneş prinde probleme suplimentare care nu sunt prinse în modul de parsare vioi şi furnizează mesaje de eroare mai detaliate.

Mapări XML versiune 602

Fluxurile de mediere dezvoltate iniţial înainte de WebSphere Integration Developer Version 6.1 pot conţine primitive Mapping care utilizează o hartă sau o foaie de stil care nu poate rula direct în modul de parsare XML mai lent. La migrarea unei aplicaţii pentru utilizarea în modul leneş de parsare XML, fişierele hartă asociate cu primitivele Mapping pot fi actualizate în mod automat de către vrăjitorul de migrare pentru a rula în noul mod. Totuşi, în cazul în care o primitivă Mapping se referă în mod direct la o foaie de stil care a fost editată manual, foaia de stil nu este migrată şi nu poate rula in modul XML mai lent.

API-uri nepublicate private

Dacă o aplicaţie profită de interfeţe de programare obiecte business specifice implementării, private, nepublicate, aplicaţia are toate şansele să eşueze compilarea când este schimbat modul de parsare. În mod de parsare vioi, aceste interfeţe private sunt tipic clase de implementare obiect business definite de EMF (Eclipse Modeling Framework).

În toate cazurile, este recomandat cu tărie ca API-urile private să fie înlăturate din aplicaţie.

API-uri EMF Obiect mesaj serviciu

O componentă de mediere din IBM Integration Designer furnizează abilitatea de a manipula conţinut mesaj utilizând clase şi interfeţe Java furnizate în pachetul com.ibm.websphere.sibx.smobo. În mod de parsare XML leneş, interfeţele Java din pachetul com.ibm.websphere.sibx.smobo pot fi încă utilizate, dar metodele care se referă direct la clasele şi interfeţele EMF (Eclipse Modeling Framework) sau care sunt moştenite de la interfeţele EMF au toate şansele să eşueze.

ServiceMessageObject şi conţinuturile sale nu pot fi atribuite obiectelor EMF în mod de parsare XML leneş.

Serviciu BOMode

Serviciul BOMode este utilizat pentru a determina dacă modul de parsare XML care se execută momentan este vioi sau leneş.

Migrare

Toate aplicaţiile dinainte de versiunea 7.0.0.0 rulează în mod de parsare XML vioi. Când sunt migrate în timpul rulării utilizând uneltele de migrare runtime BPM, continuă să ruleze în mod de parsare XML vioi.

Pentru a permite unei aplicaţii anterioare versiunii 7.0.0.0 să fie configurată pentru a utiliza modul de parsare XML, utilizaţi întâi Integration Designer pentru a migra artefactele aplicaţiei. După migrare, configuraţi aplicaţia pentru a utiliza parsare XML leneşă.

Vedeţi Migrare artefacte sursă pentru informaţii despre migrarea artefactelor din Integration Designer şi vedeţi Configurarea modului de parsare obiect business al modulelor şi bibliotecilor pentru informaţii despre setarea modului de parsare.

Relaţii

O relaţie este o asociere între două sau mai multe entităţi de date, în special obiecte business. În IBM Business Process Manager Advanced, relaţiile pot fi utilizate pentru a transforma date care sunt echivalente peste obiecte operaţionale şi alte date care sunt reprezentate diferit, sau pe care le pot utiliza pentru a trasa aplicaţii peste diferite obiecte găsite în aplicaţii diferite. Acestea pot fi partajate între aplicaţii, soluţii şi chiar produse.

Serviciul de relaţii din IBM Business Process Manager Advanced furnizează infrastructura şi operaţiile pentru gestionarea relaţiilor. Deoarece vă permite să trataţi obiecte business indiferent de locul în care acestea se află, acesta vă poate oferi o vizualizare de asamblare unificată a tuturor aplicaţiilor dintr-o întreprindere şi poate fi folosit ca un bloc de construire pentru soluţii BPM. Deoarece relaţiile pot fi extinse şi gestionate, ele pot fi utilizate în soluţii complexe de integrare.

Ce sunt relaţiile?

O relaţie este o asociere între obiecte business. Fiecare obiect business dintr-o relaţie este numit *participant* la relaţie. Fiecare participant din relaţie este distins de alţi participanţi pe baza funcţiei sau a *rolului* pe care îl are în acea relaţie. O relaţie conţine o listă de roluri.

Definiţia relaţiei descrie fiecare rol şi specifică modul în care sunt înrudite rolurile. De asemenea, aceasta descrie "forma" generală a relaţiei. De exemplu, acest rol poate avea doar un participant, dar acest alt rol poate avea câţi participanţi sunt necesari. Puteţi defini o relaţie *car-owner*, de exemplu, unde un posesor poate avea mai multe maşini. De exemplu, o instanţă poate avea următorii participanţi pentru fiecare dintre aceste roluri:

- Maşină (Ferrari)
- Posesor (John)

Definiţia relaţiei este un şablon pentru *instanţa* relaţiei. Instanţa este instanţierea relaţiei în timpul rulării. În exemplul cu *proprietarii* de *maşini*, o instanţă ar putea descrie oricare din următoarele asocieri:

- John deţine Ferrari
- Sara deţine Mazda
- Bob deţine Ferrari

Utilizarea relaţiilor vă eliberează de necesitatea de a construi personalizat persistenţa de urmărire a relaţiei din logica dumneavoastră operaţională. Pentru anumite scenarii, serviciul de relaţii face toată munca în locul dumneavoastră. Vedeţi exemplul descris în secţiunea din Relaţii de identitate.

Scenarii

Aici este un exemplu tipic de situaţie în care o soluţie de integrare poate folosi relaţii. O corporaţie mare cumpără mai multe companii sau unităţi de afaceri. Fiecare unitate operaţională foloseşte software diferit pentru a monitoriza personalul şi caietele. Compania are nevoie de o cale de a-şi monitoriza angajaţii şi carnetele lor. Aceasta vrea o soluţie care le permite:

- Vizualizarea tuturor angajaţilor din diverse unităţi de afaceri ca şi cum ar fi într-o singură bază de date
- O singură vizualizare a tuturor carnetelor
- Permite angajaţilor să se logheze în sistem şi să cumpere un carnet.
- Acomodarea diferitelor sisteme de aplicaţii de întreprindere din diverse unităţi de afaceri

Pentru a realiza acest lucru, compania are nevoie de o cale pentru a se asigura, de exemplu, că John Smith şi John A. Smith din aplicaţii diferite sunt văzuţi ca acelaşi angajat. De exemplu, aceştia au nevoie de o cale pentru a consolida o singură entitate în mai multe spaţii ale aplicaţiei.

Scenariile mai complexe de relaţii implică construirea proceselor BPEL care stabilesc relaţii între obiecte diferite găsite în mai multe aplicaţii. Cu scenarii mai complexe de relaţii, obiectele business se află în soluţia de integrare, ci nu în aplicaţii. Serviciul de relaţii asigură o platformă pentru gestionarea în mod persistent a relaţiilor. Înainte de serviciul de relaţii, trebuie să construiţi propriul serviciu de persistenţă al obiectului. Două exemple de scenarii complexe de relaţii sunt:

- Aveţi un obiect business **car** cu un număr VIN într-o aplicaţie SAP şi vreţi să urmăriţi că această maşină este deţinută de altcineva. Totuşi, relaţia de posesiune este cu cineva într-o aplicaţie PeopleSoft. În acest tipar de relaţii, aveţi două soluţii şi este nevoie să construiţi o punte laterală între ele.
- O companie mare de comerţ cu amănuntul vrea să fie capabilă să monitorizeze marfa returnată pentru primirea banilor înapoi sau pentru credit. Există două aplicaţii diferite implicate: OMS (order management system) pentru achiziţii şi RMS (returns management system) pentru returnări. Obiectele business se află în mai multe aplicaţii şi aveţi nevoie de o cale de a afişa relaţiile care există între ele.

Modele de utilizare comune

Cele mai comune tipare de relaţii sunt tiparele *echivalenţă*. Acestea sunt bazate pe referinţa încrucişată sau pe corelare. Există două tipuri de relaţii care se potrivesc cu acest tipar: *non-identitate* şi *identitate*.

- **Relaţii non-identitate** stabilesc asocieri între obiecte business sau alte date pe o bază unul-la-mulţi sau mulţi-la-mulţi. Pentru fiecare instanţă a relaţiei, pot exista una sau mai multe instanţe pentru fiecare participant. Un tip de relaţie non-identitate este o relaţie de căutare statică. Un astfel de exemplu este o relaţie în care **CA** dintr-o aplicaţie SAP este înrudită cu **California** dintr-o aplicaţie Siebel.

- **Relaţii de identitate** stabilesc asocieri între obiecte business sau alte date pe o bază unu-la-unu. Pentru fiecare instanţă a relaţiei, poate exista doar o instanţă a fiecărui participant. Relaţiile de identitate capturează referinţe încrucişate între obiecte business care sunt echivalente semantic, dar care sunt identificate în mod diferit în diferite aplicaţii. Fiecare participant la relaţie este asociat cu un obiect business care are o valoare (sau combinaţie de valori) care identifică obiectul în mod unic. În mod obişnuit, relaţiile de identitate transformă atributele cheie ale obiectelor business precum numere ID şi coduri de produs.

  De exemplu, dacă aveţi obiecte business **car** în aplicaţii SAP, PeopleSoft şi Siebel şi vreţi să construiţi o soluţie care să le sincronizeze, va fi nevoie, în mod normal, să introduceţi logica de sincronizare a relaţiei construite manual în şase mapări:

  SAP -> generic

  generic -> SAP

  PeopleSoft-> generic

  generic-> PeopleSoft

  Siebel-> generic

  generic-> Siebel

  Totuşi, dacă utilizaţi relaţii în soluţia dumneavoastră, serviciul de relaţii asigură implementări de tipare preconstruite care menţin toate aceste relaţii pentru dumneavoastră.

Unelte pentru lucrul cu relaţii

Editorul de relaţii din Integration Designer este unealta pe care o utilizaţi pentru a modela şi proiecta relaţii şi roluri de integrare business. Pentru informaţii detaliate despre experienţa necesară şi despre taskuri la crearea relaţiilor şi utilizarea editorului de relaţii, vedeţi Creare relaţii.

Serviciul de relaţii este un serviciu de infrastructură din IBM Business Process Manager care menţine relaţii şi roluri în sistem şi oferă operaţii pentru gestiunea relaţiei şi a rolului.

Managerul de relaţii este interfaţa administrativă pentru gestionarea relaţiilor. Acesta este accesat prin pagina Manager de relaţii a consolei administrative.

Relaţiile pot fi invocate programatic prin API-urile serviciului de relaţii.

Serviciu de relaţie

Serviciul de relaţii memorează date ale relaţiei în tabele de relaţii unde păstrează pista valorilor specifice aplicaţiei din aplicaţii şi soluţii. Serviciul de relaţii oferă operaţii pentru gestiunea relaţiilor şi a rolului.

Cum funcţionează relaţiile

Relaţiile şi rolurile sunt descrise folosind interfaţa grafică a uneltei editor de relaţii din Integration Designer. Serviciul de relaţii memorează date de corelare în tabele din baza de date a relaţiei în sursa implicită de date pe care o specificaţi la configurarea serviciului de relaţii. O tabelă separată (uneori numită tabelă participantă) memorează informaţii pentru fiecare participant la relaţie. Serviciul de relaţii utilizează aceste tabele ale relaţiei pentru a păstra pista valorilor înrudite specifice aplicaţiei şi pentru a propaga informaţii actualizate în toate soluţiile.

Relaţiile, care sunt artefacte business, sunt implementate într-un proiect sau într-o bibliotecă partajată. La prima implementare, serviciul de relaţii populează datele.

În timpul rulării, când mapările sau alte componente IBM Business Process Manager necesită o instanţă de relaţie, instanţele relaţiei sunt fie actualizate, fie extrase în funcţie de scenariu.

Datele instanţei de relaţie sau rol pot fi manipulate prin trei mijloace:

- Invocări componentă IBM Business Process Manager snippet Java ale API-urilor serviciului de relaţii

- Transformări ale relaţiei în serviciul de mapare a obiectului business IBM Business Process Manager
- Unealta managerului de relaţii

Pentru informaţii detaliate despre experienţa necesară şi despre taskuri la crearea relaţiilor, identificarea tipurilor de relaţii şi utilizarea editorului de relaţii, vedeţi subiectul Creare relaţii.

Manager de relaţie

Managerul de relaţii este interfaţa administrativă pentru gestionarea relaţiilor. Acesta este accesat prin pagina Manager de relaţii a consolei administrative.

Managerul de relaţii asigură o interfaţă grafică cu utilizatorul pentru crearea şi manipularea datelor relaţiei şi ale rolului în timpul rulării. Puteţi gestiona entităţi ale relaţiei la toate nivelurile: instanţă de relaţie, instanţă de rol şi niveluri ale datelor atribut şi ale datelor proprietate. Cu managerul de relaţii, puteţi:

- Vizualiza o listă a relaţiilor din sistem şi a informaţiilor detaliate despre relaţii individuale
- Gestiona instanţe de relaţii:
  - Interoga date ale relaţiei pentru a vizualiza subseturi de date ale instanţei
  - Interoga date ale relaţiei pentru a vizualiza subseturi de date ale instanţei utilizând vizualizări ale bazei de date
  - Vizualiza o listă a instanţelor relaţiei care se potrivesc unei interogări a relaţiei şi informaţii detaliate despre o instanţă
  - Edita valorile proprietăţii pentru o instanţă a relaţiei
  - Crea şi şterge instanţe de relaţie
- Gestiona roluri şi instanţe de roluri:
  - Vizualiza detalii despre un rol sau o instanţă de rol
  - Edita proprietăţile instanţei de rol
  - Crea şi şterge instanţe de rol pentru o relaţie
  - Derula înapoi date ale instanţei de relaţie la un punct în care ştiţi că datele sunt de încredere
- Importa date dintr-o relaţie statică existentă în sistemul dumneavoastră sau exporta date dintr-o relaţie statică existentă într-un fişier RI sau CSV
- Înlătura schema şi datele relaţiei din magazie când aplicaţia care le foloseşte este dezinstalată

Relaţii în medii Network Deployment

Relaţiile pot fi utilizate în medii ND (Network Deployment) fără nici o configuraţie suplimentară.

În medii ND (Network Deployment), relaţiile sunt instalate într-un cluster de aplicaţii. Atunci relaţiile sunt vizibile în cluster şi toate serverele din cluster au acces la datele instanţei memorate în baza de date a relaţiei. Abilitatea de rula serviciul de relaţii într-un mediu ND îl face scalabil şi cu disponibilitate bună.

Managerul de relaţii permite relaţiilor să fie gestionate în cluster-e diferite printr-o interfaţă administrativă centralizată. Conectaţi managerul de relaţii la un server dintr-un cluster selectând MBean-ul relaţiei sale.

API-uri ale serviciului de relaţii

Relaţiile pot fi invocate programatic prin API-urile serviciului de relaţii în sau în afara mapărilor de obiecte business.

Sunt disponibile trei tipuri de API-uri:

- API-uri de manipulare a instanţei de relaţie (inclusiv crearea, actualizarea, ştergerea directă a datelor instanţei)
- API-uri suport ale tiparului de relaţii (inclusiv correlate(), correlateforeignKeyLookup)
- Tipare căutare relaţie (API-uri de căutare)

Magistrala ESB (Enterprise Service Bus) din IBM Business Process Manager

IBM Business Process Manager suportă integrarea serviciilor aplicaţie, inclusiv aceleaşi capabilităţi ca WebSphere Enterprise Service Bus.

Conectarea serviciilor printr-o magistrală pentru serviciile întreprinderii

Cu ajutorul unui ESB (Enterprise Service Bus), puteţi maximiza flexibilitatea unui SOA. Participanţii din interacţiunea cu serviciul sunt conectaţi mai degrabă la ESB, decât direct unul cu altul.

În cazul în care solicitantul serviciului se conectează la ESB, atunci ESB-ul îşi asumă răspunderea pentru furnizarea cererilor sale, folosind mesaje, către un furnizor de servicii oferind funcţia necesară şi calitatea serviciilor. ESB uşurează interacţiunile solicitant-furnizor şi adresează protocoalele nepotrivite, tiparele de interacţiune sau capabilităţile serviciului. De asemenea, un ESB poate permite sau îmbunătăţi monitorizarea şi gestiunea. ESB oferă caracteristici de virtualizare şi management care pun în aplicare şi extind capacităţile de bază pentru SOA.

ESB prezintă pe scurt următoarele caracteristici:

Locaţie şi identitate
: Participanţii nu trebuie să ştie locaţia sau identitatea altor participanţi. De exemplu, solicitanţii nu trebuie să fie conştienţi de faptul că o cerere ar putea fi deservită de oricare dintre furnizori; furnizorii de servicii pot fi adăugaţi sau înlăturaţi fără întrerupere.

Protocol de Interacţiune
: Participanţii nu trebuie să partajeze acelaşi protocol de comunicaţie sau stil de interacţiune. De exemplu, o cerere exprimată ca SOAP peste HTTP poate fi deservită de un furnizor care înţelege doar SOAP peste JMS (Java Message Service).

Interfaţă
: Solicitanţii şi furnizorii nu trebuie să fie de acord cu o interfaţă comună. Un ESB împacă diferenţele prin transformarea mesajelor de cerere şi de răspuns într-o formă aşteptată de furnizor.

Calităţile unui serviciu (de interacţiune)
: Participanţii, sau administratorii de sisteme, îşi declară cerinţele legate de calitatea serviciilor, inclusiv autorizaţia cererilor, criptarea şi decriptarea conţinutului mesajelor, auditarea automată a interacţiunilor dintre servicii, şi modul în care cererile lor ar trebui să fie rutate (de exemplu optimizarea vitezei sau a costului).

Interpunerea ESB între participanţi vă dă posibilitatea să modulaţi interacţiunea acestora prin intermediul unei construcţii logice numite *mediere*. Medierile se aplică mesajelor aflate în zbor între solicitanţi şi furnizori. De exemplu, medierile pot fi folosite pentru a găsi servicii cu caracteristici specifice cerute de un solicitant, sau pentru a rezolva diferenţele interfeţei apărute între solicitanţi şi furnizori. Pentru interacţiuni complexe, medierile pot fi legate secvenţial.

Folosind medieri, o magistrală pentru serviciile întreprinderii efectuează următoarele acţiuni între solicitant şi serviciu:

- *Rutarea* mesajelor între servicii. O magistrală pentru serviciile întreprinderii oferă o infrastructură de comunicaţie comună care poate fi folosită pentru a conecta servicii, şi, prin urmare funcţiile business pe care acestea le reprezintă, fără a fi nevoie ca programatorii să scrie şi să menţină logica complexă de conectivitate.
- *Convertirea* protocoalelor de transport între solicitant şi serviciu. O magistrală pentru serviciile de întreprindere oferă o cale consistentă bazată pe standarde de integrare a funcţiilor business care folosesc diferite standarde IT. Acest lucru permite integrarea funcţiilor business care nu au putut comunica în mod normal, cum ar fi conectarea aplicaţiilor în silozurile departamentale sau activarea aplicaţiilor în companii diferite care participă în interacţiunile serviciului.
- *Transformarea* formatelor mesajelor între solicitant şi serviciu. O magistrală pentru serviciile de întreprindere permite funcţiilor business să schimbe informaţii în formate diferite, iar magistrala se asigură că informaţiile livrate funcţiei business sunt în formatul cerut de acea aplicaţie.
- *Tratarea* evenimentelor business din surse incompatibile. O magistrală pentru serviciile de întreprindere suportă interacţiuni bazate pe evenimente în plus faţă de schimbul de mesaje ce tratează cererile serviciului.

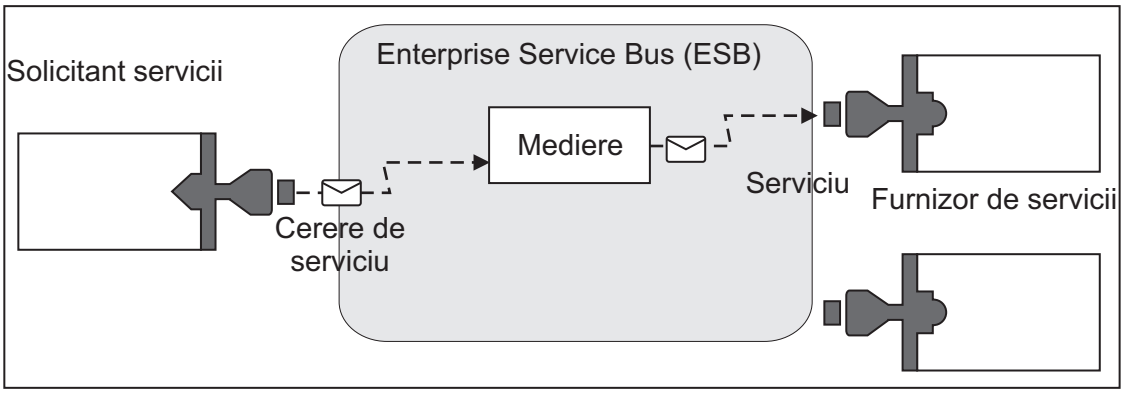


Figura 42. O magistrală pentru serviciile întreprinderii. Magistrală pentru serviciile de întreprindere rutează mesajele între aplicaţii, care pot fi solicitanţi sau furnizori de servicii. Magistrala converteşte protocoalele de transport şi transformă formatul mesajelor între solicitanţi şi furnizori. În această figură, fiecare aplicaţie utilizează un protocol diferit (reprezentate prin formele geometrice diferite ale conectorilor lor) şi foloseşte diferite formate de mesaje.

Folosind magistrala pentru serviciile de întreprindere vă puteţi concentra pe afacerea dumneavoastră de bază mai degrabă decât pe sistemele dumneavoastră computer. Puteţi modifica sau adăuga la servicii, dacă este necesar; de exemplu, pentru a răspunde la modificările din cerinţa de afaceri, pentru a adăuga capacitate de serviciu suplimentar sau pentru a adăuga noi capabilităţi. Aveţi posibilitatea să efectuaţi modificările necesare prin reconfigurarea magistralei, cu impact redus sau nul asupra serviciilor şi aplicaţiilor existente care folosesc magistrala.

Infrastructura de mesagerie ESB (Enterprise Service Bus)

IBM Business Process Manager include capabilităţi ale magistralei ESB. IBM Business Process Manager suportă integrarea tehnologiilor orientate spre servicii, orientate pe mesaj şi conduse de eveniment pentru a asigura o infrastructură a mesageriei bazată pe standarde într-o magistrală de servicii de întreprindere integrate.

Capabilităţile serviciului de întreprindere pe care le puteţi utiliza pentru aplicaţiile dumneavoastră de întreprindere asigură nu doar un nivel de transport dar şi suport de mediere pentru a facilita interacţiunile serviciului. Magistrala ESB (Enterprise Service Bus) este construită în jurul standardelor deschise şi a SOA (service-oriented architecture). Este bazată pe infrastructura Java EE robustă şi serviciile platformei asociate furnizate de IBM WebSphere Application Server Network Deployment.

IBM Business Process Manager este alimentat de aceeaşi tehnologie disponibilă cu IBM WebSphere Enterprise Service Bus. Această capabilitate este parte componentă a funcţionalităţii de bază a IBM Business Process Manager şi nu este necesară nici o licenţă suplimentară pentru WebSphere Enterprise Service Bus pentru a beneficia de aceste capabilităţi.

Totuşi, puteţi implementa licenţe autonome suplimentare ale WebSphere Enterprise Service Bus în întreprinderea dumneavoastră pentru a extinde întinderea conectivităţii soluţiilor de integrare procese motorizate de deIBM Business Process Manager. De exemplu, WebSphere Enterprise Service Bus poate fi instalat mai aproape de o aplicaţie SAP pentru a găzdui un IBM WebSphere Adapter for SAP şi pentru a transforma mesajele SAP înainte de a trimite acele informaţii în reţea într-un proces operaţional condus de IBM Business Process Manager.

Gazde de destinaţii mesagerie sau coadă:

O gazdă destinaţie de mesagerie sau coadă furnizează funcţia de mesagerie într-un server. Un server devine gazda destinaţiei mesageriei la configurarea lui ca destinaţie mesagerie.

Un motor de mesagerie rulează într-un server. Motorul mesagerie furnizează funcţii mesagerie şi un punct de conexiune pentru ca aplicaţiile să se conecteze la magistrală. Comunicaţia asincronă Service Component Architecture (SCA), importurile şi exporturile JMS, procesarea internă asincronă utilizează cozi de mesaje în motorul de mesagerie.

Mediul de implementare conectează sursa mesajului la destinaţia mesajului prin magistrală la implementarea modulelor de aplicaţii. Cunoaşterea sursei mesajului şi a destinaţiei mesajului vă ajută să determinaţi de ce tip de mediu de implementare aveţi nevoie.

Aplicaţiile pot stoca date persistente într-un depozit de date, care este un set de tabele dintr-o bază de date sau schemă, sau într-un depozit de fişiere. Motorul mesagerie utilizează o instanţă a unei surse de date JDBC pentru a interacţiona cu acea bază de date.

Configuraţi gazda destinaţie a mesageriei la definirea mediului de implementare utilizând **Server** din consola administrativă sau desemnaţi serverul ca gazdă destinaţie în timpul instalării software-ului.

Depozite de date:

Fiecare motor de mesagerie poate utiliza un depozit de date care este un set de tabele dintr-o bază de date sau o schemă care memorează date persistente.

Toate tabelele din depozitul de date sunt reţinute în aceeaşi schemă a bazei de date. Puteţi crea fiecare depozit de date într-o bază de date separată. Alternativ, puteţi crea mai multe depozite de date în aceeaşi bază de date, fiecare depozit de date utilizând o schemă diferită.

Un motor de mesagerie utilizează o instanţă a unei surse de date JDBC pentru a interacţiona cu baza de date care conţine depozitul de date pentru acel motor de mesagerie.

Furnizori JDBC:

Puteţi folosi furnizorii JDBC pentru interacţionarea aplicaţiile cu bazele de date relaţionale.

Aplicaţiile folosesc furnizori JDBC pentru interacţionarea cu bazele de date relaţionale. Furnizorii JDBC livrează clasele de implementare specifice driver-ului JDBC pentru accesul la un tip specific de bază de date. Pentru crearea unui pool de conexiuni la acea bază de date, asociaţi sursa de date cu furnizorul JDBC. Împreună, furnizorul JDBC şi obiectele sursei de date sunt echivalente funcţional cu fabrica de conexiuni Java EE Connector Architecture (JCA), care furnizează conexiunea cu o bază de date non-relaţională.

Referiţi-vă la exemplele din Setări pentru mediul tipic autonom şi Setări pentru mediul tipic de implementare din subiectul anterior.

Pentru informaţii suplimentare despre furnizorii JDBC, vedeţi “Furnizori JDBC” în Centrul de informare WebSphere Application Server.

Magistrale de integrare a serviciului pentru IBM Business Process Manager:

O magistrală de integrare a serviciului este un mecanism de comunicare gestionat care suportă integrarea serviciului prin mesageria sincronă şi asincronă. O magistrală conţine motoare de mesagerie interconectate care gestionează resursele magistralei. Este una din tehnologiile WebSphere Application Server pe care este bazat IBM Business Process Manager.

Unele magistrale sunt create automat pentru utilizarea de către sistem, aplicaţiile SCA (Service Component Architecture) pe care le implementaţi şi de către alte componente. De asemenea, puteţi crea magistrale pentru a suporta logica integrării serviciului sau alte aplicaţii, de exemplu, pentru a suporta aplicaţii care acţionează ca solicitanţi şi furnizori ai serviciului în IBM Business Process Manager sau pentru a vă lega WebSphere MQ.

O destinaţie magistrală este o adresă logică la care aplicaţiile se pot ataşa ca producător, consumator sau ambele. O destinaţie coadă este o destinaţie magistrală care este utilizată pentru mesagerie punct-la-punct.

Fiecare magistrală poate avea unul sau mai mulţi membri magistrală, fiecare dintre ei fiind un server sau un cluster.

Topologie magistrală este aranjarea fizică a serverelor de aplicaţii, motoarelor de mesagerie şi a managerilor de cozi WebSphere MQ şi tiparul conexiunilor magistralei şi al legăturilor dintre ele care alcătuiesc magistrala de servicii întreprindere.

Unele magistrale de integrare servicii sunt create automat pentru a suporta IBM Business Process Manager. Între şase magistrale sunt create la crearea mediului de implementare sau la configurarea unui server sau cluster pentru a suporta aplicaţii SCA. Fiecare dintre aceste magistrale are cinci aliasuri de autentificare pe care trebuie să le configuraţi.

Magistrală de sistem SCA:

Magistrala de sistem SCA este o magistrală de integrare a serviciului care este utilizată pentru a găzdui destinaţii pentru module SCA (Service Component Architecture). Runtime-ul SCA, ce suportă module de mediere, foloseşte destinaţii coadă pe magistrala de sistem ca pe o infrastructură pentru interacţiuni asincrone între componente şi module.

Magistrala de sistem este creată automat odată cu mediul de implementare sau la configurarea serverului sau cluster-ului pentru aplicaţii SCA. Magistrala de sistem furnizează un domeniu în care resursele sunt configurate pentru module de mediere şi puncte finale de interacţiune. Magistrala permite rutarea mesajelor între punctele finale. Puteţi să specificaţi QoS (calitatea serviciului) pentru magistrală, incluzând prioritatea şi fiabilitatea.

Numele magistralei este SCA.SYSTEM.busID.Bus. Aliasul de autentificare folosit la securizarea magistralei este SCA_Auth_Alias.

Magistrală de aplicaţii SCA:

Destinaţiile magistralei de aplicaţii suportă comunicarea asincronă a WebSphere Business Integration Adapters şi a altor componente System Component Architecture.

Magistrala aplicaţiei este creată automat la crearea unui mediu de implementare sau la configurarea unui server sau cluster pentru a suporta aplicaţii SCA. Magistrala aplicaţiei este similară cu magistralele de integrare a serviciului pe care le puteţi crea pentru a suporta logica integrării serviciului sau alte aplicaţii.

Numele magistralei este SCA.APPLICATION.busID.Bus. Aliasul de autentificare utilizat pentru securizarea acestei magistrale este SCA_Auth_Alias.

Magistrala Common Event Infrastructure:

Magistrala Common Event Infrastructure este utilizată pentru transmiterea evenimentelor de bază obişnuite, în mod asincron, serverului Common Event Infrastructure configurat.

Numele magistralei este CommonEventInfrastructure_Bus. Aliasul de autentificare utilizat pentru securizarea acestei magistrale este CommonEventInfrastructureJMSAuthAlias

Magistrala Business Process Choreographer:

Utilizaţi numele magistralei Business Process Choreographer şi autentificarea pentru transmisia mesajelor interne.

Magistrala Business Process Choreographer este utilizată pentru transmiterea mesajelor intern şi pentru API-ul Java Messaging Service (JMS) al managerului de flux de afaceri.

Numele magistralei este BPC.cellName.Bus. Aliasul de autentificare este BPC_Auth_Alias

Magistrală Performance Data Warehouse:

Magistrala Performance Data Warehouse este folosită pentru a transmite mesaje intern de către infrastructură şi pentru a comunica cu clienţii IBM Business Process Manager.

Magistrala Performance Data Warehouse este creată în mod automat la crearea unui mediu de implementare.

Numele magistralei este PERFDW.busID.Bus. Aliasul de autentificare utilizat pentru securizarea acestei magistrale este PERFDWME_Auth_Alias.

Magistrală Process Server:

Magistrala Process Server este folosită pentru transmiterea internă a mesajelor de către infrastructură şi pentru a comunica cu clienţii IBM Business Process Manager.

Magistrala Process Server este creată automat când creaţi un mediu de implementare.

Numele magistralei este PROCSVR.busID.Bus. Aliasul de autentificare utilizat pentru securizarea acestei magistrale este PROCSVRME_Auth_Alias.

Aplicaţii de servicii şi module de servicii

Un modul serviciu este un modul Service Component Architecture (SCA) care oferă servicii în mediul runtime. Atunci când instalaţi un modul serviciu pe IBM Business Process Manager, construiţi o aplicaţie serviciu asociată, care este împachetată ca un fişier EAR (enterprise archive).

Modulele de servicii sunt unităţi de bază ale implementării şi pot conţine componente, biblioteci şi module de intermediere folosite de aplicaţia pentru servicii asociată. Modulele de servicii au exporturi şi, în mod opţional, importuri pentru a defini relaţiile între module şi solicitanţii şi furnizorii serviciului. WebSphere Process Server suportă module pentru serviciile business şi module de mediere. Atât modulele, cât şi modulele de mediere sunt tipuri de module SCA. Un modul de mediere permite comunicaţia între aplicaţii prin transformarea invocării serviciului într-un format înţeles de ţintă, transmiţând cererea către ţintă şi returnând rezultatul iniţiatorului. Un modul pentru un serviciu operaţional implementează logica unui proces operaţional. Totuşi, un modul poate de asemenea să includă aceeaşi logică de mediere care poate fi împachetată într-un modul de mediere.

Implementarea unei aplicaţii de servicii

Procesul de implementare a unui fişier EAR care conţine o aplicaţie de servicii este acelaşi cu cel de implementarea a oricărui fişier EAR. Puteţi modifica valorile pentru parametrii de mediere în momentul implementării. După ce au implementat un fişier EAR care conţine un modul SCA, puteţi vizualiza detaliile legate de aplicaţia de servicii şi modulele sale asociate. Puteţi vedea modul în care un modul de servicii este conectat la solicitanţii serviciului (prin exporturi) şi la furnizorii de servicii (prin importuri).

Vizualizarea detaliilor modulului SCA

Detaliile legate de modulul de servicii pe care le puteţi vizualiza depind de modulul SCA. Acestea includ următoarele atribute.

- Nume modul SCA
- Descriere modul SCA
- Nume asociat aplicaţiei
- Informaţii legate de versiunea modulului SCA, daca modulul are mai multe versiuni
- Importuri module SCA:
  - Interfeţele de import sunt definiţii prezentare pe scurt care descriu modul în care un modul SCA accesează un serviciu.
  - Legările de import sunt definiţii concrete care specifică mecanismul fizic prin care un modul SCA accesează un serviciu. De exemplu, folosirea SOAP/HTTP.
- Exporturi modul SCA:
  - Interfeţele de export sunt definiţii prezentare pe scurt care descriu modul în care solicitanţii serviciului accesează un modul SCA.

  - Legările de export sunt definiţii concrete care specifică mecanismul fizic prin care un solicitant al serviciului accesează un modul SCA, şi în mod indirect, un serviciu.
- Proprietăţile modulului SCA

Importuri şi legături import:

Importurile definesc interacţiuni între modulele SCA şi furnizorii de servicii. Modulele SCA folosesc importuri pentru a permite componentelor să acceseze servicii externe (servicii care se află în afara modulului SCA) folosind o reprezentare locală. Legările de import definesc o anumită cale prin care este accesat un serviciu extern.

Dacă modulele SCA nu au nevoie de acces la serviciile externe, atunci nu este necesar ca acestea să aibă importuri. Modulele de mediere au de obicei unul sau mai multe importuri care sunt utilizate pentru a transmite mesaje sau cereri către ţintele lor intenţionate.

Interfeţe şi legături

Importul unui modul SCA are nevoie de cel puţin o interfaţă şi are o singură legare.

- Interfeţele de import sunt definiţii abstracte care definesc un set de operaţiuni folosind Web Services Description Language (WSDL), şi limbaj XML pentru descrierea serviciilor Web. Un modul SCA poate avea mai multe interfeţe de import.
- Legările de import sunt definiţii concrete care specifică mecanismul fizic folosit de modulele SCA pentru a accesa un serviciu extern.

Legări de import suportate

IBM Business Process Manager suportă următoarele legături de import:

- Legările SCA conectează modulele SCA cu alte module SCA. De asemenea, legările SCA sunt menţionate şi ca legături implicite.
- Legările Serviciu Web permit componentelor să invoce servicii web. Protocoalele suportate sunt SOAP1.1/HTTP, SOAP1.2/HTTP, şi SOAP1.1/JMS.

  Puteţi folosi o legare SOAP1.1/HTTP sau SOAP1.2/HTTP bazată pe API-ul Java API pentru JAX-WS (XML Web Services), ceea ce permite interacţiunea cu serviciile folosind documente sau legături literale RPC şi care folosesc handler-e JAX-WS pentru a personaliza invocarea. Este furnizată o legare SOAP1.1/HTTP separată pentru a permite interacţiunea cu serviciile care folosesc o legare codată RPC sau acolo unde există o cerinţă pentru folosirea handler-elor JAX-RPC pentru a personaliza invocările.
- Legările HTTP vă permit să accesaţi aplicaţii folosind protocolul HTTP.
- Legările de import EJP (Enterprise JavaBeans) permit componentelor SCA să invoce servicii oferite de logica operaţională Java EE care rulează pe un server Java EE.
- Legările EIS (Enterprise information system) asigură conectivitatea între componente SCA şi un EIS extern. Această comunicaţie se realizează prin utilizarea adaptoarelor resurselor.
- Legările JMS (Java Message Service) 1.1 permit interoperabilitatea cu furnizorul implicit de mesaje WebSphere Application Server. JMS poate exploata diferite tipuri de transport, inclusiv TCP/IP şi HTTP sau HTTPS. Clasa JMS Message şi cele cinci subtipuri ale sale (Text, Bytes, Object, Stream şi Map) sunt acceptate automat.
- Legările JMS interoperabilitate cu furnizori JMS terţă parte care se integrează cu WebSphere Application Server folosind JMS ASF (Application Server Facility).
- Legările JMS MQ WebSphere permit interoperabilitatea cu furnizorii JMS bazaţi pe MQ WebSphere. Clasa JMS Message şi cele cinci subtipuri ale sale (Text, Bytes, Object, Stream şi Map) sunt acceptate automat. Dacă doriţi să utilizaţi WebSphere MQ pe post de furnizor JMS, folosiţi legările JMS MQ WebSphere.
- Legările MQ WebSphere permit interoperabilitatea cu furnizorii MQ WebSphere. Puteţi folosi legături WebSphere MQ doar împreună cu managerii cozii aflaţi la distanţă prin intermediul unei conexiuni client WebSphere MQ; nu le puteţi folosi cu managerii locali ai cozii. Folosiţi legături WebSphere MQ dacă doriţi să comunicaţi cu aplicaţiile WebSphere MQ native.

Invocarea dinamică a serviciilor

Serviciile pot fi invocate prin orice legare de import suportată. Un serviciu este găsit în mod normal la un punct final specificat în import. Acest punct final este numit punct final static. Invocarea unui serviciu diferit este posibilă prin înlocuirea punctului final static. Înlocuirea dinamică a punctelor finale statice vă permite să invocaţi un serviciu la un alt punct final, prin orice legare de import suportată. Invocarea dinamică a serviciilor vă permite de asemenea să invocaţi un serviciu în care legarea de import suportată nu are un punct final static.

Un import cu o legare asociată este folosit pentru a specifica protocolul şi configuraţia sa pentru invocarea dinamică. Importarea folosită pentru invocarea dinamică poate fi legată de componenta de apelare, sau poate fi selectată dinamic la runtime.

Pentru serviciul Web şi invocaţii SCA, se poate face o invocare dinamică fără importare, cu protocolul şi configurarea deduse din URL-ul punct final. Tipul ţintă de invocare este identificat din URL-ul final. Dacă este folosit un import, atunci URL-ul trebuie să fie compatibil cu protocolul legării de import.

- Un URL SCA indică invocarea unui alt modul SCA.
- Un HTTP sau un JMS URL indică în mod implicit invocări ale serviciului web; pentru aceste URL-uri este posibilă furnizarea unor valori de tip legătură suplimentare care să indice faptul că URL-ul reprezintă o invocare a modului de legare a unui HTTP sau JMS.
- Pentru un serviciu URL HTTP, modul implicit este folosirea SOAP 1.1, şi se poate specifica o valoare tip legătură care să indice folosirea SOAP 1.2.

Exporturile şi legările de export:

Exporturile definesc interacţiuni între modulele SCA şi solicitanţii de servicii. Modulele SCA folosesc exporturi pentru a oferi altora servicii. Legările de export definesc un mod specific prin care un modul SCA este accesat de solicitanţii serviciului.

Interfeţe şi legături

Un export pentru un modul SCA are nevoie de cel puţin o interfaţă.

- Interfeţele de export sunt definiţii abstracte care definesc un set de operaţiuni folosind Web Services Description Language (WSDL), şi limbaj XML pentru descrierea serviciilor Web. Un modul SCA poate avea mai multe interfeţe de export.
- Legările de export sunt definiţii concrete care specifică mecanismul fizic folosit de solicitanţii serviciului pentru a accesa un serviciu. De obicei, exportul unui modul SCA are specificată o singură legare. Un export fără legături specificate este interpretat de mediul runtime ca un export cu o legătură SCA.

Legări de export suportate

IBM Business Process Manager suportă următoarele legături de export:

- Legările SCA conectează modulele SCA cu alte module SCA. De asemenea, legările SCA sunt menţionate şi ca legături implicite.
- Legările Serviciu web permit exporturi care pot fi invocate ca şi servicii web. Protocoalele suportate sunt SOAP1.1/HTTP, SOAP1.2/HTTP, şi SOAP1.1/JMS.

  Puteţi folosi o legare SOAP1.1/HTTP sau SOAP1.2/HTTP bazată pe JAX-WS (Java API for XML Web Services), ceea ce permite interacţiunea cu serviciile folosind documente sau legături literale RPC şi care folosesc handler-e JAX-WS pentru a personaliza invocările. Este furnizată o legare SOAP1.1/HTTP separată pentru a permite interacţiunea cu serviciile care folosesc o legare codată RPC sau acolo unde există o cerinţă pentru folosirea handler-elor JAX-RPC pentru a personaliza invocările.
- Legările HTTP permit ca exporturile să fie accesate folosind protocolul HTTP.
- Legările de export EJB (Enterprise JavaBeans) permit componentelor SCA să fie exportate sub formă de EJB-uri, astfel încât logica operaţională Java EE să poată invoca componente SCA care altfel le-ar fi indisponibile.

- Legările EIS (Enterprise information system) asigură conectivitatea între componente SCA şi un EIS extern. Această comunicaţie se realizează prin utilizarea adaptoarelor resurselor.
- Legările JMS (Java Message Service) 1.1 permit interoperabilitatea cu furnizorul implicit de mesaje WebSphere Application Server. JMS poate exploata diferite tipuri de transport, inclusiv TCP/IP şi HTTP sau HTTPS. Clasa JMS Message şi cele cinci subtipuri ale sale (Text, Bytes, Object, Stream şi Map) sunt acceptate automat.
- Legările JMS interoperabilitate cu furnizori JMS terţă parte care se integrează cu WebSphere Application Server folosind JMS ASF (Application Server Facility).
- Legările JMS MQ WebSphere permit interoperabilitatea cu furnizorii JMS bazaţi pe MQ WebSphere. Clasa JMS Message şi cele cinci subtipuri ale sale (Text, Bytes, Object, Stream şi Map) sunt acceptate automat. Dacă doriţi să utilizaţi WebSphere MQ pe post de furnizor JMS, folosiţi legările JMS MQ WebSphere.
- Legările MQ WebSphere permit interoperabilitatea cu furnizorii MQ WebSphere. Puteţi utiliza o conexiune de la distanţă (sau client) pentru a vă conecta la un manager de coadă MQ aflat pe o maşină la distanţă. O conexiune locală (sau legături) este o conexiune directă la WebSphere MQ. Aceasta poate fi utilizată doar pentru o conexiune la un manager al cozii MQ de pe aceeaşi maşină. WebSphere MQ va permite ambele tipuri de conexiuni, dar legările MQ suportă doar conexiunea "de la distanţă" (sau "client").

Module de mediere:

Modulele de mediere sunt module SCA (Service Component Architecture) care pot modifica formatul, conţinutul sau ţinta cererilor serviciului.

Modulele de mediere operează asupra mesajelor care sunt în zbor între solicitanţii şi furnizorii serviciului. Puteţi ruta mesaje către diferiţi furnizori de servicii şi puteţi modifica conţinutul sau forma mesajului. Modulele de mediere pot oferi funcţii cum ar fi înregistrarea mesajului şi procesarea erorii care sunt adaptate cerinţelor dumneavoastră.

Puteţi modifica anumite aspecte ale modulelor de mediere din consola administrativă fără a fi nevoie să reimplementaţi modulul .

Componentele modulelor de mediere

Modulele de mediere cuprind următoarele elemente:

- Importuri, care definesc interacţiunea între modulele SCA si furnizorii de servicii. Acestea permit modulelor SCA să apeleze servicii externe ca şi cum acestea ar fi locale. Puteţi vizualiza importurile modulelor de mediere şi să modificaţi legarea.
- Exporturi, care definesc interacţiunea între modulele SCA şi solicitanţii serviciului. Acestea permit unui modul SCA să ofere un serviciu şi să definească interfeţele externe (puncte de acces) pentru un modul SCA. Puteţi vizualiza exporturile modulelor de mediere.
- Componente SCA, care construiesc blocuri pentru modulele SCA precum module de mediere. Aveţi posibilitatea să creaţi şi să particularizaţi module SCA şi componente în mod grafic, folosind Integration Designer. După ce aţi implementa un modul de mediere puteţi personaliza anumite aspecte ale acestuia din consola administrativă fără să fiţi nevoiţi să reimplementaţi modulul.

  De obicei, module de mediere conţine un anumit tip de componentă SCA numită *componentă a fluxului de mediere*. Componentele fluxului de mediere definesc fluxurile de mediere.

  Componente unui flux de mediere poate conţine nici o, una, sau un număr de primitive de mediere. IBM Business Process Manager suportă un set livrat de primitive de mediere care oferă funcţionalitate pentru rutarea şi transformarea mesajelor. Pentru flexibilitate suplimentară a primitivei de mediere, utilizaţi primitiva Mediere personalizată pentru apelarea logicii personalizate.

  Scopul unui modul de mediere care nu conţine o componentă pentru fluxul de mediere este de a transforma cererile serviciului de la un protocol la altul. De exemplu, cererea unui serviciu ar putea fi făcută folosind SOAP/JMS, dar ar putea avea nevoie să fie transformară în SOAP/HTTP înainte de a fi trimisă mai departe.

  Notă: Aveţi posibilitatea să vizualizaţi şi să faceţi anumite modificări asupra modulelor de mediere din IBM Business Process Manager. Totuşi, nu puteţi vizualiza sau modifica componentele SCA din interiorul unui modul din

IBM Business Process Manager. Utilizaţi Integration Designer pentru a personaliza componentele SCA.

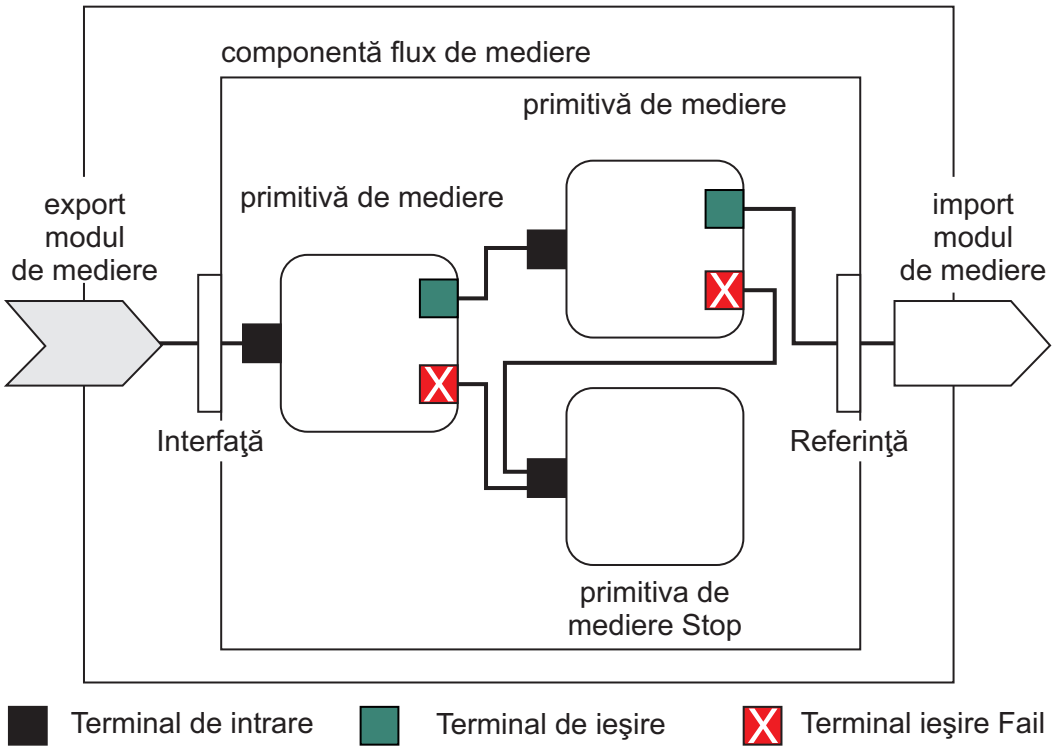


Figura 43. Exemplu simplificat de modul mediere. Modulul de mediere conţine o singură componentă a fluxului de mediere, care conţine primitive de mediere.

- Proprietăţi

  Primitivele de mediere au proprietăţi, unele dintre ele pot fi afişate în consola administrativă sub formă de proprietăţi suplimentare ale unui modul SCA.

  Pentru ca proprietăţile primitivelor de mediere să fie vizibile din consola administrativă IBM Business Process Manager, dezvoltatorul care se ocupă cu integrarea trebuie să le promoveze. Anumite proprietăţi se pretează a fi configurate administrativ şi Integration Designer le descrie ca proprietăţi ce pot fi promovate deoarece acestea pot fi promovate din cercul de integrare în ciclul administrativ. Alte proprietăţi nu sunt potrivite pentru configurare administrativă deoarece modificarea acestora poate afecta fluxul de mediere în aşa fel încât modulul de mediere va trebui să fie reimplementat. Integration Designer listează proprietăţile pe care le puteţi alege pentru a promova în proprietăţile promovate ale unei primitive de mediere.

  Puteţi folosi consola administrativă IBM Business Process Manager pentru a modifica valoarea proprietăţilor promovate fără a trebui să reimplementaţi un modul de mediere sau să reporniţi serverul sau modulul.

  În general, fluxurile de mediere utilizează imediat modificările proprietăţii. Cu toate acestea, dacă apar modificări asupra proprietăţii într-o celulă pentru managerul de implementare, acestea au efect in fiecare nod pe măsură ce acestea sunt sincronizate. De asemenea, fluxurile de mediere care sunt în curs continuă să utilizeze valorile anterioare.

  Notă: Din consola administrativă, aveţi posibilitatea să modificaţi doar valorile proprietăţii, nu grupurile acesteia, numele sau tipurile. Dacă doriţi să modificaţi grupurile de proprietăţi, nume sau tipuri, trebuie să utilizaţi Integration Designer.
- Un modul de mediere sau biblioteca dependentă poate defini, de asemenea, subfluxuri. Un subfulx încapsulează un set de primitive de mediere, legate împreună ca o piesă reutilizabilă a logicii de integrare. Pentru a invoca un subflux, se poate adăuga o primitivă în fluxul de mediere.

Implementarea modulelor de mediere

Modulele de mediere sunt create folosind Integration Designer, şi sunt implementate în general pe IBM Business Process Manager înăuntrul unui fişier EAR (enterprise archive).

Aveţi posibilitatea să modificaţi valoarea proprietăţilor promovate în momentul implementării.

Puteţi exporta un modul de mediere din Integration Designer, şi să determinaţi ca Integration Designer să împacheteze modulul de mediere într-un fişier JAR (Java archive), iar acest fişier JAR într-un fişier EAR. Apoi, aveţi posibilitatea să implementaţi apoi fişierul EAR prin instalarea unei noi aplicaţii din consola administrativă.

Modulele de mediere pot fi gândite ca o singură entitate. Totuşi, modulele SCA sunt definite de un număr de fişiere XML memorate într-un fişier JAR.

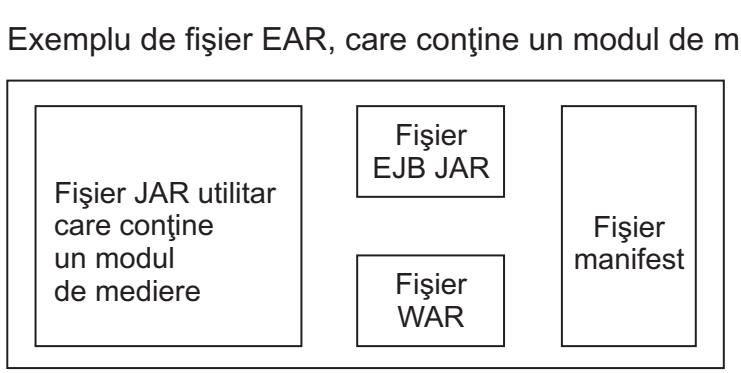


Figura 44. Un exemplu simplificat de fişier EAR care conţine un modul de mediere. Fişierul EAR conţine JAR-uri. Fişierul JAR utilitar conţine un modul de mediere.

Primitive de mediere:

Componentele fluxului de mediere operează asupra fluxurilor de mesaje între componentelor serviciului. Capabilităţile componentei de mediere sunt implementate prin *primitivele de mediere*, care implementează tipuri de implementare pentru serviciul standard.

O componentă a fluxului de mediere are una sau mai multe fluxuri. De exemplu, unul pentru cerere şi unul pentru răspuns.

IBM Business Process Manager suportă un set livrat de primitive de mediere, care implementează capabilităţile standard de mediere pentru modulele de mediere sau pentru modulele implementate în IBM Business Process Manager. Dacă aveţi nevoie de anumite capabilităţi de mediere, puteţi să vă dezvoltaţi propriile primitive personalizate de mediere.

O primitivă de mediere defineşte o operaţie de tip “in” care procesează sau manipulează mesajele care sunt reprezentate de SMO-uri (service message objects). O primitivă de mediere poate defini de asemenea o operaţie de tip “out” care trimite mesaje la o altă componentă sau la un alt modul.

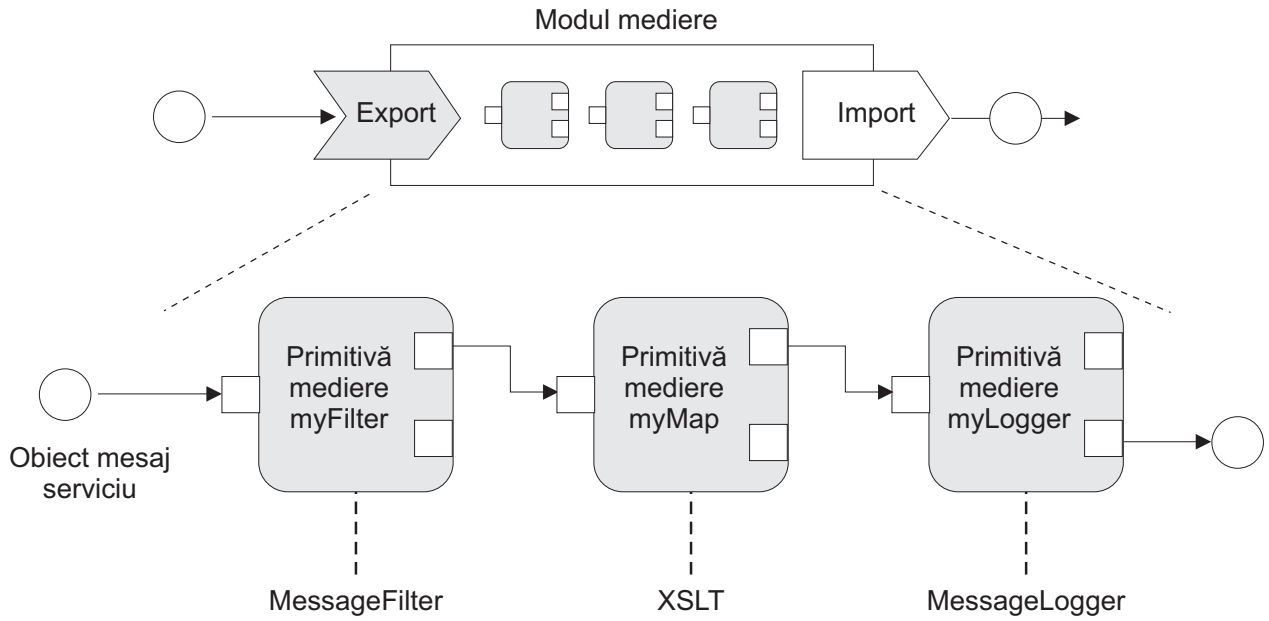


Figura 45. Modulul de mediere care conţine trei primitive de mediere

Puteţi folosi Integration Designer pentru a configura primitivele de mediere şi pentru a le seta proprietăţile. Unele dintre aceste proprietăţi pot fi făcute vizibile pentru administratorul de runtime prin promovarea acestora. Oricare dintre proprietăţile primitivelor de mediere care pot fi promovate pot fi de asemenea proprietăţi dinamice. O proprietate dinamică poate fi înlocuită în momentul execuţiei prin folosirea unui fişier de politică.

Integration Designer vă permite de asemenea să modelaţi grafic şi să asamblaţi componentele fluxului de mediere din cadrul primitivei de mediere, şi să asamblaţi module de mediere sau module din cadrul componentelor fluxului de mediere. Consola administrativă se referă la modulele de mediere şi la modulele ca la module SCA.

Integration Designer permite de asemenea definirea de subfluxuri în module sau a bibliotecilor lor dependente. Un subflux poate conţine orice primitivă de mediere, cu excepţia primitivei Policy Resolution. Un subflux este invocat din fluxul de cerere sau de răspuns, sau dintr-un alt subflux folosind primitiva de mediere Subflux. Proprietăţile promovate într-un sublux prin primitivele de mediere sunt expuse ca proprietăţi în primitivele de mediere Subflow. Apoi, acestea pot fi promovate din nou până când ajung nivelul modulului, moment în care acestea pot fi modificate de administratorul de runtime.

Primitive de mediere suportate

Următorul set de primitive de mediere este suportat de IBM Business Process Manager:

Business Object Map
Transformă mesaje.

- Defineşte transformări pentru mesaje folosind o mapare pentru obiecte business care poate fi refolosită.
- Vă permite să definiţi transformări pentru mesaje în mod grafic folosind editorul pentru maparea cu obiecte business.
- Poate modifica conţinutului unui mesaj.
- Poate transforma tipul unui mesaj de intrare într-un tip diferit de mesaj de ieşire.

Custom Mediation
Vă permite să implementaţi propria logică de mediere în codul Java. Primitiva de mediere personalizată combină flexibilitatea unei primitive de mediere definite de utilizator cu simplitatea unei primitive de mediere predefinite. Puteţi crea transformări complexe şi tipare de rutare astfel:

- Crearea de cod Java.
- Crearea propriilor proprietăţi.
- Adăugarea de terminale noi.

Puteţi apela un serviciu dintr-o primitivă de mediere personalizată, dar primitiva de mediere Service Invoke este proiectată să apeleze servicii şi să asigure funcţionalităţi suplimentare, precum reîncercarea.

Data Handler

Vă permite să modificaţi o parte a mesajului. Este folosit pentru a converti un element al mesajului din format fizic într-o structură logică sau o structură logică într-un forma fizic. Utilizarea primară a primitivei este de a converti un format fizic, precum un şir text dintr-un obiect JMS Text Message, într-o structură de tip Obiect business şi din nou înapoi. Această mediere este frecvent utilizată pentru a:

- Transformarea unei secţiuni a mesajului de intrare dintr-o structură definită în alta - un exemplu de acest fel ar fi cazul în care SMO include o valoare şir care este delimitată prin virgulă şi vreţi să o parsaţi într-un Obiect business specific.
- Altera tipul mesajului – un exemplu ar fi atunci când un export JMS a fost configurat pentru a folosi legare de date de bază JMS, iar în cadrul modulului de mediere dezvoltatorul responsabil cu integrarea decide că conţinutul ar trebui trecut la o anumită structură de BO.

Database Lookup

Modifică mesaje, utilizând informaţii dintr-o bază de date ce sunt furnizate de utilizator.

- Trebuie să configuraţi o bază de date, o sursă de date, precum şi orice setări de autentificare necesare serverului utilizate de primitiva de mediere Database Lookup. Pentru a face acest lucru, ajutaţi-vă de consola administrativă.
- Primitiva de mediere Database Lookup poate citi dintr-o singură tabelă.
- Coloana index specificată trebuie să conţină o valoare unică.
- Datele din coloanele ce conţin valori trebuie să fie ori de un tip schemă XML simplă, fie de un tip schemă XML care extinde tipul de schemă XML simplă.

Endpoint Lookup

Permite rutarea dinamică a cererilor prin căutarea în magazie a punctelor finale de servicii.

- Informaţiile legate de punctul final al serviciului sunt primite de la un WSRR (WebSphere Service Registry and Repository). Registrul WSRR poate fi local sau poate fi la distanţă.
- Faceţi modificările asupra registrului din consola administrativă WSRR.
- IBM Business Process Manager trebuie să ştie ce registru să folosească, şi de aceea, trebuie să creaţi definiţii de acces WSRR folosind consola administrativă IBM Business Process Manager.

Event Emitter

Îmbunătăţeşte monitorizarea prin permiterea trimiterii evenimentelor din interiorul unei componente a fluxului de mediere.

- Puteţi suspenda acţiunea de mediere prin debifarea casetei de bifare.
- Puteţi vizualiza evenimente Event Emitter utilizând browser-ul Common Base Events în IBM Business Process Manager.
- Ar trebui să trimiteţi evenimente doar către un singur punct important într-un flux de mediere, din motive de performanţă.
- Aveţi posibilitatea să definiţi părţile mesajului conţinut de eveniment.
- Evenimentele sunt trimise în formatul Common Base Events şi sunt trimise la un server Common Event Infrastructure.
- Pentru a utiliza pe deplin informaţiile legate de Event Emitter, consumatorii de evenimente trebuie să înţeleagă structura evenimentelor CBE (Common Base Events). Evenimentele CBE au o schemă generală, dar acest lucru nu modelează datele specifice aplicaţiei care sunt conţinute în elementele extinse de date. Pentru a modela elementele extinse de date, uneltele Integration Designer generează un fişier cu definiţii pentru catalogul de evenimente Common Event Infrastructure pentru fiecare dintre primitivele de mediere Event Emitter configurate. Fişierele de definire a catalogului de evenimente sunt artefacte exportate care

sunt furnizate pentru ajutor; nu sunt folosite de runtime-ul Integration Designer sau IBM Business Process Manager. Ar trebui să faceţi referire la fişierele cu definiţii pentru catalogul de evenimente atunci când creaţi aplicaţii care consumă evenimente Event Emitter.

- Puteţi specifica o altă monitorizare din IBM Business Process Manager. De exemplu, puteţi controla ca evenimentele să fie emise din importuri şi exporturi.

Fail Opreşte o anumită cale din flux şi generează o excepţie.

Fan In Ajută la agregarea (combinarea) mesajelor.

- Poate fi folosită doar în combinaţie cu primitiva de mediere Fan Out.
- Împreună, primitivele de mediere Fan Out şi Fan In permit agregarea datelor într-un singur mesaj de ieşire.
- Primitiva de mediere Fan In primeşte mesaje până când se ajunge la un punct de decizie, moment în care este trimis la ieşire un mesaj.
- Contextul partajat ar trebui folosit pentru reţinerea datelor agregate.

Fan Out

Ajută la divizarea şi agregarea (combinarea) mesajelor.

- Împreună, primitivele de mediere Fan Out şi Fan In permit agregarea datelor într-un singur mesaj de ieşire.
- În modul iterativ, primitiva de mediere Fan Out vă permite să iteraţi prin un singur mesaj de intrare care conţine un element repetitiv. Pentru fiecare apariţie a elementului repetitiv este trimis un mesaj.
- Contextul partajat ar trebui folosit pentru reţinerea datelor agregate.

HTTP Header Setter

Oferă un mecanism pentru gestionarea anteturilor în mesajele HTTP.

- Poate crea, seta, copia sau şterge anteturile mesajelor HTTP.
- Poate seta mai multe acţiuni pentru a modifica mai multe anteturi HTTP.

Mapping

Transformă mesaje.

- Vă permite să realizaţi transformări XSL (Extensible Stylesheet Language) sau transformări Business Object Map.
- Puteţi transforma mesajele folosind o transformare XSLT 1.0 sau XSLT 2.0 sau o transformare Business Object Map. Transformările XSL operează pe o serializare XML a mesajului, în timp ce transformarea Business Object Map operează asupra SDO (Service Data Objects).

Message Element Setter

Oferă un mecanism simplu pentru setarea conţinutului mesajelor.

- Poate modifica, adăuga sau şterge elemente din mesaj.
- Nu modifică tipul de mesaj.
- Datele din coloanele ce conţin valori trebuie să fie ori de un tip schemă XML simplă, fie de un tip schemă XML care extinde tipul de schemă XML simplă.

Message Filter

Rutează mesaje pe căi diferite, în funcţie de conţinutul acestora.

- Puteţi suspenda acţiunea de mediere prin debifarea casetei de bifare.

Message Logger

Înregistrează mesajele într-o bază de date relaţională sau prin intermediul propriului jurnalizator personalizat. Mesajele sunt memorate sub formă de XML, şi de aceea, datele pot fi procesate după aceea de aplicaţiile care suportă XML.

- Puteţi suspenda acţiunea de mediere prin debifarea casetei de bifare.
- Schema bazei de date relaţionale (structura tabelei) este definită de IBM.
- În mod implicit, primitiva de mediere Message Logger foloseşte baza de date Common. Mediul runtime mapează sursa de date la **jdbc/mediation/messageLog** pe baza de date Comună.

- Puteţi seta clasele de implementare Handler pentru a personaliza comportamentul jurnalizatorului personalizat. Opţional, puteţi furniza clase de implementare Formatter, clase de implementare Filter, sau ambele pentru a personaliza comportamentul jurnalizatorului personalizat.

MQ Header Setter

Oferă un mecanism pentru gestionarea anteturilor în mesajele MQ.

- Poate crea, seta, copia sau şterge anteturile mesajelor MQ.
- Poate seta mai multe acţiuni pentru a modifica mai multe anteturi MQ.

Policy Resolution

Permite configurarea dinamică a cererilor, prin căutarea punctelor finale ale serviciului şi a fişierelor de politică asociate, într-o magazie.

- Aveţi posibilitatea să utilizaţi un fişier de politică pentru a înlocui dinamic proprietăţile promovate ale altor primitive de mediere.
- Informaţiile punctului final al serviciului şi informaţiile legate de politică sunt extrase dintr-un WSRR (WebSphere Service Registry and Repository). Registrul WSRR poate fi local sau poate fi aflat la distanţă.
- Faceţi modificările asupra registrului din consola administrativă WSRR.
- IBM Business Process Manager trebuie să ştie ce registru să folosească, şi de aceea, trebuie să creaţi definiţii de acces WSRR folosind consola administrativă IBM Business Process Manager.

Service Invoke

Apelează un serviciu din interiorul unui flux de mediere, mai degrabă decât să aştepte până la sfârşitul unui flux de mediere şi folosirea mecanismului callout.

- Dacă serviciul returnează un defect, puteţi reîncerca acelaşi serviciu sau să apelaţi un alt serviciu.
- Primitiva de mediere Service Invoke este una puternică care poate fi folosită pe cont propriu pentru apeluri simple ale serviciului, sau în combinaţie cu alte primitive de mediere pentru medieri mai complexe.

Set Message Type

În timpul dezvoltării de integrare, vă permite să trataţi câmpurile mesajului care sunt tipizate slab, ca şi pe cele care sunt tipizate tare. Un câmp este tipizat slab dacă poate conţine mai multe tipuri de date. Un câmp este tipizat tare dacă tipul şi structura sa internă sunt cunoscute.

- La runtime, medierea Setare tip mesaj permite verificarea potrivirii între conţinutul unui mesaj şi tipurile de date pe care le aşteptaţi.

SOAP Header Setter

Oferă un mecanism pentru gestionarea anteturilor în mesajele SOAP.

- Poate crea, seta, copia sau şterge anteturile mesajelor SOAP.
- Poate seta mai multe acţiuni pentru a modifica mai multe anteturi SOAP.

Stop Opreşte o anumită cale din flux fără a genera o excepţie.

Type Filter

Vă permite să direcţionaţi mesajele pe o cale diferită a fluxului, în funcţie de tipul acestora.

Extragere WebSphere eXtreme Scale

Puteţi extrage informaţii dintr-un mediu cache server eXtreme Scale.

- Puteţi căuta valori în cache şi le puteţi păstra ca elemente în mesaj folosind un index.
- Combinând primitivele de mediere Stocare şi Extragere eXtreme Scale, puteţi memora în cache răspunsul de la un sistem back-end. Cererile viitoare nu vor necesita acces la acel sistem de back-end.
- Trebuie să creaţi definiţii eXtreme Scale utilizând consola administrativă WebSphere ESB, astfel încât puteţi specifica serverul eXtreme Scale de utilizat.

WebSphere eXtreme Scale Store

Puteţi stoca informaţii într-un mediu cache server eXtreme Scale.

- Puteţi stoca informaţii într-un cache eXtreme Scale utilizând un index sau un obiect.

- Combinând primitivele de mediere eXtreme Scale Store şi Retrieve, puteţi folosi primitiva de mediere Store pentru stocarea datelor în memoria cache şi puteţi folosi primitiva de mediere Retrieve pentru a extrage date stocate anterior în memoria cache.
- Trebuie să creaţi definiţii eXtreme Scale utilizând consola administrativă WebSphere ESB, astfel încât puteţi specifica serverul eXtreme Scale de utilizat.

Rutarea dinamică:

Puteţi ruta mesaje în diverse moduri utilizând punctele finale definite la timpul integrării sau puncte finale determinate, dinamic, la momentul rulării.

Rutarea dinamică acoperă două cazuri de rutare a mesajelor:

- Rutarea mesajelor în cazul în care fluxul este dinamic, dar toate punctele finale posibile sunt predefinite într-un modul SCA (Service Component Architecture).
- Rutarea mesajelor în cazul în care fluxul este dinamic şi selecţia punctului final este, de asemenea, dinamică. Punctele finale ale serviciului sunt selectate dintr-o sursă externă la momentul rulării

Selectarea dinamică a punctelor finale

Momentul execuţiei are capabilitatea de a ruta mesajele de cerere şi de răspuns către o adresă finală identificată printr-un element din antetul mesajului. Acest element din antetul mesajului poate fi actualizat prin intermediul primitivelor de mediere, într-un flux de mediere. Adresa finală ar putea fi actualizată cu informaţii dintr-un registru, o bază de date, sau cu informaţii din mesajul în sine. Rutarea unui mesaj de răspuns se aplică doar atunci când răspunsul este trimis de un export JAX-WS al serviciului web.

Pentru ca la momentul execuţiei să fie implementată rutarea dinamică pentru o cerere sau pentru un răspuns, modulul SCA trebuie să aibă setată proprietatea Utilizare punct final dinamic dacă este setat în antetul mesajului. Dezvoltatorii care se ocupă cu integrarea pot seta proprietatea >Utilizare punct final dinamic dacă este setat în antetul mesajului sau o pot promova (o fac vizibilă la momentul execuţiei), astfel încât administratorul din timpul rulării să o poată seta. Aveţi posibilitatea să vizualizaţi proprietăţile modulului în fereastra Proprietăţi modul. Pentru a vedea fereastra, faceţi clic pe **Aplicaţii > Module SCA > Proprietăţi Modul**. Dezvoltatorul care se ocupă cu integrarea oferă proprietăţilor promovate aliasuri, iar acestea sunt numele afişate în consola administrativă.

