

IBM Business Process Manager
Versiune 8 Ediție 5

*Privire generală asupra IBM
Business Process Manager*

IBM

Cărțile PDF și centrul de informare

Cărțile PDF sunt furnizate pentru tipărire și pentru lectură online. Pentru cele mai recente informații, vedeți centrul de informare online.

Setul de cărți PDF include același conținut ca și centrul de informare. Unele legături din cărțile PDF au fost ajustate pentru folosirea în centrele de informare și este posibil să nu funcționeze corect.

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

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

Cuprins

Cărțile PDF și centrul de informare iii

Capitolul 1. Inițiere în IBM Business Process Manager 1

Prezentarea produsului	1
Configurațiile IBM Business Process Manager	3
Capabilități de configurare IBM Business Process Manager	3
Magazia Process Center	4
Process Server și mediile runtime	5
Medii de creație	6
Unelte de administrare	7
Accesibilitatea în IBM Business Process Manager	9
Disponibilitatea limbilor naționale în IBM Business Process Manager	9
Privire generală asupra BPM (Business Process Management)	10
Privire generală asupra modelării de proces	11
Dezvoltarea procesului cu Process Center	12
Aplicații de proces: Privire generală	13
Rularea și depanarea proceselor	17
Instalarea și gestionarea aplicațiilor de proces	18
Crearea, accesarea și încorporarea serviciilor	20
Accesarea serviciilor externe unei aplicații	20
Crearea sau apelarea unui serviciu web	25
Aflați mai multe despre conceptele cheie	26
Versionarea	27
Versionarea aplicațiilor de proces	27
Versionarea modulelor și bibliotecilor	28
Module și biblioteci asociate cu aplicații de proces sau cu truse de unelte	29
Convenții de numire	29
Convenții de numire pentru implementarea pe servere Process Center	30
Convenții de numire pentru implementarea Process Server	32
Legările dependente de versiune	33
Invocarea dinamică dependentă de versiune	35
Implementarea aplicațiilor de proces cu module și proiecte Java	36
Implementarea aplicațiilor de proces cu ajutorul regulilor operaționale și a selectoarelor	36
Obiecte de configurație	36
Arhitectura de implementare	36
Celulele	36
Serverele	37
Serverele autonome	37
Cluster-ele	37
Profilurile	38
Managerii de implementare	39
Nodurile	40
Nodurile gestionate	40
Nodurile negestionate	40
Agenții de nod	40

Considerente privind numirea pentru profiluri, noduri, server, gazde și celule	40
BPMN 2.0	44
Definiții de proces operațional (BPD-uri)	46
Legările	47
Privire generală asupra legării de export și import	49
Configurarea legărilor de export și import	53
Transformarea formatului de date în importuri și exporturi	53
Selectori de funcții în legări de export	57
Tratarea defectelor	59
Interoperabilitatea între modulele SCA și serviciile Open SCA	63
Tipuri de legări	65
Selectarea legărilor corespunzătoare	65
Legări SCA	67
Legări de serviciu Web	67
Legări HTTP	91
Legări EJB	98
Legări EIS	104
Legări JMS	110
Legările Generic JMS	117
Legări JMS WebSphere MQ	124
Legări WebSphere MQ	130
Limitări ale legărilor	138
Obiectele business	140
Definirea obiectelor business	140
Lucrul cu obiecte business	141
Obiectele business speciale	143
Modul de parsare a obiectelor business	144
Considerații la alegerea modului de parsare a obiectului business	144
Beneficii ale utilizării modului de parsare lenș față de vioi	145
Considerații de migrare și de dezvoltare de aplicații	145
Relații	147
Serviciu de relații	149
Manager de relații	149
Relații în medii Network Deployment	150
API-uri ale serviciului de relații	150
Magistrala ESB (Enterprise Service Bus) din IBM Business Process Manager	150
Conectarea serviciilor printr-o magistrală pentru serviciile întreprinderii	150
Infrastructura de mesagerie ESB (Enterprise Service Bus)	152
Gazde destinație mesagerie sau coadă	152
Furnizori JDBC	152
Magistrala de integrare servicii pentru IBM Business Process Manager	153
Aplicații de servicii și module de servicii	153
Importurile și legările de import	154
Exporturile și legările de export	155
Module de mediere	156
Primitive de mediere	158

Rutarea dinamică	163
Controlul politicii de mediere a cererilor de servicii	164
WebSphere Service Registry and Repository	164
WebSphere eXtreme Scale	165
Clienți ai serviciului de mesaje.	165

**Capitolul 2. Aflați mai multe despre
conceptele cheie 167**

Versionarea	167
Versionarea aplicațiilor de proces	167
Versionarea modulelor și bibliotecilor	168
Module și biblioteci asociate cu aplicații de proces sau cu truse de unelte.	169
Convenții de numire	169
Convenții de numire pentru implementarea pe servere Process Center	170
Convenții de numire pentru implementarea Process Server	173
Legările dependente de versiune	174
Invocarea dinamică dependentă de versiune	176
Implementarea aplicațiilor de proces cu module și proiecte Java	176
Implementarea aplicațiilor de proces cu ajutorul regulilor operaționale și a selectoarelor	176
Obiecte de configurație	176
Arhitectura de implementare	177
Celulele	177
Serverele	177
Serverele autonome	177
Cluster-ele.	178
Profilurile	178
Managerii de implementare.	180
Nodurile	180
Nodurile gestionate	180
Nodurile negestionate	180
Agenții de nod	181
Considerente privind numirea pentru profiluri, noduri, server, gazde și celule	181
BPMN 2.0.	184
Definiții de proces operațional (BPD-uri)	187
Legările	188
Privire generală asupra legării de export și import	190
Configurarea legărilor de export și import	193
Transformarea formatului de date în importuri și exporturi	193
Handler-e de date	194
Legări de date.	195
Selectorii de funcții în legări de export	197
Tratarea defectelor	199
Cum sunt tratate defectele în legări de export Modul în care defectele sunt tratate în legările de import	201
Interoperabilitatea între modulele SCA și serviciile Open SCA.	203
Tipuri de legări	205
Selectarea legărilor corespunzătoare	205
Legări SCA	207
Legări de serviciu Web	207
Privire generală asupra legărilor de serviciu Web	207

Propagarea antetului SOAP.	208
Propagarea antetului Transport.	211
Lucrul cu legări de servicii web (JAX-WS)	213
Atașamente în mesajele SOAP.	216
Utilizarea legării stilului documentului WSDL cu mesaje multiple	230
Legări HTTP	231
Privire generală asupra legărilor HTTP	232
Anteturi HTTP	233
Legări de date HTTP	235
Legări EJB	238
Legările de import EJB	238
Legările de export EJB	239
Proprietățile legărilor EJB	241
Legări EIS.	244
Privire generală asupra legărilor EIS	245
Caracteristici cheie ale legărilor EIS	246
Proprietăți dinamice JCA Interaction Spec și Connection Spec	248
Clienți externi cu legări EIS	249
Legări JMS	250
Privire generală asupra legărilor JMS	250
Integrare JMS și adaptoare de resurse.	251
Legări de import și export JMS	251
Anteturi JMS	253
Schema de corelare a destinațiilor de răspuns dinamice temporare JMS	254
Clienți externi.	255
Depanare legări JMS	256
Tratarea excepțiilor	257
Legările Generic JMS	257
Privire generală asupra legărilor Generic JMS	257
Caracteristici cheie ale legărilor JMS generic	260
Anteturi Generic JMS	261
Depanarea legărilor JMS generic	262
Tratarea excepțiilor	263
Legări JMS WebSphere MQ	264
Privire generală asupra legărilor WebSphere MQ JMS	264
Caracteristici cheie ale legărilor WebSphere MQ JMS	266
Anteturi JMS	267
Clienți externi.	268
Depanarea legărilor WebSphere MQ JMS	269
Tratarea excepțiilor	269
Legări WebSphere MQ	270
Privire generală asupra legărilor WebSphere MQ.	270
Caracteristici cheie ale unei legări WebSphere MQ.	272
Anteturi WebSphere MQ	274
Adăugarea statică a MQCIH într-o legare WebSphere MQ	276
Clienți externi.	276
Depanarea legărilor WebSphere MQ	277
Tratarea excepțiilor	278
Limitări ale legărilor.	278
Limitările legării MQ	278
Limitările legărilor JMS, MQ JMS și Generic JMS	279
Obiectele business	280

Definirea obiectelor business	280	Infrastructura de mesagerie ESB (Enterprise Service Bus)	292
Lucrul cu obiecte business	281	Gazde destinație mesagerie sau coadă	292
Obiectele business speciale	283	Depozite de date	292
Modul de parsare a obiectelor business	284	Furnizori JDBC	293
Considerații la alegerea modului de parsare a obiectului business	284	Magistrala de integrare servicii pentru IBM Business Process Manager	293
Beneficii ale utilizării modului de parsare lenes față de viori	285	Aplicații de servicii și module de servicii	293
Considerații de migrare și de dezvoltare de aplicații	285	Importurile și legările de import	294
Relații	287	Exporturile și legările de export	295
Serviciu de relații	289	Module de mediere	296
Manager de relații	289	Primitive de mediere.	298
Relații în medii Network Deployment	290	Rutarea dinamică.	303
API-uri ale serviciului de relații	290	Controlul politicii de mediere a cererilor de servicii	303
Magistrala ESB (Enterprise Service Bus) din IBM		WebSphere Service Registry and Repository	304
Business Process Manager	290	WebSphere eXtreme Scale	305
Conectarea serviciilor printr-o magistrală pentru serviciile întreprinderii	290	Clienți ai serviciului de mesaje.	305

Capitolul 1. Inițiere în IBM Business Process Manager

Înțelegeți ce capacități oferă IBM® Business Process Manager pentru gestionarea proceselor operaționale și relațiile dintre diferitele faze ale gestionării proceselor operaționale, cum ar fi crearea și implementarea aplicațiilor de proces.

Aplicația de proces reprezintă containerul fundamental pentru procese și componentele lor în IBM Business Process Manager. Proiectanții procesului creează aplicații de proces în mediile de creație și ar putea include servicii, taskuri și artefacte necesare pentru a suporta execuția.

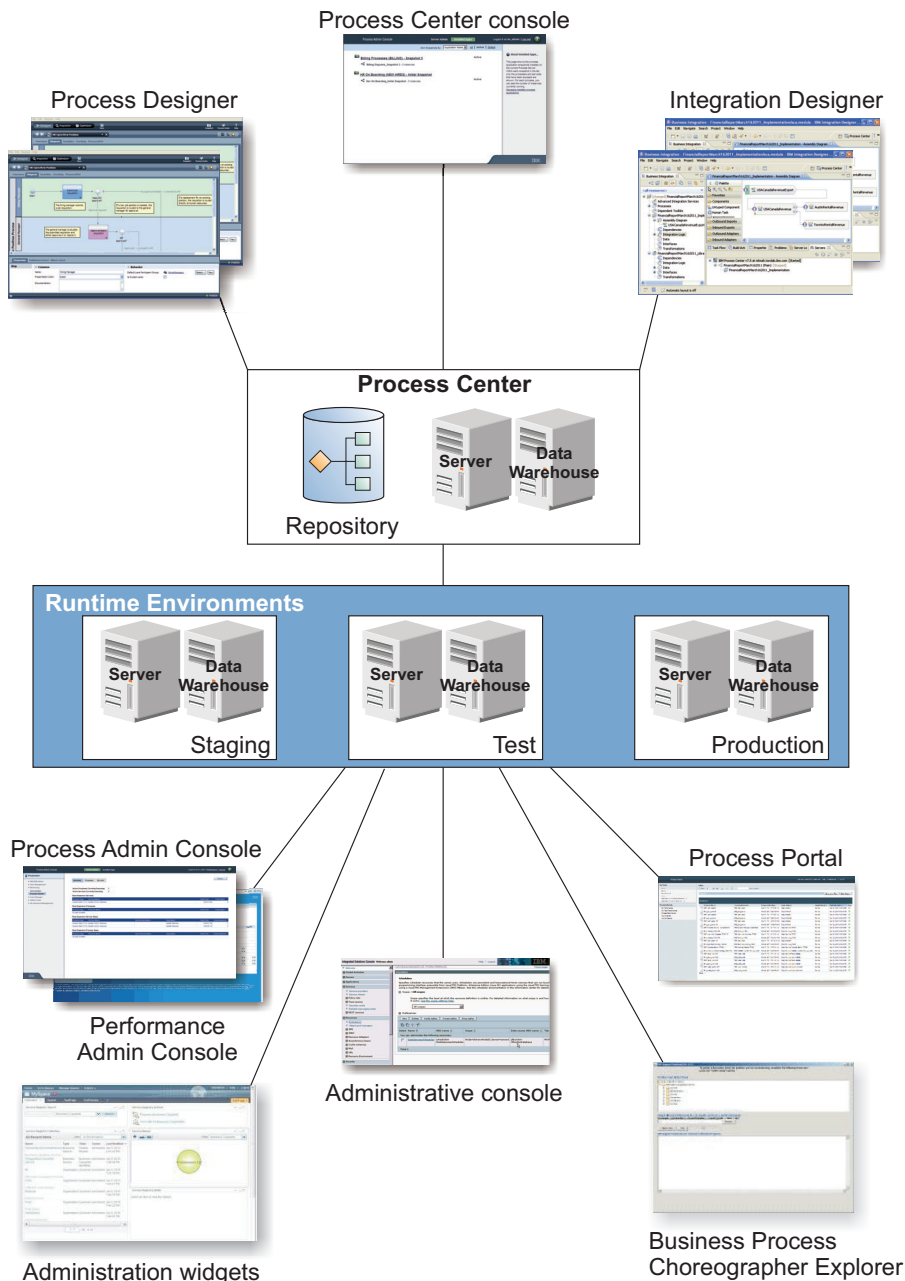
Serviciile de integrare avansată sunt implementate în IBM Integration Designer și asociate cu aplicații de proces. Din Process Center, aplicațiile de proces sunt implementate în Process Server, care reprezintă mediul runtime al proceselor pentru IBM Business Process Manager.

Similar, procesele automatizate create în Integration Designer pot utiliza servicii umane care au fost dezvoltate în IBM Process Designer.

Prezentarea produsului

IBM Business Process Manager este o platformă de procese operaționale comprehensivă, care furnizează vizibilitate completă și înțelegere a gestionării proceselor operaționale. Furnizează unelte și un mediu runtime pentru proiectare, execuție, monitorizare și optimizare proces, împreună cu suport integrare sistem de bază. Produsul poate fi configurat să suporte diverse niveluri de complexitate și implicare cu gestiunea procesului operațional.

Componentele IBM Business Process Manager furnizează o magazie unificată BPM; unelte pentru autori, administratori și utilizatori; și o platformă runtime. Produsul poate fi configurat să suporte diverse niveluri de complexitate și implicare cu gestiunea procesului operațional. Următoarea diagramă ilustrează o configurație IBM Business Process Manager obișnuită:



- Din mediile de creație IBM Process Designer și IBM Integration Designer, dezvoltatorii se conectează la IBM Process Center. Din oricare dintre aceste unelte de dezvoltare aplicație bazate pe GUI, dezvoltatorii pot crea, testa, depana și implementa aplicații operaționale. Alegeți o unealta sau alta, în funcție de tipul aplicației pe care o dezvoltați. Ar putea exista și cazuri în care utilizarea ambelor unelte aduce avantaje semnificative.
- În mediile de creație Process Designer și Integration Designer, proiectanții de procese și servicii creează aplicații de proces implementabile și truse de unelte reutilizabile. Aplicațiile de proces conțin modele de procese și implementări de servicii, inclusiv fișierele de suport care sunt necesare. Aplicațiile proces sunt stocate în magazia Process Center astfel încât să poată fi partajate.
- Process Center include două servere, serverul Process Center și serverul Performance Data Warehouse. Aceste servere permit dezvoltatorilor care lucrează în Process Designer să își ruleze aplicațiile proces și să își depoziteze datele de performanță pentru testarea și 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 funcții administrative. Din Consola Process Center, administratorii instalează aplicații proces care sunt pregătite pentru intermediere, testare sau producție pe serverele Process. Administratorii pot gestiona de asemenea instanțe care rulează ale aplicațiilor proces din medii configurate.
- Instalați aplicații proces pe un Process Server pentru intermediere, testare și producție. 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 Process Admin Console și din Performance Admin Console, administratorii pot gestiona și menține toate serverele runtime. Utilizați Process Admin Console pentru a gestiona serverul Process Center și serverele Process din mediile dumneavoastră runtime. Utilizați Performance Admin Console pentru a identifica gâtuirile de performanță și pentru a captura date de instrumentare pentru analiza suplimentară.
- Utilizați consola administrativă pentru a crea și gestiona obiecte precum resurse, aplicații și servere. În plus, utilizați consola administrativă pentru a vizualiza mesaje produs.
- Utilizați Business Space pentru a crea spații de afaceri personalizate care furnizează widget-uri pentru monitorizarea sau administrarea diferitelor aspecte ale sistemului dumneavoastră. De exemplu, puteți monitoriza activități de afaceri, servicii și sănătate sistem sau administra politici de mediere și calendare de afaceri. Puteți crea de asemenea un spațiu operațional cu widget-urile Human Task Management și îl puteți utiliza pentru a participa la procesele operaționale.
- Utilizând Process Portal, participanții 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 producție.
- Gestionați instanțele de proces Business Process Execution Language (BPEL) din Business Process Choreographer Explorer sau din Business Space.

Configurațiile IBM Business Process Manager

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

Tabela 1. Configurațiile IBM Business Process Manager

Configurare	Fază
Avansat	<p>Transformare</p> <p>Set complet de capabilități de gestiune procese operaționale</p> <ul style="list-style-type: none"> • Suport extins pentru automatizare proces de volum înalt • Componente SOA încorporate pentru integrare, orchestrare extensivă de servicii la nivel de întreprindere
Standard	<p>Program</p> <p>Configurat pentru proiecte de gestiune procese operaționale (BPM) tipice</p> <ul style="list-style-type: none"> • Pentru programe de îmbunătățire proiecte multiple, cu implicare operațională înaltă • Suport integrare sistem de bază • Timp scurt de valorificare și productivitate pe utilizator îmbunătățită
Expres	<p>Proiect</p> <p>Configurat pentru primul proiect BPM</p> <ul style="list-style-type: none"> • Timp-la-valoare scurt: productivitate utilizator îmbunătățită • Preț mic • Instalare și configurare ușoară

Capabilități de configurare IBM Business Process Manager

Înțelegeți ce produse și capabilități oferă IBM pentru gestionarea proceselor operaționale și alegeți-l pe cel potrivit pentru compania dumneavoastră.

IBM Business Process Manager reprezintă o platformă BPM singulară care combină capabilitățile centrate pe integrare și cele umane într-un produs unificat. Diferite configurații ale produsului sunt disponibile pentru diferiți utilizatori și îndeplinesc diferite necesități în întreprindere. Configurațiile produselor pot fi combinate pentru creațiile în colaborare și mediile runtime implementate pe rețea.

Tabela 2. Capabilități de configurare IBM Business Process Manager

Capabilitate	Adv.	Std.	Expres
Execuție compatibilă WebSphere Lombardi Edition	X	X	X
Process Designer (BPMN)	X	X	X
Editare în colaborare / Redare imediată	X	X	X
Interfețe utilizator "coach proces" interactive	X	X	X
Reguli de proces bazate pe ODM	X	X	X
Process Portal	X	X	X
Monitorizare și raportare în timp real	X	X	X
Optimizator și analize de performanță	X	X	X
Performance Data Warehouse	X	X	X
Process Center / magazie de aseturi partajate	X	X	X
Utilizatori finali și autori de procese nelimitați	X	X	200 utilizatori / 3 autori
Disponibilitate înaltă: funcționare în cluster și nuclee nelimitate	X	X	<ul style="list-style-type: none"> • producție pe 4 nuclee • dezvoltare pe 2 nuclee • Niciun cluster
Execuție compatibilă WebSphere Process Server	X		
Integration Designer (BPEL / SOA)	X		
ESB (Enterprise Service Bus) încorporat	X		
Suport pentru tranzacții	X		
Adaptoare de integrare	X		
Interfață utilizator Flexible Business Space	X		
Suport avansat pe platforme (Linux System z, IBM AIX, Solaris)	X	X	*Notă

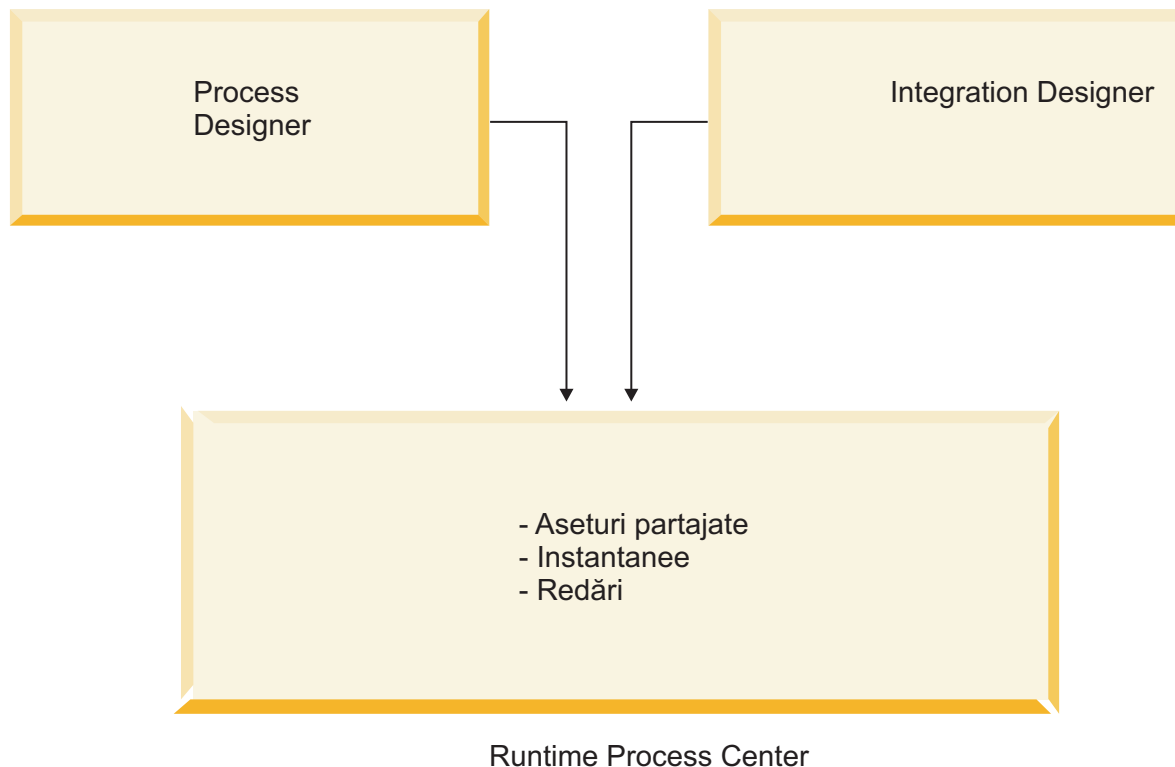
Notă: IBM BPM Express este suportat pe AIX doar pentru clienții IBM Master Data Management (MDM).

Magazia Process Center

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

Process Center este o componentă software care rulează ca un server unde Process Designer și Integration Designer împart aseturi, permițându-le de fapt să dezvolte procese operaționale cooperativ într-o manieră foarte interactivă.

În următoarea diagramă, puteți vedea mai multe componente înrudite care vă permit împreună să construiți procese operaționale complexe.



Consola Process Center oferă uneltele necesare pentru menținerea magaziei.

- Din consola Process Center, puteți crea aplicații de proces și truse de unelte și acorda altor utilizatori acces la acele aplicații de proces și truse de unelte.
- În mediile de creație, puteți crea modele de proces, servicii și alte aseturi în aplicațiile de proces.
- Process Center include două servere, serverul Process Center și serverul Performance Data Warehouse. Serverul Process Center permite dezvoltatorilor care lucrează în Process Designer să-și ruleze propriile aplicații de procesare și să depoziteze datele de performanță pentru testare și redare în timpul eforturilor de dezvoltare. Performance Data Warehouse extrage datele urmărite de la Process Server sau serverul Process Center la intervale regulate.
- Din consola Process Center, administratorii instalează aplicații de proces care sunt gata pentru testare sau producție pe serverele Process în acele medii.
- Din consola Process Center, administratorii gestionează instanțele de aplicații de proces care rulează în mediile configurate.

Consola Process Center oferă o locație convenabilă în care să creați și să mențineți containere de nivel înalt, cum ar fi aplicații de proces și 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 analiștii și dezvoltatorii BPM pot construi procesele lor și implementările care stau la baza proceselor. Alt task principal pentru administratori este gestionarea accesului la magazia Process Center prin setarea autorizărilor corespunzătoare pentru utilizatori și grupuri.

Utilizatorii cu autorizație corespunzătoare pot realiza anumite taskuri administrative direct în Process Designer și Integration Designer. De exemplu, un dezvoltator cu acces de scriere la aplicația de 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 și mediile runtime

Process Server oferă un singur mediu runtime BPM care poate suporta un interval de procese operaționale precum și orchestrare de servicii și capabilități de integrare.

În mediul dumneavoastră de creație, Process Server integrat în Process Center vă permite să rulați procesele în timp ce le construiți. Când sunteți pregătit, puteți instala și rula aceleași procese pe Process Server în mediul dumneavoastră runtime. Componenta Business Performance Data Warehouse colectează și agregă datele proceselor din procesele care rulează pe serverele Process. Puteți utiliza aceste date pentru a îmbunătăți procesele dumneavoastră operaționale.

Process Admin Console vă permite să gestionați serverele Process în mediile dumneavoastră runtime, de exemplu, intermedierea, testarea, producția precum și Process Server care face parte din Process Center.

Medii de creație

IBM Business Process Manager Advanced oferă două medii de creație. Utilizați IBM Process Designer pentru a modela eficient procese operaționale care implică taskuri umane. Utilizați IBM Integration Designer pentru a construi servicii care sunt auto-conținute sau care invocă servicii existente cum ar fi servicii web, aplicații de resurse de întreținere sau aplicații care rulează în CICS și IMS.

- “Process Designer”
- “Integration Designer” la pagina 7

Process Designer

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

Un procesor este unitatea de logică importantă din IBM Business Process Manager. Este containerul pentru toate componentele unei definiții de proces, inclusiv servicii, activități și gateway-uri; evenimente cronometru, mesaj și excepție; linii de secvență, reguli și variabile. Când modelați un proces, creați un BPD (definiție proces operațional) reutilizabil. Atât Process Designer cât și Integration Designer pot crea modele de procese care pot conține taskuri umane.