Registru

Puteţi folosi WSRR (IBM WebSphere Service Registry and Repository) pentru a memora informaţiile legate de punctul final al serviciului, iar apoi să creaţi module SCA pentru a extrage puncte finale din magazia WSRR.

Atunci când dezvoltaţi module SCA, folosiţi primitiva de mediere Endpoint Lookup pentru a permite unui flux de mediere să interogheze un registru WSRR pentru a obţine un punct final pentru serviciu sau un set de puncte finale pentru serviciu. Dacă un modul SCA modul extrage un set de puncte finale atunci acesta trebuie să folosească o altă primitivă de mediere pentru a o selecta pe cea preferată.

Controlul politicii de mediere a cererilor de servicii:

Puteţi utiliza politici de mediere pentru a controla fluxurile de mediere dintre solicitanţii serviciului şi furnizorii de servicii.

Puteţi controla fluxurile de mediere utilizând politici de mediere memorate în IBM WebSphere Service Registry and Repository (WSRR). Implementarea gestiunii politicii serviciului în WSRR este bazată pe WS-Policy (Web Services Policy Framework).

Pentru a controla cererile de servicii utilizând politici de mediere, trebuie să aveţi module SCA (Service Component Architecture) dorite şi documente ale politicii de mediere în registrul dumneavoastră WSRR.

Cum se face ataşarea unei politici de mediere la o cerere de serviciu

Atunci când dezvoltaţi un modul SCA care trebuie să facă uz de o politică de mediere, trebuie să includeţi în fluxul de mediere o primitivă de mediere Policy Resolution. În timpul rulării, primitiva de mediere Policy Resolution obţine informaţii despre politica de mediere din registru. Prin urmare, un modul SCA trebuie să conţină o componentă flux de mediere pentru a suporta controlul politicii de mediere a cererilor de servicii.

În registru, puteţi ataşa una sau mai multe politici de mediere la un modul SCA sau la un serviciu ţintă utilizat de către modulul SCA. Politicile de mediere ataşate pot fi utilizate (sunt în domeniu) pentru toate mesajele serviciului procesate de către acel modul SCA. Politicile de mediere pot avea ataşamente de politică care definesc condiţii. Condiţiile politicii de mediere permit diferitelor politici de mediere să se aplice în diferite contexte. În plus, politicile de mediere pot avea clasificări care pot fi utilizate pentru a specifica o stare de guvernare.

WebSphere Service Registry and Repository:

Produsul WSRR (WebSphere Service Registry and Repository) vă permite să memoraţi, accesaţi şi să gestionaţi informaţiile legate de punctele finale ale serviciului şi de politicile de mediere. Puteţi utiliza WSRR pentru a face ca aplicaţiile serviciului dvs. să fie mult mai dinamice şi mult mai adaptabile la modificările aduse condiţiilor de business.

Introducere

Fluxurile de mediere pot folosi WSRR pe post de maşină de căutare dinamică, oferind informaţii legate de punctele finale sau de politicile de mediere ale serviciului.

Pentru a configura accesul la WSRR, creaţi documente de definiţie WSRR folosind consola administrativă. Alternativ, aveţi posibilitatea să utilizaţi comenzile de administrare WSRR din clientul script wsadmin. Definiţiile WSRR şi proprietăţile acestora de conexiune reprezintă mecanismul de conectare la o instanţă a unui registru, şi de extragere a punctului final sau a politicii de mediere pentru un serviciu.

Puncte finale pentru serviciu

Puteţi folosi WSRR pentru a memora informaţii despre serviciile pe care le folosiţi deja, pentru cele pe care plănuiţi să le folosiţi sau pentru cele de care vreţi să fiţi conştient. Aceste servicii pot fi în sistemele dvs, sau în ale sisteme. De exemplu, aplicaţia ar putea folosi WSRR la localizarea celui mai corespunzător serviciu care îi satisface nevoile funcţionale şi de performanţă.

Atunci când dezvoltaţi un modul SCA care are nevoie să acceseze punctele finale ale serviciului din WSRR, trebuie să includeţi o primitivă de mediere Endpoint Lookup în fluxul de mediere. În momentul rulării, primitiva de mediere Endpoint Lookup obţine punctele finale ale serviciului din registru.

Politici de mediere

De asemenea, puteţi folosi WSRR pentru a memora informaţii legate de politica de mediere. Politicile de mediere vă pot ajuta să controlaţi cererile de servicii prin suprascrierea dinamică a proprietăţilor modulului. Dacă WSRR conţine politici de mediere care sunt ataşate la un obiect care reprezintă fie modulul dvs. SCA, fie serviciul dvs. ţintă, atunci politicile de mediere ar putea suprascrie proprietăţile modulului. Dacă doriţi ca în diferite contexte să se aplice politici de mediere diferite, puteţi crea condiţii pentru aceste politici.

Notă: Politicile de mediere sunt preocupate de controlul fluxurilor de mediere, nu de securitate.

Atunci când dezvoltaţi un modul SCA care trebuie să facă uz de o politică de mediere, trebuie să includeţi în fluxul de mediere o primitivă de mediere Policy Resolution. În momentul rulării, primitiva de mediere Policy Resolution obţine informaţiile legate de politica de mediere din registru.

WebSphere eXtreme Scale:

Utilizând produsul WebSphere eXtreme Scale (eXtreme Scale) puteţi asigura un sistem de memorare în cache pe care îl puteţi integra cu o aplicaţieIBM Business Process Manager. Utilizând eXtreme Scale cu IBM Business Process Manager puteţi îmbunătăţi timpul de răspuns şi fiabilitatea serviciului şi asigura funcţionalitate de integrare suplimentară.

eXtreme Scale acţionează ca o grilă de date elastică, scalabilă, în memorie. Grila de date memorează în cache, partiţionează, replică şi gestionează dinamic datele aplicaţiei şi logica operaţională pe mai multe servere. Cu eXtreme Scale, puteţi, de asemenea, să obţineţi calitate a serviciilor precum integritate tranzacţională, disponibilitate înaltă şi timpi de răspuns predictibili.

Puteţi utiliza fluxuri de mediere pentru a accesa funcţia de memorare în cache eXtreme Scale inclusiv primitivele de mediere WebSphere eXtreme Scale din fluxul dumneavoastră. La dezvoltarea unui modul SCA (Service Component Architecture) care are nevoie să stocheze informaţii într-o memorie cache eXtreme Scale, trebuie să includeţi primitiva de mediere WebSphere eXtreme Scale Store în fluxul de mediere. Dacă vreţi să extrageţi informaţii dintr-o memorie cache eXtreme Scale, trebuie să includeţi primitiva de mediere WebSphere eXtreme Scale Retrieve. Combinând cele două primitive de mediere într-un flux de mediere, puteţi memora în cache răspunsul de la un sistem back-end, astfel încât cererile viitoare să poată extrage răspunsul din cache.

Pentru a configura accesul la eXtreme Scale, trebuie să creaţi o definiţie WebSphere eXtreme Scale utilizând consola administrativă. Alternativ, puteţi utiliza comenzile de administrare WebSphere eXtreme Scale din clientul de scriptare wsadmin. O definiţie eXtreme Scale este mecanismul utilizat de primitivele de mediere WebSphere eXtreme Scale Retrieve şi Store pentru conectarea la un server eXtreme Scale.

Clienţi ai serviciului de mesaje

Clienţii pentru Serviciul de mesagerie sunt disponibili pentru C/C++ şi .NET pentru a permite aplicaţiilor care nu sunt de tip Java să se conecteze la magistrala ESB (Enterprise Service Bus).

Message Service Clients for C/C++ and .NET oferă un API numit XMS care are acelaşi set de interfeţe ca şi API-ul JMS (Java Message Service). Message Service Client for C/C++ conţine două implementări ale XMS, una de folosit de aplicaţiile C şi una de folosit de aplicaţiile C++. Message Service Client for .NET conţine o implementare complet gestionată a XMS, care poate fi folosită de orice limbaj compatibil cu .NET.

Puteţi obţine Clienţi ai Serviciului de Mesaje pentru .NET din http://www-01.ibm.com/support/docview.wss?rs=0&q1=IA9H&uid=swg24011756&loc=en_US&cs=utf-8&cc=us&lang=en

Puteţi obţine Clienţi ai Serviciului de Mesaje pentru C/C++ din http://www-01.ibm.com/support/docview.wss?rs=0&q1=ia94&uid=swg24007092&loc=en_US&cs=utf-8&cc=us&lang=en.

Puteţi instala şi folosi şi suportul pentru clientul Java EE de la WebSphere Application Server Network Deployment, inclusiv client servicii web, EJB Client, şi JMS Client.

Capitolul 2. Aflaţi mai multe despre conceptele cheie

Utilizaţi această secţiune ca un punct de plecare pentru investigarea tehnologiilor utilizate în şi de către IBM Business Process Manager.

Crearea scenariilor

Folosiţi scenarii pentru a înţelege şi lucra cu componentele şi produsele din familia de management al proceselor operaţionale.

Versionarea

Ciclul de viaţă al unei aplicaţii de proces începe cu crearea aplicaţiei de proces şi continuă cu un ciclu de actualizare, implementare, co-implementare, dezimplementare şi arhivare a aplicaţiei de proces. *Versionarea* este un mecanism folosit pentru gestionarea ciclului de viaţă al aplicaţiei de proces prin identificarea în mod unic a versiunilor individuale ale aplicaţiei de proces.

Modul în care versionarea funcţionează în IBM Business Process Manager depinde de ceea ce implementaţi –o aplicaţie de proces, implementată din magazie în IBM Process Center sau o aplicaţie de întreprindere implementată direct din IBM Integration Designer.

Aplicaţiile de proces şi trusele de unelte pe care le implementaţi într-un mediu de runtime din Process Center sunt versionate în mod implicit. Pentru aplicaţiile de întreprindere, puteţi alege să versionaţi modulele şi bibliotecile în IBM Integration Designer.

În plus, puteţi crea versiuni pentru un task uman sau pentru o stare a unei maşini, astfel încât să poate exista simultan mai multe versiuni ale taskului sau ale stării maşinii în mediul de runtime.

Versionarea aplicaţiilor de proces

Versionarea furnizează abilitatea, pentru mediul runtime, de a identifica instantaneele din ciclul de viaţă al unei aplicaţii proces şi de a fi capabil să ruleze simultan mai multe instantanee pe un server de proces.

Pentru a înţelege cum sunt versionate aplicaţiile proces, este important să vă amintiţi că o aplicaţie proces este un container care reţine diverse artefacte utilizate în sau de către aplicaţia proces (de exemplu, modele de procese sau BPD-uri, referinţe trusă de unelte, servicii, piste sau modele de monitor). Orice versionare este făcută la acest nivel de container, nu la nivelul artefactelor individuale. Pentru aplicaţii proces, aceasta înseamnă că versionarea apare când realizaţi un instantaneu.

Puteţi compara instantaneele pentru a determina diferenţele dintre versiuni. De exemplu, dacă un dezvoltator a corectat o problemă cu un serviciu şi a realizat un instantaneu al aplicaţiei de proces sau al trusei sale de unelte în acel moment, şi apoi un alt dezvoltator a făcut mai multe modificări suplimentare la acelaşi serviciu şi a realizat un nou instantaneu, managerul de proiect poate compara cele două instantanee pentru a determina ce modificări au fost făcute, unde şi de către cine. Dacă managerul de proiect a decis că modificările suplimentare la serviciu nu au valoare, managerul de proiect poate reveni la instantaneul corecţiei iniţiale.

Puteţi rula versiuni diferite (instantanee) ale unei aplicaţii de proces simultan pe un server; când instalaţi un instantaneu nou, fie înlăturaţi originalul fie lăsaţi-l să ruleze.

Context de versiune

Fiecare instantaneu are metadate unice pentru a identifica versiunea (referite ca şi context de versiune). Aţi alocat acel identificator, dar IBM recomandă folosirea unui sistem pe trei cifre în versiunea numerică în formatul <major>.<minor>.<service>. Vedeţi toate subiectele despre convenţiile de numire pentru descrieri mai detaliate ale acestei scheme de versionare.

IBM Business Process Manager asignează un spaţiu de nume global pentru fiecare aplicaţie de proces. Spaţiul de nume global este în mod specific fie sugestia aplicaţiei proces, fie un instantaneu de aplicaţie proces particular. Numele de versiune utilizat de către server nu poate fi mai lung de şapte caractere, astfel încât numele alocat este un acronim care utilizează caractere din numele instantaneului pe care l-aţi alocat. Acronimele instantaneului sunt identice cu numele instantaneului conform stilului IBM VRM recomandat şi nu sunt mai lungi de şapte caractere. De exemplu, un nume de instantaneu de 1.0.0 va avea acronimul 1.0.0 şi un nume de instantaneu of 10.3.0 va avea acronimul 10.3.0. Acronimul instantaneului va fi garantat ca şi unicitate în contextul aplicaţiei de proces din domeniul serverului Process Center. Pentru acest motiv, nu puteţi edita acronimul instantaneului.

Considerente de versionare pentru aplicaţii proces în clustere multiple

Puteţi instala aceeaşi versiune de aplicaţie proces la clustere multiple din aceeaşi celulă. Pentru a face diferenţa între aceste instalări multiple ale aceleiaşi versiuni a aplicaţie proces, creaţi un instantaneu pentru fiecare instalare şi includeţi un ID unic de celulă în numele instantaneului (de exemplu, v1.0_cell1_1 şi v1.0_cell1_2). Fiecare instantaneu este o versiune nouă a aplicaţiei proces (dintr-o perspectivă de gestiune ciclu de viaţă pură), dar conţinutul şi funcţia sunt la fel.

Când instalaţi o aplicaţie proces într-un cluster, este realizată o sincronizare automată a nodurilor.

Considerente de versionare pentru truse de unelte Process Designer

Amintiţi-vă că instantaneele de aplicaţii proces sunt realizate în mod tipic când sunteţi pregătit să testaţi sau să instalaţi. Instantaneele de truse de unelte sunt însă realizate în mod tipic când sunteţi pregătit ca acea trusă de unelte să fie utilizată de aplicaţii proces. După, dacă doriţi să actualizaţi trusa de unelte, trebuie să faceţi un alt instantaneu al "sugestiei" atunci când sunteţi pregătit şi apoi deţinătorii aplicaţiilor de proces şi ai truselor de unelte pot decide dacă doresc să treacă la noul instantaneu.

Versionarea modulelor şi bibliotecilor

Dacă un modul sau o bibliotecă este într-o aplicaţie de proces sau într-o trusă de unelte, acesta continuă ciclul de viaţă al aplicaţiei de proces sau trusei de unelte (versiuni, instantanee, urme şi aşa mai departe). Numele modulelor şi bibliotecilor trebuie să fie unice în cadrul domeniului unei aplicaţii de proces sau a unei truse de unelte.

Acest subiect descrie versionarea modulelor şi a bibliotecilor care sunt folosite împreună cu aplicaţiile de proces. Reţineţi totuşi că în cazul în care implementaţi modulele direct din IBM Integration Designer pe Process Server, puteţi continua să urmaţi procedura de asignare a numerelor de versiune modulelor în timpul implementării, aşa cum este descris în “Crearea modulelor şi bibliotecilor cu număr de versiune”.

Un modul sau o bibliotecă care este asociată cu IBM Process Center trebuie să aibă bibliotecile dependente în aceeaşi aplicaţie de proces sau într-o trusă de unelte dependentă.

Următorul tabel listează selecţiile pe care le puteţi face în editorul de dependenţe din IBM Integration Designer, atunci când o bibliotecă este asociată cu o aplicaţie de proces sau o trusă de unelte:

Tabela 31. Dependenţele pentru Modul, Aplicaţie de proces sau Trusa de unelte şi bibliotecile globale

Domeniu bibliotecă	Descriere	Poate depinde de . . .
Modul	Pe server există câte o copie a acestei biblioteci pentru fiecare modul care o foloseşte.	O bibliotecă din domeniul modulului poate depinde de toate tipurile de biblioteci.

Tabela 31. Dependenţele pentru Modul, Aplicaţie de proces sau Trusa de unelte şi bibliotecile globale (continuare)

Domeniu bibliotecă	Descriere	Poate depinde de . . .
Aplicaţie de proces sau Trusă de unelte	Biblioteca este partajată între toate modulele din domeniul aplicaţiei de proces a trusei de unelte. Această setare are efect în cazul în care implementarea se realizează prin IBM Process Center. În cazul în care implementarea are loc în afara IBM Process Center, biblioteca este copiată în fiecare modul. **Notă:** Bibliotecile create în IBM Integration Designer versiunea 8 au un nivel de partajare de **Aplicaţie de proces sau Trusă de unelte** în mod implicit.	O bibliotecă de acest tip poate depinde doar de bibliotecile globale.
Global	Biblioteca este partajată între toate modulele care rulează.	O bibliotecă globală poate depinde doar de alte biblioteci globale. **Notă:** Trebuie să configuraţi o bibliotecă partajată WebSphere pentru a putea implementa biblioteca globală. Vedeţi “Dependenţele modulelor şi bibliotecilor ” pentru informaţii suplimentare.

Module şi biblioteci asociate cu aplicaţii de proces sau cu truse de unelte

Nu aveţi nevoie să versionaţi module şi biblioteci asociate cu aplicaţii de proces sau cu truse de unelte.

Modulele şi bibliotecile asociate cu aplicaţii de proces sau cu truse de unelte nu au nevoie să fie versionate. De fapt, nu puteţi crea o versiune a unui modul sau a unei biblioteci asociate cu aplicaţii de proces sau cu truse de unelte în editorul de dependenţe. Modulele şi bibliotecile asociate cu aplicaţiile de proces sau cu trusele de unelte utilizează instantanee, o funcţie în Process Center, pentru a realiza acelaşi rezultat ca o versiune.

Bibliotecile, asociate cu aplicaţiile de proces sau cu trusele de unelte, nu vor avea un număr de versiune necesar în secţiunea Biblioteci a editorului de dependenţe, deoarece nu este necesară nicio versiune.

Convenţii de numire

O convenţie de numire este folosită pentru a diferenţia diversele versiuni ale unei aplicaţii de proces pe măsură ce se mută prin ciclul de viaţă de actualizare, implementarea, co-implementare, dezimplementare şi arhivare.

Această secţiune vă oferă convenţiile care sunt utilizate pentru a identifica în mod unic versiunile unei aplicaţii de proces.

Un *context de versiune* este o combinaţie de acronime care descriu în mod unic o aplicaţie de proces sau o trusă de unelte. Fiecare tip de acronim are o convenţie de numire. Acronimul este limitată la o lungime maximă de şapte caractere din setul de caractere [A-Z0-9_], cu excepţia acronimului instantaneului, care poate include un punct.

- Acronimul aplicaţiei de proces este creat odată cu aplicaţia. Acesta poate avea o lungime maximă de şapte caractere.
- Acronimul instantaneului este creat în mod automat la crearea instantaneului. Acesta poate avea o lungime maximă de şapte caractere.

  Dacă numele instantaneului îndeplineşte criteriile pentru un acronim valid de instantaneu, numele şi acronimul instantaneului vor fi identice.

  Notă: Atunci când utilizaţi funcţia de rutare dependentă de versiune pentru componenta fluxului de mediere, denumiţi-vă instantaneul, astfel încât să fie în concordanţă cu schema <*versiune*>.<*ediţie*>.<*modificare*> (de exemplu, **1.0.0**). Deoarece acronimul instantaneului este limitat la şapte caractere, valorile cifrelor sunt limitate la un maxim de cinci cifre (cinci cifre plus două perioade). Prin urmare, trebuie să aveţi grijă atunci când câmpurile cifre sunt incrementate deoarece orice depăşeşte primele şapte caractere este trunchiat.

De exemplu, un nume de instantaneu **11.22.33** rezultă într-un acronim de instantaneu egal cu **11.22.3**.

- Acronimul de pistă este generat în mod automat din primul caracter al fiecărui cuvânt al numelui de pistă. De exemplu, o pistă nouă ce are numele **My New Track** ar rezulta în acronimul cu valoarea **MNT**.

  Numele implicite pentru urmărire şi acronim sunt **Main**. Implementarea pe un server IBM Process Center include acronimul pistei în contextul de versionare în cazul în care acesta nu este **Main**.

O definiţie de proces operaţional într-o aplicaţie de proces este identificată de obicei prin acronimul numelui procesului operaţional, acronimul instantaneului şi numele definiţiei procesului operaţional. Alegeţi nume unice pentru definiţiile procesului operaţional ori de câte ori este posibil. Atunci când există nume duplicate, este posibil să întâmpinaţi următoarele probleme:

- S-ar putea să fiţi în imposibilitatea de a expune definiţiile procesului operaţional sub formă de servicii web fără o anumită formă de mediere.
- S-ar putea să fiţi în imposibilitatea de a invoca o definiţie a procesului operaţional creat în IBM Process Designer dintr-un proces BPEL creat în IBM Integration Designer.

Contextul de versionare variază în funcţie de cum este implementată aplicaţia de proces.

Convenţii de numire pentru implementarea pe servere Process Center

Pe serverul IBM Process Center, puteţi implementa un instantaneu al unei aplicaţii de proces, precum şi un instantaneu al unei truse de unelte. În plus, puteţi implementa sugestia unei aplicaţii de proces sau sugestia unei truse de unelte. (O *sugestie* reprezintă versiunea curentă funcţională a aplicaţiei dumneavoastră de proces sau a trusei de unelte.) Contextul de versionare variază în funcţie de tipul implementării.

Pentru aplicaţiile de proces, sugestia aplicaţiei de proces sau instantaneul specific aplicaţiei de proces este utilizat pentru a identifica versiunea în mod unic.

Trusele de unelte pot fi implementate cu una sau mai multe aplicaţii de proces, dar ciclul de viaţă al fiecărei truse de unelte este legat de ciclul de viaţă al aplicaţiei de proces. Fiecare aplicaţie de proces are propria copie a trusei de unelte dependente sau a truselor de unelte implementate pe server. O trusă de unelte implementată nu este partajată între aplicaţiile de proces.

În cazul în care o pistă asociată cu o aplicaţie de proces este numită altfel decât în mod implicit **Main**, acronimul pistei face de asemenea parte din contextul de versiune.

Pentru mai multe informaţii, vedeţi secţiunea “Exemple” la pagina 33, mai târziu în acest subiect.

Instantanee ale aplicaţiei de proces

Pentru implementarea instantaneelor aplicaţiilor de proces, contextul de versiune este o combinaţie între articolele următoare:

- Acronim pentru numele aplicaţiei de proces
- Acronim pentru pista aplicaţiei de proces (în cazul în care se foloseşte altă pistă decât **Main**)
- Acronim pentru instantaneul aplicaţiei de proces

Truse de unelte autonome

Pentru implementarea instantaneelor truselor de unelte, contextul de versionare este o combinaţie între articolele următoare:

- Acronim pentru numele trusei de unelte
- Acronim pentru pista trusei de unelte (în cazul în care se foloseşte altă pistă decât **Main**)
- Acronim pentru instantaneul trusei de unelte

Sugestii

Sugestiile pentru aplicaţia de proces sunt folosite în timpul testării în Process Designer. Acestea pot fi implementate doar pe serverele din Process Center.

Pentru implementările sugestiilor pentru aplicaţia de proces, contextul de versionare este o combinaţie între articolele următoare:

- Acronim pentru numele aplicaţiei de proces
- Acronim pentru pista aplicaţiei de proces (în cazul în care se foloseşte altă pistă decât **Main**)
- "Tip"

Sugestiile pentru trusa de unelte sunt de asemenea folosite în timpul testării iterative în Process Designer. Acestea nu sunt implementate pe un server de producţie.

Pentru implementarea sugestiilor legate de trusa de unelte, contextul de versiune este o combinaţie între articolele următoare:

- Acronim pentru numele trusei de unelte
- Acronim pentru pista trusei de unelte (în cazul în care se foloseşte altă pistă decât **Main**)
- "Tip"

Exemple

Resursele ar trebui să fie numite în mod unic şi identificate extern cu ajutorul contextului de versiune.

- Următorul tabel arată exemple de nume care sunt identificate în mod unic. În acest exemplu, o sugestie a aplicaţiei de proces foloseşte numele de pistă implicit (**Main**):

Tabela 32. Sugestie pentru aplicaţia de proces cu nume implicit pentru pistă

Tip de nume	**Exemplu**
Nume aplicaţie de proces	**Aplicaţie de proces 1**
Acronim pentru numele aplicaţiei de proces	**PA1**
Pistă aplicaţie de proces	**Main**
Acronim pentru urma aplicaţiei de proces	“” (când pista este **Main**)
Instantaneu aplicaţie de proces	
Acronim pentru instantaneul aplicaţiei de proces	**Tip**

Orice modul SCA asociat cu această sugestie legată de aplicaţia de proces include contextul de versiune, aşa cum este descris în tabelul următor:

Tabela 33. Module SCA şi fişiere EAR dependentă de versiune

Nume modul SCA	**Nume dependent de versiune**	**EAR dependent de versiune/nume aplicaţie**
M1	PA1-Tip-M1	PA1-Tip-M1.ear
M2	PA1-Tip-M2	PA1-Tip-M2.ear

- Următorul tabel arată un exemplu de sugestie pentru o aplicaţie de proces care foloseşte un nume neimplicit pentru pistă:

Tabela 34. Sugestie pentru aplicaţia de proces cu nume neimplicit pentru pistă

Tip de nume	**Exemplu**
Nume aplicaţie de proces	**Aplicaţie de proces 1**
Acronim pentru numele aplicaţiei de proces	**PA1**

Tabela 34. Sugestie pentru aplicaţia de proces cu nume neimplicit pentru pistă (continuare)

Tip de nume	Exemplu
Pistă aplicaţie de proces	**Track1**
Acronim pentru pista aplicaţiei de proces	**T1**
Instantaneu aplicaţie de proces	
Acronim pentru instantaneul aplicaţiei de proces	**Tip**

Orice modul SCA asociat cu această sugestie legată de aplicaţia de proces include contextul de versiune, aşa cum este descris în tabelul următor:

Tabela 35. Module SCA şi fişiere EAR dependentă de versiune

Nume modul SCA	Nume dependent de versiune	EAR dependent de versiune/nume aplicaţie
M1	PA1-T1-Tip-M1	PA1-T1-Tip-M1.ear
M2	PA1-T1-Tip-M2	PA1-T1-Tip-M2.ear

Convenţii de numire similare se aplică la implementările avansate de instantanee şi sugestii ale trusei de unelte. Ele se aplică şi instantaneelor avansate instalate pe Process Server.

- Următorul tabel arată exemple de nume care sunt identificate în mod unic. În acest exemplu, un instantaneu de aplicaţie de proces foloseşte numele de pistă implicit (**Main**):

Tabela 36. Instantaneu pentru aplicaţia de proces cu nume implicit pentru pistă

Tip de nume	Exemplu
Nume aplicaţie de proces	**Aplicaţie de proces 1**
Acronim pentru numele aplicaţiei de proces	**PA1**
Urmă aplicaţie de proces	**Main**
Acronim pentru urma aplicaţiei de proces	"" (când pista este **Main**)
Instantaneu aplicaţie de proces	**Process Shapshot V1**
Acronim pentru instantaneul aplicaţiei de proces	**PSV1**

Toate modulele SCA asociate cu această aplicaţie de proces includ contextul versiunii, aşa cum se arată în următorul tabel:

Tabela 37. Module SCA şi fişiere EAR dependentă de versiune

Nume modul SCA	Nume dependent de versiune	EAR dependent de versiune/nume aplicaţie
M1	PA1-PSV1-M1	PA1-PSV1-M1.ear
M2	PA1-PSV1-M2	PA1-PSV1-M2.ear

- Următorul tabel arată un exemplu de instantaneu de aplicaţie de proces care foloseşte un nume de pistă neimplicit:

Tabela 38. Instantaneu pentru aplicaţia de proces cu nume neimplicit pentru pistă

Tip de nume	Exemplu
Nume aplicaţie de proces	**Aplicaţie de proces 1**
Acronim pentru numele aplicaţiei de proces	**PA1**
Urmă aplicaţie de proces	**Track1**
Acronim pentru urma aplicaţiei de proces	**T1**
Instantaneu aplicaţie de proces	**Process Snapshot V1**
Acronim pentru instantaneul aplicaţiei de proces	**PSV1**

Toate modulele SCA asociate cu această aplicaţie de proces includ co ntextul versiunii, aşa cum se arată în următorul tabel:

Tabela 39. Module SCA şi fişiere EAR dependentă de versiune

Nume modul SCA	Nume dependent de versiune	EAR dependent de versiune/nume aplicaţie
M1	PA1-T1-PSV1-M1	PA1-T1-PSV1-M1.ear
M2	PA1-T1-PSV1-M2	PA1-T1-PSV1-M2.ear

Convenţii de numire pentru implementarea Process Server

Pe un Process Server, puteţi implementa instantaneul unei aplicaţii de proces. Acronimul instantaneului aplicaţiei de proces este folosit pentru a identifica versiunea în mod unic.

Pentru implementarea instantaneelor aplicaţiilor de proces, contextul de versionare este o combinaţie între articolele următoare:

- Acronim pentru numele aplicaţiei de proces
- Acronim instantaneu pentru aplicaţia de proces

Resursele ar trebui să fie numite în mod unic şi identificate extern cu ajutorul contextului de versionare. Următorul tabel arată exemple de nume care sunt identificate în mod unic:

Tabela 40. Exemple de nume şi acronime

Tip de nume	Exemplu
Nume aplicaţie de proces	**Aplicaţie de proces 1**
Acronim pentru numele aplicaţiei de proces	**PA1**
Instantaneu aplicaţie de proces	**1.0.0**
Acronim instantaneu pentru aplicaţia de proces	**1.0.0**

Pentru o resursă, precum un modul sau o bibliotecă, versiunea este parte componentă din identitatea sa.

Tabelul de mai jos prezintă un exemplu de două module şi modul în care fişierele EAR asociate includ contextul de versiune:

Tabela 41. Module SCA şi fişiere EAR dependente de versiune

Nume modul SCA	Nume dependent de versiune	EAR dependent de versiune/nume aplicaţie
M1	PA1-1.0.0-M1	PA1-1.0.0-M1.ear
M2	PA1-1.0.0-M2	PA1-1.0.0-M2.ear

Tabelul de mai jos prezintă un exemplu de două biblioteci din domeniul aplicaţiei de proces şi modul în care fişierele JAR asociate includ contextul versiunii:

Tabela 42. Biblioteci din domeniul aplicaţiei de proces şi fişiere JAR dependente de versiune

Nume bibliotecă din domeniul aplicaţiei de proces SCA	Nume dependent de versiune	Nume JAR dependent de versiune
Lib1	PA1-1.0.0-Lib1	PA1-1.0.0-Lib1.jar
Lib2	PA1-1.0.0-Lib2	PA1-1.0.0-Lib2.jar

Legări dependente de versiune

Aplicaţiile de proces pot conţine module SCA care includ legări de import şi export. Atunci când co-implementaţi aplicaţii, legarea pentru fiecare versiune a aplicaţiei trebuie să fie unică. Unele legări sunt actualizate automat în timpul implementării pentru a se asigura unicitatea dintre versiuni. În alte cazuri, va trebui să actualizaţi legarea după implementare pentru a-i asigura unicitatea.

O legare *dependentă de versiune* este pusă în domeniul unei anumite versiuni ale aplicaţiei de proces, lucru care garantează unicitatea între aplicaţiile de proces. Următoarele secţiuni descriu legările care sunt actualizate automat pentru a fi dependente de versiune, precum şi orice acţiune pe care aveţi nevoie să o efectuaţi la momentul rulării atunci când o legare nu este dependentă de versiune. Pentru mai multe informaţii despre lucrurile care se iau în considerare atunci când creaţi module, vedeţi “Considerente atunci când se utilizează legări”.

SCA

Ţinta unei legări SCA este redenumită în mod automat pentru a fi dependentă de versiune în timpul implementării în cazul în care legările de import şi export ale modulului sunt definite în acelaşi domeniu al aplicaţiei de proces.

În cazul în care legările nu sunt definite în acelaşi domeniu al aplicaţiei de proces, se înregistrează un mesaj de informare. Trebuie să modificaţi legarea de import după implementare pentru ca adresa ţintă a punctului final să se modifice. Puteţi utiliza consola administrativă pentru a modifica adresa ţintă a punctului final.

Serviciu Web (JAX-WS sau JAX-RPC)

Adresa ţintă a punctului final a unui serviciu web este redenumită automat pentru a fi dependentă de versiune în timpul implementării în cazul în care următoarele considerente sunt adevărate:

- Aţi urmat convenţia implicită de numire pentru adresa:

  http://*ip*:*port*/*ModuleName*Web/sca/*ExportName*
- Punctul final al adresei este SOAP/HTTP.
- Legările de import şi export ale modulului sunt definite în acelaşi domeniu al aplicaţiei de proces.

Dacă aceste condiţii nu sunt adevărate, este înregistrat un mesaj de informare. Acţiunea pe care o întreprindeţi apoi depinde de modul în care vă implementaţi aplicaţia de proces:

- Dacă vă co-implementaţi aplicaţia de proces, trebuie să redenumiţi manual URL-ul punctului final pentru SOAP/HTTP sau coada de destinaţie SOAP/JMS, astfel încât să fie unic între versiunile aplicaţiei de proces. Puteţi utiliza consola administrativă după implementare pentru a modifica adresa ţintă a punctului final.
- Dacă implementaţi doar o singură versiune a aplicaţiei de proces, aveţi posibilitatea să ignoraţi acest mesaj

Pentru co-implementarea instantaneului legării serviciului Web SOAP/ JMS, acţiunea pe care o întreprindeţi depinde de modul în care vă implementaţi aplicaţia de proces:

- În cazul în care importul şi exportul ţintă sunt în aceeaşi aplicaţie de proces, realizaţi paşii următori înainte de a publica aplicaţia de proces pe Process Center şi de a crea instantaneul:
  1. Modificaţi URL-ul punctului final pentru export. Asiguraţi-vă că destinaţia şi fabrica de conexiuni sunt unice.
  2. Modificaţi URL-ul punctului final pentru import, astfel încât să fie acelaşi cu cel pe care l-aţi pecificat pentru export la pasul anterior.
- În cazul în care importul şi exportul ţintă sunt în aplicaţii de proces diferite, realizaţi paşii următori:
  1. Modificaţi URL-ul punctului final pentru export. Asiguraţi-vă că destinaţia şi fabrica de conexiuni sunt unice.
  2. Publicaţi aplicaţia de proces la Process Center.
  3. Creaţi instantaneul.
  4. Implementaţi aplicaţia de proces pe Process Server.
  5. Utilizaţi consola administrativă WebSphere pentru a modifica URL-ul punctului final pentru importul corespunzător, astfel încât să fie acelaşi cu cel specificat pentru export.

HTTP

Adresa URL a punctului final al unei legări HTTP este redenumită automat în timpul implementării ca să ţină cont de versiune, în cazul în care toate condiţiile următoare sunt adevărate:

- Aţi urmat convenţia implicită de numire pentru adresa:
  http(s)://*ip*:*port*/*ModuleName*Web/*contextPathinExport*
- Legările de import şi export ale modulului sunt definite în acelaşi domeniu al aplicaţiei de proces.

Dacă aceste condiţii nu sunt adevărate, este înregistrat un mesaj de informare. Acţiunea pe care o întreprindeţi apoi depinde de modul în care vă implementaţi aplicaţia de proces:

- Dacă co-implementaţi aplicaţia de proces, trebuie să redenumiţi manual URL-ul punctului final, astfel încât să fie unic între versiunile aplicaţiei de proces. Puteţi utiliza consola administrativă după implementare pentru a modifica adresa ţintă a punctului final.
- Dacă implementaţi doar o singură versiune a aplicaţiei de proces, aveţi posibilitatea să ignoraţi acest mesaj

JMS şi JMS generic

Legările JMS generate de sistem şi legările generice JMS sunt conştiente de versiune în mod automat.

Notă: Pentru legările JMS definite de utilizator şi legările JMS generice, nu are loc redenumirea automată în timpul implementării pentru ca legările să devină conştiente de versiune. În cazul în care legarea este definită de utilizator, trebuie să redenumiţi următoarele atribute, astfel încât să fie unice între versiunile aplicaţiilor de proces:

- Configurarea punctului final
- Coadă destinaţie de recepţie
- Nume port ascultător (dacă este definit)

Setaţi destinaţia Trimitere corespunzătoare în cazul în care modificaţi punctul final al modulului ţintă.

MQ/JMS şi MQ

Nu apare redenumirea automată în timpul implementării pentru a permite legăturilor de tip MQ/JMS sau MQ să fie dependente de versiune.

Trebuie să redenumiţi atributele următoare, astfel încât să fie unice între versiunile aplicaţiilor de proces:

- Configurarea punctului final
- Coadă destinaţie de recepţie

Setaţi destinaţia Trimitere corespunzătoare în cazul în care modificaţi punctul final al modulului ţintă.

EJB

Nu apare redenumirea automată în timpul implementării pentru a permite legăturilor de tip EJB să fie dependente de versiune.

Trebuie să redenumiţi atributul nume JNDI, astfel încât acestea să fie unice între versiunile aplicaţiilor de proces.

Reţineţi că aplicaţiile client trebuiesc de asemenea actualizate pentru a utiliza noile nume JNDI.

EIS

Un adaptor de resurse este redenumit în timpul implementării în mod automat pentru a fi dependent de versiune, atâta timp cât numele implicit al resursei (***ModuleName*App:Descriere Adaptor**) nu a fost modificat.

În cazul în care numele implicit al resursei a fost modificat, numele adaptorului resursei trebuie să fie unic între versiunile aplicaţiei de proces.

Dacă numele adaptorului de resurse nu sunt unice, este înregistrat în timpul implementării un mesaj de informare pentru a vă atenţiona. Aveţi posibilitatea să redenumiţi manual adaptoarele de resurse după implementare folosind consola administrativă.

Invocarea dinamică dependentă de versiune

Aveţi posibilitatea să configuraţi componentele fluxului de mediere, astfel încât să ruteze mesajele către punctele finale care sunt stabilite în mod dinamic în momentul rulării. Atunci când creaţi modulul de mediere, configuraţi căutarea punctelor finale, astfel încât să folosească rutarea dependentă de versiune.

În cazul în care folosiţi stilul IBM_VRM (<*versiune*>.<*ediţie*>.<*modificare*) pentru instantaneu, puteţi exporta fişierul EAR al aplicaţiei de proces în WSRR (WebSphere Service Registry and Repository). Atunci când creaţi modulul de mediere, configuraţi atunci şi căutarea punctelor finale, astfel încât să folosească rutarea dependentă de versiune. De exemplu, selectaţi **Returnare punct final care se potriveşte cu cea mai recentă versiune compatibilă pentru serviciile bazate pe modului SCA** în câmpul **Politică potrivire**, apoi selectaţi **SCA** pentru **Tip legare**.

Versiunile viitoare ale aplicaţiei de proces sunt implementate pe server şi publicate la WSRR, iar căutarea punctului final al modulului de mediere invocă în mod dinamic cea mai recentă versiune compatibilă a punctului final al serviciului.

Reţineţi ca o alternativă, că puteţi seta ţinta în SMOHeader, iar valoarea poate fi purtată de către mesajul de cerere.

Implementarea aplicaţiilor de proces cu module şi proiecte Java

Aplicaţiile de proces pot conţine module Java EE şi proiecte Java personalizate. Atunci când co-implementaţi aplicaţii, modulul Java EE personalizat pentru fiecare versiune a aplicaţiei trebuie să fie unic.

Reţineţi că modulele Java EE şi proiectele Java personalizate sunt implementate pe un server în cazul în care acestea sunt implementate cu un modul SCA care are o dependenţă declarată în acestea. Dacă nu selectaţi **Implementare cu modul** (care este implicit) atunci când declaraţi dependenţa, trebuie să implementaţi modulul sau proiectul în mod manual.

Implementarea aplicaţiilor de proces cu ajutorul regulilor operaţionale şi a selectoarelor

În cazul în care implementaţi mai multe versiuni ale unei aplicaţii de proces care include o regulă operaţională sau o componentă de tip selector, fiţi atenţi la modul în care versiunile folosesc metadatele asociate.

Metadatele dinamice pentru o regulă operaţională sau pentru o componentă de tip selector sunt definite în momentul rulării prin numele componentei, numele spaţiului pentru componenta ţintă şi tipul componentei. În cazul în care două sau mai multe versiuni ale aplicaţiei de proces care conţin o regulă operaţională sau un selector sunt implementate în acelaşi mediu de runtime, acestea vor partaja aceeaşi logică pentru reguli (regulă operaţională) sau aceleaşi metadate (selector) pentru rutare.

Pentru a permite fiecărei versiuni a regulii operaţionale sau a componentei de tip selector ce aparţine de aplicaţia de proces să îşi utilizeze propriile metadate dinamice (logică pentru reguli sau rutare), modificaţi codul (refactorizaţi) pentru spaţiul de nume ţintă, astfel încât acesta să fie unic pentru fiecare versiune a aplicaţiei de proces.

Arhitectura de implementare

Arhitectura de implementare IBM Business Process Manager conţine procese software numite servere, unităţi topologice referite ca noduri şi celule şi magazia de configurare utilizată pentru memorarea informaţiilor de configurare.

Celulele

În IBM Business Process Manager, *celulele* sunt grupări logice de unul sau mai multe noduri dintr-o reţea distribuită.

O celulă este un concept de configurare, o cale pentru administratori de a asocia logic nodurile unul cu celălalt. Administratorii definesc nodurile care alcătuiesc o celulă în funcţie de criteriile specifice care au sens în mediile lor organizaţionale.

Datele de configurare administrative sunt memorate în fişiere XML. O celulă reţine fişiere de configurare master pentru fiecare server din fiecare nod din celulă. Fiecare nod şi server are, de asemenea, propriile fişiere locale de configurare. Modificările fişierului local de configurare a unui nod sau a unui server sunt temporare dacă serverul aparţine celulei. Când sunt efective, modificările locale înlocuiesc configuraţiile celulei. Modificările fişierelor de configurare ale serverului master şi ale nodului master făcute la nivelul celulei înlocuiesc orice modificare temporară făcută asupra nodului când documentele configuraţiei celulei sunt sincronizate cu nodurile. Sincronizarea apare la evenimente desemnate, cum ar fi pornirea unui server.

Serverele

Serverele furnizează funcţionalitatea de bază a IBM Business Process Manager. Servere de proces extind sau măresc abilitatea unui server de aplicaţii de a manipula module SCA (Service Component Architecture). Alte servere (manageri de implementare şi agenţi de nod) sunt utilizate pentru gestionarea serverelor Process.

Un Process Server poate fi un *server autonom* sau un *server gestionat*. Un server gestionat poate fi eventual membru unui *cluster*. O colecţie de servere gestionate, cluster-e de servere şi alte middleware-uri se numeşte un *mediu de implementare*. Într-un mediu de implementare, fiecare server sau cluster gestionat este configurat pentru o anumită funcţie în mediul de implementare (de exemplu, gazdă destinaţie, gazdă modul de aplicaţie sau server Common Event Infrastructure). Un server autonom este configurat să furnizeze toate funcţiile cerute.

Serverele asigură mediul runtime pentru modulele SCA, pentru resursele care sunt utilizate de către acele module (surse de date, specificări de activare şi destinaţii JMS) şi pentru resurse livrate de IBM (destinaţii de mesaje, containere Business Process Choreographer şi servere Common Event Infrastructure).

Un *agent de nod* este un agent administrativ ce reprezintă un nod din sistemul dvs. şi gestionează serverele acelui nod. Agenţii de nod monitorizează serverele de pe un sistem de gazdă şi rutează cererile administrative către servere. Agentul de nod este creat când nodul este federalizat unui manager de implementare.

Un *manager de implementare* este un agent administrativ care furnizează o vizualizare centralizată de gestionare pentru mai multe servere şi cluster-e.

Un server autonom este definit de un profil autonom; un manager de implementare este definit de un profil corespunzător; serverele gestionate sunt create într-un *nod gestionat*, ce este definit de un profil personalizat.

Serverele autonome

Un server autonom asigură un mediu pentru implementarea modulelor SCA într-un proces server. Acest proces server include, dar nu este limitat la o consolă administrativă, o ţintă de implementare, suportul de mesaje, managerul de reguli de proces operaţional şi un server Infrastructură eveniment comun.

Un server autonom este simplu de setat şi are o consolă Primii paşi din care puteţi porni şi opri serverul şi puteţi deschide galeria de eşantioane şi consola administrativă. Dacă instalaţi eşantioanele IBM Business Process Manager şi apoi deschideţi galeria de eşantioane, o soluţie exemplu este implementată în serverul autonom. Puteţi explora resursele utilizate pentru acest exemplu din consola administrativă.

Puteţi implementa propriile dumneavoastră soluţii într-un server autonom, dar un server autonom nu poate furniza capacitatea, scalabilitatea sau robusteţea care este cerută de un mediu de producţie. Pentru mediul dumneavoastră de producţie, este mai bine să utilizaţi un mediu Network Deployment.

Este posibil să porniţi cu un server autonom şi mai târziu să îl includeţi într-un mediu Network Deployment, federalizându-l la o celulă a managerului de implementare, *asigurat că nici un alt nod nu a fost federalizat la acea celulă*. Nu este posibil să federalizaţi mai multe servere autonome într-o celulă. Pentru a federaliza serverul autonom, utilizaţi consola administrativă a managerului de implementare sau comanda**addNode**. Serverul autonom nu trebuie să ruleze când îl federalizaţi utilizând comanda **addNode**.

Un server autonom este definit de către un profil server autonom.

Cluster-ele

Cluster-ele sunt grupuri de servere care sunt gestionate împreună şi care participă la gestiunea încărcării de lucru.

Un cluster poate conţine noduri sau servere individuale de aplicaţii. Un nod este de obicei un calculator fizic cu o adresă IP a gazdei distinctă care rulează unul sau mai multe servere de aplicaţii. Cluster-ele pot fi grupate sub configuraţia unei celule care asociază logic multe servere şi cluster-e cu configuraţii şi aplicaţii diferite unul cu celălalt în funcţie de discreţia administratorului şi de ceea ce are sens în mediile lor organizaţionale.

Cluster-ele sunt responsabile pentru echilibrarea încărcării de lucru în servere. Serverele care sunt parte componentă a unui cluster sunt numite membri ai cluster-ului. La instalarea unei aplicaţii sau a unui cluster, aplicaţia este instalată automat în fiecare membru al cluster-ului.

Deoarece fiecare membru al cluster-ului conţine aceleaşi aplicaţii, puteţi distribui taskuri client în funcţie de capacităţile diferitelor maşini prin asignarea de ponderi fiecărui server.

Asignarea de ponderi serverelor dintr-un cluster îmbunătăţeşte performanţa şi preluarea la defect. Taskurile sunt asignate serverelor care au capacitatea de a realiza operaţiile taskului. Dacă un server nu este disponibil pentru a realiza taskul, acesta este asignat altui membru al cluster-ului. Această capabilitate de reasignare are avantaje evidente la rularea unui singur server de aplicaţii care poate deveni supraîncărcat dacă sunt făcute prea multe cereri.

Profilurile

Un profil defineşte un mediu runtime unic, cu fişiere de comandă, fişiere de configurare şi fişiere istoric separate. Profilurile definesc trei tipuri diferite de medii în sistemele IBM Business Process Manager: server autonom, manager de implementare şi nod gestionat.

Utilizând profiluri, puteţi avea mai multe medii runtime într-un sistem, fără a trebui să instalaţi copii multiple ale fişierelor binare IBM Business Process Manager.

Utilizaţi Profile Management Tool sau utilitarul **manageprofiles** al liniei de comandă pentru a crea profiluri.

Notă: În platformele distribuite, fiecare profil are un nume unic. În platforma z/OS, toate profilurile sunt numite "implicit".

Directorul profilului

Fiecare profil din sistem are propriul său director care conţine toate fişierele sale. Specificaţi locaţia directorului de profil la crearea profilului. În mod implicit, este în directorul de profiluri din directorul în care este instalat IBM Business Process Manager. De exemplu, profilul Dmgr01 este în C:\Program Files\IBM\WebSphere\ProcServer\profiles\Dmgr01.

Consola Primii paşi

Fiecare profil din sistem are o consolă Primii paşi. Puteţi utiliza această interfaţă pentru a vă familiariza cu serverul autonom, cu managerul de implementare sau cu nodul gestionat.

Profilul implicit

Primul profil pe care îl puteţi crea într-o instalare a IBM Business Process Manager este *profilul implicit*. Profilul implicit este destinaţia implicită pentru comenzi lansate din directorul bin din directorul în care a fost instalat IBM Business Process Manager. Dacă există doar un profil într-un sistem, fiecare comandă operează în acel profil. Dacă creaţi un alt profil, puteţi să îl faceţi implicit.

Notă: Profilul implicit nu este neapărat un profil al cărui nume este "implicit".

Augmentare profiluri

Dacă aveţi deja un profil manager de implementare, un profil personalizat sau un profil server autonom creat pentru WebSphere Application Server Network Deployment sau WebSphere ESB, îl puteţi *completa* să suporte IBM Business Process Manager în plus faţă de funcţiile existente. Pentru a completa un profil, instalaţi mai întâi IBM Business Process Manager. Apoi utilizaţi Profile Management Tool sau utilitarul **manageprofiles** al liniei de comandă.

Restricţie: Nu puteţi completa un profil dacă acesta defineşte un nod gestionat care este deja federalizat la un manager de implementare.

Managerii de implementare

Un manager de implementare este un server care gestionează operaţiile pentru un grup sau celulă logică a altor servere. Managerul de implementare este locaţia centrală pentru administrarea serverelor şi a cluster-elor.

La crearea unui mediu de implementare, profilul managerului de implementare este primul profil pe care îl creaţi. Managerul de implementare are o consolă Primii paşi, de la care puteţi porni şi opri managerul de implementare şi porniţi consola administrativă. Folosiţi consola administrativă a managerului de implementare pentru a gestiona serverele şi cluster-ele din celulă. Aceasta include configurarea serverelor şi a cluster-elor, adăugarea serverelor la cluster-e, pornirea şi oprirea serverelor şi a cluster-elor şi implementarea modulelor SCA.

Deşi managerul de implementare este un tip de server, nu puteţi implementa module pe însuşi managerul de implementare.

Nodurile

Un *nod* este o grupare logică de servere gestionate.

De obicei, un nod corespunde unui sistem informatic logic sau fizic cu o adresă IP gazdă distinctă. Nodurile nu se pot extinde pe mai multe calculatoare. De obicei, numele nodului sunt identice cu numele gazdă pentru calculator.

Nodurile din topologia Network Deployment pot fi gestionate sau negestionate. Un nod gestionat are un proces al agentului de nod care gestionează configuraţia sa şi serverele. Nodurile negestionate nu au un agent de nod.

Nodurile gestionate

Un *nod gestionat* este un nod care este federalizat către un manager de implementare şi care conţine agent nod şi poate conţine servere gestionate. Într-un nod gestionat, puteţi configura şi rula serverele gestionate.

Serverele gestionate care sunt configurate pe un nod formează resursele mediului dumneavoastră de implementare. Aceste servere sunt create, configurate, pornite, oprite, gestionate şi şterse folosind consola administrativă a managerului de implementare.

Un nod gestionat are un agent nod care gestionează toate serverele de pe un nod.

Când un nod este federalizat, este creat automat un proces agent nod. Acest agent nod trebuie să ruleze pentru a putea gestiona configuraţia profilului. De exemplu, când realizaţi următoarele taskuri:

- Porniţi şi opriţi procesele serverului.
- Sincronizaţi datele de configurare de pe managerul de implementare cu copia de pe nod.

Totuşi, agentul nod nu trebuie să ruleze pentru ca aplicaţiile să poată rula sau configura resursele din nod.

Un nod gestionat poate conţine unul sau mai multe servere, care sunt gestionate de un manager de implementare. Puteţi implementa soluţii pentru servere într-un nod gestionat, dar nodul gestionat nu conţine o galerie de eşantioane de aplicaţii. Nodul gestionat este definit de un profil personalizat şi are o consolă Primii paşi.

Nodurile negestionate

Un nod negestionat nu are un agent de nod care să îi gestioneze serverele.

Nodurile negestionate din topologia Network Deployment pot avea definiţii de servere cum ar fi servere web, dar nu definiţii de servere de aplicaţii. Nodurile negestionate nu pot fi niciodată federalizate. Prin urmare, un agent de nod nu poate fi adăugat niciodată la un nod negestionat. Alt tip de nod negestionat este un server autonom. Managerul de implementare nu poate gestiona acest server autonom deoarece nu este cunoscut de celulă. Un server autonom poate fi federalizat. Când este federalizat, un agent de nod este creat în mod automat. Nodul devine un nod gestionat în celulă.

Agenţii de nod

Agenţii de nod sunt agenţi administrativi care rutează cereri administrative la servere.

Un agent de nod este un server care rulează pe fiecare calculator gazdă care participă la configuraţia Network Deployment. Este pur şi simplu un agent administrativ şi nu este implicat în funcţii care deservesc aplicaţii. Un agent de nod găzduieşte, de asemenea, alte funcţii administrative importante precum servicii de transfer de fişiere, sincronizare de configurare şi monitor de performanţă.

Considerente privind numirea pentru profiluri, noduri, server, gazde şi celule

Acest subiect discută termeni rezervaţi şi probleme pe care trebuie să le luaţi în considerare la numirea profilului, nodului, serverului, gazdei şi celulei dumneavoastră (dacă se poate aplica). Acest subiect se aplică platformelor distribuite.

Considerente privind numirea profilului

Numele profilului poate fi orice nume unic cu următoarele restricţii. Nu folosiţi niciunul din următoarele caractere când vă denumiţi profilul:

- Spaţii
- Caractere speciale care nu sunt permise în cadrul unui director al sistemului dumneavoastră de operare, precum *, & sau ?.
- Semne / sau \

Sunt permise caractere pe doi octeţi.

Windows **Considerente privind calea directorului:** Calea către directorul instalării trebuie să fie mai mică decât sau egală cu 60 caractere. Numărul de caractere din directorul *cale_director_profiluri\nume_profil* trebuie să fie mai mic decât sau egal cu 80 caractere.

Notă: Folosiţi o convenţie de numire cale scurtă când creaţi un profil într-un mediu Windows pentru a evita limitarea Windows de lungime de cale de 255 caractere.

Considerente privind numele de nod, server, gazdă şi celulă

Nume rezervate: Evitaţi utilizarea numelor rezervate ca valori ale câmpului. Utilizarea numelor rezervate poate duce la rezultate neprevăzute. Sunt rezervate următoarele cuvinte:

- cells
- nodes

- servers
- clusters
- applications
- deployments

Descrieri de câmpuri pe paginile Nod şi nume de gazde şi nod, Gazdă şi Nume celule: Utilizaţi indicaţiile corespunzătoare pentru a denumi atunci când creaţi profile.

- Profilurile serverului autonom
- Profilurile managerului de implementare
- Profiluri personalizate

Tabela 43. Indicaţii de denumire pentru profiluri de server autonome

Nume câmp	Valoare implicită	Constrângeri	Descriere
Nume nod	Linux UNIX Windows *shortHostName* Node *NodeNumber* unde: • *shortHost Name* este numele scurt al gazdei. • *NodeNumber* este un număr secvenţial care porneşte de la 01.	Evitaţi folosirea numelor rezervate.	Selectaţi orice nume doriţi. Pentru a ajuta organizarea instalaţiei dumneavoastră, folosiţi un nume unic dacă aveţi de gând să creaţi mai mult de un server pe sistem.
Nume server	Linux UNIX Windows server1	Folosiţi un nume unic pentru server.	Numele logic pentru server.
Nume de gazdă	Linux UNIX Windows Forma lungă a numelui serverului de nume domeniu (DNS - domain name server).	Numele gazdă trebuie să fie adresabil prin reţeaua dumneavoastră. Dacă doriţi să folosiţi Business Space, utilizaţi un nume de gazdă complet calificat.	Folosiţi numele DNS actual sau adresa IP a staţiei dumneavoastră de lucru pentru a permite comunicaţia cu ea. Vedeţi informaţii suplimentare despre numele gazdă urmând această tabelă.

Tabela 44. Indicaţii de denumire pentru profiluri de manager de implementare

Nume câmp	Valoare implicită	Constrângeri	Descriere
Nume nod	Linux UNIX Windows *shortHostName* Cell Manager*Node Number* unde: • *shortHost Name* este numele scurt al gazdei. • *NodeNumber* este un număr secvenţial care porneşte de la 01.	Folosiţi un nume unic pentru managerul de implementare. Evitaţi folosirea numelor rezervate.	Numele este folosit pentru administrarea în cadrul celulei manager de implementare.
Nume de gazdă	Linux UNIX Windows Forma lungă a numelui serverului de nume domeniu (DNS - domain name server).	Numele gazdă trebuie să fie adresabil prin reţeaua dumneavoastră. Evitaţi folosirea numelor rezervate. Dacă doriţi să folosiţi Business Space, utilizaţi un nume de gazdă complet calificat.	Folosiţi numele DNS actual sau adresa IP a staţiei dumneavoastră de lucru pentru a permite comunicaţia cu ea. Vedeţi informaţii suplimentare despre numele gazdă urmând această tabelă.

Tabela 44. Indicaţii de denumire pentru profiluri de manager de implementare (continuare)

Nume câmp	Valoare implicită	Constrângeri	Descriere
Nume celulă	Linux UNIX Windows *shortHostName* Cell *CellNumber* unde: • *shortHost Name* este numele scurt al gazdei. • *CellNumber* este un număr secvenţial care porneşte de la 01.	Folosiţi un nume unic pentru celula manager de implementare. Un nume de celulă trebuie să fie unic în orice circumstanţe în care rulează produsul pe aceeaşi staţie de lucru fizică sau cluster de staţii de lucru, precum un Sysplex. În plus, un nume de celulă trebuie să fie unic în orice circumstanţe în care conectarea la reţea între entităţi este necesară fie între celule, fie de la un client care trebuie să comunice cu fiecare dintre celule. Numele de celulă trebuie de asemenea să fie unice dacă spaţiile lor de nume vor fi federalizate. Altfel, aţi putea întâlni excepţii precum javax.naming.Name NotFoundException, caz în care trebuie să creaţi celule unic numite.	Toate nodurile federalizate devin membri ai celulei manager de implementare, pe care aţi numit-o în pagina Nume de noduri, gazde şi celule a Profile Management Tool.

Tabela 45. Indicaţii de denumire pentru profiluri personalizate

Nume câmp	Valoare implicită	Constrângeri	Descriere
Nume nod	Linux UNIX Windows *shortHostName* Node *NodeNumber* unde: • *shortHost Name* este numele scurt al gazdei. • *NodeNumber* este un număr secvenţial care porneşte de la 01.	Evitaţi folosirea numelor rezervate. Folosiţi un nume unic în cadrul celulei manager de implementare.	Numele este folosit pentru administrarea în cadrul celulei manager de implementare la care este adăugat profilul personalizat. Folosiţi un nume unic în cadrul celulei manager de implementare.
Nume de gazdă	Linux UNIX Windows Forma lungă a numelui serverului de nume domeniu (DNS - domain name server).	Numele gazdă trebuie să fie adresabil prin reţeaua dumneavoastră. Dacă doriţi să folosiţi Business Space, utilizaţi un nume de gazdă complet calificat.	Folosiţi numele DNS actual sau adresa IP a staţiei dumneavoastră de lucru pentru a permite comunicaţia cu ea. Vedeţi informaţii suplimentare despre numele gazdă urmând această tabelă.

Considerente privind numele gazdă:

Numele gazdă este numele de reţea pentru staţia de lucru fizică pe care este instalat nodul. Numele gazdă trebuie să rezolve un nod reţea fizică pe server. Când există mai multe plăci de reţea în server, numele gazdă sau adresa IP trebuie să fie rezolvată la una din plăcile de reţea. Nodurile la distanţă utilizează numele gazdă pentru a se conecta şi a comunica cu acest nod.

IBM Business Process Manager este compatibil cu ambele versiuni ale Internet Protocol versiunea 4 (IPv4) şi versiunea 6 (IPv6). Oricând puteţi introduce adresele IP în consola administrativă sau în altă parte, puteţi face aceasta în orice

format. Reţineţi că dacă IPv6 este implementat pe sistemul dumneavoastră, trebuie să introduceţi adresa IP în formatul IPv6 şi, invers dacă IPv6 nu vă este încă disponibilă, introduceţi adresele IP în formatul IPv4. Pentru mai multe informaţii despre IPv6, referiţi-vă la descrierea următoare: IPv6.

Următoarele indicaţii vă pot ajuta la determinarea numelui de gazdă corespunzător pentru staţia dumneavoastră de lucru:

- Selectaţi un nume de gazdă la care pot ajunge alte staţii de lucru din reţeaua dumneavoastră.
- Nu folosiţi identificatorul generic, localhost, pentru această valoare.
- Nu încercaţi să instalaţi produse IBM Business Process Manager pe un server cu un nume de gazdă care foloseşte caractere din setul de caractere pe doi octeţi (DBCS - double-byte character set). Caracterele DBCS nu sunt suportate când sunt folosite în numele gazdă.
- Evitaţi folosirea caracterului liniuţă de subliniere (_) în numele de server. Standardele Internet obligă ca numele de domenii să se conformeze cerinţelor numelor gazdă descrise în Internet Official Protocol Standards RFC 952 şi RFC 1123. Numele de domenii trebuie să conţină doar litere (majuscule sau minuscule) şi cifre. Numele de domenii pot conţine de asemenea caractere liniuţă (-), atâta timp cât liniuţele nu se află la sfârşitul numelui. Caracterele liniuţă de subliniere (_) nu sunt suportate în numele gazdă. Dacă aţi instalat IBM Business Process Manager pe un server cu un caracter liniuţă de subliniere în numele său, accesaţi serverul prin adresa sa IP până îl redenumiţi.

Dacă definiţi noduri coexistente pe acelaşi calculator cu adrese IP unice, definiţi fiecare adresă IP într-o tabelă de căutare a unui server de nume domeniu (DNS - domain name server). Fişierele de configurare pentru servere nu furnizează rezolvarea numelui domeniului pentru adresele IP multiple de pe o staţie de lucru cu o singură adresă de reţea.

Valoarea pe care o specificaţi pentru numele gazdă este folosită ca valoare pentru proprietatea hostName din documentele de configurare. Specificaţi valoarea numelui gazdei într-unul din următoarele formate:

- Şirul numelui gazdă DNS complet calificat, precum xmachine.manhattan.ibm.com
- Şirul numelui gazdă DNS scurt implicit, precum xmachine
- Adrese ID numerice, precum 127.1.255.3

Numele gazdă DNS complet calificat prezintă avantajele de a fi neambiguu şi flexibil. Aveţi flexibilitatea de a schimba adresa IP actuală pentru sistemul gazdă, fără a schimba configuraţia serverului. Această valoare pentru numele gazdă este folositoare în special dacă doriţi să schimbaţi frecvent adresa IP când folosiţi Dynamic Host Configuration Protocol (DHCP) pentru a aloca adresele IP. Un dezavantaj al acestui format este acela de a fi dependent de DNS. Dacă DNS nu este disponibil, atunci conectivitatea este compromisă.

Numele scurt al gazdei este de asemenea rezolvabil în mod dinamic. Un format de nume scurt are abilitatea de a fi redefinit pe fişierul gazdelor locale, pentru ca sistemul să poată rula serverul chiar când este deconectat de la reţea. Definiţi numele scurt la 127.0.0.1 (loopback local) în fişierul gazdelor pentru a rula în mod deconectat. Un dezavantaj al formatului de nume scurt este acela de a fi dependent de DNS pentru accesul la distanţă. Dacă DNS nu este disponibil, atunci conectivitatea este compromisă.

O adresă IP numerică prezintă avantajul de a nu necesita rezolvarea numelui prin intermediul DNS. Un nod la distanţă se poate conecta la nodul pe care îl numiţi cu o adresă IP numerică, fără ca DNS să fie disponibil. Un dezavantaj al acestui format este acela că adresa IP numerică este fixată. Trebuie să modificaţi setările proprietăţii hostName în documentele de configurare de fiecare dată când modificaţi adresa IP a staţiei de lucru. Prin urmare, nu folosiţi o adresă IP numerică dacă utilizaţi DHCP sau dacă modificaţi regulat adresele IP. Un alt dezavantaj al acestui format este acela că nu puteţi folosi nodul dacă gazda este deconectată de la reţea.

BPMN 2.0

Definiţiile de proces operaţional IBM Business Process Manager suportă subclasa Common Executable a clasei de conformitate Modelare proces BPMN 2.0 care lucrează cu modele pe care le puteţi rula.

BPMN (Business Process Model and Notation) este standardul de bază pentru procese din IBM Process Designer şi IBM Process Center. Diagramele BPD (Business process definition) sunt bazate pe specificaţia BPMN. Acest subiect introduce unele dintre modalităţile în care BPMN 2.0 este aplicat în IBM Business Process Manager. Pentru informaţii detaliate despre BPMN, vedeţi pagina Specificaţii BPMN la http://www.bpmn.org/.

IBM Business Process Manager suportă următoarele tipuri de taskuri BPMN 2.0:

- None (task abstract în specificaţia BPMN 2.0)
- Task de sistem (task al serviciului în specificaţia BPMN 2.0)
- Task utilizator
- Script
- Task de decizie (task de reguli operaţionale în specificaţia BPMN 2.0)

Evenimentele mesaj intermediar IBM BPM oferă funcţii asemănătoare cu taskul de trimitere BPMN şi cu taskul de primire.

Notare BPMN 2.0

Începând cu V7.5.1, Process Designer pictogramele de taskuri BPMN 2.0 din diagramele BPD sunt colectate într-o paletă simplificată şi afişate în diagrame de proces. Pictogramele afişează dacă activitatea dumneavoastră este un task de sistem, un task utilizator, un task de decizie, script sau proces legat. Activităţile din modelele care au fost create în versiuni anterioare afişează, de asemenea, tipuri de taskuri BPMN 2.0 corespunzătoare şi pictograme de taskuri la vizualizarea lor în versiunea 7.5.1 sau mai târziu.

Activităţi şi taskuri

Există câteva modificări de terminologie de la versiunile anterioare a Process Designer. Un număr din aceste modificări implică tipuri de activităţi care au fost redenumite.

- Activităţile serviciu (automate) sunt acum taskuri sistem.
- Activităţile serviciu (task) într-un culoar non-sistem sunt acum taskuri utilizator.
- Activităţile serviciu (task) într-un culoar sistem sunt acum taskuri de decizie dacă facă referire la un serviciu de decizie.
- Activităţile serviciu (task) într-un culoar sistem sunt acum taskuri sistem dacă fac referire la orice tip de serviciu altul decât serviciu de decizie.
- Activităţile Javascript sunt acum taskuri script.
- Activităţile proces imbricat sunt acum procese legate.
- Activităţile externe de la versiunile anterioare ale Process Designer sunt disponibile ca implementări externe pentru taskurile utilizator sau pentru taskurile sistem.

Gateway-uri

Nu există modificări de notaţie în gateway-urile din versiunile anterioare. Totuşi, există trei modificări de terminologie. Gateway-ul de decizie este acum *gateway exclusiv*, gateway-ul simplu divizat sau unit este acum *gateway paralel*, iar gateway-ul condiţional separat sau unit este acum *gateway inclusiv*.

De asemenea, există un nou tip de gateway, *gateway de eveniment*. Un gateway de eveniment reprezintă un punct de ramificare într-un proces unde căile alternative care urmează gateway-ul se bazează pe evenimente care apar mai degrabă decât pe evaluarea expresiilor folosind datele de proces (precum cu un gateway exclusiv sau inclusiv). Un eveniment specific, de obicei, primirea unui mesaj, determină calea care va fi urmată.

Evenimente care nu întrerup

BPMN 2.0 a adăugat o notare pentru evenimente care nu întrerup. În mod implicit, un eveniment graniţă întrerupe activitatea de care este ataşată. Atunci când evenimentul este declanşat, activitatea se opreşte iar jetonul continuă

secvenţa iieşire a evenimentului. Dacă evenimentul este setat ca non-întrerupere atunci când evenimentul este declanşat activitatea ataşată continuă în paralel şi un nou jeton este generat şi este transmis în fluxul secvenţă de ieşire al evenimentului. Graniţa evenimentului se modifică într-o linie întreruptă pentru evenimente care nu întrerup.

Evenimentele intermediare care sunt ataşate activităţilor sunt evenimente intermediare de întrerupere dacă închid activităţile ataşate lor sau evenimente intermediare non-întrerupere dacă nu închid activităţile ataşate lor.

Eveniment de pornire

Specificaţia BPMN permite modelor de proces pentru să omită simbolurile de evenimente de pornire şi oprire. Process Designer necesită ca modelele de proces să folosească evenimente de pornire şi oprire.

Există diverse tipuri de evenimente de pornire disponibile în Process Designer:

procese

- fără
- mesaj
- ad hoc

subprocese

- fără

subprocese eveniment

- eroare
- mesaj
- cronometru

Puteţi modifica tipul unui eveniment de pornire prin editarea proprietăţilor evenimentului. Puteţi avea numeroase evenimente pornire mesaj într-un proces, dar puteţi folosi doar un eveniment fără pornire.

Oprire evenimente

Sunt disponibile patru tipuri de evenimente: *mesaj*, *terminare*, *eroare* şi *none*. Puteţi modifica tipul unui eveniment de oprire.

Când un proces părinte apelează un proces copil şi procesul copil o acţiune de terminare eveniment, procesul copil se opreşte şi procesul părinte continuă apoi cu următorii paşi.

Subprocese

Specificaţia BPMN defineşte două tipuri de subprocese, înglobate şi refolosibile. Puteţi crea ambele tipuri în Process Designer. Subprocesele înglobate sunt numite doar *subprocese* în Process Designer şi sunt noi în versiunea 7.5.1. Subprocesul reutilizabil BPMN este numit *proces legat* în Process Designer.

Un subproces există în cadrul procesului care îl conţine şi este o modalitate de grupare a paşilor procesului pentru a reduce complexitatea diagramei şi confuzia. Subprocesele restrâng mai mulţi paşi într-o singură activitate. Subprocesul poate fi văzut doar de către procesul în care este definit. Un subproces există în domeniul apelantului său şi are acces la toate variabilele din acel mediu. Nu există nici un parametru care să fie transmis în şi în afara subprocesului înglobat.

Separat de subproces şi de procesul legat, Process Designer are un subproces eveniment care este un subproces specializat care este utilizat pentru manipularea evenimentului. Nu este conectat la alte activităţi prin flux de secvenţe, şi apare doar dacă evenimentul de pornire este declanşat.

Procese legate

Un subproces BPMN reutilizabil este numit un *proces legat* în Process Designer. Acesta este un proces creat în afara procesului actual care poate fi apelat de către procesul actual. Acesta este reutilizabil deoarece alte definiţii de proces pot de asemenea apela acest proces. Procesul legat defineşte parametrii de intrare şi de ieşire şi nu are niciun acces la domeniul sau mediul apelantului. Procesul legat este similar cu procesul imbricat disponibil în versiunile anterioare; nu există nici o modificare în comportamentul activităţii. Procesele anterioare imbricate sunt migrate la procese legate. Procesul legat arată ca un subproces cu o limită compactă şi este evidenţiat în fereastra Inspector.