Process Designer vă ajută să dezvoltați procese operaționale. Cu o unealtă grafică ușor de utilizat, puteți crea o secvență de acțiuni care compun un proces operațional și în timp puteți reface acel proces pe măsură ce se modifică circumstanțele. Dacă una sau mai multe activități necesită acces la sisteme backend mari sau servicii care furnizează date pentru procesul operațional, de exemplu pentru a obține informații despre clienți, puteți îndeplini acea nevoie utilizând Integration Designer. Utilizând o interfață 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 și îmbunătăți date și adaptoare pentru a ajunge la multe sisteme backend în modul standard. Pe scurt, Process Designer se concentrează pe procesul operațional și Integration Designer se concentrează pe servicii automatizate pentru a completa procesul operațional. Vedeți Inițiere în IBM Process Designer.

Toate proiectele Process Designer sunt conținute în aplicații de proces. Memorați acele aplicații de proces și artefactele asociate în magazia Process Center. Aplicațiile de proces pot partaja aseturi care au fost amplasate în truse de unelte.

IBM Business Process Manager furnizează mai multe interfețe de utilizator pentru a vă permite să modelați, să implementați, să simulați și să inspectați procese operaționale. Creați și gestionați aplicații de proces, truse de unelte, piste și instanțanele din Process Center Console. Puteți crea modele de proces, rapoarte și servicii simple în Process Designer. Puteți rula și depana procese în Inspector. Și puteți rula simulări în Optimizator.

Aplicațiile de proces dezvoltate în Process Designer pot fi rulate în orice moment pe serverul Process Center sau salvate la un instantaneu și implementate pe Process Server. Același lucru este adevărat despre serviciile dezvoltate în Integration Designer și asociate cu aplicații de proces.

Un punct de extensie autentificator de înregistrare este definit în Process Designer pentru a deschide un punct de personalizare pentru logica de logare de pe partea de client pentru a adresa cerințe de autentificare speciale de pe partea de server. Când autentificarea este declanșată, Process Designer primește autentificatorul și invocă logica de autentificare. Punctul de extensie autentificator este furnizat în format plug-in pentru Eclipse. plug-in format. Pentru a utiliza plug-in-ul autentificator, consultați: Instalarea IBM Process Designer.

Integration Designer

Integration Designer furnizează editoare și rutine de ajutor (aids) pentru a ajuta dezvoltatorii să creeze procese și servicii complet automate (cum ar fi module SCA, medieri și procese BPEL). Este disponibil ca parte a pachetului 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 aplicații integrate de construire. Aplicațiile integrate nu sunt simple. Ele pot apela aplicații pe sisteme EIS (Enterprise Information Systems), implica procese operaționale de-a lungul departamentelor sau întreprinderilor și invoca aplicații scrise local sau la distanță într-o varietate de limbaje și rulând pe o varietate de sisteme de operare. Componentele sunt create și asamblate în alte aplicații integrate (adică, aplicații create dintr-un set de componente) prin editori vizuali. Editorii vizuali reprezintă un strat de abstracție între componente și implementările lor. Un dezvoltator care utilizează uneltele poate asambla o aplicație integrată fără cunoștințe detaliate despre implementarea fundamentală a fiecărei componente.

Uneltele Integration Designer sunt bazate pe o arhitectură orientată spre servicii. Componentele sunt servicii și o aplicație integrată care implică mai multe componente este și 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 ușor de utilizat care sunt conforme cu standardul industrial Business Process Execution Language.

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

Modulele și bibliotecile pot fi asociate cu o aplicație de proces pentru utilizare cu Process Center și pot fi utilizate ca servicii de procese create în Process Designer. În astfel de cazuri, ele pot fi de asemenea implementate cu aplicația de proces.

Alternativ, modulele și bibliotecile pot fi implementate direct la mediul de test sau la Process Server. Puteți utiliza module de mediere pentru a crea fluxuri de mediere, pe care le puteți implementa la WebSphere Enterprise Service Bus sau la Process Server.

IBM Integration Designer furnizează de asemenea capacitatea de creare a tipurilor de date și a mapărilor xml care pot fi implementate pe dispozitivul WebSphere DataPower. Puteți de asemenea să transferați fișiere la și de la WebSphere DataPower.

Unelte de administrare

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

Unelte linie de comandă

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

- Uneltele de linie de comandă sunt programe simple pe care le rulați dintr-un prompt linie de comandă al unui sistem de operare pentru a realiza taskuri specifice. Utilizând aceste unelte, puteți porni și opri serverele de aplicații, verifica starea serverului, adăuga sau înlătura noduri și alte taskuri.
- Programul de scriptare administrativă WebSphere (wsadmin) este un mediu de interpretor de comenzi non-grafic care vă permite să rulați opțiuni administrative într-un limbaj de scriptare (Jython sau Jacl) și să lansați în execuție programele scrise în limbajul de scriptare. Acesta suportă aceleași taskuri ca și consola administrativă, precum și multe dintre taskurile consolei Process Center. Unealta wsadmin este intenționată pentru medii de producție și operații nesupravegheate.
- Interfețele de programare administrative sunt un set de clase și metode Java sub specificația JMX (Java Management Extensions) care oferă suport pentru administrarea SCA (Service Component Architecture) și a obiectelor business. Fiecare interfață de programare include o descriere a scopului său, un exemplu care demonstrează modul de utilizare a interfeței sau clasei și referințe la descrierile metodei individuale.

Process Center Console

Process Center Console oferă o locație comodă pentru utilizatori pentru crearea și menținerea elementelor de bibliotecă de nivel înalt cum ar fi aplicațiile de proces și trusele de unelte. Acest lucru ajută la furnizarea unui cadru în care analiștii și dezvoltatorii BPM își pot construi procesele și implementările fundamentale. În plus, Process Center Console oferă unelte pentru menținerea magaziei, inclusiv setarea autorizației corespunzătoare pentru utilizatori și grupuri.

Accesați consola Process Center prin intermediul unui browser web (de exemplu, <http://host:9080/ProcessCenter>).

Consolă Administrare proces

Consola Administrare proces este folosită pentru a administra serverele Process din mediul dumneavoastră, inclusiv utilizatorii și instanțele instalate pentru fiecare server. În plus, aceasta oferă unelte pentru a vă ajuta să gestionați cozile și memorarea în cache.

Consola Administrare proces include Inspectorul de proces, o unealtă pentru vizualizarea și gestionarea instanțelor de proces pentru aplicații de proces care rulează pe un Process Server specific.

Accesați Process Admin Console prin intermediul unui 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 performanță din mediul dumneavoastră. Puteți utiliza această unealtă pentru gestionarea cozilor de server și monitorizarea performanței serverului.

Accesați Business Performance Admin Console prin intermediul unui browser web (de exemplu, <http://host:9080/PerformanceAdmin>).

Consola administrativă WebSphere Application Server

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

Dacă ați instalat un profil autonom, aveți un singur nod în propriul său domeniu administrativ, cunoscut ca o celulă. Utilizați consola administrativă pentru a gestiona aplicații, magistrale, servere și resurse în cadrul acelui domeniu administrativ. În mod similar, dacă ați instalat și configurat o celulă Network Deployment, aveți un nod manager de implementare și unul sau mai multe noduri gestionate în aceeași celulă. Utilizați consola administrativă pentru a gestiona aplicații, seta noduri gestionate în celulă și monitoriza și controla acele noduri și resursele lor.

Accesați această consolă prin intermediul unui browser web (de exemplu, <http://host:9060/ibm/console> sau <https://host:9043/ibm/console>).

Business Process Choreographer Explorer și Business Process Archive Explorer

În funcție de rolul dumneavoastră de utilizator, puteți utiliza aceste interfețe client pentru a gestiona procesele BPEL și taskurile umane create în IBM Integration Designer, puteți lucra cu taskurile alocate dumneavoastră, vizualiza procesele BPEL și taskurile umane complete care sunt într-o bază de date arhivă sau șterge procesele și taskurile din arhivă.

Widget-urile de administrare

Widget-urile de administrare oferă o cale de a gestiona și monitoriza anumite componente ale soluției dumneavoastră de gestiune proces operațional, inclusiv module și servicii Advanced Integration Service. Utilizați aceste widget-uri într-un spațiu operațional pentru a furniza vizibilitate în aplicația dumneavoastră de servicii și module și pentru a răspunde la întrebări precum acestea:

- Ce servicii sunt consumate sau expuse de un modul și care sunt timpul de răspuns și debitul pe o perioadă de timp definită pentru aceste servicii?
- Care este starea unui modul?
- Există evenimente eșuate în modul?
- Ce politici de mediere sunt asociate modulului?

- Ce procese BPEL și taskuri umane sunt utilizate într-un modul?
- Există calendare de afaceri sau reguli operaționale în modul?

Utilizați unul sau mai multe dintre widget-uri pentru a obține un instantaneu al sănătății de sistem generale a soluției dumneavoastră de afaceri, inclusiv starea topologiei dumneavoastră (medii de implementare, cluster-e), a aplicațiilor sistem (de exemplu, managerul de eveniment eşuat sau Business Process Choreographer), a surselor de date, a motoarelor de mesagerie și a coziilor de mesagerie.

Managerul de Reguli de procese operaționale

Managerul de reguli de procese operaționale este o unealtă bazată pe web care ajută analistul de afaceri în răsfoirea și modificarea valorilor regulilor operaționale. Unealta este o opțiune a IBM Process Server pe care o puteți selecta pentru instalare în timpul creării profilului sau după instalarea serverului.

Accesibilitatea în IBM Business Process Manager

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

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

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

Disponibilitatea limbilor naționale în IBM Business Process Manager

IBM Business Process Manager este disponibil într-un număr de limbi. Lista descrie nivelul suportului pentru o limbă dată.

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

- Chineză simplificată
- Chineză tradițională
- Cehă
- Engleză (S.U.A.)
- Franceză
- Germană
- Maghiară
- Italiană
- Japoneză
- Coreeană
- Poloneză
- Portugheză braziliană
- Rusă
- Spaniolă

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

- Arabă (tradusă pentru widget-uri de taskuri umane BPEL, widget-uri Business Process Choreographer Explorer, cadrul de lucru Business Space și widget-uri Business Space Monitor)
- Daneză (tradusă pentru widget-uri de monitorizare Business Space și cadrul de lucru Business Space)
- Olandeză (tradusă pentru cadre de lucru Process Designer, Process Center, BPD Modeler, Service Modeler, JSEditor, Process Designer, Business Space și widget-uri Business Space Monitor)

- Finlandeză (tradusă pentru widget-uri de monitorizare Business Space, cadru de lucru Business Space, BPD Modeler, Service Modeler, JSEditor și Process Designer)
- Greacă (tradusă pentru Process Designer, Process Center și Business Space)
- Ebraică (tradusă pentru taskuri umane BPEL, Business Process Choreographer Explorer, Business Space framework și widget-uri Business Space Monitor)
- Norvegiană (tradusă pentru widget-uri de monitorizare Business Space și cadrul de lucru Business Space)
- Portugheza din Portugalia (Process Designer, Process Center, BPD Modeler, Service Modeler și JSEditor)
- Română (tradusă pentru operațiile runtime)
- Slovacă (tradusă pentru Business Space, BPD Modeler, Service Modeler, JSEditor și Process Designer)
- Suedeză (tradusă pentru widget-uri de monitorizare Business Space și cadru de lucru Business Space)
- Turcă (tradusă pentru Business Space)

Notă: Pentru locale-ul turc, trebuie să setați intrarea **case-insensitive-security-cache** în fișierul 60Database.xml la **false** pentru a permite ca numele de utilizatori și parolele să conțină litera *i*. Fișierul 60Database.xml este localizat în directorul *rădăcină_instalare*\profiles\nume_profil\config\cells\cell_name\nodes\node_name\servers\server_name\process-center\config\system\.

IBM Business Process Manager asigură suport pentru utilizatori pentru a introduce șiruri bidirecționale în mediul Process Designer, în coach-uri și în Process Portal. Acesta oferă API-uri JavaScript pentru manipularea bidirecțională a testului de limbă.

Coach-urile și Process Portal suportă utilizarea calendarelor ebraice și arabe.

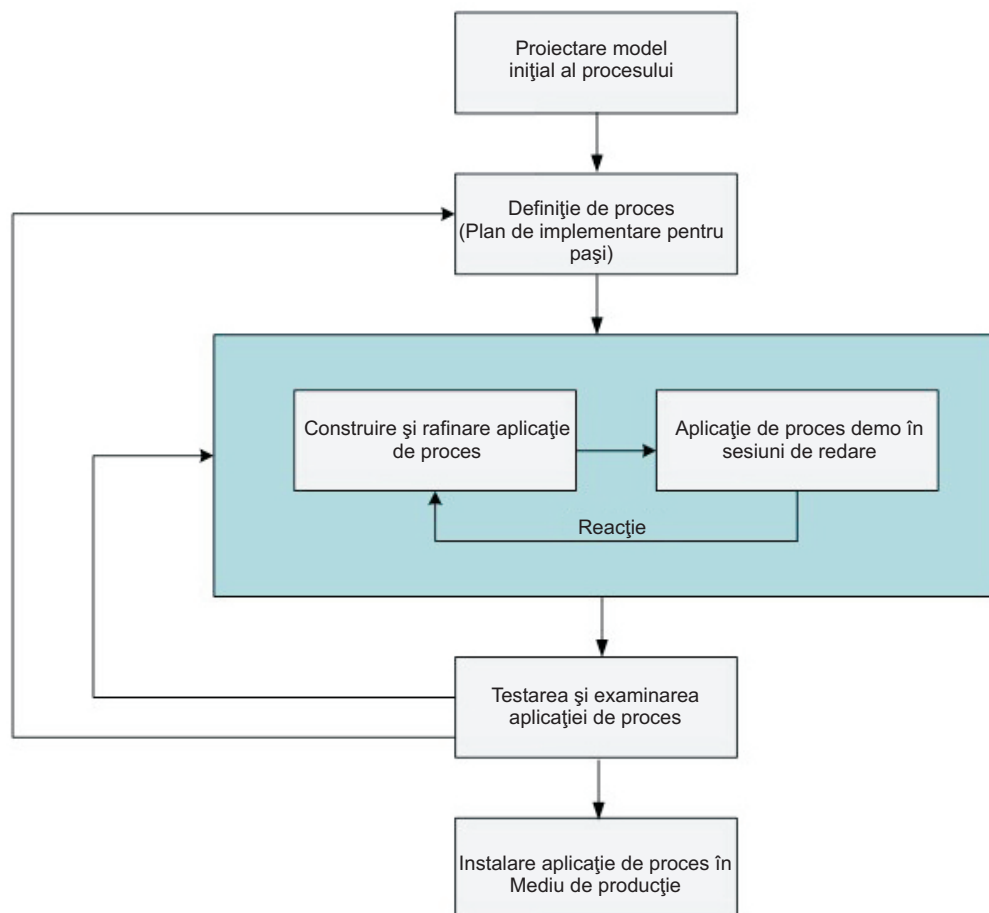
Privire generală asupra BPM (Business Process Management)

La dezvoltarea proceselor operaționale în Process Designer, trebuie să planificați instalarea eventuală a aplicațiilor dumneavoastră de proces pe servere din mediile dumneavoastră de test și de producție.

Process Designer poate fi găsit în IBM Business Process Manager Express, IBM Business Process Manager Standard și IBM Business Process Manager Advanced. În această secțiune ne vom concentra pe versiunea Avansat, care este proiectată pentru automatizare de volum mare și 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 mulți profesioniști 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 inițiați în procesele operaționale sau care nu au nevoie de acces la multe sisteme externe.

Următoarea diagramă afișează ciclul de viață al unui efort de dezvoltare de proces tipic. Include pași pentru construirea și rafinarea unui serviciu de instalare, astfel încât să vă puteți instala aplicațiile de proces în mediul de producție.

După cum arată această diagramă, puteți lucra exclusiv în mediul dumneavoastră de dezvoltare. Dar trebuie să configurați serverele Process Server pentru ambele medii, de test și de producție.



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 definiții proces, inclusiv servicii, activități și gateway-uri; evenimente cronometru, mesaj și excepție; linii de secvență, reguli și variabile. Când modelați un proces, creați un BPD (Business Process Definition) reutilizabil.

Componentele proces vă permit să definiți fluxul de lucru al procesului pentru utilizatori finali, crearea de logică în interiorul unui proces și integrarea cu alte aplicații și surse de date. Pentru a înțelege ce se întâmplă în interiorul unui proces în timpul rulării, este important să înțelegeți componentele care compun un proces în timpul proiectării.

Construirea proceselor în IBM BPM

Mulți indivizi diferiți din diverse organizații sunt implicați de obicei în dezvoltarea proceselor utilizând IBM BPM. Preocuparea de înlocuire este să vă asigurați că construiți cea mai bună soluție posibilă pentru îndeplinirea scopurilor declarate ale proiectului dumneavoastră. Pentru a asigura rezultate satisfăcătoare, membrii echipei ar trebui să lucreze împreună pentru a îndeplini cerințele de proces și a dezvolta interactiv modelul și implementările acestuia.

Re-utilizarea elementelor în Process Designer

Process Designer permite dezvoltatorilor de procese să re-utilizeze elementele existente și în aplicațiile de proces și peste acestea. De exemplu, dacă cunoașteți câteva servicii ce există deja și care includ Antrenori și alte elemente partajate de care dumneavoastră și alți dezvoltatori aveți nevoie, puteți accesa și re-utiliza acele elemente incluzându-le pe acestea într-o trusă de unelte. Apoi, din aplicația dumneavoastră de proces, puteți adăuga o dependență la trusa de

unelte în care se află elementele partajate. Acest lucru vă permite să folosiți unul din serviciile existente atunci când alegeți implementarea pentru o activitate. Elementele din trusa de unelte pot de asemenea fi utilizate de către alți dezvoltatori în diferite aplicații de proces.

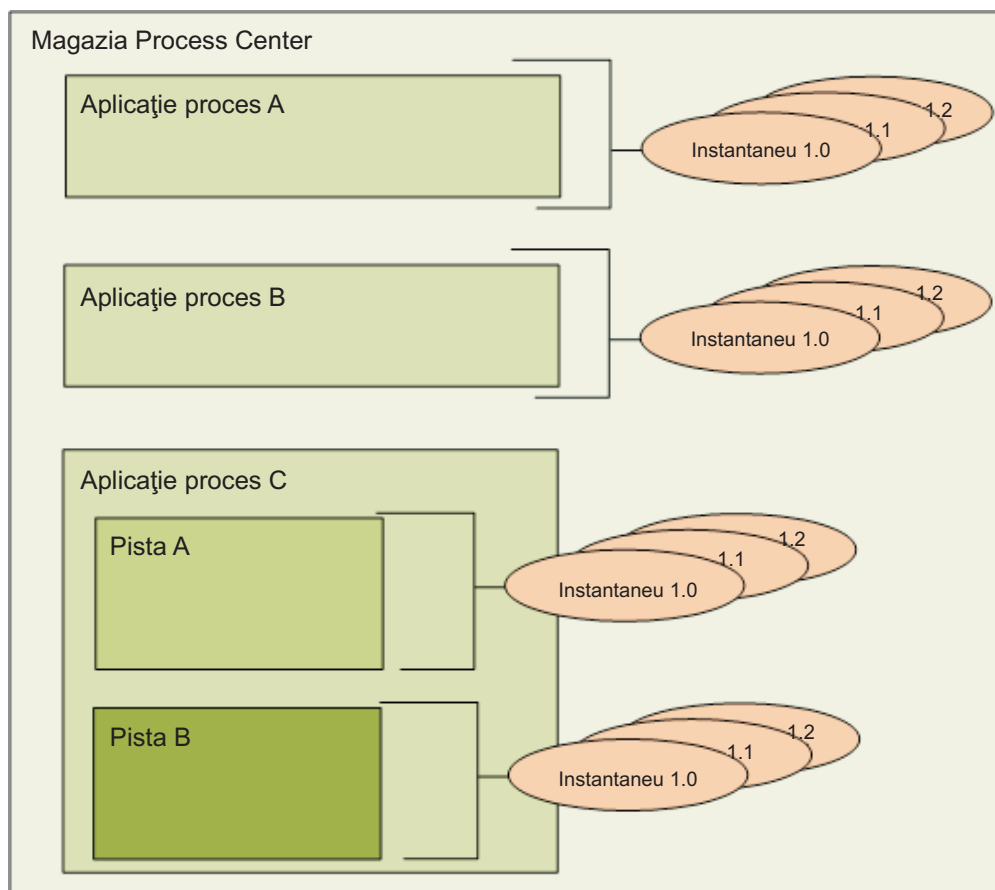
Utilizarea Designer în IBM Process Designer

Interfața Designer furnizează uneltele de care aveți nevoie pentru a modela procesele dumneavoastră în IBM BPM.

Dezvoltarea procesului cu Process Center

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

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



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

Tip de conținut	Descriere
Aplicații de Proces	Containere pentru modelele de proces și implementările de suport pe care analiștii și dezvoltatorii BPM le creează în Designer în IBM Process Designer.
Piste	Subdivizii opționale într-o aplicație de proces bazată pe taskuri de echipă sau versiuni ale aplicației de proces. Atunci când sunt activate, piste permit realizarea dezvoltării paralele. Administratorii determină dacă piste suplimentare sunt necesare și, prin urmare, activate pentru fiecare aplicație de proces.

Tip de conținut	Descriere
Instantanee	Înregistrați starea elementelor dintr-o aplicație de proces sau pistă la un anumit moment de timp. De obicei, instantaneele reprezintă o bornă sau sunt utilizate pentru derulările înapoi sau instalări. Puteți compara instantaneele și reveni la cele anterioare. Dacă un administrator activează piste pentru o aplicație de proces, un instantaneu este utilizat ca bază pentru noua pistă.

Aplicații de proces: Privire generală

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

Aplicațiile de proces conțin unele dintre sau toate artefactele următoare:

- Unul sau mai multe modele de proces, numite de asemenea BPD-uri (definiții de procese operaționale)
- Referințe la truse de unelte
- Serviciile necesare pentru a implementa activități 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)
- Oricare alte articole necesare pentru a rula procesul

Pentru un video introductiv despre aplicația de proces iterativă și dezvoltarea de unelte cu tip-uri, instantanee și piste, urmăriți “Iterative Process Application and Toolkit Development”, disponibil pe YouTube sau pe Centrul de informare IBM Education Assistant. Este disponibilă o transcriere a materialului video.

Sugestie pentru aplicația de proces, instantanee și piste

Toate modificările pe care le faceți la o aplicație de proces sunt salvate dinamic la magazia Process Center la sugestie, care este versiunea curentă a aplicației de proces care funcționează. Puteți utiliza sesiuni de redare pe sugestie pentru a testa instant și gestiona versiunea curentă care funcționează a aplicației de proces.

Aplicația de proces rămâne la nivel de sugestie până când decideți să creați un instantaneu, care înregistrează starea articolelor de bibliotecă dintr-o aplicație de proces sau pistă la un anumit moment. Tipic, realizați un instantaneu de fiecare dată când sunteți pregătit să testați integrarea sau vreți să instalați aplicația de proces pe un server Process Center sau un Process Server pentru testare, intermediere sau producție.

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

Implicit, fiecare aplicație de proces are o singură pistă, denumită Principală. Dacă vreți să permiteți dezvoltarea paralelă pe o aplicație de proces, puteți crea piste suplimentare. Aceste subdivizii opționale din aplicația de proces țin modificările izolate. De exemplu, imaginați-vă că compania dumneavoastră este în proces de schimbare de imagine; în timpul acestei tranziții, aplicațiile proces curente trebuie să fie menținute în timp ce sunt dezvoltate versiuni noi bazate pe identitatea actualizată a corporației. În această situație, o echipă poate face modificări minore pe versiunea curentă a aplicației de proces (în pista Principală) în timp ce altă echipă construiește o versiune nouă a aplicației de proces într-o pistă separată.

Truse de unelte pentru aplicații proces

Trusele de unelte sunt containere care stochează articole de bibliotecă (de exemplu, BPD-uri) pentru reutilizare de către aplicațiile proces sau de alte truse de unelte. Aplicațiile de proces pot împărți articole de bibliotecă din una sau mai multe truse de unelte, și trusele de unelte pot împărți articole din alte truse de unelte. Dacă aveți acces la o trusă de unelte, puteți crea o dependență pe ea și să folosiți articolele bibliotecă ale acelei truse de unelte în aplicația dumneavoastră de proces.

Aplicații de proces și aplicații la nivel operațional

O aplicație de procesare are o aplicație la nivel operațional (BLA), care se comportă ca un container pentru aplicația de procesare și aseturile sale (aseturile includ lucruri ca modelele SCA, seturi de unelte și biblioteci). Fiecare instanțaneu de aplicație de proces are propriul său BLA. Multe dintre taskurile de administrare al unui instanțaneu (de exemplu, oprirea sau pornirea acestuia într-un server de producție) sunt efectuate la nivelul BLA, permițând administrarea mult mai rapidă și mai simplă a instanțaneului și a tuturor aseturilor sale.

Dezvoltarea aplicației iterative de proces și a trusei de unelte

Tabela 3. Introducere

Scena	Audio	Acțiune pe-ecran
1	Bine ați venit la această introducere la dezvoltarea aplicației iterative de proces și a trusei de unelte în IBM Business Process Manager.	Ecranul de întâmpinare afișează titlul materialului video, <i>Dezvoltarea aplicației iterative de proces și a trusei de unelte</i> și subtitlul, <i>Învățați cum se utilizează instanțaneele de top și pistele</i> . Copyright 2013, IBM Corporation.
2	În acest material video, veți studia aplicațiile de proces și trusele de unelte și cum puteți utiliza tip-urile, instanțaneele și pistele pentru a le gestiona dezvoltarea și implementarea.	Este afișată o listă cu marcatori cu subiectele acoperite în materialul video.
3	Procesele se desfășoară în timp funcție de cum se modifică nevoile organizației dumneavoastră. Pe măsură ce aplicațiile de proces și trusele de unelte trec prin ciclul de dezvoltare, testare și producție, puteți gestiona aseturi de proces și versiuni cu IBM Business Process Manager.	Este afișată o diagramă a ciclului de proiectare iterativă. Săgețile se mișcă în diagramă pentru a arăta natura ciclică a procesului de proiectare.
4	În IBM Business Process Manager, aplicațiile de proces sunt containere pentru modelele de proces și implementări de suport pe care le creează dezvoltorii în IBM Process Designer.	Este afișată fila Aplicații de proces în consola Process Center. Cursorul trece peste numele unei aplicații de proces și apoi trece peste o legătură la locul unde utilizatorul o poate deschide în IBM Process Designer. Este afișat textul de ecran “Aplicație de proces: container instalabil pentru modelul de proces”.
5	Trusele de unelte sunt containere care păstrează aceleași artefacte ca aplicațiile de proces. Totuși, spre deosebire de aplicațiile de proces, acestea nu pot fi instalate și rulează pe Process Server. Trusele de unelte rețin artefacte care sunt reutilizabile de una sau mai multe aplicații de proces. Trusele de unelte sunt apoi instalate indirect cu fiecare aplicație de proces care se referă la ele.	Este afișată fila Truse de unelte în consola Process Center. Cursorul trece peste numele unei truse de unelte și apoi trece peste o legătură la locul unde utilizatorul o poate deschide în IBM Process Designer. Este afișat textul pe ecran “Trusă de unelte: container pentru artefacte reutilizabile”.