Bucle

BPMN oferă noţiunea de activitate care poate fi repetată. Activitatea poate fi atomică, însemnând că activitatea se repetă, sau poate fi un subproces, încapsulând o serie de paşi care se repetă. Dacă expandaţi activitatea repetată, vedeţi activităţile conţinute care urmează să fie rulate în mod repetat. Condiţia este evaluată întotdeauna la începutul fiecărei iteraţii buclă. Nu există nici o abilitate de a evalua la sfârşitul fiecărei iteraţii buclă.

IBM Business Process Manager are o *buclă cu mai multe instanţe* care este rulată de un număr finit de ori cu activităţile conţinute în cadrul rulării secvenţiale sau în paralel.

Import procese non-BPMN

Puteţi importa modele care au fost create în IBM WebSphere Business Modeler şi le puteţi utiliza în Process Designer. Pentru informaţii despre importul BPMN 2.0, vedeţi elementele Mapare IBM WebSphere Business Modeler în construcţiile IBM Business Process Manager. De asemenea, puteţi importa modele BPMN 2.0 care au fost create în IBM WebSphere Business Compass, Rational Software Architect sau alte medii de modelare.

Definiţii de proces operaţional (BPD-uri)

Pentru a modela un proces în IBM Process Designer, trebuie să creaţi o definiţie a procesului operaţional (BPD). Business Process Definition poate fi bazată pe un model BPMN importat.

Un BPD este un model reutilizabil al unui proces, definind ceea ce este comun tuturor instanţelor runtime ale acelui model de proces. Un BPD trebuie să conţină un eveniment de pornire, un eveniment sfârşit, cel puţin un strat, şi una sau mai multe activităţi. Consultaţi "Convenţiile de numire IBM Process Designer" în legăturile înrudite pentru detalii despre limitarea caracterului care se aplică BPD-urilor.

Business Process Definition (BPD) trebuie să includă un strat pentru fiecare sistem sau grup de utilizatori care participă într-un proces. Un strat poate fi un strat participant sau un strat sistem. Totuşi puteţi crea un BPD care grupează activităţile unui grup şi a unui sistem într-un singur strat dacă acea este preferinţa dumneavoastră. Consultaţi "Crearea unei definiţii a procesului operaţional (BPD)" în legăturile înrudite pentru informaţii despre cum se creează un BPD.

Puteţi desemna orice persoană sau grup specific pentru a fi responsabil pentru activităţi într-un strat participant. Fiecare strat creat este alocat grupului de participanţi Toţi utilizatorii în mod implicit. Puteţi folosi acest grup implicit de participanţi pentru rularea şi testarea BPD în Inspector. Grupul Toţi utilizatorii include toţi utilizatorii care sunt membrii ai grupului de securitate tw_allusers, care este un grup de securitate special care include în mod automat toţi utilizatorii din sistem.

Un strat sistem conţine activităţi manipulate de un sistem IBM Process Center specific. Fiecare activitate necesită o implementare, care defineşte activitatea şi seturile de proprietăţi pentru task. În timpul implementării, un dezvoltator creează un serviciu sau scrie JavaScript necesar pentru a finaliza activităţile din stratul sistem. Consultaţi "Înţelegerea tipurilor de serviciu" în legăturile înrudite pentru informaţii despre servicii.

Pentru fiecare BPD creat, trebuie să declaraţi variabile pentru a captura datele de afacere care sunt transmise de la o activitate la altă activitate în procesul dumneavoastră. Consultaţi "Gestionarea şi maparea variabilelor" în legăturile înrudite pentru a învăţa despre implementarea variabilelor.

Puteţi de asemenea să adăugaţi evenimente unui BPD. Evenimentele IBM BPM pot fi determinate de trecerea unei date scadente, de sosirea unei excepţii sau a unui mesaj. Declanşatorul dorit determină tipul de eveniment pe care îl alegeţi pentru a implementa. Pentru informaţii detaliate despre tipurile de evenimente disponibile şi despre declanşatorii lor, consultaţi "Modelarea evenimentelor".

Legările

La nucleul unei arhitecturi orientată-pe-servicii este conceptul unui *serviciu*, o unitate de funcţionalitate realizată de o interacţiune între dispozitive de calcul. Un *export* defineşte interfaţa externă (sau punctul de acces) a unui modul, astfel încât componentele SCA (Service Component Architecture) din modul să-şi poată oferi serviciile clienţilor externi. Un *import* defineşte o interfaţă către serviciile din afara modulului, astfel încât serviciile să poată fi apelate din modul. Utilizaţi *legări* specifice-protocolului cu exporturi şi importuri pentru a specifica mijloacele de a transporta datele în şi din modul.

Exporturile

Clienţii externi pot invoca componente SCA într-un modul de integrare peste o varietate de protocoale (cum ar ca HTTP, JMS, MQ şi RMI/IIOP) cu date într-o varietate de formate (cum ar fi XML, CSV, COBOL şi JavaBeans). Exporturile sunt componente ce primesc aceste cereri de la surse externe şi apoi invocă componente IBM Business Process Manager folosind modelul de programare SCA.

De exemplu, în următoarea figură, un export primeşte o cerere prin protocolul HTTP de la o aplicaţie client. Datele sunt transformate într-un obiect business, formatul folosit de componenta SCA. Componenta este apoi invocată cu acel obiect de date.

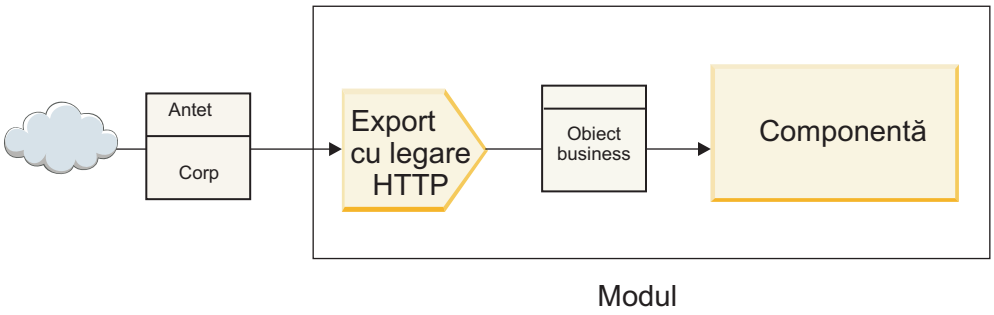


Figura 46. Un export cu legare HTTP

Importuri

O componentă SCA ar putea dori să invoce un serviciu extern non-SCA ce aşteaptă date într-un format diferit. Un import este utilizat de o componentă SCA pentru a invoca serviciul extern folosind modelul de programare SCA. Importul invocă apoi serviciul ţintă în modul aşteptat de serviciu.

De exemplu, în următoarea figură, o cerere de la o componentă SCA este trimisă, de către import, unui serviciu extern. Obiectul business, care este formatul folosit de componenta SCA, este transformat în formatul aşteptat de serviciu şi serviciul este invocat.

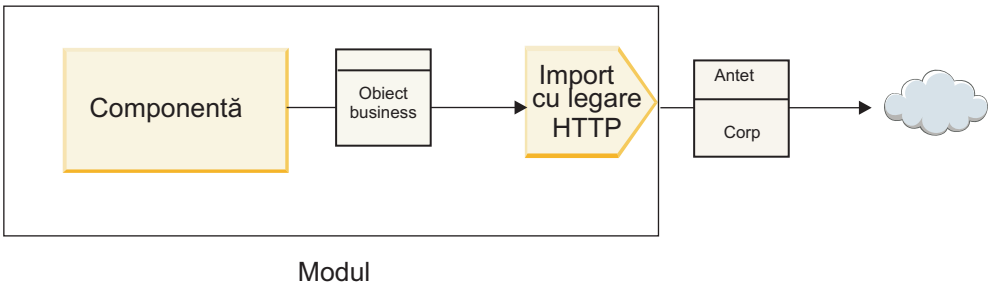


Figura 47. Un import cu legare HTTP

Lista de legări

Utilizaţi Integration Designer pentru a genera o legare pentru un import sau un export şi pentru a configura legarea. Tipurile de legări ce sunt disponibile sunt descrise în următoarea listă.

- SCA

  Legarea SCA, care este cea implicită, dă voie serviciului dumneavoastră să comunice cu servicii din alte module SCA. Utilizaţi un import cu legare SCA pentru a accesa un serviciu dintr-un alt modul SCA. Utilizaţi un export cu legare SCA pentru a oferi un serviciu altor module SCA.

- Serviciu Web

  O legare de serviciu web vă permite să accesaţi un serviciu extern folosind mesajele SOAP interoperabile şi calităţile de serviciu. Puteţi folosi, de asemenea, legăturile serviciului web pentru a include ataşamente ca parte componentă într-un mesaj SOAP.

  Legarea serviciului web poate folosi ca protocol de transport fie SOAP/HTTP (SOAP peste HTTP), fie SOAP/JMS (SOAP peste JMS). Indiferent de transport (HTTP sau JMS) utilizat pentru a pentru a transmite mesajele SOAP, legările serviciului manipulează întotdeauna interacţiunile cerere/răspuns în mod sincron.

- HTTP

  Legarea HTTP vă lasă să accesaţi un serviciu extern folosind protocolul HTTP, unde sunt folosite mesaje non-SOAP sau unde este necesar acces HTTP direct. Legarea este utilizată atunci când lucraţi cu servicii web care se bazează pe modelul HTTP (adică, servicii care folosesc operaţii bine cunoscute din interfaţa HTTP precum GET, PUT, DELETE, şi aşa mai departe).

- Enterprise JavaBeans (EJB)

  Legările EJB permit componentelor SCA să interacţioneze cu servicii furnizate de logica operaţională a Java EE care rulează pe un server Java EE.

- EIS

  Legarea EIS (enterprise information system), când este folosită cu un adaptor de resurse JCA, vă lasă să accesaţi servicii de pe un sistem de informaţii de întreprindere sau să vă faceţi serviciile disponibile EIS-ului.

- Legări JMS

  Legările Java Message Service (JMS), JMS generic şi WebSphere MQ JMS (MQ JMS) sunt folosite pentru interacţiuni cu sisteme de mesagerie, unde comunicarea asincronă prin cozi de mesaje este critică pentru fiabilitate.

  Un export cu una din legările JMS urmăreşte coada pentru sosirea unui mesaj şi trimite asincron răspunsul, dacă există, cozii de răspunsuri. Un import cu una din legările JMS construieşte şi trimite un mesaj unei cozi JMS şi urmăreşte o coadă pentru sosirea răspunsului, dacă există.

  – JMS

    Legarea JMS vă lasă să accesaţi furnizorul JMS încorporat în WebSphere.

  – Generic JMS

    Legarea JMS generic vă lasă să accesaţi un sistem de mesagerie al unui vendor non-IBM.

  – MQ JMS

    Legarea MQ JMS vă lasă să accesaţi subsetul JMS al unui sistem de mesagerie WebSphere MQ. Veţi folosi această legare când subsetul JMS de funcţii este suficient pentru aplicaţia dumneavoastră.

- MQ

  Legarea WebSphere MQ vă lasă să comunicaţi cu aplicaţii native MQ, aducându-le în cadrul de lucru al arhitecturii orientate spre servicii şi furnizând acces la informaţii de antet specifice-MQ. Veţi folosi această legare când aveţi nevoie să folosiţi funcţii native MQ.

Privire generală asupra legării de export şi import

Un export vă permite să faceţi serviciile dintr-un modul de integrare disponibile clienţilor externi, iar un import permite componentelor dumneavoastră SCA dintr-un modul de integrare să apeleze servicii externe. Legarea asociată cu exportul sau importul specifică relaţia dintre mesajele de protocol şi obiectele business. De asemenea specifică modul în care sunt selectate operaţiile şi defectele.

Fluxul de informaţii printr-un export

Un export primeşte o cerere, care este intenţionată pentru componenta la care este cablat exportul, peste un anumit transport determinat de legarea asociată (de exemplu, HTTP).

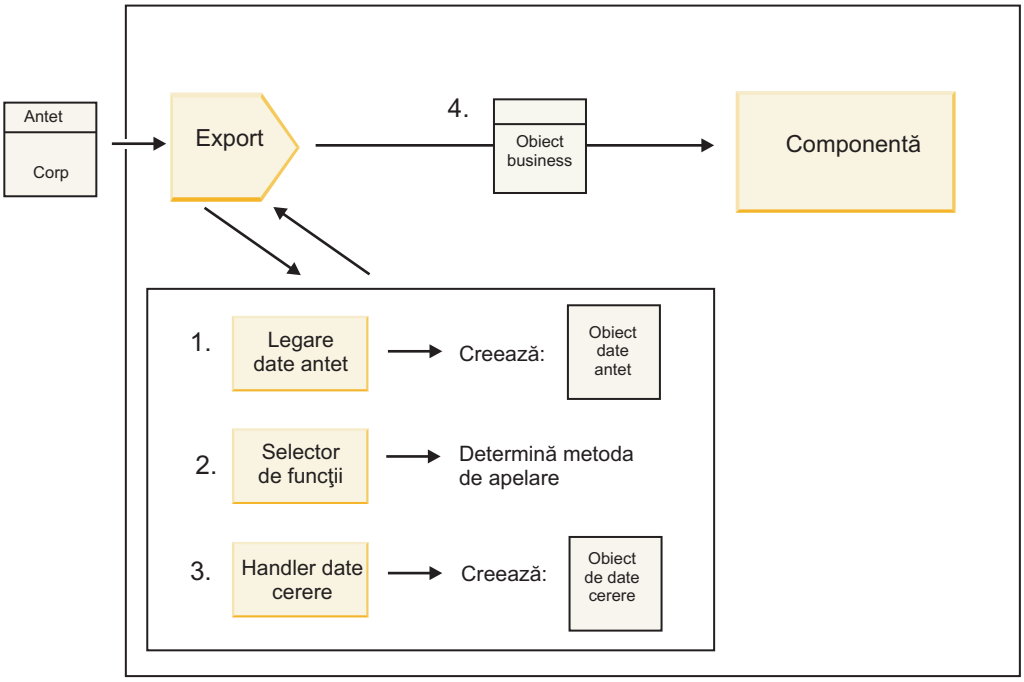


Figura 48. Fluxul unei cereri prin export către o componentă

Când exportul primeşte cererea, are loc următoarea secvenţă de evenimente:

1. Doar pentru legări WebSphere MQ, legarea datelor de antet transformă antetul protocolului într-un obiect de tip date de antet.
2. Selectorul de funcţii determină numele metodei native din mesajul de protocol. Numele metodei native este mapat de către configuraţia exportului de numele unei operaţii de pe interfaţa exportului.
3. Handler-ul de date sau legarea de date de cerere de pe metodă transformă cererea într-un obiect business cerere.
4. Exportul invocă metoda componentei cu obiectul business de cerere.
   - Legarea de export HTTP, legarea de export a serviciului web şi legarea de export EJB invocă componenta SCA în mod sincron.
   - Legările de export JMS, Generic JMS, MQ JMS şi WebSphere MQ invocă componenta SCA asincron.

Notaţi că un export poate propaga anteturile şi proprietăţile de utilizator pe care le primeşte prin protocol, dacă propagarea contextului este activată. Componentele cablate de export pot accesa apoi aceste anteturi şi proprietăţi de utilizator. Vedeţi subiectul “Propagare” din centrul de informare WebSphere Integration Developer pentru informaţii suplimentare.

Dacă aceasta este o operaţie pe două direcţii, componenta returnează un răspuns.

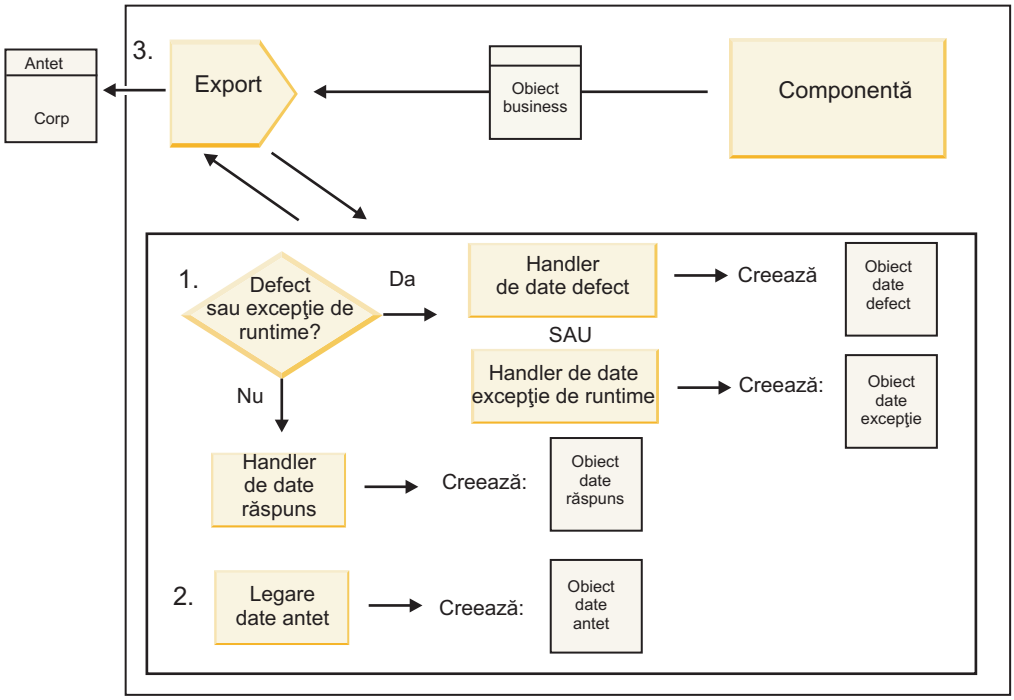


Figura 49. Fluxul unui răspuns înapoi prin export

Are loc următoarea secvenţă de paşi:

1. Dacă este primit un mesaj normal de răspuns de către legarea de export, handler-ul de date sau legarea de date de răspuns de pe metodă transformă obiectul business într-un răspuns.

   Dacă răspunsul este un defect, handler-ul de date sau legarea de date de defecte de pe metodă transformă defectul într-un răspuns defect.

   Doar pentru legări de export HTTP,dacă răspunsul este o excepţie din timpul rulării, handler-ul de date pentru excepţii din timpul rulării, dacă este configurat, este apelat.
2. Doar pentru legări WebSphere MQ, legarea de date de antet transformă obiectele de date de antet în anteturi de protocol.
3. Exportul trimite răspunsul serviciului prin transport.

Fluxul informaţiilor printr-un import

Componentele trimit cereri serviciilor din afara modulului folosind un import. Cererea este trimisă, printr-un anumit transport determinat de legarea asociată.

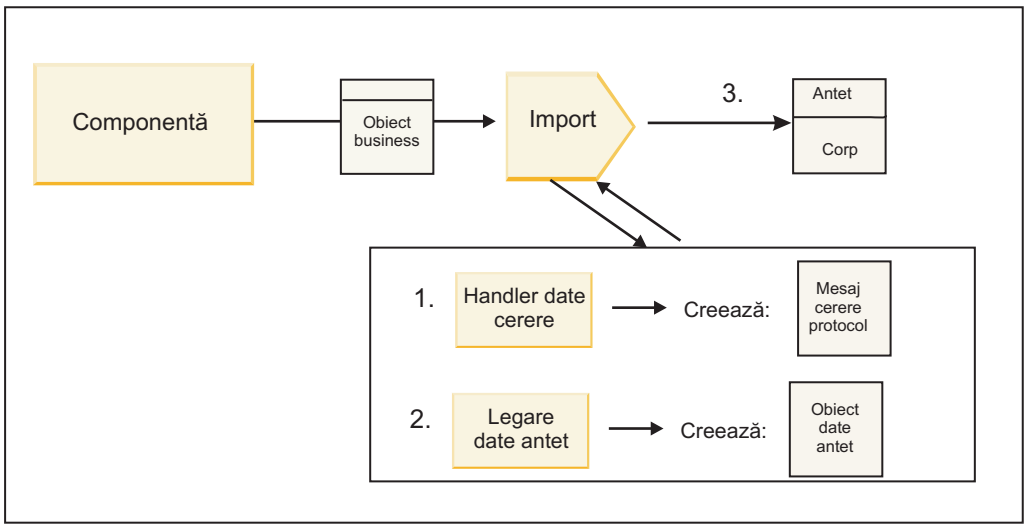


Figura 50. Fluxul de la o componentă prin import către un serviciu

Componenta invocă importul cu un obiect business de cerere.

Notă:

- Legarea de import HTTP, legarea de import a serviciului web şi legarea de import EJB ar trebui să fie invocate în mod sincron de componenta apelantă.
- Legarea de import JMS, Generic JMS, MQ JMS şi WebSphere MQ trebuie invocate asincron.

După ce componenta invocă importul, are loc următoarea secvenţă de evenimente:

1. Handler-ul de date sau legarea de date de cerere de pe metodă transformă obiectul business de cerere într-un mesaj de cerere de protocol.
2. Doar pentru legări WebSphere MQ, legarea de date de antet de pe metodă setează obiectul business de antet în antetul protocolului.
3. Importul invocă serviciul cu cererea de serviciu prin transport.

Dacă aceasta este o operaţie pe două-căi, serviciul returnează un răspuns şi are loc următoarea secvenţă de paşi.

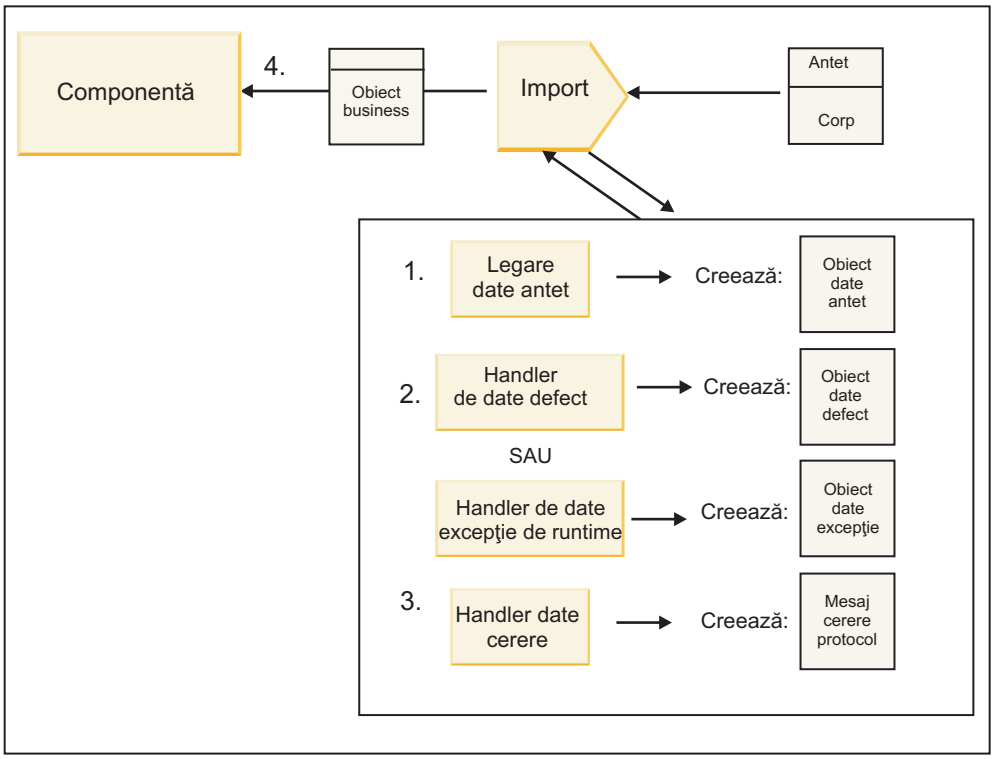


Figura 51. Fluxul unui răspuns înapoi prin import

1. Doar pentru legări WebSphere MQ, legarea datelor de antet transformă antetul protocolului într-un obiect de tip date de antet.
2. Este făcută o determinare dacă răspunsul este un defect sau nu.
   - Dacă răspunsul este un defect, selectorul de defecte inspectează defectul pentru a determina către ce defect WSDL mapează. Handler-ul de date de defect de pe metodă transformă atunci defectul într-un răspuns defect.
   - Dacă răspunsul este o excepţie din timpul rulării, handler-ul de date pentru excepţii din timpul rulării, dacă este configurat, este apelat.
3. Handler-ul de date sau legarea de răspuns de pe metodă transformă răspunsul într-un obiect business de răspuns.
4. Importul returnează obiectul business de răspuns componentei.

Configurarea legărilor de export şi import

Unul din aspectele cheie ale legărilor de export şi import este transformarea formatelor de date, ce indică cum sunt mapate (deserializate) datele de la un format de fire nativ la un obiect business sau cum sunt mapate (serializate) de la un obiect business la un format de fire nativ. Pentru legări asociate cu exporturi, puteţi de asemenea specifica un selector de funcţii care să indice ce operaţie ar trebui efectuată pe date. Pentru legări asociate cu exporturi sau importuri, puteţi indica cum ar trebui tratate defectele ce au loc în timpul procesării.

În plus, specificaţi informaţii specifice transportului pe legări. De exemplu, pentru o legare HTTP, specificaţi URL-ul punctului final. Pentru legarea HTTP, informaţiile specifice transportului sunt descrise în subiectele “Generarea unei legări de import HTTP” şi “Generarea unei legări de export HTTP”. Puteţi de asemenea găsi informaţii despre alte legări în centrul de informare.

Transformarea formatului de date în importuri şi exporturi

La configurarea unei legări export sau import în IBM Integration Designer, una dintre proprietăţile de configurare pe care le specificaţi este formatul de date utilizat de legare.

- Pentru legări de export, unde o aplicaţie client trimite cereri către şi primeşte răspunsuri de la o componentă SCA, indicaţi formatul datelor native. În funcţie de format, sistemul selectează handler-ul de date sau legarea de date

potrivită pentru a transforma datele native într-un obiect business (care este folosit de componenta SCA) şi invers pentru a transforma obiectul business în date native (care este răspunsul către aplicaţia client).

- Pentru legări de import, unde o componentă SCA trimite cereri către şi primeşte răspunsuri de la un serviciu din afara modulului, indicaţi formatul de date al datelor native. În funcţie de format, sistemul selectează handler-ul de date sau legarea de date potrivită pentru a transforma obiectul business în date native şi viceversa.

IBM Business Process Manager oferă un set de formate de date predefinite şi handler-e de date sau legări de date corespunzătoare ce suportă formatele. Vă puteţi de asemenea crea propriile handler-e de date personalizate şi înregistra formatul de date pentru acele handler-e de date. Pentru informaţii suplimentare, vedeţi subiectul "Dezvoltare handler-e de date" din Centrul de informare IBM Integration Designer.

- *Handler-ele de date* sunt neutre din punct de vedere al protocolului şi transformă datele dintr-un format în altul. În IBM Business Process Manager, handler-ele de date transformă în mod tipic datele native (precum XML, CSV şi COBOL) într-un obiect business şi un obiect business în date native. Deoarece sunt neutre din punct de vedere al protocolului, puteţi reutiliza acelaşi handler de date cu o varietate de legări de export şi import. De exemplu, puteţi folosi acelaşi handler de date XML cu o legare de export sau import HTTP sau cu o legare de export sau import JMS.
- *Legările de date* de asemenea transformă date native într-un obiect business (şi viceversa), dar sunt specifice-protocolului. De exemplu, o legare de date HTTP poate fi folosită doar cu o legare de export sau import HTTP. Spre deosebire de handler-ele de date, o legare de date HTTP nu poate fi refolosită cu o legare de export sau import MQ.

  Notă: Trei legări de date HTTP (HTTPStreamDataBindingSOAP, HTTPStreamDataBindingXML şi HTTPServiceGatewayDataBinding) sunt depreciate începând cu IBM Business Process Manager Versiunea 7.0. Folosiţi handler-e de date oricând este posibil.

După cum s-a menţionat mai devreme, puteţi crea handler-e de date personalizate, dacă este necesar. Puteţi de asemenea crea legări de date personalizate; totuşi, este recomandat să creaţi handler-e de date personalizate deoarece pot fi folosite peste multiple legări.

Handler-e de date:

Handler-ele de date sunt configurate funcţie de legările de export şi import pentru a transforma datele dintr-un format în altul într-un stil neutru protocolului. Mai multe handler-e de date sunt furnizate ca parte din produs, dar vă puteţi de asemenea crea propriul handler de date, dacă este necesar. Puteţi asocia un handler de date cu o legare de export sau import la unul din două niveluri: îl puteţi asocia cu toate operaţiile din interfaţa exportului sau importului sau îl puteţi asocia cu o anumită operaţie pentru cerere sau răspuns.

Handler-e de date predefinite

Utilizaţi IBM Integration Designer pentru a specifica handler-ul de date pe care vreţi să îl utilizaţi.

Handler-ele de date ce sunt predefinite pentru folosul dumneavoastră sunt menţionate în următorul tabel, ce de asemenea descrie cum fiecare handler de date transformă datele de intrare sau de ieşire.

Notă: Cu excepţia notificărilor, aceste handler-e de date pot fi utilizate cu legări JMS, Generic JMS, MQ JMS, WebSphere MQ şi HTTP.
Vedeţi subiectul "Handler-e de date" din centrul de informare Integration Designer pentru informaţii mai detaliate.

Tabela 46. Handler-e de date predefinite

Handler de date	Date native la obiect business	Obiect business la date native
ATOM	Parsează alimentări ATOM într-un obiect business de alimentare ATOM.	Serializează un obiect business de alimentare ATOM în alimentări ATOM.
Delimitat	Parsează date delimitate într-un obiect business.	Serializează un obiect business în date delimitate, inclusiv CSV.

Tabela 46. Handler-e de date predefinite (continuare)

Handler de date	Date native la obiect business	Obiect business la date native
Lăţime fixată	Parsează date cu lăţime fixată într-un obiect business.	Serializează un obiect business în date cu lăţime fixată.
Tratat de WTX	Delegă transformarea formatului datelor către WebSphere Transformation Extender (WTX). Numele mapării WTX este derivat de handler-ul de date.	Delegă transformarea formatului datelor către WebSphere Transformation Extender (WTX). Numele mapării WTX este derivat de handler-ul de date.
Tratat de WTX Invoker	Delegă transformarea formatului datelor către WebSphere Transformation Extender (WTX). Numele mapării WTX este furnizat de utilizator.	Delegă transformarea formatului datelor către WebSphere Transformation Extender (WTX). Numele mapării WTX este furnizat de utilizator.
JAXB	Serializează bean-urile Java într-un obiect business folosind regulile de mapare definite de specificaţia Java Architecture for XML Binding (JAXB).	Deserializează un obiect de business în bean-uri Java folosind regulile de mapare definite de specificaţia JAXB.
JAXWS **Notă:** Handler-ul de date JAXWS poate fi folosit doar cu legarea EJB.	Utilizat de o legare EJB pentru a transforma obiect Java răspuns sau obiect Java excepţie într-un obiect business răspuns folosind regulile de mapare definite de specificaţia Java API for XML Web Services (JAX-WS).	Utilizat de o legare EJB pentru a transforma un obiect business în parametri Java de ieşire folosind regulile de mapare definite de specificaţia JAX-WS.
JSON	Parsează date JSON într-un obiect business.	Serializează un obiect business în date JSON.
Corp nativ	Parsează octeţii, textul, maparea, fluxul sau obiectul nativ în unul din cinci obiecte business de bază (text, octeţi, mapare, flux sau obiect).	Transformă cele cinci obiecte business de bază în octet, text, mapare, flux sau obiect.
SOAP	Parsează mesajul SOAP (şi antetul) într-un obiect business.	Serializează un obiect business într-un mesaj SOAP.
XML	Parsează date XML într-un obiect business.	Serializează un obiect business în date XML.
UTF8XMLDataHandler	Parsează date XML codificate UTF-8 într-un obiect business.	Serializează un obiect business în date XML codificate UTF-8 când se trimite un mesaj.

Crearea unui handler de date

Informaţii detaliate despre crearea unui handler de date pot fi găsite în subiectul “Dezvoltarea handler-elor de date” din centrul de informare Integration Designer.

Legări de date:

Legările de date sunt configurate contra legărilor de export şi import pentru a transforma datele dintr-un format în altul. Legările de date sunt specifice unui protocol. Mai multe legări de date sunt oferite ca parte din produs, dar vă puteţi de asemenea crea propria legare de date, dacă este necesar. Puteţi asocia o legare de date cu o legare de export sau import la unul din două niveluri–o puteţi asocia cu toate operaţiile din interfaţa exportului sau importului sau o puteţi asocia cu o anumită operaţie pentru cerere sau răspuns.

Utilizaţi IBM Integration Designer pentru a specifica legarea de date pe care vreţi să o utilizaţi sau pentru a crea propria dumneavoastră legare de date. O discuţie despre crearea legăturilor de date poate fi găsită în secţiunea “Privire generală asupra legărilor JMS, MQ JMS şi JMS generic” a Centrului de informare IBM Integration Designer.

Legări JMS

Următorul tabel listează legările de date ce pot fi folosite cu:

- Legări JMS
- Legări Generic JMS
- Legări WebSphere MQ JMS

Tabelul include, de asemenea, o descriere a taskurilor realizate de legările de date.

Tabela 47. Legări de date predefinite pentru legări JMS

Legare de date	Date native la obiect business	Obiect business la date native
Obiect Java serializat	Transformă obiectul serializat Java într-un obiect business (care este mapat ca tipul de intrare sau ieşire în WSDL).	Serializează un obiect business în obiectul serializat Java în mesajul obiect JMS.
Octeţi înfăşuraţi	Extrage octeţii din mesajul de octeţi JMS de intrare şi îi înfăşoară în obiectul business JMSBytesBody.	Extrage octeţii din obiectul business JMSBytesBody şi îi înfăşoară în mesajul de octeţi JMS de ieşire
Intrare mapare înfăşurată	Extrage informaţiile despre nume, valoare şi tip pentru fiecare intrare din mesajul de mapare JMS de intrare şi creează o listă cu obiecte business MapEntry. Înfăşoară apoi lista în obiectul business JMSMapBody	Extrage informaţiile despre nume, valoare şi tip din lista MapEntry din obiectul business JMSMapBody şi creează intrările corespunzătoare în mesajul de mapare JMS de ieşire.
Obiect înfăşurat	Extrage obiectul din mesajul obiect JMS de intrare şi îl înfăşoară în obiectul business JMSObjectBody.	Extrage obiectul din obiectul business JMSObjectBody şi îl înfăşoară în mesajul obiect JMS de ieşire.
Text înfăşurat	Extrage textul din mesajul text JMS de intrare şi îl înfăşoară în obiectul business JMSTextBody.	Extrage textul din obiectul business JMSTextBody şi îl înfăşoară în mesajul text JMS de ieşire.

Legări WebSphere MQ

Următorul tabel listează legările de date ce pot fi folosite cu WebSphere MQ şi descrie taskurile realizate de legările de date.

Tabela 48. Legări de date predefinite pentru legări WebSphere MQ

Legare de date	Date native la obiect business	Obiect business la date native
Obiect Java serializat	Transformă obiectul serializat Java din mesajul de intrare într-un obiect business (care este mapat ca tipul de intrare sau ieşire în WSDL).	Transformă un obiect business în obiectul serializat Java din mesajul de ieşire
Octeţi înfăşuraţi	Extrage octeţii din mesajul de octeţi MQ nestructurat şi îi înfăşoară în obiectul business JMSBytesBody.	Extrage octeţii dintr-un obiect business JMSBytesBody şi înfăşoară octeţii în mesajul de octeţi MQ nestructurat de ieşire.
Text înfăşurat	Extrage textul dintr-un mesaj text MQ nestructurat şi îl înfăşoară într-un obiect business JMSTextBody.	Extrage textul dintr-un obiect business JMSTextBody şi îl înfăşoară într-un mesaj text MQ nestructurat.
Intrare flux înfăşurată	Extrage informaţiile despre nume şi tip pentru fiecare intrare din mesajul flux JMS de intrare şi creează o listă cu obiectele business StreamEntry. Înfăşoară apoi lista în obiectul business JMSStreamBody.	Extrage informaţiile despre nume şi tip din lista StreamEntry din obiectul business JMSStreamBody şi creează intrări corespunzătoare în JMSStreamMessage de ieşire.

În plus faţă de legările de date menţionate în Tabela 20 la pagina 57, WebSphere MQ foloseşte de asemenea legări de date de antet. Vedeţi Centrul de informare IBM Integration Designer pentru detalii.

Legări HTTP

Următorul table listează legările de date ce pot fi folosite cu HTTP şi descrie taskurile realizate de legările de date.

Tabela 49. Legări de date predefinite pentru legări HTTP

Legare de date	Date native la obiect business	Obiect business la date native
Octeţi înfăşuraţi	Extrage octeţii din corpul mesajului HTTP de intrare şi îi înfăşoară în obiectul business HTTPBytes.	Extrage octeţii din obiectul business HTTPBytes şi îi adaugă la corpul mesajului HTTP de ieşire.
Text înfăşurat	Extrage textul din corpul mesajului HTTP de intrare şi îl înfăşoară în obiectul business HTTPText.	Extrage textul din obiectul business HTTPText şi îl adaugă la corpul mesajului HTTP de ieşire.

Selectori de funcţii în legări de export

Un selector de funcţii este folosit pentru a indica ce operaţie ar trebui efectuată pe date pentru un mesaj cerere. Selectorii de funcţii sunt configuraţi ca parte dintr-o legare de export.

Consideraţi un export SCA ce expune o interfaţă. Interfaţa conţine două operaţii–Creare şi Actualizare. Exportul are o legare JMS ce citeşte dintr-o coadă.

Când un mesaj ajunge în coadă, exportului îi sunt transmise datele asociate, dar ce operaţie din interfaţa exportului ar trebui invocată pe componenta cablată? Operaţia este determinată de selectorul de funcţii şi configuraţia legării de export.

Selectorul de funcţii returnează numele funcţiei native (numele funcţiei din sistemul client ce a trimis mesajul). Numele funcţiei native este apoi mapat de numele operaţiei sau funcţiei de pe interfaţa asociată cu exportul. De exemplu, în următoarea figură, selectorul de funcţii returnează numele funcţiei native (CRT) din mesajul de intrare, numele funcţiei native este mapat de operaţia Creare şi obiectul business este trimis componentei SCA cu operaţia Creare.

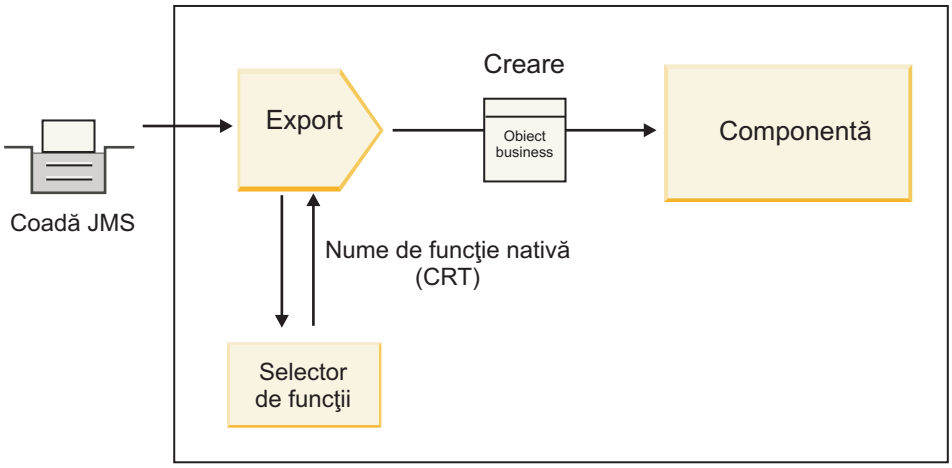


Figura 52. Selectorul de funcţii

Dacă interfaţa are o singură operaţie, nu este nevoie să specificaţi un selector de funcţii.

Mai mulţi selectori de funcţii preambalaţi sunt disponibili şi sunt menţionaţi în secţiunile ce urmează.

Legări JMS

Următorul tabel listează selectorii de funcţii ce pot fi folosiţi cu:

- Legări JMS
- Legări JMS generice
- Legări WebSphere MQ JMS

Tabela 50. Selectori de funcţii predefiniţi pentru legări JMS

Selector de funcţii	Descriere
Selector de funcţii JMS pentru legări simple de date JMS	Foloseşte proprietatea JMSType a mesajului pentru a selecta numele operaţiei.
Selector de funcţii pentru proprietatea antetului JMS	Returnează valoarea Proprietăţii şir JSM, TargetFunctionName, din antet.
Selector funcţie gateway de servicii JMS	Determină dacă cererea este o operaţie cu un singur sens sau cu sens dublu prin examinarea proprietăţii JMSReplyTo setată de client.

Legări WebSphere MQ

Următorul tabel listează selectorii de funcţii ce pot fi utilizaţi cu legări WebSphere MQ.

Tabela 51. Selectorii de funcţii predefiniţi pentru legări WebSphere MQ

Selector de funcţii	Descriere
Selector de funcţii MQ handleMessage	Returnează handleMessage ca o valoare ce este mapată folosind legările metodei de export de numele unei operaţii de pe interfaţă.
MQ foloseşte selector de funcţii implicit JMS	Citeşte operaţia nativă din proprietatea TargetFunctionName din folderul unui antet MQRFH2.
MQ foloseşte formatul corpului mesajului ca funcţie nativă	Găseşte câmpul Format al ultimului antet şi returnează acel câmp ca un Şir.
Selector de funcţii de tip MQ	Creează o metodă în legarea dumneavoastră de export prin extragerea unui URL ce conţine proprietăţile Msd, Set, Type şi Format aflate în antetul MQRFH2.
Selector funcţie gateway de servicii MQ	Foloseşte proprietatea MsgType din antetul MQMD pentru a determina numele proprietăţii.

Legări HTTP

Următorul tabel listează selectorii de funcţii ce pot fi utilizaţi cu legări HTTP.

Tabela 52. Selectori de funcţii predefiniţi pentru legări HTTP

Selector de funcţii	Descriere
Selector de funcţii HTTP bazat pe antetul TargetFunctionName	Utilizează proprietatea antetului HTTP TargetFunctionName de la client pentru a determina ce operaţie să invoce în momentul rulării din export.
Selector de funcţii HTTP bazat pe metoda URL şi HTTP	Foloseşte calea relativă de la URL adăugată la sfârşitul metodei HTTP de la client pentru a determina operaţia nativă definită pe export.
Selector funcţie gateway de servicii HTTP bazat pe URL cu un nume de operaţie	Determină metoda de invocat în funcţie de URL dacă s-a adăugat "operationMode = oneway" la URL-ul cererii.

Notă: Vă puteţi de asemenea crea propriul selector de funcţii, folosind IBM Integration Designer. Informaţii despre crearea unui selector de funcţii sunt furnizate în centrul de informare IBM Integration Designer. De exemplu, o descriere a creării unui selector de funcţii pentru legări WebSphere MQ poate fi găsită în “Privire generală asupra selectorilor de funcţii MQ”.

Tratarea defectelor

Vă puteţi configura legările de import şi export să trateze defecte (de exemplu, excepţii operaţionale) ce au loc în timpul procesării prin specificarea handler-elor de date ale defectelor. Puteţi configura un handler de date al defectelor la trei niveluri–puteţi asocia un handler de date al defectelor cu un defect, cu o operaţie sau pentru toate operaţiile cu o legare.

Un handler de date al defectelor procesează date ale defectelor şi le transformă în formatul corect pentru a fi trimise de legarea de export sau import.

- Pentru o legare de export, handler-ul de date al defectelor transformă obiectul business al excepţiei trimis de la componentă într-un mesaj răspuns ce poate fi folosit de aplicaţia client.
- Pentru o legare de import, handler-ul de date al defectelor transformă datele defectelor sau mesajul răspuns trimis de la un serviciu într-un obiect business al excepţiei ce poate fi folosit de componenta SCA.

Pentru legări de import, legarea apelează selectorul de defecte, ce determină dacă mesajul răspuns este un răspuns normal, un defect operaţional sau o excepţie runtime.

Puteţi specifica un handler de date ale defectelor pentru un anumit defect, pentru o operaţie şi pentru toate operaţiile cu o legare.

- Dacă handler-ul de date ale defectelor este setat la toate cele trei niveluri, handler-ul de date asociat cu un anumit defect este apelat.
- Dacă handler-ele de date ale defectelor sunt setate la nivelurile de operaţie şi legare, este apelat handler-ul de date asociat cu operaţia.

Sunt folosite două editoare în IBM Integration Designer pentru specificarea tratării defectelor. Editorul de interfeţe este folosit pentru a indica dacă va fi un defect pe o operaţie. După ce este generată o legare cu această interfaţă, editorul din vizualizarea proprietăţilor vă dă voie să configuraţi cum va fi tratat defectul. Pentru informaţii suplimentare, vedeţi subiectul “Selectori de defecte” din centrul de informare IBM Integration Designer.

Cum sunt tratate defectele în legările de export:

Când apare un defect în timpul procesării cererii de la o aplicaţie client, legarea de export poate returna informaţiile despre defect clientului. Configuraţi legarea de export să specifice cum ar trebui procesat defectul şi returnat clientului.

Configuraţi legarea de export folosind IBM Integration Designer.

În timpul procesării cererii, un client invocă un export cu o cerere şi exportul invocă componenta SCA. În timpul procesării cererii, componenta SCA poate fie returna un răspuns operaţional fie poate arunca o excepţie operaţională de serviciu sau o excepţie runtime de serviciu. Când apare aceasta, legarea de export transformă excepţia într-un mesaj de defect şi îl trimite clientului, după cum este arătat în figura următoare şi descris în secţiunile ce urmează.

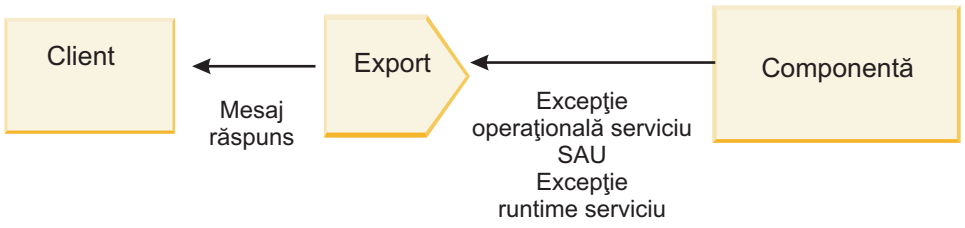


Figura 53. Cum sunt trimise informaţiile despre defect de la componentă prin legarea de export clientului

Puteţi crea un handler de date sau o legare de date personalizată pentru a trata defectele.

Defecte operaţionale

Defectele operaţionale sunt erori sau excepţii operaţionale ce au loc în timpul procesării.

Consideraţi următoarea interfaţă, ce are o operaţie createCustomer pe ea. Această operaţie are două defecte operaţionale definite: CustomerAlreadyExists şi MissingCustomerId.

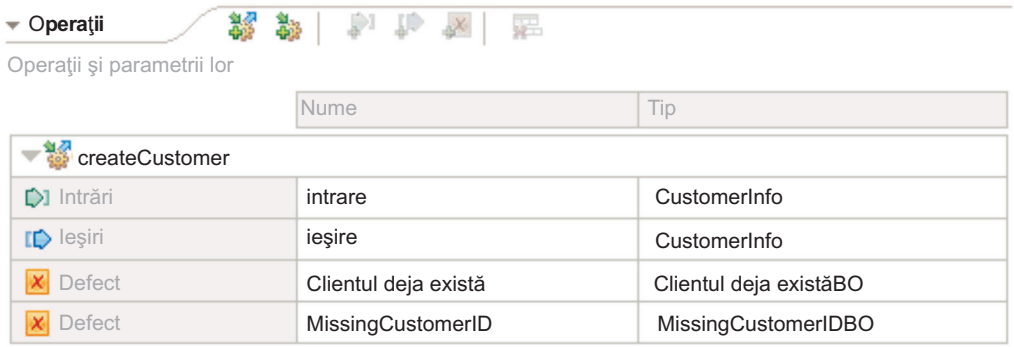


Figura 54. Interfaţă cu două defecte

În acest exemplu, dacă un client trimite o cerere pentru crearea unui client (acestei componente SCA) şi acel client există deja, componenta aruncă un defect CustomerAlreadyExists exportului. Exportul trebuie să propage acest defect operaţional înapoi la clientul apelant. Pentru a face aceasta, utilizează handler-ul de date de defect configurat pe legarea de export.

Când un defect operaţional este primit de legarea de export, are loc următoarea procesare:

1. Legarea determină ce handler de date de defect să invoce pentru tratarea defectului. Dacă excepţia operaţională de servicii conţine numele defectului, este apelat handler-ul de date ce este configurat pe defect. Dacă excepţia operaţională de servicii nu conţine numele defectului, numele defectului este derivat prin potrivirea cu tipurile de defecte.
2. Legarea apelează handler-ul de date de defecte cu obiectul de date din excepţia operaţională de servicii.
3. Handler-ul de date de defecte transformă obiectul de date de defect într-un mesaj răspuns şi îl returnează legării de export.
4. Exportul returnează mesajul răspuns clientului.

Dacă excepţia operaţională de servicii conţine numele defectului, este apelat handler-ul de date ce este configurat pe defect. Dacă excepţia operaţională de servicii nu conţine numele defectului, numele defectului este derivat prin potrivirea cu tipurile de defecte.

Excepţii în timpul rulării

O excepţie runtime este o excepţie ce are loc în aplicaţia SCA în timpul procesării unei cereri ce nu corespunde unui defect operaţional. Spre deosebire de defectele operaţionale, excepţiile runtime nu sunt definite pe interfaţă.

În anumite scenarii, aţi putea dori să propagaţi aceste excepţii runtime către aplicaţia client, astfel încât aplicaţia client să poată lua măsurile corespunzătoare.

De exemplu, dacă un client trimite o cerere (componentei SCA) pentru a crea un client şi are loc o eroare de autorizare în timpul procesării cererii, componenta aruncă o excepţie runtime. Această excepţie runtime trebuie să fie propagată înapoi către clientul apelant, astfel încât să poată lua măsurile necesare referitor la autorizaţie. Aceasta se obţine prin handler-ul de date al excepţiei runtime configurat pe legarea de export.

Notă: Puteţi configura un handler de date al excepţiei runtime doar pe legări HTTP.

Procesarea unei excepţii runtime este similară cu procesarea unui defect operaţional. Dacă a fost configurat un handler de date al excepţiei runtime, are loc următoarea procesare:

1. Legarea de export apelează handler-ul de date corespunzător cu excepţia runtime de servicii.
2. Handler-ul de date transformă obiectul de date al defectului într-un mesaj răspuns şi îl returnează legării de export.
3. Exportul returnează mesajul răspuns clientului.

Tratarea defectelor şi tratarea excepţiilor runtime sunt opţionale. Dacă nu doriţi să propagaţi defecte sau excepţii runtime către clientul apelant, nu configuraţi handler-ul de date al defectelor sau handler-ul de date al excepţiei runtime.

Modul în care defectele sunt tratate în legările de import:

O componentă foloseşte un import pentru a trimite o cerere către un serviciu care se află în afara modulului. Atunci când apare un defect în timpul procesării cererii, serviciul returnează defectul către legarea de import. În momentul în care configuraţi legarea de import puteţi specifica modul în care defectul ar trebui procesat şi returnat către componentă.

Configuraţi legarea de import folosind IBM Integration Designer. Puteţi specifica un handler pentru datele ce conţin defecte (sau o legare de date), şi, de asemenea, puteţi specifica un selector pentru defect.

Handler-e pentru datele ce conţin defecte

Serviciul care procesează cererea trimite, către legarea de import, informaţii despre defect sub forma unei excepţii sau a unui mesaj de răspuns care conţine datele de defect.

Legarea de import transformă excepţia serviciului sau mesajul de răspuns într-un excepţie a serviciului operaţional sau o excepţie de runtime serviciu, după cum se arată în figura de mai jos şi cum este descris în secţiunile care urmează.

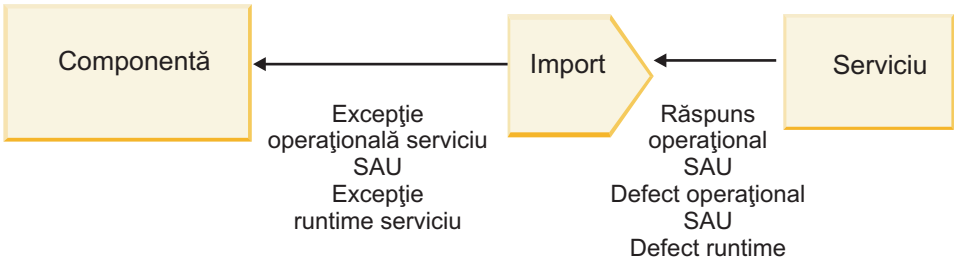


Figura 55. Modul în care informaţiile despre eroare sunt transmise cu ajutorul importului de la serviciu către componentă

Puteţi crea un handler de date personalizat sau o legare de date pentru a trata defectele.

Selectori de defect

Atunci când configuraţi o legare de import, aveţi posibilitatea să specificaţi un selector de defect. Selectorul de defect determină dacă răspunsul de import este un răspuns real, o excepţie operaţională sau un defect apărut în timpul rulării. De asemenea, determină din corpul sau antetul răspunsului, numele nativ al defectului, care este mapat prin configuraţia legării de numele defectului din interfaţa asociată.

Două tipuri de selectoare pentru defecte împachetate sunt disponibile pentru utilizarea cu importurile JMS, MQ JMS, Generic JMS, WebSphere MQ şi HTTP:

Tabela 53. Selectoare pentru defecte împachetate

Tip selector defect	Descriere
Bazat pe antet	Stabileşte dacă un mesaj de răspuns este un defect operaţional, o excepţie apărută în timpul rulării sau un mesaj normal bazat pe anteturile din mesajul de răspuns de intrare.
SOAP	Determină dacă mesajul SOAP de răspuns este un răspuns normal, o defectare a afacerii sau o excepţie în timpul rulării.

În continuare sunt arătate exemple de selectoare pentru anteturi bazate pe defecte şi pentru selectorul de defect SOAP.

- Selector de defect bazat pe antet

  Dacă o aplicaţie doreşte să indice că mesajul primit este un defect operaţional, atunci trebuie să existe două anteturi în mesajul de intrare pentru defectele operaţionale, aşa cum este prezentat în continuare:

  ```
  Header name = FaultType, Header value = Business
  Header name = FaultName, Header value = <user defined native fault name>
  ```

  Dacă o aplicaţie doreşte să indice că mesajul primit este o excepţie apărută în timpul rulării, atunci trebuie să existe un antet în mesajul de intrare, aşa cum este prezentat în continuare:

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

- Selector de defect SOAP

  Un defect operaţional poate fi trimis ca parte componentă a mesajului SOAP care are în compoziţie următorul antet SOAP personalizat. "CustomerAlreadyExists" este numele defectării în acest caz.

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

Selectorul defectării este opţional. Dacă nu specificaţi un selector pentru defectare, legarea de import nu poate determina tipul răspunsului. Prin urmare, legarea îl tratează ca un răspuns operaţional (business) şi apelează handler-ul datelor de răspuns sau legarea de date.

Puteţi crea un selector personalizat pentru defect. Paşii pentru crearea unui selector personalizat pentru defect sunt furnizate în subiectul “Dezvoltarea unui selector personalizat pentru defect” din Centrul de informare IBM Integration Designer.

Defecte operaţionale

O eroare operaţională poate să apară atunci când există o eroare în procesarea unei cereri. De exemplu, dacă trimiteţi o cerere pentru a crea un client, iar acel client există deja, serviciul trimite o excepţie operaţională către legarea de import.

Atunci când legarea primeşte o excepţie operaţională, etapele de prelucrare depind de faptul dacă un selector de defect a fost setat pentru legare.

- Dacă nu a fost setat nici un selector de defect, legarea apelează handler-ul de date pentru răspuns sau legarea de date.
- Dacă a fost setat un selector de defect, are loc următoarea prelucrare:
  1. Legarea de import apelează selectorul de defect pentru a determina dacă răspunsul este un defect operaţional (business) sau un defect apărut în timpul rulării.
  2. În cazul în care răspunsul este un defect operaţional, legarea de import apelează selectorul de defect pentru a furniza numele nativ al defectului.
  3. Legarea de import determină defectul WSDL corespunzător numelui nativ al defectului returnat de către selectorul de defect.
  4. Legarea de import determină handler-ul da date pentru defect care este configurat pentru acest defect WSDL.
  5. Legarea de import apelează acest handler-ul da date pentru defect cu datele defectului.
  6. Handler-ul da date pentru defect transformă datele defectului într-un obiect de date şi îl returnează către legarea de import.

7. Legarea de import construieşte un obiect de tip excepţie pentru serviciul operaţional cu obiectul de date şi numele defectului.
8. Importul returnează obiectul de tip excepţie pentru serviciul operaţional către componentă.

Excepţii din timpul rulării

O excepţie de runtime poate apărea atunci când există o problemă în comunicarea cu serviciul. Procesarea unei excepţii apărute în timpul rulării este similară cu cea a unei excepţii operaţionale (business). Dacă a fost setat un selector de defect, are loc următoarea prelucrare:

1. Legarea de import apelează handler-ul corespunzător pentru datele excepţiei runtime cu datele excepţiei.
2. Handler-ul da date pentru excepţia runtime transformă datele excepţiei într-un obiect de tip excepţie runtime serviciu şi îl returnează către legarea de import.
3. Importul returnează obiectul excepţie runtime serviciu către componentă.

Interoperabilitatea între modulele SCA şi serviciile Open SCA

IBM WebSphere Application Server V7.0 Feature Pack for Service Component Architecture (SCA) oferă un model de programare simplu, dar foarte puternic pentru construirea aplicaţiilor bazate pe specificaţiile Open SCA. Modulele SCA din IBM Business Process Manager folosesc legările de import şi export pentru a interopera cu serviciile Open SCA dezvoltate într-un mediu Rational Application Developer şi găzduite de WebSphere Application Server Feature Pack for Service Component Architecture.

O aplicaţie SCA invocă o aplicaţie Open SCA prin intermediul unei legări de import. O aplicaţie SCA primeşte un apel de la aplicaţie Open SCA prin intermediul unei legări de export. O listă de legări acceptate este prezentată în “Invocarea serviciilor peste legările interoperabile” la pagina 66.

Invocarea serviciilor Open SCA din module SCA

Aplicaţiile SCA dezvoltate în IBM Integration Designer pot invoca aplicaţiile Open SCA dezvoltate într-un mediu Rational Application Developer. Această secţiune oferă un exemplu de invocare a un serviciu Open SCA dintr-un modul SCA utilizând o legare de import SCA.

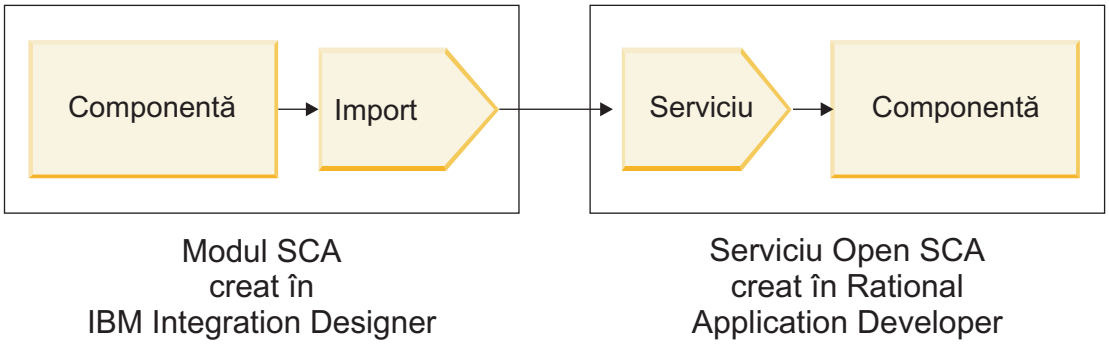


Figura 56. Componenta din modulul SCA invocă un serviciu Open SCA

Nu este nevoie de nici o configuraţie specială pentru a invoca un serviciu Open SCA.

Pentru a vă conecta la un serviciu Open SCA prin intermediul unei legări de import SCA, furnizaţi numele componentei şi numele serviciului pentru serviciul Open SCA în legarea de import.

1. Pentru a obţine numele componentei ţintă şi serviciul din Open SCA compus, efectuaţi următorii paşi:
   a. Asiguraţi-vă că fila **Proprietăţi** este deschisă făcând clic pe **Fereastră > Afişare vizualizare > Proprietăţi**.

b. Deschideţi editorul pentru compoziţie făcând dublu-clic pe diagrama compusă care conţine componenta şi serviciul. De exemplu, pentru o componentă ce are numele **customer**, diagrama compusă este **customer.composite_diagram**.
c. Faceţi clic pe componenta ţintă.
d. În câmpul **Nume** din fila **Proprietăţi**, notaţi numele componentei ţintă.
e. Faceţi clic pe pictograma serviciului asociată cu componenta.
f. În câmpul **Nume** din fila **Proprietăţi**, notaţi numele serviciului.

2. Pentru a configura importul IBM Business Process Manager, astfel încât să se conecteze la serviciul Open SCA, efectuaţi paşii următori:
   a. În IBM Integration Designer, mergeţi la fila **Proprietăţi** din importul SCA pe care vreţi să îl conectaţi la serviciul Open SCA.
   b. În câmpul **Nume modul**, introduceţi numele componentei de la pasul 1d la pagina 65.
   c. În câmpul **Nume export**, introduceţi numele serviciului de la pasul 1f la pagina 65.
   d. Salvaţi-vă munca apăsând Ctrl+S.

Invocarea modulelor SCA din serviciile Open SCA

Deschideţi aplicaţiile SCA dezvoltate într-un mediu Rational Application Developer. Acest mediu poate invoca aplicaţiile SCA dezvoltate cu IBM Integration Designer. Această secţiune oferă un exemplu de invocare a unui modul SCA (prin intermediul unei legări de export SCA) dintr-un serviciu Open SCA.

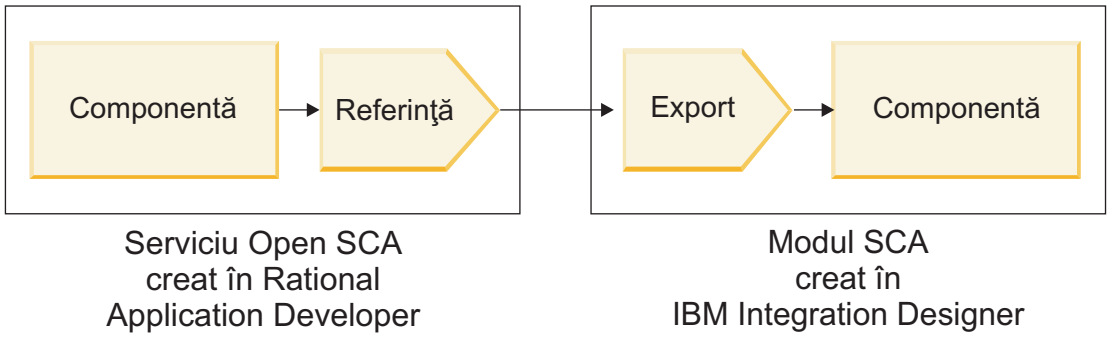


Figura 57. Deschiderea serviciului SCA prin invocarea componentei în modulul SCA

Pentru a vă conecta la o componentă SCA printr-o legare de referinţă a unui Open SCA, furnizaţi numele modulului şi numele de export.

1. Pentru a obţine numele modulului şi exportului ţintă, efectuaţi următorii paşi:
   a. În IBM Integration Designer, deschideţi modulul în editorul de asamblare făcând dublu-clic pe modul.
   b. Faceţi clic pe export.
   c. În câmpul **Nume** din fila **Proprietăţi**, notaţi numele exportului.
2. Configuraţi referinţa Open SCA pe care doriţi să o conectaţi la modulul IBM Business Process Manager şi exportaţi:
   a. În Rational Application Developer, deschideţi editorul pentru compoziţie făcând dublu-clic pe diagrama compusă care conţine componenta şi serviciul.
   b. Faceţi clic pe pictograma referinţei componentei pentru a afişa aceste proprietăţi de referinţă în fila **Proprietăţi**.
   c. Faceţi clic pe fila **Legare** din partea stânga a paginii.
   d. Faceţi clic pe **Legări**, iar apoi faceţi clic pe **Adăugare**.
   e. Selectaţi legarea **SCA**.

f. În câmpul **Uri**, introduceţi numele modulului IBM Business Process Manager, urmat de o bară oblică (“/”), urmat de numele de export (pe care îl stabiliţi la pasul 1c la pagina 65).

g. Apăsaţi **OK**.

h. Salvaţi-vă munca apăsând Ctrl+S.

Invocarea serviciilor peste legările interoperabile

Sunt acceptate următoarele legări pentru interoperabilitatea cu un serviciu Open SCA.

- Legare SCA

  În IBM Business Process Manager, atunci când un modul SCA invocă un serviciu Open SCA prin intermediul unei legături de import SCA, sunt acceptate următoarele stiluri de invocare:

  – Asincron (într-un singur sens)

  – Sincron (cerere/răspuns)

  Interfaţa de import SCA şi interfaţa serviciului Open SCA trebuie să utilizeze o interfaţă WSDL conformă WS-I.

  Reţineţi că legarea SCA suportă propagarea contextului de tranzacţie şi de securitate.

- Legare (JAX-WS) a serviciului Web fie cu protocol SOAP1.1/HTTP, fie cu protocol SOAP1.2/HTTP

  Interfaţa de import SCA şi interfaţa serviciului Open SCA trebuie să utilizeze o interfaţă WSDL conformă WS-I.

  În plus, sunt suportate următoarele calităţi ale serviciului:

  – Tranzacţia atomică a serviciilor Web

  – Securitatea serviciilor Web

- Legare EJB

  Pentru a defini interacţiunea dintre un modul SCA şi un serviciu Open SCA se foloseşte o interfaţă Java atunci când este utilizată legarea EJB.

  Reţineţi că legarea EJB suportă propagarea contextului de tranzacţie şi de securitate.

- Legări JMS

  Interfaţa de import SCA şi interfaţa serviciului Open SCA trebuie să utilizeze o interfaţă WSDL conformă WS-I.

  Sunt suportaţi următorii furnizori JMS:

  – Platforma de mesagerie WebSphere (Legarea JMS)

  – WebSphere MQ (Legarea MQ JMS)

Notă: Graficele operaţionale nu sunt interoperabile pe orice legări SCA şi, prin urmare, nu sunt acceptate în interfeţele utilizate pentru a interopera cu WebSphere Application Server Feature Pack for Service Component Architecture.

Tipuri de legări

Folosiţi *legări* specifice-protocolului cu importuri şi exporturi pentru a specifica mijloacele de transportare a datelor în sau dintr-un modul.

Selectarea legărilor corespunzătoare

Atunci când creaţi o aplicaţie, trebuie să ştiţi cum să selectaţi legarea care corespunde cel mai bine necesităţilor aplicaţiei dumneavoastră.

Legările disponibile din IBM Integration Designer furnizează o gamă de alegeri. În această listă puteţi determina care tip de legătură este mai potrivită nevoilor aplicaţiei dumneavoastră.

Luaţi în considerare o legătură *Service Component Architecture (SCA)* atunci când se pot aplica aceşti factori:

- Toate serviciile sunt conţinute în module; ceea ce înseamnă că nu există servicii externe.
- Vreţi să separaţi funcţia în module SCA diferite care interacţionează direct unul cu celălalt.
- Modulele sunt cuplate strâns.

Luaţi în considerare o legare *serviciu web* atunci când se aplică aceşti factori:

- Trebuie să accesaţi un serviciu extern de pe Internet sau să furnizaţi un serviciu pe Internet.
- Serviciile sunt cuplate slab.
- Comunicaţia sincronă este preferabilă; adică o cerere de la un serviciu poate aştepta pentru un răspuns de la altul.
- Protocolul serviciului extern pe care îl accesaţi sau al serviciului pe care vreţi să îl furnizaţi este SOAP/HTTP sau SOAP/JMS.

Consideraţi o legare *HTTP* când aceşti factori sunt aplicabili:
- Este nevoie să accesaţi un serviciu extern de pe Internet sau să furnizaţi un serviciu de pe Internet şi lucraţi cu alte servicii web cum ar fi GET, PUT şi DELETE.
- Serviciile sunt cuplate slab.
- Comunicaţia sincronă este preferabilă; adică o cerere de la un serviciu poate aştepta pentru un răspuns de la altul.

Luaţi în considerare o legătură *Enterprise JavaBeans (EJB)* atunci când se pot aplica aceşti factori:
- Legarea este pentru un serviciu importat care este el însuşi un EJB sau care trebuie să fie accesat de clienţi EJB.
- Serviciul importat este cuplat slab.
- Interacţiunile EJB stateful nu sunt necesare.
- Comunicaţia sincronă este preferabilă; adică o cerere de la un serviciu poate aştepta pentru un răspuns de la altul.

Luaţi în considerare o legătură *Enterprise Information Szstems(EIS)* atunci când se pot aplica aceşti factori:
- Trebuie să accesaţi un serviciu de pe un sistem EIS utilizând un adaptor de resurse.
- Transmisia de date sincronă este preferată în locul celei asincrone.

Luaţi în considerare o legătură *Java Message Service (JMS)* atunci când se pot aplica aceşti factori:

Important: Există mai multe tipuri de legări JMS. Dacă vă aşteptaţi să schimbaţi mesajele SOAP folosind JMS, luaţi în considerare legarea de servicii web cu protocolul SOAP/JMS. Vedeţi “Legări de serviciu Web” la pagina 68.
- Trebuie să accesaţi un sistem de mesagerie.
- Serviciile sunt cuplate slab.
- Transmisia de date asincron este preferată în locul celei sincron.

Luaţi în considerare o legătură *Generic Java Message Service (JMS)* atunci când se pot aplica aceşti factori:
- Trebuie să accesaţi sistem de mesagerie a unui furnizor non-IBM.
- Serviciile sunt cuplate slab.
- Fiabilitatea este mai importantă decât performanţa; adică, transmisia de date asincronă este preferată în locul celei sincrone.

Luaţi în considerare o legătură *Message Queue (MQ)* atunci când se pot aplica aceşti factori:
- Trebuie să accesaţi un sistem de mesagerie WebSphere MQ şi să utilizaţi funcţiile native MQ.
- Serviciile sunt cuplate slab.
- Fiabilitatea este mai importantă decât performanţa; adică, transmisia de date asincronă este preferată în locul celei sincrone.

Luaţi în considerare o legare *MQ JMS* când se aplică aceşti factori:
- Trebuie să accesaţi un sistem de mesagerie WebSphere MQ dar puteţi face astfel într-un context JMS; adică, subsetul JMS al funcţiilor este suficient pentru aplicaţia dumneavoastră.
- Serviciile sunt cuplate slab.
- Fiabilitatea este mai importantă decât performanţa; adică, transmisia de date asincronă este preferată în locul celei sincrone.

Legări SCA

O legarea SCA (Service Component Architecture) permite unui serviciu să comunice cu alte servicii din alte module. Un import cu o legare SCA vă permite să accesaţi un serviciu dintr-un alt modul SCA. Un export cu o legare SCA vă permite să oferiţi un serviciu către alte module.