Tabela 4. Tip-uri, instanțanee și piste

Scena	Audio	Acțiune pe-ecran
6	IBM Business Process Manager suportă o abordare iterativă la dezvoltarea aplicației de proces și a trusei de unelte cu caracteristici cum ar fi: <ul style="list-style-type: none"> • Tip-ul, care este versiunea curentă de lucru • Instantanee, care înregistrează starea articolelor din bibliotecă la un anumit moment în timp • Piste, care sunt ramuri opționale într-o aplicație de proces sau o trusă de unelte pe care le utilizați pentru a dezvolta concurrent versiuni multiple 	Este afișată o listă cu marcatori de caracteristici incluse în IBM BPM care suportă o abordare iterativă pentru dezvoltarea aplicației de proces și a trusei de unelte.

Tabela 4. Tip-uri, instanțanee și piste (continuare)

Scena	Audio	Acțiune pe-ecran
7	Când dezvoltăți o aplicație de proces sau o trusă de unelte în IBM Process Designer, toate modificările pe care le faceți sunt salvate în magazia Process Center. Această versiune curentă de lucru a aplicației dumneavoastră de proces sau a trusei de unelte este numită tip.	Este afișată o diagramă de proces în IBM Process Designer. Este deschisă o componentă de diagramă și cuvântul <i>date</i> este modificat la cuvântul <i>dată</i> într-un Coach. Modificările sunt apoi salvate. Este afișat textul pe-ecran “Tip: versiunea curentă de lucru”.
8	Puteți utiliza sesiuni de redare pe pentru a testa instantaneu versiunea curentă de lucru a aplicației de proces sau a trusei de unelte. Puteți rula tip-ul numai pe serverul Process Center și nu puteți instala un tip pe Process Server.	Este deschisă pagina de sus a diagramei de proces și este pornită sesiunea de redare. Aceasta deschide interfața Inspector și afișează Coach-ul modificat într-o fereastră de browser. Este afișat textul pe ecran “Tip: redare pentru testare instantanee”.
9	Când ați terminat modificarea aplicației dumneavoastră de proces sau a trusei de unelte, puteți crea un instantaneu al acesteia. Instantaneul înregistrează starea tuturor elementelor dintr-o aplicație de proces sau truselor de la un anumit moment de timp. După ce creați un instantaneu, procesele dumneavoastră și elementele înrudite sunt încă disponibile să fie editate pe tip. În Process Designer, instanțanele disponibile sunt afișate sub Istoric revizii. Deși puteți vizualiza aceste instanțanee, acestea nu sunt editabile.	Fereastra de browser este închisă și este creat un instantaneu al aplicației de proces. Este afișată interfața Designer și cursorul trece peste numele noului instantaneu sub secțiunea Istoric revizii. Este afișat textul pe-ecran “Instantaneu: înregistrări ale stării tuturor elementelor”.
<	Puteți de asemenea compara instanțanee pentru a vedea informații cum ar fi momentul când au fost create și toate procesele care au fost adăugate la ele.	Este afișată vizualizarea comparației. Cursorul trece peste data când a fost creat un instantaneu și peste numele unui articol din proces care a fost modificat. Este afișat textul pe ecran “Compararea modificărilor între instanțanee”.
11	Când creați un instantaneu nou pentru o trusă de unelte, aplicațiile de proces care se referă la acea trusă de unelte vor continua să utilizeze vechiul instantaneu. Când deschideți aplicația de proces în IBM Process Designer pentru a o edita, un avertisment vă notifică asupra unei noi versiuni disponibile pentru trusa de unelte.	Este deschisă consola Process Center. Este afișată pagina Truse de unelte și este deschisă trusa de unelte. Este creat un nou instantaneu al trusei de unelte. Este deschisă pagina Aplicații de proces și este deschisă o aplicație de proces în Process Designer. Cursorul trece peste avertismentul care apare și a articolelor din meniu înrudite. Este afișat textul pe ecran “Aplicațiile de proces mențin referințele la vechiul instantaneu de trusă de unelte”.
12	Când aplicația dumneavoastră de proces sau trusa de unelte este gata să fie implementată, un administrator poate instala unul sau mai multe instanțanee din ele pe Process Server utilizând consola Process Center. Aplicațiile de proces instalate se pot referi numai la instanțanee ale trusei de unelte și nu la versiunea curentă de lucru a unei truse de unelte.	Este deschisă consola Process Center și apoi este deschisă o aplicație de proces. Cursorul trece peste o legătură pe un instantaneu pe care un administrator poate face clic pentru a instala instantaneul. Este afișat textul pe-ecran “Instalare instanțanee aplicație de proces pe Process Server”.
13	Administratorii pot gestiona și instanțanee din consola Process Center, cum ar fi prin crearea, arhivarea și exportul instanțaneelor.	Meniul derulant de lângă numele instantaneului este deschis și cursorul trece peste fiecare opțiune din meniu. Este afișat textul pe ecran “Gestionare instanțanee utilizând consola Process Center”.
14	Implicit, fiecare aplicație de proces sau trusă de unelte are o singură pistă, denumită Principală. Administratorii pot activa crearea pistelor suplimentare pentru a permite dezvoltare în paralel. Aceste subdivizii opționale țin modificările izolate.	Este deschisă pagina Gestionare a aplicației de proces. Este verificată opțiunea “Permiterea pentru vizitatori să creeze piste în această aplicație de proces”. Este afișat textul pe ecran “Piste: subdiviziuni opționale care păstrează modificările izolate”.

Tabela 4. Tip-uri, instantanee și piste (continuare)

Scena	Audio	Acțiune pe-ecran
15	Creai o nouă pistă dintr-un instantaneu. Aceasta copiază instantaneul selectat în noua pistă. Alte instantanee din sursa sursă nu sunt copiate pe noua pistă.	Este deschisă pagina Instantanee a aplicației de proces. Este creată o nouă pistă numită Pista 2. Noua pistă este deschisă și cursorul trece peste numele instantaneului peste care a fost copiată. Este afișat textul pe-ecran “Crearea unei piste dintr-un instantaneu”.
16	Fiecare pistă are un tip sau o versiune curentă de lucru distinctivă. Într-o aplicație de proces sau o trusă de unelte cu piste multiple, puteți copia aseturi dintr-un instantaneu pe o pistă la tip sau la altă pistă. Când faceți acest lucru, aseturile care au fost revizuite suprascriu aseturile care sunt deja pe pista de destinație și noile aseturi sunt adăugate.	Este deschis noul instantaneu și este afișată o pagină cu articolele și setările de aplicație de proces. Este afișată definiția procesului operațional. Aseturile sunt copiate la tip-ul pistei Principale. Apoi sunt afișate paginile unde vor apărea listele de seturi noi, actualizate și în conflict. Este afișat textul pe ecran “Copiați aseturi la alte piste”.
17	Ierarhia instantaneelor aplicațiilor de proces și a pistelor din magazia Process Center este afișată aici. În această diagramă, aplicațiile de proces A și B au o singură pistă implicită, instantanee multiple și un singur tip. Aplicația de proces C are două piste, fiecare cu instantanee multiple și un tip. Instantaneele și tip-ul pe fiecare pistă sunt independente între ele.	Este afișată o diagramă care afișează relațiile dintre tip-uri, instantanee și piste. Diagrama afișează reprezentări ale următoarelor aplicații de proces: <ul style="list-style-type: none"> • Aplicația de proces A, care are o pistă, patru instantanee și un tip • Aplicația de proces B, care are o pistă, trei instantanee și un tip • Aplicația de proces C, care are două piste. O pistă are trei instantanee și un tip și cealaltă pistă are două instantanee și un tip.

Tabela 5. Scenariu de utilizare tip-uri, instantanee și piste

Scena	Audio	Acțiune pe-ecran
18	Acum că știți care sunt aplicațiile de proces și trusele de unelte și cum se utilizează tip-uri, instantanee și piste, să vedem când ați putea să le utilizați. Veți merge pe urma lui Robert, un programator de afaceri pentru Compania ABC.	Este afișată o fotografie a unui bărbat zâmbitor. Fotografia are eticheta “Robert, programator de afaceri, Compania ABC”.
19	Robert dezvoltă un proces care modelează cum angajează compania sa noi angajați. Procesul lui are pista implicită, Principală. Robert face modificări la proces pe tip. Pentru a vă asigura că procesul rulează lîn, el pornește o sesiune de redare, care rulează tip-ul pe serverul Process Center.	Este afișată o diagramă care reprezintă un mediu de dezvoltare de aplicații de proces. <p>Sub secțiunea numită Dezvoltare, este afișat un set de trei dreptunghiuri imbricate. Dreptunghiul exterior reprezintă magazia Process Center, dreptunghiul din mijloc reprezintă aplicația de proces și dreptunghiul interior reprezintă pista Principală.</p> <p>În dreapta secțiunii Dezvoltare este o secțiune numită Testare. Această secțiune conține un singur dreptunghi care reprezintă serverul Process Center.</p> <p>Sub secțiunea Testare există o secțiune numită Producție. Această secțiune conține un singur dreptunghi care reprezintă Process Server.</p> <p>Pentru a-l arăta pe Robert făcând modificări la tip, este afișat un triunghi care reprezintă tip-ul pe pista Principală. Când Robert rulează o sesiune de redare, o săgeată indică tip-ul ca fiind copiat pe serverul Process Center.</p>

Tabela 5. Scenariu de utilizare tip-uri, instantanee și piste (continuare)

Scena	Audio	Acțiune pe-ecran
20	Robert și echipa sa au terminat dezvoltarea și testarea aplicației de proces și aceasta este gata să fie implementată pe mediul de producție. Pentru a face acest lucru, Robert creează un instantaneu al aplicației de proces. Apoi, Alice, administratorul, instalează instantaneul pe Process Server. Compania ABC poate utiliza acum procesul lui Robert pentru a superviza angajarea de noi salariați.	Pentru a-l arăta pe Robert creând un instantaneu, apare un oval care reprezintă un instantaneu lângă tip-ul de pe pista Principal. Pentru a o indica pe Alice instalând instantaneul, o săgeată indică instantaneul care este copiat pe Process Server.
21	Mai târziu, Compania ABC anunță că va face modificare de imagine și că va fi cunoscută în viitor drept Compania DEF. În timpul acestei tranziții, procesul de angajare pe care l-a creat Robert trebuie să fie menținut în timp ce este dezvoltat o nouă versiune bazată pe identitatea actualizată a corporației. Echipa de întreținere face corecții minore asupra procesului original pe pista Principală în timp ce Robert și echipa sa creează o versiune actualizată la noua imagine a aplicației de angajare în pista Schimbare de imagine la DEF.	Apare un instantaneu suplimentar pentru a arăta că dezvoltarea mai departe a fost făcută pe aplicația de proces. Este adăugat un dreptunghi care reprezintă pista Schimbare de imagine la DEF în interiorul aplicației de proces. Pista Schimbare de imagine la DEF are inițial un instantaneu și un tip și apoi este adăugat alt instantaneu pentru a arăta dezvoltarea mai departe pe pistă.
22	Atât echipa de întreținere cât și echipa de schimbare de imagine pot rula sesiuni separate de redare pe serverul Process Center. Când aplicațiile de proces sunt gata să fie implementate, fiecare pistă poate instala unul sau mai multe instantanee pe Process Server.	Săgețile indică direcția tip-urilor de la ambele piste care sunt rulate pe serverul Process Center în același timp. Apoi, săgețile indică instantaneele de la fiecare pistă care sunt rulate pe serverul Process Center în același timp.

Tabela 6. Concluzie

Scena	Audio	Acțiune pe-ecran
23	În acest material video, am inclus aplicațiile de proces și trusele de unelte și cum suportă IBM Business Process Manager dezvoltarea iterativă a proceselor cu tip-uri, instantanee și piste.	Este afișată o listă cu marcatori cu subiectele acoperite de materialul video.
24	Pentru informații suplimentare despre IBM Business Process Manager și utilizarea tip-urilor, instantaneelor și pistelor, răsfoiți alte materiale video ale noastre pe YouTube sau vizitați alte resurse oficiale.	Este afișată o listă cu marcatori a următoarelor resurse suplimentare: <ul style="list-style-type: none"> • Canalul WebSphereEducation YouTube • Documentația IBM Business Process Manager V8.5 • IBM developerWorks • IBM Education Assistant

Rularea și depanarea proceselor

Utilizând Inspectorul, dezvoltatorii individuali pot rula procese și servicii pe Process Center Runtime-ul server sau la distanță 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 și implementarea procesului curent în sesiunile de redare. Sesiunile de redare ajută capturarea informațiilor importante de la diferiți deținători de acțiuni (stakeholder) într-un proces, precum management, utilizatori finali și analiști business. Aplicarea unei abordări iterative dezvoltării procesului asigură îndeplinirea scopurilor și nevoile tuturor celor implicați de către aplicațiile de proces.

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

Task	Descriere
Gestionare instanțe ale proceselor	Când rulați un proces, puteți vizualiza toate instanțele rulate anterior sau rulate curent pe serverele IBM Business Process Manager din mediul dumneavoastră. Puteți gestiona instanțele care rulează prin oprirea și repornirea lor, de exemplu. Puteți gestiona de asemenea instanțele rulate anterior prin filtrarea sau ștergerea înregistrărilor specifice.
Executarea pas cu pas și depanarea unui proces	Pentru o instanță selectată, vedeți pasul care se execută momentan și apoi mergeți înainte prin proces, evaluând executarea procesului pas cu pas. O afișare arbore a procesului combinată cu indicatorii numiți jetoane într-o diagramă proces face mai ușoară înțelegerea locului în care vă aflați în proces. Aveți de asemenea avantajul cunoașterii variabilelor utilizate în fiecare pas și valorile lor corespunzătoare (unde este aplicabil).

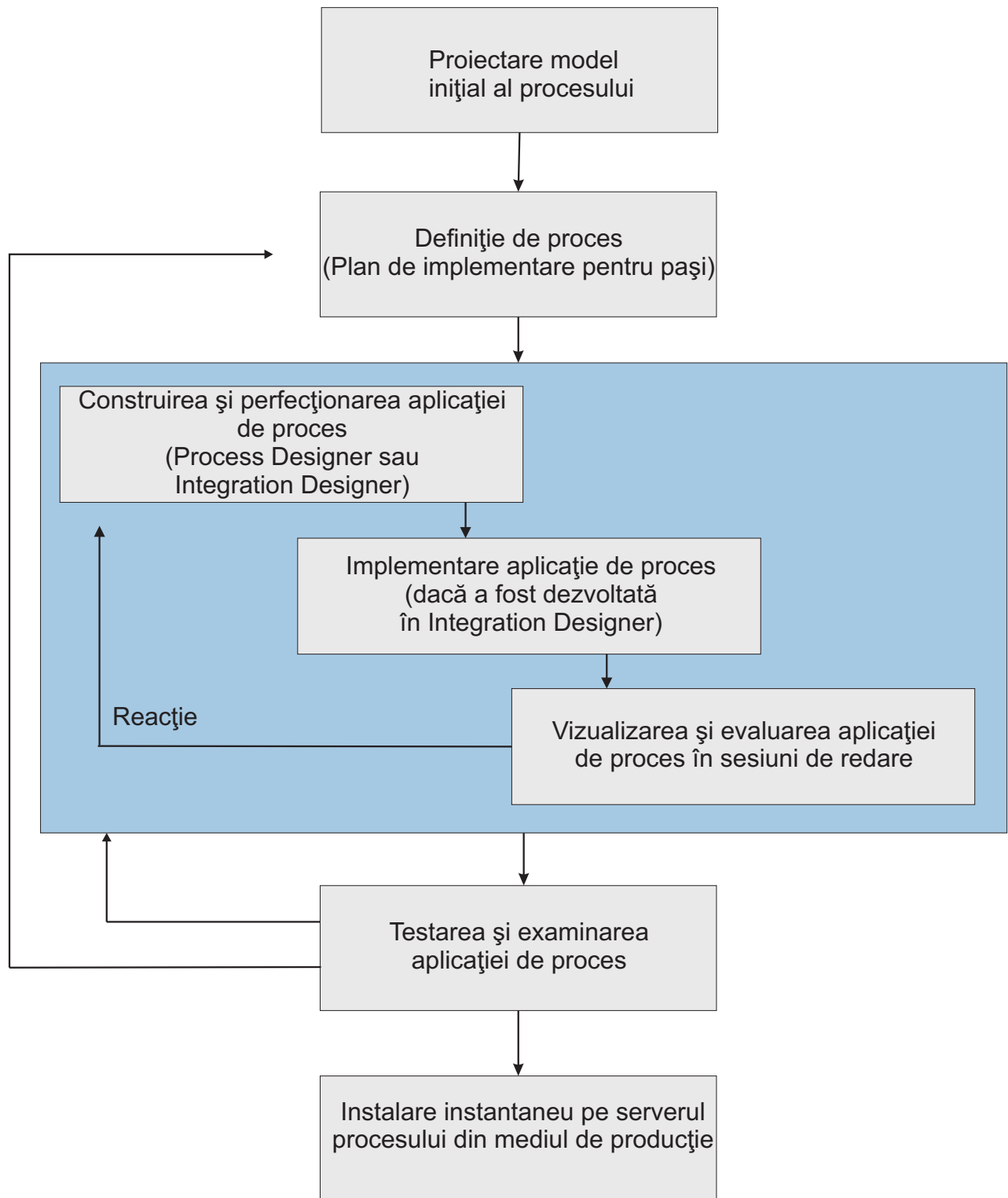
Dacă lucrați în IBM Integration Designer, puteți utiliza Inspectorul dacă proiectul dumneavoastră este asociat cu o aplicație de proces. Puteți avea de asemenea alte unelte de depanare sau test disponibile pentru dumneavoastră. Pentru informații suplimentare despre aceste unelte Integration Designer, vedeți "Module de testare" și "Utilizarea depanatorului de integrare pentru determinarea problemei" în legăturile înrudite.

Instalarea și gestionarea aplicațiilor de proces

Ciclul de viață al aplicației de proces include instalarea, administrarea și dezimplementarea instanțelor. Considerațiile de versionare sunt, de asemenea, parte componentă a ciclului de viață.

Atunci când dezvoltați procese, puteți profita de abordarea iterativă suportată de uneltele din cadrul Process Designer. Procesele evoluează în timp, inițial de la o stare de dezvoltare și apoi la testare și apoi la producție. Chiar și în producție, procesele dvs. își pot continua desfășurarea din cauza necesității modificării. Este important să fiți pregătit pentru ciclul de viață în curs de desfășurare al proceselor dumneavoastră și vă va ajuta să proiectați efectiv de la început.

Figura următoare ilustrează o abordare iterativă a dezvoltării procesului.



O configurație tipică a Managerului procesului operațional include trei medii pentru a suporta dezvoltarea și instalarea eventuală a proceselor dumneavoastră.

Mediu	Descriere
Dezvoltare	Construiți și rafinați aplicațiile de proces în IBM Process Designer. Creați modele de proces și implementați pașii în acele modele utilizând Designer. Utilizând Inspector, demonstrați progresul dezvoltării dumneavoastră în sesiuni de simulare, astfel încât să puteți evalua rapid și să vă puteți rafina prototipul. Utilizând consola Process Center, instalați aplicațiile de proces pe servere de testare sau de producție ale procesului.
Test	Utilizând consola Process Center, instalați aplicațiile de proces în Process Server din mediul dumneavoastră de testare pentru a implementa teste formale de siguranță a calității. Puteți utiliza Inspector pentru a ajuta la verificarea și rezolvarea problemelor.
Producție	Când toate problemele raportate la testarea formală sunt rezolvate, utilizați consola Process Center pentru a instala aplicațiile de proces în Process Server din mediul dumneavoastră de producție. Puteți utiliza Inspector pentru a investiga și rezolva orice probleme raportate în mediul dumneavoastră de producție.

Dacă doriți să testați, instalați sau să administrați un instantaneu pentru aplicația de procesare care are conținut IBM Business Process Manager Advanced , utilizatorul sau grupul din care faceți parte trebuie să fie atribuit Configuratorului, Operatorului și Implementatorului rol securitate administrativă. Dacă în prezent nu sunteți alocat tuturor acestor roluri, faceți clic pe **Utilizatori și grupuri** în consola administrativă WebSphere pentru a modifica rolurile utilizator sau grup. Vedeți Rolurile de securitate IBM Business Process Manager.

Strategii de eliberare și instalare

Pentru a vă asigura că aplicațiile de proces pe care le implementați și instalați îndeplinesc standardele de calitate ale organizației dumneavoastră, considerați definirea unei strategii de eliberare și instalare. Când ați identificat obiectivele și cerințele necesare ediției și instalării noii și actualizatei aplicații de proces, puteți automatiza procesele necesare pentru a aproba și a lansa programele.

De exemplu, poate doriți să direcționați un proces către mai mulți manageri diferiți care folosesc structuri diferite de raportare în cadrul organizației dumneavoastră. Doar după ce fiecare manager renunță la procesul nou sau actualizat, acesta poate fi instalat în mediul dumneavoastră de producție și prezentat utilizatorilor finali. Puteți crea și implementa pașii implicați într-o asemenea examinare a IBM Business Process Manager Advanced pentru a vă asigura că toate indicațiile corporației au fost îndeplinite și că aveți semnăturile necesare. Ultimul pas al examinării ar putea fi notificarea echipei de IT asupra faptului că aplicația de proces aprobată este gata pentru instalare.

Crearea, accesarea și încorporarea serviciilor

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

Accesarea serviciilor externe unei aplicații

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

Notă: Acest scenariu este aplicabil pentru IBM Business Process Manager Advanced.

Într-o aplicație operațională integrată, *serviciile operaționale* interacționează între ele pentru a furniza o funcție necesară. Un serviciu operațional realizează o funcție sau task repetabile ce contribuie la realizarea unui scop operațional. Dar munca de a localiza un serviciu și conectarea la el nu este înrudită cu funcția operațională. Separarea funcției operaționale de taskul de a gestiona conexiuni la servicii oferă flexibilitate unei soluții.

Interacțiunea cu serviciul începe când un *solicitant de servicii* trimite o cerere unui *furnizor de servicii* să realizeze o funcție operațională. Această cerere este trimisă sub forma unui *mesaj*, ce definește funcția ce urmează să fie realizată. Furnizorul de servicii efectuează funcția cerută și trimite rezultatul într-un mesaj solicitantului de servicii. În mod normal, mesajele trebuie să fie procesate pentru a permite serviciilor să schimbe date și să implementeze alte funcții IT

de nivel scăzut ce sunt independente de funcțiile și datele operaționale. De exemplu, rutarea, conversia protoalelor, transformarea, reîncercarea unei invocări eșuate și 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 conțină logică operațională (precum procese operaționale, reguli operaționale și mașini cu stări operaționale) și module de mediere, ce implementează fluxuri de mediere. Deși există o suprapunere de funcții între cele două tipuri de module, în general, recomandăm ca logica operațională să fie izolată în module operaționale și logica de mediere să fie realizată de module de mediere.

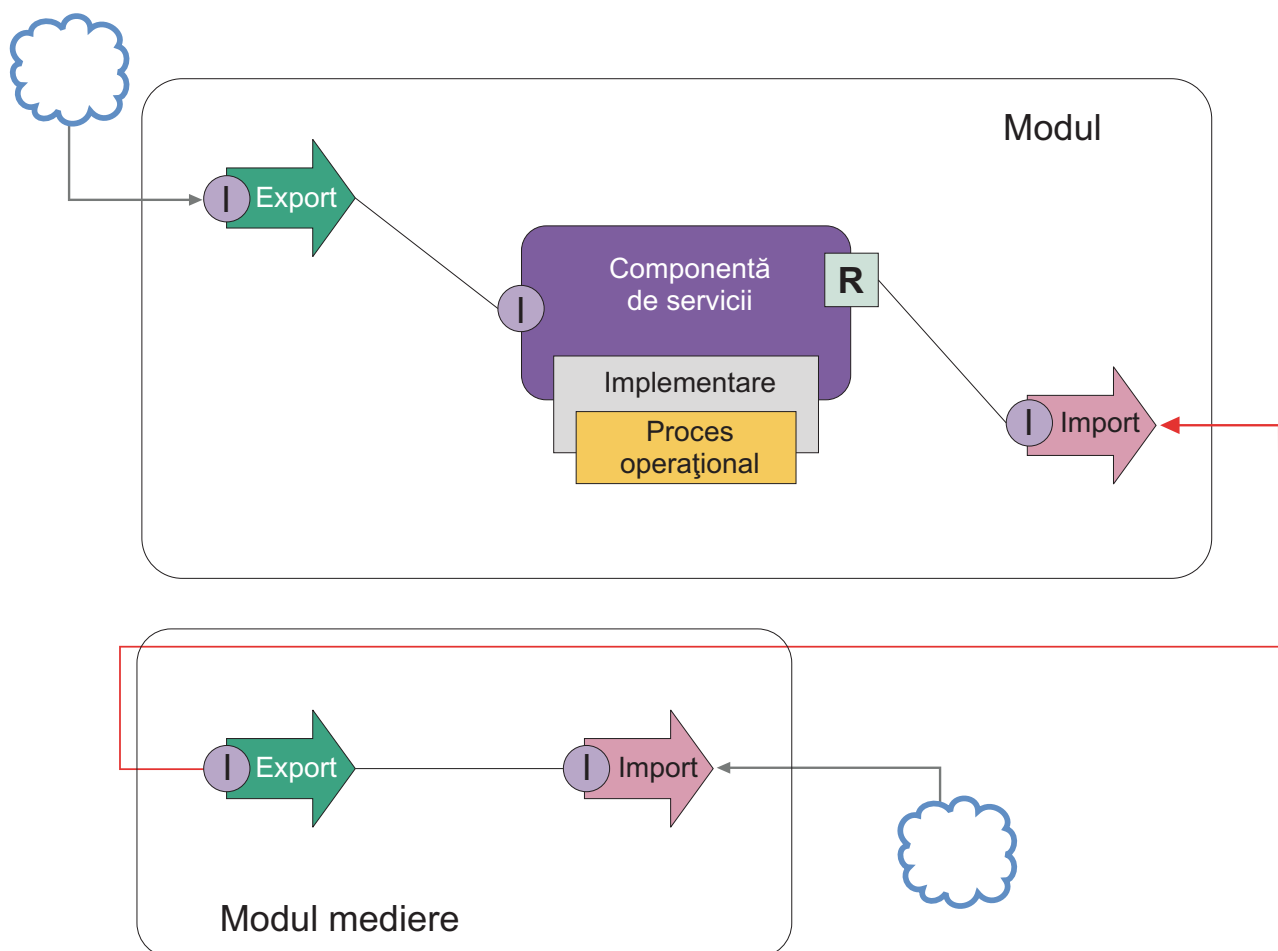
Însă nu există întotdeauna o separare clară între logica operațională și cea de mediere. În aceste cazuri, luați în considerare cantitatea de *stare* sau date din variabile ce vor trebui să fie procesate între invocări de servicii. În general, dacă este nevoie de puțină procesare de stare sau deloc, luați în considerare utilizarea unei componente de flux de mediere. Dacă trebuie să memorați starea între invocări de servicii sau dacă aveți date ce vor trebui să fie stocate în variabile și procesate, considerați utilizarea unei componente de proces operațional. De exemplu, dacă apeleți mai multe servicii și înregistrați informațiile returnate din fiecare, astfel încât după invocarea tuturor serviciilor, doriți să faceți procesări mai departe cu datele returnate, utilizați un proces operațional unde puteți alocă ușor informațiile returnate variabilelor. Cu alte cuvinte, când aveți prea multe stări ați traversat linia spre logica operațională. Secțiunile următoare ajută la clarificarea acestei indicații.

Nu există un singur scenariu de integrare și nu există un răspuns greșit din punct de vedere tehnic. Indicațiile discutate aici sunt un exercițiu bun pentru a permite flexibilitatea și reutilizarea și sunt prezentate pentru considerațiile dumneavoastră. Ca de obicei, trebuie să considerați cu grijă beneficiile și dezavantajele implementării acestor tipare pentru aplicație dumneavoastră de Business Integration. Haideți să considerăm câteva situații.

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 și în această situație, puteți accesa serviciul extern dintr-un modul de mediere, decât să-l accesați direct dintr-un modul operațional. Aceasta ar permite flexibilitate în viitor, pentru modificarea punctului final sau calitatea serviciului sau guvernării (de exemplu, adăugarea jurnalizării) fără a afecta componentele operaționale ce consumă serviciul. Acest tipar arhitectural este cunoscut ca "separare de interese".

Înainte să vă decideți să implementați acest tipar, cântăriți beneficiile tiparului contra potențialelor efecte ale regiei introduse de alt modul. Dacă principala dumneavoastră cerință este flexibilitatea și veți face modificări frecvente asupra serviciilor accesate, luați în considerare utilizarea unui modul separat după cum este arătat aici. Dacă performanța este cea mai importantă și sunteți dispuși să actualizați și să reimplementați logica operațională, luați în considerare utilizarea unui singur modul.



Iată taskurile de nivel înalt pentru realizarea acestui exemplu.

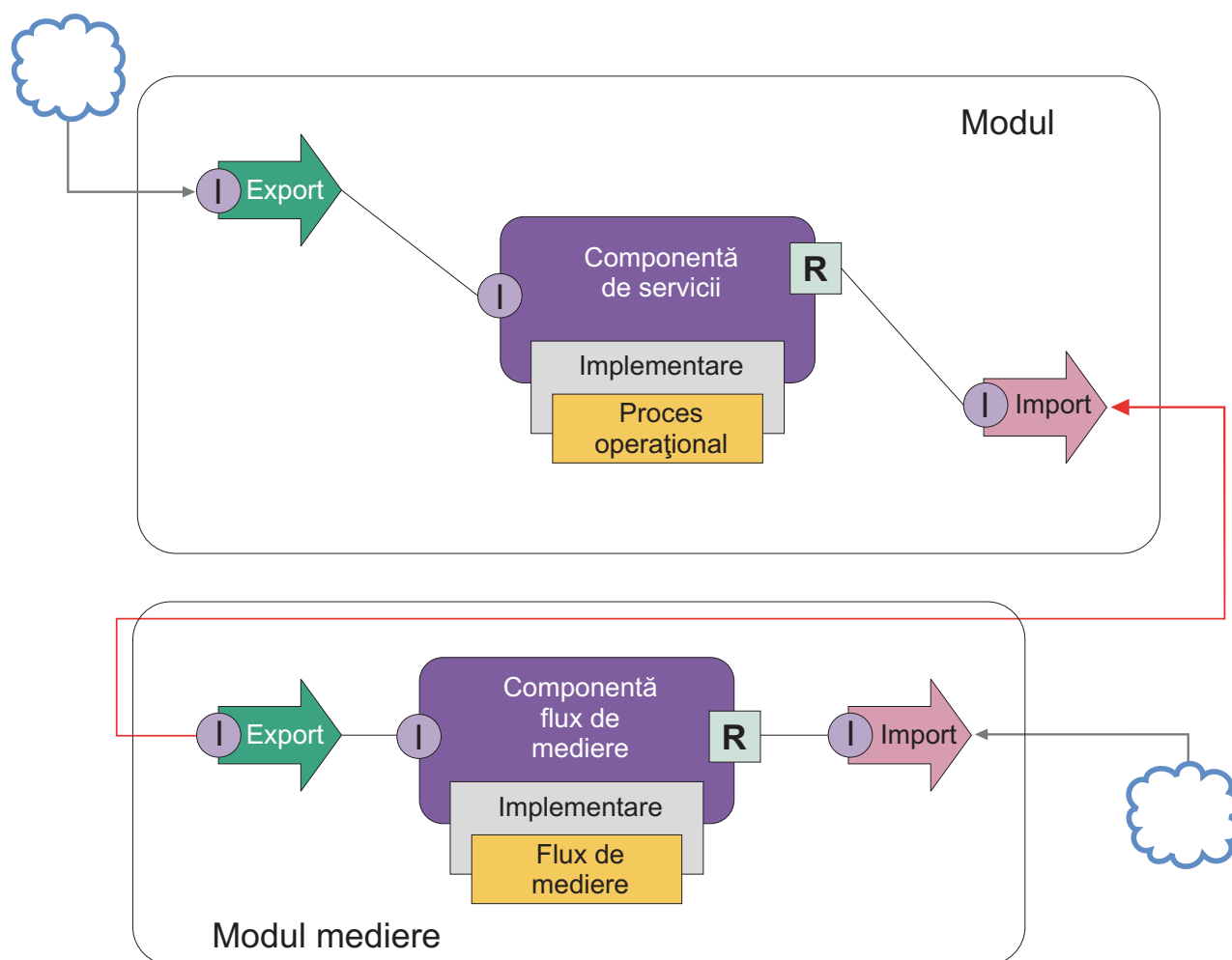
1. Crearea unui modul de mediere. Pentru instrucțiuni pas-cu-pas, vedeți Crearea modulelor de mediere .
2. În modulul de mediere, creați un import cu legarea corespunzătoare pentru serviciul extern pe care doriți să-l accesați. Pentru instrucțiuni pas-cu-pas, vedeți Crearea importurilor. Pentru informații suplimentare despre legări, vedeți Legări
3. Creați un export și dați-i aceeași interfață ca și importului. Pentru instrucțiuni pas-cu-pas, vedeți Crearea exporturilor.
4. Generarea unei legări SCA pentru export. Pentru instrucțiuni pas-cu-pas, vedeți Generarea legărilor SCA
5. În montajul modulului de mediere, cablați exportul la import. Salvați modulul de mediere.

6. Crearea unui modul. Pentru instrucțiuni pas-cu-pas, vedeți Crearea unui modul pentru servicii operaționale
7. Adăugați un export și o componentă.
8. În vizualizarea Business Integration, trageți exportul pe care l-ați creat în modulul de mediere (la pasul 4) în asamblarea modulului. Va fi creat un import cu aceeași legare ca exportul.
9. Cablați exportul de componentă și componenta de import.
10. Adăugați implementarea componentei. Pentru informații despre tipuri de implementare, vedeți Implementări

Mai târziu, puteți adăuga logică de mediere precum logare sau direcționare către modulul de mediere fără a afecta modulul operațional.

Adăugarea medierii

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



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

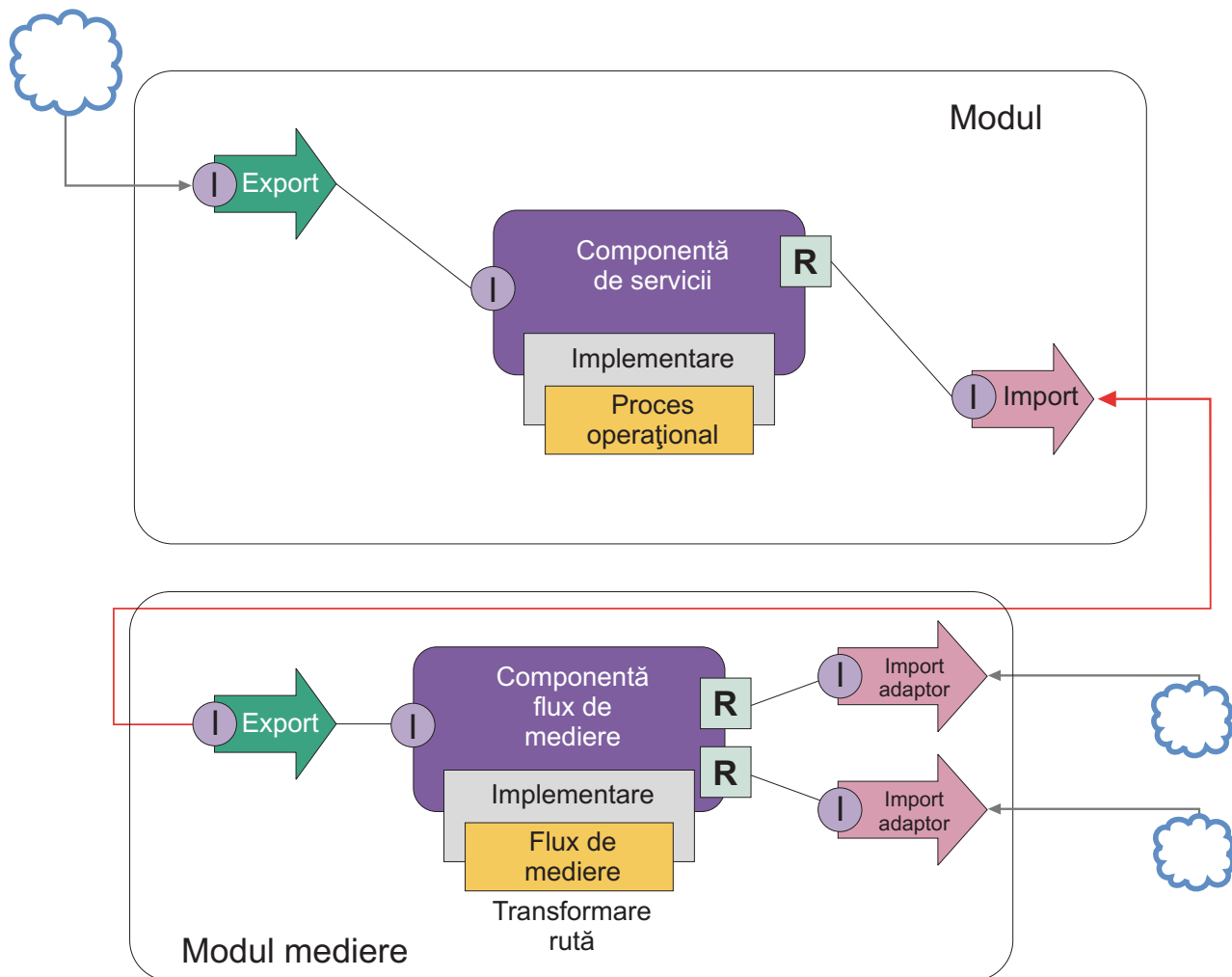
- Setarea anteturilor de protocol. Pentru informații suplimentare, vedeți subiectul Conversie de protocoale din centrul de informare WebSphere Enterprise Service Bus.
- Transformarea interfeței sau parametrilor prin utilizarea unei Mapări de obiecte business sau a unei primitive Mapare transformare. Transformarea mesajelor
- Selectarea unui anumit serviciu dintr-o listă statică, prin utilizarea unei primitive Filtre de mesaje. Filtre de mesaje

- Invocarea mai multor servicii pentru agregarea rezultatelor, prin utilizarea primitivelor Fan Out și Fan In. Agregarea și difuzarea mesajelor
- Tratarea eșurilor invocării de servicii prin reîncercarea aceluiași serviciu sau invocarea unui serviciu diferit, prin utilizarea unei primitive Service Invoke. Reîncercarea unei invocări de serviciu eșuate
- Rutare dinamică prin alegerea cărui serviciu să folosiți în timpul rulării, mai degrabă decât în timpul integrării, ceea ce permite serviciilor să fie cuplate mai liber și afacerilor să reacționeze 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. Selecting endpoints dynamically

Accesarea sistemelor EIS (Enterprise Information System)

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

Utilizarea unui modul de mediere intermediar între un modul și un sistem gazdă îl face mai reutilizabil. În exemplul de mai jos, un flux de mediere este folosit pentru rutarea la sistemul gazdă corect și 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 folosiți. Pentru informații despre cum să folosiți vrăjitorul de servicii externe, vedeți Tipar de accesare a serviciilor externe cu adaptoare
2. Crearea unui modul. Pentru instrucțiuni pas-cu-pas, vedeți Crearea unui modul pentru servicii operaționale
3. Adăugarea unui export, unei componente și unui import cu legare SCA. Pentru informații suplimentare, vedeți Apelarea serviciilor
4. Adăugarea unei interfețe exportului și cablarea exportului la componentă.
5. Adăugați implementarea componentei. În implementare, setați o proprietate ce indică ce serviciu gazdă va fi accesat. Pentru informații despre tipuri de implementare, vedeți Implementări
6. Crearea unui modul de mediere cu un export ce are o legare SCA și aceeași interfață ca importul modulului pe care l-ați creat la pasul 2.
7. Cablarea exportului la o componentă de flux de mediere.
8. Crearea unui import pentru fiecare sistem gazdă pe care doriți să-l accesați, folosind adaptorul de ieșire corespunzător din paleta editorului de montaj.
9. Cablarea componentei de flux de mediere la importuri.
10. Implementarea componentei de flux de mediere. Folosiți o primitivă Filtru de mesaje pentru a alege importul pe baza unui set de proprietăți din logica operațională și folosiți o primitivă Mapare pentru fiecare import adaptor.
11. În modul, selectați exportul modulului de mediere ca serviciul de importat în modul. Pentru informații pas-cu-pas, vedeți Invocarea unui serviciu dintr-un alt modul.

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

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 creați interfețe, obiecte business și legări pentru importuri și exporturi. Vedeți Maparea unui mesaj la o interfață SCA.

Fluxurile de mediere folosesc mesaje, ce oferă acces la informații de context și antet în plus față de obiectele business. Dacă doriți acces la informații de antet JMS sau o proprietate JMS personalizată, folosiți un flux de mediere. Dacă integrați cu un sistem MQ și doriți să accesați informații de antet MQ, folosiți un flux de mediere.

Crearea sau apelarea unui serviciu web

Serviciile Web sunt aplicații auto-conținute ce realizează funcții operaționale, de la o simplă interogare la interacțiuni de procese operaționale complexe. Puteți apela un serviciu web existent sau puteți dezvolta un nou serviciu web care să se potrivească cu necesitățile dumneavoastră. Acest scenariu va descrie pașii și vă va ghida către informațiile suplimentare.

Deși s-ar putea să nu vă creați toate serviciile de la zero folosind IBM Integration Designer, unele dintre serviciile dumneavoastră vor fi într-adevăr create astfel. Când lucrați cu editorul de montaje și cu editorul de procese operaționale pentru a monta servicii într-un proces operațional, este posibil să descoperiți că unele servicii lipsesc. De aceea, poate fi util să creați acele servicii lipsă folosind unelte IBM Integration Designer. Opusul este de asemenea adevărat - după ce ați creat un nou proces, puteți decide că ar fi util să expuneți toate sau un subset din operațiile de proces ca servicii pentru ca alții să le consume.

Notă: Acest scenariu se aplică utilizatorilor de IBM Integration Designer for IBM Process Server.

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

- Crearea serviciilor în IBM Integration Designer vă permite să implementați serviciul folosind reguli operaționale.
- Dezvoltarea în IBM Integration Designer vă permite să dezvoltați un serviciu Java™ și să îl expuneți atât ca serviciu web, cât și prin SCA.

- Maparea interfețelor fără a fi nevoie să codați este un avantaj. Puteți scoate toate mapările de date din codul Java lăsând un simplu program Java de cutie neagră pentru dezvoltatorul Java.
- IBM Integration Designer afișează toate serviciile și relațiile într-un singur loc.
- Abilitatea de a refactoriza va ajuta, de asemenea, la dezvoltarea de servicii web care utilizează IBM Integration Designer.

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

Exporturi, importuri și legări

IBM Integration Designer vă permite să importați servicii web standard și să utilizați aceste servicii în aplicațiile dumneavoastră compuse.

În IBM Integration Designer, utilizați editorul de montare pentru a dezvolta servicii. Urmați procesul standard pentru a crea module, module de mediere, biblioteci și componente. Apoi, puteți utiliza exporturi, importuri și legări pentru a partaja și accesa acele servicii. Pașii pentru acele taskuri elementare sunt afișați mai jos și legăturile conduc către informații mai detaliate pentru fiecare task.

Puteți utiliza oricare dintre cele două legări de servicii web - o legare de serviciu web sau o legare HTTP. O legare de serviciu web oferă o specificație pentru transmiterea mesajelor către și de la un serviciu web. Uneltele vă ajută să generați în mod automat o legare de serviciu web. O legare HTTP este un protocol cerere-și-răspuns standard între clienți și server după cum este definit de protocolul HTTP publicat de W3C (World Wide Web consortium). Va trebui să furnizați câteva informații inițiale de configurare a legării dacă folosiți 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. Creați un import pentru a apela un serviciu existent ce nu face parte din serviciul pe care îl montați.
 - Generarea unei legări de serviciu web pentru import.
 - Generarea unei legări de import HTTP.

Citiți subiectul legat dacă vreți să Invocare serviciu web din pagini JavaServer.

Capabilități de dezvoltare ale serviciilor Web

La deschiderea unui editor asociat cu procesul de creație a serviciilor web, ați putea vedea fereastra Confirmare activare care afișează următoarele informații:

Această acțiune necesită activare "Implementare servicii web".
Activați capabilitatea necesară?

IBM Integration Designer oferă o funcție de filtrare cunoscută drept *capabilități*. În setările Preferințelor, funcțiile și uneltele sunt clasificate în capabilități și puteți activa sau dezactiva categorii ale capabilităților sau funcțiile de subset ale oricărei categorii. Vedeți Capabilități pentru informații suplimentare.

Aflați mai multe despre conceptele cheie

Utilizați această secțiune ca un punct de plecare pentru investigarea tehnologiilor utilizate în și de către IBM Business Process Manager.

Crearea scenariilor

Folosiți scenariii pentru a înțelege și lucra cu componentele și produsele din familia gestionarea proceselor operaționale.

Versionarea

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

Modul în care funcționează această versionare în IBM Business Process Manager depinde de ce anume implementați - o aplicație de proces, implementată din magazie în IBM Process Center sau o aplicație de întreprindere implementată direct din IBM Integration Designer.

Aplicațiile de proces și trusele de unelte pe care le implementați într-un mediu de runtime din Process Center sunt versionate în mod implicit. Pentru aplicațiile de întreprindere, puteți alege să versionați modulele și bibliotecile în IBM Integration Designer.

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

Versionarea aplicațiilor de proces

Versionarea furnizează abilitatea, pentru mediul runtime, de a identifica instantaneele din ciclul de viață al unei aplicații de proces și de a fi capabil să ruleze simultan mai multe instantanee pe un Process Server.

Pentru a înțelege modul în care sunt versionate aplicațiile de proces, este important să reținem că o aplicație de procesare este un container care reține diverse artefacte folosite în sau de aplicația de proces (de exemplu, modelele de procesare sau BPD-urile, referințele seturilor de instrumente, serviciile, or tracks). Orice versionare este făcută la acest nivel de container, nu la nivelul artefactelor individuale. Pentru aplicații de proces, aceasta înseamnă că versionarea apare când realizați un instantaneu.

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

Puteți rula versiuni diferite (instantanee) ale unei aplicații de proces simultan pe un server; când instalați un instantaneu nou, fie înlăturați originalul fie lăsați-l să ruleze.

Context de versiune

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

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

Considerente de versionare pentru aplicații proces în clustere multiple

Puteți instala aceeași versiune de aplicație de proces la clustere multiple din aceeași celulă. Pentru a face diferența între aceste instalări multiple ale aceleiași versiuni a aplicației de proces, creați un instanțaneu pentru fiecare instalare și includeți un ID unic de celulă în numele instanțaneului (de exemplu, v1.0_cell1_1 și v1.0_cell1_2). Fiecare instanțaneu este o versiune nouă a aplicației de proces (dintr-o perspectivă de gestiune ciclu de viață pură), dar conținutul și funcția sunt la fel.

Când instalați o aplicație de proces într-un cluster, este realizată o sincronizare automată a nodurilor.

Considerente de versionare pentru truse de unelte Process Designer

Amintiți-vă că instanțaneele de aplicații proces sunt realizate în mod tipic când sunteți pregătit să testați sau să instalați. Instanțaneele de truse de unelte sunt însă realizate în mod tipic când sunteți pregătit ca acea trusă de unelte să fie utilizată de aplicații proces. După, dacă doriți să actualizați trusa de unelte, trebuie să faceți un alt instanțaneu al "sugestiei" atunci când sunteți pregătit și apoi deținătorii aplicațiilor de proces și ai truselor de unelte pot decide dacă doresc să treacă la noul instanțaneu.

Versionarea modulelor și bibliotecilor

Dacă un modul sau o bibliotecă este într-o aplicație de proces sau într-o trusă de unelte, acesta continuă ciclul de viață al aplicației de proces sau trusei de unelte (versiuni, instanțaneu, urme și așa mai departe). Numele modulelor și bibliotecilor trebuie să fie unice în cadrul domeniului unei aplicații de proces sau a unei truse de unelte.

Acest subiect descrie versionarea modulelor și a bibliotecilor care sunt folosite împreună cu aplicațiile de proces. Rețineți totuși că în cazul în care implementați modulele direct din IBM Integration Designer pe Process Server, puteți continua să urmați procedura de asignare a numerelor de versiune modulelor în timpul implementării, așa cum este descris în "Crearea modulelor și bibliotecilor cu număr de versiune".

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

Următorul tabel listează selecțiile pe care le puteți face în editorul de dependențe din IBM Integration Designer, atunci când o bibliotecă este asociată cu o aplicație de proces sau o trusă de unelte:

Tabela 7. Dependențele pentru Modul, Aplicație de proces sau Trusa de unelte și bibliotecile globale

Domeniu bibliotecă	Descriere	Poate depinde de . . .
Modul	Pe server există câte o copie a acestei biblioteci pentru fiecare modul care o folosește.	O bibliotecă din domeniul modului poate depinde de toate tipurile de biblioteci.
Aplicație de proces sau Trusă de unelte	Biblioteca este partajată între toate modulele din domeniul aplicației 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 Aplicație 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 configurați o bibliotecă partajată WebSphere pentru a putea implementa biblioteca globală. Vedeți "Dependențele modulelor și bibliotecilor" pentru informații suplimentare.

Module și biblioteci asociate cu aplicații de proces sau cu truse de unelte

Nu aveți nevoie să versionați module și biblioteci asociate cu aplicații de proces sau cu truse de unelte.

Modulele și bibliotecile asociate cu aplicații de proces sau cu truse de unelte nu au nevoie să fie versionate. De fapt, nu puteți crea o versiune a unui modul sau bibliotecă asociate cu aplicații de proces sau cu truse de unelte în editorul de dependențe. Modulele și bibliotecile asociate cu aplicații de proces sau cu truse de unelte utilizează instanțane, o funcție în Process Center, pentru a realiza același rezultat ca o versiune.

Bibliotecile, asociate cu aplicații de proces sau cu truse de unelte, nu vor avea un număr de versiune necesar în secțiunea Bibliotecii a editorului de dependențe, deoarece nu este necesară nicio versiune.

Convenții de numire

O convenție de numire este folosită pentru a diferenția diversele versiuni ale unei aplicații de proces pe măsură ce se mută prin ciclul de viață de actualizare, implementarea, co-implementare, dezimplementare și arhivare.

Această secțiune vă oferă convențiile care sunt utilizate pentru a identifica în mod unic versiunile unei aplicații de proces.

Un *context de versiune* este o combinație de acronime care descriu în mod unic o aplicație de proces sau o trusă de unelte. Fiecare tip de acronim are o convenție de numire. Acronimul este limitat la o lungime maximă de șapte caractere din setul de caractere [A-Z0-9_], cu excepția acronimului instanțaneului, care poate include un punct.

- Acronimul aplicației de proces este creat o dată cu aplicația. Acesta poate avea o lungime maximă de șapte caractere.
- Acronimul instanțaneului este creat în mod automat la crearea instanțaneului. Acesta poate avea o lungime maximă de șapte caractere.

Dacă numele instanțaneului îndeplinește criteriile pentru un acronim valid de instanțaneu, numele și acronimul instanțaneului vor fi identice.

Notă: Atunci când utilizați funcția de rutare dependentă de versiune pentru componenta fluxului de mediere, denumiți-vă instanțaneul, astfel încât să fie în concordanță cu schema <versiune>.<ediție>.<modificare> (de exemplu, **1.0.0**). Deoarece acronimul instanțaneului este limitat la șapte caractere, valorile cifrelor sunt limitate la un maxim de cinci cifre (cinci cifre plus două perioade). Prin urmare, trebuie să aveți grijă atunci când câmpurile cifre sunt incrementate deoarece orice oricare depășește primele șapte caractere este trunchiat.