Utilizaţi IBM Integration Designer pentru a genera şi configura legături SCA în importuri şi exporturi din module SCA.

În cazul în care modulele rulează pe acelaşi server sau sunt implementate în acelaşi cluster, o legare SCA este cel mai uşor şi cel mai rapid tip de legare ce poate fi utilizat.

După ce modulul care conţine legarea SCA este implementat pe server, puteţi utiliza consola administrativă pentru a vizualiza informaţii despre legare, sau în cazul unei legări de import, pentru a modifica proprietăţile selectate ale legării.

Legări de serviciu Web

Legarea unui serviciu web este mijlocul de transmitere a mesajelor de la o componentă SCA (Service Component Architecture) către un serviciu web (şi vice versa).

Privire generală asupra legărilor de serviciu Web:

O legare de import a serviciului web vă permite să apelaţi un serviciu web extern din componentele dvs SCA (Service Component Architecture). O legare de export a serviciului web vă permite să expuneţi componentele SCA clienţilor sub formă de servicii web.

Cu o legare de serviciu web, puteţi accesa serviciile externe folosind mesaje SOAP interoperabile şi calităţile de serviciu (QoS).

Utilizaţi Integration Designer pentru a genera şi configura legările serviciului web la importuri şi exporturi în modeule SCA. Sunt disponibile următoarele tipuri de legături de servicii web:

- SOAP1.2/HTTP şi SOAP1.1/HTTP

  Aceste b se bazează pe API-ul Java API pentru serviciile web XML (JAX-WS), un API de programare Java pentru crearea serviciilor web.

  – Utilizaţi SOAP1.2/HTTP dacă serviciul dvs. web corespunde specificaţiei SOAP 1.2.
  – Utilizaţi SOAP1.1/HTTP dacă serviciul dvs. web corespunde specificaţiei SOAP 1.1.

  Important: Când implementaţi o aplicaţie cu o legare de serviciu web (JAX-WS), serverul ţintă nu trebuie să aibă selectată opţiunea **Pornire componente după cum este necesar**. Pentru detalii, vedeţi “Verificarea configuraţiei serverului” la pagina 76.

  Când selectaţi una din aceste legări, puteţi trimite ataşamente cu mesajele dvs. SOAP.

  Legările serviciului web funcţionează cu mesaje SOAP standard. Folosind una dintre legările serviciului web JAX-WS, totuşi, aveţi posibilitatea să personalizaţi modul în care sunt parsate sau scrise mesajele SOAP. De exemplu, puteţi manipula elementele care nu sunt standard în mesajele SOAP sau puteţi aplica prelucrări suplimentare mesajelor SOAP. Când configuraţi legarea specificaţi un handler personalizat pentru date, care efectuează această prelucrare într-un mesaj SOAP.

  Puteţi folosi seturi de politici cu o legare a serviciului web (JAX-WS). Un set de politici este o colecţie de tipuri de politici, fiecare dintre ele oferind o calitate a serviciilor (QoS). De exemplu, setul de politici WSAddressing oferă o cale de transport neutră pentru adresarea uniformă a serviciilor web şi a mesajelor. Utilizaţi Integration Designer pentru a selecta setul de politici pentru legare.

  Notă: Dacă doriţi să utilizaţi un set de politici de tip SAML (Security Assertion Markup Language), trebuie să realizaţi câteva configuraţii suplimentare, aşa cum este descris în “Importul seturilor de politici” la pagina 74.

- SOAP1.1/HTTP

  Utilizaţi această legare dacă doriţi să creaţi servicii web care folosesc un mesaj codat SOAP care se bazează pe JAX-RPC (Java API for XML-based RPC).

- SOAP1.1/JMS

  Utilizaţi această legare pentru a trimite sau pentru a primi mesaje SOAP folosind o destinaţie JMS (Java Message Service).

Indiferent de transportul (HTTP sau JMS) utilizat pentru transmiterea mesajelor SOAP, legările serviciului manipulează întotdeauna interacţiunile cerere/răspuns în mod sincron. Firul de execuţie care face invocarea către furnizorul de servicii este blocat până când este primit un răspuns de la furnizor. Vedeţi “Invocarea sincronă” pentru informaţii suplimentare despre acest stil de invocare.

Important: Următoarele combinaţii de legături de servicii web nu pot fi utilizate pentru exporturile către acelaşi modul. Dacă aveţi nevoie să expuneţi componente folosind mai mult de una dintre aceste legări de export, este nevoie ca fiecare dintre ele să fie într-un modul separat, iar apoi să conectaţi acele module la componentele dumneavoastră folosind legarea SCA:

- SOAP 1.1/JMS şi SOAP 1.1/HTTP folosind JAX-RPC
- SOAP 1.1/HTTP folosind JAX-RPC şi SOAP 1.1/HTTP folosind JAX-WS
- SOAP 1.1/HTTP folosind JAX-RPC şi SOAP 1.2/HTTP folosind JAX-WS

După ce modulul SCA care conţine legarea serviciului web este implementată pe server, aveţi posibilitatea să utilizaţi consola administrativă pentru a vizualiza informaţii legate de legare sau să modificaţi proprietăţile selectate a legăturii.

Notă: Serviciile Web permit aplicaţiilor să interopereze prin utilizarea descrierilor standard ale servicii şi formatelor standard pentru mesajele pe care le schimbă între ei. De exemplu, legările de import şi export ale serviciului web pot interopera cu servicii care sunt implementate folosind WSE (Web Services Enhancements) Versiunea 3.5 şi WCF (Windows Communication Foundation) Versiunea 3.5 pentru Microsoft .NET. Atunci când interacţionaţi cu astfel de servicii, trebuie să vă asiguraţi că:

- Fişierul WSDL care este folosit pentru accesarea unui export de serviciu web include o valoare validă pentru acţiune pentru fiecare operaţie din interfaţă.
- Clientul serviciului web setează fie antetul SOAPAction, fie antetul wsa:Action atunci când trimite un mesaj către exportul serviciului web.

Propagarea antetului SOAP:

În momentul în care manipulaţi mesaje SOAP, este posibil să aveţi nevoie să accesaţi informaţii din anumite anteturi SOAP din mesaje ce sunt primite, ca să vă asiguraţi că mesajele cu anteturi SOAP sunt trimise cu anumite valori sau ca să permiteţi anteturilor SOAP să parcurgă un modul.

Când configuraţi legarea unui serviciu web în Integration Designer, puteţi indica faptul că doriţi ca anteturile SOAP să fie propagate.

- Atunci când cererile sunt primite într-un export sau răspunsurile sunt primite la un import, informaţiile din antetul SOAP pot fi accesate, permiţând ca logica din modul să se bazeze pe valorile antetului şi permiţând acestor anteturi să poată fi modificate.
- Atunci când cererile sunt trimise dintr-un export sau răspunsurile sunt trimise dintr-un import, anteturile SOAP pot fi incluse în aceste mesaje.

Forma şi prezenţa anteturilor SOAP transmise pot fi afectate de seturile de politici configurate la import sau export, aşa cum este explicat în Tabela 26 la pagina 71.

Pentru a configura transmiterea anteturilor SOAP pentru un import sau export, selectaţi (din modul de vizualizare Proprietăţi al Integration Designer) fila **Antet Protocol de Transmitere** şi selectaţi opţiunile de care aveţi nevoie.

Anteturi de Adresare WS

Antetul WS-Addressing poate fi propagat de legarea serviciului web (JAX-WS).

Atunci când transmiteţi antetul de adresare WS, fiţi atenţi la următoarele informaţii:

- Dacă activaţi transmiterea pentru antetul de adresare WS, antetul va fi transmis prin modul în următoarele circumstanţe:
  - Atunci când cererile sunt primite la un export
  - Atunci când răspunsurile sunt primite la un import
- Antetul WS-Addressing nu este transmis în mesajele de ieşire de la IBM Business Process Manager (ceea ce înseamnă că, antetul nu este transmis atunci când cererile sunt trimise de la un import sau atunci când răspunsurile sunt trimise de la un export).

Antetul de securitate WS

Antetul WS-Security poate fi propagat atât prin legarea serviciului web (JAX-WS), cât şi prin legarea (JAX-RPC) a serviciului web.

Specificaţiile SecuritateWS ale serviciilor web descriu îmbunătăţirile aduse mesajelor SOAP pentru a oferi calitatea protecţiei prin integritatea mesajelor, confidenţialitatea mesajelor şi şi autentificarea unică a mesajului. Aceste mecanisme pot fi folosite pentru a acomoda o mare varietate de modele de securitate şi tehnologii de criptare.

Atunci când transmiteţi antetul WS-Security, fiţi atenţi la următoarele informaţii:
- Dacă activaţi transmiterea pentru antetul de securitate WS, antetul va fi transmis prin modul în următoarele circumstanţe:
  - Atunci când cererile sunt primite la un export
  - Atunci când cererile sunt trimise de la un import
  - Atunci când răspunsurile sunt primite la un import
- Implicit, antetul *nu* va fi propagat atunci când răspunsurile sunt trimise de la export. Totuşi, dacă setaţi proprietatea JVM **WSSECURITY.ECHO.ENABLED** cu **true**, antetul va fi transmis atunci când răspunsurile sunt trimise de la export. În acest caz, dacă antetul WS-Security din calea cererii nu este modificat, anteturile WS-Security ar putea fi trimise înapoi sub formă de răspunsuri.
- Forma exactă a mesajului SOAP trimis de la un import pentru o cerere sau dintr-un export pentru un răspuns ar putea să nu se potrivească exact cu mesajul SOAP primit iniţial. Din acest motiv, se presupune că orice semnătură digitală devine nevalidă. În cazul în care este necesară o semnătură digitală în mesajele trimise, aceasta trebuie să fie stabilită prin utilizarea setului de politici de utilizare corespunzător, iar anteturile WS-Security înrudite cu semnătura digitală din mesajele primite ar trebui să fie eliminate din cadrul modulului.

Pentru a propaga antetul WS-Security, trebuie să includeţi schema WS-Security în modulul aplicaţiei. Vedeţi “Includerea schemei WS-Security într-un modul de aplicaţie” la pagina 71 pentru modul de includere al schemei în procedură.

Modul în care anteturile sunt propagate

Modul în care sunt propagate anteturile depinde de politica de securitate setată în legarea de import sau export, aşa cum este arătat în Tabela 26 la pagina 71:

Tabela 54. Modul în care anteturile de securitate sunt transmise

	Legare de import fără politică de securitate	**Legare de import cu politică de securitate**
Legare de import fără politică de securitate	Anteturile de securitate sunt transmise aşa cum sunt prin modul. Ele nu sunt decriptate. Anteturile sunt trimise către ieşire în aceeaşi formă în care au fost primite. Semnătura digitală ar putea deveni nevalidă.	Anteturile de securitate sunt decriptate şi transmise prin modul cu verificarea semnăturii şi autentificare. Anteturile decriptate sunt trimise către ieşire. Semnătura digitală ar putea deveni nevalidă.

Tabela 54. Modul în care anteturile de securitate sunt transmise (continuare)

	Legare de import fără politică de securitate	**Legare de import cu politică de securitate**
Legare de import cu politică de securitate	Anteturile de securitate sunt transmise aşa cum sunt prin modul. Ele nu sunt decriptate. Anteturile nu ar trebui să fie propagate către import. În caz contrar, din cauza dublării apare o eroare.	Anteturile de securitate sunt decriptate şi transmise prin modul cu verificarea semnăturii şi autentificare. Anteturile nu ar trebui să fie propagate către import. În caz contrar, din cauza dublării apare o eroare.

Configuraţi seturile de politici corespunzătoare pentru legările de export şi import, deoarece acestea izolează solicitantul serviciului de modificările aduse configuraţiei sau cerinţelor QoS ale furnizorului de servicii. Anteturile SOAP standard vizibile într-un modul pot fi utilizate pentru a influenţa prelucrarea (de exemplu, înregistrarea în istoric şi urmărirea) în modulul. Propagarea anteturilor SOAP printr-un modul ce aparţin de un mesaj recepţionat către un mesaj trimis înseamnă că beneficiile izolării prin modul sunt reduse.

Anteturilor standard, precum anteturilor WS-Security, nu ar trebui să fie propagate la cererea unui import sau la răspunsul unui export atunci când importul sau exportul au asociate un set de politici care în mod normal rezultă în generarea acestor anteturi. Altfel, va apărea o eroare din cauza duplicării anteturilor. În schimb, anteturile ar trebui să fie eliminate în mod explicit sau legarea de import sau export ar trebui să fie configurat în aşa fel încât să prevină propagarea anteturilor protocolului.

Accesarea anteturilor SOAP

Atunci când este primit un mesaj care conţine anteturi SOAP printr-un import sau export de serviciu web, anteturile sunt puse în secţiunea anteturi a obiectului mesaj al serviciului (SMO - service message object). Puteţi accesa informaţiile din antet, aşa cum este descris în “Accesarea informaţiilor din antetul SOAP în SMOO”.

Includerea schemei WS-Security într-un modul de aplicaţie

Procedura următoare defineşte paşii pentru includerea schemei în modulul aplicaţiei:

- În cazul în care computerul pe care rulează Integration Designer are acces la Internet, efectuaţi următorii paşi:
  1. În perspectiva Business Integration, selectaţi **Dependenţe** pentru proiectul dumneavoastră.
  2. Extindeţi **Resurse predefinite** şi selectaţi fie **Fişiere schemă WS-Security 1.0**, fie **Fişiere schemă WS-Security 1.1** pentru a importa schema în modulul dumneavoastră.
  3. Curăţaţi şi reconstruiţi proiectul.
- În cazul în care un computer pe care rulează Integration Designer nu are acces la Internet, puteţi descărca schema pe un al doilea computer care are acces la Internet. Apoi o puteţi copia pe computerul pe care rulează Integration Designer.
  1. De pe computerul care are acces la Internet, descărcaţi schema de la distanţă:
     a. Faceţi clic pe **Fişier > Import > Business Integration > WSDL şi XSD**.
     b. Selectaţi **WSDL la distanţă** sau **Fişier XSD**.
     c. Importaţi următoarele scheme:

        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. Copiaţi schemele pe un computer care nu are acces la Internet.
  3. De pe computerul care nu are acces la Internet, importaţi schema:
     a. Faceţi clic pe **Fişier > Import > Business Integration > WSDL şi XSD**.
     b. Selectaţi **WSDL Local** sau **Fişier XSD**.

4. Modificaţi locaţiile schemei pentru oasis-wss-wssecurity_secext-1.1.xsd:
   a. Deschideţi schema în *locaţie_spaţiu_de_lucru*/*nume_modul*/StandardImportFilesGen/oasis-wss-wssecurity-secext-1.1.xsd.
   b. Modificaţi:

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

      în:

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

   c. Modificaţi:

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

      în:

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

5. Modificaţi locaţia schemei pentru oasis-200401-wss-wssecurity-secext-1.0.xsd:
   a. Deschideţi schema în *locaţie_spaţiu_de_lucru*/*nume_modul*/StandardImportFilesGen/oasis-200401-wss-wssecurity-secext-1.0.xsd.
   b. Modificaţi:

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

      în:

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

6. Curăţaţi şi reconstruiţi proiectul.

Propagarea antetului Transport:

În momentul în care manipulaţi mesaje SOAP, este posibil să aveţi nevoie să accesaţi informaţii din anumite anteturi de transport din mesaje ce sunt primite, asiguraţi-vă că mesajele cu anteturi de transport sunt trimise cu anumite valori, sau permiteţi anteturilor de transport să parcurgă un modul.

Când configuraţi legarea unui serviciu web în Integration Designer, puteţi indica faptul că doriţi ca anteturile transport să fie propagate.

- Atunci când cererile sunt primite într-un export sau răspunsurile sunt primite la un import, informaţiile din antetul de transport pot fi accesate, permiţând ca logica din modul să se bazeze pe valorile antetului şi permiţând acestor anteturi să poată fi modificate.
- Atunci când răspunsurile sunt trimise dintr-un export sau cererile sunt trimise dintr-un import, anteturile de transport pot fi incluse în aceste mesaje.

Specificarea propagării anteturilor

Pentru a configura propagarea anteturilor de transport pentru un import sau export, realizaţi paşii următori:

1. Din vizualizarea Proprietăţi Integration Designer, selectaţi **Legare > Propagare**.
2. Setaţi opţiunea de propagare a antetului de transport de care aveţi nevoie.

Notă: Propagarea anteturilor de transport este dezactivată în mod implicit şi poate fi implementată doar în mediu de runtime Versiunea 7.0.0.3 (sau mai recentă). De asemenea, reţineţi faptul că, pentru versiunea 7.0.0.3, propagarea antetului de transport se limitează doar la anteturile de transportul HTTP.

Dacă activaţi propagarea anteturilor de transport, anteturile din mesajele primite vor fi propagate printr-un modul, iar dacă nu le eliminaţi în mod explicit aceste vor fi utilizate în invocaţiile ulterioare prin acelaşi fir de execuţie.

Notă: Anteturile de transport nu pot fi propagate atunci când folosiţi o legare pentru serviciul web (JAX-RPC).

Accesarea informaţiilor din antet

Atunci când propagarea antetului de transport este activată pentru mesajele primite, toate anteturile de transport (inclusiv cele definite de client) sunt vizibile în SMO (service message objec). Puteţi seta anteturile cu valori diferite sau puteţi crea altele noi. Cu toate acestea, reţineţi că nu există nicio verificare sau validare a valorilor setate de dvs., şi orice antet necorespunzător sau incorect poate cauza probleme în timpul execuţiei serviciului web.

Luaţi în considerare următoarele informaţii cu privire la setarea anteturilor HTTP:

- Orice modificare adusă anteturilor rezervate pentru motorul de servicii web nu vor fi onorate în mesajul de ieşire. De exemplu, versiunea sau metoda HTTP, anteturile Content-Type, Content-Length and SOAPAction sunt rezervate pentru motorul serviciului web.
- În cazul în care valoarea antetului este un număr, numărul (mai degrabă decât şirul) ar trebui stabilit în mod direct. De exemplu, utilizaţi **Max-Forwards = 5** (în loc de **Max-Forwards = Max-Forwards: 5**) şi **Age = 300** (în loc de **Age = Age: 300**).
- Dacă mesajul de solicitare are o dimensiune mai mică de 32 KB, motorul de servicii web înlătură antetul Transfer-Encoding şi setează în loc antetul Content-Length pentru dimensiune fixată a mesajului.
- Conţinutul de limbă este resetat de WAS.channel.http pe calea de răspuns.
- O setare invalidă pentru Modernizare rezultă într-o eroare 500.
- Anteturile următoare adaugă la sfârşit valorile rezervate de motorul serviciilor web la setările clientului:
  - User-Agent
  - Cache-Control
  - Pragma
  - Accept
  - Conexiune

Puteţi accesa informaţiile din antet într-unul din următoarele moduri:

- Utilizarea unei primitive de mediere pentru accesarea structurilor de tip SMO

  Vedeţi linkurile “Informaţii înrudite” pentru a găsi informaţii despre utilizarea primitivelor de mediere.
- Utilizarea serviciului de context SPI

  Următorul exemplu de cod citeşte anteturile de transport HTTP din serviciul de context:

```
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();
     }
}
```

Depanarea

În cazul în care întâlniţi probleme în momentul în care trimiteţi anteturi revizuite, puteţi intercepta mesajul TCP/IP folosind unelte precum TCP/IP Monitor din Integration Designer. Puteţi accesa TCP/IP Monitor selectând **Rulare/Depanare > TCP/IP Monitor** în pagina Preferinţe.

De asemenea puteţi vedea valorile antetului folosind urma motorului JAX-WS: **org.apache.axis2.*=all: com.ibm.ws.websvcs.*=all:**

Lucrul cu legări de servicii web (JAX-WS):

Atunci când utilizaţi legări de servicii web (JAX-WS) cu aplicaţiile dvs., aveţi posibilitatea să adăugaţi o calitate SAML (Security Assertion Markup Language) a serviciului (QOS) la legare. În primul rând trebuie să utilizaţi consola administrativă pentru a importa setul de politici. De asemenea, puteţi utiliza consola administrativă pentru a vă asigura că serverul este configurat în mod corespunzător pentru utilizarea legării (JAX-WS) a serviciului web.

Importul seturilor de politici:

Security Assertion Markup Language (SAML) este un standard OASIS bazat pe XML pentru schimbarea identităţii şi atributelor de securitate ale informaţiei. Atunci când configuraţi un serviciu web (JAX-WS) legat cu Integration Designer, puteţi specifica un set de politici SAML. Se foloseşte mai întâi consola administrativă IBM Business Process Manager pentru a face seturile de politici SAML disponibile, astfel încât ele să poată fi importate în Integration Designer.

Seturile de politici SAML sunt localizate în mod normal în directorul de configurare al profilului:

rădăcină_profil/config/templates/PolicySets

Înainte de a începe această procedură, se verifică dacă următoarele directoare (ce conţin seturile de politici) sunt localizate în directorul de configurare al profilului:

- 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
- Username WSHTTPS default

Dacă directoarele nu sunt în directorul de configurare al profilului, copiaţi-le în acel director din următoarea locaţie:

rădăcină_app_server/profileTemplates/default/documents/config/templates/PolicySets

Importaţi seturile de politici în consola administrativă, selectaţi una pe care doriţi să o faceţi disponibilă pentru Integration Designer şi apoi salvaţi un fişier .zip pentru fiecare dintre aceste seturi de politici într-o locaţie care este accesibilă prin Integration Designer.

1. Se vor importa seturile de politici prin urmărirea paşilor următori:
   a. Din consola administrativă, se face clic pe **Servicii > Seturi de politici > Seturi de politici aplicaţie**.
   b. Se face clic pe **Import > Din magazia implicită**.
   c. Se selectează seturile de politici implicite SAML şi se face clic pe **OK**.
2. Se vor exporta seturile de politici, astfel încât ele să poată fi utilizate de către Integration Designer:
   a. Din pagina Seturi de politici aplicaţie, se selectează setul de politici SAML care se vrea a fi exportat şi se face clic pe **Export**.

      Notă: Dacă pagina Seturi de politici aplicaţie nu este momentan afişată, se face clic pe **Servicii > Seturi de politici > Seturi de politici aplicaţie** din consola administrativă.
   b. Pe pagina următoare, se face clic pe legătura la fişierului .zip pentru setul de politici.

c. În fereastra Descărcare fişier, se face clic pe **Salvare** şi se indică o locaţie ce este accesibilă prin Integration Designer.
d. Se face clic pe **Înapoi**.
e. Finalizaţi paşii de la 2a la pagina 75 la 2d la pagina 75 pentru fiecare set de politici care se vrea a fi exportat.

Seturile de politici SAML sunt salvate în fişierul .zip şi sunt gata să fie importate în Integration Designer.

Se vor importa seturile de politici în Integration Designer, aşa cum este descris în subiectul “Seturi de politici”.

Invocarea serviciilor web ce necesită autentificare de bază HTTP:

Autentificarea de bază HTTP are un nume şi parolă de utilizator pentru a autentifica un serviciu cleint la un punct final sigur. Puteţi seta autentificare de bază HTTP la trimiterea sau primirea cererilor de serviciu web.

Setaţi autentificarea de bază HTTP pentru recepţionarea cererilor de servicii web prin configurarea Java API pentru XML Web Services (JAX-WS) legături de export, aşa cum este descris în Crearea şi asignarea rolurilor de securitate pentru exportări de servicii web.

Autentificarea de bază HTTP poate fi activată pentru cererile de serviciu web ce sunt trimise de către o legare de import JAX-WS în unul sau două modalităţi:

- La configurarea legării de import într-un modul SCA, puteţi selecta setul politicii de autentificare HTTP furnizat denumit BPMHTTPBasicAuthentication (ce este furnizat prin intermediul legării de import a serviciului web (JAX-WS)) sau orice alt set de politică ce include politica HTTPTransport.
- La construirea modulului SCA, puteţi utiliza capabilităţi de mediere a fluxului pentru a crea dinamic un nou antet de autentificate HTTP şi specifica informaţii cu numele şi parola de utilizator în antet.

Notă: Setul de politică are precedenţă peste valoarea specificată în antet. Dacă doriţi să utilizaţi setul de valori în antetul de autentificare HTTP la momentul rulării, nu potriviţi un set de politică ce include politica HTTPTransport. Specific, nu utilizaţi setul de politică BPMHTTPBasicAuthentication şi dacă aveţi un set de politică definit, asiguraţi-vă că include politica HTTPTransport.

Pentru informaţii suplimentare despre seturile de politici pentru servicii web şi legături de politici şi cum sunt utilizate acestea, vedeţi Seturi de politici de servicii web din WebSphere Application Server Information Center.

- Pentru a utiliza setul de politică furnizat, realizaţi următorii paşi:
  1. Opţional: În consola administrativă, creaţi o legătură de politică generală sau editaţi una existentă care include politica HTTPTransport cu valorile ID-ului de utilizator şi parolei necesare.
  2. În IBM Integration Designer, generaţi o legare de import a serviciului web (JAX-WS) şi ataşaţi setul de politică BPMHTTPBasicAuthentication.
  3. Realizaţi *unul* din următorii paşi:
     – În IBM Integration Designer, în proprietăţile de legare de import al serviciului web (JAX-WS), specificaţi numele unei legări a politicii existente de client generale ce include politica HTTPTransport.
     – După implementarea modulului SCA, utilizaţi consola administrativă fie pentru a selecta o legătură de politică pentru un client existent, fie creaţi o nouă legătură de politică de client şi apoi o asociaţi cu legarea de import.
  4. Opţional: În consola administrativă a serverului Process editaţi legarea setului de politici selectat pentru a specifica ID-ul şi parola necesare.
- Pentru a specifica numele şi parola de utilizator în antetul de autentificare HTTP, realizaţi unul din următorul set de paşi:
  – Utilizaţi medierea simplă HTTP Header Setter în IBM Integration Designer pentru a crea antetul de autentificare HTTP şi specificaţi numele şi parola de utilizator.
  – Dacă o logică suplimentară este necesară, utilizaţi codul Java într-o mediere simplă personalizată (asemenea următorului exemplu) pentru a:

1. Crea un antet de autentificare HTTP.
2. Specifica informaţii privind numele şi parola de utilizator.
3. Adăuga un nou antet de autentificare HTTP la HTTPControl.
4. Seta HTTPControl actualizat înapoi în serviciul Context.

```
//Obţineţi HeaderInfoType din contextService
ContextService contextService = (ContextService) ServiceManager.INSTANCE
.locateService("com/ibm/bpm/context/ContextService");
HeaderInfoType headers = contextService.getHeaderInfo();
if(headers == null){
    headers = ContextObjectFactory.eINSTANCE.createHeaderInfoType();
}
//Get the HTTP header and HTTP Control from HeaderInfoType
HTTPHeaderType httpHeaderType = headers.getHTTPHeader();
HTTPControl cp = httpHeaderType.getControl();
HeadersFactory factory = HeadersFactory.eINSTANCE;
if(cp == null){
    cp = factory.createHTTPControl();
}
//Creaţi HTTPAuthentication nou şi setaţi 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);
// Setaţi informaţii de antet înapoi în contextul curent de execuţie.
contextService.setHeaderInfo(headers);
```

Verificarea configuraţiei serverului:

Atunci când implementaţi o aplicaţie cu legarea unui serviciu web (JAX-WS), trebuie să vă asiguraţi că serverul pe care este implementată aplicaţia nu are selectată opţiunea **Pornire componente după cum este necesar**.

Se poate verifica pentru a vedea dacă această opţiune este selectată prin realizarea următorilor paşi din consola administrativă:

1. Se face clic pe **Servere > Tipuri de servere > Servere de aplicaţii WebSphere**.
2. Se face clic pe nume server.
3. Din fişa Configuraţie, se determină dacă este selectat **Pornire componente după cum este necesar**.
4. Se vor realiza următorii paşi:
   - Dacă **Pornire componente după cum este necesar** este selectat, se va şterge caseta de bifare şi apoi se va face clic pe **Aplicare**.
   - Dacă **Pornire componente după cum este necesar** nu este selectat, se va faceţi clic pe **Anulare**.

Ataşamente în mesajele SOAP:

Aveţi posibilitatea să trimiteţi şi să primiţi mesaje SOAP care includ date binare (precum fişiere PDF sau imagini JPEG) sub formă de ataşamente. Ataşamentele pot fi cu *referinţă* (adică sunt reprezentate în mod explicit ca părţi componente ale mesajului în interfaţa serviciului) sau *fără referinţă* (în care pot fi incluse numere arbitrare şi tipuri de ataşamente).

Un ataşament la care se face referire poate fi reprezentat în unul dintre următoarele moduri:

- Ataşamentele MTOM folosesc codarea specifică SOAP Message Transmission Optimization Mechanism (http://www.w3.org/TR/soap12-mtom/). Ataşamentele MTOM sunt activate prin opţiuni de configurare în legările de import şi de export şi reprezintă modul recomandat de codare a ataşamentelor pentru noi aplicaţii.
- Sub forma unui element de tip wsi:swaRef-typed în schema mesajului

Ataşamentele definite folosind tipul wsi:swaRef conform WS-I (Web Services Interoperability Organization), *Attachments Profile Version 1.0* (http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html), care defineşte modul în care elementele mesajului sunt legate de părţile componente MIME.

- Ca o parte componentă de nivel înalt a mesajului, folosiţi un tip de scheme binare

  Ataşamentele reprezentate sub formă de părţi componente de nivel înalt ale mesajului în concordanţă cu specificaţiile *Mesaje SOAP cu Ataşamente* (http://www.w3.org/TR/SOAP-attachments).

  Ataşamentele reprezentate sub formă de părţi componente de nivel înalt pot fi de asemenea configurate pentru a vă asigura că mesajele şi documentele WSDL produse de legare în concordanţă cu WS-I *Attachments Profile Version 1.0* şi WS-I *Basic Profile Version 1.1* (http://www.ws-i.org/Profiles/BasicProfile-1.1.html).

Un ataşament către care nu se face referinţă este purtat într-un mesaj SOAP fără nici o reprezentare în schema mesajului.

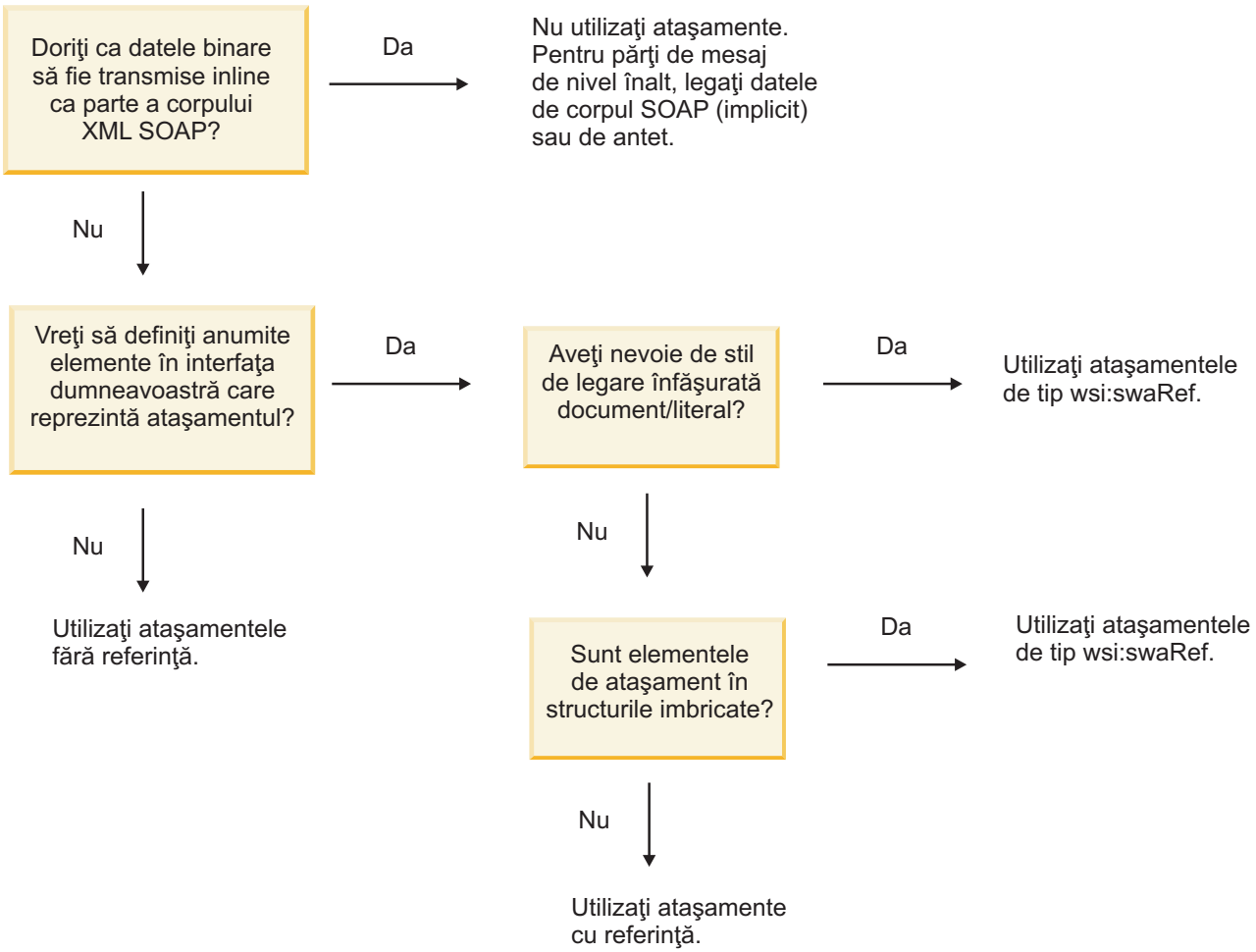
În toate cazurile, ataşamentele MTOM, legarea WDSL SOAP ar trebui să includă legarea MIME pentru ataşamente care vor fi folosite, şi dimensiunea maximă a ataşamentelor nu ar trebui să depăşească 20 MB.

Notă: Pentru a trimite sau a primi mesaje SOAP cu ataşamente, trebuie să folosiţi una dintre legările serviciului web care se bazează pe JAX-WS (Java API for XML Web Services).

Cum alegeţi stilul corespunzător de ataşamente:

Când proiectaţi o interfaţă nouă pentru serviciu care include date binare, luaţi în considerare modul în care datele binare sunt transportate în mesajele SOAP care sunt trimise şi primite de către serviciu.

Message Transmission Optimization Mechanism (MTOM) ar trebui folosit pentru ataşamente dacă aplicaţia serviciului web conectat suportă acest lucru. Dacă nu, diagrama următoare arată modul în care sunt alese alte stiluri de ataşamente. Utilizaţi următorul set de întrebări pentru a determina stilul corespunzător de ataşamente:



Ataşamente MTOM: părţi de mesaj la nivel de top:

Puteţi trimite şi primi mesaje de servicii web care includ ataşamente MTOM (Message Transmission Optimization Mechanism - Mecanism de optimizare a transmisiei mesajului) SOAP. Într-un mesaj SOAP cu mai multe părţi MIME, corpul SOAP este prima parte a mesajului, iar ataşamentul sau ataşamentele sunt în părţi ulterioare.

Trimiţând sau primind un ataşament cu referinţă într-un mesaj SOAP, datele binare care alcătuiesc ataşamentul (care deseori este destul de mare) sunt păstrate separat de corpul mesajului SOAP, pentru a nu fi necesară parsarea lor ca XML. Acest lucru rezultă în mai multe procesări eficiente decât dacă datele binare ar fi fost ţinute într-un element XML.

Iată un eşantion de mesaj MTOM SOAP:

```
... 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>

... aici sunt datele binare ...
--MIMEBoundaryurn_uuid_0FE43E4D025F0BF3DC11582467646812--
```

Reţineţi că în eşantionul MTOM, tipul de conţinut pentru plicul SOAP este **application/xop+xml** şi datele binare sunt înlocuite de un element **xop:Include** ca mai jos:

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

Procesarea la intrare a ataşamentelor cu referinţă

Atunci când un client trece un mesaj SOAP cu un ataşament către o componentă Service Component Architecture (SCA), legarea de export (JAX-WS) a serviciului web înlătură mai întâi ataşamentul. Apoi transmite partea de tip SOAP a mesajului şi creează un obiect business. În sfârşit, legarea setează binarele ataşamentului în obiectul business.

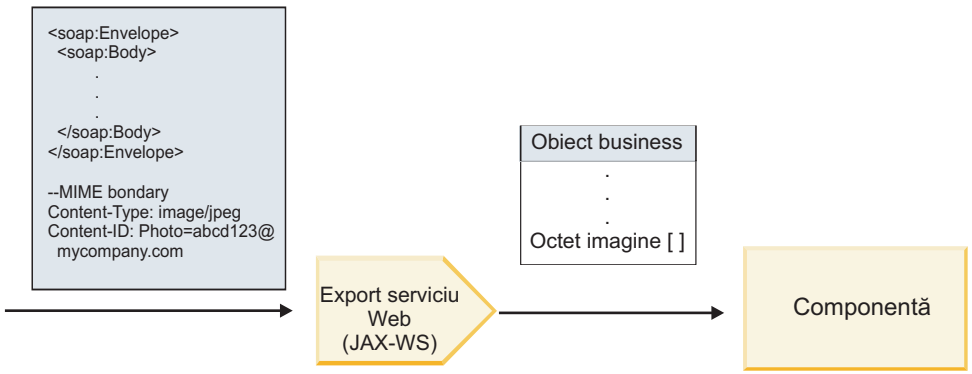


Figura 58. Cum procesează legarea export (JAX-WS) a serviciului web un mesaj SOAP cu un ataşament referit

Atribute ataşament MTOM

- MTOM poate suporta elemente ataşament în structuri imbricate.
- MTOM este disponibil numai pentru tipul base64Binary.
- MTOM poate suporta elemente ataşament în structuri imbricate, ceea ce înseamnă că **bodyPath** pentru ataşamentele MTOM este locaţia **xpath** pentru elementul unde este ţinut ataşamentul MTOM. Logica de calcul pentru **bodyPath** urmează strict schema de generare a locaţiei **xpath** aşa cum se arată în exemplele de mai jos:
  - Pentru un tip non-matrice (**maxOccurs** este 1): /sendImage/input/imageData
  - Pentru un tip matrice (**maxOccurs** > 1): /sendImage/input/imageData[1]
- Tipurile de ataşament amestecate nu sunt suportate, ceea ce înseamnă că, dacă MTOM este activat pe legarea de import, se va genera ataşamentul MTOM. Dacă MTOM este dezactivat sau dacă valoarea de configurare MTOM este lăsată la valoarea implicită pe legarea de export, mesajul MTOM de intrare nu este suportat.

Ataşamente către care se face referinţă: ataşamente de tip swaRef:

Aveţi posibilitatea să trimiteţi şi să primiţi mesaje SOAP care includ ataşamente care sunt reprezentate în interfaţa serviciului sub formă de elemente de tip swaRef.

Un element de tip swaRef este definit în WS-I (Web Services Interoperability Organization) *Attachments Profile* Versiunea 1.0 (http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html), în care este definit modul în care elementele mesajului sunt înrudite cu părţile componente MIME.

În mesajul SOAP, corpul conţine un element de tip swaRef care identifică ID-ul conţinutului ataşamentului.

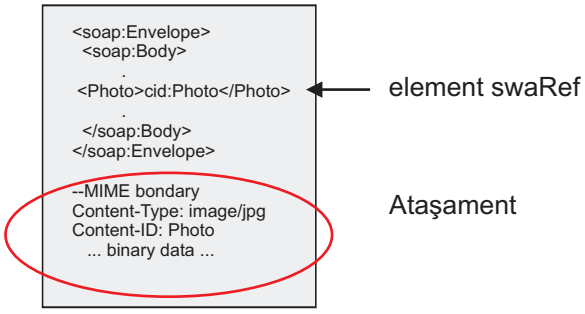


Figura 59. Un mesaj SOAP cu un element swaRef

WSDL-ul pentru acest mesaj SOAP conţine un element de tip swaRef într-o parte componentă a mesajului care identifică ataşamentul.

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

WSDL-ul ar trebui să conţină de asemenea o legare MIME care să indice mesajele MIME compuse din mai multe părţi ce urmează să fie utilizate.

Notă: WSDL *nu* include o legare MIME pentru elementul de mesaj de tip swaRef specific, deoarece legările MIME se aplică doar asupra părţilor componente de nivel înalt ale mesajului.

Ataşamentele reprezentate sub formă de elemente de tip swaRef pot fi transmise doar peste componentele fluxului de mediere. În cazul în care un ataşament trebuie să fie accesat sau transmis către o componentă de un alt tip, folosiţi o componentă a fluxului de mediere pentru a muta ataşamentul la o locaţie care este accesibilă acelei componente.

Procesarea la intrare a ataşamentelor

Folosiţi Integration Designer pentru a configura o legare de export, astfel încât să recepţioneze ataşamentul. Creaţi un modul şi interfaţa şi operaţiile sale asociate, incluzând un element de tip swaRef. Puteţi crea apoi o legare (JAX-WS) pentru serviciul web.

Notă: Vedeţi subiectul “Lucrul cu ataşamente” din Centrul de informare Integration Designer pentru informaţii mai detaliate.

Atunci când un client transmite un mesaj SOAP cu un ataşament swaRef către o componentă SCA (Service Component Architecture), legarea de export (JAX-WS) a serviciului web înlătură mai întâi ataşamentul. Apoi transmite partea de tip SOAP a mesajului şi creează un obiect business. În sfârşit, legarea setează Id-ul conţinutului ataşamentului în obiectul business.

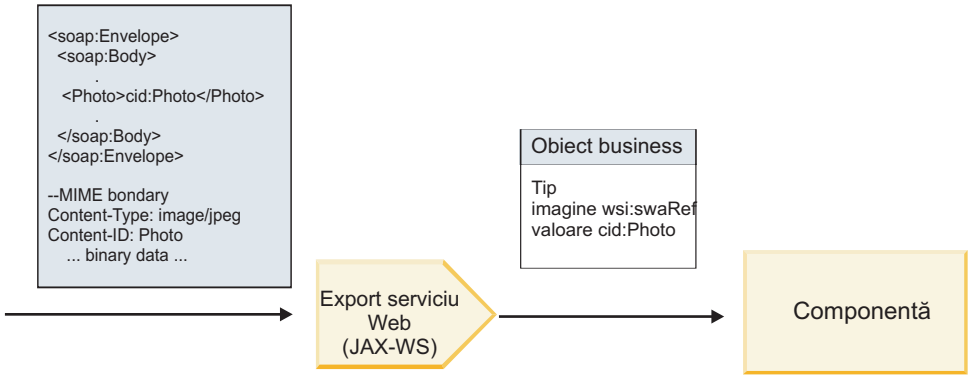


Figura 60. Modul în care o legare externă (JAX-WS) a serviciului web procesează un mesaj SOAP cu un ataşament swaRef

Accesarea metadatelor ataşamentului într-o componentă a fluxului de mediere

Aşa cum se arată în Figura 16 la pagina 81, atunci când ataşamentele de tip swaRef sunt accesate de componente, identificatorul conţinutului ataşamentului apare sub forma unui element de tip swaRef.

Fiecare ataşament al unui mesaj SOAP are, de asemenea, un element **attachments** corespunzător în SMO. Atunci când este folosit tipul swaRef WS-I, elementul **attachments** include tipul conţinutul ataşamentului şi ID-ul conţinutului, precum şi datele binare reale ale ataşamentului.

Pentru a putea obţine valoarea unui ataşament de tip swaRef, este necesară obţinerea valorii elementului de tip swaRef, iar apoi localizarea elementului **attachments** cu valoarea **contentID** corespunzătoare. Reţineţi că valoarea **contentID** are de obicei prefixul **cid:** înlăturat din valoarea swaRef.

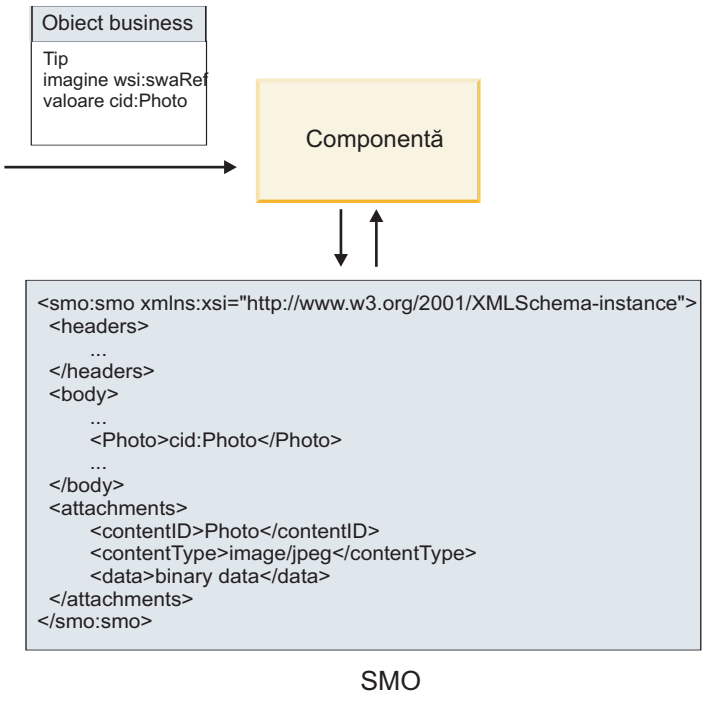


Figura 61. Modul în care ataşamentele swaRef apar în SMO

Procesarea la ieşire

Utilizaţi Integration Designer pentru a configura legarea de import (JAX-WS) a serviciului web pentru a invoca un serviciu web extern. Legarea de import este configurată cu un document WSDL care descrie serviciul web ce trebuie invocat şi defineşte ataşamentul care va fi trimis către serviciul web.

Atunci când legarea unui serviciu web (JAX-WS) primeşte un mesaj SCA, elementele de tip swaRef sunt trimise ca ataşament în cazul în care importul este legat la componenta unui flux de mediere şi elementul de tip swaRef are un element **attachments** corespunzător.

Pentru procesarea la ieşire, elementele de tip swaRef sunt trimise întotdeauna cu valorile lor de ID de conţinut; totuşi, modulul de mediere trebuie să se asigure că există un element **attachments** corespunzător valorii **contentID**.

Notă: Pentru a fi în concordanţă cu WS-I Attachments Profile, valoarea **content ID** ar trebui să fie conformă "codării id-ului de conţinut," aşa cum este descrisă în secţiunea 3.8 din WS-I *Attachments Profile* 1.0.

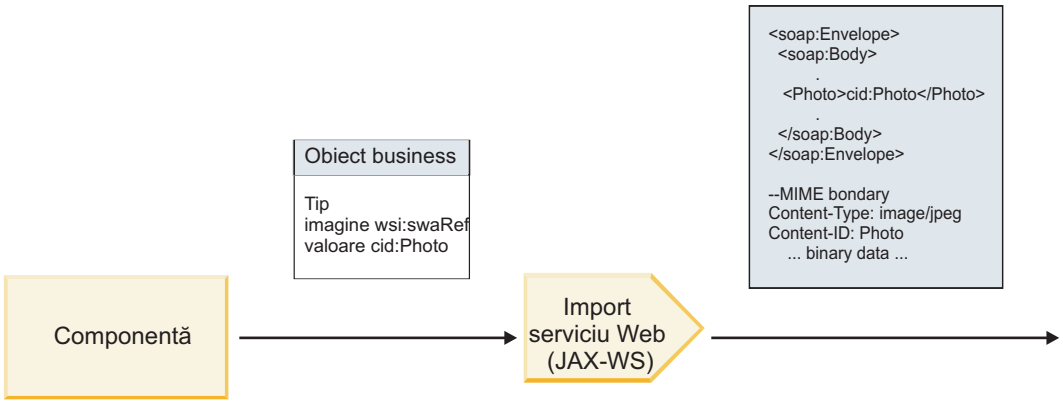


Figura 62. Modul în care o legare de import (JAX-WS) a serviciului web generează un mesaj SOAP cu un ataşament swaRef

Setarea metadatelor ataşamentului într-o componentă a fluxului de mediere

În cazul în care, în SMO există o valoarea pentru un element de tip swaRef şi un element **attachments**, legarea pregăteşte mesajul SOAP (cu ataşament) şi îl trimite către un destinatar.

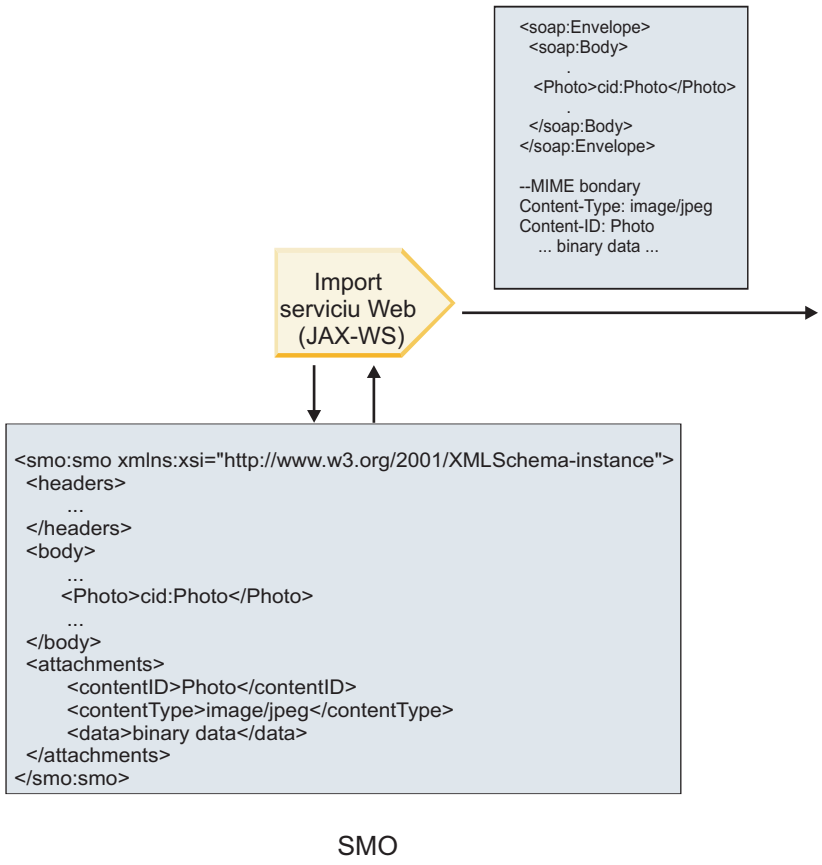


Figura 63. Modul în care un ataşament de tip swaRef din SMO este accesat pentru a crea un mesaj de tip SOAP

Elementul **attachments** este prezent în SMO doar dacă o componentă a fluxului de mediere este conectată direct la import sau la export; aceasta nu a trecut prin alte tipuri de componente. În cazul în care valorile sunt necesare într-un modul care conţine alte tipuri de componente, ar trebui să fie utilizată o componentă a fluxului de mediere pentru a copia valorile într-o locaţie în care pot fi accesate din modul, şi o altă componentă a fluxului de mediere utilizată pentru a seta valorile corecte înainte de o invocare de ieşire prin intermediul unui serviciu web de import.

Important: Aşa cum este descris în “Reprezentare XML a SMO,” primitiva de mediere Mapping transformă mesajele utilizând o transformare XSLT 1.0. Transformarea operează asupra unei serializări XML a SMO. Primitiva de mediere Mapping permite specificarea rădăcinii de serializare şi elementul rădăcină a documentului XML reflectă această rădăcină.

Atunci când trimiteţi mesaje SOAP cu ataşamente, elementul rădăcină pe care îl alegeţi determină modul în care ataşamentele sunt propagate.

- Dacă folosiţi “/body” drept rădăcină a mapării XML, toate ataşamentele sunt propagate implicit de-a lungul mapării.
- Dacă folosiţi “/” drept rădăcină pentru mapare, puteţi controla propagarea ataşamentelor.

Ataşamente la care se face referire: părţi componente de nivel înalt ale mesajului:

Aveţi posibilitatea să trimiteţi şi să primiţi mesaje SOAP care includ ataşamente binare ce sunt declarate ca părţi componente în interfaţa serviciului dumneavoastră.

Într-un mesaj SOAP cu mai multe părţi MIME, corpul SOAP este prima parte a mesajului, iar ataşamentul sau ataşamentele se află în părţile componente următoare.

Care este avantajul de a trimite sau de a primi un ataşament cu referinţă într-un mesaj SOAP? Datele binare care alcătuiesc ataşamentul (care este adesea destul de mare), sunt ţinute separat de corpul mesajului SOAP, astfel încât acesta să nu trebuiască să fie parsat ca XML. Acest lucru rezultă în mai multe procesări eficiente decât dacă datele binare ar fi fost ţinute într-un element XML.

Tipuri de mesaje SOAP cu ataşamente către care se face referinţă

Începând cu versiunea 7.0.0.3 a IBM Business Process Manager, aveţi posibilitatea de a alege modul în care mesajul SOAP este generat:

- **Mesaje conforme cu WS-I**

  Runtime-ul poate genera mesaje SOAP care sunt conforme cu WS-I *Attachments Profile Version 1.0* şi cu WS-I *Basic Profile Version 1.1*. Într-un mesaj SOAP, care este în conformitate cu aceste profiluri, doar o singură parte este legată de corpul SOAP; pentru cele care sunt legate ca ataşamente, se utilizează codarea părţii content-id (aşa cum este descrisă în WS-I *Attachments Profile Version 1.0*) pentru a asocia ataşamentul la partea componentă a mesajului.
- **Mesaje care nu sunt conforme cu WS-I**

  Runtime-ul poate genera mesaje SOAP care sunt conforme cu profilurile WS-I, dar care sunt compatible cu mesajele generate în Versiunea 7.0 sau 7.0.0.2 a IBM Business Process Manager. Mesajele SOAP folosesc elemente de nivel înalt numite după partea mesajului care are un atribut **href** care reţine ataşamentul **content-id**, dar codarea părţii content-id nu este folosită (aşa cum este descris în WS-I *Attachments Profile Version 1.0*).

Selectarea compatibilităţii WS-I pentru exporturile de servicii web

Folosiţi Integration Designer pentru a configura o legare de export. Creaţi un modul şi interfaţa şi operaţiile sale asociate. Puteţi crea apoi o legare (JAX-WS) pentru serviciul web. Pagina Ataşamente cu referinţă afişează toate părţile componente binare care aparţin de operaţiile create, apoi selectaţi care părţi vor fi ataşate. Apoi specificaţi, în pagina Specificare compatibilitatea WS-I AP 1.0 pentru Integration Designer, una dintre următoarele alegeri:

- **Utilizare WS-I AP 1.0 compatibil cu mesajul SOAP**

  Dacă selectaţi această opţiune, specificaţi de asemenea şi care parte a mesajului ar trebui să fie legată de corpul SOAP.

  Notă: Această opţiune poate fi utilizată doar atunci când fişierul WSDL corespunzător este de asemenea conforme cu WS-I .

  Un fişier WSDL care este generat de Integration Designer Versiunea 7.0.0.3 este în conformitate cu WS-I. Totuşi, dacă importaţi un fişier WSDL care nu este în conformitate cu WS-I, nu puteţi selecta această opţiune.
- **Utilizare mesaje SOAP care nu sunt conforme cu WS-I AP 1.0**

  Dacă selectaţi această opţiune, care este implicită, prima parte a mesajului este legată de corpul SOAP.

Notă: Doar părţile de nival înalt ale mesajului (adică, elementele definite în portType WSDL drept părţi componente din cadrul mesajului de intrare sau ieşire) care au un tip binar (fie base64Binary, fie hexBinary) pot fi trimise sau primite sub formă de ataşamente cu referinţă.
Vedeţi subiectul “Lucrul cu ataşamente” din Centrul de informare Integration Designer pentru detalii suplimentare.

Pentru mesajele conforme WS-I, content-ID-ul care este generat în mesajul SOAP este o concatenare a următoarelor elemente:

- Valoarea atributului **nume** din elementul **wsdl:part** către care se face referinţă prin **mime:content**
- Caracterul **=**
- O valoare unică globală, precum un UUID
- Caracterul **@**
- Un nume de domeniu valid

Procesarea la intrare a ataşamentelor cu referinţă

Atunci când un client transmite un mesaj SOAP cu un ataşament către o componentă SCA (Service Component Architecture), legarea de export (JAX-WS) a serviciului web înlătură mai întâi ataşamentul. Apoi transmite partea de tip SOAP a mesajului şi creează un obiect business. În sfârşit, legarea setează binarele ataşamentului în obiectul business.

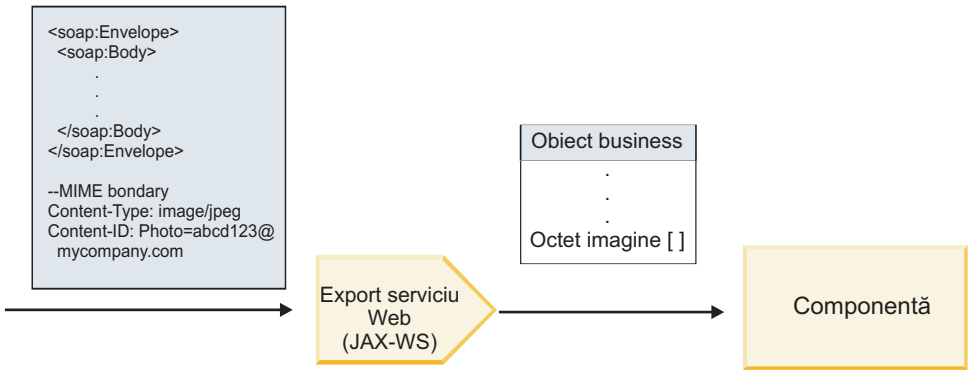


Figura 64. Modul în care o legare externă (JAX-WS) a serviciului web procesează un mesaj SOAP conform WS-I cu un ataşament către care se face o referinţă

Accesarea metadatelor ataşamentului într-o componentă a fluxului de mediere

Aşa cum se arată în Figura 19 la pagina 85, atunci când ataşamentele cu referinţă sunt accesate de componente, datele ataşamentului apar sub forma unei matrice cu date de tip octet.

Fiecare ataşament cu referinţă al unui mesaj SOAP are, de asemenea, un element **attachments** corespunzător în SMO. Elementul **attachments** include tipul conţinutul ataşamentului şi calea către corpul mesajului în care este păstrat ataşamentul.

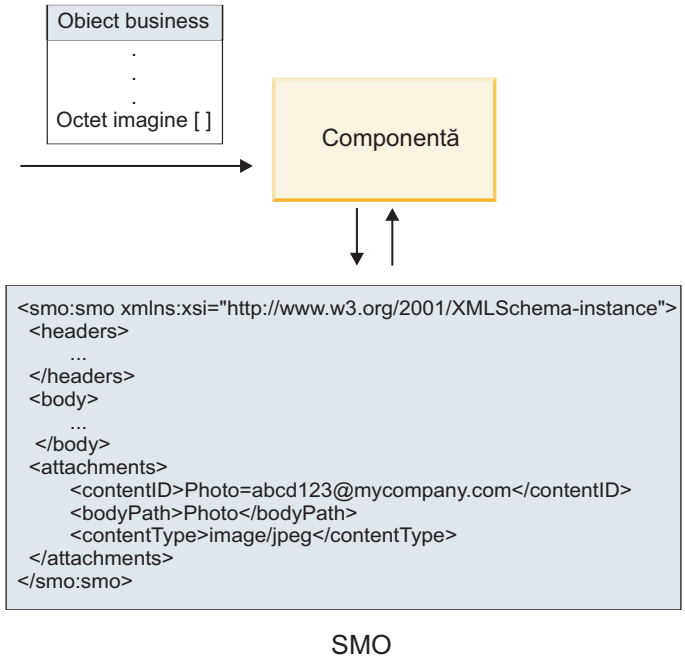


Figura 65. Modul în care ataşamentele către care se face referinţă apar în SMO

Important: Calea către elementul corp al mesajului nu este actualizată automat în cazul în care mesajul este transformat şi ataşamentul mutat. Puteţi utiliza fluxul de mediere pentru a actualiza elementul **attachments** cu noua cale (de exemplu, ca parte a transformării sau a utilizării unui setter separat pentru elementul mesaj).

Modul în care sunt construite mesajele SOAP de ieşire

Utilizaţi Integration Designer pentru a configura legarea de import (JAX-WS) a serviciului web pentru a invoca un serviciu web extern. Legarea de import este configurată cu un document WSDL care descrie serviciul web ce trebuie invocat şi defineşte ce părţi din mesaj ar trebui să fie trimise ca ataşamente. De asemenea, puteţi indicata, în pagina Specificare compatibilităţii cu WS-I AP 1.0 a Integration Designer, una dintre următoarele alegeri:

- **Utilizare WS-I AP 1.0 compatibil cu mesajul SOAP**

  Dacă selectaţi această opţiune, specificaţi de asemenea şi care parte a mesajului ar trebui să fie legată de corpul SOAP; toate celelalte sunt legate de ataşamente sau anteturi. Mesajele trimise de legare nu includ elemente în corpul SOAP care se referă la echipamente; relaţia este exprimată prin intermediul ID-ului de conţinut al ataşamentului, inclusiv numele părţii mesajului.
- **Utilizare mesaje SOAP care nu sunt conforme cu WS-I AP 1.0**

  Dacă selectaţi această opţiune, care este implicită, prima parte a mesajului este legată de corpul SOAP; toate celelalte sunt legate de ataşamente sau anteturi. Mesajele trimise de legare nu includ unul sau mai multe elemente în corpul SOAP care se referă la ataşamente prin intermediul unui atribut **href**.

Notă: Partea care reprezintă un ataşament, aşa cum este definit în WSDL, trebuie să fie de un tip simplu (fie base64Binary, fie hexBinary). În cazul în care o parte este definită de un complexType, acea parte nu poate fi legată ca un ataşament.

Procesarea la intrare a ataşamentelor cu referinţă

Legarea de import foloseşte informaţiile din SMO pentru a determina modul în care părţile componente de nivel înalt ale mesajului sunt trimise ca ataşamente.

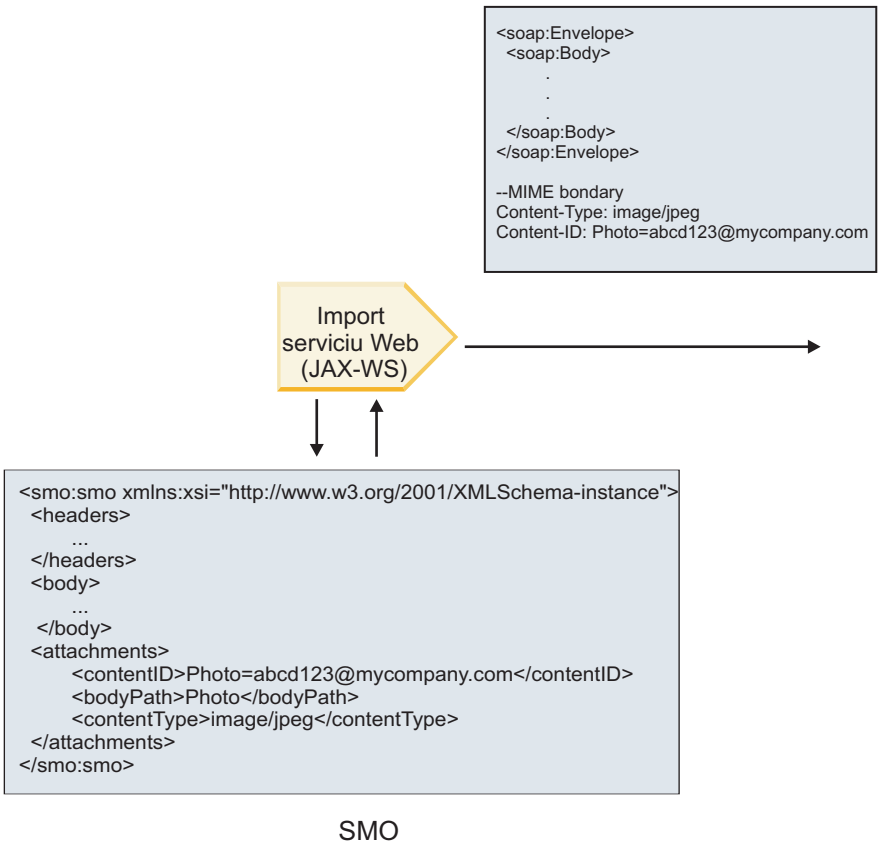


Figura 66. Modul în care ataşamentul cu referinţă din SMO este accesat pentru a crea mesajul SOAP

Elementul **attachments** este prezent în SMO doar dacă o componentă a fluxului de mediere este conectată direct la import sau la export; aceasta nu a trecut prin alte tipuri de componente. În cazul în care valorile sunt necesare într-un modul care conţine alte tipuri de componente, ar trebui să fie utilizată o componentă a fluxului de mediere pentru a copia valorile într-o locaţie în care pot fi accesate din modul, şi o altă componentă a fluxului de mediere utilizată pentru a seta valorile corecte înainte de o invocare de ieşire prin intermediul unui serviciu web de import.

Legarea foloseşte o combinaţie între următoarele condiţii pentru a determina modul în care (sau dacă) este trimis mesajul:

- În cazul în care există o legare MIME WSDL pentru partea binară de nivel înalt a mesajului, iar dacă există, modul în care este definit tipul conţinutului
- În cazul în care există un element **attachments** în SMO a cărui valoare **bodyPath** face referinţă către o parte binară de nivel înalt

Modul în care sunt create ataşamentele atunci când există un element attachment în SMO

Tabelul următor arată modul în care este creat şi trimis un ataşament în cazul în care SMO conţine un element **atachment** cu un **bodyPath** care se potriveşte cu o parte a numelui mesajului:

Tabela 55. Modul în care este generat ataşamentul

Starea legării WSDL MIME pentru partea componentă binară de nivel înalt a mesajului	**Modul în care mesajul este creat şi transmis**
Prezent cu una dintre următoarele: • Nu este definit un tip de conţinut pentru partea componentă a mesajului • Sunt definite mai multe tipuri de conţinut • Este definit tipul conţinutului metacaracterului	Partea componentă a mesajului este trimisă ca un ataşament. **Content-Id** este setat cu o valoare în elementul attachments, în cazul în care acesta este prezent; altfel, se generează una. **Content-Type** este setat cu o valoare în elementul attachments în cazul în care acesta este prezent; altfel, este setat cu **application/octet-stream**.
Prezent cu conţinut unic, fără metacaractere pentru partea componentă a mesajului	Partea componentă a mesajului este trimisă ca un ataşament. **Content-Id** este setat cu o valoare în elementul attachments, în cazul în care acesta este prezent; altfel, se generează una. **Content-Type** este setat cu o valoare în elementul attachments, în cazul în care acesta este prezent; altfel, este setat cu tipul definit în elementul de conţinut WSDL MIME.
Nu este prezent	Partea componentă a mesajului este trimisă ca un ataşament. **Content-Id** este setat cu o valoare în elementul attachments, în cazul în care acesta este prezent; altfel, se generează una. **Content-Type** este setat cu o valoare în elementul attachments în cazul în care acesta este prezent; altfel, este setat cu **application/octet-stream**. **Notă:** Trimiţând părţi componente ale sub formă de ataşamente atunci când nu sunt definite ca atare în WSDL poate întrerupe compatibilitatea cu WS-I Attachments Profile 1.0, iar acest lucru ar trebui evitat pe cât posibil.

Modul în care sunt create ataşamentele atunci când nu există nici un element attachment în SMO

Tabelul următor arată modul în care este creat şi trimis un ataşament în cazul în care SMO nu conţine un element **atachment** cu un **bodyPath** care se potriveşte cu o parte din numele mesajului:

Tabela 56. Modul în care este generat ataşamentul

Starea legării WSDL MIME pentru partea componentă binară de nivel înalt a mesajului	**Modul în care mesajul este creat şi transmis**
Prezent cu una dintre următoarele: • Nu este definit un tip de conţinut pentru partea componentă a mesajului • Sunt definite mai multe tipuri de conţinut • Este definit tipul conţinutului metacaracterului	Partea componentă a mesajului este trimisă ca un ataşament. **Content-Id** este generat. **Content-Type** este setat cu **application/octet-stream**.
Prezent cu conţinut unic, fără metacaractere pentru partea componentă a mesajului	Partea componentă a mesajului este trimisă ca un ataşament. **Content-Id** este generat. **Content-Type** este setat cu tipul definit în elementul conţinut WSDL MIME.
Nu este prezent	Partea componentă a mesajului nu este trimisă ca un ataşament.

Important: Aşa cum este descris în “Reprezentare XML a SMO,” primitiva de mediere Mapping transformă mesajele utilizând o transformare XSLT 1.0. Transformarea operează asupra unei serializări XML a SMO. Primitiva de mediere Mapping permite specificarea rădăcinii de serializare şi elementul rădăcină a documentului XML reflectă această rădăcină.

Atunci când trimiteţi mesaje SOAP cu ataşamente, elementul rădăcină pe care îl alegeţi determină modul în care ataşamentele sunt propagate.

- Dacă folosiţi “/body” drept rădăcină a mapării XML, toate ataşamentele sunt propagate implicit de-a lungul mapării.
- Dacă folosiţi “/” drept rădăcină pentru mapare, puteţi controla propagarea ataşamentelor.

Ataşamente fără referinţă:

Aveţi posibilitatea să trimiteţi şi să primiţi ataşamente *fără referinţă* care nu sunt declarate în interfaţa serviciului.

Într-un mesaj SOAP cu mai multe părţi MIME, corpul SOAP este prima parte a mesajului, iar ataşamentele se află în părţile următoare. Nu este inclusă nici o referinţă către ataşament în corpul SOAP.

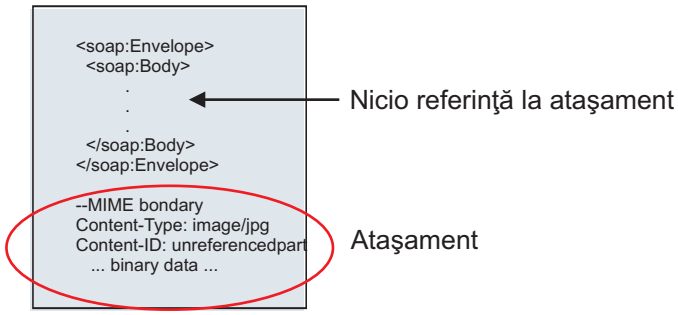


Figura 67. Un mesaj SOAP cu un ataşament către care nu se face referinţă

Aveţi posibilitatea să trimiteţi un mesaj SOAP cu un ataşament fără referinţă printr-un export de serviciu web către importul unui serviciu web. Mesajul de ieşire, care este trimis la serviciul web ţintă, conţine ataşamentul.