De exemplu, un nume de instanțaneu **11.22.33** rezultă într-un acronim de instanțaneu 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 și acronimul implicit pentru pistă 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 definiție de proces operațional într-o aplicație de proces este identificată de obicei prin acronimul numelui procesului operațional, acronimul instanțaneului și numele definiției procesului operațional. Alegeți nume unice pentru definițiile procesului operațional ori de câte ori este posibil. Atunci când există nume duplicate, este posibil să întâmpinați următoarele probleme:

- S-ar putea să fiți în imposibilitatea de a expune definițiile procesului operațional sub formă de servicii web fără o anumită formă de mediere.
- S-ar putea să fiți în imposibilitatea de a invoca o definiție a procesului operațional creat în IBM Process Designer dintr-un proces BPEL creat în IBM Integration Designer.

Contextul de versionare variază în funcție de cum este implementată aplicația de proces.

Convenții de numire pentru implementarea pe servere Process Center:

Pe serverul IBM Process Center, puteți implementa un instantaneu al unei aplicații de proces, precum și un instantaneu al unei truse de unelte. În plus, puteți implementa sugestia unei aplicații de proces sau sugestia unei truse de unelte. (O *sugestie* reprezintă versiunea curentă funcțională a aplicației dumneavoastră de proces sau a trusei de unelte.) Contextul de versionare variază în funcție de tipul implementării.

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

Trusele de unelte pot fi implementate cu una sau mai multe aplicații de proces, dar ciclul de viață al fiecărei truse de unelte este legat de ciclul de viață al aplicației de proces. Fiecare aplicație 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 aplicațiile de proces.

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

Pentru mai multe informații, vedeți secțiunea "Exemple" la pagina 31, mai târziu în acest subiect.

Instantanee ale aplicației de proces

Pentru implementarea instantaneelor aplicațiilor de proces, contextul de versiune este o combinație între articolele următoare:

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

Truse de unelte autonome

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

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

Sugestii

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

Pentru implementările sugestiilor pentru aplicația de proces, contextul de versionare este o combinație între articolele următoare:

- Acronim pentru numele aplicației de proces
- Acronim pentru pista aplicației de proces (în cazul în care se folosește 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 producție.

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

- Acronim pentru numele trusei de unelte

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

Exemple

Resursele ar trebui să fie numite în mod unic și 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 aplicației de proces folosește numele de pistă implicit (**Main**):

Tabela 8. Sugestie pentru aplicația de proces cu nume implicit pentru pistă

Tip de nume	Exemplu
Nume aplicație de proces	Process Application 1
Acronim pentru numele aplicației de proces	PA1
Pistă aplicație de proces	Main
Acronim pentru urma aplicației de proces	"" (când pista este Main)
Instantaneu aplicație de proces	
Acronim pentru instantaneul aplicației de proces	Type

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

Tabela 9. Module SCA și fișiere EAR dependentă de versiune

Nume modul SCA	Nume dependent de versiune	EAR dependent de versiune/nume aplicație
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 aplicație de proces care folosește un nume neimplicit pentru pistă:

Tabela 10. Sugestie pentru aplicația de proces cu nume neimplicit pentru pistă

Tip de nume	Exemplu
Nume aplicație de proces	Process Application 1
Acronim pentru numele aplicației de proces	PA1
Pistă aplicație de proces	Track1
Acronim pentru urma aplicației de proces	T1
Instantaneu aplicație de proces	
Acronim pentru instantaneul aplicației de proces	Type

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

Tabela 11. Module SCA și fișiere EAR dependentă de versiune

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

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

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

Tabela 12. Instantaneu pentru aplicația de proces cu nume implicit pentru pistă

Tip de nume	Exemplu
Nume aplicație de proces	Process Application 1
Acronim pentru numele aplicației de proces	PA1
Pistă aplicație de proces	Main
Acronim pentru urma aplicației de proces	"" (când pista este Main)
Instantaneu aplicație de proces	Process Shapshot V1
Acronim pentru instantaneul aplicației de proces	PSV1

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

Tabela 13. Module SCA și fișiere EAR dependentă de versiune

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

- Următorul tabel arată un exemplu de instantaneu de aplicație de proces care folosește un nume de pistă neimplicit:

Tabela 14. Instantaneu pentru aplicația de proces cu nume neimplicit pentru pistă

Tip de nume	Exemplu
Nume aplicație de proces	Process Application 1
Acronim pentru numele aplicației de proces	PA1
Pistă aplicație de proces	Track1
Acronim pentru urma aplicației de proces	T1
Instantaneu aplicație de proces	Process Snapshot V1
Acronim pentru instantaneul aplicației de proces	PSV1

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

Tabela 15. Module SCA și fișiere EAR dependentă de versiune

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

Convenții de numire pentru implementarea Process Server:

Pe un Process Server, puteți implementa instantaneul unei aplicații de proces. Acronimul instantaneului aplicației de proces este folosit pentru a identifica versiunea în mod unic.

Pentru implementarea instantaneelor aplicațiilor de proces, contextul de versionare este o combinație între articolele următoare:

- Acronim pentru numele aplicației de proces
- Acronim instantaneu pentru aplicația de proces

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

Tabela 16. Exemple de nume și acronime

Tip de nume	Exemplu
Nume aplicație de proces	Process Application 1
Acronim pentru numele aplicației de proces	PA1
Instantaneu aplicație de proces	1.0.0
Acronim instantaneu pentru aplicația 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 și modul în care fișierele EAR asociate includ contextul de versiune:

Tabela 17. Module SCA și fișiere EAR dependente de versiune

Nume modul SCA	Nume dependent de versiune	EAR dependent de versiune/nume aplicație
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 aplicației de proces și modul în care fișierele JAR asociate includ contextul versiunii:

Tabela 18. Biblioteci din domeniul aplicației de proces și fișiere JAR dependente de versiune

Nume bibliotecă din domeniul aplicației 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

Legările dependente de versiune

Aplicațiile de proces pot conține module SCA care includ legări de import și export. Atunci când co-implementați aplicații, legarea pentru fiecare versiune a aplicației 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 actualizați legarea după implementare pentru a îi asigura unicitatea.

O legare *dependentă de versiune* este pusă în domeniul unei anumite versiuni ale aplicației de proces, lucru care garantează unicitatea între aplicațiile de proces. Următoarele secțiuni descriu legările care sunt actualizate automat pentru a fi dependente de versiune, precum și orice acțiune pe care aveți nevoie să o efectuați la momentul rulării atunci când o legare nu este dependentă de versiune. Pentru mai multe informații despre lucrurile care se iau în considerare atunci când creați module, vedeți “Considerente atunci când se utilizează legări”.

SCA

Ținta unei legări SCA este redenumită în mod automat pentru a fi dependentă de versiune în timpul implementării în cazul în care legările de import și export ale modulului sunt definite în același domeniu al aplicației de proces.

În cazul în care legările nu sunt definite în același domeniu al aplicației de proces, se înregistrează un mesaj de informare. Trebuie să modificați legarea de import după implementare pentru ca adresa țintă a punctului final să se modifice. Puteți utiliza consola administrativă pentru a modifica adresa țintă a punctului final.

Serviciu Web (JAX-WS sau JAX-RPC)

Adresa țintă 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:

- Ați urmat convenția implicită de numire pentru adresa:
`http://ip:port/ModuleNameWeb/sca/ExportName`
- Punctul final al adresei este SOAP/HTTP.
- Legările de import și export ale modulului sunt definite în același domeniu al aplicației de proces.

Dacă aceste condiții nu sunt adevărate, este înregistrat un mesaj de informare. Acțiunea pe care o întreprindeți apoi depinde de modul în care vă implementați aplicația de proces:

- Dacă vă co-implementați aplicația de proces, trebuie să redenumiți manual URL-ul punctului final pentru SOAP/HTTP sau coada de destinație SOAP/JMS, astfel încât să fie unic între versiunile aplicației de proces. Puteți utiliza consola administrativă după implementare pentru a modifica adresa țintă a punctului final.
- Dacă implementați doar o singură versiune a aplicației de proces, aveți posibilitatea să ignorați acest mesaj

Pentru co-implementarea instantaneului legării serviciului Web SOAP/ JMS, acțiunea pe care o întreprindeți depinde de modul în care vă implementați aplicația de proces:

- În cazul în care importul și exportul țintă sunt în aceeași aplicație de proces, realizați pașii următori înainte de a publica aplicația de proces pe Process Center și de a crea instantaneul:
 1. Modificați URL-ul punctului final pentru export. Asigurați-vă că destinația și fabrica de conexiuni sunt unice.
 2. Modificați URL-ul punctului final pentru import, astfel încât să fie același cu cel pe care l-ați specificat pentru export la pasul anterior.
- În cazul în care importul și exportul țintă sunt în aplicații de proces diferite, realizați pașii următori:
 1. Modificați URL-ul punctului final pentru export. Asigurați-vă că destinația și fabrica de conexiuni sunt unice.
 2. Publicați aplicația de proces la Process Center.
 3. Creați instantaneul.
 4. Implementați aplicația de proces pe Process Server.
 5. Utilizați consola administrativă WebSphere pentru a modifica URL-ul punctului final pentru importul corespunzător, astfel încât să fie același cu cel specificat pentru export.

HTTP

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

- Ați urmat convenția implicită de numire pentru adresa:
`http(s)://ip:port/ModuleNameWeb/contextPathinExport`
- Legările de import și export ale modulului sunt definite în același domeniu al aplicației de proces.

Dacă aceste condiții nu sunt adevărate, este înregistrat un mesaj de informare. Acțiunea pe care o întreprindeți apoi depinde de modul în care vă implementați aplicația de proces:

- Dacă co-implementați aplicația de proces, trebuie să redenumiți manual URL-ul punctului final, astfel încât să fie unic între versiunile aplicației de proces. Puteți utiliza consola administrativă după implementare pentru a modifica adresa țintă a punctului final.
- Dacă implementați doar o singură versiune a aplicației de proces, aveți posibilitatea să ignorați acest mesaj

JMS și Generic JMS

Legările JMS generate de sistem și legările generice JMS sunt conștiente de versiune în mod automat.

Notă: Pentru legările JMS definite de utilizator și legările Generic JMS, nu are loc redenumirea automată în timpul implementării pentru ca legările să devină conștiente de versiune. În cazul în care legarea este definită de utilizator, trebuie să redenumiți următoarele atribute, astfel încât să fie unice între versiunile aplicațiilor de proces:

- Configurarea punctului final
- Coadă destinație de recepție
- Nume port ascultător (dacă este definit)

Setați destinația Trimite corespunzătoare în cazul în care modificați punctul final al modulului țintă.

MQ/JMS și MQ

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

Trebuie să redenumiți atributele următoare, astfel încât să fie unice între versiunile aplicațiilor de proces:

- Configurarea punctului final
- Coadă destinație de recepție

Setați destinația Trimite corespunzătoare în cazul în care modificați punctul final al modulului țintă.

EJB

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

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

Rețineți că aplicațiile client trebuie să fie 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 (**ModuleNameApp: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 aplicației de proces.

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

Invocarea dinamică dependentă de versiune

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

În cazul în care folosiți stilul IBM_VRM (<versiune>.<ediție>.<modificare>) pentru instantaneu, puteți exporta fișierul EAR al aplicației de proces în WSRR (WebSphere Service Registry and Repository). Atunci când creați modulul de mediere, configurați atunci și căutarea punctelor finale, astfel încât să folosească rutarea dependentă de versiune. De exemplu, selectați **Returnare punct final care se potrivește cu cea mai recentă versiune compatibilă pentru serviciile bazate pe modulul SCA** în câmpul **Politică potrivire**, apoi selectați **SCA** pentru **Tip legare**.

Versiunile viitoare ale aplicației de proces sunt implementate pe server și 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.

Rețineți ca o alternativă, că puteți seta ținta în SMOHeader, iar valoarea poate fi purtată de către mesajul de cerere.

Implementarea aplicațiilor de proces cu module și proiecte Java

Aplicațiile de proces pot conține module Java EE și proiecte Java personalizate. Atunci când co-implementați aplicații, modulul Java EE personalizat pentru fiecare versiune a aplicației trebuie să fie unic.

Rețineți că modulele Java EE și proiectele Java personalizate sunt implementate pe un server în cazul în care acestea sunt implementate cu un modul SCA care are o dependență declarată în acestea. Dacă nu selectați **Implementare cu modul** (care este implicit) atunci când declarați dependența, trebuie să implementați modulul sau proiectul în mod manual.

Implementarea aplicațiilor de proces cu ajutorul regulilor operaționale și a selectoarelor

În cazul în care implementați mai multe versiuni ale unei aplicații de proces care include o regulă operațională sau o componentă de tip selector, fiți atenți la modul în care versiunile folosesc metadatele asociate.

Metadatele dinamice pentru o regulă operațională sau pentru o componentă de tip selector sunt definite în momentul rulării prin numele componentei, numele spațiului pentru componenta țintă și tipul componentei. În cazul în care două sau mai multe versiuni ale aplicației de proces care conțin o regulă operațională sau un selector sunt implementate în același mediu de runtime, acestea vor partaja aceeași logică pentru reguli (regulă operațională) sau aceleași metadate (selector) pentru rutare.

Pentru a permite fiecărei versiuni a regulii operaționale sau a componentei de tip selector ce aparține de aplicația de proces să își utilizeze propriile metadate dinamice (logică pentru reguli sau rutare), modificați codul (refactorizați) pentru spațiul de nume țintă, astfel încât acesta să fie unic pentru fiecare versiune a aplicației de proces.

Obiecte de configurație

Puteți folosi comenzile AdminConfig ale uneltei de administrare în linie de comandă WebSphere (wsadmin) pentru a accesa și modifica baza de date și proprietățile de securitate din IBM Business Process Manager.

Termenul *obiect configurație* se referă la un obiect care este accesat prin folosirea comenzilor wsadmin AdminConfig. Pentru mai multe informații, vedeți Proprietățile configurației de securitate.

Arhitectura de implementare

Arhitectura de implementare IBM Business Process Manager conține procese software numite servere, unități topologice referite ca noduri și celule și magazia de configurare utilizată pentru memorarea informațiilor de configurare.

Celulele

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

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

Datele de configurare administrative sunt memorate în fișiere XML. O celulă reține fișiere de configurare master pentru fiecare server din fiecare nod din celulă. Fiecare nod și server are, de asemenea, propriile fișiere locale de configurare. Modificările fișierului local de configurare a unui nod sau a unui server sunt temporare dacă serverul aparține celei. Când sunt efective, modificările locale înlocuiesc configurațiile celei. Modificările fișierelor de configurare ale serverului master și ale nodului master făcute la nivelul celei înlocuiesc orice modificare temporară făcută asupra nodului când documentele configurației celei sunt sincronizate cu nodurile. Sincronizarea apare la evenimente desemnate, cum ar fi pornirea unui server.

Serverele

Serverele furnizează funcționalitatea de bază a IBM Business Process Manager. Servere Process extind sau măresc abilitatea unui server de aplicații de a manipula module SCA (Service Component Architecture). Alte servere (manageri de implementare și agenți 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 colecție de servere gestionate, cluster-e de servere și alte middleware-uri se numește un *mediu de implementare*. Într-un mediu de implementare, fiecare server sau cluster gestionat este configurat pentru o anumită funcție în mediul de implementare (de exemplu, gazdă destinație, gazdă modul de aplicație sau server Common Event Infrastructure). Un server autonom este configurat să furnizeze toate funcțiile cerute.

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

Un *agent de nod* este un agent administrativ ce reprezintă un nod din sistemul dvs. și gestionează serverele acelui nod. Agenții de nod monitorizează serverele de pe un sistem de gazdă și 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 și cluster-e.

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

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 țintă de implementare, suportul de mesaje, managerul de reguli de proces operațional și un server Common Event Infrastructure.

Un server autonom este ușor de setat și are o consolă Pornire rapidă de unde puteți porni și opri serverul și puteți deschide galerie de eșantioane și consola administrativă. Dacă instalați eșantioanele IBM Business Process Manager și apoi deschideți galeria de eșantioane, o soluție exemplu este implementată în serverul autonom. Puteți explora resursele utilizate pentru acest eșantion din consola administrativă.

Puteți implementa propriile dumneavoastră soluții într-un server autonom, dar un server autonom nu poate furniza capacitatea, scalabilitatea sau robustețea care este cerută de un mediu de producție. Pentru mediul dumneavoastră de producție, este mai bine să utilizați un mediu Network Deployment.

Este posibil să porniți cu un server autonom și mai târziu să îl includeți într-un mediu Network Deployment, federalizându-l la o celulă a managerului de implementare, *asigurat că niciun alt nod nu a fost federalizat la acea celulă*. Nu este posibil să federalizați mai multe servere autonome într-o celulă. Pentru a federaliza serverul autonom, utilizați consola administrativă a managerului de implementare sau comanda **addNode**. Serverul autonom nu trebuie să ruleze când îl federalizați 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 și care participă la gestiunea încărcării de lucru.

Un cluster poate conține noduri sau servere individuale de aplicații. Un nod este de obicei un calculator fizic cu o adresă IP a gazdei distinctă care rulează unu sau mai multe servere de aplicații. Cluster-ele pot fi grupate sub configurația unei celule care asociază logic multe servere și cluster-e cu configurații și aplicații diferite unul cu celălalt în funcție de discreția administratorului și de ceea ce are sens în mediile lor organizaționale.

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

Deoarece fiecare membru al cluster-ului conține aceleași aplicații, puteți distribui taskuri client în funcție de capacitățile diferitelor mașini prin asignarea de ponderi fiecărui server.

Asignarea de ponderi serverelor dintr-un cluster îmbunătățește performanța și preluarea la defect. Taskurile sunt asignate serverelor care au capacitatea de a realiza operațiile 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 aplicații care poate deveni supraîncărcat dacă sunt făcute prea multe cereri.

Profilurile

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

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

Utilizați Utilitarul de linie de comandă BPMConfig pentru a crea profiluri IBM BPM. Utilitarul linie de comandă **manageprofiles** sau interfața lui grafică de utilizator, Profile Management Tool (PMT), poate fi utilizat ca mod alternativ de a crea managerul de implementare sau profiluri de noduri gestionate. PMT nu mai este suportate pentru crearea unui profil autonom.

Notă: În platformele distribuite, fiecare profil are un nume unic. Pe z/OS, toate profilurile sunt numite “default”; nu puteți redenumi, edita, copia și șterge profiluri pe z/OS.

Tipuri de profiluri

Sunt disponibile următoarele tipuri de profiluri IBM BPM cu IBM Business Process Manager V8.5:

Profil autonom IBM BPM

Profilul autonom definește servere autonome cu capabilități și funcționalitate specifice în configurațiile IBM BPM Express. Puteți crea profilul autonom utilizând șablonul de profil BPM/BpmServer, care este suportat numai în IBM BPM Express.

Profil de manager de implementare IBM BPM

Profilul de manager de implementare definește un manager de implementare care asigură o interfață administrativă pentru un grup logic de servere pe una sau mai multe spații de lucru. Puteți crea profilul de manager de implementare utilizând șablonul de profile BPM/BpmDmgr, care este suportat numai în IBM BPM Standard și IBM BPM Advanced.

Profil de nod gestionat IBM BPM

Profilul de nod gestionat definește un nod gestionat când nodul este federalizat la un manager de implementare. Puteți crea profilul de nod gestionat utilizând șablonul de profiluri BPM/BpmNode, care este suportat numai în IBM BPM Standard și IBM BPM Advanced.

Tipurile specifice de profiluri sunt disponibile cu fiecare configurație IBM BPM.

Tabela 19. Tipuri disponibile de profiluri IBM BPM

Configurarea IBM BPM	Tipuri de profiluri		
	Autonom	Managerul de implementare	Nod gestionat
IBM BPM Express	Da	Nu	Nu
IBM BPM Standard	Nu	Da	Da

Tabela 19. Tipuri disponibile de profiluri IBM BPM (continuare)

Configurarea IBM BPM	Tipuri de profiluri		
	Autonom	Managerul de implementare	Nod gestionat
IBM BPM Advanced	Nu	Da	Da
Unit Test Environment (UTE) for IBM Integration Designer	Da	Opțional	Opțional

Directorul profilului

Fiecare profil din sistem are propriul său director care conține toate fișierele sale. Specificați locația 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 Dmgr (manager de implementare) este în C:\Program Files\IBM\WebSphere\ProcServer\profiles\Dmgr.

Profilul implicit

Primul profil pe care îl puteți crea într-o instalare a IBM Business Process Manager este *profilul implicit*. Profilul implicit este destinația 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 creați un alt profil, puteți să îl faceți implicit.

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

Augmentare profiluri

Dacă aveți deja un profil de manager de implementare, un profil de nod gestionat sau un profil de server autonom creat pentru WebSphere Application Server Network Deployment, îl puteți *augmenta* pentru a suporta IBM Business Process Manager în plus față de funcția existentă. Pentru a augmenta un profil, instalați mai întâi IBM Business Process Manager. Apoi, utilizați utilitarul linie de comandă **manageprofiles** pentru a augmenta un profil autonom, sau utilizați fie Profile Management Tool fie utilitarul linie de comandă **manageprofiles** pentru a augmenta un profil de manager de implementare sau de nod gestionat.

Important: Într-un mediu Network Deployment, trebuie să augmentați mai întâi profilul de manager de implementare, și apoi augmentați profilurile de nod gestionat.

Restricție: Nu puteți augmenta un profil de manager de implementare sau autonom unde registrul de utilizatori WebSphere VMM implicit a fost modificat, de exemplu, să folosească LDAP.

Managerii de implementare

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

La crearea unui mediu de implementare, profilul managerului de implementare este primul profil pe care îl creați. Fiecare mediu de implementare pe care-l creați are o consolă Pornire rapidă (Quick Start), din care puteți porni și opri managerul de implementare și puteți porni consola sa administrativă. Folosiți consola administrativă a managerului de implementare pentru a gestiona serverele și cluster-ele din celulă. Aceasta include configurarea serverelor și a cluster-elor, adăugarea serverelor la cluster-e, pornirea și oprirea serverelor și a cluster-elor și implementarea modulelor SCA.

Deși managerul de implementare este un tip de server, nu puteți implementa module pe însuși 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 configurația sa și 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 și care conține agent nod și poate conține servere gestionate. Într-un nod gestionat, puteți configura și rula serverele gestionate.

Serverele care sunt configurate pe un nod gestionate formează resursele mediului dumneavoastră de implementare. Aceste servere sunt create, configurate, pornite, oprite, gestionate și șterse 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 configurația profilului. De exemplu, când realizați următoarele taskuri:

- Porniți și opriți procesele serverului.
- Sincronizați datele de configurare de pe managerul de implementare cu copia de pe nod.

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

Un nod gestionat poate conține unul sau mai multe servere, care sunt gestionate de un manager de implementare. Puteți implementa soluții pentru servere într-un nod gestionat, dar nodul gestionat nu conține o galerie de eșantioane de aplicații. Nodul gestionat este definit de un profil de nod gestionat și are o consolă Pornire rapidă (Quick Start).

Nodurile negestionate:

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

Nodurile negestionate din topologia Network Deployment pot avea definiții de servere cum ar fi servere web, dar nu definiții de servere de aplicații. 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.

Agenții de nod

Agenții de nod sunt agenți administrativi care rutează cereri administrative la servere.

Un agent de nod este un server care rulează pe fiecare calculator gazdă care participă la configurația Network Deployment. Este pur și simplu un agent administrativ și nu este implicat în funcții care deserveșc aplicații. Un agent de nod găzduiește, de asemenea, alte funcții administrative importante precum servicii de transfer de fișiere, sincronizare de configurare și monitor de performanță.

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

Acest subiect discută termeni rezervați și probleme pe care trebuie să le luați în considerare la numirea profilului, nodului, serverului, gazdei și 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 restricții. Nu folosiți niciunul din următoarele caractere când vă denumiți profilul:

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

Sunt permise caractere pe doi octeți.

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: Folosiți o convenție de numire cale scurtă când creați 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 și celulă

Nume rezervate: Evitați 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 și nume de gazde și nod, Gazdă și Nume celule: Utilizați indicațiile corespunzătoare pentru a denumi atunci când creați profiluri.

- Profilurile serverului autonom
- Profilurile managerului de implementare
- Profile de noduri gestionate

Tabela 20. Indicații de denumire pentru profiluri de server autonome

Nume câmp	Valoare implicită	Constrângeri	Descriere
Nume nod	Linux UNIX Windows <i>shortHostName</i> Node <i>NodeNumber</i> unde: <ul style="list-style-type: none">• <i>shortHost Name</i> este numele scurt al gazdei.• <i>NodeNumber</i> este un număr secvențial care pornește de la 01.	Evitați folosirea numelor rezervate.	Selectați orice nume doriți. Pentru a ajuta organizarea instalației dumneavoastră, folosiți un nume unic dacă aveți de gând să creați mai mult de un server pe sistem.
Nume server	Linux UNIX Windows server1	Folosiți un nume unic pentru server.	Numele logic pentru server.

Tabela 20. Indicații de denumire pentru profiluri de server autonome (continuare)

Nume câmp	Valoare implicită	Constrângeri	Descriere
Nume de gazdă	<p>Linux UNIX</p> <p>Windows Forma lungă a numelui serverului de nume domeniu (DNS - domain name server).</p>	Folosiți un nume de gazdă complet calificat care este adresabilă prin rețea.	Folosiți numele DNS actual sau adresa IP a stației dumneavoastră de lucru pentru a permite comunicația cu ea. Vedeți informații suplimentare despre numele gazdă urmând această tabelă.

Tabela 21. Indicații de denumire pentru profiluri de manager de implementare

Nume câmp	Valoare implicită	Constrângeri	Descriere
Nume nod	<p>Linux UNIX</p> <p>Windows <i>shortHostName</i> Cell ManagerNode Number unde:</p> <ul style="list-style-type: none"> <i>shortHost Name</i> este numele scurt al gazdei. <i>NodeNumber</i> este un număr secvențial care pornește de la 01. 	Folosiți un nume unic pentru managerul de implementare. Evitați folosirea numelor rezervate.	Numele este folosit pentru administrarea în cadrul celei manager de implementare.
Nume de gazdă	<p>Linux UNIX</p> <p>Windows Forma lungă a numelui serverului de nume domeniu (DNS - domain name server).</p>	Folosiți un nume de gazdă complet calificat care este adresabilă prin rețea. Evitați folosirea numelor rezervate.	Folosiți numele DNS actual sau adresa IP a stației dumneavoastră de lucru pentru a permite comunicația cu ea. Vedeți informații suplimentare despre numele gazdă urmând această tabelă.
Nume celulă	<p>Linux UNIX</p> <p>Windows <i>shortHostName</i> Cell CellNumber unde:</p> <ul style="list-style-type: none"> <i>shortHost Name</i> este numele scurt al gazdei. <i>CellNumber</i> este un număr secvențial care pornește de la 01. 	Folosiți un nume unic pentru celula manager de implementare. Un nume de celulă trebuie să fie unic în orice circumstanțe în care rulează produsul pe aceeași stație de lucru fizică sau cluster de stații de lucru, precum un Sysplex. În plus, un nume de celulă trebuie să fie unic în orice circumstanțe în care conectarea la rețea între entități 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 spațiile lor de nume vor fi federalizate. Altfel, ați putea întâlni excepții precum <code>javax.naming.NameNotFoundException</code> , caz în care trebuie să creați celule unic numite.	Toate nodurile federalizate devin membri ai celei manager de implementare, pe care ați numit-o în pagina Nume de noduri, gazde și celule a Profile Management Tool.

Tabela 22. Indicații de denumire pentru profiluri de noduri gestionate

Nume câmp	Valoare implicită	Constrângeri	Descriere
Nume nod	<p>Linux UNIX</p> <p>Windows <i>shortHostName</i></p> <p>Node <i>NodeNumber</i> unde:</p> <ul style="list-style-type: none"> <i>shortHost Name</i> este numele scurt al gazdei. <i>NodeNumber</i> este un număr secvențial care pornește de la 01. 	<p>Evitați folosirea numelor rezervate.</p> <p>Folosiți un nume unic în cadrul celei manager de implementare.</p>	<p>Numele este folosit pentru administrarea în cadrul celei manager de implementare la care este adăugat profilul de nod gestionat. Folosiți un nume unic în cadrul celei manager de implementare.</p>
Nume de gazdă	<p>Linux UNIX</p> <p>Windows Forma lungă a numelui serverului de nume domeniu (DNS - domain name server).</p>	<p>Folosiți un nume de gazdă complet calificat care este adresabilă prin rețea.</p>	<p>Folosiți numele DNS actual sau adresa IP a stației dumneavoastră de lucru pentru a permite comunicația cu ea. Vedeți informații suplimentare despre numele gazdă urmând această tabelă.</p>

Considerente privind numele gazdă:

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

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

Următoarele indicații vă pot ajuta la determinarea numelui de gazdă corespunzător pentru stația dumneavoastră de lucru:

- Selectați un nume de gazdă la care pot ajunge alte stații de lucru din rețeaua dumneavoastră.
- Nu folosiți identificatorul generic, localhost, pentru această valoare.
- Nu încercați să instalați produse IBM Business Process Manager pe un server cu un nume de gazdă care folosește caractere din setul de caractere pe doi octeți (DBCS - double-byte character set). Caracterele DBCS nu sunt suportate când sunt folosite în numele gazdă.
- Evitați folosirea caracterului liniuță de subliniere (_) în numele de server. Standardele Internet obligă ca numele de domenii să se conformeze cerințelor numelor gazdă descrise în Internet Official Protocol Standards RFC 952 și RFC 1123. Numele de domenii trebuie să conțină doar litere (majuscule sau minuscule) și cifre. Numele de domenii pot conține de asemenea caractere liniuță (-), atâta timp cât liniuțele nu se află la sfârșitul numelui. Caracterele liniuță de subliniere (_) nu sunt suportate în numele gazdă. Dacă ați instalat IBM Business Process Manager pe un server cu un caracter liniuță de subliniere în numele său, accesați serverul prin adresa sa IP până îl redenumiți.

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

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

- Șirul numelui gazdă DNS complet calificat, precum `xmachine.manhattan.ibm.com`
- Șirul 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 și flexibil. Aveți flexibilitatea de a schimba adresa IP actuală pentru sistemul gazdă, fără a schimba configurația serverului. Această valoare pentru numele gazdă este folosită în special dacă doriți să schimbați frecvent adresa IP când folosiți 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 fișierul gazdelor locale, pentru ca sistemul să poată rula serverul chiar când este deconectat de la rețea. Definiți numele scurt la 127.0.0.1 (loopback-ul local) în fișierul 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 distanță. 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 distanță se poate conecta la nodul pe care îl numiți 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 modificați setările proprietății hostName în documentele de configurare de fiecare dată când modificați adresa IP a stației de lucru. Prin urmare, nu folosiți o adresă IP numerică dacă utilizați DHCP sau dacă modificați regulat adresele IP. Un alt dezavantaj al acestui format este acela că nu puteți folosi nodul dacă gazda este deconectată de la rețea.

BPMN 2.0

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

BPMN (Business Process Model and Notation) este standardul de bază pentru procese din IBM Process Designer și IBM Process Center. Diagramele BPD (Business process definition) sunt bazate pe specificația BPMN. Acest subiect introduce unele dintre modalitățile în care BPMN 2.0 este aplicat în IBM Business Process Manager. Pentru informații detaliate despre BPMN, vedeți pagina Specificații BPMN la <http://www.bpmn.org/>.

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

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

Evenimentele mesaj intermediar IBM BPM oferă funcții asemănătoare cu taskul de trimitere BPMN și 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 și afișate în diagrame de proces. Pictogramele afișează dacă activitatea dumneavoastră este un task de sistem, un task utilizator, un task de decizie, script sau proces legat. Activitățile din modelele care au fost create în versiuni anterioare afișează, de asemenea, tipuri de taskuri BPMN 2.0 corespunzătoare și pictograme de taskuri la vizualizarea lor în versiunea 7.5.1 sau mai târziu.

Activități și taskuri

Există câteva modificări de terminologie de la versiunile anterioare ale Process Designer. Un număr al acestor modificări implică tipuri de activități care au fost redenumite.

- Activitățile serviciu (automate) sunt acum taskuri sistem.
- Activitățile serviciu (task) într-un culoar non-sistem sunt acum taskuri utilizator.

- Activitățile serviciu (task) într-un culoar sistem sunt acum taskuri de decizie dacă facă referire la un serviciu de decizie.
- Activitățile serviciu (task) într-un culoar sistem sunt acum taskuri sistem dacă fac referire la orice tip de serviciu altul decât serviciu de decizie.
- Activitățile Javascript sunt acum taskuri script.
- Activitățile proces imbricat sunt acum procese legate.
- Activitățile 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 notație în gateway-urile din versiunile anterioare. Totuși, 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 condițional 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 graniță întrerupe activitatea de care este atașată. Atunci când evenimentul este declanșat, activitatea se oprește iar jetonul continuă secvența ieșire a evenimentului. Dacă evenimentul este setat ca non-întrerupere atunci când evenimentul este declanșat activitatea atașată continuă în paralel și un nou jeton este generat și este transmis în fluxul secvență de ieșire al evenimentului. Granița evenimentului se modifică într-o linie întreruptă pentru evenimente care nu întrerup.

Evenimentele intermediare care sunt atașate activităților sunt evenimente intermediare de întrerupere dacă închid activitățile atașate lor sau evenimente intermediare non-întrerupere dacă nu închid activitățile atașate lor.

Eveniment de start

Specificația BPMN permite modelor de proces pentru să omită simbolurile de evenimente de start și de sfârșit. Process Designer necesită ca modelele de proces să folosească evenimente de start și de sfârșit.

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

proces

- fără
- mesaj
- ad hoc

subproces

- fără

subproces eveniment

- eroare
- mesaj
- cronometru

Puteți modifica tipul unui eveniment de start prin editarea proprietăților evenimentului. Puteți avea numeroase evenimente de start mesaj într-un proces, dar puteți folosi doar un eveniment fără start.

Oprire evenimente

Sunt disponibile patru tipuri de evenimente: *mesaj*, *terminare*, *eroare* și *none*. Puteți modifica tipul unui eveniment de sfârșit.

Când un proces părinte apelează un proces copil și procesul copil o acțiune de terminare eveniment, procesul copil se oprește și procesul părinte continuă apoi cu următorii pași.

Subproces

Specificația BPMN definește două tipuri de subproces, înglobate și refolosibile. Puteți crea ambele tipuri în Process Designer. Subprocesele înglobate sunt numite doar *subproces* în Process Designer și 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 conține și este o modalitate de grupare a pașilor procesului pentru a reduce complexitatea diagramei și confuzia. Subprocesele restrâng mai mulți pași într-o singură activitate. Subprocesul poate fi văzut doar de către procesul în care este definit. Un subproces există în domeniul apelantului său și are acces la toate variabilele din acel mediu. Nu există niciun parametru care să fie transmis în și în afara subprocesului înglobat.

Separat de subproces și 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 activități prin flux de secvențe, și apare doar dacă evenimentul de start este declanșat.

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 definiții de proces pot de asemenea apela acest proces. Procesul legat definește parametrii de intrare și de ieșire și nu are niciun acces la domeniul sau mediul apelantului. Procesul legat este similar cu procesul imbricat disponibil în versiunile anterioare; nu există nicio modificare în comportamentul activității. Procesele anterioare imbricate sunt migrate la procese legate. Procesul legat arată ca un subproces cu o limită compactă și este evidențiat în fereastra Inspector.

Bucle

BPMN oferă noțiunea de activitate care poate fi repetată. Activitatea poate fi atomică, însemnând că activitatea se repetă, sau poate fi un subproces, încapsulând o serie de pași care se repetă. Dacă expandați activitatea repetată, vedeți activitățile conținute care urmează să fie rulate în mod repetat. Condiția este evaluată întotdeauna la începutul fiecărei iterații buclă. Nu există nicio abilitate de a evalua la sfârșitul fiecărei iterații buclă.

IBM Business Process Manager are o *buclă cu mai multe instanțe* care este rulată de un număr finit de ori cu activitățile conținute în cadrul rulării secvențiale sau în paralel.

Import procese non-BPMN

Puteți importa modele care au fost create în IBM WebSphere Business Modeler și le puteți utiliza în Process Designer. Pentru informații despre importul BPMN 2.0, vedeți elementele Mapare IBM WebSphere Business Modeler în construcțiile IBM Business Process Manager. De asemenea, puteți importa modele BPMN 2.0 care au fost create în IBM WebSphere Business Compass, Rational Software Architect sau alte medii de modelare.

Definiții de proces operațional (BPD-uri)

Pentru a modela un proces în IBM Process Designer, trebuie să creați o definiție a procesului operațional (BPD). Definiția de proces operațional (BPD) poate fi bazată pe un model BPMN importat.

Un BPD este un model reutilizabil al unui proces, definind ceea ce este comun tuturor instanțelor runtime ale acelui model de proces. Un BPD trebuie să conțină un eveniment de start, un eveniment sfârșit, cel puțin un strat, și una sau mai multe activități. Consultați "Convențiile de numire IBM Process Designer" în legăturile înrudite pentru detalii despre limitarea caracterului care se aplică BPD-urilor.

Definiția de proces operațional (BPD) trebuie să includă un culoar pentru fiecare sistem sau grup de utilizatori care participă într-un proces. Un culoar poate fi un culoar participant sau un culoar sistem. Totuși puteți crea un BPD care grupează activitățile unui grup și a unui sistem într-un singur culoar dacă acea este preferința dumneavoastră. Consultați "Crearea unei definiții a procesului operațional (BPD)" în legăturile înrudite pentru informații despre cum se creează un BPD.

Puteți desemna orice persoană sau grup specific pentru a fi responsabil pentru activități într-un strat participant. Fiecare culoar pe care-l creați este atribuit implicit grupului Toți utilizatorii. Puteți folosi acest grup pentru a rula și testa definiția de proces operațional (BPD) în Inspector. Grupul Toți utilizatorii include toți utilizatorii care sunt membri grupului de securitate `tw_allusers`, care este un grup de securitate special care include automat toți utilizatorii din sistem.

Un strat sistem conține activități manipulate de un sistem IBM Process Center specific. Fiecare activitate necesită o implementare, care definește activitatea și seturile de proprietăți pentru task. În timpul implementării, un dezvoltator creează un serviciu sau scrie JavaScript necesar pentru a finaliza activitățile din stratul sistem. Consultați "Înțelegerea tipurilor de serviciu" în legăturile înrudite pentru informații despre servicii.

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

Puteți de asemenea să adăugați evenimente unui BPD. Evenimentele IBM BPM pot fi determinate de trecerea unei date de scadență, de o excepție sau primirea unui mesaj. Declanșatorul dorit determină tipul de eveniment pe care îl alegeți să-l implementați. Pentru informații detaliate despre tipurile de evenimente disponibile și despre declanșatorii lor, consultați "Modelarea evenimentelor".

Legările

La nucleul unei arhitecturi orientată-pe-servicii este conceptul unui *serviciu*, o unitate de funcționalitate realizată de o interacțiune între dispozitive de calcul. Un *export* definește interfața externă (sau punctul de acces) a unui modul, astfel încât componentele SCA (Service Component Architecture) din modul să-și poată oferi serviciile clienților externi. Un *import* definește o interfață către serviciile din afara modulului, astfel încât serviciile să poată fi apelate din modul. Utilizați *legări* specifice-protocolului cu exporturi și importuri pentru a specifica mijloacele de a transporta datele în și din modul.

Exporturile

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

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

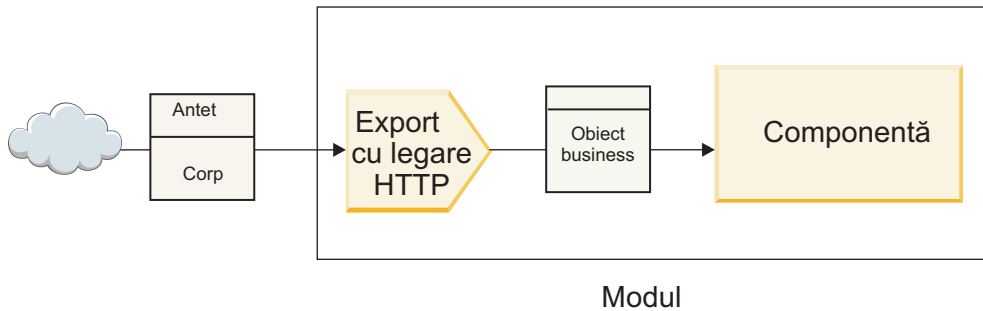


Figura 1. Un export cu legare HTTP

Importuri

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

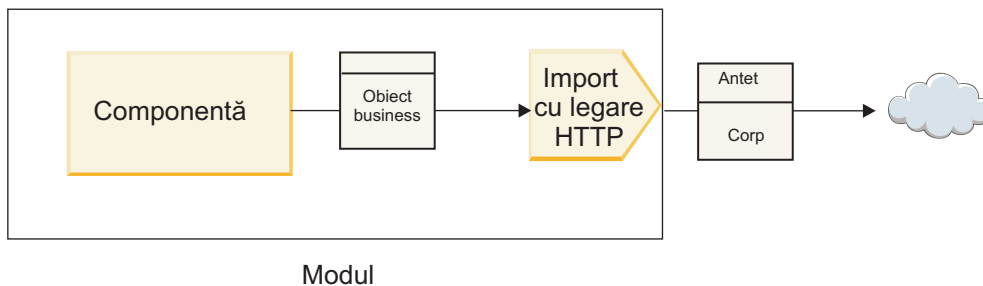


Figura 2. Un import cu legare HTTP

Listă cu legări

Utilizați Integration Designer pentru a genera o legare pentru un import sau un export și 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. Utilizați un import cu legare SCA pentru a accesa un serviciu dintr-un alt modul SCA. Utilizați un export cu legare SCA pentru a oferi un serviciu altor module SCA.
- Serviciu Web
O legare de serviciu web vă permite să accesați un serviciu extern folosind mesajele SOAP interoperabile și calitățile de serviciu. Puteți folosi, de asemenea, legările serviciului web pentru a include atașamente 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 transmite mesajele SOAP, legările serviciului manipulează întotdeauna interacțiunile cerere/răspuns în mod sincron.
- HTTP
Legarea HTTP vă lasă să accesați un serviciu extern folosind protocolul HTTP, unde sunt folosite mesaje non-SOAP sau unde este necesar acces HTTP direct. Legarea este utilizată atunci când lucrați cu servicii web care se bazează pe modelul HTTP (adică, servicii care folosesc operații bine cunoscute din interfața HTTP precum GET, PUT, DELETE, și așa mai departe).

- Enterprise JavaBeans (EJB)
Legările EJB permit componentelor SCA să interacționeze cu servicii furnizate de logica operațională 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 accesați servicii de pe un sistem de informații de întreprindere sau să vă faceți serviciile disponibile EIS-ului.
- Legări JMS
Legările Java Message Service (JMS), Generic JMS și WebSphere MQ JMS (MQ JMS) sunt folosite pentru interacțiuni cu sisteme de mesagerie, unde comunicarea asincronă prin cozi de mesaje este critică pentru fiabilitate. Un export cu una din legările JMS urmărește coada pentru sosirea unui mesaj și trimite asincron răspunsul, dacă există, cozii de răspunsuri. Un import cu una din legările JMS construiește și trimite un mesaj unei cozi JMS și urmărește o coadă pentru sosirea răspunsului, dacă există.
 - JMS
Legarea JMS vă lasă să accesați furnizorul JMS încorporat în WebSphere.
 - Generic JMS
Legarea Generic JMS vă lasă să accesați un sistem de mesagerie a unui vendor non-IBM.
 - MQ JMS
Legarea MQ JMS vă lasă să accesați subsetul JMS al unui sistem de mesagerie WebSphere MQ. Veți folosi această legare când subsetul JMS de funcții este suficient pentru aplicația dumneavoastră.
- MQ
Legarea WebSphere MQ vă lasă să comunicați cu aplicații native MQ, aducându-le în cadrul de lucru al arhitecturii orientate spre servicii și furnizând acces la informații de antet specifice MQ. Veți folosi această legare când aveți nevoie să folosiți funcții native MQ.

Privire generală asupra legării de export și import

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

Fluxul de informații printr-un export

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

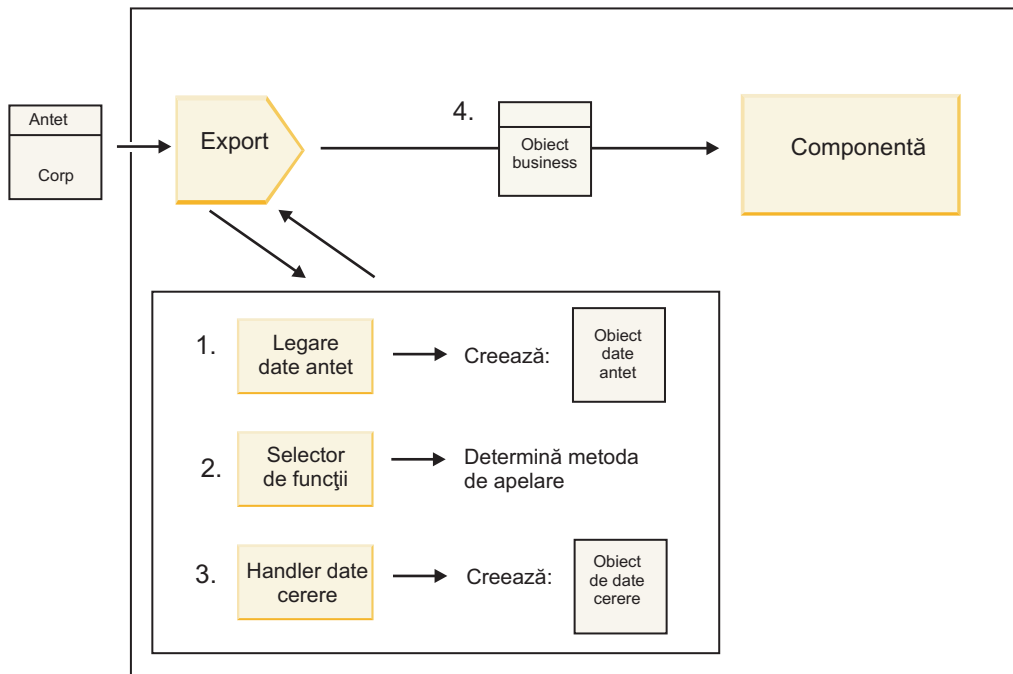


Figura 3. Fluxul unei cereri prin export către o componentă

Când exportul primește cererea, are loc următoarea secvență 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 funcții determină numele metodei native din mesajul de protocol. Numele metodei native este mapat de către configurația exportului de numele unei operații de pe interfața 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 și legarea de export EJB invocă componenta SCA în mod sincron.
 - Legările de export JMS, Generic JMS, MQ JMS și WebSphere MQ invocă componeta SCA asincron.

Notați că un export poate propaga anteturile și proprietățile de utilizator pe care le primește prin protocol, dacă propagarea contextului este activată. Componentele cablate de export pot accesa apoi aceste anteturi și proprietăți de utilizator. Vedeți subiectul “Propagare” din centrul de informare WebSphere Integration Developer pentru informații suplimentare.

Dacă aceasta este o operație pe două direcții, componenta returnează un răspuns.

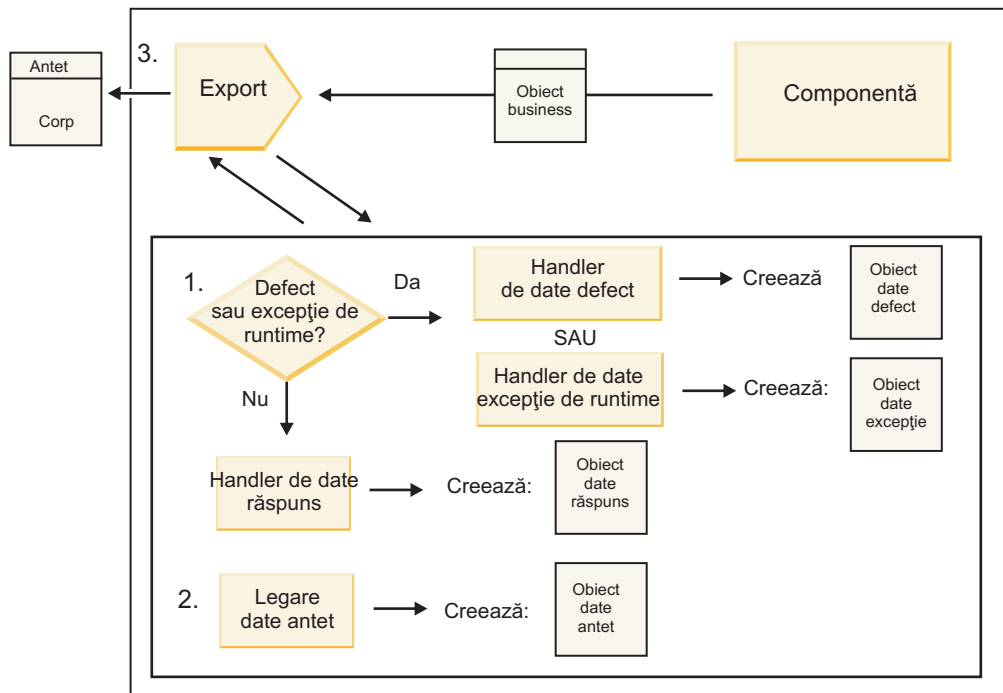


Figura 4. Fluxul unui răspuns înapoi prin export

Are loc următoarea secvență de pași:

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 excepție din timpul rulării, handler-ul de date pentru excepții 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 informațiilor printr-un import

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

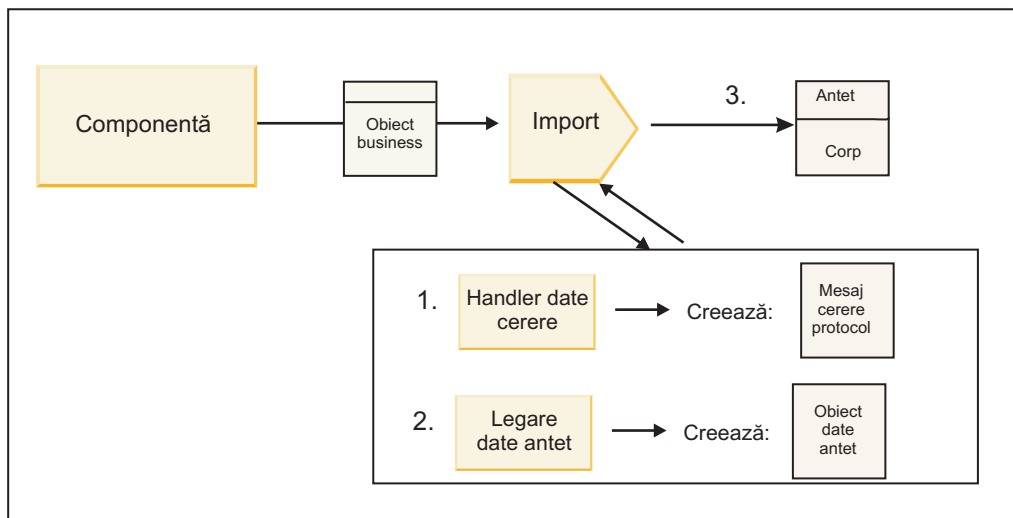


Figura 5. Fluxul de la o componentă prin import către un serviciu

Componenta invocă importul cu un obiect business de cerere.

Notă:

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

După ce componenta invocă importul, are loc următoarea secvență 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 operație pe două-căi, serviciul returnează un răspuns și are loc următoarea secvență de pași.

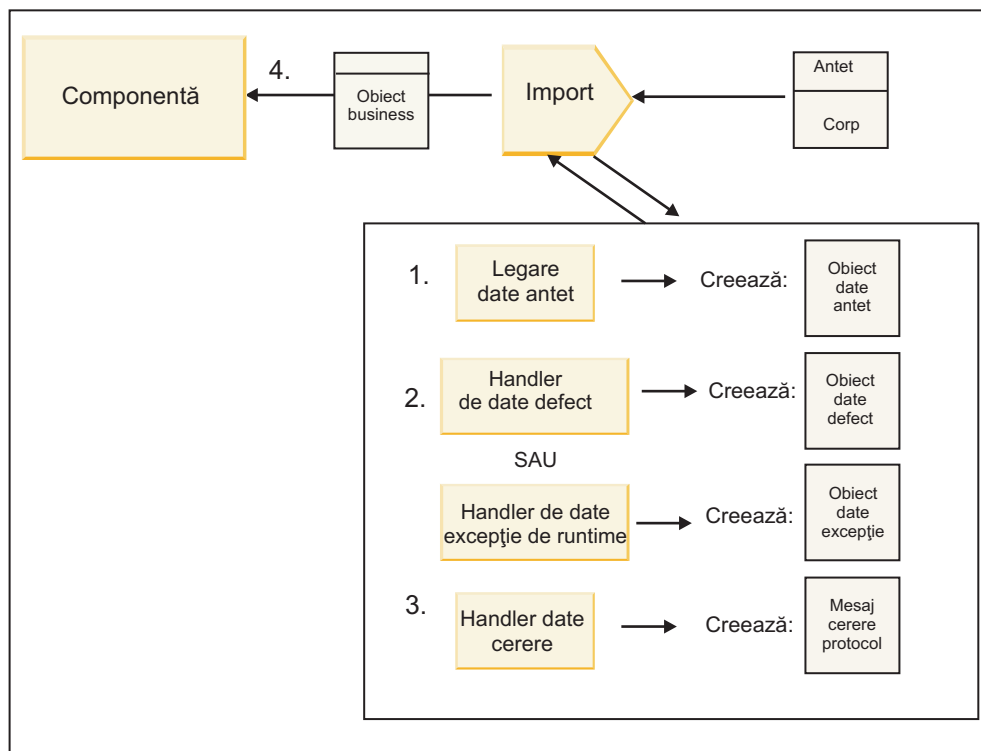


Figura 6. Fluxul unui răspuns înapoi prin import

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 excepție din timpul rulării, handler-ul de date pentru excepții 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 și import

Unul din aspectele cheie ale legărilor de export și 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, puteți de asemenea specifica un selector de funcții care să indice ce operație ar trebui efectuată pe date. Pentru legări asociate cu exporturi sau importuri, puteți indica cum ar trebui tratate defectele ce au loc în timpul procesării.