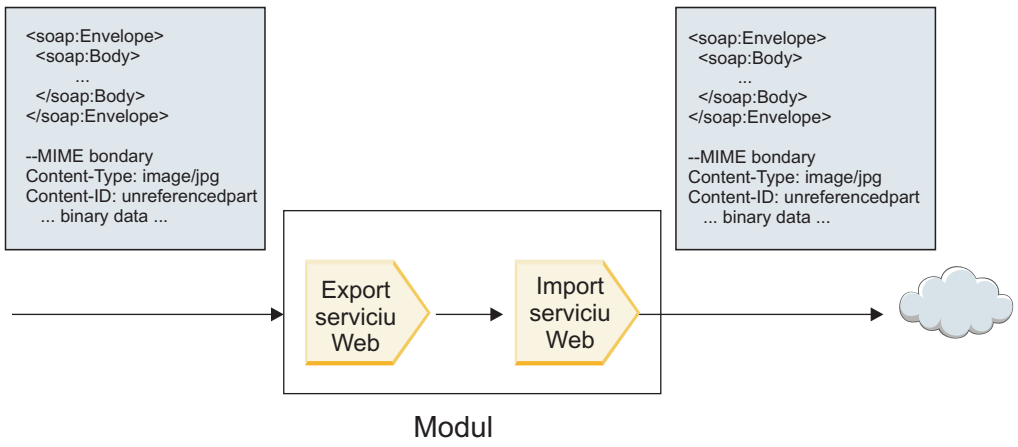


Figura 68. Un ataşament care trece printr-un modul SCA

În Figura 23 la pagina 89, mesajul SOAP, cu ataşament, este transmis fără modificări.

De asemenea, puteţi modifica mesajul SOAP prin utilizarea unei componente de tip flux de mediere. De exemplu, aveţi posibilitatea să utilizaţi componenta de tip flux de mediere pentru a extrage datele din mesajul SOAP (în acest caz, date binare din corpul mesajului) şi pentru a crea un mesaj SOAP cu ataşamente. Datele sunt prelucrate ca parte componentă

a elementului ce conţine ataşamentele dintr-un SMO (service message object).

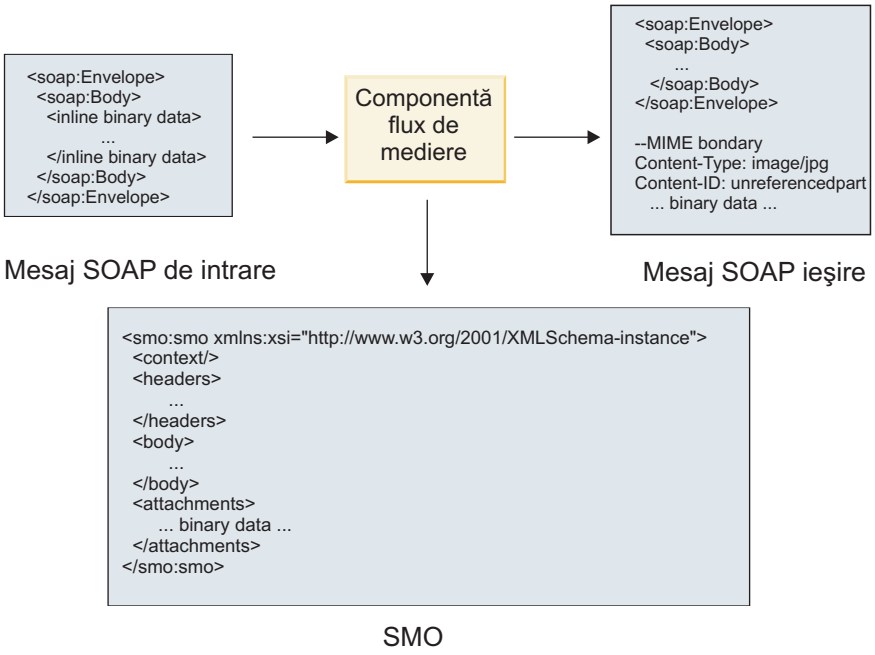


Figura 69. Un mesaj procesat de o componentă a fluxului de mediere

În schimb, componenta de fluxul de mediere se poate transforma mesajul de intrare prin extragerea şi codificarea ataşamentului, iar apoi transmite mesajul fără nici un ataşament.

În loc de extragerea datelor dintr-un mesaj SOAP de intrare pentru a forma un mesaj SOAP cu ataşamente, puteţi obţine datele ataşamentului la o sursă externă,⌂cum ar fi o bază de date.

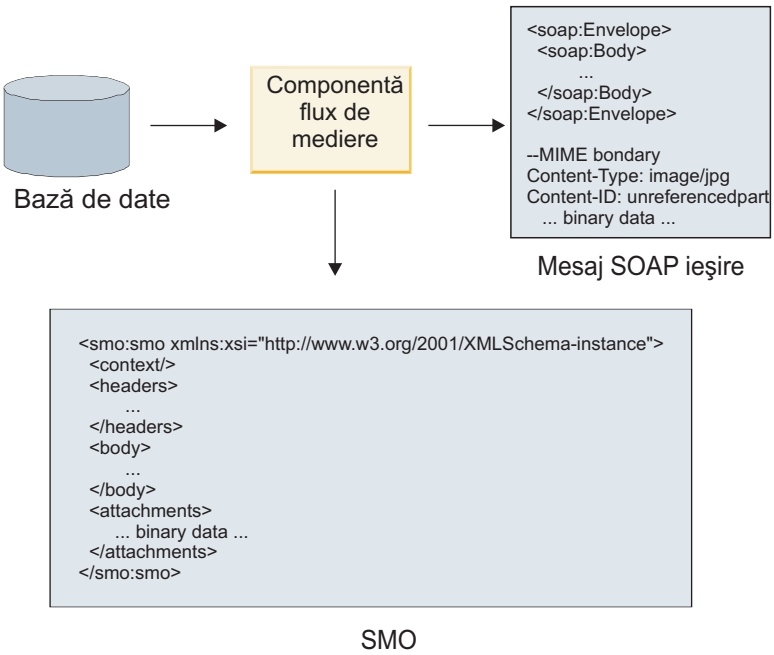


Figura 70. Un ataşament obţinut dintr-o bază de date şi adăugat la mesajul SOAP

În schimb, componenta fluxului de mediere poate extrage ataşamentul dintr-un mesaj SOAP de intrare şi poate procesa mesajul (de exemplu, păstrează ataşamentul într-o bază de date).

Ataşamentele fără referinţă pot fi transmise doar peste componentele fluxului de mediere. În cazul în care un ataşament trebuie să fie accesat sau transmis către o componentă de un alt tip, folosiţi o componentă a fluxului de mediere pentru a muta ataşamentul la o locaţie care este accesibilă acelei componente.

Important: Aşa cum este descris în "Reprezentare XML a SMO," primitivă de mediere Mapping transformă mesajele utilizând o transformare XSLT 1.0. Transformarea operează asupra unei serializări XML a SMO. Primitiva de mediere Mapping permite specificarea rădăcinii de serializare şi elementul rădăcină a documentului XML reflectă această rădăcină.

Atunci când trimiteţi mesaje SOAP cu ataşamente, elementul rădăcină pe care îl alegeţi determină modul în care ataşamentele sunt propagate.

- Dacă folosiţi "/body" drept rădăcină a mapării XML, toate ataşamentele sunt propagate implicit de-a lungul mapării.
- Dacă folosiţi "/" drept rădăcină pentru mapare, puteţi controla propagarea ataşamentelor.

Utilizarea legării stilului documentului WSDL cu mesaje multiple:

Organizaţia WS-I (Web Services Interoperability Organization) a definit un set de reguli legate de modul în care serviciile web ar trebui să fi descrise prin intermediul unui WSDL şi modul în care mesajele SOAP corespunzătoare ar trebui să fie formate, în scopul asigurării interoperabilităţii.

Aceste reguli sunt specificate în WS-I *Basic Profile Version 1.1* (http://www.ws-i.org/Profiles/BasicProfile-1.1.html). În special, WS-I Basic Profile 1.1 R2712 declară: "O legare literală de document TREBUIE să fie serializată ca PLIC cu un soap:Body al cărui element copil este o instanţă a declaraţiei elementului global referită de partea componentă wsdl:message corespunzătoare."

Aceasta înseamnă că, atunci când utilizaţi o legare SOAP de stil document pentru o operaţie cu mesaje (intrare, ieşire sau greşeală) care sunt definite cu părţi componente multiple, doar una dintre acele părţi ar trebui să fie legată de corpul SOAP pentru a fi compatibilă cu WS-I Basic Profile 1.1.

Suplimentar, WS-I Attachments Profile 1.0 R2941 declară: "O wsdl:binding dintr-o DESCRIERE AR TREBUI să lege fiecare wsdl:part a unui wsdl:message din wsdl:portType la care se referă la unul dintre soapbind:body, soapbind:header, soapbind:fault, soapbind:headerfault sau mime:content.".

Aceasta înseamnă că, atunci când utilizaţi o legare SOAP de stil document pentru o operaţie cu mesaje (intrare, ieşire sau greşeală) care sunt definite cu părţi componente multiple, toate părţile componente în afară de cea selectată care vor fi legate la corpul SOAP trebuie să fie legate ca ataşamente sau anteturi.

Următoarea abordare este utilizată atunci când descrierile WSDL sunt generate pentru exporturi cu legări de servicii web (JAX-WS şi JAX-RPC) în acest caz:

- Puteţi alege care parte a mesajului să fie legată la corpul SOAP dacă există mai multe elemente de tip non-binar. Dacă există un singur element de tip non-binar, acel element este legat automat la corpul SOAP.
- Pentru legarea JAX-WS, toate celelalte părţi componente ale mesajului de tip "hexBinary" sau "base64Binary" sunt legate ca ataşamente referite. Vedeţi "Ataşamente la care se face referire: părţi componente de nivel înalt ale mesajului" la pagina 83.
- Toate celelalte părţi componente ale mesajului sunt legate ca anteturi SOAP.

Legările de import JAX-RPC şi JAX-WS respectă legarea SOAP într-un document WSDL existent cu mesaje compuse de tip document chiar dacă nu leagă mai multe părţi la corpul SOAP; totuşi, nu sunteţi în măsură să generaţi clienţi pentru serviciile web pentru astfel de documente WSDL în Rational Application Developer.

Notă: Legarea JAX-RPC nu suportă ataşamente.

Tiparul recomandat când utilizaţi mesaje multiple cu o operaţie care are legare SOAP de stil document este prin urmare:

1. Utilizaţi stilul înfăşurat document/literal. În acest caz, mesajele au mereu o sngură parte componentă; totuşi, ataşamentele trebuie să fie nereferite (după cum este descris în “Ataşamente fără referinţă” la pagina 89) sau swaRef-typed (după cum este descris în “Ataşamente către care se face referinţă: ataşamente de tip swaRef” la pagina 79) în acest caz.
2. Utilizaţi stilul RPC/literal. În acest caz, nu există restricţii asupra legării WSDL în ceea ce priveşte numărul de părţi componente legate de corpul SOAP; mesajul SOAP care rezultă are întotdeauna un singur copil care reprezintă operaţia care este invocată, cu părţile componente ale mesajului fiind copiii acelui element.
3. Pentru legarea JAX-WS, trebuie să aveţi cel mult o parte componentă mesaj care nu este de tip "hexBinary" sau "base64Binary", doar dacă nu este acceptabil să legaţi celelalte părţi componente non-binare la anteturi SOAP.
4. Orice alte cazuri sunt supuse comportamentului descris.

Notă: Există restricţii suplimentare când utilizaţi mesaje SOAP care nu sunt compatibile cu WS-I *Basic Profile Version 1.1*.

- Prima parte componentă a mesajului ar trebui să fie non-binară.
- Când recepţionaţi mesaje SOAP stil document multiple cu ataşamente referite, legarea JAX-WS aşteaptă ca fiecare ataşament referit să fie reprezentat de un element copil al corpului SOAP cu o valoare atribut href care identifică ataşamentul după ID-ul conţinutului său. Legarea JAX-WS trimite ataşamente referite pentru astfel de mesaje în acelaşi mod. Acest comportament nu este compatibil cu WS-I Basic Profile.

  Pentru a vă asigura că mesajele dumneavoastră sunt compatibile cu Basic Profile, urmaţi abordarea 1 la pagina 92 sau 2 la pagina 92 din lista anterioară sau evitaţi utilizarea ataşamentelor referite pentru astfel de mesaje şi utilizaţi în schimb ataşamente nereferite sau de tip swaRef.

Legări HTTP

Legarea HTTP este proiectată să furnizeze conectivitate SCA (Service Component) la HTTP. În consecinţă, aplicaţii HTTP existente sau nou-dezvoltate pot participa în medii SOA (Service Oriented Architecture).

HTTP (Hypertext Transfer Protocol) este un protocol utilizat pe scară largă pentru transferul de informaţiilor pe web. Când lucraţi cu o aplicaţie externă ce foloseşte protocolul HTTP, este necesară o legare HTTP. Legarea HTTP transformă datele transmise ca un mesaj într-un format nativ unui obiect operaţional într-o aplicaţie SCA. Legarea HTTP poate de asemenea transforma datele transmise în afară ca un obiect operaţional în formatul nativ aşteptat de aplicaţia externă.

Notă: Dacă doriţi să interacţionaţi cu clienţii şi serviciile care utilizează protocolul SOAP/HTTP pentru serviciile web, luaţi în considerare utilizarea uneia dintre legările serviciilor web, care oferă funcţionalităţi suplimentare cu privire la manipulare standardului calităţii serviciilor pentru serviciile web.

Câteva scenarii comune pentru utilizarea legării HTTP sunt descrise în următoarea listă:

- Serviciile găzduite de SCA pot invoca aplicaţii HTTP folosind un import HTTP.
- Serviciile găzduite de SCA se pot expune ca aplicaţii cu HTTP activat, pentru a putea fi folosite de clienţi HTTP, folosind un export HTTP.
- IBM Business Process Manager şi Process Server pot comunica între ei peste o infrastructură HTTP, în consecinţă utilizatorii îşi pot gestiona comunicările conform standardelor de corporaţie.
- IBM Business Process Manager şi Process Server pot acţiona ca mediatori de comunicaţii HTTP, transformând şi direcţionând mesaje, ceea ce îmbunătăţeşte integrarea aplicaţiilor folosind o reţea HTTP.
- IBM Business Process Manager şi Process Server pot fi folosite ca o punte între HTTP şi alte protocoale, precum servicii Web SOAP/HTTP, adaptoare de resurse bazate pe JCA (Java Connector Architecture), JMS, şi aşa mai departe.

Informaţii detaliate despre crearea legărilor de import şi export HTTP pot fi găsite în centrul de informare Integration Designer. Vedeţi subiectele **Dezvoltarea aplicaţiilor de integrare > Accesarea serviciilor externe cu HTTP>**.

Privire generală asupra legărilor HTTP:

Legarea HTTP oferă conectivitate la aplicaţii găzduite-HTTP. Mediază comunicarea dintre aplicaţii HTTP şi permite aplicaţiilor bazate pe HTTP existente să fie apelate dintr-un modul.

Legări de import HTTP

Legarea de import HTTP oferă conectivitate de ieşire de la aplicaţii SCA (Service Component Architecture) la un server sau aplicaţii HTTP.

Importul invocă un URL de punct final HTTP. URL-ul poate fi specificat în unul din trei moduri:

- URL-ul poate fi setat dinamic în anteturile HTTP prin URL-ul de înlocuire dinamic.
- URL-ul poate fi setat dinamic în elementul de adresă ţintă SMO.
- URL-ul poate fi specificat ca o proprietate de configurare pe import.

Această invocare este întotdeauna sincronă în natură.

Deşi invocările HTTP sunt întotdeauna cerere-răspuns, importul HTTP suportă şi operaţii cu sens unic şi cu sens dublu şi ignoră răspunsul în cazul unei operaţii cu sens unic.

Legări de export HTTP

Legarea de export HTTP oferă conectivitate de intrare de la aplicaţii HTTP la o aplicaţie SCA.

Un URL este definit pe exportul HTTP. Aplicaţiile HTTP ce vor să trimită mesaje cerere exportului folosesc acest URL pentru a invoca exportul.

Exportul HTTP de asemenea suportă ping-uri.

Legări HTTP la momentul rulării

Un import cu o legare HTTP în momentul rulării trimite o cerere cu sau fără date în corpul mesajului din aplicaţia SCA către serviciul web extern. Cererea se face din aplicaţia SCA către serviciul web extern, după cum se arată în Figura 26 la pagina 93.

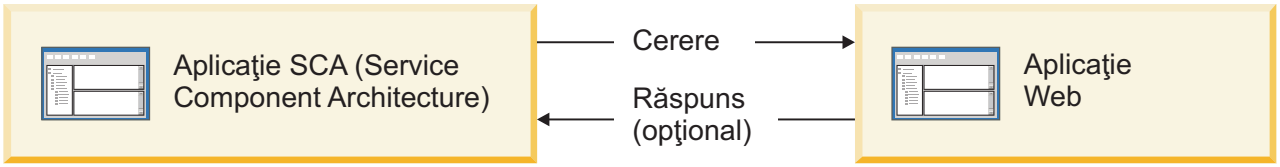


Figura 71. Fluxul unei cereri de la aplicaţia SCA către aplicaţia web

Op⌂ional, importul cu legarea HTTP poate primi date înapoi de la aplicaţia web într-un răspuns la cerere.

Cu un export, cererea se face de către o aplicaţie client către un serviciu web, după cum se arată în Figura 27 la pagina 93.

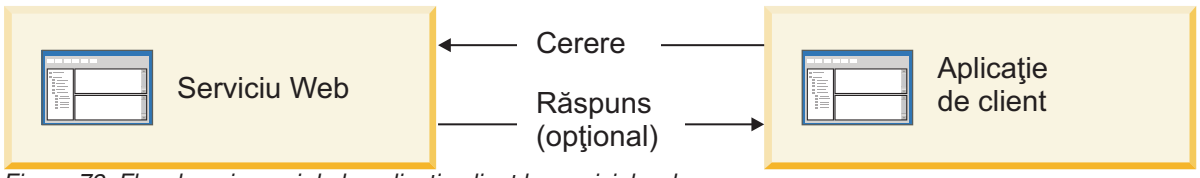


Figura 72. Fluxul unei cereri de la aplicaţia client la serviciul web.

Serviciul web este o aplicaţie web care rulează pe server. Exportul este implementat în aplicaţia web ca un servlet, astfel încât clientul trimite cererea către o adresă URL. Servletul transmite cererea aplicaţiei SCA în timpul rulării.

Opţional, exportul poate trimite date aplicaţiei client ca răspuns la cerere.

Anteturi HTTP:

Legările de import şi export HTTP permit ca configurarea anteturilor HTTP şi a valorilor lor să fie folosită pentru mesaje de ieşire. Importul HTTP foloseşte aceste anteturi pentru cereri şi exportul HTTP le foloseşte pentru răspunsuri.

Anteturile configurate static şi informaţiile de control prevalează asupra valorilor setate dinamic în timpul rulări. Totuşi, valorile de control ale URL-ului de înlocuire dinamic, Versiunii şi Metodei înlocuiesc valorile statice, ce sunt altfel considerate implicite.

Legarea suportă natura dinamică a URL-ului importului HTTP prin determinarea valorii URL-ului ţintă HTTP, Versiunii şi Metodei la momentul rulării. Aceste valori sunt determinate prin extragerea valorii Referinţei de punct final, URL-ului de înlocuire dinamic, Versiunii şi Metodei.

- Pentru Referinţa punctului final, folosiţi API-uri com.ibm.websphere.sca.addressing.EndpointReference APIs sau setaţi câmpul /headers/SMOHeader/Target/address în antetul SMO.
- Pentru URL de înlocuire dinamic, Versiune şi Metodă, folosiţi secţiunea parametrilor de control HTTP a mesajului SCA (Service Component Architecture). Reţineţi că URL-ul de înlocuire dinamic are precedenţă în faţa Referinţei de punct final ţintă; totuşi, Referinţa punctului final se aplică peste legări, deci este abordarea preferată şi ar trebui folosită unde este posibil.

Informaţiile de control şi antet pentru mesaje de ieşire sub legările de export şi import HTTP sunt procesate în următoarea ordine:

1. Informaţii de antet şi control excluzând URL de înlocuire dinamic HTTP, Versiune şi Metodă din Mesajul SCA (cea mai scăzută prioritate)
2. Modificări de la consola administrativă la nivelul de export/import
3. Modificări de la consola administrativă la nivelul metodei al exportului sau importului
4. Adresa ţintă specificată prin intermediul Referinţei punctului final sau antetului SMO
5. URL de înlocuire dinamic, Versiune şi Metodă din mesajul SCA
6. Informaţii de antet şi control de la handler-ul de date sau legarea de date (cea mai înaltă prioritate)

Exportul şi importul HTTP vor popula anteturile de direcţie de intrare şi parametri de control cu date din mesajul de intrare (HTTPExportRequest şi HTTPImportResponse) doar dacă propagarea antetului de protocol este setată la **Adevărat**. Invers, exportul şi importul HTTP vor citi şi procesa anteturile de ieşire şi parametri de controls (HTTPExportResponse şi HTTPImportRequest) doar dacă propagarea antetului de protocol este setată la **Adevărat**.

Notă: Modificările handler-ului de date sau legării de date asupra anteturilor sau parametrilor de control din răspunsul de import sau cererea de export nu vor altera instrucţiunile de procesare ale mesajului din interiorul legării de import sau export şi ar trebui folosite doar pentru propagarea valorilor modificate către componentele SCA din aval.

Serviciul de context este responsabil pentru propagarea contextului (inclusiv anteturile de protocol, precum antetul HTTP şi contextul de utilizator, precum ID cont) de-a lungul unei căi de invocare SCA. În timpul dezvoltării în IBM Integration Designer, puteţi controla propagarea contextului prin proprietăţile de import şi export. Pentru detalii suplimentare, vedeţi informaţiile legărilor de import şi export din centrul de informare al IBM Integration Designer.

Structuri furnizate de anteturi HTTP şi suport

Tabela 29 la pagina 95 detaliază parametri de cerere/răspuns pentru cereri şi răspunsuri Import HTTP şi Export HTTP.

Tabela 57. Informaţii de antet HTTP furnizate

Nume control	Cerere Import HTTP	Răspuns Import HTTP	Cerere Export HTTP	Răspuns Export HTTP
URL	Ignorat	Nesetat	Citit din mesajul de cerere. **Notă:** Şirul de interogare face de asemenea parte din parametrul de control URL.	Ignorat
Versiune (valori posibile: 1.0, 1.1; cea implicită este 1.1)	Ignorat	Nesetat	Citit din mesajul de cerere	Ignorat
Metodă	Ignorat	Nesetat	Citit din mesajul de cerere	Ignorat
URL de înlocuire dinamic	Dacă este setat în handler-ul de date sau legarea de date, înlocuieşte URL-ul de import HTTP. Scris în mesaj în linia de cerere. **Notă:** Şirul de interogare face de asemenea parte din parametrul de control URL.	Nesetat	Nesetat	Ignorat
Versiune de înlocuire dinamică	Dacă este setat, înlocuieşte Versiunea de import HTTP. Scris în mesaj în linia de cerere.	Nesetat	Nesetat	Ignorat
Metodă de înlocuire dinamică	Dacă este setat, înlocuieşte Metoda de import HTTP. Scris în mesaj în linia de cerere.	Nesetat	Nesetat	Ignorat
Tip media (Acest parametru de control transmite parte din valoarea antetului HTTP Tip-conţinut.)	Dacă este prezent, este scris în mesaj ca parte din antetul Tip-conţinut. **Notă:** Această valoare a elementului de control ar trebui furnizată de handler-ul de date sau legarea de date.	Citit din mesajul răspuns, antetul Tip-conţinut	Citit din mesajul cerere, antetul Tip-conţinut	Dacă este prezent, scris în mesaj ca parte din antetul Tip-conţinut. **Notă:** Această valoare a elementului de control ar trebui furnizată de handler-ul de date sau legarea de date.
Set de caractere (implicit: UTF-8)	Dacă este prezent, scris în mesaj ca parte din antetul Tip-conţinut. **Notă:** Această valoare a elementului de control ar trebui furnizată de legarea de date.	Citit din mesajul răspuns, antetul Tip-conţinut	Citit din mesajul cerere, antetul Tip-conţinut	Suportat; scris în mesaj ca parte din antetul Tip-conţinut. **Notă:** Această valoare a elementului de control ar trebui furnizată de legarea de date.

Tabela 57. Informaţii de antet HTTP furnizate (continuare)

Nume control	Cerere Import HTTP	Răspuns Import HTTP	Cerere Export HTTP	Răspuns Export HTTP
Codare de transfer (Valori posibile: chunked, identity; implicit este identity)	Dacă este prezent, scris în mesaj ca un antet şi controlează cum este codată transformarea mesajului.	Citit din mesajul răspuns	Citit din mesajul de cerere	Dacă este prezent, scris în mesaj ca un antet şi controlează cum este codată transformarea mesajului.
Codare de conţinut (Valori posibile: gzip, x-gzip, deflate, identity; implicit este identity)	Dacă este prezent, scris în mesaj ca un antet şi controlează cum sunt codate datele utile (payload).	Citit din mesajul răspuns	Citit din mesajul de cerere	Dacă este prezent, scris în mesaj ca un antet şi controlează cum sunt codate datele utile (payload).
Lungime-Conţinut	Ignorat	Citit din mesajul răspuns	Citit din mesajul de cerere	Ignorat
StatusCode (implicit: 200)	Nesuportat	Citit din mesajul răspuns	Nesuportat	Dacă este prezent, scris în mesaj în linia de răspuns
ReasonPhrase (implicit: OK)	Nesuportat	Citit din mesajul răspuns	Nesuportat	Valoare de control ignorată. Valoarea liniei răspunsului mesajului este generată din StatusCode.
Autentificare (conţine proprietăţi multiple)	Dacă este prezent, folosit pentru a construi antetul Autentificare de bază. **Notă:** Valoarea pentru acest antet va fi codată doar pe protocolul HTTP. În SCA, va fi decodată şi transmisă ca text în clar.	Nu se aplică	Citit din antetul Autentificare de bază al mesajului răspuns. Prezenţa acestui antet nu indică faptul că utilizatorul a fost autentificat. Autentificarea ar trebui controlată în configurarea servletului. **Notă:** Valoarea pentru acest antet va fi codată doar pe protocolul HTTP. În SCA, va fi decodată şi transmisă ca text în clar.	Nu se aplică
Proxy (conţine proprietăţi multiple: Gazdă, Port, Autentificare)	Dacă este prezent, este folosit pentru a stabili conexiunea prin proxy.	Nu se aplică	Nu se aplică	Nu se aplică
SSL (conţine proprietăţi multiple: Keystore, Keystore Password, Trustore, Trustore Password, ClientAuth)	Dacă este populat şi url-ul destinaţie este HTTPS, este folosit pentru a stabili o conexiune prin SSL.	Nu se aplică	Nu se aplică	Nu se aplică

Legări de date HTTP:

Pentru fiecare mapare de date între un mesaj SCA (Service Component Architecture) şi un mesaj de protocol HTTP, trebuie configurate un handler de date sau o legare de date HTTP. Handler-ele de date oferă o interfaţă neutră-legării ce permite reutilizarea peste legări de transport şi reprezintă abordarea recomandată; legările de date sunt specifice unei anumite legări de transport. Sunt furnizate clase de legări de date specifice-HTTP; puteţi de asemenea scrie handler-e de date sau legări de date personalizate.

Notă: Cele trei clase de legări de date HTTP descrise în acest subiect (HTTPStreamDataBindingSOAP, HTTPStreamDataBindingXML şi HTTPServiceGatewayDataBinding) sunt depreciate începând cu IBM Business Process Manager Versiunea 7.0. În loc de a folosi legările de date descrise în acest subiect, consideraţi următoarele handler-e de date:

- Folosiţi SOAPDataHandler în loc de HTTPStreamDataBindingSOAP.
- Folosiţi UTF8XMLDataHandler în loc de HTTPStreamDataBindingXML
- Folosiţi GatewayTextDataHandler în loc de HTTPServiceGatewayDataBinding

Legările de date sunt furnizate pentru utilizare cu importuri HTTP şi exporturi HTTP: legare de date binare, legare de date XML şi legare de date SOAP. O legare de date răspuns nu este necesară pentru operaţii cu sens unic. O legare de date este reprezentată de numele unei clase Java ale cărei instanţe pot converti atât de la HTTP la ServiceDataObject şi vice-versa. Un selector de funcţii trebuie folosit pe un export care, în conjuncţie cu legări de metode, poate determina ce legare de date este folosită şi ce operaţie este invocată. Legările de date livrate sunt:

- Legări de date binare, ce tratează corpul ca date binare nestructurate. Implementarea schemei XSD a legării de date binare este după cum urmează:

```
<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>
```

- Legările de date XML, care suportă corpul ca date XML. Implementarea legării de date XML este similară cu legarea de date XML JMS şi nu are restricţii pe schema interfeţei.
- Legări de date SOAP, ce suportă corpul ca date SOAP. Implementarea legării de date SOAP nu are restricţii pe schema interfeţei.

Implementarea legărilor de date HTTP personalizate

Această secţiune descrie cum se implementează o legare de date HTTP personalizată.

Notă: Abordarea recomandată este să implementaţi un handler de date personalizat deoarece poate fi reutilizat peste legări de transport.

HTTPStreamDataBinding este principala interfaţă pentru tratarea mesajelor HTTP personalizate. Interfaţa este proiectată să permită tratarea datelor utile mari. Totuşi, pentru ca astfel de implementări să funcţioneze, această legare de date trebuie să returneze informaţiile de control şi anteturile înainte de a scrie mesajul în flux.

Metodele şi ordinea lor de execuţie, listate mai jos, trebuie implementate de legarea de date personalizată.

Pentru a personaliza o legare de date, scrieţi o clasă ce implementează HTTPStreamDataBinding. Legarea de date ar trebui să aibă patru proprietăţi private:

- private DataObject pDataObject
- private HTTPControl pCtrl
- private HTTPHeaders pHeaders

- private yourNativeDataType nativeData

Legarea HTTP va invoca legarea de date personalizată în următoarea ordine:

- Procesare de ieşire (DataObject în format Nativ):
  1. setDataObject(...)
  2. setHeaders(...)
  3. setControlParameters(...)
  4. setBusinessException(...)
  5. convertToNativeData()
  6. getControlParameters()
  7. getHeaders()
  8. write(...)
- Procesare de intrare (format Nativ în DataObject):
  1. setControlParameters(...)
  2. setHeaders(...)
  3. convertFromNativeData(...)
  4. isBusinessException()
  5. getDataObject()
  6. getControlParameters()
  7. getHeaders()

Trebuie să invocaţi setDataObject(...) în convertFromNativeData(...) pentru a seta valoarea lui dataObject, ce este convertit din date native în proprietatea privată "pDataObject".

```
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;
}
/*
* Adăugaţi antetul http "IsBusinessException" în pHeaders.
* Doi paşi:
* 1.Înlăturaţi toate anteturile cu numele IsBusinessException (nesensibil la majuscule) întâi.
*   Aceasta este pentru a vă asigura că doar un singur antet este prezent.
* 2.Adăugaţi noul antet "IsBusinessException"
*/
public void setBusinessException(boolean isBusinessException) {
  //înlăturaţi toate anteturile cu numele IsBusinessException (nesensibil la majuscule) întâi.
  //Aceasta este pentru a vă asigura că doar un singur antet este prezent.
  //adăugaţi noul antet "IsBusinessException", exemplu de cod:
  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;
}
/*
* Obţineţi antetul "IsBusinessException" din pHeaders, returnaţi-i valoarea booleană
*/
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){
  //Metodă dezvoltată de client pentru a
  //Citi date din HTTPInputStream
  //Convertirea lor la DataObject
  DataObject dataobject=realConvertWorkFromNativeDataToSDO(arg0);
  setDataObject(dataobject);
}
public void write(HTTPOutputStream output) throws IOException {
  if (nativeData != null)
  output.write(nativeData);
}
```

Legări EJB

Legările de import EJB (Enterprise JavaBeans) permit componentelor SCA (Service Component Architecture) să invoce serviciile furnizate de o logică operaţională Java EE ce rulează pe un server Java EE. Legările de export EJB permit componentelor SCA să fie afişate sub formă de Enterprise JavaBeans, astfel încât logica operaţională Java EE să poată invoca componentele SCA, care altfel sunt indisponibile pentru ele.

Legările de import EJB:

Legările de import EJB permit unui modul SCA să apeleze implementările EJB prin specificarea modului în care modulul de consum este legat de EJB-ul extern. Importul serviciilor dintr-o implementare EJB externă permite utilizatorilor să îşi conecteze logica operaţională în mediul IBM Business Process Manager şi să participe într-un proces operaţional.

Folosiţi Integration Designer pentru a crea legări de import EJB. Puteţi utiliza oricare dintre următoarele proceduri pentru a genera legările:

- Creare import EJB folosind vrăjitorul de servicii externe

  Puteţi folosi vrăjitorul pentru servicii externe în Integration Designer pentru a construi un import EJB bazat pe implementarea existentă. Vrăjitorul pentru servicii externe creează servicii pe criteriile furnizate de dumneavoastră. Apoi generează obiecte business, interfeţe şi importă fişiere în baza serviciilor descoperite.
- Creare legări de import EJB folosind editorul de asamblare

  Puteţi să creaţi un import EJB într-o diagramă de asamblare utilizând editorul de asamblare Integration Designer. Din paletă, puteţi folosi fie un Import, fie o clasă Java pentru a crea o legare EJB.

Importul generat are legări de date care fac conexiunea Java-WSDL în loc de a cere o componentă punte Java. Aveţi posibilitatea să legaţi în mod direct o componentă cu o referinţă WSDL (Web Services Description Language) la importul EJB care comunică cu un serviciu bazat pe EJB folosind o interfaţa Java.

Importul EJB poate interacţiona cu logica operaţională Java EE folosind fie modelul de programare EJB 2.1, fie modelul de programare EJB 3.0.

Invocarea logicii operaţionale Java EE poate fi locală (doar pentru EJB 3.0) sau de la distanţă.

- Invocarea locală este utilizată atunci când doriţi să apelaţi logica operaţională Java EE care se află pe acelaşi server ca şi importul.

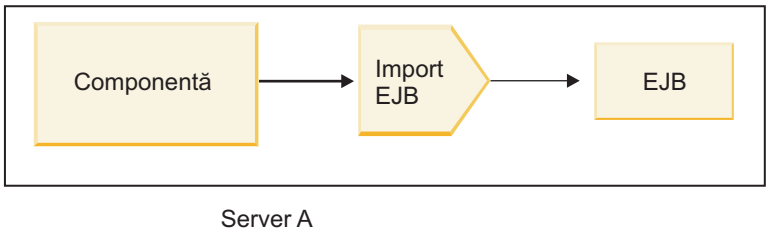


Figura 73. Invocarea locală a unui EJB (doar EJB 3.0)

- Invocarea de la distanţă este utilizată atunci când doriţi să apelaţi logica operaţională Java EE care nu se află pe acelaşi server ca şi importul.

  De exemplu, în figura de mai jos, un import EJB utilizează RMI/IIOP (Remote Method Invocation over Internet InterORB Protocol) pentru a invoca o metodă EJB de pe un alt server.

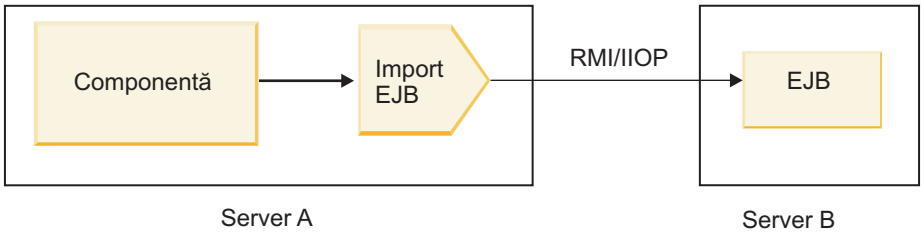


Figura 74. Invocare de la distanţă a unui EJB

Atunci când configurează legarea EJB, Integration Designer foloseşte numele JNDI pentru a determina nivelul modelului de programare EJB şi tipul de invocare (local sau de la distanţă).

Legările de export EJB conţin următoarele componente majore:
- Handler date JAX-WS
- Selector de defectare EJB
- Selector pentru funcţia de import EJB

Dacă scenariul dvs. de utilizator nu se bazează pe maparea JAX-WS, s-ar putea să aveţi nevoie de un handler de date personalizat, un selectorul de funcţie şi de un selectorul de defect pentru a realiza taskurile care altfel ar fi fost finalizate de către componentele care fac parte din legările de import EJB. Acest lucru include maparea care ar fi finalizată în mod normal de către algoritmul personalizat de mapare.

Legările de export EJB:

Aplicaţiile Java EE externe pot invoca o componentă SCA prin intermediul unei legări de export EJB. Folosind un export EJB puteţi expune componente SCA, astfel încât aplicaţiile Java EE externe pot invoca aceste componente folosind modelul de programare EJB.

Notă: Exportul EJB este un bean stateless.

Folosiţi Integration Designer pentru a crea legări EJB. Puteţi utiliza oricare dintre următoarele proceduri pentru a genera legările:
- Crearea legărilor de export EJB folosind vrăjitorul pentru servicii externe

Puteţi folosi vrăjitorul pentru servicii externe în Integration Designer pentru a construi un serviciu de export EJB bazat pe implementarea existentă. Vrăjitorul pentru servicii externe creează servicii pe criteriile furnizate de dumneavoastră. Apoi acesta generează obiecte business, interfeţe şi exportă fişiere pe baza serviciilor descoperite.

- Crearea legărilor de export EJB folosind editorul de asamblare

  Puteţi să creaţi un export EJB folosind editorul de asamblare Integration Designer.

Important: Un client Java 2 Platform, Standard Edition (J2SE) nu poate invoca clientul de export EJB care este generat în Integration Designer.

Puteţi genera legarea dintr-o componentă SCA existentă, sau puteţi genera un export cu o legare EJB pentru o interfaţă Java.

- Atunci când generaţi un export dintr-o componentă SCA existentă care are o interfaţă WSDL existentă, exportului i se alocă o interfaţă Java.
- Atunci când generaţi un export dintr-o interfaţă Java, puteţi selecta fie un WSDL, fie o interfaţă Java pentru export.

  Notă: O interfaţă Java folosită pentru a crea un export EJB are următoarele limitări în ceea ce priveşte obiectele (parametrii de ieşire şi intrare şi excepţii) transmise ca parametrii la un apel de la distanţă:
  - Trebuie să fie de un tip concret (în loc de un tip interfaţă sau abstract).
  - Acestea trebuie să fie în concordanţă cu specificaţia Enterprise JavaBeans. Trebuie să fie serializabili şi să aibă un constructor implicit fără argumente, iar toate proprietăţile trebuie să fie accesibile prin intermediul metodelor getter si setter.

    Consultaţi site-ul web Sun Microsystems, Inc., http://java.sun.com pentru informaţii legate de specificaţiile JavaBeans Enterprise.

  În plus, excepţia trebuie să fie o excepţie verificată, moştenită din java.lang.Exception, şi trebuie să fie unică (adică, nu suportă să arunce mai multe tipuri de excepţii verificate).

  De asemenea, reţineţi că interfaţa afacerii pentru un EnterpriseBean Java este o interfaţă Java simplă şi nu trebuie să extindă javax.ejb.EJBObject sau javax.ejb.EJBLocalObject. Metodele interfeţei de afaceri nu ar trebui să arunce java.rmi.Remote.Exception.

Legările de export EJB pot interacţiona cu logica operaţională Java EE fie folosind modelul de programare EJB 2.1, fie modelul de programare EJB 3.0.

Invocarea poate fi locală (doar pentru EJB 3.0) sau de la distanţă.

- Invocarea locală este utilizată atunci când logica operaţională Java EE apelează o componentă SCA care se află pe acelaşi server ca şi exportul.
- Invocarea de la distanţă este utilizată atunci când logica operaţională the Java EE business logic nu se află pe acelaşi server ca şi exportul.

  De exemplu, în figura următoare, un EJB foloseşte RMI/IIOP pentru a apela o componentă SCA aflată pe un server diferit.

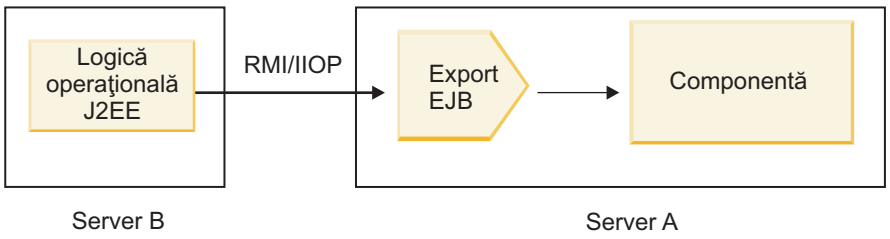


Figura 75. Un apel la distanţă de la un client către o componentă SCA prin intermediul unui export EJB

Atunci când configurează legarea EJB, Integration Designer foloseşte numele JNDI pentru a determina nivelul modelului de programare EJB şi tipul de invocare (local sau de la distanţă).

Legările de export EJB conţin următoarele componente majore:

- Handler date JAX-WS
- Selector pentru funcţia de export EJB

Dacă scenariul dvs. de utilizator nu se bazează pe maparea JAX-WS, s-ar putea să aveţi nevoie de un handler de date personalizat şi de un selectorul de funcţie pentru a realiza taskurile care altfel ar fi finalizate de către componentele care fac parte din legările export EJB. Acest lucru include maparea care ar fi finalizată în mod normal de către algoritmul personalizat de mapare.

Proprietăţile legărilor EJB:

Legările pentru import EJB îşi folosesc numele JNDI configurate pentru a determina nivelul modelului de programare EJB şi tipul de invocare (local sau de la distanţă). Legările de import şi export EJB utilizează handler-ul de date JAX-WS pentru transformarea datelor. Legarea de import EJB foloseşte un selector pentru funcţia de import EJB şi un selector pentru defectul EJB, iar legarea de export EJB foloseşte un selector pentru funcţia de export EJB.

Numele JNDI şi legările de import EJB:

Atunci când configurează legarea EJB pentru un import, Integration Designer foloseşte numele JNDI pentru a determina nivelul modelului de programare EJB şi tipul de invocare (local sau de la distanţă).

În cazul în care nu este specificat nici un nume JNDI, se foloseşte legarea interfeţei EJB implicite. Numele implicite care sunt create depind de faptul că invocaţi EJB 2.1 JavaBeans sau EJB 3.0 JavaBeans.

Notă: Consultaţi subiectul "Privire generală asupra legărilor de aplicaţie EJB 3.0" din Centrul de informare WebSphere Application Server pentru informaţii suplimentare detaliate despre convenţiile de numire.

- EJB 2.1 JavaBeans

  Numele JNDI implicit preselectat de către Integration Designer este legarea EJB 2.1 implicită, care ia formularul **ejb/** plus interfaţa home, separate prin linii oblice.

  De exemplu, pentru interfaţa home a EJB 2.1 JavaBeans pentru com.mycompany.myremotebusinesshome, legarea implicită este:

  ```
  ejb/com/mycompany/myremotebusinesshome
  ```

  Pentr EJB 2.1, este suportată doar invocarea EJB de la distanţă.

- EJB 3.0 JavaBeans

  Numele JNDI implicit preselectat de Integration Designer pentru JNDI-ul local este numele clasei complet calificate al interfeţei locale precedate de **ejblocal:**. De exemplu, pentru interfaţa complet calificată a interfeţei com.mycompany.mylocalbusiness locale, EJB-ul 3.0 JNDI preselectat este:

  ```
  ejblocal:com.mycompany.mylocalbusiness
  ```

  Pentru interfaţa com.mycompany.myremotebusiness de la distanţă, EJB-ul 3.0 JNDI preselectat este interfaţa complet calificată:

  ```
  com.mycompany.myremotebusiness
  ```

  Legările aplicaţiei implicite EJB 3.0 sunt descrise la următoarea locaţie: Privire generală asupra legărilor de aplicaţie EJB 3.0.

  Integration Designer va folosi numele "scurt" drept locaţia JNDI implicită pentru EJB-uri folosind versiunea 3.0 a modelului de programare.

  Notă: În cazul în care referinţa JNDI implementată a ţintei EJB este diferită de locaţia legării JNDI implicite deoarece a fost utilizată sau configurată o mapare personalizată, numele JNDI-ului ţintă trebuie să fie specificat în

mod corespunzător. Aveţi posibilitatea să specificaţi numele în Integration Designer înainte de implementare, sau , pentru legarea de import, puteţi modifica numele în consola administrativă (după implementare) pentru a potrivi numele JNDI pentru EJB ţintă.

Pentru informaţii suplimentare despre crearea legărilor EJB, vedeţi secţiunea dedicată pentru Lucrul cu legări EJB din Centrul de informare Integration Designer.

Handler date JAX-WS:

Legarea EJB (Enterprise JavaBeans) de import utilizează handler-ul de date JAX-WS pentru a transforma obiecte business cerere în parametri de obiect Java şi pentru a transforma valoarea returnată a obiectului Java în obiecte business răspuns. Legarea EJB de import foloseşte handler-ul de date JAX-WS pentru a transforma EJB-urile cerere în obiecte business cerere şi pentru a transforma obiectul business răspuns într-o valoare returnată.

Acest handler de date mapează datele de la interfaţa WSDL specificată în SCA cu interfaţa Java EJB ţintă (şi vice versa) folosind specificaţiile pentru JAX-WS (Java API for XML Web Services) şi specificaţiile pentru JAXB (Java Architecture for XML Binding).

Notă: Suportul actual este limitat la specificaţiile pentru JAX-WS 2.1.1 şi JAXB 2.1.3.

Handler-ul de date specificat la nivelul legării EJB este folosit pentru a realiza procesarea cererilor, răspunsurilor, defectelor şi excepţiilor apărute în timpul rulării.

Notă: Pentru defecte, poate fi specificat un anumit handler de date pentru fiecare dintre ele prin specificarea proprietăţii de configurare faultBindingType. Acesta înlocuieşte valoarea specificată la nivelul legării EJB.

Handler-ul de date JAX-WS este folosit în mod implicit atunci când legarea EJB are o interfaţă WSDL. Acest handler-ul de date nu poate fi folosit pentru a transforma un mesaj SOAP care reprezintă o invocare JAX-WS către un obiect de date.

Legarea de import EJB foloseşte un handler de date pentru a transforma un obiect de date într-o matrice cu elemente de tip Object Java (Object[]). În timpul comunicaţiilor ce au loc la ieşire în, are loc următoarea procesare:

1. Legarea EJB setează tipul aşteptat, elementul aşteptat şi numele metodei ţintă în BindingContext pentru a se potrivi cu cele specificate în WSDL.
2. Legarea EJB invocă metoda de transformare pentru obiectul de date care are nevoie de transformarea datelor.
3. Handler-ul de date returnează un Object[] care în care sunt reprezentaţi parametrii metodei (în ordinea definirii lor în cadrul metodei).
4. Legarea EJB foloseşte Object[] pentru a invoca metoda în interfaţa EJB ţintă.

Legarea pregăteşte de asemenea un Object[] pentru a procesa răspunsul din invocarea EJB.

- Primul element din Object[] este valoarea returnată din invocarea metodei Java.
- Valorile următoare reprezintă parametrii de intrare pentru metodă.

Aceasta este necesară pentru a suporta parametrii de tip In/Out şi Out.

Pentru parametrii de tip Out, valorile trebuie să fie returnate în obiectul datelor de răspuns.

Handler-ul de date procesează şi transformă valorile găsite în Object[], iar apoi returnează un răspuns către obiectul de date.

Handler-ul de date suportă xs:AnyType, xs:AnySimpleType şi xs:Any împreună cu alte tipuri de date XSD. Pentru a activa suportul pentru xs:Any, utilizaţi **@XmlAnyElement (lax=true)** pentru proprietatea JavaBeans în codul Java aşa cum este afişat în exemplul următor:

```
public class TestType {
 private Object[] object;

 @XmlAnyElement (lax=true)
 public Object[] getObject() {
  return object;
 )

 public void setObject (Object[] object) {
  this.object=object;
 )
)
```

Acest lucru face obiectul proprietate din TestType un câmp xs:any. Valoarea clasei Java folosite în câmpul xs:any ar trebui să aibă adnotarea **@XmlAnyElement**. De exemplu, dacă clasa Java folosită pentru a popula obiectul de tip matrice este Address, atunci această clasă ar trebui să aibă adnotarea **@XmlRootElement**.

Notă: Pentru a personaliza maparea de la tipul XSD la tipurile Java definite prin specificaţia JAX-WS, modificaţi adnotările JAXB, astfel încât să se potrivească nevoilor afacerii dvs. Handler-ul de date JAX-WS suportă xs:any, xs:anyType şi xs:anySimpleType.

Pentru handler-ul de date JAX-WS se aplică următoarele restricţii:

- Handler-ul de date nu include suport pentru adnotarea **@WebParam** din atributul antetului.
- Spaţiul de nume pentru fişierele schemei obiectului business (fişiere XSD) nu include maparea implicită din numele pachetului Java. Adnotarea **@XMLSchema** din package-info.java, de asemenea, nu funcţionează. Singura modalitate de a crea un XSD cu un spaţiu de nume este de a utiliza adnotările **@XmlType** şi **@XmlRootElement**. **@XmlRootElement** defineşte spaţiul nume ţintă pentru elementul global în tipuri JavaBeans.
- Vrăjitorul pentru importul EJB nu creează fişiere XSD pentru clasele fără legătură. Versiunea 2.0 nu suportă adnotare **@XmlSeeAlso**, aşa că, dacă nu se face referire în mod direct de la clasa părinte către clasa copil, nu se creează un XSD. Soluţia la această problemă este să rulaţi SchemaGen pentru astfel de clase copil.

  SchemaGen este un utilitar al liniei de comandă (localizat în directorul *WPS_Install_Home*/bin) furnizat pentru a crea fişiere XSD pentru un bean dat. Aceste XSD-uri trebuie să fie copiate manual în modulul pentru ca soluţia să funcţioneze.

Selector defect EJB:

Selectorul defectului EJB determină dacă o invocare EJB a rezultat într-un defect, o excepţie de runtime sau într-un răspuns cu succes.

În cazul în care este detectat un defect, selectorul de defect EJB returnează numele defectului nativ către runtime-ul legării, astfel încât handler-ul datelor JAX-WS pot converti obiectul de tip excepţie într-un obiect business defect.

Într-un răspuns cu succes (fără defect), legarea de import EJB asamblează un obiect Java de tip matrice (Object[]) pentru a returna valorile.

- Primul element din Object[] este valoarea returnată din invocarea metodei Java.
- Valorile următoare reprezintă parametrii de intrare pentru metodă.

Aceasta este necesar pentru a suporta parametrii de tip In/Out şi Out.

Pentru scenariile de excepţie, legarea asamblează un Object[], iar primul element reprezintă excepţia aruncată de metodă.

Selectorul de defect poate returna oricare dintre următoarele valori:

Tabela 58. Valori returnate

Tip	Valoare returnată	Descriere
Defect	**ResponseType.FAULT**	Returnată atunci când Object[] transmis conţine un obiect de tip excepţie.
Excepţie runtime	**ResponseType.RUNTIME**	Returnată în cazul în care obiectul de tip excepţie nu se potriveşte cu nici unul dintre tipurile de excepţii declarate în cadrul metodei.
Răspuns normal	**ResponseType.RESPONSE**	Returnată în toate celelalte cazuri.

În cazul în care selectorul de defect returnează o valoare **ResponseType.FAULT**, atunci este returnat numele defectului nativ. Acest nume nativ de defect este folosit de legare pentru a determina numele defectului WSDL corespunzător din model şi pentru a invoca handler-ul corect pentru datele defecte.

Selector funcţie EJB:

Legările EJB folosesc un selector pentru funcţia de import (pentru procesarea la ieşire) sau un selector pentru funcţia de export (pentru procesarea la intrare) pentru a determina ce metodă EJB să apeleze.

Selector pentru funcţia de import

Pentru procesarea la ieşire, selectorul pentru funcţia de import derivă tipul metodei EJB în funcţie de numele operaţiei invocate de componenta SCA care este legată de importul EJB. Selectorul funcţiei caută adnotarea @WebMethod în clasa Java adnotată JAX-WS generată de Integration Designer pentru a determina numele operaţiei ţintă asociate.

- În cazul în care adnotarea @WebMethod există, selectorul funcţiei foloseşte această adnotare pentru a determina maparea corectă a metodei Java pentru metoda WSDL.
- În cazul în care adnotarea @WebMethod lipseşte, selectorul funcţiei presupune că numele metodei Java este acelaşi cu cel al operaţiei invocate.

Notă: Acest selector de funcţie este valid doar pentru o interfaţă de tip WSDL într-un import EJB, şi nu pentru o interfaţă de tip Java dintr-un import EJB.

Selectorul funcţiei returnează un obiect java.lang.reflect.Method care reprezintă metoda interfeţei EJB.

Selectorul funcţiei foloseşte un Obiect Java de tip matrice (Object[]) pentru a reţine răspunsul de la metoda ţintă. Primul element din Object[] este o metodă Java care are numele WSDL, iar al doilea element din Object[] este obiectul business de intrare.

Selector pentru funcţia de export

Pentru procesarea la intrare, selectorul pentru funcţia de export derivă metoda ţintă, astfel încât să fie invocată din metoda Java.

Selectorul funcţiei de export mapează numele operaţiei Java invocate de clientul EJB în numele operaţiei din interfaţa componentei ţintă. Numele metodei este returnat sub forma unui şir de caractere şi este rezolvat de runtime-ul SCA în funcţie de tipul interfeţei al componentei ţintă.

Legări EIS

Legările EIS (Enterprise information system) asigură conectivitatea între componente SCA şi un EIS extern. Această comunicaţie este realizată folosind exporturile EIS şi importurile EIS care suportă adaptoarele de resurse JCA 1.5 şi WebSphere Adapters.

Componentele dumneavoastră SCA ar putea impune ca datele să fie transferate către sau de la un EIS extern. Atunci când creaţi un modul SCA care necesită o astfel de conectivitate, veţi include (în plus faţă de componenta dvs. SCA) un import sau un export cu o legare EIS pentru comunicaţia cu un anumit EIS extern.

Adaptoarele de resurse din IBM Integration Designer sunt utilizate în contextul unui import sau export. Dezvoltaţi un import sau un export cu ajutorul vrăjitorului de servicii externe, iar în timpul dezvoltării includeţi adaptorul de resurse. Un import EIS care permite aplicaţiei dumneavoastră să invoce un serviciu într-un sistem EIS sau un export EIS care permite unei aplicaţii dintr-un sistem EIS să invoce un serviciu dezvoltat în IBM Integration Designer sunt create cu un adaptor de resurse. De exemplu, veţi crea un import cu adaptorul JD Edwards pentru a invoca un serviciu în sistemul JD Edwards.

Atunci când utilizaţi vrăjitorul pentru servicii externe, informaţiile legate de legarea EIS sunt create pentru dvs. De asemenea, puteţi utiliza o altă unealtă, editorul de asamblare, pentru a adăuga sau modifica informaţiile legate de legare. Vedeţi Accesarea serviciilor externe cu adaptoare pentru informaţii suplimentare.

După ce modulul care conţine legarea EIS este implementat pe server, puteţi utiliza consola administrativă pentru a vizualiza informaţii despre legare sau pentru a configura legarea.

Privire generală asupra legărilor EIS:

Legarea EIS (enterprise information system), când este folosită cu un adaptor de resurse JCA, vă lasă să accesaţi servicii de pe un sistem de informaţii de întreprindere sau să vă faceţi serviciile disponibile EIS-ului.

Următorul exemplu arată cum un modul SCA numit ContactSyncModule sincronizează informaţii de contact între un sistem Siebel şi un sistem SAP.

1. Componenta SCA numită ContactSync ascultă (prin intermediul unui export de aplicaţie EIS numit Contact Siebel) modificări asupra contactelor Siebel.
2. Însăşi componenta SCA ContactSync foloseşte o aplicaţie SAP (printr-un import de aplicaţie EIS) pentru a actualiza informaţiile de contact SAP corespunzător.

Deoarece structurile de date folosite pentru memorarea contactelor sunt diferite în sistemele Siebel şi SAP, componenta SCA ContactSync trebuie să ofere mapare.

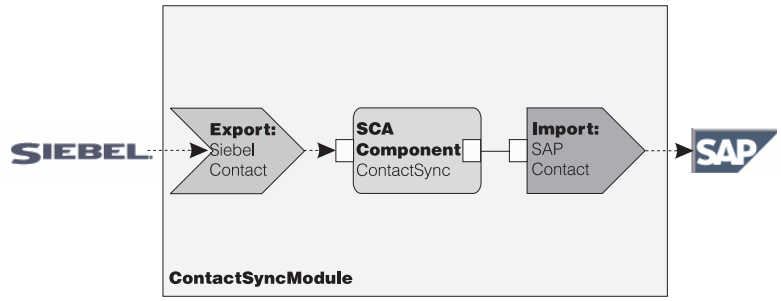


Figura 76. Flux de la un sistem Siebel la un sistem SAP

Exportul Contact Siebel şi importul Contact SAP au configurate adaptoarele de resurse corespunzătoare.

Caracteristici cheie ale legărilor EIS:

Un import EIS este un import SCA ce permite componentelor din modulul SCA să utilizeze aplicaţii EIS definite în afara modulului. Un import EIS este utilizat pentru a transfera date de la componenta SCA la un EIS extern; un export EIS este utilizat pentru a transfera date de la un EIS extern la modulul SCA.

Importuri

Rolul importului EIS este de a umple golul dintre componentele SCA şi sistemele EIS externe. Aplicaţiile externe pot fi tratate ca un import EIS. În acest caz, importul EIS trimite date către EIS extern şi primeşte opţional date în răspuns.

Importul EIS furnizează componentelor SCA cu o vizualizare uniformă a aplicaţilor externe modulului. Acest lucru permite componentelor să comunice cu un EIS extern, cum ar fi SAP, Siebel, sau PeopleSoft, utilizând un model consistent SCA.

Pe partea clientului al importului, există o interfaţă, expusă de aplicaţia de import EIS, cu una sau mai multe metode, fiecare luând obiecte de date ca argumente şi returnează valori. Pe partea de implementare, există un CCI implementat de un adaptor de resurse.

Implementarea runtime a importului EIS conectează interfaţa de pe partea clientului şi CCI-ul. Importul mapează invocarea metodei pe interfaţă cu invocarea de pe CCI.

Legările sunt create la trei niveluri: legarea de interfaţă, ce apoi utilizează legările de metode conţinute, ce mai apoi utilizează legările de date.

Legarea de interfaţă leagă interfaţa importului cu conexiunea la sistemul EIS ce furnizează aplicaţia. Acest lucru reflectă faptul că setul de aplicaţii, reprezentat de interfaţă, este furnizat de instanţa specifică a EIS, iar conexiunea furnizează accesul la această instanţă. Elementul de legare conţine proprietăţi cu destule informaţii pentru a crea conexiunea (aceste proprietăţi fac parte din instanţa javax.resource.spi.ManagedConnectionFactory).

Legarea de metodă asociază metoda cu interacţiunea specifică cu sistemul EIS. Pentru JCA, interacţiunea este caracterizată de setul de proprietăţi al implementării de interfaţă javax.resource.cci.InteractionSpec. Elementul de interacţiune al legării de metodă conţine aceste proprietăţi, împreună cu numele clasei, astfel furnizând destule informaţii pentru a realiza interacţiunea. Legarea de metodă utilizează legări de date ce descriu maparea argumentului şi rezultatului metodei de interfaţă la reprezentarea EIS.

Scenariul runtime pentru un import EIS este după cum urmează:

1. Metoda de pe interfaţa de import este invocată utilizând modelul de programare SCA.
2. Cererea, ce ajunge la importul EIS, conţine numele metodei şi argumentele acesteia.
3. Importul întâi creează o implementare legare de interfaţă; apoi, utilizând date din legarea de import, acesta creează o ConnectionFactory şi le asociază pe cele două. Adică, importul apelează setConnectionFactory pe legarea de interfaţă.
4. este creată implementarea legării de metodă ce se potriveşte cu metoda invocată.
5. Instanţa javax.resource.cci.InteractionSpec este creată şi populată; apoi, legările de date sunt utilizate pentru a lega argumentele metodei la un format înţeles de adaptorul de resurse.
6. Interfaţa CCI este utilizată pentru a realiza interacţiunea.
7. Atunci când apelul este returnat, legarea de date este utilizată pentru a crea rezultatul invocării, iar acesta este returnat apelantului.

Exporturile

Rolul exportului EIS este de a face o punte între o componentă SCA şi un EIS extern. Aplicaţiile externe pot fi tratate ca un export EIS. În acest caz, aplicaţia externă trimite datele sale în formă de notificări periodice. Un export EIS poate fi gândit ca o aplicaţie de abonare ce ascultă o cerere externă de la EIS. Componenta SCA ce utilizează exportul EIS o vizualizează ca o aplicaţie locală.

Exportul EIS furnizează componentelor SCA o vizualizare uniformă a aplicaţiilor externe modulului. Acest lucru le permite componentelor să comunice cu un EIS, cum ar fi SAP, Siebel, sau PeopleSoft, utilizând un model SCA consistent.

Exportul prezintă o implementare ascultător ce primeşte cereri de la EIS. Ascultătorul implementează o interfaţă ascultător specifică adaptorului de resurse. Exportul conţine de asemenea o interfaţă ce implementează componente, expusă la EIS prin export.

Implementarea runtime a unui export EIS conectează ascultătorul cu interfaţa ce implementează componente. Exportul mapează cererea EIS cu invocarea operaţiilor corespunzătoare de pe componente. Legările sunt create la trei niveluri: o legare ascultător, ce apoi utilizează o metodă nativă conţinută , ce apoi utilizează o legare de date .

Legarea ascultător leagă ascultătorul ce recepţionează cererile cu componenta expusă prin export. Definiţia de export conţine numele componentei; runtime-ul o localizează şi înaintează cererile la aceasta.

Legarea de metodă nativă asociază metoda nativă sau tipul de eveniment recepţionat de către ascultător cu operaţia implementată de componenta expusă prin calea exportului. Nu există nicio relaţie între metoda invocată pe ascultător şi tipul de eveniment; toate evenimentele ajung prin una sau mai multe metode ale ascultătorului. Legarea de metodă nativă utilizează selectorul de funcţii definit în export pentru a extrage numele metodei netive din datele de intrare şi legările de date pentru a lega formatul de date al EIS cu un format înţeles de componentă.

Scenariul runtime pentru un export EIS este după cum urmează:

1. Cererea EIS declanşează invocarea metodei pe implementarea ascultătorului.
2. Ascultătorul localizează şi invocă exportul, pasându-i toate argumentele de invocare.
3. Exportul creează implementarea legării ascultătorului.
4. Exportul instanţiază selectorul de funcţii şi îl setează pe legarea ascultătorului.
5. Exportul iniţializează legările metodelor native şi le adaugă la legarea ascultătorului. Pentru fiecare legare de metodă nativă, legările de date sunt de asemenea iniţializate.
6. Exportul invocă legarea ascultătorului.
7. Legarea ascultătorului localizează componentele exportate şi utilizează selectorul de funcţii pentru a extrage numele metodei native.
8. Acest nume este utilizat pentru a localiza legarea metodei native, ce apoi invocă componente destinaţie.

Stilul de interacţiune al adaptorului permite legării de export EIS să invoce componenta destinaţie fie asincron (implicit) fie sincron.

Adaptoare de resurse

Dezvoltaţi un import sau un export cu vrăjitorul de servicii externe şi, în dezvoltarea acestuia, includeţi un adaptor de resurse. Adaptoarele ce vin cu IBM Integration Designer folosite pentru a accesa sisteme CICS, IMS, JD Edwards, PeopleSoft, SAP şi Siebel sunt intenţionate doar pentru scopuri de dezvoltare şi testare. Acest lucru presupune că le utilizaţi pentru a dezvolta şi a testa aplicaţiile dumneavoastră.

Odată ce aţi implementat aplicaţia, veţi avea nevoie de adaptoare runtime licenţiate pentru a o rula. Totuşi, atunci când construiţi serviciul dumneavoastră, puteţi îngloba adaptorul împreună cu serviciul. Licenţierea adaptorului vă poate permite să utilizaţi adaptorul înglobat ca adaptor runtime licenţiat. Aceste adaptoare se potrivesc cu Arhitectura Java EE Connector (JCA 1.5). JCA, un standard deschis, este standardul Java EE pentru conectivitatea EIS. JCA furnizează un cadru de lucru gestionat; care este, Quality of Service (QoS) este furnizat de un server de aplicaţii, care oferă gestiunea şi securitatea ciclului de viaţă pentru tranzacţii. Acestea se potrivesc de asemenea cu specificaţiile Enterprise Metadata Discovery cu excepţia adaptorului de resurse IBM CICS ECI şi a conectorului IBM IMS pentru Java.

Adaptoarele WebSphere Business Integration, un set mai vechi de adaptoare, sunt de asemenea suportate de către vrăjitor.

Resursele Java EE

Modulul EIS, un modul SCA ce urmează tiparul modulului EIS, poate fi implementat pe platforma Java EE.

Implementarea modulului EIS pe platforma Java EE are ca rezultat o aplicaţie care este gata de pornire, împachetată ca fişier EAr şi implementată pe server. Toate artefactele Java EE şi resursele sunt create; aplicaţia este configurată şi gata de rulare.

Proprietăţi dinamice JCA InteractionSpec şi ConnectionSpec:

Legarea EIS poate accepta intrare pentru InteractionSpec şi ConnectionSpec specificate prin utilizarea unui obiect bine definit de date copil ce acompaniază datele utile. Aceasta permite interacţiuni cerere-răspuns dinamice cu un adaptor de resurse prin InteractionSpec şi autentificare de componentă prin ConnectionSpec.

javax.cci.InteractionSpec conţine informaţii despre cum ar trebui tratată cererea de interacţiune cu adaptorul de resurse. Poate de asemenea conţine informaţii despre cum a fost obţinută interacţiunea după cerere. Aceste comunicări pe două căi prin interacţiuni sunt uneori referite ca *conversaţii*.

Legarea EIS aşteaptă datele utile ce vor fi un argument pentru adaptorul de resurse să conţină un obiect de date copil numit **proprietăţi**. Acest obiect de date proprietăţi va conţine perechi nume/valoare, cu numele proprietăţilor InteractionSpec într-un anumit format. Regulile de formatare sunt:

- Numele trebuie să înceapă cu prefixul **IS**, urmat de numele proprietăţii. De exemplu, un o specificaţie de interacţiune cu o proprietate JavaBeans numită **InteractionId** ar specifica numele proprietăţii ca **ISInteractionId**.
- Perechea nume/valoare reprezintă numele şi valoarea tipului simplu al proprietăţii InteractionSpec.

În acest exemplu, o interfaţă specifică faptul că intrarea unei operaţii este un obiect de date **Account**. Această interfaţă invocă o aplicaţie de legare de import EIS cu intenţia de a trimite şi recepţiona o proprietate InteractionSpec dinamică numită **workingSet** cu valoarea **xyz**.