În plus, specificați informații specifice transportului pe legări. De exemplu, pentru o legare HTTP, specificați URL-ul punctului final. Pentru legarea HTTP, informațiile specifice transportului sunt descrise în subiectele “Generarea unei legări de import HTTP” și “Generarea unei legări de export HTTP”. Puteți de asemenea găsi informații despre alte legări în centrul de informare.

Transformarea formatului de date în importuri și exporturi:

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

- Pentru legări de export, unde o aplicație client trimite cereri către și primește răspunsuri de la o componentă SCA, indicați formatul datelor native. În funcție 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) și invers pentru a transforma obiectul business în date native (care este răspunsul către aplicația client).

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

IBM Business Process Manager oferă un set de formate de date predefinite și handler-e de date sau legări de date corespunzătoare ce suportă formatele. Vă puteți de asemenea crea propriile handler-e de date personalizate și înregistra formatul de date pentru acele handler-e de date. Pentru informații suplimentare, vedeți 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 și 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 și COBOL) într-un obiect business și un obiect business în date native. Deoarece sunt neutre din punct de vedere al protocolului, puteți reutiliza același handler de date cu o varietate de legări de export și import. De exemplu, puteți folosi același 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 (și 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 și HTTPServiceGatewayDataBinding) sunt depreciate începând cu IBM Business Process Manager Versiunea 7.0. Folosiți handler-e de date oricând este posibil.

După cum s-a menționat mai devreme, puteți crea handler-e de date personalizate, dacă este necesar. Puteți de asemenea legări de date personalizate; totuși, este recomandat să creați handler-e de date personalizate deoarece pot fi folosite peste multiple legări.

Handler-e de date:

Handler-ele de date sunt configurate funcție de legările de export și 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 puteți de asemenea crea propriul handler de date, dacă este necesar. Puteți asocia un handler de date cu o legare de export sau import la unul din două niveluri: îl puteți asocia cu toate operațiile din interfața exportului sau importului sau îl puteți asocia cu o anumită operație pentru cerere sau răspuns.

Handler-e de date predefinite

Utilizați IBM Integration Designer pentru a specifica handler-ul de date pe care vreți să îl utilizați.

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

Notă: Cu excepția notificărilor, aceste handler-e de date pot fi utilizate cu legări JMS, Generic JMS, MQ JMS, WebSphere MQ și HTTP.

Vedeți subiectul “Handler-e de date” din centrul de informare Integration Designer pentru informații mai detaliate.

Tabela 23. Handler-e de date predefinite

Handler de date	Date native în obiect business	Obiect business în 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 23. Handler-e de date predefinite (continuare)

Handler de date	Date native în obiect business	Obiect business în date native
Lățime fixată	Parsează date cu lățime fixată într-un obiect business.	Serializează un obiect business în date cu lățime 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 specificația Java Architecture for XML Binding (JAXB).	Deserializează un obiect de business în bean-uri Java folosind regulile de mapare definite de specificația 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 excepție într-un obiect business răspuns folosind regulile de mapare definite de specificația Java API for XML Web Services (JAX-WS).	Utilizat de o legare EJB pentru a transforma un obiect business în parametri Java de ieșire folosind regulile de mapare definite de specificația JAX-WS.
JSON	Parsează date JSON într-un obiect business.	Serializează un obiect business în date JSON.
Corp nativ	Parsează octeții, textul, maparea, fluxul sau obiectul nativ în unul din cinci obiecte business de bază (text, octeți, mapare, flux sau obiect).	Transformă cele cinci obiecte business de bază în octet, text, mapare, flux sau obiect.
SOAP	Parsează mesajul SOAP (și 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

Informații 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 și 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 puteți de asemenea crea propria legare de date, dacă este necesar. Puteți asocia o legare de date cu o legare de export sau import la unul din două niveluri, o puteți asocia cu toate operațiile din interfața exportului sau importului, sau o puteți asocia cu o operație specifică pentru cerere sau răspuns.

Utilizați IBM Integration Designer pentru a specifica legarea de date pe care vreți să o utilizați sau pentru a crea propria dumneavoastră legare de date. O discuție despre crearea legărilor de date poate fi găsită în secțiunea “Privire generală asupra JMS, MQ JMS și legările Generic JMS” 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 24. Legări de date predefinite pentru legări JMS

Legare de date	Date native în obiect business	Obiect business în date native
Obiect Java serializat	Transformă obiectul serializat Java într-un obiect business (care este mapat ca tipul de intrare sau ieșire în WSDL).	Serializează un obiect business în obiectul serializat Java în mesajul obiect JMS.
Octeți înfășurați	Extrage octeții din mesajul de octeți JMS de intrare și îi înfășoară în obiectul business JMSBytesBody.	Extrage octeții din obiectul business JMSBytesBody și îi înfășoară în mesajul de octeți JMS de ieșire
Intrare mapare înfășurată	Extrage informațiile despre nume, valoare și tip pentru fiecare intrare din mesajul de mapare JMS de intrare și creează o listă cu obiecte business MapEntry. Înfășoară apoi lista în obiectul business JMSMapBody	Extrage informațiile despre nume, valoare și tip din lista MapEntry din obiectul business JMSMapBody și creează intrările corespunzătoare în mesajul de mapare JMS de ieșire.
Obiect înfășurat	Extrage obiectul din mesajul obiect JMS de intrare și îl înfășoară în obiectul business JMSTextBody.	Extrage obiectul din obiectul business JMSTextBody și îl înfășoară în mesajul obiect JMS de ieșire.
Text înfășurat	Extrage textul din mesajul text JMS de intrare și îl înfășoară în obiectul business JMSTextBody.	Extrage textul din obiectul business JMSTextBody și îl înfășoară în mesajul text JMS de ieșire.

Legări WebSphere MQ

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

Tabela 25. Legări de date predefinite pentru legări WebSphere MQ

Legare de date	Date native în obiect business	Obiect business în 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 ieșire în WSDL).	Transformă un obiect business în obiectul serializat Java din mesajul de ieșire
Octeți înfășurați	Extrage octeții din mesajul de octeți MQ nestructurat și îi înfășoară în obiectul business JMSBytesBody.	Extrage octeții dintr-un obiect business JMSBytesBody și înfășoară octeții în mesajul de octeți MQ nestructurat de ieșire.
Text înfășurat	Extrage textul dintr-un mesaj text MQ nestructurat și îl înfășoară într-un obiect business JMSTextBody.	Extrage textul dintr-un obiect business JMSTextBody și îl înfășoară într-un mesaj text MQ nestructurat.
Intrare flux înfășurată	Extrage informațiile despre nume și tip pentru fiecare intrare din mesajul flux JMS de intrare și creează o listă cu obiectele business StreamEntry. Înfășoară apoi lista în obiectul business JMSSStreamBody.	Extrage informațiile despre nume și tip din lista StreamEntry din obiectul business JMSSStreamBody și creează intrări corespunzătoare în JMSSStreamMessage de ieșire.

În plus față de legările de date menționate în Tabela 25 la pagina 56, WebSphere MQ folosește de asemenea legări de date de antet. Vedeți Centrul de informare IBM Integration Designer pentru detalii.

Legări HTTP

Următorul tabel listează legările de date ce pot fi folosite cu HTTP și descrie taskurile realizate de legările de date.

Tabela 26. Legări de date predefinite pentru legări HTTP

Legare de date	Date native în obiect business	Obiect business în date native
Octeți înfășurați	Extrage octeții din corpul mesajului HTTP de intrare și îi înfășoară în obiectul business HTTPBytes.	Extrage octeții din obiectul business HTTPBytes și îi adaugă la corpul mesajului HTTP de ieșire.
Text înfășurat	Extrage textul din corpul mesajului HTTP de intrare și îl înfășoară în obiectul business HTTPText.	Extrage textul din obiectul business HTTPText și îl adaugă la corpul mesajului HTTP de ieșire.

Selectorii de funcții în legări de export:

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

Considerați un export SCA ce expune o interfață. Interfața conține două operații - Creare și Actualizare. Exportul are o legare JMS ce citește dintr-o coadă.

Când un mesaj ajunge în coadă, exportului îi sunt transmise datele asociate, dar ce operație din interfața exportului ar trebui invocată pe componenta cablată? Operația este determinată de selectorul de funcții și configurația legării de export.

Selectorul de funcții returnează numele funcției native (numele funcției din sistemul client ce a trimis mesajul). Numele funcției native este apoi mapat de numele operației sau funcției de pe interfața asociată cu exportul. De exemplu, în următoarea figură, selectorul de funcții returnează numele funcției native (CRT) din mesajul de intrare, numele funcției native este mapat de operația Creare și obiectul business este trimis componentei SCA cu operația Creare.

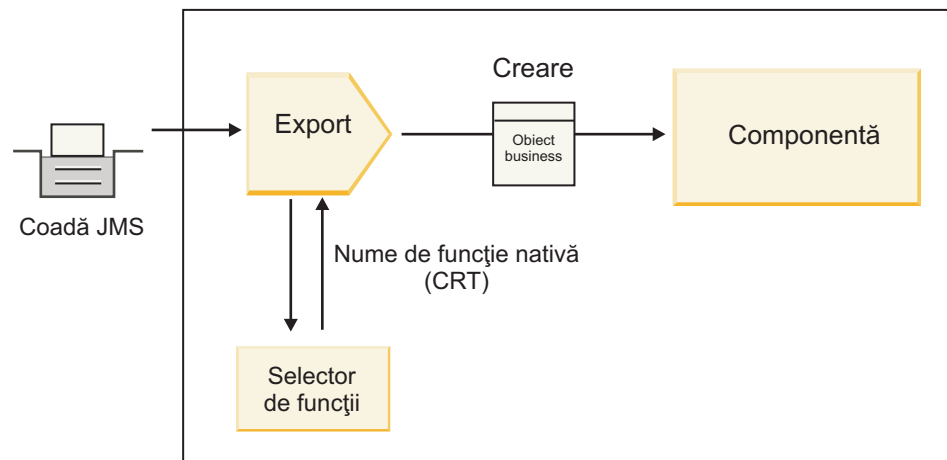


Figura 7. Selectorul de funcții

Dacă interfața are o singură operație, nu este nevoie să specificați un selector de funcții.

Mai mulți selectorii de funcții preambalați sunt disponibili și sunt menționați în secțiunile ce urmează.

Legări JMS

Următorul tabel listează selectorii de funcții ce pot fi folosiți cu:

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

Tabela 27. Selectorii de funcții predefiniți pentru legări JMS

Selector de funcții	Descriere
Selector de funcții JMS pentru legări simple de date JMS	Folosește proprietatea JMSType a mesajului pentru a selecta numele operației.
Selector de funcții a proprietății antetului JMS	Returnează valoarea Proprietății șir JSM, TargetFunctionName, din antet.
Selector funcție gateway de servicii JMS	Determină dacă cererea este o operație cu un singur sens sau cu sens dublu prin examinarea proprietății JMSReplyTo setată de client.

Legări WebSphere MQ

Următorul tabel listează selectorii de funcții ce pot fi utilizați cu legări WebSphere MQ.

Tabela 28. Selectorii de funcții predefiniți pentru legări WebSphere MQ

Selector de funcții	Descriere
Selector de funcții MQ handleMessage	Returnează handleMessage ca o valoare, ce este mapată folosind legările metodei de export de numele unei operații de pe interfață.
MQ folosește selector de funcții implicit JMS	Citește operația nativă din proprietatea TargetFunctionName din folderul unui antet MQRFH2.
MQ folosește formatul corpului mesajului ca funcție nativă	Găsește câmpul Format al ultimului antet și returnează acel câmp ca un Șir.
Selector de funcții de tip MQ	Crează o metodă în legarea dumneavoastră de export prin extragerea unui URL ce conține proprietățile Msd, Set, Type și Format aflate în antetul MQRFH2.
Selector funcție gateway de servicii MQ	Folosește proprietatea MsgType din antetul MQMD pentru a determina numele proprietății.

Legări HTTP

Următorul tabel listează selectorii de funcții ce pot fi utilizați cu legări HTTP.

Tabela 29. Selectorii de funcții predefiniți pentru legări HTTP

Selector de funcții	Descriere
Selector de funcții HTTP bazat pe antetul TargetFunctionName	Utilizează proprietatea antetului HTTP TargetFunctionName de la client pentru a determina ce operație să invoce în momentul rulării din export.
Selector de funcții HTTP bazat pe metoda URL și HTTP	Folosește calea relativă de la URL adăugată la sfârșitul metodei HTTP de la client pentru a determina operația nativă definită pe export.
Selector funcție gateway de servicii HTTP bazat pe URL cu un nume de operație	Determină metoda de invocare în funcție de URL dacă "operationMode = oneway" a fost adăugat la URL-ul cererii.

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

Tratarea defectelor:

Vă puteți configura legările de import și export să trateze defecte (de exemplu, excepții operaționale) ce au loc în timpul procesării prin specificarea handler-elor de date ale defectelor. Puteți un handler de date de defecte la trei niveluri - puteți asocia un handler de date de defecte cu un defect, cu o operație sau pentru toate operațiile cu o legare.

Un handler de date al defectelor procesează date ale defectelor și 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 excepției trimis de la componentă într-un mesaj răspuns ce poate fi folosit de aplicația 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 excepției 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 operațional sau o excepție runtime.

Puteți specifica un handler de date ale defectelor pentru un anumit defect, pentru o operație și pentru toate operațiile 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-urile de date ale defectelor sunt setate la nivelurile de operație și legare, este apelat handler-ul de date asociat cu operația.

Sunt folosite două editoare în IBM Integration Designer pentru specificarea tratării defectelor. Editorul de interfețe este folosit pentru a indica dacă va fi un defect pe o operație. După ce este generată o legare cu această interfață, editorul din vizualizarea proprietăților vă dă voie să configurați cum va fi tratat defectul. Pentru informații suplimentare, vedeți subiectul “Selectorii de defecte” din centrul de informare IBM Integration Designer.

Cum sunt tratate defectele în legări de export:

Când apare un defect în timpul procesării cererii de la o aplicație client, legarea de export poate returna informațiile despre defect clientului. Configurați legarea de export să specifice cum ar trebui procesat defectul și returnat clientului.

Configurați legarea de export folosind IBM Integration Designer.

În timpul procesării cererii, un client invocă un export cu o cerere și exportul invocă componenta SCA. În timpul procesării cererii, componenta SCA poate fie returna un răspuns operațional fie poate arunca o excepție operațională de serviciu sau o excepție runtime de serviciu. Când apare aceasta, legarea de export transformă excepția într-un mesaj de defect și îl trimite clientului, după cum este arătat în figura următoare și descris în secțiunile ce urmează.



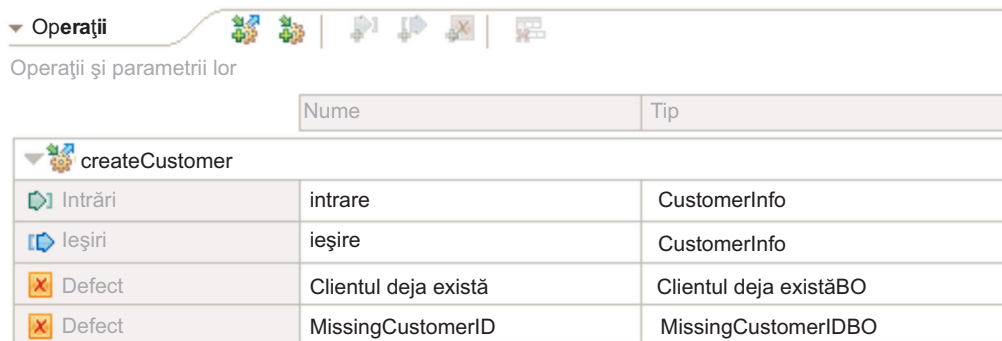
Figura 8. Cum sunt trimise informațiile despre defect de la componentă prin legarea de export clientului

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

Defecte operaționale

Defectele operaționale sunt erori sau excepții operaționale ce au loc în timpul procesării.

Considerați următoarea interfață, ce are o operație createCustomer pe ea. Această operație are două defecte operaționale definite: CustomerAlreadyExists și MissingCustomerId.



	Nume	Tip
▼ createCustomer		
▶ Intrări	intrare	CustomerInfo
▶ Ieșiri	ieșire	CustomerInfo
▶ Defect	Clientul deja există	Clientul deja existăBO
▶ Defect	MissingCustomerId	MissingCustomerIdBO

Figura 9. Interfață cu două defecte

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

Când un defect operațional 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 excepția operațională de servicii conține numele defectului, este apelat handler-ul de date ce este configurat pe defect. Dacă excepția operațională de servicii nu conține 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 excepția operațională de servicii.
3. Handler-ul de date de defecte transformă obiectul de date de defect într-un mesaj răspuns și îl returnează legării de export.
4. Exportul returnează mesajul răspuns clientului.

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

Excepții în timpul rulării

O excepție runtime este o excepție ce are loc în aplicația SCA în timpul procesării unei cereri ce nu corespunde unui defect operațional. Spre deosebire de defectele operaționale, excepțiile runtime nu sunt definite pe interfață.

În anumite scenarii, ați putea dori să propagați aceste excepții runtime către aplicația client, astfel încât aplicația client să poată lua măsurile corespunzătoare.

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

Notă: Puteți configura un handler de date al excepției runtime doar pe legări HTTP.

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

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

Tratarea defectelor și tratarea excepțiilor runtime sunt opționale. Dacă nu doriți să propagați defecte sau excepții runtime către clientul apelant, nu configurați handler-ul de date al defectelor sau handler-ul de date al excepției runtime.

Modul în care defectele sunt tratate în legările de import:

O componentă folosește 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 configurați legarea de import puteți specifica modul în care defectul ar trebui procesat și returnat către componentă.

Configurați legarea de import folosind IBM Integration Designer. Puteți specifica un handler pentru datele ce conțin defecte (sau o legare de date), și, de asemenea, puteți specifica un selector pentru defect.

Handler-e pentru datele ce conțin defecte

Serviciul care procesează cererea trimite, către legarea de import, informații despre defect sub forma unei excepții sau a unui mesaj de răspuns care conține datele de defect.

Legarea de import transformă excepția serviciului sau mesajul de răspuns într-un excepție a serviciului operațional sau o excepție de runtime serviciu, după cum se arată în figura de mai jos și cum este descris în secțiunile care urmează.

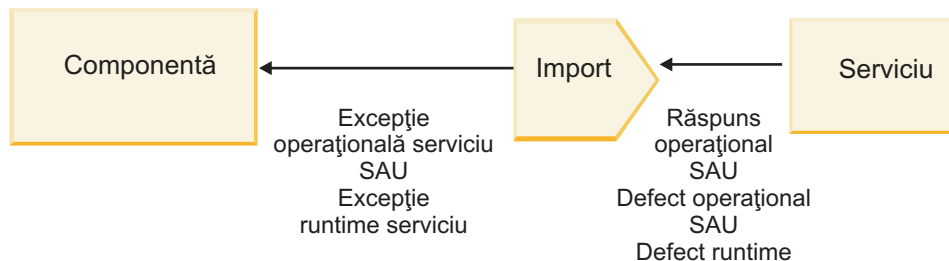


Figura 10. Modul în care informațiile despre eroare sunt transmise cu ajutorul importului de la serviciu către componentă

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

Selectorii defect

Atunci când configurați o legare de import, aveți posibilitatea să specificați un selector de defect. Selectorul de defect determină dacă răspunsul de import este un răspuns real, o excepție operațională sau un defect apărut în timpul rulării. De asemenea, determină din corpul sau antetul răspunsului, numele nativ al defectului, care este mapat prin configurația legării de numele defectului din interfața asociată.

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

Tabela 30. Selectoare pentru defecte împachetate

Tip selector defect	Descriere
Bazat pe antet	Stabilește dacă un mesaj de răspuns este un defect operațional, o excepție 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 excepție în timpul rulării.

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

- Selector de defect bazat pe antet

Dacă o aplicație dorește să indice că mesajul primit este un defect operațional, atunci trebuie să existe două anteturi în mesajul de intrare pentru defectele operaționale, așa cum este prezentat în continuare:

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

Dacă o aplicație dorește să indice că mesajul primit este o excepție apărută în timpul rulării, atunci trebuie să existe un antet în mesajul de intrare, așa cum este prezentat în continuare:

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

- Selector de defect SOAP

Un defect operațional poate fi trimis ca parte componentă a mesajului SOAP care are în compoziție 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 opțional. Dacă nu specificați un selector pentru defecare, legarea de import nu poate determina tipul răspunsului. Prin urmare, legarea îl tratează ca un răspuns operațional (business) și apelează handler-ul datelor de răspuns sau legarea de date.

Puteți crea un selector personalizat pentru defect. Pașii 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 operaționale

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

Atunci când legarea primește o excepție operațională, etapele de prelucrare depind de faptul dacă un selector de defect a fost setat pentru legare.

- Dacă nu a fost setat niciun 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 operațional (business) sau un defect apărut în timpul rulării.
 2. În cazul în care răspunsul este un defect operațional, 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 de date pentru defect care este configurat pentru acest defect WSDL.
 5. Legarea de import apelează acest handler-ul de date pentru defect cu datele defectului.
 6. Handler-ul de date pentru defect transformă datele defectului într-un obiect de date și îl returnează către legarea de import.

7. Legarea de import construiește un obiect de tip excepție pentru serviciul operațional cu obiectul de date și numele defectului.
8. Importul returnează obiectul de tip excepție pentru serviciul operațional către componentă.

Excepții din timpul rulării

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

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

Interoperabilitatea între modulele SCA și 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 aplicațiilor bazate pe specificațiile Open SCA. Modulele SCA din IBM Business Process Manager folosesc legările de import și export pentru a interopera cu serviciile Open SCA dezvoltate într-un mediu Rational Application Developer și găzduite de WebSphere Application Server Feature Pack for Service Component Architecture.

O aplicație SCA invocă o aplicație Open SCA prin intermediul unei legări de import. O aplicație SCA primește un apel de la aplicație Open SCA prin intermediul unei legări de export. O listă de legări acceptate este prezentată în “Invocarea serviciilor peste legările interoperabile” la pagina 65.

Invocarea serviciilor Open SCA din module SCA

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

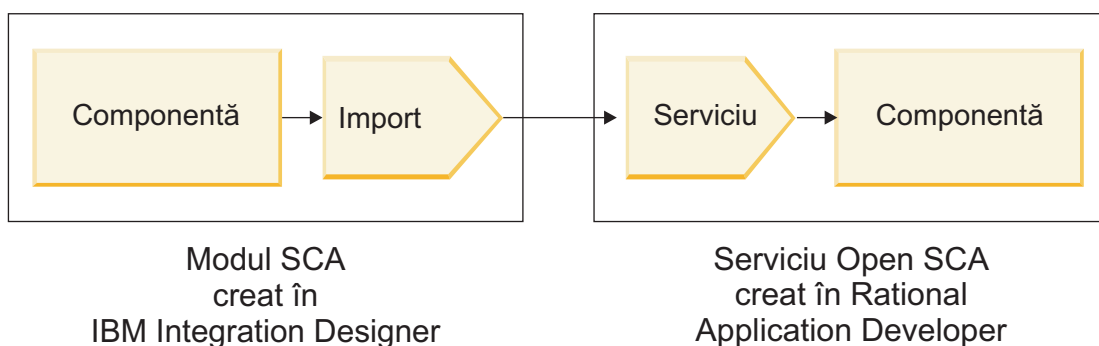


Figura 11. Componenta din modulul SCA invocă un serviciu Open SCA

Nu este nevoie de nicio configurație specială pentru a invoca un serviciu Open SCA.

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

1. Pentru a obține numele componentei țintă și serviciul din Open SCA compus, efectuați următorii pași:
 - a. Asigurați-vă că fila **Proprietăți** este deschisă făcând clic pe **Fereastră > Afișare vizualizare > Proprietăți**.

- b. Deschideți editorul pentru compus făcând dublu-clic pe diagrama compusă care conține componenta și serviciul. De exemplu, pentru o componentă ce are numele **customer**, diagrama compusă este **customer.composite_diagram**.
 - c. Faceți clic pe componenta țintă.
 - d. În câmpul **Nume** din fila **Proprietăți**, notați numele componentei țintă.
 - e. Faceți clic pe pictograma serviciului asociată cu componenta.
 - f. În câmpul **Nume** din fila **Proprietăți**, notați numele serviciului.
2. Pentru a configura importul IBM Business Process Manager, astfel încât să se conecteze la serviciul Open SCA, efectuați pașii următori:
 - a. În IBM Integration Designer, mergeți la fila **Proprietăți** din importul SCA pe care vreți să îl conectați la serviciul Open SCA.
 - b. În câmpul **Nume modul**, introduceți numele componentei de la pasul 1d.
 - c. În câmpul **Nume export**, introduceți numele serviciului de la pasul 1f.
 - d. Salvați-vă munca apăsând Ctrl+S.