Graficul operaţional sau obiectele business de pe server conţin un obiect business **properties** de bază ce permite trimiterea datelor specifice protocolului cu datele utile (payload). Acest obiect business **properties** este încorporat şi nu trebuie specificat în schema XML când construiţi un obiect business. Trebuie doar creat şi utilizat. Dacă aveţi definite propriile tipuri de date bazat pe o schemă XML, trebuie să specificaţi un element **properties** ce vă conţine perechile nume/valoare aşteptate.

```
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");
```

Creaţi datele utile.

```
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");
```

Pentru numele workingSet, setaţi valoarea aşteptată (**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");
```

Puteţi folosi proprietăţi ConnectionSpec pentru autentificarea componentelor dinamice. Se aplică aceleaşi reguli ca mai sus, exceptând faptul că prefixul numelui proprietăţii trebiue să fie **CS** (în loc de **IS**). Proprietăţile ConnectionSpec nu sunt cu sens dublu. Acelaşi obiect de date de **proprietăţi** poate conţine atât proprietăţi IS cât şi CS.

Pentru a utiliza proprietăţi ConnectionSpec, setaţi **resAuth** specificat pe legarea de import la **Application**. De asemenea, asiguraţi-vă că adaptorul de resurse suportă autorizarea componentelor. Vedeţi capitolul 8 din J2EE Connector Architecture Specification pentru detalii suplimentare.

Clienţi externi cu legări EIS:

Serverul poate trimite mesaje către, sau primi mesaje de la, clienţi externi folosind legări EIS.

Un client extern, de exemplu un portal web sau un EIS, trebuie să trimită un mesaj către un modul SCA din server sau trebuie să fie invocat de o componentă din cadrul serverului.

Clientul invocă importul EIS la fel ca orice altă aplicaţie, folosind fie DII (Dynamic Invocation Interface) fie interfaţa Java.

1. Clientul extern creează o instanţă a ServiceManager şi caută importul EIS folosindu-i numele de referinţă. Rezultatul căutării este o implementare a interfeţei serviciului.
2. Clientul creează un argument de intrare, un obiect de date generic, creat dinamic folosind schema obiectului de date. Acest pas este realizat folosind implementarea interfeţei Service Data Object DataFactory.
3. Clientul extern invocă EIS-ul şi obţine rezultatele necesare.

Alternativ, clientul poate invoca importul EIS folosind interfaţa Java.

1. Clientul creează o instanţă a ServiceManager şi caută importul EIS folosindu-i numele de referinţă. Rezultatul căutării este o interfaţă Java a importului EIS.
2. Clientul creează un argument de intrare şi un obiect de date tastat.
3. Clientul invocă EIS şi obţine rezultatele necesare.

Interfaţa exportului EIS defineşte interfaţa componentei SCA exportate ce este disponibilă aplicaţiilor EIS externe. Vă puteţi gândi la această interfaţă ca la interfaţa pe care o va invoca o aplicaţie externă (precum SAP sau PeopleSoft) prin implementarea runtime-ului aplicaţiei exportului EIS.

Exportul foloseşte EISExportBinding pentru a lega servicii exportate de aplicaţia EIS externă. Vă permite să abonaţi o aplicaţie conţinută în modulul dumneavoastră SCA să asculte cereri de servicii EIS. Legarea de export EIS specifică maparea dintre definiţia evenimentelor de intrare aşa cum este înţeleasă de adaptorul de resurse (folosind interfeţe Java EE Connector Architecture) şi invocarea operaţiilor SCA.

EISExportBinding necesită ca serviciile EIS externe să fie bazate pe contracte de intrare Java EE Connector Architecture 1.5. EISExportBinding necesită ca un handler de date sau o legare de date să fie specificate, fie la nivel de legare, fie la nivel de metodă.

Legări JMS

Un furnizor JMS (Java Message Service) permite mesageria bazată pe API-ul Java Messaging Service şi pe modelul de programare. Acesta asigură fabrici de conexiune JMS pentru a crea conexiuni pentru destinaţiile JMS şi pentru a trimite şi primi mesaje.

Legările JMS pot fi folosite atunci când interacţionaţi cu legarea furnizorului SIB (Service Integration Bus) şi sunt conforme cu JMS şi JCA 1.5.

Puteţi utiliza legările de export şi import JMS ale unui modul SCA (Service Component Architecture) pentru a apela şi primi mesaje de la sisteme JMS externe.

Legările de import şi export JMS asigură integrarea în aplicaţiile JMS folosind furnizorul SIB JMS bazat pe JCA 1.5 care face parte din WebSphere Application Server. Alte adaptoare pentru resurse JMS bazate pe JCA 1.5 nu sunt suportate

Privire generală asupra legărilor JMS:

Legările JMS asigură conectivitatea între mediul SCA (Service Component Architecture) şi sistemele JMS.

Legări JMS

Componentele majore ale ambelor legări JMS pentru import şi export sunt:

- Adaptor resursă: activează conectivitatea bidirecţională, gestionată între un modul SCA şi sistemele JMS externe
- Conexiuni: încapsulează o conexiune virtuală între un client şi o aplicaţie furnizor
- Destinaţii: sunt folosite de un client pentru a specifica ţinta mesajelor pe care le produce sau sursa mesajelor pe care le primeşte
- Date de autentificare: sunt folosite pentru a securiza accesul la legare

Caracteristicile cheie ale legărilor JMS

Anteturi speciale

Proprietăţile anteturilor speciale sunt utilizate în importurile şi exporturile JMS pentru a-i spune destinaţiei cum să trateze mesajul.

De exemplu, TargetFunctionName mapează de la metoda nativă la metoda de funcţionare.

Resurse Java EE

Un număr de resurse Java EE este creat la implementarea importurilor şi exporturilor JMS într-un mediu Java EE.

ConnectionFactory
Utilizată de clienţi pentru a crea o conexiune la furnizorul JMS.

ActivationSpec
Importurile utilizează acest lucru pentru a recepţiona răspunsul la o cerere; exporturile o utilizează la configurarea punctelor finale ale mesajelor ce reprezintă ascultători de mesaje în interacţiunea acestora cu sistemul de mesagerie.

Destinaţii

- Destinaţie de trimitere: pe un import, aceasta este unde cererea sau mesajul de ieşire este trimis; pe un export, aceasta este destinaţia unde mesajul de răspuns va fi trimis, dacă nu este depreciată de câmpul antetului JMSReplyTo din mesajul de intrare.
- Destinaţie de recepţionare: unde mesajul de intrare va fi plasat; cu importuri, acesta este un răspuns; cu exporturi, este o cerere.
- Destinaţie de callback: destinaţia sistemului SCA JMS utilizată pentru a stoca informaţii de corelare. Nu citiţi sau scrieţi la această destinaţie.

Operaţia de instalare creează ConnectionFactory şi trei destinaţii. De asemenea creează ActivationSpec pentru a permite ascultătorului de mesaje runtime să asculte pentru replici pe destinaţia de recepţionare. Proprietăţile ale acestor resurse sunt specificate în fişierul de import sau de export.

Integrarea JMS şi adaptoarele de resurse:

JMS (Java Message Service) asigură integrare printr-un adaptor disponibil de resurse bazat pe JMS JCA 1.5. Suportul complet pentru integrarea JMS este furnizat pentru adaptorul de resurse JMS SIB (Service Integration Bus).

Utilizaţi un furnizor JMS pentru adaptorul de resurse JCA 1.5 atunci când doriţi integrarea într-un sistem extern JMS conform cu JCA 1.5. Serviciile externe conforme cu JCA 1.5 pot primi şi trimite mesaje pentru a se integra în componentele SCA-ului (service component architecture) dumneavoastră folosind adaptorul de resurse JMS SIB.

Utilizarea altor adaptoare de resurse JCA 1.5 specifice furnizorului nu este acceptată.

Legări de import şi export JMS:

Puteţi face ca modulele SCA să interacţioneze cu serviciile furnizate de aplicaţiile JMS externe utilizând legături de import şi export JMS.

Legări de import JMS

Conexiunile către furnizorul JMS asociat destinaţiilor JMS sunt create folosind o fabrică de conexiuni JMS. Utilizaţi obiecte administrative pentru fabrica de conexiuni pentru a gestiona fabricile de conexiuni JMS pentru furnizorul implicit de mesaje.

Interacţiunea cu sistemele JMS externe include utilizarea destinaţiilor pentru trimiterea cererilor sau primirea răspunsurilor.

Se oferă suport pentru două tipuri de scenarii de utilizare pentru legările de import JMS, în funcţie de tipul operaţiei invocate:

- Primul tip: Importul JMS pune un mesaj în destinaţia de trimitere configurată în legarea de import. Nu este setat nimic în câmpul replyTo din antetul JMS.
- Al doilea tip (cerere-răspuns): Importul JMS pune un mesaj în destinaţia de trimitere, iar apoi persistă răspunsul pe care îl primeşte de la componenta SCA.

  Legarea de import poate fi configurată (folosind câmpul **chema de corelare a răspunsului** în Integration Designer), astfel încât să aştepte ID-ul de corelare al mesajului pentru răspuns ce a fost copiat din ID-ul mesajului de cerere (cel implicit) sau din ID-ul de corelare al mesajului de cerere. Legarea de import poate fi de asemenea configurată astfel încât să utilizez o destinaţie de răspuns dinamică temporară pentru a corela răspunsurile cu cererile. Pentru fiecare cerere se creează o destinaţie temporară, iar importul foloseşte această destinaţie pentru a primi răspunsul.

  Destinaţia receive este setată în proprietatea antetului replyTo din mesajul de ieşire. Un ascultător de mesaje este implementat pentru a asculta la destinaţia de recepţionare, iar în momentul în care este primit un răspuns, acesta transminte răspunsul înapoi către componentă.

Pentru ambele scenarii de utilizare pot fi specificate atât proprietăţile dinamice, cât şi cele statice ale antetului. Proprietăţile statice pot fi setate din legarea metodei de import JMS. Unele dintre aceste proprietăţi au semnificaţii speciale pentru runtime-ul JMS SCA.

Este important de menţionat că JMS este o legare asincronă. Dacă o componentă apelantă invocă un import JMS în mod sincron (pentru o operaţie bidirecţională), atunci aceasta este blocată până când răspunsul este returnat de serviciul JMS.

Figura 32 la pagina 113 ilustrează modul în care importul este legat de serviciul extern.

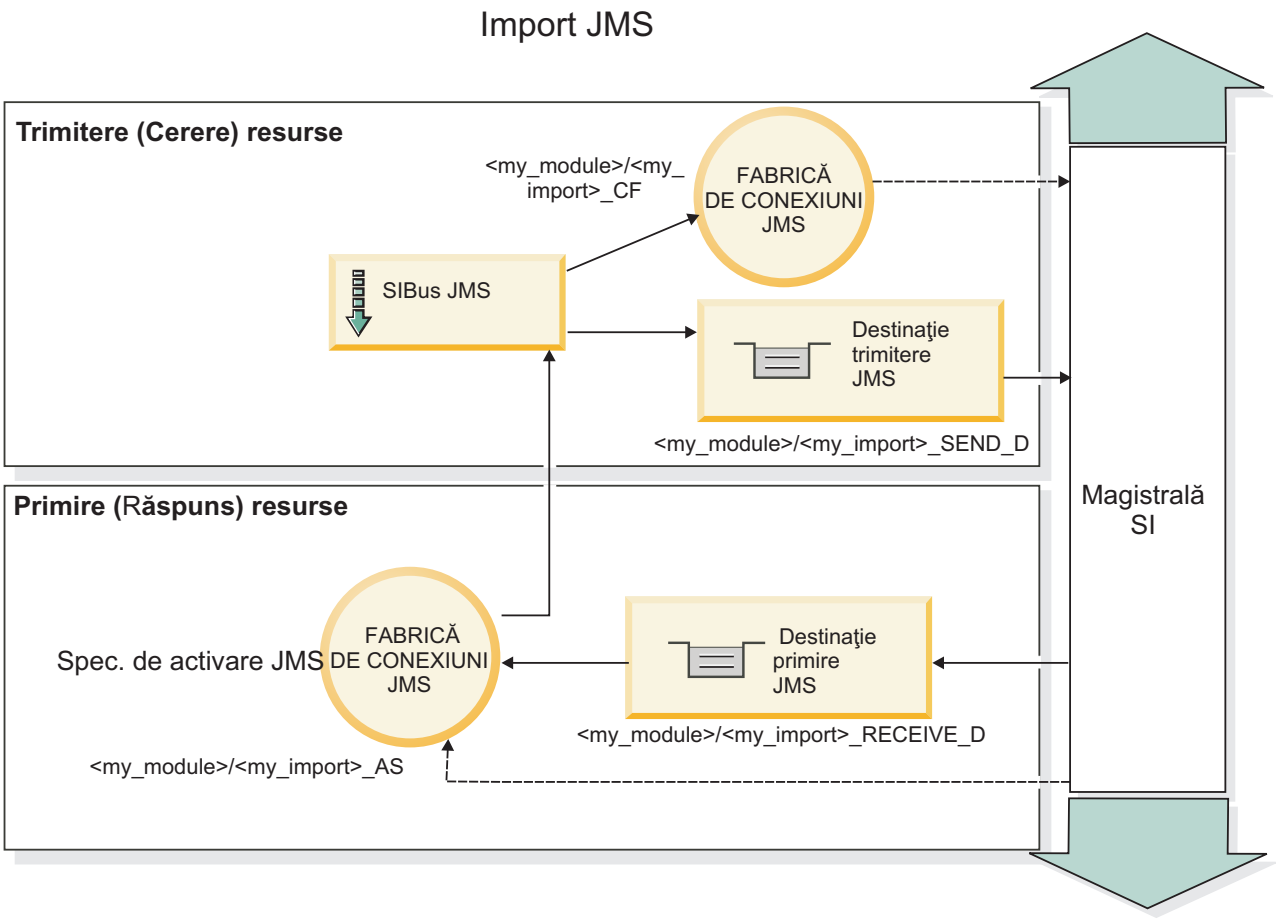


Figura 77. Resurse pentru legarea de import JMS

Legări de export JMS

Legări de export JMS pun la dispoziţie mijloacele prin care modulele SCA asigură servicii către aplicaţiile JMS externe.

Conexiunea care face parte dintr-un export JMS este o specificaţie de activare configurabilă.

Un export JMS a trimis şi a primit destinaţiile.

- Destinaţia receive este acolo unde ar trebui să fie pus mesajul de intrare pentru componenta ţintă.
- Destinaţia send este acolo unde va fi trimis răspunsul, cu excepţia cazului în mesajul de intrare a înlocuit-o folosind proprietatea antetului replyTo.

Un ascultător de mesaje este implementat pentru a asculta cererile ce intră prin destinaţia primire specificată în legarea de export. Destinaţia specificată în câmpul send este utilizat pentru a trimite răspunsul către cererea de intrare în cazul în care componenta invocată furnizează un răspuns. Destinaţia specificată în câmpul replyTo din mesajul de intrare înlocuieşte destinaţia specificată în send.

Figura 33 la pagina 114 ilustrează modul în care solicitantul extern este legat de export.

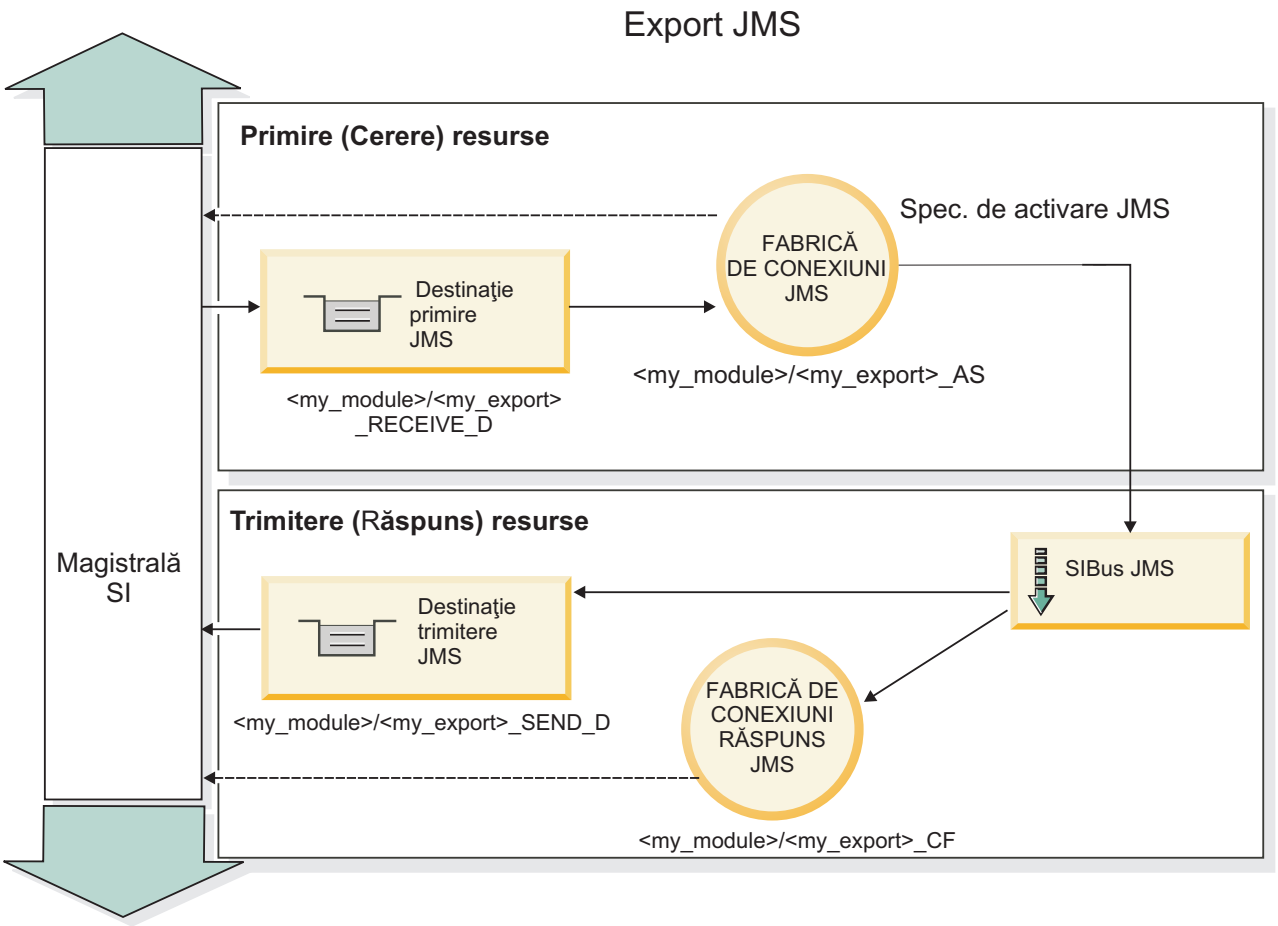


Figura 78. Resurse pentru legarea de export JMS

Anteturi JMS:

Un mesaj JMS conţine două tipuri de anteturi- antetul sistemului JMS şi mai multe proprietăţi JMS. Ambele tipuri de anteturi pot fi accesate fie într-un modul de mediere din SMO (Service Message Object) fie folosind API-ul ContextService.

Antetul sistemului JMS

Antetul sistemului JMS este reprezentat în SMO prin elementul JMSHeader care conţine toate câmpurile care se găsesc de obicei într-un antet JMS. Deşi acestea pot fi modificate în modul de mediere (sau ContextService), unele câmpuri din antetul sistemului JMS setate în SMO nu vor fi trimise în mesajul JMS de ieşire pe măsură ce acestea sunt înlocuite de sistem sau valori statice.

Câmpurile cheie din antetul sistemului JMS care pot fi actualizate într-un modul mediere (sau ContextService) sunt:

- **JMSType** şi **JMSCorrelationID** – valorile proprietăţilor specifice antetului mesajului predefinit
- **JMSDeliveryMode** – valori pentru modul de livrare (persistent sau nepersistent; valoarea implicită este persistent)
- **JMSPriority** – valoare prioritate (de la 0 la 9; valoarea implicită este JMS_Default_Priority)

Proprietăţi JMS

Proprietăţile JMS sunt reprezentate în SMO sub formă de intrări în lista Proprietăţi. Proprietăţile pot fi adăugate, actualizate sau şterse într-o mediere sau folosind API-ul ContextService.

De asemenea, proprietăţile pot fi setate în legarea JMS. Proprietăţile care sunt setate în mod static înlocuiesc setările (cu acelaşi nume) care sunt setate în mod dinamic.

Proprietăţile utilizatorului propagate din alte legări (de exemplu, o legare HTTP) va fi pusă în legarea JMS sub formă de proprietăţi JMS.

Setările propagării anteturilor

Propagarea antetului şi proprietăţilor sistemului JMS fie de la mesajul JMS de intrare către componentele următoare, fie de la componentele anterioare către mesajul JMS de ieşire poate fi controlată prin steguleţul Propagate Protocol Header din legare.

Atunci când Propagate Protocol Header este setat, informaţiilor de antet le este permis să circule către mesaj sau către componenta ţintă, aşa cum este descris în următoarea listă:

- Cerere de export JMS

  Antetul JMS primit în mesaj va fi propagat către o componentă ţintă prin intermediul serviciului de context. Proprietăţile JMS primite în mesaj vor fi propagate către o componentă ţintă prin intermediul serviciului de context.
- Răspuns la exportul JMS

  Oricare dintre câmpurile din antetul JMS din serviciul de contextul va fi utilizat în mesajul de ieşire, dacă nu este înlocuită de proprietăţile statice stabilite în legarea de export JMS. Oricare dintre proprietăţile setate în serviciul de context va fi utilizată în mesajul de ieşire, dacă nu este înlocuită de proprietăţile statice stabilite în legarea de export JMS.
- Cerere de import JMS

  Oricare dintre câmpurile din antetul JMS din serviciul de context va fi utilizat în mesajul de ieşire, dacă nu este înlocuit de proprietăţile statice stabilite în legarea de import JMS. Oricare dintre proprietăţile setate în serviciul de context va fi utilizată în mesajul de ieşire, dacă nu este înlocuită de proprietăţile statice stabilite în legarea de import JMS.
- Răspuns import JMS

  Antetul JMS primit în mesaj va fi propagat către o componentă ţintă prin intermediul serviciului de context. Proprietăţile JMS primite în mesaj vor fi propagate către o componentă ţintă prin intermediul serviciului de context.

Schema de corelare a destinaţiilor de răspuns dinamice temporare JMS:

Schema de corelare a destinaţiilor de răspuns dinamice temporare determină o coadă dinamică unică sau un subiect care urmează să fie creat pentru fiecare cerere trimisă.

Destinaţia statică de răspuns menţionată în import este utilizată pentru a obţine natura cozii dinamice temporare de destinaţie sau subiectul. Acest lucru este setat în câmpul **ReplyTo** al cererii, iar importul JMS ascultă pentru răspunsuri pe această destinaţie. În cazul în care răspunsul este primit este cerut de destinaţia statică a răspunsului pentru prelucrarea asincronă. Câmpul **CorrelationID** din răspuns nu este utilizat şi nu este nevoie să fie setat.

Probleme legate de Tranzacţii

Atunci când este folosită o destinaţie dinamică temporară, cererea trebuie să fie prelucrată în acelaşi fir de execuţie ca răspunsul trimis. Cererea trebuie să fie expediată în afara tranzacţiei globale, şi trebuie să fie comisă înainte să fie primită de serviciul de back-end, şi ca un răspuns să fie returnat.

Persistenţă

Cozile temporare dinamice sunt entităţi de scurtă durată şi nu garantează acelaşi nivel de persistenţă asociat cu o coadă sau subiect static. O coadă dinamică temporară sau subiect nu va mai exista după repornirea serverului şi nici mesajele. După ce mesajul a fost cerut la destinaţia statică de răspuns, acesta îşi păstrează persistenţa definită în mesaj.

Timeout

Importul aşteaptă să primească răspunsul la destinaţia temporară pentru răspunsuri dinamice pentru o durată fixă de timp. Acest interval de timp va fi extras din calificativul de tip Expirare răspuns SCA dacă este setat, iar, în caz contrar, timpul prestabilit este de 60 de secunde. În cazul în care timpul de aşteptare este depăşit, importul aruncă o excepţie de tip ServiceTimeoutRuntimeException.

Clienţi externi:

Serverul poate trimite mesaje către, sau primi mesaje de la, clienţi externi folosind legări JMS.

Un client extern (de exemplu, un portal web sau un sistem de informaţii pentru întreprindere) poate trimite un mesaj către un modul SCA din server, sau poate fi invocat de o componentă din cadrul serverului.

Componentele exportului JMS dezvoltă ascultători de mesaje pentru a asculta cereri ce intră în destinaţia de primire specificată în legarea de export. Destinaţia specificată în câmpul de trimitere este folosită pentru a trimite răspunsul către cererea de intrare dacă aplicaţia invocată furnizează un răspuns. Astfel, un client extern este capabil să invoce aplicaţii cu legarea de export.

Importurile JMS interacţionează cu clienţi externi prin trimiterea de mesaje şi primirea de mesaje de la cozi JMS.

Lucrul cu clienţi externi:

Un client extern (ce este în afara serverului) ar putea avea nevoie să interacţioneze cu o aplicaţie instalată pe server.

Consideraţi un scenariu foarte simplu în care un client extern vrea să interacţioneze cu o aplicaţie de pe server. Figura descrie un scenariu tipic simplu.

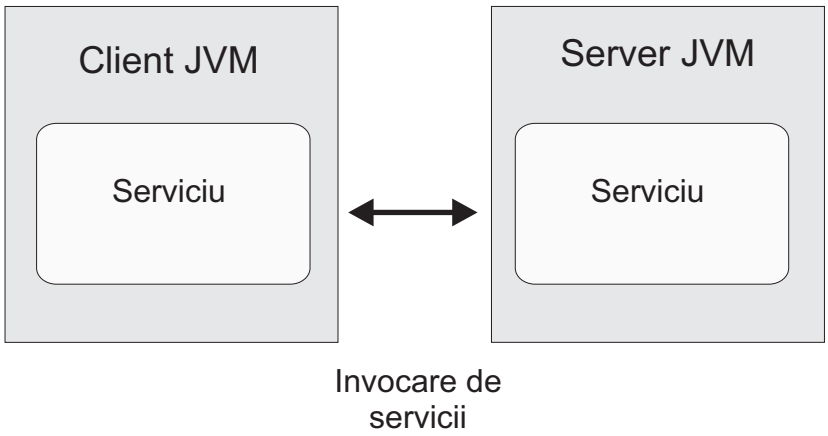


Figura 79. Scenariu simplu utilizare-caz: client extern interacţionează cu aplicaţia serverului

Aplicaţia SCA include un export cu o legare JMS; aceasta face aplicaţia disponibilă pentru clienţi externi.

Atunci când aveţi un client extern în maşina virtuală Java (JVM) separat de serverul dumneavoastră, există mai mult paşi pe care trebuie să-i realizaţi pentru a realiza o conexiune şi pentru a interacţiona cu un export JMS. Clientul obţine un InitialContext cu valorile corecte şi apoi caută resursele prin JNDI. Apoi clientul utilizează clientul specificaţie JMS 1.1 pentru a accesa destinaţiile şi mesajele de trimitere şi primire a destinaţiilor.

Numele implicite JNDI ale resurselor create automat prin runtime sunt listate în subiectul de configurare al acestei secţiuni. Totuşi, dacă aveţi resurse create anterior, utilizaţi acele nume JNDI.

1. Configuraţi destinaţiile JMS şi fabrica de conexiuni pentru a trimite mesajul.

2. Asiguraţi-vă de faptul că contextul JNDI, portul pentru adaptorul de resurse SIB şi portul bootstrapping al mesageriei sunt corecte.

   Serverul utilizează câteva porturi implicite, dar dacă sunt mai multe server instalate pe acel sistem, porturi alternative sunt create la timpul instalării pentru a evita conflicte cu alte instanţe de server. Puteţi utiliza consola administrativă pentru a determina ce porturi ale serverului dumneavoastră sunt implementate. Mergeţi la **Servere > servere de aplicaţii > *nume_server* > Configurare** şi apăsaţi pe **Porturi** din **Comunicaţie**. Puteţi edita portul ce este utilizat.
3. Clientul obţine un context iniţial cu valorile corecte şi apoi caută resurse prin JNDI.
4. Utilizând specificaţiile JMS 1.1, clientul accesează destinaţiile şi mesajele de trimitere şi recepţionare de pe destinaţii.

Depanare legări JMS:

Puteţi diagnostica şi repara problemele cu legările JMS.

Excepţii de implementare

Ca răspuns la condiţii de eroare, implementarea JMS import şi export poate returna una din cele două tipuri de excepţii:

- Service Business Exception: această excepţie este returnată dacă fault specificată pe interfaţa serviciului business (WSDL port type) a survenit.
- Service Runtime Exception: apărută în toate celelalte cazuri. În cele mai multe cazuri, excepţia cause va conţine excepţia originală (JMSException).

  De exemplu, un import aşteaptă numai un mesaj de răspuns pentru fiecare mesaj de cerere. Dacă mai mult de un răspuns soseşte, sau dacă un răspuns întârziat (unul pentru care expirarea răspunsului SCA are loc) soseşte, este aruncată o excepţie Service Runtime. Tranzacţia este repetată, şi mesajul de răspuns este retras din coadă sau este manevrat de către managerul de eveniment.

Condiţii de eşec primar

Condiţiile de eşec primar ale legărilor JMS sunt determinate de către sensurile tranzacţionale, de către configuraţia furnizorului JMS sau prin referire la comportamente existente în alte componente. Condiţiile de eşec primar includ:

- Eşec la conexiunea la furnizorul sau destinaţia JMS.

  Un eşec în conexiunea la furnizorul JMS pentru a primi mesaje va avea ca rezultat începutul căderii ascultătorului de mesaje. Această condiţie va fi jurnalizată în istoricul WebSphere Application Server. Mesajele persistente vor rămâne la destinaţie până când vor fi retrase cu succes (sau expirate).

  Un eşec la conectarea la furnizorul JMS pentru a trimite mesaje în exterior va cauza derularea înapoi a tranzacţiei ce controlează trimiterea.
- Eşecul în analiza unui mesaj de intrare sau în construirea unui mesaj de ieşire.

  Un eşec în legarea de date sau în handler-ul de date cauzează derularea înapoi a tranzacţiei care controlează lucrul.
- Eşec la trimiterea mesajului de ieşire.

  Un eşec la trimiterea unui mesaj cauzează derularea înapoi a tranzacţiei relevante.
- Mesaje de răspuns multiple sau întârziate neaşteptat.

  Importul aşteaptă numai un mesaj de răspuns pentru fiecare mesaj de cerere. De asemenea perioada de timp validă în care un răspuns poate fi primit este determinată de către calificativul SCA Response Expiration de pe cerere. Când un răspuns soseşte sau timpul de expirare este depăşit, înregistrarea de corelare este ştearsă. Dacă mesajele de răspuns ajung neaşteptat sau sosesc târziu, este aruncată o excepţie Service Runtime.
- Excepţie runtime timeout serviciu cauzată de către răspunsul întârziat când se foloseşte schema de corelare a destinaţiei de răspuns dinamic temporară.

  Importul JMS va genera timeout după o perioadă de timp determinată de către calficativul expirare răspuns SCA sau dacă acesta nu este setat va avea valoarea implicită de 60 de secunde.

Mesajele SCA bazate pe JMS ce nu apar în managerul de evenimente eşuate

Dacă mesajele SCA îşi au originea într-o cădere a JMS, ar trebui să vă aşteptaţi să găsiţi aceste mesaje în managerul de evenimente eşuate. Dacă astfel de mesaje nu apar în managerul de evenimente eşuate, asiguraţi-vă că destinaţia de bază SIB a destinaţiei JMS are o valoare maximă de livrări nereuşite mai mare decât **1**. Setarea acestei valori la **2** sau mai mult activează interacţiunea cu managerul de evenimente eşuate în timpul invocărilor SCA pentru legările JMS.

Tratarea excepţiilor:

Modul în care este configurată legarea determină cum excepţiile ce sunt ridicate de handler-e de date sau legări de date sunt tratate. În plus, natura fluxului de mediere dictează comportamentul sistemului când este aruncată o astfel de excepţie.

Poate apărea o varietate de probleme când un handler de date sau o legare de date este apelat(ă) de legarea dumneavoastră. De exemplu, este posibil ca handler-ul de date să primească un mesaj care are date utile corupte, sau este posibil să încerce să citească un mesaj care are un format incorect.

Modul în care legarea dumneavoastră tratează o astfel de excepţie este determinat de modul în care implementaţi handler-ul de date sau legarea de date. Comportamentul recomandat este să vă proiectaţi legarea de date să arunce o **DataBindingException**.

Când orice excepţie runtime, inclusiv o **DataBindingException**, este aruncată:

- Dacă fluxul de mediere este configurat să fie tranzacţional, mesajul JMS , implicit, este memorat în Managerul de evenimente eşuate pentru reluare sau ştergere manuală.

  Notă: Puteţi modifica modul de recuperare de pe legare, astfel încât mesajul să fie derulat înapoi în loc să fie stocat în Managerul de evenimente eşuate.
- Dacă fluxul de mediere nu este tranzacţional, excepţia este înregistrată în istoric şi mesajul este pierdut.

Situaţia este similară pentru un handler de date. Din moment ce handler-ul de date este invocat de legarea de date, orice excepţie a handler-ului de date este înfăşurată într-o excepţie a legării de date. Prin urmare o **DataHandlerException** vă este raportată ca o **DataBindingException**.

Legările Generic JMS

Conectivitatea furnizorilor de legări Generic JMS cu furnizorii terţă parte care sunt conformi cu JMS 1.1. Operarea legărilor Generic JMS este similară cu cea a legărilor JMS.

Serviciul pus la dispoziţie prin intermediul unei legări JMS permite unui modul SCA (Service Component Architecture) să apeleze sau să primească mesaje de la sisteme externe. Sistemul poate fi un sistem extern JMS.

Legarea Generic JMS asigură integrarea cu furnizorii JMS care nu sunt compatibili cu JCA 1.5 şi care suportă JMS 1.1 şi implementează facilitatea opţională JMS Application Server. Legarea JMS Generic suportă acei furnizori JMS (inclusiv Oracle AQ, TIBCO, SonicMQ, WebMethods şi BEA WebLogic) care nu suportă JCA 1.5, dar suportă Application Server Facility a specificaţiei JMS 1.1. Furnizorul de SIBJMS (WebSphere embedded JMS), care este un furnizor JCA 1.5 JMS, nu este suportat de această legare; atunci când se utilizează acest furnizor, utilizaţi “Legări JMS” la pagina 110.

Utilizaţi această legare generică la integrarea unui sistem JMS conform cu care nu este de tip JCA 1.5 în cadrul unui mediu SCA. Aplicaţiile externe ţintă pot primi şi trimite apoi mesaje pentru a se integra într-o componentă SCA.

Privire generală asupra legărilor JMS generice:

Legările JMS generice sunt legări non-JCA JMS care asigură conectivitate între SCA mediul (Service Component Architecture) şi sistemele JMS care sunt în conformitate cu JMS 1.1 şi care implementează facilitatea opţională a serverului de aplicaţii JMS.

Legări JMS generice

Aspectele majore ale legărilor generice pentru import şi export JMS includ următoarele:

- Port ascultător: permite furnizorilor JMS ce nu se bazează pe JCA să primească mesaje şi să le trimită către un MDB (Message Driven Bean)
- Conexiuni: încapsulează o conexiune virtuală între un client şi o aplicaţie furnizor
- Destinaţii: sunt folosite de un client pentru a specifica ţinta mesajelor pe care le produce sau sursa mesajelor pe care le primeşte
- Date de autentificare: sunt folosite pentru a securiza accesul la legare

Legări de import JMS generice

Legările generice de import JMS permit componentelor din modulul dumneavoastră SCA să comunice cu serviciile oferite de furnizorii externi conformi non-JCA 1.5 JMS.

Partea de conexiuni a importului JMS este o fabrică de conexiuni. O fabrică de conexiuni, obiectul folosit de un client pentru a crea o conexiune către un furnizor, încapsulează un set de parametri de configurare definiţi de către un administrator. Fiecare fabrică de conexiuni este o instanţă a uneia dintre interfeţele ConnectionFactory, QueueConnectionFactory sau TopicConnectionFactory.

Interacţiunea cu sistemele JMS externe include utilizarea destinaţiilor pentru trimiterea cererilor sau primirea răspunsurilor.

Se oferă suport pentru două tipuri de scenarii de utilizare pentru legările generice de import JMS, în funcţie de tipul operaţiei invocate:

- Primul tip: Importul generic JMS pune un mesaj în destinaţia de trimitere configurată în legarea de import. Nu se trimite nimic către câmpul replyTo din antetul JMS.
- Al doilea tip (cerere-răspuns): Importul JMS generic pune un mesaj în destinaţia de trimitere, iar apoi persistă răspunsul pe care îl primeşte de la componenta SCA.

  Destinaţia receive este setată în proprietatea antetului replyTo din mesajul de ieşire. Un MDB (bean controlat de mesaj) este implementat pentru a asculta la destinaţia de recepţionare, iar în momentul în care este primit un răspuns, acesta transmite răspunsul înapoi către componentă.

  Legarea de import poate fi configurată (folosind câmpul **Schema de corelare a răspunsulu** în Integration Designer), astfel încât să aştepte ID-ul de corelare al mesajului pentru răspuns ce a fost copiat din ID-ul mesajului de cerere (cel implicit) sau din ID-ul de corelare al mesajului de cerere.

Pentru ambele scenarii de utilizare pot fi specificate atât proprietăţile dinamice, cât şi cele statice ale antetului. Proprietăţile statice pot fi setate din legarea metodei de import generic JMS. Unele dintre aceste proprietăţi au semnificaţii speciale pentru runtime-ul JMS SCA.

Este important de menţionat faptul că JMS este o legare asincronă. Dacă o componentă apelantă invocă un import JMS generic în mod sincron (pentru o operaţie bidirecţională), atunci aceasta este blocată până când răspunsul este returnat de serviciul JMS.

Figura 35 la pagina 120 ilustrează modul în care importul este legat de serviciul extern.

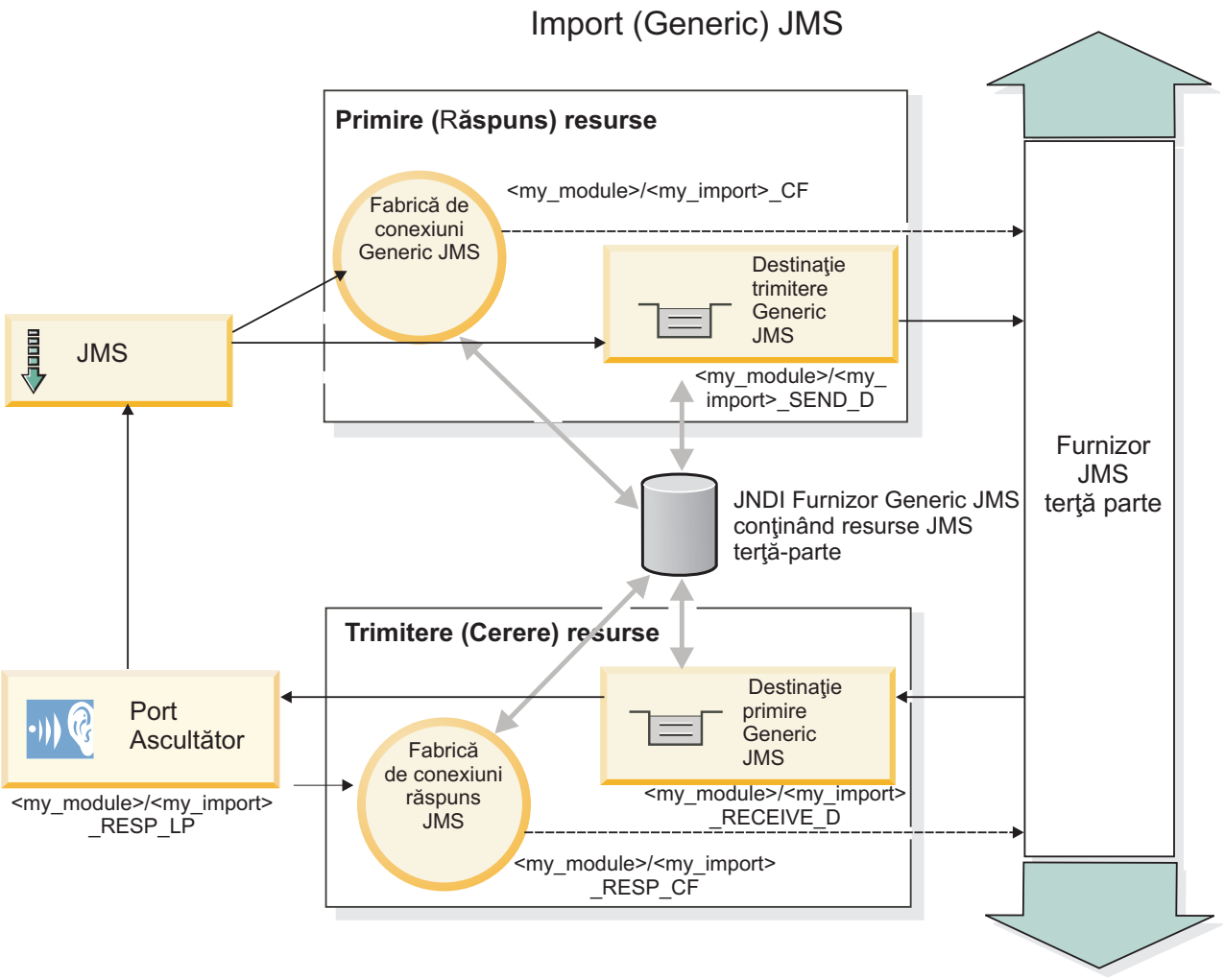


Figura 80. Resurse pentru legarea generică de import JMS

Legări de export JMS generice

Legările de export JMS generice pun la dispoziţie mijloacele prin care modulele SCA asigură servicii către aplicaţiile JMS externe.

Partea de conexiuni a unui export JMS este compus dintr-un ConnectionFactory şi un ListenerPort.

Un export generic JMS a trimis şi a primit destinaţiile.

- Destinaţia primire este acolo unde ar trebui să fie pus mesajul de intrare pentru componenta ţintă.
- Destinaţia trimitere este acolo unde va fi trimis răspunsul, cu excepţia cazului în mesajul de intrare a înlocuit-o folosind proprietatea antetului replyTo.

Este implementat un MDB pentru a asculta cererile ce intră prin destinaţia primire specificată în legarea de export.

- Destinaţia specificată în câmpul trimitere este folosită pentru a trimite răspunsul către cererea de intrare în cazul în care aplicaţia invocată oferă un răspuns.
- Destinaţia specificată în câmpul replyTo din mesajul de intrare înlocuieşte destinaţia specificată în câmpul trimitere.
- Pentru scenariile cerere/răspuns, legarea de import poate fi configurată (folosind câmpul **Schema de corelare a răspunsului** din Integration Designer), astfel încât să se aştepte ca răspunsul să copieze ID-ul mesajului de cerere către câmpul correlation ID al mesajului de răspuns (implicit) sau răspunsul poate copia correlation ID al cererii în câmpul correlation ID din mesajul de răspuns.

Figura 36 la pagina 121 ilustrează modul în care solicitantul extern este legat de export.

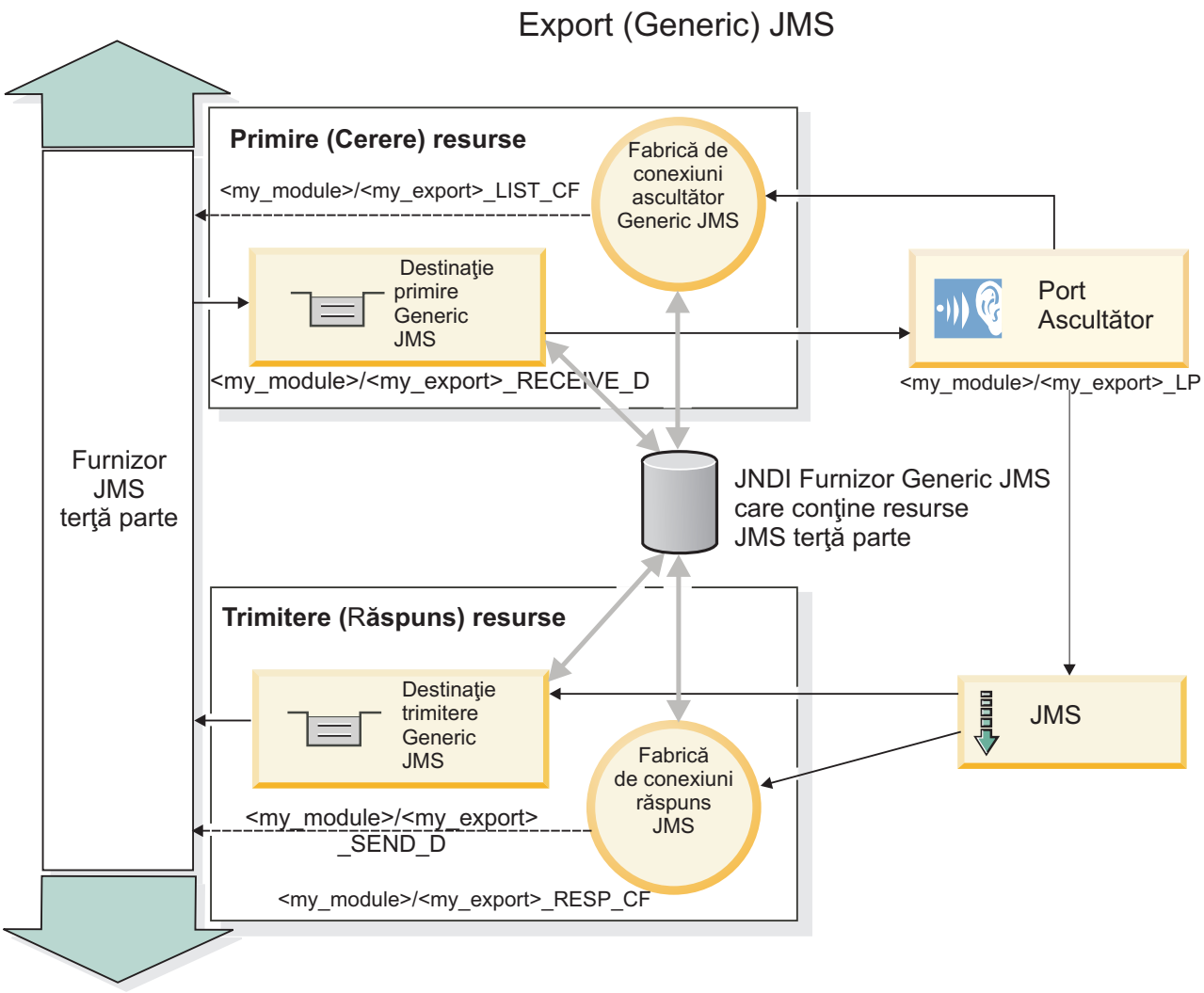


Figura 81. Resurse pentru legarea generică pentru exportul JMS

Caracteristici cheie ale legărilor JMS generic:

Caracteristicile legărilor de import şi export JMS generic sunt consistente cu acelea ale legărilor de import MQ JMS şi cele JMS imbricate WebSphere. Caracteristicile cheie includ definiţii de antet şi acces la resurse Java EE existente. Totuşi, din cauza naturii sale generice, nu există nicio opţiune de conectivitate specifică furnizorilor JMS, iar această legare are capacitate limitată pentru a genera resurse la implementare şi instalare.

Importuri generice

La fel ca aplicaţia de import MQ JMS, implementarea Generic JMS este asincronă şi suportă trei invocări: o cale, două căi (de asemenea cunoscută ca cerere-răspuns), şi callback.

Atunci când importul JMS este implementat, un MDB furnizat de mediul de runtime este implementat. MDB-ul ascultă pentru replici la mesajul de cerere. MDB- ul este asociat cu (ascultă pe) destinaţia trimisă cu cerere în câmpul de antetreplyTo al mesajului JMS.

Exporturi generice

Legările de exporturi JMS generice diferă de legările de export EIS în cazul tratării returnării rezultatului. Un export Generic JMS trimite explicit răspunsul la destinaţia replyTo specificată în mesajul de intrare. Dacă niciuna nu este specificată, destinaţia de trimitere este utilizată.

Atunci când este implementat exportul Generic JMS, un MDB (diferit de cel utilizat pentru importurile Generic JMS) este implementat. Acesta ascultă pentru cereri de intrare pe destinaţia de recepţionare şi apoi dispecerizează cererile pentru a fi procesate de runtime-ul SCA.

Anteturi speciale

Proprietăţile anteturilor speciale sunt utilizate în importurile şi exporturile Generic JMS pentru a informa legarea destinaţie cum să fie tratat mesajul.

De exemplu, proprietatea TargetFunctionName este utilizată de către selectorul de funcţii implicit pentru a identifica numele operaţiei din interfaţa export ce este invocată.

Notă: Legarea de import poate fi configurată pentru a seta antetul TargetFunctionName la numele de operaţie al fiecărei operaţii.

Resurse Java EE

Un număr de resurse Java EE sunt create atunci când o legare JMS este implementată într-un mediu Java EE.

- Portul ascultător pentru ascultarea pe destinaţia de recepţionare (răspuns) (doar două căi) pentru importuri şi pe destinaţia de recepţionare (cerere) pentru exporturi
- Fabrica de conexiuni JMS generic pentru outboundConnection (import) şi inboundConnection (export)
- Destinaţia JMS generic pentru destinaţiile de trimitere (import) şi recepţionare (export; doar două căi)
- Fabrica de conexiuni JMS generic pentru responseConnection (doar două căi şi opţional; altfel, outboundConnection este utilizat pentru importuri, iar inboundConnection este utilizat pentru exporturi)
- Destinaţia JMS generic pentru destinaţiile de recepţionare (import) şi trimitere (export) (doar două căi)
- Destinaţia JMS de callback a furnizorului de mesagerie implicit utilizată pentru a accesa destinaţia coadă de callback SIB (doar două căi)
- Fabrica de conexiuni JMS de callback a furnizorului de mesagerie implicit utilizată pentru a accesa destinaţia JMS de callback (doar două căi)
- Destinaţia coadă de callback SIB utilizată pentru a memora informaţiile despre mesajul de cerere pentru utilizarea în timpul procesării răspunsului (doar două căi)

Taskul de instalare creeazăConnectionFactory, cele trei destinaţii, şi ActivationSpec din informaţiile din fişierele de import şi export.

Anteturi JMS Generic:

Anteturile JMS Generic sunt SDO-uri (Service Data Objects) ce conţin toate proprietăţile proprietăţilor mesajului JMS Generic. Aceste proprietăţi pot fi de la mesajul de intrare sau pot fi proprietăţile ce vor fi aplicate mesajului de ieşire.

Un mesaj JMS conţine două tipuri de anteturi – antetul de sistem JSM şi mai multe proprietăţi JMS. Ambele tipuri de anteturi pot fi accesate fie într-un modul de mediere din SMO (Service Message Object), fie folosind API-ul ContextService.

Următoarele proprietăţi sunt setate static pe methodBinding:

- JMSType
- JMSCorrelationID
- JMSDeliveryMode
- JMSPriority

Legarea JMS Generic suportă modificarea dinamică a anteturilor şi proprietăţilor JMS în acelaşi mod ca legările JMS şi MQ JMS.

Unii furnizori de JMS Generic pun restricţii pe ce proprietăţi pot fi setate de aplicaţie şi în ce combinaţii. Trebuie să vă consultaţi documentaţia produsului de terţă parte pentru informaţii suplimentare. Totuşi, o proprietate suplimentară a fost adăugată la methodBinding, ignoreInvalidOutboundJMSProperties, ce permite să fie propagate orice excepţii.

Anteturile şi proprietăţile mesajelor JMS Generic sunt folosite doar când comutatorul de bază al legării SCDL a arhitecturii componentei de servicii este pornit. Când comutatorul este pornit, informaţiile de context sunt propagate. Implicit, comutatorul este pornit. Pentru a preveni propagarea informaţiilor de context, modificaţi valoarea în **fals**.

Când propagarea contextelor este activată, informaţiile de antet sunt permise să curgă către mesaj sau către componenta ţintă. Pentru a porni şi opri propagarea contextelor, specificaţi **adevărat** sau **fals** pentru atributul contextPropagationEnabled al legărilor de import şi export. De exemplu:

```
<esbBinding xsi:type="eis:JMSImportBinding" contextProgagationEnabled="true">
```

Valoarea implicită este **adevărat**.

Depanarea legăturilor JMS generic:

Puteţi diagnostica şi rezolva probleme cu legarea JMS generic.

Excepţii de implementare

Ca răspuns la diversele condiţii de eroare, implementarea Generic JMS import şi export poate returna una din cele două tipuri de excepţii:

- Service Business Exception: această excepţie este returnată dacă fault specificată pe interfaţa serviciului business (WSDL port type) a survenit.
- Service Runtime Exception: apărută în toate celelalte cazuri. În cele mai multe cazuri, excepţia cause va conţine excepţia (JMSException) originală.

Depanarea mesajului de expirare Generic JMS

Un mesaj de cerere de către furnizorul JMS este subiect de expirare.

Expirare cerere face referire la expirarea mesajului de cerere de către furnizorul JMS atunci când timpul JMSExpiration din mesajul de cerere este atins. Ca şi în cazul altor legări JMS, legarea Generic JMS manevrează expirarea prin setarea expirării la mesajul de apel invers plasat prin import pentru a fi asemănător cu cererea de ieşire. Notificarea expirării mesajului de apel invers va indica faptul că mesajul de cerere a expirat şi clientul ar trebui notificat printr-o excepţie operaţională.

Dacă destinaţia de apel invers este mutată pe un furnizor terţă parte, totuşi, acest tip de cerere de expirare nu este suportat.

Expirare răspuns face referire la expirarea mesajului de răspuns de către furnizorul JMS atunci când timpul JMSExpiration din mesajul de răspuns este atins.

Expirarea răspunsului pentru legarea generică JMS nu este suportat din cauza faptului că expirarea comportamentului exact al furnizorului JMS nu este definită. Totuşi, puteţi verifica dacă răspunsul este expirat atunci când este primit.

Pentru mesaje de cerere de ieşire, valoarea JMSExpiration va fi calculată în timpul aşteptat şi din valorile requestExpiration efectuate în asyncHeader, dacă este setat.

Depanarea erorilor fabricii de conexiuni Generic JMS

Atunci când definiţia anumite tipuri de fabrici de conexiuni în furnizorul dumneavoastră Generic JMS, s-ar putea să primiţi un mesaj de eroare atunci când încercaţi să porniţi aplicaţia. Puteţi modifica fabrica de conexiuni externă pentru a evita această problemă.

Atunci când lansaţi o aplicaţie, este posibil să primiţi următorul mesaj de eroare:

```
Tipul MDB Listener Port JMSConnectionFactory nu corespunde
tipului JMSDestination.
```

Această problemă poate apărea atunci când definiţi fabrici de conexiuni externe. Specific, excepţia poate fi aruncată atunci când creaţi o fabrică de conexiuni subiect JMS 1.0.2, în locul unei fabrici de conexiuni (unificate) JMS 1.1 (ce este una capabilă să suporte atât comunicarea punct-la-punct, cât şi cea de publicare/anonare).

Pentru a rezolva această problemă, realizaţi următorii paşi:

1. Accesaţi furnizorul Generic JMS pe care îl utilizaţi.
2. Înlocuiţi fabrica de conexiuni subiect JMS 1.0.2 pe care aţi definit-o cu fabrica de conexiuni (unificate) JMS 1.1.

Când lansaţi aplicaţia cu fabrica de conexiuni JMS 1.1 definită mai nou, ar trebui să nu mai primiţi un mesaj de eroare.

Mesajele SCA bazate pe Generic JMS nu apar în managerul de evenimente eşuate

Dacă mesajele SCA au originea într-o eşuare a interacţiunii JMS, v-aţi aştepta să găsiţi aceste mesaje în managerul de evenimente eşuate. Dacă asemenea mesaje nu apar în managerul de evenimente eşuate, asiguraţi-vă că valoarea proprietăţii de maxim de reîncercări din ascultătorul ce stă la bază este mai mare sau egală cu 1. Setarea aceste valori la 1 sau mai mult, permite interacţiunea cu managerul de evenimente eşuate în timpul invocării SCA pentru legările Generic JMS.

Tratarea excepţiilor:

Modul în care este configurată legarea determină cum sunt tratate excepţiile ridicate de handler-ele de date sau legările de date. În plus, natura fluxului de mediere dictează comportamentul sistemului când este aruncată o astfel de excepţie.

Poate apărea o varietate de probleme când un handler de date sau o legare de date este apelat(ă) de legarea dumneavoastră. De exemplu, este posibil ca handler-ul de date să primească un mesaj care are date utile corupte, sau este posibil să încerce să citească un mesaj care are un format incorect.

Modul în care legarea dumneavoastră tratează o astfel de excepţie este determinat de cum implementaţi handler-ul de date sau legarea de date. Comportamentul recomandat este să vă proiectaţi legarea de date să arunce o **DataBindingException**.

Situaţia este similară pentru un handler de date. Din moment ce handler-ul de date este invocat de legarea de date, orice excepţie a handler-ului de date este înfăşurată într-o excepţie a legării de date. Prin urmare o **DataHandlerException** vă este raportată ca o **DataBindingException**.

Când orice excepţie runtime, inclusiv o excepţie **DataBindingException**, este aruncată:

- Dacă fluxul de mediere este configurat să fie tranzacţional, mesajul JMS este memorat în Managerul de evenimente eşuate implicit pentru reluare sau ştergere manuală.

  Notă: Puteţi modifica modul de recuperare de pe legare, astfel încât mesajul să fie derulat înapoi în loc să fie stocat în managerul de evenimente eşuate.
- Dacă fluxul de mediere nu este tranzacţional, excepţia este înregistrată în istoric şi mesajul este pierdut.

Situaţia este similară pentru un handler de date. Deoarece handler-ul de date este apelat de legarea de date, o excepţie a handler-ului de date este produsă înăuntrul unei excepţii a legării de date. Prin urmare, o **DataHandlerException** vă este raportată ca o **DataBindingException**.

Legări JMS WebSphere MQ

Legarea WebSphere MQ JMS asigură integrarea în aplicaţiile externe care folosesc un furnizor bazat pe WebSphere MQ JMS.

Utilizaţi WebSphere MQ JMS legările de export şi import pentru a integra direct în sistemele JMS sau MQ JMS externe din mediul serverului dumneavoastră. Acest lucru elimină nevoia de a utiliza caracteristicile MQ Link sau Client Link ale Magistralei de Integrale a Serviciului.

Atunci când o componentă interacţionează cu un serviciu bazat pe WebSphere MQ JMS prin intermediul unui import, legarea de import WebSphere MQ JMS utilizează o destinaţie către care datele vor fi trimise şi o destinaţie unde va fi primit răspunsul. Conversia datelor către şi de la un mesaj JMS se realizează prin intermediul componentei edge de legare handler sau date pentru datele JMS.

Atunci când un modul SCA oferă un serviciu clienţilor WebSphere MQ JMS, legarea de export WebSphere MQ JMS utilizează o destinaţie la care vor fi recepţionate datele şi către care poate fi trimis răspunsul. Conversia datelor către şi de la un mesaj JMS se realizează prin intermediul handler-ului de date JMS sau prin legarea datelor.

Selectorul funcţiei asigură o mapare operaţiei în cadrul componentei ţintă care va fi invocată.

Privire generală asupra legărilor WebSphere MQ JMS:

Legarea WebSphere MQ JMS asigură integrarea în aplicaţiile externe care folosesc furnizorul WebSphere MQ JMS.

Taskuri administrative WebSphere MQ

Se aşteaptă ca administratorul de sistem pentru WebSphere MQ să creeze WebSphere MQ Queue Manager de bază; acesta va fi folosit de legările WebSphere MQ JMS înainte de a rula o aplicaţie care conţine aceste legări.

Legări de import WebSphere MQ JMS

WebSphere MQ JMS de import permit componentelor din modulul dumneavoastră SCA să comunice cu serviciile oferite de furnizorii bazaţi pe WebSphere MQ JMS. Trebuie să utilizaţi o versiune suportată de WebSphere MQ. Cerinţe detaliate despre hardware şi software se pot găsi pe paginile de suport IBM.

Se oferă suport pentru două tipuri de scenarii de utilizare pentru legările de import WebSphere MQ JMS, în funcţie de tipul de operaţie care este invocat:

- Primul tip: Importul WebSphere MQ JMS pune un mesaj în destinaţia de trimitere configurată în legarea de import. Nu se trimite nimic către câmpul replyTo din antetul JMS.
- Al doilea tip (cerere-răspuns): Importul WebSphere MQ JMS pune un mesaj în destinaţia de trimitere.

  Destinaţia receive este setată în câmpul replyTo din antet. Un MDB (message-driven bean) este implementat pentru a asculta la destinaţia receive (primire), iar în momentul în care este primit un răspuns, acesta transmite răspunsul înapoi către componentă.

  Legarea de import poate fi configurată (folosind câmpul **Schema de corelare a răspunsului** în Integration Designer), astfel încât să aştepte ID-ul de corelare al mesajului pentru răspuns ce a fost copiat din ID-ul mesajului de cerere (cel implicit) sau din ID-ul de corelare al mesajului de cerere.

Pentru ambele scenarii de utilizare pot fi specificate atât proprietăţile dinamice, cât şi cele statice ale antetului. Proprietăţile statice pot fi setate din legarea metodei de import JMS. Unele dintre aceste proprietăţi au semnificaţii speciale pentru runtime-ul JMS SCA.

Este important de menţionat faptul că WebSphere MQ JMS este o legare asincronă. Dacă o componentă apelantă invocă un import WebSphere MQ JMS în mod sincron (pentru o operaţie bidirecţională), atunci aceasta este blocată până când răspunsul este returnat de serviciul JMS.

Figura 37 la pagina 126 ilustrează modul în care importul este legat de serviciul extern.

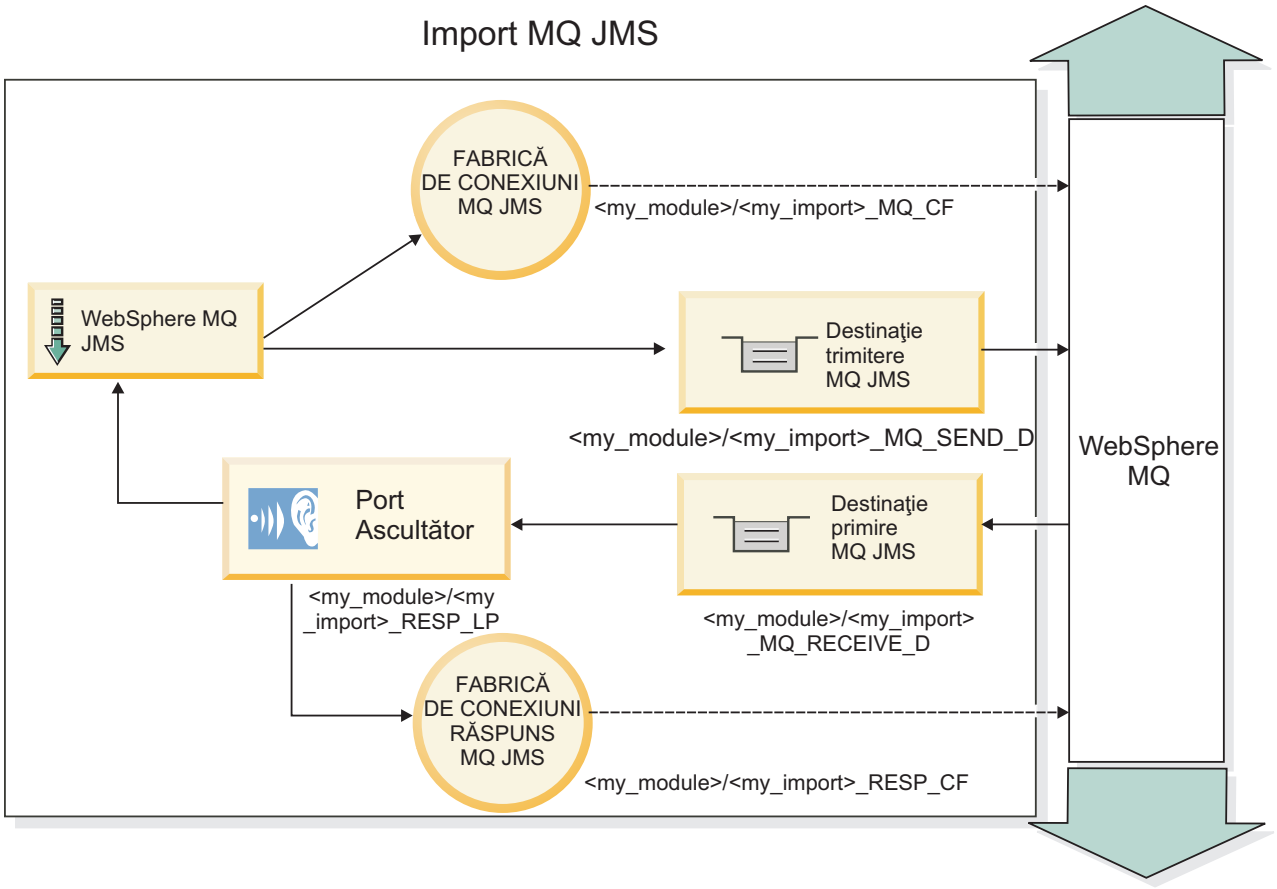


Figura 82. Resurse pentru legarea de import WebSphere MQ JMS

Legări pentru export WebSphere MQ JMS

Legarea de export WebSphere MQ JMS pune la dispoziţie mijloacele prin care modulele SCA asigură servicii către aplicaţiile externe bazate pe WebSphere MQ.

Este implementat un MDB pentru a asculta cererile ce intră prin destinaţia primire specificată în legarea de export. Destinaţia specificată în câmpul trimite este utilizat pentru a trimite răspunsul către cererea de intrare în cazul în care componenta invocată furnizează un răspuns. Destinaţia specificată în câmpul replyTo din mesajul de răspuns înlocuieşte destinaţia specificată în câmpul trimite.

Figura 38 la pagina 127 ilustrează modul în care solicitantul extern este legat de export.

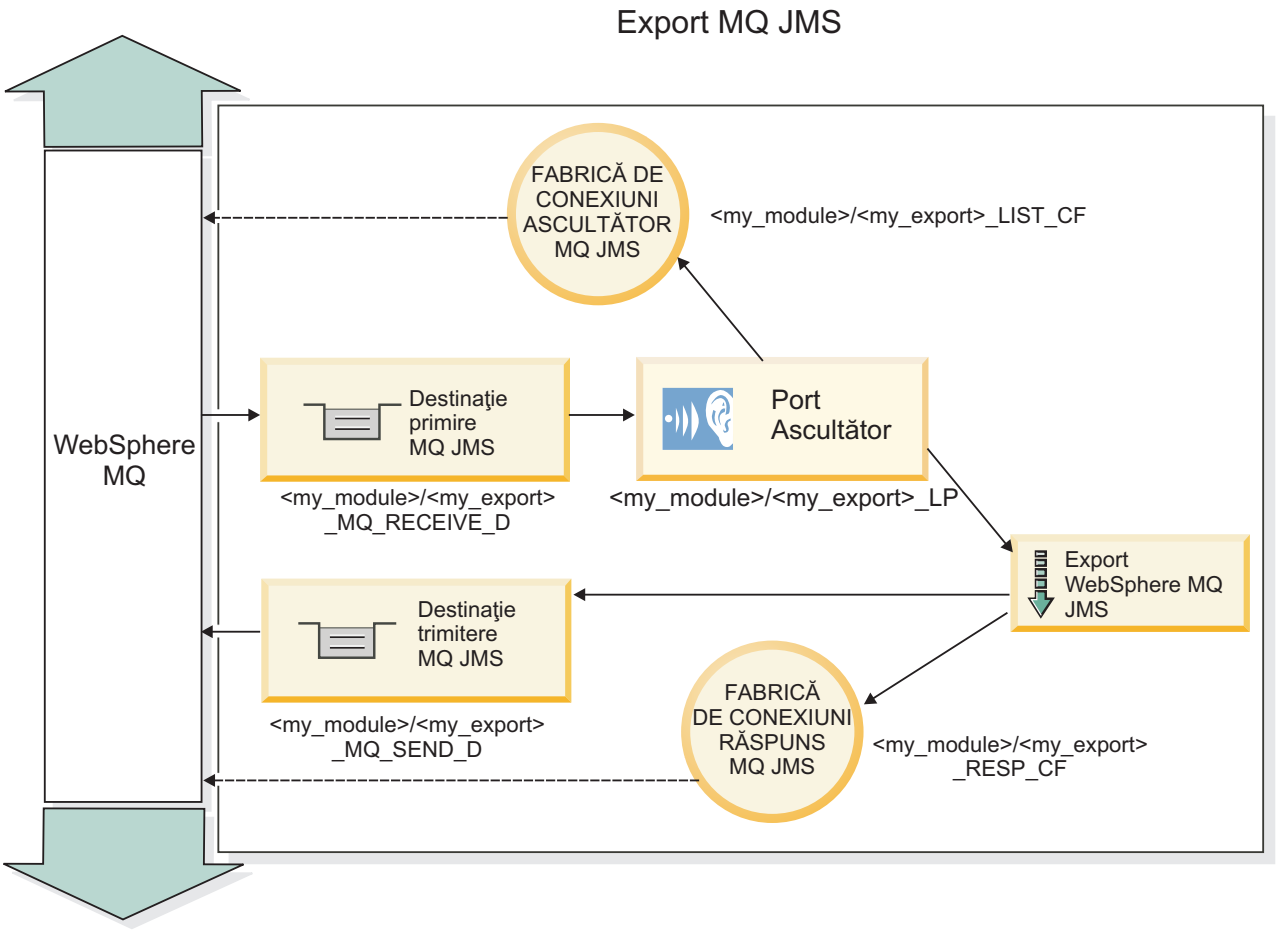


Figura 83. Resurse pentru legarea de export WebSphere MQ JMS

Notă: Figura 37 la pagina 126 şi Figura 38 la pagina 127 ilustrează modul în care o aplicaţie dintr-o versiune anterioară a IBM Business Process Manager este legată de un serviciu extern. Pentru aplicaţiile dezvoltate pentru IBM Business Process Manager Versiunea 7.0, se foloseşte Specificare activare în loc de Port ascultător şi Fabrică de conexiuni.

Caracteristici cheie ale legărilor WebSphere MQ JMS:

Caracteristicile cheie ale legărilor WebSphere MQ JMS includ anteturi, artefacte Java EE, şi resurse create Java EE.

Anteturile

Un antet de mesaj JMS conţine un număr de câmpuri predefinite ce conţin valori utilizate de şi de clienţi şi de furnizori pentru a identifica şi a ruta mesajele. Puteţi folosi proprietăţi de legătură pentru a configura aceste antete cu valori fixe, sau antetele pot fi specificate dinamic la rulare.

JMSCorrelationID

Legări la un mesaj înrudit. În mod tipic, acest câmp este setat la şirul identificatorului mesajului căuria îi este dat un răspuns.

TargetFunctionName

Acest antet este utilizat de unul din selectorii de funcţii furnizaţi pentru a identifica operaţia ce este invocată. Setarea proprietăţii de antet JMS TargetFunctionName în mesajele trimise la un export JMS permite acestui selector de funcţii să fie utilizat. Proprietatea poate fi setată direct în aplicaţiile clientului JMS sau la conectarea unui import cu o legare JMS la un export. În acest caz, legarea de import JMS ar trebui configurată pentru a seta antetul TargetFunctionName pentru fiecare operaţie din interfaţă la numele acesteia.

Scheme de corelare

Legările JMS WebSphere MQ furnizează scheme de corelare variate ce sunt utilizate pentru a determina cum vor fi corelate mesajele de crere cu mesajele de răspuns.

RequestMsgIDToCorrelID
JMSMessageID este copiat la câmpul JMSCorrelationID . Aceasta este setarea implicită.

RequestCorrelIDToCorrelID
JMSCorrelationID este copiat la câmpul JMSCorrelationID .

Resurse Java EE

Un număr de resurse Java EE sunt create atunci când un import MQ JMS este implementat la un mediu Java EE.

Parametri

Fabrica de conexiuni MQ
Utilizată de clienţi pentru a crea o conexiune la furnizorul MQ JMS.

Fabrica de conexiuni răspuns
Folosit de runtime-ul SCA MQ JMS atunci când destinaţia de trimitere este pe un Manager coadă diferit de cel al destinaţiei de primire.

Specificaţii de activare
O specificaţie de activare MQ JMS este asociată cu una sau mai multe bean-uri bazate pe mesaje şi furnizează configuraţia necesară pentru a recepţiona mesajele.

Destinaţii

- Trimitere destinaţie:
  - Importuri: Unde sunt trimise mesajele de cerere sau de ieşire.
  - Exporturi: Unde vor fi trimise mesajele de răspuns dacă nu este înlocuit de câmpul de antet JMSReplyTo al mesajului de intrare.
- Destinaţie primire:
  - Importuri: Unde vor fi plasate mesajele de intrare şi de răspuns.
  - Exporturi: Unde vor fi plasate mesajele de intrare şi de cerere.

Anteturi JMS:

Un mesaj JMS conţine două tipuri de anteturi- antetul sistemului JMS şi mai multe proprietăţi JMS. Ambele tipuri de anteturi pot fi accesate fie într-un modul de mediere din SMO (Service Message Object) fie folosind API-ul ContextService.

Antetul sistemului JMS

Antetul sistemului JMS este reprezentat în SMO prin elementul JMSHeader care conţine toate câmpurile care se găsesc de obicei într-un antet JMS. Deşi acestea pot fi modificate în modul de mediere (sau ContextService), unele câmpuri din antetul sistemului JMS setate în SMO nu vor fi trimise în mesajul JMS de ieşire pe măsură ce acestea sunt înlocuite de sistem sau valori statice.

Câmpurile cheie din antetul sistemului JMS care pot fi actualizate într-un modul mediere (sau ContextService) sunt:

- **JMSType** şi **JMSCorrelationID** – valorile proprietăţilor specifice antetului mesajului predefinit
- **JMSDeliveryMode** – valori pentru modul de livrare (persistent sau nepersistent; valoarea implicită este persistent)
- **JMSPriority** – valoare prioritate (de la 0 la 9; valoarea implicită este JMS_Default_Priority)

Proprietăţi JMS

Proprietăţile JMS sunt reprezentate în SMO sub formă de intrări în lista Proprietăţi. Proprietăţile pot fi adăugate, actualizate sau şterse într-o mediere sau folosind API-ul ContextService.

De asemenea, proprietăţile pot fi setate în legarea JMS. Proprietăţile care sunt setate în mod static înlocuiesc setările (cu acelaşi nume) care sunt setate în mod dinamic.

Proprietăţile utilizatorului propagate din alte legări (de exemplu, o legare HTTP) va fi pusă în legarea JMS sub formă de proprietăţi JMS.

Setările propagării anteturilor

Propagarea antetului şi proprietăţilor sistemului JMS fie de la mesajul JMS de intrare către componentele următoare, fie de la componentele anterioare către mesajul JMS de ieşire poate fi controlată prin steguleţul Propagate Protocol Header din legare.

Atunci când Propagate Protocol Header este setat, informaţiilor de antet le este permis să circule către mesaj sau către componenta ţintă, aşa cum este descris în următoarea listă:

- Cerere de export JMS

  Antetul JMS primit în mesaj va fi propagat către o componentă ţintă prin intermediul serviciului de context. Proprietăţile JMS primite în mesaj vor fi propagate către o componentă ţintă prin intermediul serviciului de context.
- Răspuns la exportul JMS

  Oricare dintre câmpurile din antetul JMS din serviciul de contextul va fi utilizat în mesajul de ieşire, dacă nu este înlocuită de proprietăţile statice stabilite în legarea de export JMS. Oricare dintre proprietăţile setate în serviciul de context va fi utilizată în mesajul de ieşire, dacă nu este înlocuită de proprietăţile statice stabilite în legarea de export JMS.
- Cerere de import JMS

  Oricare dintre câmpurile din antetul JMS din serviciul de context va fi utilizat în mesajul de ieşire, dacă nu este înlocuit de proprietăţile statice stabilite în legarea de import JMS. Oricare dintre proprietăţile setate în serviciul de context va fi utilizată în mesajul de ieşire, dacă nu este înlocuită de proprietăţile statice stabilite în legarea de import JMS.
- Răspuns import JMS

  Antetul JMS primit în mesaj va fi propagat către o componentă ţintă prin intermediul serviciului de context. Proprietăţile JMS primite în mesaj vor fi propagate către o componentă ţintă prin intermediul serviciului de context.

Clienţi externi:

Serverul poate trimite mesaje către, sau să primească mesaje de la, clienţi externi folosind legări JMS WebSphere MQ.

Un client extern (de exemplu, un portal web sau un sistem de informaţii pentru întreprindere) poate trimite un mesaj către o componentă SCA din aplicaţie prin intermediul unui export sau poate fi invocat de o componentă SCA din aplicaţie prin intermediul unui import.

Legarea de export WebSphere MQ JMS implementează MDB-uri (bean-uri controlate de mesaj) cu scopul de a asculta cererile ce intră prin destinaţia receive specificată în legarea de export. Destinaţia specificată în câmpul send este folosită pentru a trimite răspunsul către cererea de intrare în cazul în care aplicaţia invocată oferă un răspuns. Cu toate acestea, un client extern poate invoca aplicaţii prin intermediul legării de export.

WebSphere MQ JMS importă legarea în, şi poate trimite un mesaj către, clienţii externi. Acest mesaj ar putea sau nu să solicite un răspuns de la clientul extern.

Informaţii suplimentare despre modul de interacţionare cu clienţii externi folosind WebSphere MQ pot fi găsite în Centrul de informare WebSphere MQ.

Depanarea legărilor WebSphere MQ JMS:

Puteţi diagnostica şi rezolva probleme cu legările WebSphere MQ JMS.

Excepţii de implementare

Ca răspuns la diferite condiţii de eroare, implementarea MQ JMS import şi export poate returna unul din cele doua tipuri de excepţii:

- Service Business Exception: această excepţie este returnată dacă fault specificată pe interfaţa serviciului business (WSDL port type) a survenit.
- Service Runtime Exception: apărută în toate celelalte cazuri. În cele mai multe cazuri, cause exception va conţine excepţia originală (JMSException).

  De exemplu, un import aşteaptă numai un mesaj de răspuns pentru fiecare mesaj de cerere. Dacă mai mult de un răspuns soseşte, sau dacă un răspuns întârziat (unul pentru care expirarea răspunsului SCA are loc) soseşte, o excepţie Service Runtime Exception este aruncată. Tranzacţia este repetată, şi mesajul de răspuns este retras din coadă sau este manevrat de către managerul de eveniment.

Mesajele SCA bazate pe WebSphere MQ JMS care nu apar în managerul de evenimente eşuate.

Dacă mesajele SCA îşi au originea într-un eşec de interacţiune WebSphere MQ JMS, ar trebui să vă aşteptaţi să găsiţi aceste mesaje în managerul de evenimente eşuate. Dacă astfel de mesaje nu apar în managerul de evenimente eşuate, asiguraţi-vă că valoarea proprietăţii maximului de reîncercări de pe portul ascultător de bază este egală sau mai mare decât **1**. Setarea acestei valori la **1** sau mai mult activează interacţiunea cu managerul de evenimente eşuate în timpul invocărilor SCA pentru legările MQ JMS.

Scenarii de folosire greşită: comparaţie cu legările WebSphere MQ

Legările WebSphere MQ JMS sunt proiectate să interacţioneze cu aplicaţiile JMS implementate în comparaţie cu WebSphere MQ, care expune mesajele în concordanţă cu modelul de mesaje JMS. Totuşi, importul şi exportul WebSphere MQ , sunt în principal proiectate să interacţioneze cu aplicaţiile native WebSphere MQ şi să expună întregul conţinut al corpului mesajului WebSphere MQ la mediere.

Următoarele scenarii ar trebui construite folosind legarea WebSphere MQ JMS, şi nu legarea WebSphere MQ:

- Invocarea bean-ului JMS controlat de mesaj (MDB) dintr-un modul SCA, unde MDB este implementat împotriva furnizorului WebSphere MQ JMS. Se utilizează un import WebSphere MQ JMS.
- Permisiunea modulului SCA de a fi apelat dintr-un servlet componentă Java EE sau EJB prin intermediul căii JMS. Se va folosi un export WebSphere MQ JMS.
- Medierea conţinutului unui JMS MapMessage, la traversare de-a lungul WebSphere MQ. Se va folosi un export sau import WebSphere MQ JMS împreună cu handler-ul de date sau legarea de date potrivită.

Există situaţii în care legarea WebSphere MQ şi legarea WebSphere MQ JMS se aşteaptă să interacţioneze. În particular, când se face o punte între aplicaţii Java EE şi non-Java EE WebSphere MQ, se va folosi export WebSphere MQ şi import WebSphere MQ JMS (sau invers) în concordanţă cu legările de date potrivite sau modulele de mediere potrivite (sau ambele).

Tratarea excepţiilor:

Modul în care este configurată legarea determină cum sunt tratate excepţiile ridicate de handler-ele de date sau legările de date. În plus, natura fluxului de mediere dictează comportamentul sistemului când este aruncată o astfel de excepţie.

Poate apărea o varietate de probleme când un handler de date sau o legare de date este apelat(ă) de legarea dumneavoastră. De exemplu, este posibil ca handler-ul de date să primească un mesaj care are date utile corupte, sau este posibil să încerce să citească un mesaj care are un format incorect.

Modul în care legarea dumneavoastră tratează o astfel de excepţie este determinat de cum implementaţi handler-ul de date sau legarea de date. Comportamentul recomandat este să vă proiectaţi legarea de date să arunce o **DataBindingException**.

Situaţia este similară pentru un handler de date. Din moment ce handler-ul de date este invocat de legarea de date, orice excepţie a handler-ului de date este înfăşurată într-o excepţie a legării de date. Prin urmare o **DataHandlerException** vă este raportată ca o **DataBindingException**.

Când orice excepţie runtime, inclusiv o excepţie **DataBindingException**, este aruncată:

- Dacă fluxul de mediere este configurat să fie tranzacţional, mesajul JMS este memorat în Managerul de evenimente eşuate implicit pentru reluare sau ştergere manuală.

  Notă: Puteţi modifica modul de recuperare de pe legare, astfel încât mesajul să fie derulat înapoi în loc să fie stocat în managerul de evenimente eşuate.
- Dacă fluxul de mediere nu este tranzacţional, excepţia este înregistrată în istoric şi mesajul este pierdut.

Situaţia este similară pentru un handler de date. Deoarece handler-ul de date este apelat de legarea de date, o excepţie a handler-ului de date este produsă înăuntrul unei excepţii a legării de date. Prin urmare, o **DataHandlerException** vă este raportată ca o **DataBindingException**.

Legări WebSphere MQ

Legarea WebSphere MQ asigură conectivitatea Service Component Architecture (SCA) cu aplicaţiile WebSphere MQ.

Utilizaţi legările de export şi de import WebSphere MQ pentru integrare directă cu un sistem bazat pe WebSphere MQ din mediul dumneavoastră de server. Acest lucru elimină nevoia de a utiliza caracteristicile MQ Link sau Client Link ale Magistralei de Integrale a Serviciului.

Atunci când o componentă interacţionează cu un serviciu WebSphere MQ prin intermediul unui import, legarea de import WebSphere MQ utilizează o coadă către care vor fi trimise datele şi o coadă unde poate fi primit răspunsul.

Atunci când un modul SCA oferă un serviciu clienţilor WebSphere MQ, legarea de export WebSphere MQ foloseşte o coadă în care pot fi primite cererile şi către care pot fi trimise răspunsurile. Selectorul funcţiei asigură o mapare operaţiei în cadrul componentei ţintă care va fi invocată.

Conversia datelor de sarcină utilă către şi de la un mesaj MQ se realizează prin intermediul handler-ului de date al corpului MQ sau prin legarea datelor. Conversia datelor din antet către şi de la un mesaj MQ se realizează prin intermediul legării datelor din antetul MQ.

Pentru informaţii despre versiunile WebSphere MQ suportate, vedeţi pagina web cu cerinţe detaliate de sistem.

Privire generală asupra legărilor WebSphere MQ:

Legarea WebSphere MQ asigură integrarea în aplicaţiile native bazate pe MQ.

Taskuri administrative WebSphere MQ

Se aşteaptă ca administratorul de sistem al WebSphere MQ să creeze WebSphere MQ Queue Manager de bază; acesta va fi folosit de legările WebSphere MQ înainte de a rula o aplicaţie care conţine aceste legări.

Taskuri administrative WebSphere

Trebuie să setaţi proprietatea **Cale bibliotecă nativă** din adaptorul pentru resurse MQ în WebSphere cu versiunea WebSphere MQ suportată de server şi să reporniţi serverul. Acest lucru asigură faptul că sunt utilizate bibliotecile unei versiuni acceptate a WebSphere MQ. Cerinţe detaliate despre hardware şi software se pot găsi pe Paginile de suport IBM .

Legări de import WebSphere MQ

Legarea de import WebSphere MQ permite componentelor din modulul dumneavoastră SCA să comunice cu serviciile oferite de aplicaţiile externe bazate pe WebSphere MQ. Trebuie să utilizaţi o versiune suportată de WebSphere MQ. Cerinţe detaliate despre hardware şi software se pot găsi pe Paginile de suport IBM .

Interacţiunea cu sistemele WebSphere MQ externe include utilizarea cozilor pentru trimiterea cererilor sau primirea răspunsurilor.

Se oferă suport pentru două tipuri de scenarii de utilizare pentru legările de import WebSphere MQ, în funcţie de tipul de operaţie invocat:

- Într-o direcţie: Importul WebSphere MQ pune un mesaj pe coada configurată în câmpul **Coadă destinaţie trimitere** al legării de import. Nu se trimite nimic către câmpul replyTo din antetul MQMD.
- În ambele sensuri (cerere-răspuns): Importul WebSphere MQ pune un mesaj în coada configurată în câmpul **Coadă destinaţie de trimitere**

  Coada de primire este setată în câmpul replyTo din antetul MQMD. Un MDB (bean controlat de mesaj) este implementat pentru a asculta la coada de primire, iar în momentul în care este primit un răspuns, acesta transmite răspunsul înapoi către componentă.

  Legarea de import poate fi configurată (folosind câmpul **Schema de corelare a răspunsului**), astfel încât să aştepte ID-ul de corelare al mesajului pentru răspuns ce a fost copiat din ID-ul mesajului de cerere (cel implicit) sau din ID-ul de corelare al mesajului de cerere.

Este important de menţionat faptul că WebSphere MQ este o legare asincronă. Dacă o componentă apelantă invocă un import în mod sincron WebSphere MQ (pentru o operaţie bidirecţională), atunci aceasta este blocată până când răspunsul este returnat de serviciul WebSphere MQ.

Figura 39 la pagina 132 ilustrează modul în care importul este legat de serviciul extern.

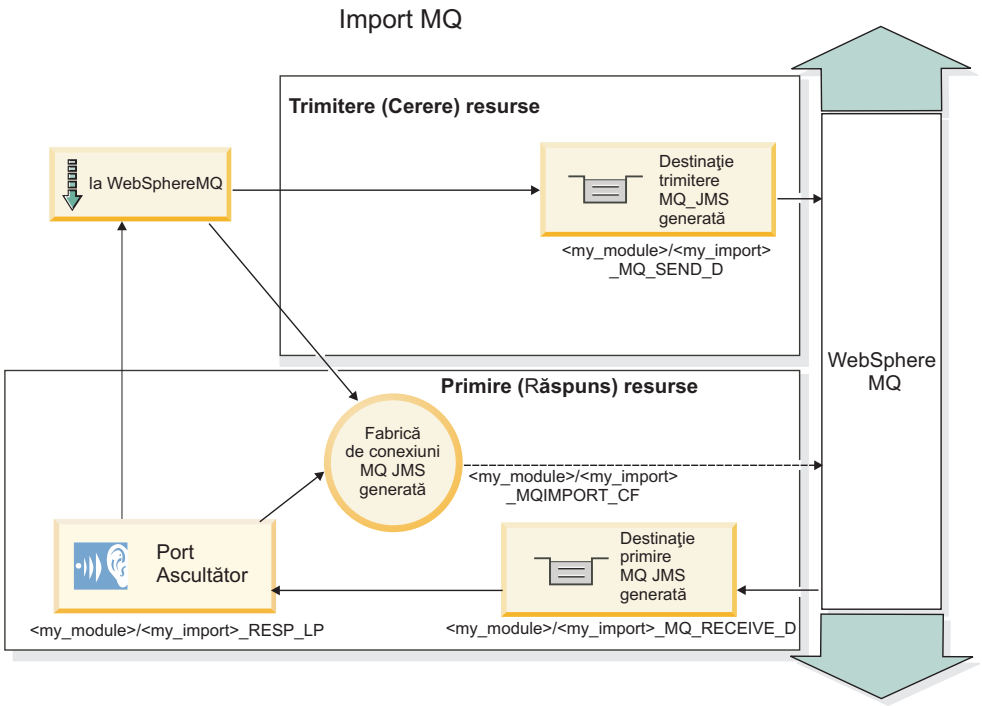


Figura 84. Resurse pentru legarea de import WebSphere MQ

Legări pentru export WebSphere MQ

Legarea de export WebSphere MQ pune la dispoziţie mijloacele prin care modulele SCA asigură servicii către aplicaţiile externe bazate pe WebSphere MQ.

Este implementat un MDB pentru a asculta cererile de intrare în **Coadă destinaţii de primire** specificată în legarea de export. Coada specificată în câmpul **Coadă destinaţii de trimitere** este folosit pentru a trimite răspunsul către cererea de intrare în cazul în care componenta invocată oferă un răspuns. Coada specificată în câmpul replyTo din mesajul de răspuns înlocuieşte coada specificată în câmpul **Coadă destinaţii de trimitere** field.

Figura 40 la pagina 133 ilustrează modul în care solicitantul extern este legat de export.

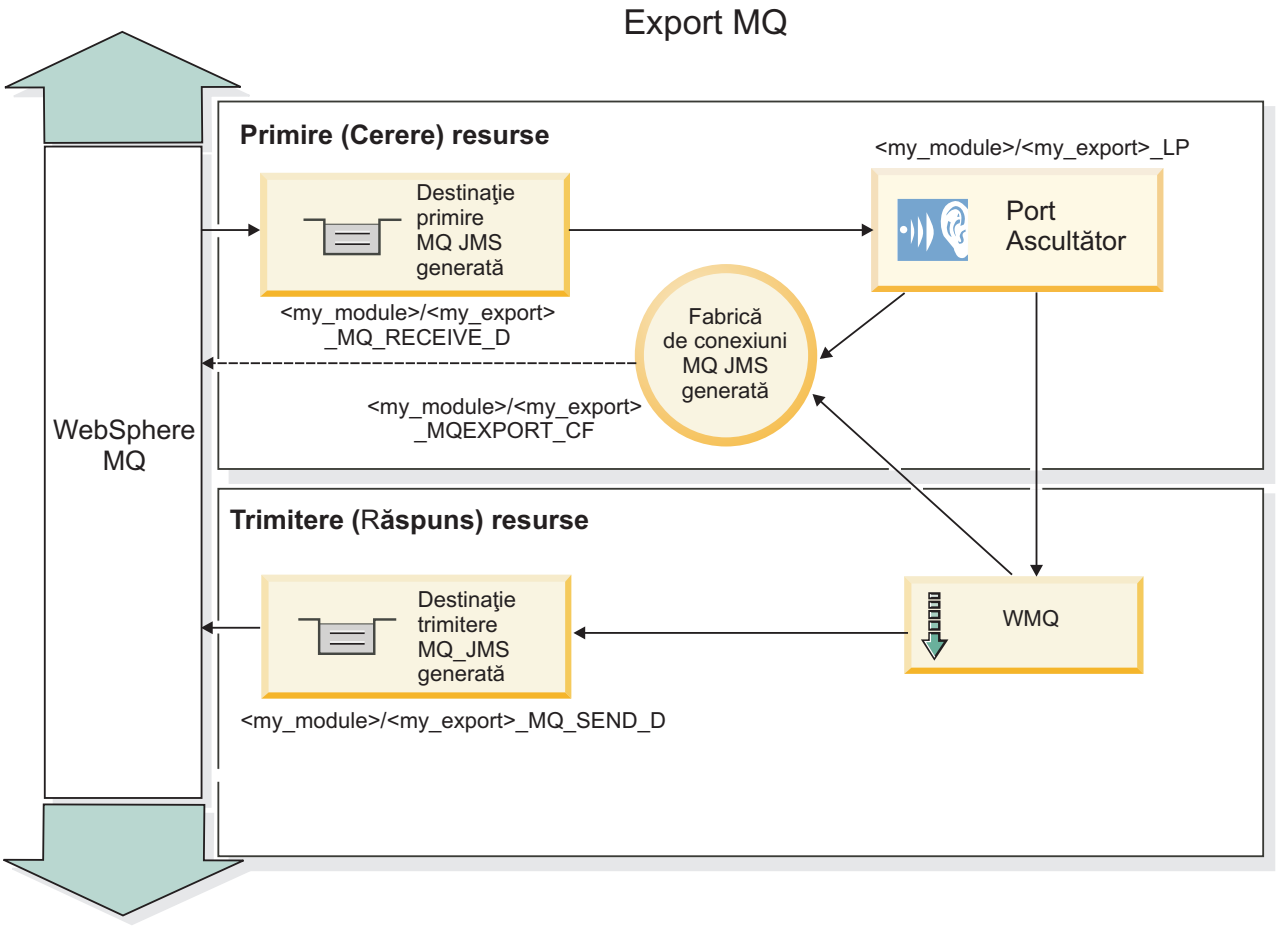


Figura 85. Resurse pentru legarea de export WebSphere MQ

Notă: Figura 39 la pagina 132 şi Figura 40 la pagina 133 ilustrează modul în care o aplicaţie dintr-o versiune anterioară a IBM Business Process Manager este legată de un serviciu extern. Pentru aplicaţiile dezvoltate pentru IBM Business Process Manager Versiunea 7, se foloseşte Specificare activare în loc de Port ascultător şi Fabrică de conexiuni.

Caracteristici cheie ale unei legări WebSphere MQ:

Caracteristicile cheie ale unei legări WebSphere MQ includ anteturi, artefacte Java EE, şi resurse create Java EE.

Scheme de corelare

O aplicaţie de cerere/răspuns WebSphere MQ poate utiliza un număr de tehnici pentru a corela mesajele de răspuns cu cererile, construite în jurul câmpurilor MQMD MessageID şi CorrelID. În marea majoritate a cazurilor, solicitantul lasă managerul de cozi să selecteze un MessageID şi aşteaptă ca aplicaţia de răspuns să copieze aceasta în CorrelID al răspunsului. În majoritatea cazurilor, solicitantul şi aplicaţia de răspuns ştiu implicit ce tehnică de corelare este în uz. Ocazional aplicaţia de răspuns va onora diverse steguleţe în câmpul Report al cererii ce descriu cum vor fi tratate aceste câmpuri.

Legările de export pentru mesajele WebSphere MQ pot fi configurate cu următoarele opţiuni:

Opţiuni răspuns MsgId:

MsgID nou
Permite managerului de coadă să selecteze un MsgId unic pentru răspuns (implicit).

Copiere de la MsgID cerere
Copiază câmpul MsgId de la câmpul MsgId din cerere.

Copiere de la mesaj SCA
Setează MsgId pentru a fi transportat în anteturile WebSphere MQ din mesajul de răspuns SCA, sau permite managerului de coadă să definească un nou Id dacă valoarea nu există.

Ca opţiuni de raport
Inspectează câmpul Raport al MQMD din cerere pentru o sugestie despre cum să fie tratat MsgId. Opţiunile MQRO_NEW_MSG_ID şi MQRO_PASS_MSG_ID sunt suportate şi se comportă ca New MsgId şi Copy from Request MsgID.

Opţiuni CorrelId răspuns:

Copy from Request MsgID
Copiază câmpul CorrelId de la câmpul MsgId din cerere (implicit).

Copy from Request CorrelID
Copiază câmpul CorrelId de la câmpul CorrelId din cerere.

Copy from SCA message
Setează CorrelId pentru a fi transportat în anteturile WebSphere MQ din mesajele de răspuns SCA sau îl lasă gol dacă valoarea nu există.

As Report Options
Inspectează câmpul Report al MQMD din cerere pentru o sugestie despre cum să fie tratat CorrelId. Opţiunile MQRO_COPY_MSG_ID_TO_CORREL_ID şi MQRO_PASS_CORREL_ID sunt suportate şi se comportă ca Copy fromRequest MsgID şi Copy from Request CorrelID.

Legările de import pentru mesajele WebSphere MQ pot fi configurate cu următoarele opţiuni:

Opţiuni MsgId cerere:

New MsgID
Permite managerului cozii să selecteze un MsgId unic pentru cerere (implicit).

Copy from SCA message
Setează MsgId pentru a fi transportat în anteturile WebSphere MQ din mesajul de cerere SCA sau permite managerului de coadă să definească un nou Id dacă valoarea nu există.

Opţiuni corelare răspuns:

Response has CorrelID copied from MsgId
Se aşteaptă ca mesajul de răspuns să aibă un câmp CorrelId setat, per MsgId al cererii (implicit).

Response has MsgID copied from MsgId
Se aşteaptă ca mesajul de răspuns să aibă un câmp MsgId setat, per MsgId al cererii.

Response has CorrelID copied from CorrelId
Se aşteaptă ca mesajul de răspuns să aibă un câmp CorrelId setat, per CorrelId al cererii.

Resurse Java EE

Un număr de resurse Java EE sunt create atunci când o legare WebSphere MQ este implementată într-un mediu Java EE.

Parametri

Fabrica de conexiuni MQ
Utilizată de clienţi pentru a crea o conexiune la furnizorul WebSphere MQ.

Fabrica de conexiuni răspuns
Utilizată de runtime-ul SCA MQ atunci când destinaţia de trimitere este pe un Manager de coadă diferit ca destinaţia de recepţionare.

Specificaţii de activare
O specificaţie de activare MQ JMS este asociată cu una sau mai multe bean-uri bazate pe mesaje şi furnizează configuraţia necesară pentru a recepţiona mesajele.

Destinaţii
- Destinaţie de trimitere: unde cererea sau mesajul de ieşire este trimis (import); unde mesajul de răspuns va fi trimis (export), dacă nu este înlocuit de câmpul de antet MQMD ReplyTo din mesajul de intrare.
- Destinaţie de primire: unde răspunsul/cererea sau mesajul de intrare ar trebui plasat.

Anteturi WebSphere MQ:

Anteturile WebSphere MQ încorporează anumite convenţii pentru conversie la mesaje SCA (Service Component Architecture).

Mesajele WebSphere MQ conţin un antet sistem (MQMD), zero sau mai multe alte anteturi MQ (sistem sau personalizate) şi un corp mesaj. Dacă există mai multe anteturi în mesaj, ordinea anteturilor este semnificativă.

Fiecare antet conţine informaţii care descriu structura antetului următor. MQMD-ul descrie primul antet.

Cum sunt parsate anteturile MQ

O legare de date de Antet MQ este utilizată pentru a parsa anteturi MQ. Următoarele anteturi sunt suportate automat:
- MQRFH
- MQRFH2
- MQCIH
- MQIIH

Anteturile care încep cu **MQH** sunt manipulate diferit. Anumite câmpuri ale antetului nu sunt parsate; ele rămân ca octeţi neparsaţi.

Pentru alte anteturi MQ, puteţi scrie legări de date antet MQ pentru a parsa acele anteturi.

Cum sunt accesate anteturile MQ

Anteturile MQ pot fi accesate din produs într-unul din două moduri:
- Prin SMO (Service Message Object) într-o mediere
- Prin API-ul ContextService

Anteturile MQ sunt reprezentate intern cu elementul SMO MQHeader. MQHeader este un container de date antet care extinde MQControl dar conţine un element valoare de orice tip. Conţine MQMD, MQControl (informaţii de control corp mesaj MQ) şi o listă de alte anteturi MQ.

- MQMD reprezintă conţinuturile descrierii mesajului WebSphere MQ, cu excepţia informaţiilor care determină structura şi codarea corpului.
- MQControl conţine informaţii care determină structura şi codarea unui corp de mesaj.
- MQHeaders conţine o listă de obiecte MQHeader.

Lanţul antet MQ este desfăcut astfel încât, în interiorul SMO, fiecare antet MQ are propriile sale informaţii de control (CCSID, Codare şi Format). Anteturile pot fi adăugate sau şterse cu uşurinţă, fără a modifica alte date ale antetului.

Setarea câmpurilor în MQMD

Puteţi actualiza MQMD-ul utilizând API-ul Context sau prin SMO (Service Message Object) într-o mediere. Câmpurile următoare sunt propagate automat la mesajul MQ de ieşire:

- Encoding
- CodedCharacterSet
- Format
- Report
- Expiry
- Feedback
- Priority
- Persistence
- CorrelId
- MsgFlags

Configuraţi legarea MQ pe un Import sau Export pentru a propaga următoarele proprietăţi la mesajul MQ de ieşire:

MsgID
Setaţi **ID mesaj de cerere** la copiere din mesaj SCA.

MsgType
Curăţaţi caseta de bifare **Setare tip mesaj la MQMT_DATAGRAM sau MQMT_REQUEST pentru operaţii cerere-răspuns**.

ReplyToQ
Curăţaţi caseta de bifare **Înlocuire răspuns la coadă de mesaje de cerere**.

ReplyToQMgr
Curăţaţi caseta de bifare **Înlocuire răspuns la coadă de mesaje de cerere**.

De la versiunea 7.0 înainte, câmpurile context pot fi înlocuite utilizând o proprietate personalizată de pe definiţia destinaţiei JNDI. Setaţi proprietatea personalizată MDCTX cu valoarea SET_IDENTITY_CONTEXT pe destinaţia de trimitere pentru a propaga câmpurile următoare la mesajul MQ de ieşire:

- UserIdentifier
- AppIdentityData

Setaţi proprietatea personalizată MDCTX cu valoarea SET_ALL_CONTEXT pe destinaţia de trimitere pentru a propaga proprietăţile următoare la mesajul MQ de ieşire:

- UserIdentifier
- AppIdentityData
- PutApplType
- PutApplName

- ApplOriginData

Unele câmpuri nu sunt la mesajul MQ de ieşire. Câmpurile următoare sunt înlocuite în timpul trimiterii mesajului:
- BackoutCount
- AccountingToken
- PutDate
- PutTime
- Offset
- OriginalLength

Adăugarea statică a MQCIH într-o îmbinare WebSphere MQ:

IBM Business Process Manager suportă adăugarea informaţiilor de antet MQCIH static, fără folosirea unui modul de mediere.

Există diferite moduri de a adăuga informaţii de antet MQCIH unui mesaj (de exemplu, folosind primitiva de mediere Header Setter). Poate fi util să adăugaţi aceste informaţii de antet static, fără folosirea unui modul de mediere suplimentar. Informaţii de antet statice, inclusiv numele de program CICS, ID-ul tranzacţiei şi alte detalii de antet de format de date, pot fi definite ca şi parte a legării WebSphere MQ.

WebSphere MQ, MQ CICS Bridge şi CICS trebuie să fie configurat pentru ca informaţiile de antet MQCIH să fie adăugate static.

Puteţi folosi Integration Designer pentru a configura importul WebSphere MQ cu valori statice care sunt necesare pentru informaţiile de antet MQCIH.

Atunci când un mesaj ajunge şi este procesat de către importul WebSphere MQ, este efectuată o verificare pentru vedeţi dacă informaţia de antet MQCIH este deja prezentă în mesaj. Dacă MQCIH este prezent, valorile statice definite în importul WebSphere MQ sunt folosite pentru a înlocui valorile dinamice corespunzătoare din mesaj. Dacă MQCIH nu este prezent, este creat unul în mesaj şi valorile statice definite în importul WebSphere MQ sunt adăugate.

Valorile statice definite în importul WebSphere MQ sunt specifice unei metode. Puteţi specifica valori statice MQCIH diferite pentru diferite metode din acelaşi import WebSphere MQ.

Această facilitate nu este folosită pentru a furniza valori implicite dacă MQCIH nu conţine informaţii de antet specifice deoarece o valoare statică definită în importul WebSphere MQ va înlocui o valoare corespunzătoare furnizată în mesajul de intrare.

Clienţi externi:

IBM Business Process Manager poate trimite mesaje către, sau să primească mesaje de la, clienţi externi folosind legări WebSphere MQ.

Un client extern (de exemplu, un portal web sau un sistem de informaţii pentru întreprindere) poate trimite un mesaj către o componentă SCA din aplicaţie prin intermediul unui export sau poate fi invocat de o componentă SCA din aplicaţie prin intermediul unui import.

Legarea de export WebSphere MQ implementează MDB-uri (bean-uri controlate de mesaj) cu scopul de a asculta cererile ce intră prin destinaţia receive specificată în legarea de export. Destinaţia specificată în câmpul trimite este folosită pentru a trimite răspunsul către cererea de intrare în cazul în care aplicaţia invocată oferă un răspuns. Cu toate acestea, un client extern poate invoca aplicaţii prin intermediul legării de export.

WebSphere MQ importă legarea la, şi poate trimite un mesaj către, clienţii externi. Acest mesaj ar putea sau nu să solicite un răspuns de la clientul extern.

Informaţii suplimentare despre modul de interacţionare cu clienţii externi folosind WebSphere MQ pot fi găsite în Centrul de informare WebSphere MQ.

Depanarea legărilor WebSphere MQ:

Puteţi diagnostica şi rezolva condiţiile de eşec ce apar în legările WebSphere MQ.

Condiţii de eşec primar

Condiţiile de eşec primar ale legărilor WebSphere MQ sunt determinate de către sensurile tranzacţionale, de către configuraţia WebSphere MQ sau prin referire la comportamente existente în alte componente. Condiţiile de eşec primar includ:

- Eşec în conexiunea la managerul de coadă sau la coada WebSphere MQ.

  Un eşec la conectarea la WebSphere MQ pentru a primi mesaje va avea ca rezultat eşecul în pornirea portului MDB Listener. Această condiţie va fi jurnalizată în istoricul WebSphere Application Server. Mesaje persistente vor rămâne pe coada WebSphere MQ până în momentul în care sunt extrase cu succes (sau expirate de către WebSphere MQ).

  Un eşec la conectarea la WebSphere MQ pentru a trimite mesaje de ieşire va duce la derularea înapoi a operaţiei de control a trimiterii.
- Eşecul în analiza unui mesaj de intrare sau în construirea unui mesaj de ieşire.

  Un eşec la legarea de date sau la handler-ul de date cauzează derularea înapoi a tranzacţiei care controlează lucrul.
- Eşec la trimiterea mesajului de ieşire.

  Un eşec la trimiterea unui mesaj cauzează derularea înapoi a tranzacţiei relevante.
- Mesaje de răspuns multiple sau neaşteptate.

  Importul aşteaptă numai un mesaj de răspuns pentru fiecare mesaj de cerere. Dacă mai mult de un răspuns soseşte, sau dacă un răspuns întârziat (unul pentru care expirarea răspunsului SCA are loc) soseşte, este aruncată o excepţie Service Runtime. Tranzacţia este repetată, şi mesajul de răspuns este retras din coadă sau este manevrat de către managerul de eveniment.

Utilizarea greşită a scenariilor: comparaţie cu legările WebSphere MQ JMS

WebSphere MQ import şi export sunt în principal proiectate să interacţioneze cu aplicaţiile native WebSphere MQ şi să expună întregul conţinut al corpului mesajului WebSphere MQ la medieri. Legările WebSphere MQ JMS sunt proiectate să interacţioneze cu aplicaţiile JMS implementate în comparaţie cu WebSphere MQ, care expune mesajele în concordanţă cu modelul de mesaje JMS.

Următoarele scenarii ar trebui construite folosind legarea WebSphere MQ JMS, şi nu legarea WebSphere MQ:

- Invocarea bean-ului JMS controlat de mesaj (MDB) dintr-un modul SCA, unde MDB este implementat împotriva furnizorului WebSphere MQ JMS. Se utilizează un import WebSphere MQ JMS.
- Permisiunea modulului SCA de a fi apelat dintr-un servlet componentă Java EE sau EJB prin intermediul căii JMS. Utilizaţi un export WebSphere MQ JMS.
- Medierea conţinutului unui JMS MapMessage, la traversare de-a lungul WebSphere MQ. Utilizaţi un export şi import WebSphere MQ JMS împreună cu legarea de date potrivită.

Există situaţii în care legarea WebSphere MQ şi legarea WebSphere MQ JMS se aşteaptă să interacţioneze. În particular, când se face o punte între aplicaţii Java EE şi non-Java EE WebSphere MQ, utilizaţi export WebSphere MQ şi import WebSphere MQ JMS (sau invers) în concordanţă cu legările de date corespunzătoare sau modulele de mediere potrivite (sau ambele).

Mesaje netrimise

Dacă WebSphere MQ nu poate livra un mesaj la destinaţia sa intenţionată (de exemplu, din cauza erorilor de configuraţie), trimite mesaje în schimb la o coadă de scrisori moarte.

Pentru realizarea acestui lucru, are nevoie de un antet cu mesaj nelivrat la începutul corpului mesajului. Acest antet conţine motivele de eşec, destinaţia originală şi alte informaţii.

Mesajele SCA bazate pe MQ nu apar în managerul de evenimente eşuate

Dacă mesajele SCA îşi au originea într-un eşec de interacţiune WebSphere, ar trebui să vă aşteptaţi să găsiţi aceste mesaje în managerul de evenimente eşuate. Dacă astfel de mesaje nu apar în managerul de evenimente eşuate, asiguraţi-vă că destinaţia WebSphere MQ ce stă la bază are valoarea maximă de livrări eşuate mai mare decât 1. Setarea acestei valori la 2 sau mai mult permite interacţiunea cu managerul de evenimente eşuate în timpul invocărilor SCA pentru legările WebSphere MQ.

Evenimentele eşuate MQ sunt reluate pe managerul de coadă greşit

Atunci când o fabrică de conexiuni predefinită trebuie să fie utilizată pentru conexiuni de ieşire, proprietăţile de conexiune trebuie să se potrivească cu acelea definite în specificaţia de activare utilizată pentru conexiunile de intrare.

Fabrica de conexiuni predefinită este utilizată pentru a crea o conexiune la reluarea unui eveniment eşuat şi, prin urmare, trebuie configurată pentru a utiliza acelaşi manager de coadă ca cel din care a fost primit iniţial mesajul.

Tratarea excepţiilor:

Modul în care este configurată legarea determină cum sunt tratate excepţiile ridicate de handler-ele de date sau legările de date. În plus, natura fluxului de mediere dictează comportamentul sistemului când este aruncată o astfel de excepţie.

Poate apărea o varietate de probleme când un handler de date sau o legare de date este apelat(ă) de legarea dumneavoastră. De exemplu, este posibil ca handler-ul de date să primească un mesaj care are date utile corupte, sau este posibil să încerce să citească un mesaj care are un format incorect.

Modul în care legarea dumneavoastră tratează o astfel de excepţie este determinat de cum implementaţi handler-ul de date sau legarea de date. Comportamentul recomandat este să vă proiectaţi legarea de date să arunce o **DataBindingException**.

Situaţia este similară pentru un handler de date. Din moment ce handler-ul de date este invocat de legarea de date, orice excepţie a handler-ului de date este înfăşurată într-o excepţie a legării de date. Prin urmare o **DataHandlerException** vă este raportată ca o **DataBindingException**.

Când orice excepţie runtime, inclusiv o excepţie **DataBindingException**, este aruncată:

- Dacă fluxul de mediere este configurat să fie tranzacţional, mesajul JMS este memorat în Managerul de evenimente eşuate implicit pentru reluare sau ştergere manuală.

  Notă: Puteţi modifica modul de recuperare de pe legare, astfel încât mesajul să fie derulat înapoi în loc să fie stocat în managerul de evenimente eşuate.
- Dacă fluxul de mediere nu este tranzacţional, excepţia este înregistrată în istoric şi mesajul este pierdut.

Situaţia este similară pentru un handler de date. Deoarece handler-ul de date este apelat de legarea de date, o excepţie a handler-ului de date este produsă înăuntrul unei excepţii a legării de date. Prin urmare, o **DataHandlerException** vă este raportată ca o **DataBindingException**.

Limitări ale legărilor

Legările au unele limitări în utilizarea lor, care sunt menţionate aici.

Limitările legării MQ:

Legarea MQ are unele limitări în utilizarea sa. Acestea sunt menţionate aici.

Distribuţia mesajelor din abonamentele nepublicate

Metoda de abonare fără publicare a mesajelor distribuite nu este suportată în prezent de legarea MQ prin intermediul propriului WMQ. Totuşi, legarea MQ JMS nu suportă această metodă de distribuţie.

Cozi de primire partajate

Mai multe legări de export şi import WebSphere MQ aşteaptă ca orice mesaj prezent în coada lor configurată pentru recepţie să fie intenţionate pentru acel export sau import. Legările pentru importuri şi exporturi ar trebui să fie configurate ţinând cont de următoarele considerente:

- Fiecare import MQ trebuie să aibă o coadă diferită de recepţionare deoarece legarea de import MQ presupune că toate mesajele din coada de recepţionare sunt răspunsuri la cererile pe care le trimite. În cazul în care coada de recepţionare este partajată între mai multe importuri, răspunsurile ar putea fi recepţionate de un import greşit şi nu vor putea fi corelate cu mesajul de cerere iniţial.
- Fiecare export MQ ar trebui să aibă o coadă diferită de recepţionare, deoarece altfel nu se poate anticipa care dintre exporturi va primi oricare dintre mesajele de cerere individuale.
- Importurile şi exporturile MQ pot face referire către aceiaşi coadă de trimitere.

Limitările legărilor JMS, MQ JMS şi JMS generice:

Legările JMS şi MQ JMS au unele limitări.

Implicaţiile generării legărilor implicite

Limitările utilizării legărilor JMS, MQ JMS şi JMS sunt discutate în următoarele secţiuni:

- Implicaţiile generării legărilor implicite
- Schema de corelare a răspunsurilor
- Suport bidirecţional

Atunci când generaţi o legare, vor fi completate în mod implicit mai multe câmpuri în cazul în care nu alegeţi să introduceţi singur valorile. De exemplu, numele unui fabrici de conexiuni va fi creat pentru dvs. Dacă ştiţi că veţi pune aplicaţia dumneavoastră pe un server şi o veţi accesa de la distanţă cu un client, ar trebui ca în momentul creării legării să introduceţi numele JNDI în locul celor implicite, deoarece probabil veţi dori să controlaţi aceste valori prin intermediul consolei administrative în momentul rulării.

Totuşi, dacă aţi acceptat valorile implicite, iar apoi aţi aflat mai târziu că nu vă puteţi accesa aplicaţia de la un client aflat la distanţă, aveţi posibilitatea să utilizaţi consola administrativă pentru a seta în mod explicit valoarea fabricii de conexiuni. Localizaţi câmpul pentru punctele finale care aparţin de furnizor în setările fabricii de conexiuni şi adăugaţi valoarea în forma <nume_server>:7276 (dacă se utilizează numărul de port implicit).

Schema de corelare a răspunsurilor

În cazul în care folosiţi schema de corelare a răspunsurilor CorrelationId To CorrelationId, care este folosită pentru a corela mesaje într-o operaţie de tip răspuns-răspuns, trebuie să aveţi un Id de corelare dinamic în mesaj.

Pentru a crea un ID de corelare dinamic într-un modul de mediere folosind editorul de fluxul de mediere, adăugaţi o primitivă de mediere Mapping înainte de importul cu legarea JMS. Deschideţi editorul de mapare. Anteturile arhitecturii componentei serviciului cunoscut vor fi disponibile în mesajul ţintă. Trageţi un câmp care conţine un ID unic în mesajul sursă în ID-ul de corelare din antetul JMS din mesajul ţintă.

Suportul bidirecţional

Doar caracterele ASCII sunt acceptate pentru numele JNDI (Java Naming and Directory Interface) la momentul execuţiei.

Cozi de primire partajate

Mai multe legări de export şi import aşteaptă ca orice mesaj prezent în coada lor configurată pentru recepţie să fie intenţionate pentru acel export sau import. Legările pentru importuri şi exporturi ar trebui să fie configurate ţinând cont de următoarele considerente:

- Fiecare legare pentru import trebuie să aibă o coadă diferită de recepţionare deoarece aceasta presupune că toate mesajele din coada de recepţionare sunt răspunsuri la cererile pe care le trimite. În cazul în care coada de recepţionare este partajată între mai multe importuri, răspunsurile ar putea fi recepţionate de un import greşit şi nu vor putea fi corelate cu mesajul de cerere iniţial.
- Fiecare legare de export ar trebui să aibă o coadă diferită de primire, deoarece altfel nu se poate anticipa care dintre exporturi va primi oricare dintre mesajele de cerere individuale.
- Importurile şi exporturile pot face referire la aceeaşi coadă de trimitere.

Obiectele business

Industria software-ului pentru computer a dezvoltat mai multe modele de programare şi cadre de lucru în care *obiectele business* furnizează o reprezentare naturală a datelor operaţionale pentru procesarea aplicaţiei.

În general, aceste obiecte business:

- Sunt definite utilizând standarde de industrie
- Mapează transparent date la tabele de baze de date sau sisteme de informaţii de întreprindere
- Suportă protocoale de invocare la distanţă
- Furnizează fundaţia modelului de programare de date pentru programare aplicaţie

Process Designer şi Integration Designer le furnizează dezvoltatorilor un astfel de model de obiect business comun pentru reprezentarea diferitelor tipuri de entităţi de afaceri din diferite domenii. În timpul dezvoltării, acest model permite dezvoltatorilor să definească obiecte business ca definiţii de scheme XML.

În timpul rulării, datele operaţionale definite de definiţiile de scheme XML sunt reprezentate ca obiecte business Java. În acest model, obiectele business sunt bazate pe ciorne anterioare ale specificaţiei SDO (Service Data Object) şi furnizează setul complet de interfeţe aplicaţie model de programare necesare pentru manipularea datelor operaţionale.

Definirea obiectelor business

Definiţi obiecte business utilizând editorul de obiecte business din Integration Designer. Editorul de obiecte business memorează obiectele business ca definiţii schemă XML.

Utilizarea schemei XML pentru a defini obiecte business furnizează mai multe avantaje:

- Schema XML furnizează un model de definiţie de date bazat pe standarde şi o fundaţie pentru interoperabilitatea dintre aplicaţii şi sisteme eterogene incompatibile. Schemele XML sunt utilizate împreună cu WSDL (Web Services Description Language) pentru a furniza contracte de interfaţă bazate pe standarde de-a lungul componentelor, aplicaţiilor şi sistemelor.
- Schemele XML definesc un model de definiţie date bogat pentru reprezentarea datelor operaţionale. Acest model include tipuri complexe, tipuri simple, tipuri definite de utilizator, moştenire tip şi cardinalitate printre alte caracteristici.
- Obiectele business pot fi definite de interfeţe operaţionale şi date definite din Web Services Description Language, precum şi de schema XML din organizaţii de standarde industriale sau din alte sisteme şi aplicaţii. Integration Designer poate importa aceste obiecte business direct.

Integration Designer furnizează de asemenea suport pentru descoperirea datelor operaţionale din baze de date şi sisteme de informaţii de întreprindere şi generarea definiţiei obiectului business al schemei XML bazată pe standarde a acelor date operaţionale. Obiectele business generate în acest fel sunt adesea adresate ca *obiecte business specifice aplicaţiei* deoarece imită structura datelor operaţionale definite în sistemul de informaţii de întreprindere.

Când un proces manipulează date din mai multe sisteme de informaţii diferite, poate fi preţios să transformaţi reprezentarea disparată a datelor operaţionale (de exemplu, CustomerEIS1 şi CustomerEIS2 sau OrderEIS1 şi OrderEIS2) într-o singură reprezentare canonică (de exemplu, Customer sau Order). Reprezentarea canonică este adesea adresată ca *obiectul business generic*.

Definiţiile de obiecte business, în special pentru obiecte business generice, sunt utilizate frecvent de mai multe aplicaţii. Pentru a susţine această reutilizare, Integration Designer permite ca obiectele business să fie create în biblioteci care pot fi apoi asociate cu module de aplicaţii multiple.

Web Services Description Language (WSDL) defineşte contractele pentru serviciile furnizate şi consumate de un modul de aplicaţii SCA (Service Component Architecture) precum şi contractele utilizate pentru a crea componentele dintr-un modul de aplicaţii. Într-un contract, un WSDL poate reprezenta atât operaţii cât şi obiecte operaţionale (care sunt definite de scheme XML pentru a reprezenta datele operaţionale).

Lucrul cu obiecte business

SCA (Service Component Architecture) furnizează cadrul de lucru pentru definirea unui modul de aplicaţii, serviciile pe care le furnizează, serviciile pe care le consumă şi alcătuirea componentelor care furnizează logica operaţională a modulului de aplicaţii. Obiectele business joacă un rol important în aplicaţie, definind datele operaţionale care sunt utilizate pentru a descrie contractele componentei şi datele operaţionale pe care le manipulează componentele.

Diagrama următoare descrie un modul de aplicaţii SCA şi ilustrează multe dintre locurile în care dezvoltatorul lucrează cu obiecte business.

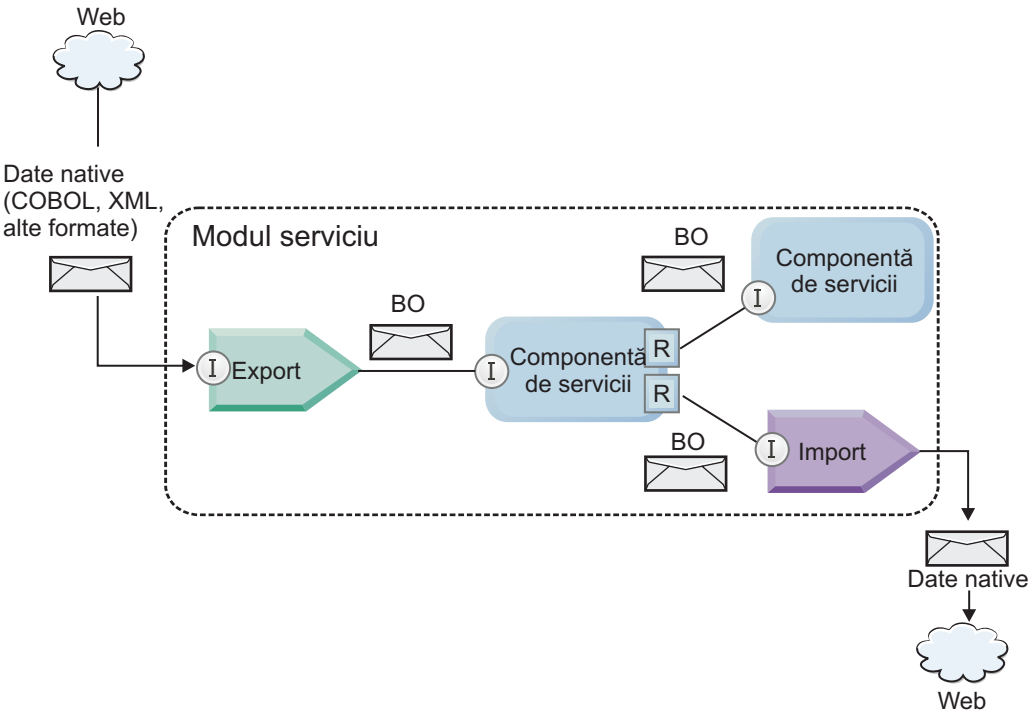


Figura 86. Obiectele business reprezintă datele care curg între servicii într-o aplicaţie

Notă: Acest subiect descrie cum sunt utilizate obiectele business de către modulele de aplicaţii SCA. Dacă utilizaţi interfeţe Java, modulele de aplicaţii SCA pot procesa de asemenea obiecte Java.

Model de programare obiect business

Modelul de programare obiect business conţine un set de interfeţe Java care reprezintă:

- Definiţia obiectului business şi date instanţă

- Un set de servicii care suportă operaţiile de pe obiectele business

Definiţiile tipului de obiect business sunt reprezentate de interfeţele commonj.sdo.Type şi commonj.sdo.Property. Modelul de programare obiect business furnizează un set de reguli pentru maparea informaţiilor de tip complex ale schemei XML la interfaţa Tip şi fiecare dintre elementele din definiţia tipului complex la interfaţa Proprietate.

Instanţele obiectului business sunt reprezentate de interfaţa commonj.sdo.DataObject. Modelul de programare obiect business nu are tip, ceea ce înseamnă că aceeaşi interfaţă commonj.sdo.DataObject poate fi utilizată pentru a reprezenta diferite definiţii de obiect business, cum ar fi Client sau Comandă. Definiţia căror proprietăţi pot fi setate şi extrase din fiecare obiect business este determinată de informaţiile de tip definite în schema XML asociată cu fiecare obiect business.

Comportamentul modelului de programare obiect business este bazat pe specificaţia Service Data Object 2.1. Pentru informaţii suplimentare, consultaţi SDO 2.1 pentru specificaţiile Java, îndrumare şi javadocs pe web: http://osoa.org/display/Main/Service+Data+Objects+Specifications.

Serviciile obiectului business suportă diverse operaţii de ciclu de viaţă (cum ar fi creare, egalitate, parsare şi serializare) pe obiecte business.

Pentru lucruri specifice despre modelul de programare al obiectului business, vedeţi Programare utilizând servicii de obiecte business şi Documentaţie API şi SPI generată despre obiecte business.

Legări, legări de date şi handler-e de date

După cum a fost afişat în Figura 41 la pagina 142, datele operaţionale care sunt utilizate pentru a invoca servicii furnizate de module de aplicaţii SCA sunt transformate în obiecte business, astfel încât componentele SCA să poate manipula datele operaţionale. În mod similar, obiectele business manipulate de componente SCA sunt convertite în formatul de date cerut de serviciile externe.

În unele cazuri, cum ar fi legarea de servicii Web, legarea utilizată pentru a exporta şi importa servicii transformă automat datele în formatul corespunzător. În alte cazuri, cum ar fi legarea JMS, dezvoltatorii pot furniza o legare de date sau handler de date care converteşte formatele nenative în obiecte business reprezentate de interfaţa DataObject.

Pentru informaţii suplimentare despre dezvoltarea legărilor de date şi handler-elor de date, consultaţi “Handler-e de date” la pagina 55 şi “Legări de date” la pagina 56.

Componente

Componentele SCA definesc contractele lor de servicii de provizionare şi consum utilizând o combinaţie de Web Services Description Language şi schemă XML. Datele operaţionale pe care SCA le transmite între componente sunt reprezentate ca obiecte business utilizând interfaţa DataObject. SCA verifică dacă aceste tipuri de obiecte business sunt compatibile cu contractul interfeţei definit de componenta care va fi invocată.

Teoriile modelului de programare pentru manipularea obiectelor business variază de la componentă la componentă. Componenta POJO şi primitiva personalizată a componentei flux de mediere furnizează manipulare directă a obiectelor business prin activarea programării Java direct utilizând interfeţele şi serviciile de programare obiect business. Majoritatea componentelor furnizează teorii de nivel superior pentru manipularea obiectelor business, dar furnizează de asemenea fragmente de cod Java pentru definirea comportamentului personalizat din interfeţele şi serviciile obiectului business.

Obiectele business pot fi transformate utilizând fie combinaţia componentei Interface Flow Mediation şi Business Object Map, fie componenta flux de mediere şi primitiva sa XML Map. Aceste capabilităţi de transformare obiecte business sunt utile pentru convertirea obiectelor business specifice aplicaţiei la şi de la obiecte business generice.

Obiecte business speciale

Obiectele de mesaje de servicii şi graficele operaţionale sunt două tipuri specializate de obiecte business care sunt utilizate în scopuri de aplicaţii specifice.

Obiect mesaj serviciu

Un SMO (Service Message Object) este un obiect business specializat care este utilizat de componentele fluxului de mediere pentru a reprezenta colecţia de date asociată cu o invocare de servicii.

Un SMO are o structură de nivel înalt corectată alcătuită din anteturi, context, corp şi ataşamente (dacă sunt prezente).

- Anteturile transmit informaţii înrudite cu invocarea de servicii peste un anumit protocol sau legare. Exemple sunt anteturile SOAP şi anteturile JMS.
- Datele de context transmit informaţii logice suplimentare asociate cu invocarea în timp ce sunt procesate de componenta fluxului de mediere. Aceste informaţii nu fac parte în mod tipic din datele de aplicaţie trimise sau recepţionate de clienţi.
- Corpul SMO-ului transmite datele operaţionale ale datelor utile, care reprezintă mesajul de aplicaţie nucleu sau datele de invocare sub forma unui obiect business standard.

SMO poate transporta de asemenea date în ataşament pentru invocările serviciilor web folosind SOAP cu ataşamente.

Fluxurile de mediere realizează astfel de taskuri precum dirijare cerere şi transformare de date, şi SMO-ul furnizează vizualizarea combinată de conţinut antet şi date utile (payload) într-o singură structură unificată.

Grafic operaţional

Un grafic operaţional este un obiect business special utilizat pentru a furniza suport pentru sincronizare date în scenarii de integrare.

Consideraţi un exemplu în care două sisteme de informaţii de întreprindere au o reprezentare a unei anumite comenzi. Când se modifică comanda dintr-un sistem, poate fi trimis un mesaj la celălalt sistem pentru a sincroniza datele comenzii. Graficele operaţionale suportă noţiunea de trimitere doar a porţiunii de comandă care s-a modificat la celălalt sistem şi adnotarea ei cu informaţii de modificare sumar pentru a defini tipul de modificare.

În acest exemplu, un Grafic operaţional de comenzi va transmite celuilalt sistem că unul dintre articolele de linie din comandă a fost şters şi că proprietatea dată de livrare proiectată a comenzii a fost actualizată.

Graficele operaţionale pot fi adăugate cu uşurinţă la obiecte business existente din Integration Designer. Sunt găsite de cele mai multe ori în scenarii în care sunt utilizate adaptoarele WebSphere şi pentru a ajuta migrarea aplicaţiilor WebSphere InterChange Server.

Mod de parsare obiect business

Integration Designer furnizează o proprietate pe module şi biblioteci pe care le puteţi vedea pentru a configura modul de parsare XML pentru obiecte business fie la vioi, fie la leneş.

- Dacă opţiunea este setată la *vioi*, fluxurile de octeţi XML sunt parsate cu nerăbdare pentru a crea obiectul business.
- Dacă opţiunea este setată la *leneş*, obiectul business este creat normal, dar parsarea reală a fluxului de octeţi XML este amânată şi parsată parţial doar când sunt accesate proprietăţile obiectului business.

În oricare dintre modurile de parsare XML, datele non-XML sunt întotdeauna parsate vioi pentru a crea obiectul business.

Considerente la alegerea modului de parsare a obiectului business

Modul de parsare a obiectului business determină modul în care sunt parsate datele XML în timpul rulării. Un mod de parsare de obiect business este definit pe un modul sau o bibliotecă atunci când este creat(ă). Puteţi modifica modul de parsare pentru modul sau bibliotecă, totuşi ar trebui să fiţi conştient de implicaţii.

Modul de parsare pentru obiectul business este setat la nivelul modulului sau al bibliotecii. Modulele care au fost create într-o versiune de IBM Integration Designer anterioară versiunii 7 vor rula în modul de parsare rapidă fără modificări necesare. Implicit, modulele şi bibliotecile care sunt create în IBM Integration Designer versiunea 7 şi ulterioare vor primi cel mai adecvat mod de parsare în funcţie de numărul de factori, cum ar fi modul de parsare al proiectelor existente în spaţiul dumneavoastră de lucru sau modul de parsare al proiectelor dependente sau al altor proiecte din aceeaşi soluţie şi aşa mai departe. Puteţi modifica modul de parsare de obiect business al unui mod sau bibliotecă pentru a se potrivi cu implementarea dumneavoastră, totuşi ar trebui să fiţi conştienţi de următoarele considerente.

Considerente

- Modul de parsare obiect business leneş procesează datele XML mai repede; totuşi există diferenţe de compatibilitate între modul vioi şi modul leneş de care trebuie să fiţi conştient înainte de modificarea configuraţiei unui modul sau a unei biblioteci. Aceste diferenţe vor afecta comportamentul din timpul rulării modulelor. Pentru informaţii despre modul optim de parsare pentru aplicaţia dumneavoastră, vedeţi "Beneficii ale utilizării modului de parsare lent versus modul de parsare vioi" în legăturile înrudite.
- Un modul poate fi configurat doar pentru rularea într-un mod de parsare. Bibliotecile pot fi configurate fie pentru suportul nodurilor de parsare, fie a ambelor moduri de parsare. O bibliotecă configurată pentru suportul ambelor moduri de parsare ar putea fi referită de ambele module, utilizând modul de parsare vioi, şi de un modul utilizând modul de parsare leneş. Modul de parsare al unei biblioteci la momentul rulării este determinat de modulele care fac referire la bibliotecă. La momentul execuţiei, un modul îşi declară modul de parsare, iar acel mod de parsare este utilizat de modul şi de orice bibliotecă folosită de modul.
- Modulele şi bibliotecile care sunt configurate pentru moduri de parsare diferite sunt compatibile în următoarele cazuri:
  - Modulele şi bibliotecile configurate cu modul de parsare leneş sunt compatibile cu bibliotecile care utilizează fie modul de parsare leneş, fie ambele moduri de parsare, leneş şi vioi.
  - Modulele şi bibliotecile configurate cu modul de parsare vioi sunt compatibile cu bibliotecile care utilizează fie modul de parsare vioi, fie ambele moduri de parsare, leneş şi vioi.
  - Bibliotecile configurate cu modurile de parsare leneş şi vioi sunt compatibile doar cu bibliotecile care utilizează ambele moduri de parsare, leneş şi vioi.
- Utilizaţi acelaşi mod de parsare pentru modulele interactive care comunică utilizând legarea SCA. Dacă modulele comunică utilizând diferite moduri de parsare, ar putea rezulta probleme de performanţă.

Concepte înrudite:

“Beneficii ale utilizării modului de parsare leneş faţă de cel vioi” la pagina 145
Unele aplicaţii beneficiază de modul leneş de parsare XML în timp ce altele văd o performanţă îmbunătăţită cu mod de parsarea vioi. Este recomandat să măsuraţi aplicaţia dumneavoastră în ambele moduri de parsare pentru a determina ce mod se potriveşte cel mai bine caracteristicilor specifice ale aplicaţiei dumneavoastră.

Beneficii ale utilizării modului de parsare leneş faţă de cel vioi

Unele aplicaţii beneficiază de modul leneş de parsare XML în timp ce altele văd o performanţă îmbunătăţită cu mod de parsarea vioi. Este recomandat să măsuraţi aplicaţia dumneavoastră în ambele moduri de parsare pentru a determina ce mod se potriveşte cel mai bine caracteristicilor specifice ale aplicaţiei dumneavoastră.

Este posibil ca aplicaţiile care parsează fluxuri de date XML mari să vadă îmbunătăţiri de performanţă când este utilizat modul de parsare XML leneş. Beneficiile performanţei cresc pe măsură ce dimensiunea şirului de octeţi XML creşteţi cantitatea de date de la şirul de octeţi care este accesat de către aplicaţie scade.

Este posibil ca aplicaţiile următoare să funcţioneze mai bine folosind un mod de parsarea vioi:

- Aplicaţii care parsează fluxuri de date non-XML
- Aplicaţii care folosesc mesajele care sunt create folosind serviciul BOFactory
- Aplicaţii care parsează mesaje XML foarte mici

Referinţe înrudite:

“Considerente la alegerea modului de parsare a obiectului business” la pagina 144
Modul de parsare a obiectului business determină modul în care sunt parsate datele XML în timpul rulării. Un mod de parsare de obiect business este definit pe un modul sau o bibliotecă atunci când este creat(ă). Puteţi modifica modul de

parsare pentru modul sau bibliotecă, totuşi ar trebui să fiţi conştient de implicaţii.

Consideraţii de migrare şi de dezvoltare de aplicaţii

Dacă configuraţi o aplicaţie care a fost dezvoltată iniţial utilizând un mod de parasare vioi pentru a utiliza acum un mod de parsare leneş, sau dacă plănuiţi să comutaţi o aplicaţie între modul de parsare leneş şi cel vioi, fiţi conştient de diferenţele dintre moduri şi de considerente când comutaţi moduri.

Tratarea erorilor

Dacă fluxul de octeţi XML care este parsat este format greşit, apar excepţii de parsare.

- În modul de parsare XML vioi, acele excepţii apar de îndată ce este parsat obiectul business din fluxul XML de intrare.
- Dacă este configurat modul de parsare XML leneş, excepţiile de parsare apar cu întârziere când sunt accesate proprietăţile obiectului business şi este parsată porţiunea XML-ulu icare este format greşit.

Pentru a vă ocupa de XML format greşit, selectaţi una dintre următoarele opţiuni:

- Implementarea unei magistrale de servicii de întreprindere pe margini pentru a valida XML de intrare
- Logica de detecţie eroare autor în punctul în care sunt accesate proprietăţile de obiect business

Mesaje şi stive de excepţii

Deoarece modurile de parsare XML leneş şi vioi au implementări fundamentale diferite, urmele de stivă aruncate de interfeţele şi serviciile de programare ale obiectului business au acelaşi nume de clasă de excepţii,, dar ar putea să nu conţină acelaşi mesaj de excepţie sau set înfăşurat de clase de excepţii specifice implementării.

Format serializare XML

Modul de parsare XML leneş furnizează o optimizare de performanţă care încearcă să copieze XML nemodificat de la fluxul de octeţi de intrare la fluxul de octeţi de ieşire pe serializare. Rezultatul este performanţă crescută, dar formatul de serializare al şirului de octeţi XML de ieşire poate fi diferit dacă întregul obiect business a fost actualizat în mod de parsare XML leneş sau dacă rulează în mod de parsare XML vioi.

Deşi formatul de serializare XML ar putea să nu fie întocmai echivalent din punct de vedere sintactic, valoarea semantică furnizată de obiectul business este echivalentă independent de modurile de parsare şi XML poate fi transmis în siguranţă între aplicaţii care rulează în moduri de parsare diferite cu echivalenţă semantică.

Validator instanţă obiect business

Validatorul instanţei modulului obiectului business de parsare XML leneş furnizează o validare de fidelitate mai înaltă a obiectelor business, în special validare faţetă de valori proprietate. Din cauza acestor îmbunătăţiri, validatorul instanţei modulului de parsare leneş prinde probleme suplimentare care nu sunt prinse în modul de parsare vioi şi furnizează mesaje de eroare mai detaliate.

Mapări XML versiune 602

Fluxurile de mediere dezvoltate iniţial înainte de WebSphere Integration Developer Version 6.1 pot conţine primitive Mapping care utilizează o hartă sau o foaie de stil care nu poate rula direct în modul de parsare XML mai lent. La migrarea unei aplicaţii pentru utilizarea în modul leneş de parsare XML, fişierele hartă asociate cu primitivele Mapping pot fi actualizate în mod automat de către vrăjitorul de migrare pentru a rula în noul mod. Totuşi, în cazul în care o primitivă Mapping se referă în mod direct la o foaie de stil care a fost editată manual, foaia de stil nu este migrată şi nu poate rula in modul XML mai lent.

API-uri nepublicate private

Dacă o aplicaţie profită de interfeţe de programare obiecte business specifice implementării, private, nepublicate, aplicaţia are toate şansele să eşueze compilarea când este schimbat modul de parsare. În mod de parsare vioi, aceste interfeţe private sunt tipic clase de implementare obiect business definite de EMF (Eclipse Modeling Framework).

În toate cazurile, este recomandat cu tărie ca API-urile private să fie înlăturate din aplicaţie.

API-uri EMF Obiect mesaj serviciu

O componentă de mediere din IBM Integration Designer furnizează abilitatea de a manipula conţinut mesaj utilizând clase şi interfeţe Java furnizate în pachetul com.ibm.websphere.sibx.smobo. În mod de parsare XML leneş, interfeţele Java din pachetul com.ibm.websphere.sibx.smobo pot fi încă utilizate, dar metodele care se referă direct la clasele şi interfeţele EMF (Eclipse Modeling Framework) sau care sunt moştenite de la interfeţele EMF au toate şansele să eşueze.

ServiceMessageObject şi conţinuturile sale nu pot fi atribuite obiectelor EMF în mod de parsare XML leneş.

Serviciu BOMode

Serviciul BOMode este utilizat pentru a determina dacă modul de parsare XML care se execută momentan este vioi sau leneş.

Migrare

Toate aplicaţiile dinainte de versiunea 7.0.0.0 rulează în mod de parsare XML vioi. Când sunt migrate în timpul rulării utilizând uneltele de migrare runtime BPM, continuă să ruleze în mod de parsare XML vioi.

Pentru a permite unei aplicaţii anterioare versiunii 7.0.0.0 să fie configurată pentru a utiliza modul de parsare XML, utilizaţi întâi Integration Designer pentru a migra artefactele aplicaţiei. După migrare, configuraţi aplicaţia pentru a utiliza parsare XML leneşă.

Vedeţi Migrare artefacte sursă pentru informaţii despre migrarea artefactelor din Integration Designer şi vedeţi Configurarea modului de parsare obiect business al modulelor şi bibliotecilor pentru informaţii despre setarea modului de parsare.

Relaţii

O relaţie este o asociere între două sau mai multe entităţi de date, în special obiecte business. În IBM Business Process Manager Advanced, relaţiile pot fi utilizate pentru a transforma date care sunt echivalente peste obiecte operaţionale şi alte date care sunt reprezentate diferit, sau pe care le pot utiliza pentru a trasa aplicaţii peste diferite obiecte găsite în aplicaţii diferite. Acestea pot fi partajate între aplicaţii, soluţii şi chiar produse.

Serviciul de relaţii din IBM Business Process Manager Advanced furnizează infrastructura şi operaţiile pentru gestionarea relaţiilor. Deoarece vă permite să trataţi obiecte business indiferent de locul în care acestea se află, acesta vă poate oferi o vizualizare de asamblare unificată a tuturor aplicaţiilor dintr-o întreprindere şi poate fi folosit ca un bloc de construire pentru soluţii BPM. Deoarece relaţiile pot fi extinse şi gestionate, ele pot fi utilizate în soluţii complexe de integrare.

Ce sunt relaţiile?

O relaţie este o asociere între obiecte business. Fiecare obiect business dintr-o relaţie este numit *participant* la relaţie. Fiecare participant din relaţie este distins de alţi participanţi pe baza funcţiei sau a *rolului* pe care îl are în acea relaţie. O relaţie conţine o listă de roluri.

Definiţia relaţiei descrie fiecare rol şi specifică modul în care sunt înrudite rolurile. De asemenea, aceasta descrie "forma" generală a relaţiei. De exemplu, acest rol poate avea doar un participant, dar acest alt rol poate avea câţi participanţi sunt necesari. Puteţi defini o relaţie *car-owner*, de exemplu, unde un posesor poate avea mai multe maşini. De exemplu, o instanţă poate avea următorii participanţi pentru fiecare dintre aceste roluri:

- Maşină (Ferrari)
- Posesor (John)

Definiţia relaţiei este un şablon pentru *instanţa* relaţiei. Instanţa este instanţierea relaţiei în timpul rulării. În exemplul cu *proprietarii* de *maşini*, o instanţă ar putea descrie oricare din următoarele asocieri:

- John deţine Ferrari
- Sara deţine Mazda
- Bob deţine Ferrari

Utilizarea relaţiilor vă eliberează de necesitatea de a construi personalizat persistenţa de urmărire a relaţiei din logica dumneavoastră operaţională. Pentru anumite scenarii, serviciul de relaţii face toată munca în locul dumneavoastră. Vedeţi exemplul descris în secţiunea din Relaţii de identitate.

Scenarii

Aici este un exemplu tipic de situaţie în care o soluţie de integrare poate folosi relaţii. O corporaţie mare cumpără mai multe companii sau unităţi de afaceri. Fiecare unitate operaţională foloseşte software diferit pentru a monitoriza personalul şi caietele. Compania are nevoie de o cale de a-şi monitoriza angajaţii şi carnetele lor. Aceasta vrea o soluţie care le permite:

- Vizualizarea tuturor angajaţilor din diverse unităţi de afaceri ca şi cum ar fi într-o singură bază de date
- O singură vizualizare a tuturor carnetelor
- Permite angajaţilor să se logheze în sistem şi să cumpere un carnet.
- Acomodarea diferitelor sisteme de aplicaţii de întreprindere din diverse unităţi de afaceri

Pentru a realiza acest lucru, compania are nevoie de o cale pentru a se asigura, de exemplu, că John Smith şi John A. Smith din aplicaţii diferite sunt văzuţi ca acelaşi angajat. De exemplu, aceştia au nevoie de o cale pentru a consolida o singură entitate în mai multe spaţii ale aplicaţiei.

Scenariile mai complexe de relaţii implică construirea proceselor BPEL care stabilesc relaţii între obiecte diferite găsite în mai multe aplicaţii. Cu scenarii mai complexe de relaţii, obiectele business se află în soluţia de integrare, ci nu în aplicaţii. Serviciul de relaţii asigură o platformă pentru gestionarea în mod persistent a relaţiilor. Înainte de serviciul de relaţii, trebuie să construiţi propriul serviciu de persistenţă al obiectului. Două exemple de scenarii complexe de relaţii sunt:

- Aveţi un obiect business **car** cu un număr VIN într-o aplicaţie SAP şi vreţi să urmăriţi că această maşină este deţinută de altcineva. Totuşi, relaţia de posesiune este cu cineva într-o aplicaţie PeopleSoft. În acest tipar de relaţii, aveţi două soluţii şi este nevoie să construiţi o punte laterală între ele.
- O companie mare de comerţ cu amănuntul vrea să fie capabilă să monitorizeze marfa returnată pentru primirea banilor înapoi sau pentru credit. Există două aplicaţii diferite implicate: OMS (order management system) pentru achiziţii şi RMS (returns management system) pentru returnări. Obiectele business se află în mai multe aplicaţii şi aveţi nevoie de o cale de a afişa relaţiile care există între ele.

Modele de utilizare comune

Cele mai comune tipare de relaţii sunt tiparele *echivalenţă*. Acestea sunt bazate pe referinţa încrucişată sau pe corelare. Există două tipuri de relaţii care se potrivesc cu acest tipar: *non-identitate* şi *identitate*.

- **Relaţii non-identitate** stabilesc asocieri între obiecte business sau alte date pe o bază unul-la-mulţi sau mulţi-la-mulţi. Pentru fiecare instanţă a relaţiei, pot exista una sau mai multe instanţe pentru fiecare participant. Un tip de relaţie non-identitate este o relaţie de căutare statică. Un astfel de exemplu este o relaţie în care **CA** dintr-o aplicaţie SAP este înrudită cu **California** dintr-o aplicaţie Siebel.

- **Relaţii de identitate** stabilesc asocieri între obiecte business sau alte date pe o bază unu-la-unu. Pentru fiecare instanţă a relaţiei, poate exista doar o instanţă a fiecărui participant. Relaţiile de identitate capturează referinţe încrucişate între obiecte business care sunt echivalente semantic, dar care sunt identificate în mod diferit în diferite aplicaţii. Fiecare participant la relaţie este asociat cu un obiect business care are o valoare (sau combinaţie de valori) care identifică obiectul în mod unic. În mod obişnuit, relaţiile de identitate transformă atributele cheie ale obiectelor business precum numere ID şi coduri de produs.

  De exemplu, dacă aveţi obiecte business **car** în aplicaţii SAP, PeopleSoft şi Siebel şi vreţi să construiţi o soluţie care să le sincronizeze, va fi nevoie, în mod normal, să introduceţi logica de sincronizare a relaţiei construite manual în şase mapări:

  SAP -> generic

  generic -> SAP

  PeopleSoft-> generic

  generic-> PeopleSoft

  Siebel-> generic

  generic-> Siebel

  Totuşi, dacă utilizaţi relaţii în soluţia dumneavoastră, serviciul de relaţii asigură implementări de tipare preconstruite care menţin toate aceste relaţii pentru dumneavoastră.

Unelte pentru lucrul cu relaţii

Editorul de relaţii din Integration Designer este unealta pe care o utilizaţi pentru a modela şi proiecta relaţii şi roluri de integrare business. Pentru informaţii detaliate despre experienţa necesară şi despre taskuri la crearea relaţiilor şi utilizarea editorului de relaţii, vedeţi Creare relaţii.

Serviciul de relaţii este un serviciu de infrastructură din IBM Business Process Manager care menţine relaţii şi roluri în sistem şi oferă operaţii pentru gestiunea relaţiei şi a rolului.

Managerul de relaţii este interfaţa administrativă pentru gestionarea relaţiilor. Acesta este accesat prin pagina Manager de relaţii a consolei administrative.

Relaţiile pot fi invocate programatic prin API-urile serviciului de relaţii.

Serviciu de relaţie

Serviciul de relaţii memorează date ale relaţiei în tabele de relaţii unde păstrează pista valorilor specifice aplicaţiei din aplicaţii şi soluţii. Serviciul de relaţii oferă operaţii pentru gestiunea relaţiilor şi a rolului.

Cum funcţionează relaţiile

Relaţiile şi rolurile sunt descrise folosind interfaţa grafică a uneltei editor de relaţii din Integration Designer. Serviciul de relaţii memorează date de corelare în tabele din baza de date a relaţiei în sursa implicită de date pe care o specificaţi la configurarea serviciului de relaţii. O tabelă separată (uneori numită tabelă participantă) memorează informaţii pentru fiecare participant la relaţie. Serviciul de relaţii utilizează aceste tabele ale relaţiei pentru a păstra pista valorilor înrudite specifice aplicaţiei şi pentru a propaga informaţii actualizate în toate soluţiile.

Relaţiile, care sunt artefacte business, sunt implementate într-un proiect sau într-o bibliotecă partajată. La prima implementare, serviciul de relaţii populează datele.

În timpul rulării, când mapările sau alte componente IBM Business Process Manager necesită o instanţă de relaţie, instanţele relaţiei sunt fie actualizate, fie extrase în funcţie de scenariu.

Datele instanţei de relaţie sau rol pot fi manipulate prin trei mijloace:

- Invocări componentă IBM Business Process Manager snippet Java ale API-urilor serviciului de relaţii

- Transformări ale relaţiei în serviciul de mapare a obiectului business IBM Business Process Manager
- Unealta managerului de relaţii

Pentru informaţii detaliate despre experienţa necesară şi despre taskuri la crearea relaţiilor, identificarea tipurilor de relaţii şi utilizarea editorului de relaţii, vedeţi subiectul Creare relaţii.

Manager de relaţie

Managerul de relaţii este interfaţa administrativă pentru gestionarea relaţiilor. Acesta este accesat prin pagina Manager de relaţii a consolei administrative.

Managerul de relaţii asigură o interfaţă grafică cu utilizatorul pentru crearea şi manipularea datelor relaţiei şi ale rolului în timpul rulării. Puteţi gestiona entităţi ale relaţiei la toate nivelurile: instanţă de relaţie, instanţă de rol şi niveluri ale datelor atribut şi ale datelor proprietate. Cu managerul de relaţii, puteţi:

- Vizualiza o listă a relaţiilor din sistem şi a informaţiilor detaliate despre relaţii individuale
- Gestiona instanţe de relaţii:
  - Interoga date ale relaţiei pentru a vizualiza subseturi de date ale instanţei
  - Interoga date ale relaţiei pentru a vizualiza subseturi de date ale instanţei utilizând vizualizări ale bazei de date
  - Vizualiza o listă a instanţelor relaţiei care se potrivesc unei interogări a relaţiei şi informaţii detaliate despre o instanţă
  - Edita valorile proprietăţii pentru o instanţă a relaţiei
  - Crea şi şterge instanţe de relaţie
- Gestiona roluri şi instanţe de roluri:
  - Vizualiza detalii despre un rol sau o instanţă de rol
  - Edita proprietăţile instanţei de rol
  - Crea şi şterge instanţe de rol pentru o relaţie
  - Derula înapoi date ale instanţei de relaţie la un punct în care ştiţi că datele sunt de încredere
- Importa date dintr-o relaţie statică existentă în sistemul dumneavoastră sau exporta date dintr-o relaţie statică existentă într-un fişier RI sau CSV
- Înlătura schema şi datele relaţiei din magazie când aplicaţia care le foloseşte este dezinstalată

Relaţii în medii Network Deployment

Relaţiile pot fi utilizate în medii ND (Network Deployment) fără nici o configuraţie suplimentară.

În medii ND (Network Deployment), relaţiile sunt instalate într-un cluster de aplicaţii. Atunci relaţiile sunt vizibile în cluster şi toate serverele din cluster au acces la datele instanţei memorate în baza de date a relaţiei. Abilitatea de rula serviciul de relaţii într-un mediu ND îl face scalabil şi cu disponibilitate bună.

Managerul de relaţii permite relaţiilor să fie gestionate în cluster-e diferite printr-o interfaţă administrativă centralizată. Conectaţi managerul de relaţii la un server dintr-un cluster selectând MBean-ul relaţiei sale.

API-uri ale serviciului de relaţii

Relaţiile pot fi invocate programatic prin API-urile serviciului de relaţii în sau în afara mapărilor de obiecte business.

Sunt disponibile trei tipuri de API-uri:

- API-uri de manipulare a instanţei de relaţie (inclusiv crearea, actualizarea, ştergerea directă a datelor instanţei)
- API-uri suport ale tiparului de relaţii (inclusiv correlate(), correlateforeignKeyLookup)
- Tipare căutare relaţie (API-uri de căutare)

Magistrala ESB (Enterprise Service Bus) din IBM Business Process Manager

IBM Business Process Manager suportă integrarea serviciilor aplicaţie, inclusiv aceleaşi capabilităţi ca WebSphere Enterprise Service Bus.

Conectarea serviciilor printr-o magistrală pentru serviciile întreprinderii

Cu ajutorul unui ESB (Enterprise Service Bus), puteţi maximiza flexibilitatea unui SOA. Participanţii din interacţiunea cu serviciul sunt conectaţi mai degrabă la ESB, decât direct unul cu altul.

În cazul în care solicitantul serviciului se conectează la ESB, atunci ESB-ul îşi asumă răspunderea pentru furnizarea cererilor sale, folosind mesaje, către un furnizor de servicii oferind funcţia necesară şi calitatea serviciilor. ESB uşurează interacţiunile solicitant-furnizor şi adresează protocoalele nepotrivite, tiparele de interacţiune sau capabilităţile serviciului. De asemenea, un ESB poate permite sau îmbunătăţi monitorizarea şi gestiunea. ESB oferă caracteristici de virtualizare şi management care pun în aplicare şi extind capacităţile de bază pentru SOA.

ESB prezintă pe scurt următoarele caracteristici:

Locaţie şi identitate
: Participanţii nu trebuie să ştie locaţia sau identitatea altor participanţi. De exemplu, solicitanţii nu trebuie să fie conştienţi de faptul că o cerere ar putea fi deservită de oricare dintre furnizori; furnizorii de servicii pot fi adăugaţi sau înlăturaţi fără întrerupere.

Protocol de Interacţiune
: Participanţii nu trebuie să partajeze acelaşi protocol de comunicaţie sau stil de interacţiune. De exemplu, o cerere exprimată ca SOAP peste HTTP poate fi deservită de un furnizor care înţelege doar SOAP peste JMS (Java Message Service).

Interfaţă
: Solicitanţii şi furnizorii nu trebuie să fie de acord cu o interfaţă comună. Un ESB împacă diferenţele prin transformarea mesajelor de cerere şi de răspuns într-o formă aşteptată de furnizor.

Calităţile unui serviciu (de interacţiune)
: Participanţii, sau administratorii de sisteme, îşi declară cerinţele legate de calitatea serviciilor, inclusiv autorizaţia cererilor, criptarea şi decriptarea conţinutului mesajelor, auditarea automată a interacţiunilor dintre servicii, şi modul în care cererile lor ar trebui să fie rutate (de exemplu optimizarea vitezei sau a costului).

Interpunerea ESB între participanţi vă dă posibilitatea să modulaţi interacţiunea acestora prin intermediul unei construcţii logice numite *mediere*. Medierile se aplică mesajelor aflate în zbor între solicitanţi şi furnizori. De exemplu, medierile pot fi folosite pentru a găsi servicii cu caracteristici specifice cerute de un solicitant, sau pentru a rezolva diferenţele interfeţei apărute între solicitanţi şi furnizori. Pentru interacţiuni complexe, medierile pot fi legate secvenţial.

Folosind medieri, o magistrală pentru serviciile întreprinderii efectuează următoarele acţiuni între solicitant şi serviciu:

- *Rutarea* mesajelor între servicii. O magistrală pentru serviciile întreprinderii oferă o infrastructură de comunicaţie comună care poate fi folosită pentru a conecta servicii, şi, prin urmare funcţiile business pe care acestea le reprezintă, fără a fi nevoie ca programatorii să scrie şi să menţină logica complexă de conectivitate.
- *Convertirea* protocoalelor de transport între solicitant şi serviciu. O magistrală pentru serviciile de întreprindere oferă o cale consistentă bazată pe standarde de integrare a funcţiilor business care folosesc diferite standarde IT. Acest lucru permite integrarea funcţiilor business care nu au putut comunica în mod normal, cum ar fi conectarea aplicaţiilor în silozurile departamentale sau activarea aplicaţiilor în companii diferite care participă în interacţiunile serviciului.
- *Transformarea* formatelor mesajelor între solicitant şi serviciu. O magistrală pentru serviciile de întreprindere permite funcţiilor business să schimbe informaţii în formate diferite, iar magistrala se asigură că informaţiile livrate funcţiei business sunt în formatul cerut de acea aplicaţie.

- *Tratarea* evenimentelor business din surse incompatibile. O magistrală pentru serviciile de întreprindere suportă interacţiuni bazate pe evenimente în plus faţă de schimbul de mesaje ce tratează cererile serviciului.

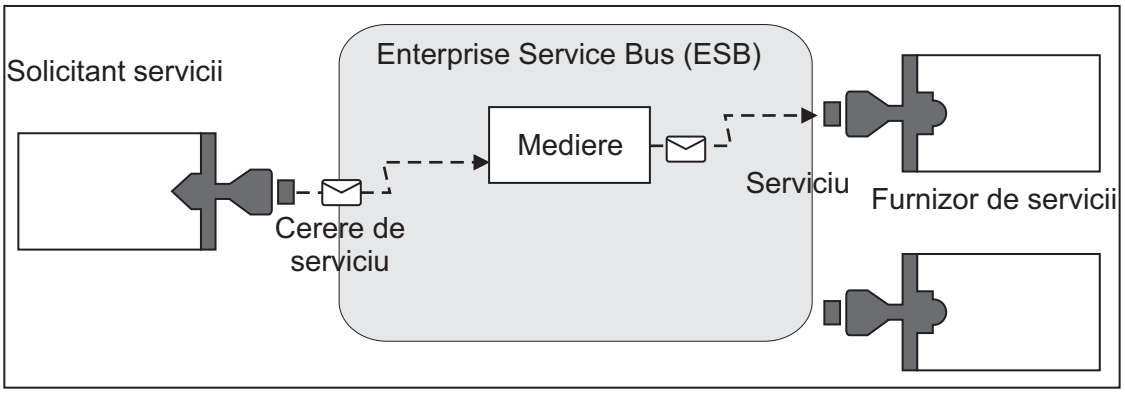


Figura 87. O magistrală pentru serviciile întreprinderii. Magistrală pentru serviciile de întreprindere rutează mesajele între aplicaţii, care pot fi solicitanţi sau furnizori de servicii. Magistrala converteşte protocoalele de transport şi transformă formatul mesajelor între solicitanţi şi furnizori. În această figură, fiecare aplicaţie utilizează un protocol diferit (reprezentate prin formele geometrice diferite ale conectorilor lor) şi foloseşte diferite formate de mesaje.

Folosind magistrala pentru serviciile de întreprindere vă puteţi concentra pe afacerea dumneavoastră de bază mai degrabă decât pe sistemele dumneavoastră computer. Puteţi modifica sau adăuga la servicii, dacă este necesar; de exemplu, pentru a răspunde la modificările din cerinţa de afaceri, pentru a adăuga capacitate de serviciu suplimentar sau pentru a adăuga noi capabilităţi. Aveţi posibilitatea să efectuaţi modificările necesare prin reconfigurarea magistralei, cu impact redus sau nul asupra serviciilor şi aplicaţiilor existente care folosesc magistrala.

Infrastructura de mesagerie ESB (Enterprise Service Bus)

IBM Business Process Manager include capabilităţi ale magistralei ESB. IBM Business Process Manager suportă integrarea tehnologiilor orientate spre servicii, orientate pe mesaj şi conduse de eveniment pentru a asigura o infrastructură a mesageriei bazată pe standarde într-o magistrală de servicii de întreprindere integrate.

Capabilităţile serviciului de întreprindere pe care le puteţi utiliza pentru aplicaţiile dumneavoastră de întreprindere asigură nu doar un nivel de transport dar şi suport de mediere pentru a facilita interacţiunile serviciului. Magistrala ESB (Enterprise Service Bus) este construită în jurul standardelor deschise şi a SOA (service-oriented architecture). Este bazată pe infrastructura Java EE robustă şi serviciile platformei asociate furnizate de IBM WebSphere Application Server Network Deployment.

IBM Business Process Manager este alimentat de aceeaşi tehnologie disponibilă cu IBM WebSphere Enterprise Service Bus. Această capabilitate este parte componentă a funcţionalităţii de bază a IBM Business Process Manager şi nu este necesară nici o licenţă suplimentară pentru WebSphere Enterprise Service Bus pentru a beneficia de aceste capabilităţi.

Totuşi, puteţi implementa licenţe autonome suplimentare ale WebSphere Enterprise Service Bus în întreprinderea dumneavoastră pentru a extinde întinderea conectivităţii soluţiilor de integrare procese motorizate de deIBM Business Process Manager. De exemplu, WebSphere Enterprise Service Bus poate fi instalat mai aproape de o aplicaţie SAP pentru a găzdui un IBM WebSphere Adapter for SAP şi pentru a transforma mesajele SAP înainte de a trimite acele informaţii în reţea într-un proces operaţional condus de IBM Business Process Manager.

Gazde de destinaţii mesagerie sau coadă

O gazdă destinaţie de mesagerie sau coadă furnizează funcţia de mesagerie într-un server. Un server devine gazda destinaţiei mesageriei la configurarea lui ca destinaţie mesagerie.

Un motor de mesagerie rulează într-un server. Motorul mesagerie furnizează funcţii mesagerie şi un punct de conexiune pentru ca aplicaţiile să se conecteze la magistrală. Comunicaţia asincronă Service Component Architecture (SCA), importurile şi exporturile JMS, procesarea internă asincronă utilizează cozi de mesaje în motorul de mesagerie.

Mediul de implementare conectează sursa mesajului la destinaţia mesajului prin magistrală la implementarea modulelor de aplicaţii. Cunoaşterea sursei mesajului şi a destinaţiei mesajului vă ajută să determinaţi de ce tip de mediu de implementare aveţi nevoie.

Aplicaţiile pot stoca date persistente într-un depozit de date, care este un set de tabele dintr-o bază de date sau schemă, sau într-un depozit de fişiere. Motorul mesagerie utilizează o instanţă a unei surse de date JDBC pentru a interacţiona cu acea bază de date.

Configuraţi gazda destinaţie a mesageriei la definirea mediului de implementare utilizând **Server** din consola administrativă sau desemnaţi serverul ca gazdă destinaţie în timpul instalării software-ului.

Depozite de date:

Fiecare motor de mesagerie poate utiliza un depozit de date care este un set de tabele dintr-o bază de date sau o schemă care memorează date persistente.

Toate tabelele din depozitul de date sunt reţinute în aceeaşi schemă a bazei de date. Puteţi crea fiecare depozit de date într-o bază de date separată. Alternativ, puteţi crea mai multe depozite de date în aceeaşi bază de date, fiecare depozit de date utilizând o schemă diferită.

Un motor de mesagerie utilizează o instanţă a unei surse de date JDBC pentru a interacţiona cu baza de date care conţine depozitul de date pentru acel motor de mesagerie.

Furnizori JDBC

Puteţi folosi furnizorii JDBC pentru interacţionarea aplicaţiile cu bazele de date relaţionale.

Aplicaţiile folosesc furnizori JDBC pentru interacţionarea cu bazele de date relaţionale. Furnizorii JDBC livrează clasele de implementare specifice driver-ului JDBC pentru accesul la un tip specific de bază de date. Pentru crearea unui pool de conexiuni la acea bază de date, asociaţi sursa de date cu furnizorul JDBC. Împreună, furnizorul JDBC şi obiectele sursei de date sunt echivalente funcţional cu fabrica de conexiuni Java EE Connector Architecture (JCA), care furnizează conexiunea cu o bază de date non-relaţională.

Referiţi-vă la exemplele din Setări pentru mediul tipic autonom şi Setări pentru mediul tipic de implementare din subiectul anterior.

Pentru informaţii suplimentare despre furnizorii JDBC, vedeţi “Furnizori JDBC” în Centrul de informare WebSphere Application Server.

Magistrale de integrare a serviciului pentru IBM Business Process Manager

O magistrală de integrare a serviciului este un mecanism de comunicare gestionat care suportă integrarea serviciului prin mesageria sincronă şi asincronă. O magistrală conţine motoare de mesagerie interconectate care gestionează resursele magistralei. Este una din tehnologiile WebSphere Application Server pe care este bazat IBM Business Process Manager.

Unele magistrale sunt create automat pentru utilizarea de către sistem, aplicaţiile SCA (Service Component Architecture) pe care le implementaţi şi de către alte componente. De asemenea, puteţi crea magistrale pentru a suporta logica integrării serviciului sau alte aplicaţii, de exemplu, pentru a suporta aplicaţii care acţionează ca solicitanţi şi furnizori ai serviciului în IBM Business Process Manager sau pentru a vă lega WebSphere MQ.

O destinaţie magistrală este o adresă logică la care aplicaţiile se pot ataşa ca producător, consumator sau ambele. O destinaţie coadă este o destinaţie magistrală care este utilizată pentru mesagerie punct-la-punct.

Fiecare magistrală poate avea unul sau mai mulţi membri magistrală, fiecare dintre ei fiind un server sau un cluster.

Topologie magistrală este aranjarea fizică a serverelor de aplicaţii, motoarelor de mesagerie şi a managerilor de cozi WebSphere MQ şi tiparul conexiunilor magistralei şi al legăturilor dintre ele care alcătuiesc magistrala de servicii întreprindere.

Unele magistrale de integrare servicii sunt create automat pentru a suporta IBM Business Process Manager. Între şase magistrale sunt create la crearea mediului de implementare sau la configurarea unui server sau cluster pentru a suporta aplicaţii SCA. Fiecare dintre aceste magistrale are cinci aliasuri de autentificare pe care trebuie să le configuraţi.

Magistrală de sistem SCA:

Magistrala de sistem SCA este o magistrală de integrare a serviciului care este utilizată pentru a găzdui destinaţii pentru module SCA (Service Component Architecture). Runtime-ul SCA, ce suportă module de mediere, foloseşte destinaţii coadă pe magistrala de sistem ca pe o infrastructură pentru interacţiuni asincrone între componente şi module.

Magistrala de sistem este creată automat odată cu mediul de implementare sau la configurarea serverului sau cluster-ului pentru aplicaţii SCA. Magistrala de sistem furnizează un domeniu în care resursele sunt configurate pentru module de mediere şi puncte finale de interacţiune. Magistrala permite rutarea mesajelor între punctele finale. Puteţi să specificaţi QoS (calitatea serviciului) pentru magistrală, incluzând prioritatea şi fiabilitatea.

Numele magistralei este SCA.SYSTEM.busID.Bus. Aliasul de autentificare folosit la securizarea magistralei este SCA_Auth_Alias.

Magistrală de aplicaţii SCA:

Destinaţiile magistralei de aplicaţii suportă comunicarea asincronă a WebSphere Business Integration Adapters şi a altor componente System Component Architecture.

Magistrala aplicaţiei este creată automat la crearea unui mediu de implementare sau la configurarea unui server sau cluster pentru a suporta aplicaţii SCA. Magistrala aplicaţiei este similară cu magistralele de integrare a serviciului pe care le puteţi crea pentru a suporta logica integrării serviciului sau alte aplicaţii.

Numele magistralei este SCA.APPLICATION.busID.Bus. Aliasul de autentificare utilizat pentru securizarea acestei magistrale este SCA_Auth_Alias.

Magistrala Common Event Infrastructure:

Magistrala Common Event Infrastructure este utilizată pentru transmiterea evenimentelor de bază obişnuite, în mod asincron, serverului Common Event Infrastructure configurat.

Numele magistralei este CommonEventInfrastructure_Bus. Aliasul de autentificare utilizat pentru securizarea acestei magistrale este CommonEventInfrastructureJMSAuthAlias

Magistrala Business Process Choreographer:

Utilizaţi numele magistralei Business Process Choreographer şi autentificarea pentru transmisia mesajelor interne.

Magistrala Business Process Choreographer este utilizată pentru transmiterea mesajelor intern şi pentru API-ul Java Messaging Service (JMS) al managerului de flux de afaceri.

Numele magistralei este BPC.cellName.Bus. Aliasul de autentificare este BPC_Auth_Alias

Magistrală Performance Data Warehouse:

Magistrala Performance Data Warehouse este folosită pentru a transmite mesaje intern de către infrastructură şi pentru a comunica cu clienţii IBM Business Process Manager.

Magistrala Performance Data Warehouse este creată în mod automat la crearea unui mediu de implementare.

Numele magistralei este PERFDW.busID.Bus. Aliasul de autentificare utilizat pentru securizarea acestei magistrale este PERFDWME_Auth_Alias.

Magistrală Process Server:

Magistrala Process Server este folosită pentru transmiterea internă a mesajelor de către infrastructură şi pentru a comunica cu clienţii IBM Business Process Manager.

Magistrala Process Server este creată automat când creaţi un mediu de implementare.

Numele magistralei este PROCSVR.busID.Bus. Aliasul de autentificare utilizat pentru securizarea acestei magistrale este PROCSVRME_Auth_Alias.

Aplicaţii de servicii şi module de servicii

Un modul serviciu este un modul Service Component Architecture (SCA) care oferă servicii în mediul runtime. Atunci când instalaţi un modul serviciu pe IBM Business Process Manager, construiţi o aplicaţie serviciu asociată, care este împachetată ca un fişier EAR (enterprise archive).

Modulele de servicii sunt unităţi de bază ale implementării şi pot conţine componente, biblioteci şi module de intermediere folosite de aplicaţia pentru servicii asociată. Modulele de servicii au exporturi şi, în mod opţional, importuri pentru a defini relaţiile între module şi solicitanţii şi furnizorii serviciului. WebSphere Process Server suportă module pentru serviciile business şi module de mediere. Atât modulele, cât şi modulele de mediere sunt tipuri de module SCA. Un modul de mediere permite comunicaţia între aplicaţii prin transformarea invocării serviciului într-un format înţeles de ţintă, transmiţând cererea către ţintă şi returnând rezultatul iniţiatorului. Un modul pentru un serviciu operaţional implementează logica unui proces operaţional. Totuşi, un modul poate de asemenea să includă aceeaşi logică de mediere care poate fi împachetată într-un modul de mediere.

Implementarea unei aplicaţii de servicii

Procesul de implementare a unui fişier EAR care conţine o aplicaţie de servicii este acelaşi cu cel de implementarea a oricărui fişier EAR. Puteţi modifica valorile pentru parametrii de mediere în momentul implementării. După ce au implementat un fişier EAR care conţine un modul SCA, puteţi vizualiza detaliile legate de aplicaţia de servicii şi modulele sale asociate. Puteţi vedea modul în care un modul de servicii este conectat la solicitanţii serviciului (prin exporturi) şi la furnizorii de servicii (prin importuri).

Vizualizarea detaliilor modulului SCA

Detaliile legate de modulul de servicii pe care le puteţi vizualiza depind de modulul SCA. Acestea includ următoarele atribute.

- Nume modul SCA
- Descriere modul SCA
- Nume asociat aplicaţiei
- Informaţii legate de versiunea modulului SCA, daca modulul are mai multe versiuni
- Importuri module SCA:
  - Interfeţele de import sunt definiţii prezentare pe scurt care descriu modul în care un modul SCA accesează un serviciu.
  - Legările de import sunt definiţii concrete care specifică mecanismul fizic prin care un modul SCA accesează un serviciu. De exemplu, folosirea SOAP/HTTP.
- Exporturi modul SCA:
  - Interfeţele de export sunt definiţii prezentare pe scurt care descriu modul în care solicitanţii serviciului accesează un modul SCA.

  - Legările de export sunt definiţii concrete care specifică mecanismul fizic prin care un solicitant al serviciului accesează un modul SCA, şi în mod indirect, un serviciu.
- Proprietăţile modulului SCA

Importuri şi legături import

Importurile definesc interacţiuni între modulele SCA şi furnizorii de servicii. Modulele SCA folosesc importuri pentru a permite componentelor să acceseze servicii externe (servicii care se află în afara modulului SCA) folosind o reprezentare locală. Legările de import definesc o anumită cale prin care este accesat un serviciu extern.

Dacă modulele SCA nu au nevoie de acces la serviciile externe, atunci nu este necesar ca acestea să aibă importuri. Modulele de mediere au de obicei unul sau mai multe importuri care sunt utilizate pentru a transmite mesaje sau cereri către ţintele lor intenţionate.

Interfeţe şi legături

Importul unui modul SCA are nevoie de cel puţin o interfaţă şi are o singură legare.

- Interfeţele de import sunt definiţii abstracte care definesc un set de operaţiuni folosind Web Services Description Language (WSDL), şi limbaj XML pentru descrierea serviciilor Web. Un modul SCA poate avea mai multe interfeţe de import.
- Legările de import sunt definiţii concrete care specifică mecanismul fizic folosit de modulele SCA pentru a accesa un serviciu extern.

Legări de import suportate

IBM Business Process Manager suportă următoarele legături de import:

- Legările SCA conectează modulele SCA cu alte module SCA. De asemenea, legările SCA sunt menţionate şi ca legături implicite.
- Legările Serviciu Web permit componentelor să invoce servicii web. Protocoalele suportate sunt SOAP1.1/HTTP, SOAP1.2/HTTP, şi SOAP1.1/JMS.

  Puteţi folosi o legare SOAP1.1/HTTP sau SOAP1.2/HTTP bazată pe API-ul Java API pentru JAX-WS (XML Web Services), ceea ce permite interacţiunea cu serviciile folosind documente sau legături literale RPC şi care folosesc handler-e JAX-WS pentru a personaliza invocarea. Este furnizată o legare SOAP1.1/HTTP separată pentru a permite interacţiunea cu serviciile care folosesc o legare codată RPC sau acolo unde există o cerinţă pentru folosirea handler-elor JAX-RPC pentru a personaliza invocările.
- Legările HTTP vă permit să accesaţi aplicaţii folosind protocolul HTTP.
- Legările de import EJP (Enterprise JavaBeans) permit componentelor SCA să invoce servicii oferite de logica operaţională Java EE care rulează pe un server Java EE.
- Legările EIS (Enterprise information system) asigură conectivitatea între componente SCA şi un EIS extern. Această comunicaţie se realizează prin utilizarea adaptoarelor resurselor.
- Legările JMS (Java Message Service) 1.1 permit interoperabilitatea cu furnizorul implicit de mesaje WebSphere Application Server. JMS poate exploata diferite tipuri de transport, inclusiv TCP/IP şi HTTP sau HTTPS. Clasa JMS Message şi cele cinci subtipuri ale sale (Text, Bytes, Object, Stream şi Map) sunt acceptate automat.
- Legările JMS interoperabilitate cu furnizori JMS terţă parte care se integrează cu WebSphere Application Server folosind JMS ASF (Application Server Facility).
- Legările JMS MQ WebSphere permit interoperabilitatea cu furnizorii JMS bazaţi pe MQ WebSphere. Clasa JMS Message şi cele cinci subtipuri ale sale (Text, Bytes, Object, Stream şi Map) sunt acceptate automat. Dacă doriţi să utilizaţi WebSphere MQ pe post de furnizor JMS, folosiţi legările JMS MQ WebSphere.
- Legările MQ WebSphere permit interoperabilitatea cu furnizorii MQ WebSphere. Puteţi folosi legături WebSphere MQ doar împreună cu managerii cozii aflaţi la distanţă prin intermediul unei conexiuni client WebSphere MQ; nu le puteţi folosi cu managerii locali ai cozii. Folosiţi legături WebSphere MQ dacă doriţi să comunicaţi cu aplicaţiile WebSphere MQ native.

Invocarea dinamică a serviciilor

Serviciile pot fi invocate prin orice legare de import suportată. Un serviciu este găsit în mod normal la un punct final specificat în import. Acest punct final este numit punct final static. Invocarea unui serviciu diferit este posibilă prin înlocuirea punctului final static. Înlocuirea dinamică a punctelor finale statice vă permite să invocaţi un serviciu la un alt punct final, prin orice legare de import suportată. Invocarea dinamică a serviciilor vă permite de asemenea să invocaţi un serviciu în care legarea de import suportată nu are un punct final static.

Un import cu o legare asociată este folosit pentru a specifica protocolul şi configuraţia sa pentru invocarea dinamică. Importarea folosită pentru invocarea dinamică poate fi legată de componenta de apelare, sau poate fi selectată dinamic la runtime.

Pentru serviciul Web şi invocaţii SCA, se poate face o invocare dinamică fără importare, cu protocolul şi configurarea deduse din URL-ul punct final. Tipul ţintă de invocare este identificat din URL-ul final. Dacă este folosit un import, atunci URL-ul trebuie să fie compatibil cu protocolul legării de import.

- Un URL SCA indică invocarea unui alt modul SCA.
- Un HTTP sau un JMS URL indică în mod implicit invocări ale serviciului web; pentru aceste URL-uri este posibilă furnizarea unor valori de tip legătură suplimentare care să indice faptul că URL-ul reprezintă o invocare a modului de legare a unui HTTP sau JMS.
- Pentru un serviciu URL HTTP, modul implicit este folosirea SOAP 1.1, şi se poate specifica o valoare tip legătură care să indice folosirea SOAP 1.2.

Exporturile şi legările de export

Exporturile definesc interacţiuni între modulele SCA şi solicitanţii de servicii. Modulele SCA folosesc exporturi pentru a oferi altora servicii. Legările de export definesc un mod specific prin care un modul SCA este accesat de solicitanţii serviciului.

Interfeţe şi legături

Un export pentru un modul SCA are nevoie de cel puţin o interfaţă.

- Interfeţele de export sunt definiţii abstracte care definesc un set de operaţiuni folosind Web Services Description Language (WSDL), şi limbaj XML pentru descrierea serviciilor Web. Un modul SCA poate avea mai multe interfeţe de export.
- Legările de export sunt definiţii concrete care specifică mecanismul fizic folosit de solicitanţii serviciului pentru a accesa un serviciu. De obicei, exportul unui modul SCA are specificată o singură legare. Un export fără legături specificate este interpretat de mediul runtime ca un export cu o legătură SCA.

Legări de export suportate

IBM Business Process Manager suportă următoarele legături de export:

- Legările SCA conectează modulele SCA cu alte module SCA. De asemenea, legările SCA sunt menţionate şi ca legături implicite.
- Legările Serviciu web permit exporturi care pot fi invocate ca şi servicii web. Protocoalele suportate sunt SOAP1.1/HTTP, SOAP1.2/HTTP, şi SOAP1.1/JMS.

  Puteţi folosi o legare SOAP1.1/HTTP sau SOAP1.2/HTTP bazată pe JAX-WS (Java API for XML Web Services), ceea ce permite interacţiunea cu serviciile folosind documente sau legături literale RPC şi care folosesc handler-e JAX-WS pentru a personaliza invocările. Este furnizată o legare SOAP1.1/HTTP separată pentru a permite interacţiunea cu serviciile care folosesc o legare codată RPC sau acolo unde există o cerinţă pentru folosirea handler-elor JAX-RPC pentru a personaliza invocările.
- Legările HTTP permit ca exporturile să fie accesate folosind protocolul HTTP.
- Legările de export EJB (Enterprise JavaBeans) permit componentelor SCA să fie exportate sub formă de EJB-uri, astfel încât logica operaţională Java EE să poată invoca componente SCA care altfel le-ar fi indisponibile.

- Legările EIS (Enterprise information system) asigură conectivitatea între componente SCA şi un EIS extern. Această comunicaţie se realizează prin utilizarea adaptoarelor resurselor.
- Legările JMS (Java Message Service) 1.1 permit interoperabilitatea cu furnizorul implicit de mesaje WebSphere Application Server. JMS poate exploata diferite tipuri de transport, inclusiv TCP/IP şi HTTP sau HTTPS. Clasa JMS Message şi cele cinci subtipuri ale sale (Text, Bytes, Object, Stream şi Map) sunt acceptate automat.
- Legările JMS interoperabilitate cu furnizori JMS terţă parte care se integrează cu WebSphere Application Server folosind JMS ASF (Application Server Facility).
- Legările JMS MQ WebSphere permit interoperabilitatea cu furnizorii JMS bazaţi pe MQ WebSphere. Clasa JMS Message şi cele cinci subtipuri ale sale (Text, Bytes, Object, Stream şi Map) sunt acceptate automat. Dacă doriţi să utilizaţi WebSphere MQ pe post de furnizor JMS, folosiţi legările JMS MQ WebSphere.
- Legările MQ WebSphere permit interoperabilitatea cu furnizorii MQ WebSphere. Puteţi utiliza o conexiune de la distanţă (sau client) pentru a vă conecta la un manager de coadă MQ aflat pe o maşină la distanţă. O conexiune locală (sau legături) este o conexiune directă la WebSphere MQ. Aceasta poate fi utilizată doar pentru o conexiune la un manager al cozii MQ de pe aceeaşi maşină. WebSphere MQ va permite ambele tipuri de conexiuni, dar legările MQ suportă doar conexiunea "de la distanţă" (sau "client").

Module de mediere

Modulele de mediere sunt module SCA (Service Component Architecture) care pot modifica formatul, conţinutul sau ţinta cererilor serviciului.

Modulele de mediere operează asupra mesajelor care sunt în zbor între solicitanţii şi furnizorii serviciului. Puteţi ruta mesaje către diferiţi furnizori de servicii şi puteţi modifica conţinutul sau forma mesajului. Modulele de mediere pot oferi funcţii cum ar fi înregistrarea mesajului şi procesarea erorii care sunt adaptate cerinţelor dumneavoastră.

Puteţi modifica anumite aspecte ale modulelor de mediere din consola administrativă fără a fi nevoie să reimplementaţi modulul .

Componentele modulelor de mediere

Modulele de mediere cuprind următoarele elemente:

- Importuri, care definesc interacţiunea între modulele SCA si furnizorii de servicii. Acestea permit modulelor SCA să apeleze servicii externe ca şi cum acestea ar fi locale. Puteţi vizualiza importurile modulelor de mediere şi să modificaţi legarea.
- Exporturi, care definesc interacţiunea între modulele SCA şi solicitanţii serviciului. Acestea permit unui modul SCA să ofere un serviciu şi să definească interfeţele externe (puncte de acces) pentru un modul SCA. Puteţi vizualiza exporturile modulelor de mediere.
- Componente SCA, care construiesc blocuri pentru modulele SCA precum module de mediere. Aveţi posibilitatea să creaţi şi să particularizaţi module SCA şi componente în mod grafic, folosind Integration Designer. După ce aţi implementa un modul de mediere puteţi personaliza anumite aspecte ale acestuia din consola administrativă fără să fiţi nevoiţi să reimplementaţi modulul.

  De obicei, module de mediere conţine un anumit tip de componentă SCA numită *componentă a fluxului de mediere*. Componentele fluxului de mediere definesc fluxurile de mediere.

  Componente unui flux de mediere poate conţine nici o, una, sau un număr de primitive de mediere. IBM Business Process Manager suportă un set livrat de primitive de mediere care oferă funcţionalitate pentru rutarea şi transformarea mesajelor. Pentru flexibilitate suplimentară a primitivei de mediere, utilizaţi primitiva Mediere personalizată pentru apelarea logicii personalizate.

  Scopul unui modul de mediere care nu conţine o componentă pentru fluxul de mediere este de a transforma cererile serviciului de la un protocol la altul. De exemplu, cererea unui serviciu ar putea fi făcută folosind SOAP/JMS, dar ar putea avea nevoie să fie transformară în SOAP/HTTP înainte de a fi trimisă mai departe.

  Notă: Aveţi posibilitatea să vizualizaţi şi să faceţi anumite modificări asupra modulelor de mediere din IBM Business Process Manager. Totuşi, nu puteţi vizualiza sau modifica componentele SCA din interiorul unui modul din IBM Business Process Manager. Utilizaţi Integration Designer pentru a personaliza componentele SCA.

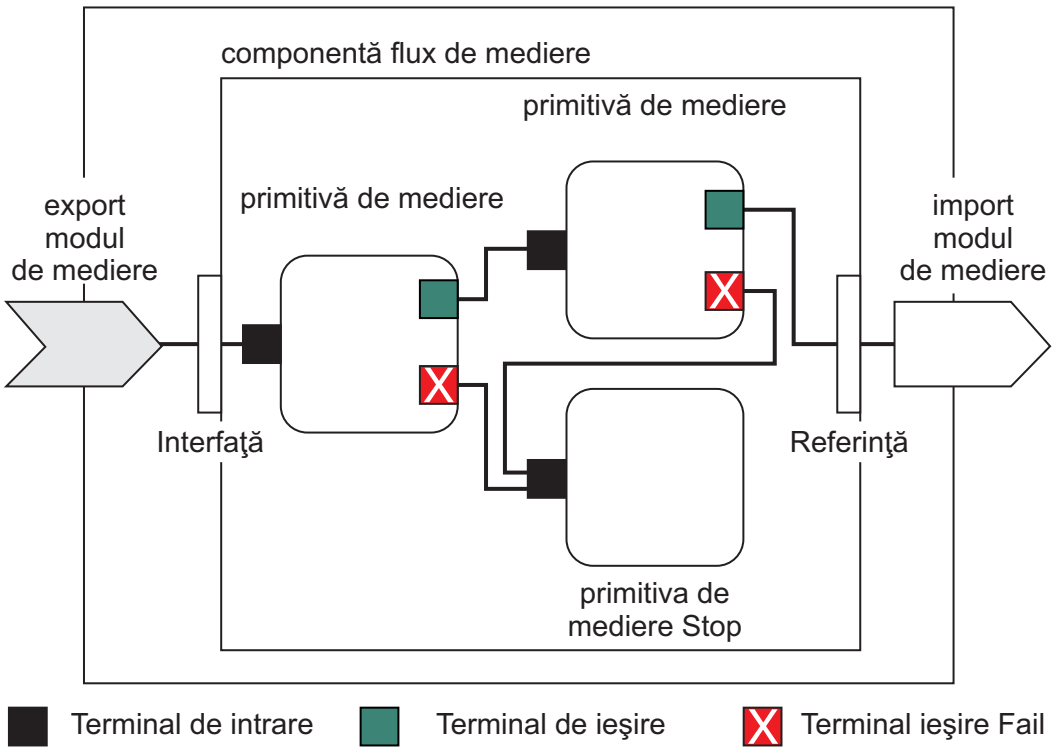


Figura 88. Exemplu simplificat de modul mediere. Modulul de mediere conţine o singură componentă a fluxului de mediere, care conţine primitive de mediere.

- Proprietăţi

  Primitivele de mediere au proprietăţi, unele dintre ele pot fi afişate în consola administrativă sub formă de proprietăţi suplimentare ale unui modul SCA.

  Pentru ca proprietăţile primitivelor de mediere să fie vizibile din consola administrativă IBM Business Process Manager, dezvoltatorul care se ocupă cu integrarea trebuie să le promoveze. Anumite proprietăţi se pretează a fi configurate administrativ şi Integration Designer le descrie ca proprietăţi ce pot fi promovate deoarece acestea pot fi promovate din cercul de integrare în ciclul administrativ. Alte proprietăţi nu sunt potrivite pentru configurare administrativă deoarece modificarea acestora poate afecta fluxul de mediere în aşa fel încât modulul de mediere va trebui să fie reimplementat. Integration Designer listează proprietăţile pe care le puteţi alege pentru a promova în proprietăţile promovate ale unei primitive de mediere.

  Puteţi folosi consola administrativă IBM Business Process Manager pentru a modifica valoarea proprietăţilor promovate fără a trebui să reimplementaţi un modul de mediere sau să reporniţi serverul sau modulul.

  În general, fluxurile de mediere utilizează imediat modificările proprietăţii. Cu toate acestea, dacă apar modificări asupra proprietăţii într-o celulă pentru managerul de implementare, acestea au efect in fiecare nod pe măsură ce acestea sunt sincronizate. De asemenea, fluxurile de mediere care sunt în curs continuă să utilizeze valorile anterioare.

  Notă: Din consola administrativă, aveţi posibilitatea să modificaţi doar valorile proprietăţii, nu grupurile acesteia, numele sau tipurile. Dacă doriţi să modificaţi grupurile de proprietăţi, nume sau tipuri, trebuie să utilizaţi Integration Designer.
- Un modul de mediere sau biblioteca dependentă poate defini, de asemenea, subfluxuri. Un subfulx încapsulează un set de primitive de mediere, legate împreună ca o piesă reutilizabilă a logicii de integrare. Pentru a invoca un subflux, se poate adăuga o primitivă în fluxul de mediere.

Implementarea modulelor de mediere

Modulele de mediere sunt create folosind Integration Designer, şi sunt implementate în general pe IBM Business Process Manager înăuntrul unui fişier EAR (enterprise archive).

Aveţi posibilitatea să modificaţi valoarea proprietăţilor promovate în momentul implementării.

Puteţi exporta un modul de mediere din Integration Designer, şi să determinaţi ca Integration Designer să împacheteze modulul de mediere într-un fişier JAR (Java archive), iar acest fişier JAR într-un fişier EAR. Apoi, aveţi posibilitatea să implementaţi apoi fişierul EAR prin instalarea unei noi aplicaţii din consola administrativă.

Modulele de mediere pot fi gândite ca o singură entitate. Totuşi, modulele SCA sunt definite de un număr de fişiere XML memorate într-un fişier JAR.

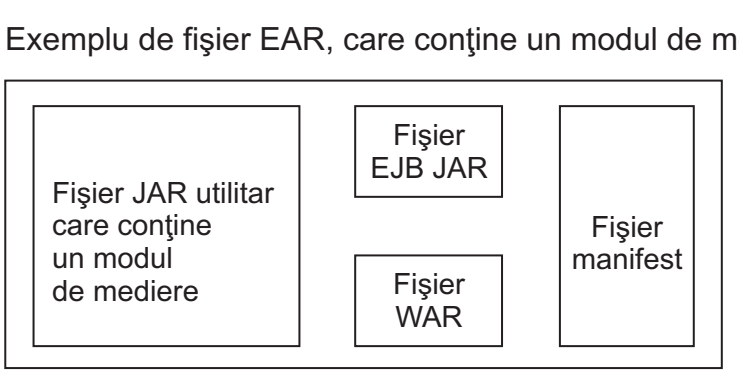


Figura 89. Un exemplu simplificat de fişier EAR care conţine un modul de mediere. Fişierul EAR conţine JAR-uri. Fişierul JAR utilitar conţine un modul de mediere.

Primitive de mediere

Componentele fluxului de mediere operează asupra fluxurilor de mesaje între componentelor serviciului. Capabilităţile componentei de mediere sunt implementate prin *primitivele de mediere*, care implementează tipuri de implementare pentru serviciul standard.

O componentă a fluxului de mediere are una sau mai multe fluxuri. De exemplu, unul pentru cerere şi unul pentru răspuns.

IBM Business Process Manager suportă un set livrat de primitive de mediere, care implementează capabilităţile standard de mediere pentru modulele de mediere sau pentru modulele implementate în IBM Business Process Manager. Dacă aveţi nevoie de anumite capabilităţi de mediere, puteţi să vă dezvoltaţi propriile primitive personalizate de mediere.

O primitivă de mediere defineşte o operaţie de tip “in” care procesează sau manipulează mesajele care sunt reprezentate de SMO-uri (service message objects). O primitivă de mediere poate defini de asemenea o operaţie de tip “out” care trimite mesaje la o altă componentă sau la un alt modul.

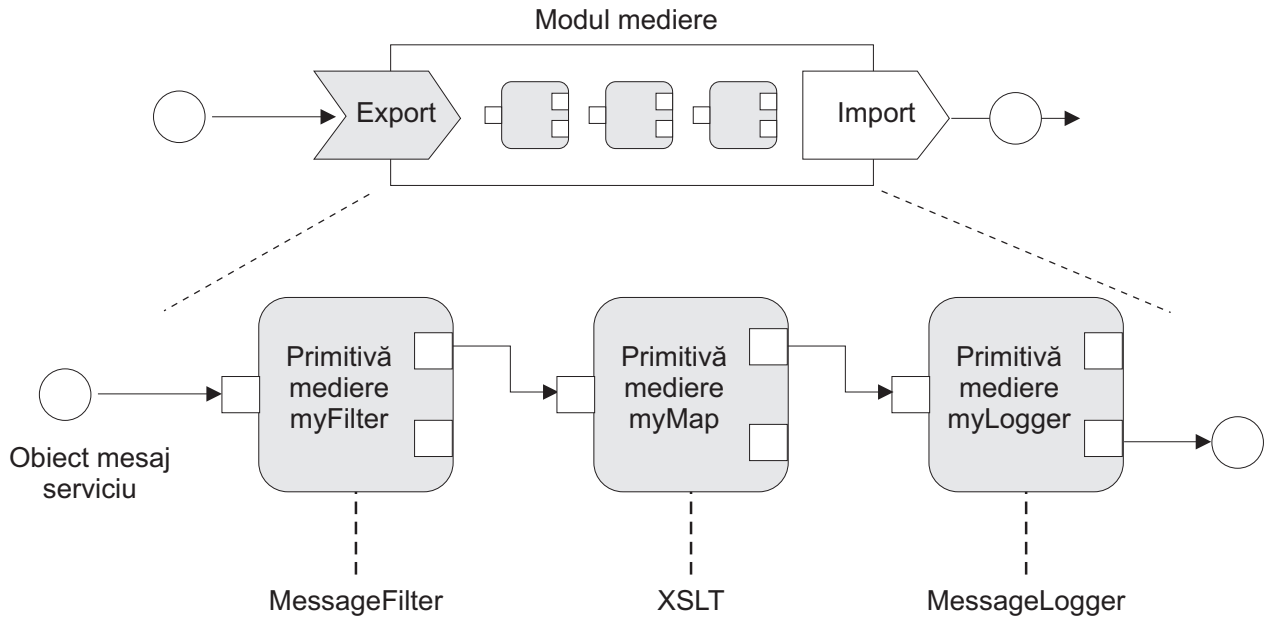


Figura 90. Modulul de mediere care conţine trei primitive de mediere

Puteţi folosi Integration Designer pentru a configura primitivele de mediere şi pentru a le seta proprietăţile. Unele dintre aceste proprietăţi pot fi făcute vizibile pentru administratorul de runtime prin promovarea acestora. Oricare dintre proprietăţile primitivelor de mediere care pot fi promovate pot fi de asemenea proprietăţi dinamice. O proprietate dinamică poate fi înlocuită în momentul execuţiei prin folosirea unui fişier de politică.

Integration Designer vă permite de asemenea să modelaţi grafic şi să asamblaţi componentele fluxului de mediere din cadrul primitivei de mediere, şi să asamblaţi module de mediere sau module din cadrul componentelor fluxului de mediere. Consola administrativă se referă la modulele de mediere şi la modulele ca la module SCA.

Integration Designer permite de asemenea definirea de subfluxuri în module sau a bibliotecilor lor dependente. Un subflux poate conţine orice primitivă de mediere, cu excepţia primitivei Policy Resolution. Un subflux este invocat din fluxul de cerere sau de răspuns, sau dintr-un alt subflux folosind primitiva de mediere Subflux. Proprietăţile promovate într-un sublux prin primitivele de mediere sunt expuse ca proprietăţi în primitivele de mediere Subflow. Apoi, acestea pot fi promovate din nou până când ajung nivelul modulului, moment în care acestea pot fi modificate de administratorul de runtime.

Primitive de mediere suportate

Următorul set de primitive de mediere este suportat de IBM Business Process Manager:

Business Object Map

Transformă mesaje.

- Defineşte transformări pentru mesaje folosind o mapare pentru obiecte business care poate fi refolosită.
- Vă permite să definiţi transformări pentru mesaje în mod grafic folosind editorul pentru maparea cu obiecte business.
- Poate modifica conţinutului unui mesaj.
- Poate transforma tipul unui mesaj de intrare într-un tip diferit de mesaj de ieşire.

Custom Mediation

Vă permite să implementaţi propria logică de mediere în codul Java. Primitiva de mediere personalizată combină flexibilitatea unei primitive de mediere definite de utilizator cu simplitatea unei primitive de mediere predefinite. Puteţi crea transformări complexe şi tipare de rutare astfel:

- Crearea de cod Java.
- Crearea propriilor proprietăţi.
- Adăugarea de terminale noi.

Puteţi apela un serviciu dintr-o primitivă de mediere personalizată, dar primitiva de mediere Service Invoke este proiectată să apeleze servicii şi să asigure funcţionalităţi suplimentare, precum reîncercarea.

Data Handler

Vă permite să modificaţi o parte a mesajului. Este folosit pentru a converti un element al mesajului din format fizic într-o structură logică sau o structură logică într-un forma fizic. Utilizarea primară a primitivei este de a converti un format fizic, precum un şir text dintr-un obiect JMS Text Message, într-o structură de tip Obiect business şi din nou înapoi. Această mediere este frecvent utilizată pentru a:

- Transformarea unei secţiuni a mesajului de intrare dintr-o structură definită în alta - un exemplu de acest fel ar fi cazul în care SMO include o valoare şir care este delimitată prin virgulă şi vreţi să o parsaţi într-un Obiect business specific.
- Altera tipul mesajului – un exemplu ar fi atunci când un export JMS a fost configurat pentru a folosi legare de date de bază JMS, iar în cadrul modulului de mediere dezvoltatorul responsabil cu integrarea decide că conţinutul ar trebui trecut la o anumită structură de BO.

Database Lookup

Modifică mesaje, utilizând informaţii dintr-o bază de date ce sunt furnizate de utilizator.

- Trebuie să configuraţi o bază de date, o sursă de date, precum şi orice setări de autentificare necesare serverului utilizate de primitiva de mediere Database Lookup. Pentru a face acest lucru, ajutaţi-vă de consola administrativă.
- Primitiva de mediere Database Lookup poate citi dintr-o singură tabelă.
- Coloana index specificată trebuie să conţină o valoare unică.
- Datele din coloanele ce conţin valori trebuie să fie ori de un tip schemă XML simplă, fie de un tip schemă XML care extinde tipul de schemă XML simplă.

Endpoint Lookup

Permite rutarea dinamică a cererilor prin căutarea în magazie a punctelor finale de servicii.

- Informaţiile legate de punctul final al serviciului sunt primite de la un WSRR (WebSphere Service Registry and Repository). Registrul WSRR poate fi local sau poate fi la distanţă.
- Faceţi modificările asupra registrului din consola administrativă WSRR.
- IBM Business Process Manager trebuie să ştie ce registru să folosească, şi de aceea, trebuie să creaţi definiţii de acces WSRR folosind consola administrativă IBM Business Process Manager.

Event Emitter

Îmbunătăţeşte monitorizarea prin permiterea trimiterii evenimentelor din interiorul unei componente a fluxului de mediere.

- Puteţi suspenda acţiunea de mediere prin debifarea casetei de bifare.
- Puteţi vizualiza evenimente Event Emitter utilizând browser-ul Common Base Events în IBM Business Process Manager.
- Ar trebui să trimiteţi evenimente doar către un singur punct important într-un flux de mediere, din motive de performanţă.
- Aveţi posibilitatea să definiţi părţile mesajului conţinut de eveniment.
- Evenimentele sunt trimise în formatul Common Base Events şi sunt trimise la un server Common Event Infrastructure.
- Pentru a utiliza pe deplin informaţiile legate de Event Emitter, consumatorii de evenimente trebuie să înţeleagă structura evenimentelor CBE (Common Base Events). Evenimentele CBE au o schemă generală, dar acest lucru nu modelează datele specifice aplicaţiei care sunt conţinute în elementele extinse de date. Pentru a modela elementele extinse de date, uneltele Integration Designer generează un fişier cu definiţii pentru catalogul de evenimente Common Event Infrastructure pentru fiecare dintre primitivele de mediere Event Emitter configurate. Fişierele de definire a catalogului de evenimente sunt artefacte exportate care

sunt furnizate pentru ajutor; nu sunt folosite de runtime-ul Integration Designer sau IBM Business Process Manager. Ar trebui să faceţi referire la fişierele cu definiţii pentru catalogul de evenimente atunci când creaţi aplicaţii care consumă evenimente Event Emitter.

- Puteţi specifica o altă monitorizare din IBM Business Process Manager. De exemplu, puteţi controla ca evenimentele să fie emise din importuri şi exporturi.

Fail Opreşte o anumită cale din flux şi generează o excepţie.

Fan In Ajută la agregarea (combinarea) mesajelor.

- Poate fi folosită doar în combinaţie cu primitiva de mediere Fan Out.
- Împreună, primitivele de mediere Fan Out şi Fan In permit agregarea datelor într-un singur mesaj de ieşire.
- Primitiva de mediere Fan In primeşte mesaje până când se ajunge la un punct de decizie, moment în care este trimis la ieşire un mesaj.
- Contextul partajat ar trebui folosit pentru reţinerea datelor agregate.

Fan Out

Ajută la divizarea şi agregarea (combinarea) mesajelor.

- Împreună, primitivele de mediere Fan Out şi Fan In permit agregarea datelor într-un singur mesaj de ieşire.
- În modul iterativ, primitiva de mediere Fan Out vă permite să iteraţi prin un singur mesaj de intrare care conţine un element repetitiv. Pentru fiecare apariţie a elementului repetitiv este trimis un mesaj.
- Contextul partajat ar trebui folosit pentru reţinerea datelor agregate.

HTTP Header Setter

Oferă un mecanism pentru gestionarea anteturilor în mesajele HTTP.

- Poate crea, seta, copia sau şterge anteturile mesajelor HTTP.
- Poate seta mai multe acţiuni pentru a modifica mai multe anteturi HTTP.

Mapping

Transformă mesaje.

- Vă permite să realizaţi transformări XSL (Extensible Stylesheet Language) sau transformări Business Object Map.
- Puteţi transforma mesajele folosind o transformare XSLT 1.0 sau XSLT 2.0 sau o transformare Business Object Map. Transformările XSL operează pe o serializare XML a mesajului, în timp ce transformarea Business Object Map operează asupra SDO (Service Data Objects).

Message Element Setter

Oferă un mecanism simplu pentru setarea conţinutului mesajelor.

- Poate modifica, adăuga sau şterge elemente din mesaj.
- Nu modifică tipul de mesaj.
- Datele din coloanele ce conţin valori trebuie să fie ori de un tip schemă XML simplă, fie de un tip schemă XML care extinde tipul de schemă XML simplă.

Message Filter

Rutează mesaje pe căi diferite, în funcţie de conţinutul acestora.

- Puteţi suspenda acţiunea de mediere prin debifarea casetei de bifare.

Message Logger

Înregistrează mesajele într-o bază de date relaţională sau prin intermediul propriului jurnalizator personalizat. Mesajele sunt memorate sub formă de XML, şi de aceea, datele pot fi procesate după aceea de aplicaţiile care suportă XML.

- Puteţi suspenda acţiunea de mediere prin debifarea casetei de bifare.
- Schema bazei de date relaţionale (structura tabelei) este definită de IBM.
- În mod implicit, primitiva de mediere Message Logger foloseşte baza de date Common. Mediul runtime mapează sursa de date la **jdbc/mediation/messageLog** pe baza de date Comună.

- Puteţi seta clasele de implementare Handler pentru a personaliza comportamentul jurnalizatorului personalizat. Opţional, puteţi furniza clase de implementare Formatter, clase de implementare Filter, sau ambele pentru a personaliza comportamentul jurnalizatorului personalizat.

MQ Header Setter

Oferă un mecanism pentru gestionarea anteturilor în mesajele MQ.

- Poate crea, seta, copia sau şterge anteturile mesajelor MQ.
- Poate seta mai multe acţiuni pentru a modifica mai multe anteturi MQ.

Policy Resolution

Permite configurarea dinamică a cererilor, prin căutarea punctelor finale ale serviciului şi a fişierelor de politică asociate, într-o magazie.

- Aveţi posibilitatea să utilizaţi un fişier de politică pentru a înlocui dinamic proprietăţile promovate ale altor primitive de mediere.
- Informaţiile punctului final al serviciului şi informaţiile legate de politică sunt extrase dintr-un WSRR (WebSphere Service Registry and Repository). Registrul WSRR poate fi local sau poate fi aflat la distanţă.
- Faceţi modificările asupra registrului din consola administrativă WSRR.
- IBM Business Process Manager trebuie să ştie ce registru să folosească, şi de aceea, trebuie să creaţi definiţii de acces WSRR folosind consola administrativă IBM Business Process Manager.

Service Invoke

Apelează un serviciu din interiorul unui flux de mediere, mai degrabă decât să aştepte până la sfârşitul unui flux de mediere şi folosirea mecanismului callout.

- Dacă serviciul returnează un defect, puteţi reîncerca acelaşi serviciu sau să apelaţi un alt serviciu.
- Primitiva de mediere Service Invoke este una puternică care poate fi folosită pe cont propriu pentru apeluri simple ale serviciului, sau în combinaţie cu alte primitive de mediere pentru medieri mai complexe.

Set Message Type

În timpul dezvoltării de integrare, vă permite să trataţi câmpurile mesajului care sunt tipizate slab, ca şi pe cele care sunt tipizate tare. Un câmp este tipizat slab dacă poate conţine mai multe tipuri de date. Un câmp este tipizat tare dacă tipul şi structura sa internă sunt cunoscute.

- La runtime, medierea Setare tip mesaj permite verificarea potrivirii între conţinutul unui mesaj şi tipurile de date pe care le aşteptaţi.

SOAP Header Setter

Oferă un mecanism pentru gestionarea anteturilor în mesajele SOAP.

- Poate crea, seta, copia sau şterge anteturile mesajelor SOAP.
- Poate seta mai multe acţiuni pentru a modifica mai multe anteturi SOAP.

Stop Opreşte o anumită cale din flux fără a genera o excepţie.

Type Filter

Vă permite să direcţionaţi mesajele pe o cale diferită a fluxului, în funcţie de tipul acestora.

Extragere WebSphere eXtreme Scale

Puteţi extrage informaţii dintr-un mediu cache server eXtreme Scale.

- Puteţi căuta valori în cache şi le puteţi păstra ca elemente în mesaj folosind un index.
- Combinând primitivele de mediere Stocare şi Extragere eXtreme Scale, puteţi memora în cache răspunsul de la un sistem back-end. Cererile viitoare nu vor necesita acces la acel sistem de back-end.
- Trebuie să creaţi definiţii eXtreme Scale utilizând consola administrativă WebSphere ESB, astfel încât puteţi specifica serverul eXtreme Scale de utilizat.

WebSphere eXtreme Scale Store

Puteţi stoca informaţii într-un mediu cache server eXtreme Scale.

- Puteţi stoca informaţii într-un cache eXtreme Scale utilizând un index sau un obiect.

- Combinând primitivele de mediere eXtreme Scale Store şi Retrieve, puteţi folosi primitiva de mediere Store pentru stocarea datelor în memoria cache şi puteţi folosi primitiva de mediere Retrieve pentru a extrage date stocate anterior în memoria cache.
- Trebuie să creaţi definiţii eXtreme Scale utilizând consola administrativă WebSphere ESB, astfel încât puteţi specifica serverul eXtreme Scale de utilizat.

Rutarea dinamică

Puteţi ruta mesaje în diverse moduri utilizând punctele finale definite la timpul integrării sau puncte finale determinate, dinamic, la momentul rulării.

Rutarea dinamică acoperă două cazuri de rutare a mesajelor:

- Rutarea mesajelor în cazul în care fluxul este dinamic, dar toate punctele finale posibile sunt predefinite într-un modul SCA (Service Component Architecture).
- Rutarea mesajelor în cazul în care fluxul este dinamic şi selecţia punctului final este, de asemenea, dinamică. Punctele finale ale serviciului sunt selectate dintr-o sursă externă la momentul rulării

Selectarea dinamică a punctelor finale

Momentul execuţiei are capabilitatea de a ruta mesajele de cerere şi de răspuns către o adresă finală identificată printr-un element din antetul mesajului. Acest element din antetul mesajului poate fi actualizat prin intermediul primitivelor de mediere, într-un flux de mediere. Adresa finală ar putea fi actualizată cu informaţii dintr-un registru, o bază de date, sau cu informaţii din mesajul în sine. Rutarea unui mesaj de răspuns se aplică doar atunci când răspunsul este trimis de un export JAX-WS al serviciului web.

Pentru ca la momentul execuţiei să fie implementată rutarea dinamică pentru o cerere sau pentru un răspuns, modulul SCA trebuie să aibă setată proprietatea Utilizare punct final dinamic dacă este setat în antetul mesajului. Dezvoltatorii care se ocupă cu integrarea pot seta proprietatea >Utilizare punct final dinamic dacă este setat în antetul mesajului sau o pot promova (o fac vizibilă la momentul execuţiei), astfel încât administratorul din timpul rulării să o poată seta. Aveţi posibilitatea să vizualizaţi proprietăţile modulului în fereastra Proprietăţi modul. Pentru a vedea fereastra, faceţi clic pe **Aplicaţii > Module SCA > Proprietăţi Modul**. Dezvoltatorul care se ocupă cu integrarea oferă proprietăţilor promovate aliasuri, iar acestea sunt numele afişate în consola administrativă.

Registru

Puteţi folosi WSRR (IBM WebSphere Service Registry and Repository) pentru a memora informaţiile legate de punctul final al serviciului, iar apoi să creaţi module SCA pentru a extrage puncte finale din magazia WSRR.

Atunci când dezvoltaţi module SCA, folosiţi primitiva de mediere Endpoint Lookup pentru a permite unui flux de mediere să interogheze un registru WSRR pentru a obţine un punct final pentru serviciu sau un set de puncte finale pentru serviciu. Dacă un modul SCA modul extrage un set de puncte finale atunci acesta trebuie să folosească o altă primitivă de mediere pentru a o selecta pe cea preferată.

Controlul politicii de mediere a cererilor de servicii

Puteţi utiliza politici de mediere pentru a controla fluxurile de mediere dintre solicitanţii serviciului şi furnizorii de servicii.

Puteţi controla fluxurile de mediere utilizând politici de mediere memorate în IBM WebSphere Service Registry and Repository (WSRR). Implementarea gestiunii politicii serviciului în WSRR este bazată pe WS-Policy (Web Services Policy Framework).

Pentru a controla cererile de servicii utilizând politici de mediere, trebuie să aveţi module SCA (Service Component Architecture) dorite şi documente ale politicii de mediere în registrul dumneavoastră WSRR.

Cum se face ataşarea unei politici de mediere la o cerere de serviciu

Atunci când dezvoltaţi un modul SCA care trebuie să facă uz de o politică de mediere, trebuie să includeţi în fluxul de mediere o primitivă de mediere Policy Resolution. În timpul rulării, primitiva de mediere Policy Resolution obţine informaţii despre politica de mediere din registru. Prin urmare, un modul SCA trebuie să conţină o componentă flux de mediere pentru a suporta controlul politicii de mediere a cererilor de servicii.

În registru, puteţi ataşa una sau mai multe politici de mediere la un modul SCA sau la un serviciu ţintă utilizat de către modulul SCA. Politicile de mediere ataşate pot fi utilizate (sunt în domeniu) pentru toate mesajele serviciului procesate de către acel modul SCA. Politicile de mediere pot avea ataşamente de politică care definesc condiţii. Condiţiile politicii de mediere permit diferitelor politici de mediere să se aplice în diferite contexte. În plus, politicile de mediere pot avea clasificări care pot fi utilizate pentru a specifica o stare de guvernare.

WebSphere Service Registry and Repository

Produsul WSRR (WebSphere Service Registry and Repository) vă permite să memoraţi, accesaţi şi să gestionaţi informaţiile legate de punctele finale ale serviciului şi de politicile de mediere. Puteţi utiliza WSRR pentru a face ca aplicaţiile serviciului dvs. să fie mult mai dinamice şi mult mai adaptabile la modificările aduse condiţiilor de business.

Introducere

Fluxurile de mediere pot folosi WSRR pe post de maşină de căutare dinamică, oferind informaţii legate de punctele finale sau de politicile de mediere ale serviciului.

Pentru a configura accesul la WSRR, creaţi documente de definiţie WSRR folosind consola administrativă. Alternativ, aveţi posibilitatea să utilizaţi comenzile de administrare WSRR din clientul script wsadmin. Definiţiile WSRR şi proprietăţile acestora de conexiune reprezintă mecanismul de conectare la o instanţă a unui registru, şi de extragere a punctului final sau a politicii de mediere pentru un serviciu.

Puncte finale pentru serviciu

Puteţi folosi WSRR pentru a memora informaţii despre serviciile pe care le folosiţi deja, pentru cele pe care plănuiţi să le folosiţi sau pentru cele de care vreţi să fiţi conştient. Aceste servicii pot fi în sistemele dvs, sau în ale sisteme. De exemplu, aplicaţia ar putea folosi WSRR la localizarea celui mai corespunzător serviciu care îi satisface nevoile funcţionale şi de performanţă.

Atunci când dezvoltaţi un modul SCA care are nevoie să acceseze punctele finale ale serviciului din WSRR, trebuie să includeţi o primitivă de mediere Endpoint Lookup în fluxul de mediere. În momentul rulării, primitiva de mediere Endpoint Lookup obţine punctele finale ale serviciului din registru.

Politici de mediere

De asemenea, puteţi folosi WSRR pentru a memora informaţii legate de politica de mediere. Politicile de mediere vă pot ajuta să controlaţi cererile de servicii prin suprascrierea dinamică a proprietăţilor modulului. Dacă WSRR conţine politici de mediere care sunt ataşate la un obiect care reprezintă fie modulul dvs. SCA, fie serviciul dvs. ţintă, atunci politicile de mediere ar putea suprascrie proprietăţile modulului. Dacă doriţi ca în diferite contexte să se aplice politici de mediere diferite, puteţi crea condiţii pentru aceste politici.

Notă: Politicile de mediere sunt preocupate de controlul fluxurilor de mediere, nu de securitate.

Atunci când dezvoltaţi un modul SCA care trebuie să facă uz de o politică de mediere, trebuie să includeţi în fluxul de mediere o primitivă de mediere Policy Resolution. În momentul rulării, primitiva de mediere Policy Resolution obţine informaţiile legate de politica de mediere din registru.

WebSphere eXtreme Scale

Utilizând produsul WebSphere eXtreme Scale (eXtreme Scale) puteţi asigura un sistem de memorare în cache pe care îl puteţi integra cu o aplicaţieIBM Business Process Manager. Utilizând eXtreme Scale cu IBM Business Process Manager puteţi îmbunătăţi timpul de răspuns şi fiabilitatea serviciului şi asigura funcţionalitate de integrare suplimentară.

eXtreme Scale acţionează ca o grilă de date elastică, scalabilă, în memorie. Grila de date memorează în cache, partiţionează, replică şi gestionează dinamic datele aplicaţiei şi logica operaţională pe mai multe servere. Cu eXtreme Scale, puteţi, de asemenea, să obţineţi calitate a serviciilor precum integritate tranzacţională, disponibilitate înaltă şi timpi de răspuns predictibili.

Puteţi utiliza fluxuri de mediere pentru a accesa funcţia de memorare în cache eXtreme Scale inclusiv primitivele de mediere WebSphere eXtreme Scale din fluxul dumneavoastră. La dezvoltarea unui modul SCA (Service Component Architecture) care are nevoie să stocheze informaţii într-o memorie cache eXtreme Scale, trebuie să includeţi primitiva de mediere WebSphere eXtreme Scale Store în fluxul de mediere. Dacă vreţi să extrageţi informaţii dintr-o memorie cache eXtreme Scale, trebuie să includeţi primitiva de mediere WebSphere eXtreme Scale Retrieve. Combinând cele două primitive de mediere într-un flux de mediere, puteţi memora în cache răspunsul de la un sistem back-end, astfel încât cererile viitoare să poată extrage răspunsul din cache.

Pentru a configura accesul la eXtreme Scale, trebuie să creaţi o definiţie WebSphere eXtreme Scale utilizând consola administrativă. Alternativ, puteţi utiliza comenzile de administrare WebSphere eXtreme Scale din clientul de scriptare wsadmin. O definiţie eXtreme Scale este mecanismul utilizat de primitivele de mediere WebSphere eXtreme Scale Retrieve şi Store pentru conectarea la un server eXtreme Scale.

Clienţi ai serviciului de mesaje

Clienţii pentru Serviciul de mesagerie sunt disponibili pentru C/C++ şi .NET pentru a permite aplicaţiilor care nu sunt de tip Java să se conecteze la magistrala ESB (Enterprise Service Bus).

Message Service Clients for C/C++ and .NET oferă un API numit XMS care are acelaşi set de interfeţe ca şi API-ul JMS (Java Message Service). Message Service Client for C/C++ conţine două implementări ale XMS, una de folosit de aplicaţiile C şi una de folosit de aplicaţiile C++. Message Service Client for .NET conţine o implementare complet gestionată a XMS, care poate fi folosită de orice limbaj compatibil cu .NET.

Puteţi obţine Clienţi ai Serviciului de Mesaje pentru .NET din http://www-01.ibm.com/support/docview.wss?rs=0&q1=IA9H&uid=swg24011756&loc=en_US&cs=utf-8&cc=us&lang=en

Puteţi obţine Clienţi ai Serviciului de Mesaje pentru C/C++ din http://www-01.ibm.com/support/docview.wss?rs=0&q1=ia94&uid=swg24007092&loc=en_US&cs=utf-8&cc=us&lang=en.

Puteţi instala şi folosi şi suportul pentru clientul Java EE de la WebSphere Application Server Network Deployment, inclusiv client servicii web, EJB Client, şi JMS Client.