Invocarea modulelor SCA din serviciile Open SCA

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

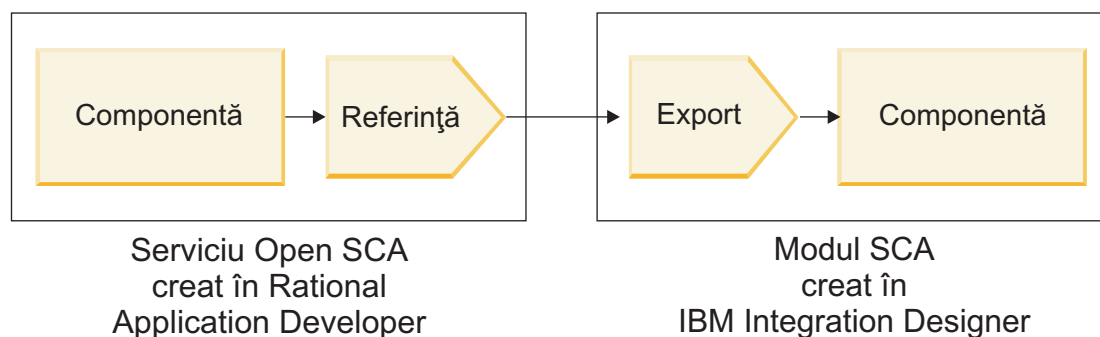


Figura 12. Deschiderea serviciului SCA prin invocarea componentei în modulul SCA

Pentru a vă conecta la o componentă SCA printr-o legare de referință a unui Open SCA, furnizați numele modulului și numele de export.

1. Pentru a obține numele modulului și exportului țintă, efectuați următorii pași:
 - a. În IBM Integration Designer, deschideți modulul în editorul de asamblare făcând dublu-clic pe modul.
 - b. Faceți clic pe export.
 - c. În câmpul **Nume** din fila **Proprietăți**, notați numele exportului.
2. Configurați referința Open SCA pe care doriți să o conectați la modulul IBM Business Process Manager și exportați:
 - a. În Rational Application Developer, deschideți editorul pentru compus făcând dublu-clic pe diagrama compusă care conține componenta și serviciul.
 - b. Faceți clic pe pictograma referinței componentei pentru a afișa aceste proprietăți de referință în fila **Proprietăți**.
 - c. Faceți clic pe fila **Legare** din partea stânga a paginii.
 - d. Faceți clic pe **Legări**, iar apoi faceți clic pe **Adăugare**.
 - e. Selectați legarea **SCA**.

- f. În câmpul **Uri**, introduceți numele modulului IBM Business Process Manager, urmat de o bară oblică (“/”), urmat de numele de export (pe care îl stabiliți la pasul 1c la pagina 64).
- g. Apăsați **OK**.
- h. Salvați-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 legări de import SCA, sunt acceptate următoarele stiluri de invocare:

- Asincron (într-un singur sens)
- Sincron (cerere/răspuns)

Interfața de import SCA și interfața serviciului Open SCA trebuie să utilizeze o interfață WSDL conformă WS-I. Rețineți că legarea SCA suportă propagarea contextului de tranzacție și de securitate.

- Legare (JAX-WS) a serviciului Web fie cu protocol SOAP1.1/HTTP, fie cu protocol SOAP1.2/HTTP

Interfața de import SCA și interfața serviciului Open SCA trebuie să utilizeze o interfață WSDL conformă WS-I.

În plus, sunt suportate următoarele calități ale serviciului:

- Tranzacția atomică a serviciilor Web
- Securitatea serviciilor Web

- Legare EJB

Pentru a defini interacțiunea dintre un modul SCA și un serviciu Open SCA se folosește o interfață Java atunci când este utilizată legarea EJB.

Rețineți că legarea EJB suportă propagarea contextului de tranzacție și de securitate.

- Legări JMS

Interfața de import SCA și interfața serviciului Open SCA trebuie să utilizeze o interfață WSDL conformă WS-I.

Sunt suportați următorii furnizori JMS:

- Platforma de mesagerie WebSphere (Legarea JMS)
- WebSphere MQ (Legarea MQ JMS)

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

Tipuri de legări

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

Selectarea legărilor corespunzătoare:

Atunci când creați o aplicație, trebuie să știți cum să selectați legarea care corespunde cel mai bine necesităților aplicației dumneavoastră.

Legările disponibile din IBM Integration Designer furnizează o gamă de alegeri. În această listă puteți determina care tip de legare este mai potrivită nevoilor aplicației dumneavoastră.

Luați în considerare o legare *SCA (Service Component Architecture)* atunci când se pot aplica acești factori:

- Toate serviciile sunt conținute în module; ceea ce înseamnă că nu există servicii externe.
- Vreți să separați funcția în module SCA diferite care interacționează direct unul cu celălalt.
- Modulele sunt cuplate strâns.

Luăți în considerare o legare *serviciu web* atunci când se aplică acești factori:

- Trebuie să accesați un serviciu extern de pe Internet sau să furnizați un serviciu pe Internet.
- Serviciile sunt cuplate slab.
- Comunicația sincronă este preferabilă; adică o cerere de la un serviciu poate aștepta pentru un răspuns de la altul.
- Protocolul serviciului extern pe care îl accesați sau al serviciului pe care vreți să îl furnizați este SOAP/HTTP sau SOAP/JMS.

Considerați o legare *HTTP* când acești factori sunt aplicabili:

- Este nevoie să accesați un serviciu extern de pe Internet sau să furnizați un serviciu de pe Internet și lucrați cu alte servicii web cum ar fi GET, PUT și DELETE.
- Serviciile sunt cuplate slab.
- Comunicația sincronă este preferabilă; adică o cerere de la un serviciu poate aștepta pentru un răspuns de la altul.

Luăți în considerare o legare *EJB (Enterprise JavaBeans)* atunci când se pot aplica acești factori:

- Legarea este pentru un serviciu importat care este el însuși un EJB sau care trebuie să fie accesat de clienți EJB.
- Serviciul importat este cuplat slab.
- Interacțiunile EJB stateful nu sunt necesare.
- Comunicația sincronă este preferabilă; adică o cerere de la un serviciu poate aștepta pentru un răspuns de la altul.

Luăți în considerare o legătură *EIS (Enterprise Information Systems)* atunci când se pot aplica acești factori:

- Trebuie să accesați un serviciu de pe un sistem EIS utilizând un adaptor de resurse.
- Transmisia de date sincronă este preferată în locul celei asincrone.

Luăți în considerare o legare *JMS (Java Message Service)* atunci când se pot aplica acești factori:

Important: Există mai multe tipuri de legări JMS. Dacă vă așteptați să schimbați mesajele SOAP folosind JMS, luați în considerare legarea de servicii web cu protocolul SOAP/JMS. Vedeți “Legări de serviciu Web” la pagina 67.

- Trebuie să accesați un sistem de mesagerie.
- Serviciile sunt cuplate slab.
- Transmisia de date asincron este preferată în locul celei sincron.

Luăți în considerare o legare *JMS (Java Message Service) generic* atunci când se pot aplica acești factori:

- Trebuie să accesați sistem de mesagerie a unui vendor non-IBM.
- Serviciile sunt cuplate slab.
- Fiabilitatea este mai importantă decât performanța; adică, transmisia de date asincronă este preferată în locul celei sincron.

Luăți în considerare o legare *MQ (Message Queue)* atunci când se pot aplica acești factori:

- Trebuie să accesați un sistem de mesagerie WebSphere MQ și să utilizați funcțiile native MQ.
- Serviciile sunt cuplate slab.
- Fiabilitatea este mai importantă decât performanța; adică, transmisia de date asincronă este preferată în locul celei sincron.

Luăți în considerare o legare *MQ JMS* când se aplică acești factori:

- Trebuie să accesați un sistem de mesagerie WebSphere MQ dar puteți face astfel într-un context JMS; adică, subsetul JMS al funcțiilor este suficient pentru aplicația dumneavoastră.
- Serviciile sunt cuplate slab.
- Fiabilitatea este mai importantă decât performanța; adică, transmisia de date asincronă este preferată în locul celei sincron.

Legări SCA:

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

Utilizați IBM Integration Designer pentru a genera și configura legări SCA în importuri și exporturi din module SCA.

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

După ce modulul care conține legarea SCA este implementat pe server, puteți utiliza consola administrativă pentru a vizualiza informații despre legare sau, în cazul unei legări de import, pentru a modifica proprietățile 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 (și vice versa).

Privire generală asupra legărilor de serviciu Web: **Advanced**

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

Cu o legare de serviciu web, puteți accesa serviciile externe folosind mesaje SOAP interoperabile și calitățile de serviciu (QoS).

Utilizați Integration Designer pentru a genera și configura legările serviciului web la importuri și exporturi în module SCA. Sunt disponibile următoarele tipuri de legări serviciu web:

- SOAP1.2/HTTP și SOAP1.1/HTTP

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

- Utilizați SOAP1.2/HTTP dacă serviciul dvs. web corespunde specificației SOAP 1.2.
- Utilizați SOAP1.1/HTTP dacă serviciul dvs. web corespunde specificației SOAP 1.1.

Important: Când implementați o aplicație cu o legare de serviciu web (JAX-WS), serverul țintă nu trebuie să aibă selectată opțiunea **Pornire componente după cum este necesar**. Pentru detalii, vedeți “Verificarea configurației serverului” la pagina 75.

Când selectați una din aceste legări, puteți trimite atașamente cu mesajele dvs. SOAP.

Legările serviciului web funcționează cu mesaje SOAP standard. Folosind una dintre legările serviciului web JAX-WS, totuși, aveți posibilitatea să personalizați modul în care sunt parsate sau scrise mesajele SOAP. De exemplu, puteți manipula elementele care nu sunt standard în mesajele SOAP sau puteți aplica prelucrări suplimentare mesajelor SOAP. Când configurați legarea specificați un handler personalizat pentru date, care efectuează această prelucrare într-un mesaj SOAP.

Puteți folosi seturi de politici cu o legare a serviciului web (JAX-WS). Un set de politici este o colecție 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 și a mesajelor. Utilizați Integration Designer pentru a selecta setul de politici pentru legare.

Notă: Dacă doriți să utilizați un set de politici de tip SAML (Security Assertion Markup Language), trebuie să realizați câteva configurații suplimentare, așa cum este descris în “Importul seturilor de politici” la pagina 73.

- SOAP1.1/HTTP

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

- SOAP1.1/JMS

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

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

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

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

După ce modulul SCA care conține legarea serviciului web este implementată pe server, aveți posibilitatea să utilizați consola administrativă pentru a vizualiza informații legate de legare sau să modificați proprietățile selectate a legării.

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

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

Propagarea antetului SOAP: **Advanced**

În momentul în care manipulați mesaje SOAP, este posibil să aveți nevoie să accesați informații din anumite anteturi SOAP din mesaje ce sunt primite, ca să vă asigurați că mesajele cu anteturi SOAP sunt trimise cu anumite valori sau ca să permiteți anteturilor SOAP să parcurgă un modul.

Când configurați legarea unui serviciu web în Integration Designer, puteți indica faptul că doriți ca anteturile SOAP să fie propagate.

- Atunci când cererile sunt primite într-un export sau răspunsurile sunt primite la un import, informațiile din antetul SOAP pot fi accesate, permițând ca logica din modul să se bazeze pe valorile antetului și permițând 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 și prezența anteturilor SOAP transmise pot fi afectate de seturile de politici configurate la import sau export, așa cum este explicat în Tabela 31 la pagina 70.

Pentru a configura transmiterea anteturilor SOAP pentru un import sau export, selectați (din modul de vizualizare Proprietăți al Integration Designer) fila **Antet Protocol de Transmitere** și selectați opțiunile de care aveți nevoie.

Anteturi de Adresare WS

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

Atunci când transmiteți antetul de adresare WS, fiți atenți la următoarele informații:

- Dacă activați transmiterea pentru antetul de adresare WS, antetul va fi transmis prin modul în următoarele circumstanțe:
 - 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 ieșire 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 WS-Security

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

Specificațiile WS-Security ale serviciilor web descriu îmbunătățirile aduse mesajelor SOAP pentru a oferi calitatea protecției prin integritatea mesajelor, confidențialitatea mesajelor și autentificarea unică a mesajului. Aceste mecanisme pot fi folosite pentru a acomoda o mare varietate de modele de securitate și tehnologii de criptare.

Atunci când transmiteți antetul WS-Security, fiți atenți la următoarele informații:

- Dacă activați transmiterea pentru antetul de WS-Security, antetul va fi transmis prin modul în următoarele circumstanțe:
 - 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. Totuși, dacă setați 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 inițial. 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 includeți schema WS-Security în modulul aplicației. Vedeți “Includerea schemei WS-Security într-un modul de aplicație” la pagina 70 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, așa cum este arătat în Tabela 31 la pagina 70:

Tabela 31. 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	<p>Anteturile de securitate sunt transmise așa cum sunt prin modul. Ele nu sunt decriptate.</p> <p>Anteturile sunt trimise către ieșire în aceeași formă în care au fost primite.</p> <p>Semnătura digitală ar putea deveni nevalidă.</p>	<p>Anteturile de securitate sunt decriptate și transmise prin modul cu verificarea semnăturii și autentificare.</p> <p>Anteturile decriptate sunt trimise către ieșire.</p> <p>Semnătura digitală ar putea deveni nevalidă.</p>
Legare de import cu politică de securitate	<p>Anteturile de securitate sunt transmise așa cum sunt prin modul. Ele nu sunt decriptate.</p> <p>Anteturile nu ar trebui să fie propagate către import. În caz contrar, din cauza duplicării apare o eroare.</p>	<p>Anteturile de securitate sunt decriptate și transmise prin modul cu verificarea semnăturii și autentificare.</p> <p>Anteturile nu ar trebui să fie propagate către import. În caz contrar, din cauza duplicării apare o eroare.</p>

Configurați seturile de politici corespunzătoare pentru legările de export și import, deoarece acestea izolează solicitantul serviciului de modificările aduse configurației sau cerințelor QoS ale furnizorului de servicii. Anteturile SOAP standard vizibile într-un modul pot fi utilizate pentru a influența prelucrarea (de exemplu, înregistrarea în istoric și urmărirea) în modulul. Propagarea anteturilor SOAP printr-un modul ce aparțin de un mesaj recepționat 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 așa fel încât să prevină propagarea anteturilor protocolului.

Accesarea anteturilor SOAP

Atunci când este primit un mesaj care conține anteturi SOAP printr-un import sau export de serviciu web, anteturile sunt puse în secțiunea anteturi a obiectului mesaj al serviciului (SMO - service message object). Puteți accesa informațiile din antet, așa cum este descris în "Accesarea informațiilor din antetul SOAP în SMOO".

Includerea schemei WS-Security într-un modul de aplicație

Procedura următoare definește pașii pentru includerea schemei în modulul aplicației:

- În cazul în care computerul pe care rulează Integration Designer are acces la Internet, efectuați următorii pași:
 1. În perspectiva Business Integration, selectați **Dependente** pentru proiectul dumneavoastră.
 2. Extindeți **Resurse predefinite** și selectați fie **Fișiere schemă WS-Security 1.0**, fie **Fișiere schemă WS-Security 1.1** pentru a importa schema în modulul dumneavoastră.
 3. Curățați și reconstruiți proiectul.
- În cazul în care un computer pe care rulează Integration Designer nu are acces la Internet, puteți descărca schema pe un al doilea computer care are acces la Internet. Apoi o puteți copia pe computerul pe care rulează Integration Designer.
 1. De pe computerul care are acces la Internet, descărcați schema de la distanță:
 - a. Faceți clic pe **Fișier > Import > Business Integration > WSDL și XSD**.
 - b. Selectați **WSDL la distanță** sau **Fișier XSD**.
 - c. Importați 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/xmlsig-core/xmlsig-core-schema.xsd

2. Copiați schemele pe un computer care nu are acces la Internet.
3. De pe computerul care nu are acces la Internet, importați schema:
 - a. Faceți clic pe **Fișier > Import > Business Integration > WSDL și XSD**.
 - b. Selectați **WSDL Local** sau **Fișier XSD**.
4. Modificați locațiile schemei pentru oasis-wss-wssecurity-secext-1.1.xsd:
 - a. Deschideți schema în *locație spațiu de lucru/nume_modul/StandardImportFilesGen/oasis-wss-wssecurity-secext-1.1.xsd*.
 - b. Modificați:

```
<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. Modificați:

```
<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. Modificați locația schemei pentru oasis-200401-wss-wssecurity-secext-1.0.xsd:
 - a. Deschideți schema în *locație spațiu de lucru/nume_modul/StandardImportFilesGen/oasis-200401-wss-wssecurity-secext-1.0.xsd*.
 - b. Modificați:

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

în:

```
<xsd:import namespace="http://www.w3.org/2000/09/xmlsig#"  
schemaLocation="../w3/tr/_2002/rec_xmlsig_core_20020212/xmlsig-core-schema.xsd" />
```
6. Curățați și reconstruiți proiectul.

Propagarea antetului Transport: **Advanced**

În momentul în care manipulați mesaje SOAP, este posibil să aveți nevoie să accesați informații din anumite anteturi de transport din mesaje ce sunt primite, asigurați-vă că mesajele cu anteturi de transport sunt trimise cu anumite valori, sau permiteți anteturilor de transport să parcurgă un modul.

Când configurați legarea unui serviciu web în Integration Designer, puteți indica faptul că doriți ca anteturile transport să fie propagate.

- Atunci când cererile sunt primite într-un export sau răspunsurile sunt primite la un import, informațiile din antetul de transport pot fi accesate, permițând ca logica din modul să se bazeze pe valorile antetului și permițând 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, realizați pașii următori:

1. Din vizualizarea Proprietăți Integration Designer, selectați **Legare > Propagare**.

2. Setați opțiunea de propagare a antetului de transport de care aveți nevoie.

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

Dacă activați propagarea anteturilor de transport, anteturile din mesajele primite vor fi propagate printr-un modul, iar dacă nu le eliminați în mod explicit aceste vor fi utilizate în invocațiile ulterioare prin același fir de execuție.

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

Accesarea informațiilor 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 object). Puteți seta anteturile cu valori diferite sau puteți crea altele noi. Cu toate acestea, rețineți că nu există nicio verificare sau validare a valorilor setate de dvs., și orice antet necorespunzător sau incorect poate cauza probleme în timpul execuției serviciului web.

Luați în considerare următoarele informații cu privire la setarea anteturilor HTTP:

- Orice modificare adusă anteturilor rezervate pentru motorul de servicii web nu vor fi onorate în mesajul de ieșire. De exemplu, versiunea sau metoda HTTP, anteturile Content-Type, Content-Length și SOAPAction sunt rezervate pentru motorul serviciului web.
- În cazul în care valoarea antetului este un număr, numărul (mai degrabă decât șirul) ar trebui stabilit în mod direct. De exemplu, utilizați **Max-Forwards = 5** (în loc de **Max-Forwards = Max-Forwards: 5**) și **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 și setează în loc antetul Content-Length pentru dimensiune fixată a mesajului.
- Conținutul 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 sfârșit valorile rezervate de motorul serviciilor web la setările clientului:
 - User-Agent
 - Cache-Control
 - Pragma
 - Accept
 - Connection

Puteți accesa informațiile din antet într-unul din următoarele moduri:

- Utilizarea unei primitive de mediere pentru accesarea structurilor de tip SMO
Vedeți legăturile “Informații înrudite” pentru a găsi informații despre utilizarea primitivelor de mediere.
- Utilizarea serviciului de context SPI

Următorul exemplu de cod citește 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 întâlniți probleme în momentul în care trimiteți anteturi revizuite, puteți intercepta mesajul TCP/IP folosind unelte precum TCP/IP Monitor din Integration Designer. Puteți accesa TCP/IP Monitor selectând **Rulare/Depanare > TCP/IP Monitor** în pagina Preferințe.

De asemenea puteți 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): **Advanced**

Atunci când utilizați legări de servicii web (JAX-WS) cu aplicațiile dvs., aveți posibilitatea să adăugați o calitate SAML (Security Assertion Markup Language) a serviciului (QOS) la legare. În primul rând trebuie să utilizați consola administrativă pentru a importa setul de politici. De asemenea, puteți 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: **Advanced**

Security Assertion Markup Language (SAML) este un standard OASIS bazat pe XML pentru schimbarea identității și atributelor de securitate ale informației. Atunci când configurați un serviciu web (JAX-WS) legat cu Integration Designer, puteți specifica un set de politici SAML. Se folosește 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 a profilului:

profile_root/config/templates/PolicySets

Înainte de a începe această procedură, se verifică dacă următoarele directoare (ce conțin seturile de politici) sunt localizate în directorul de configurare a 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 a profilului, copiați-le în acel director din următoarea locație:

app_server_root/profileTemplates/default/documents/config/templates/PolicySets

Importați seturile de politici în consola administrativă, selectați una pe care doriți să o faceți disponibilă pentru Integration Designer și apoi salvați un fișier .zip pentru fiecare dintre aceste seturi de politici într-o locație care este accesibilă prin Integration Designer.

1. Se va importa seturile de politici prin urmărirea pașilor următori:

- a. Din consola administrativă, se face clic pe **Servicii > Seturi de politici > Seturi de politici aplicație**.
 - b. Se face clic pe **Import > Din magazia implicită**.
 - c. Se selectează seturile de politici implicite SAML și 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 aplicație, se selectează setul de politici SAML care se vrea a fi exportat și se face clic pe **Export**.

Notă: Dacă pagina Seturi de politici aplicație nu este momentan afișată, se face clic pe **Servicii > Seturi de politici > Seturi de politici aplicație** din consola administrativă.

- b. Pe pagina următoare, se face clic pe legătura la fișierului .zip pentru setul de politici.
- c. În fereastra Descărcare fișier, se face clic pe **Salvare** și se indică o locație ce este accesibilă prin Integration Designer.
- d. Se face clic pe **Înapoi**.
- e. Finalizați pașii de la 2a la 2d pentru fiecare set de politici care se vrea a fi exportat.

Seturile de politici SAML sunt salvate în fișierul .zip și sunt gata să fie importate în Integration Designer.

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

Invocarea serviciilor web ce necesită autentificare de bază HTTP: 

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

Setați autentificarea de bază HTTP pentru recepționarea cererilor de servicii web prin configurarea Java API pentru XML Web Services (JAX-WS) legări de export, așa cum este descris în Crearea și asignarea rolurilor de securitate pentru exporturi 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 modalități:

- La configurarea legării de import într-un modul SCA, puteți 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, puteți utiliza capabilități de mediere a fluxului pentru a crea dinamic un nou antet de autentificate HTTP și specifica informații cu numele și parola de utilizator în antet.

Notă: Setul de politică are precedență peste valoarea specificată în antet. Dacă doriți să utilizați setul de valori în antetul de autentificare HTTP la momentul rulării, nu potriviți un set de politică ce include politica HTTPTransport. Specific, nu utilizați setul de politică BPMHTTPBasicAuthentication și dacă aveți un set de politică definit, asigurați-vă că include politica HTTPTransport.

Pentru informații suplimentare despre seturile de politici pentru servicii web și legări de politici și cum sunt utilizate acestea, vedeți Seturi de politici de servicii web din WebSphere Application Server Information Center.

- Pentru a utiliza setul de politică furnizat, realizați următorii pași:
 1. Opțional: În consola administrativă, creați o legare de politică generală sau editați una existentă care include politica HTTPTransport cu valorile ID-ului de utilizator și parolei necesare.
 2. În IBM Integration Designer, generați o legare de import a serviciului web (JAX-WS) și atașați setul de politică BPMHTTPBasicAuthentication.
 3. Realizați *unul* din următorii pași:
 - În IBM Integration Designer, în proprietățile de legare de import al serviciului web (JAX-WS), specificați numele unei legări a politicii existente de client generale ce include politica HTTPTransport.

- După implementarea modului SCA, utilizați consola administrativă fie pentru a selecta o legare de politică pentru un client existent, fie creați o nouă legare de politică de client și apoi o asociați cu legarea de import.
- 4. Opțional: În consola administrativă a serverului Process editați legarea setului de politici selectat pentru a specifica ID-ul și parola necesare.
- Pentru a specifica numele și parola de utilizator în antetul de autentificare HTTP, realizați unul din următorul set de pași:
 - Utilizați medierea simplă HTTP Header Setter în IBM Integration Designer pentru a crea antetul de autentificare HTTP și specificați numele și parola de utilizator.
 - Dacă o logică suplimentară este necesară, utilizați codul Java într-o mediere simplă personalizată (asemenea următorului exemplu) pentru a:
 1. Crea un antet de autentificare HTTP.
 2. Specifica informații privind numele și parola de utilizator.
 3. Adăuga un nou antet de autentificare HTTP la HTTPControl.
 4. Setă HTTPControl actualizat înapoi în serviciul Context.

```
//Get the HeaderInfoType from 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();
}
//Create new HTTPAuthentication and set the 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);
// Set header info back to the current execution context.
contextService.setHeaderInfo(headers);
```

Verificarea configurației serverului: Advanced

Atunci când implementați o aplicație cu legarea unui serviciu web (JAX-WS), trebuie să vă asigurați că serverul pe care este implementată aplicația nu are selectată opțiunea **Pornire componente după cum este necesar**.

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

1. Se face clic pe **Servere > Tipuri de servere > Servere de aplicații WebSphere**.
2. Se face clic pe nume server.
3. Din fișa Configurație, se determină dacă este selectat **Pornire componente după cum este necesar**.
4. Se vor realiza următorii pași:
 - Dacă **Pornire componente după cum este necesar** este selectat, se va șterge caseta de bifare și apoi se va face clic pe **Aplicare**.
 - Dacă **Pornire componente după cum este necesar** nu este selectat, se va faceți clic pe **Anulare**.

Atașamente în mesajele SOAP: **Advanced**

Aveți posibilitatea să trimiteți și să primiți mesaje SOAP care includ date binare (precum fișiere PDF sau imagini JPEG) sub formă de atașamente. Atașamentele pot fi cu *referință* (adică sunt reprezentate în mod explicit ca părți componente ale mesajului în interfața serviciului) sau *fără referință* (în care pot fi incluse numere arbitrare și tipuri de atașamente).

Un atașament către care se face referință poate fi reprezentat în unul dintre următoarele moduri:

- Atașamentele MTOM folosesc codarea specifică SOAP Message Transmission Optimization Mechanism (<http://www.w3.org/TR/soap12-mtom/>). Atașamentele MTOM sunt activate prin opțiuni de configurare în legările de import și de export și sunt modul recomandat de încodare a atașamentelor pentru noi aplicații.

- Sub forma unui element de tip wsi:swaRef-typed în schema mesajului

Atașamentele 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 definește modul în care elementele mesajului sunt asociate cu părțile MIME.

- Ca o parte de nivel înalt a mesajului, folosind un tip de schemă binar

Atașamentele reprezentate sub formă de părți de nivel înalt ale mesajului în concordanță cu specificațiile *SOAP Messages with Attachments* (<http://www.w3.org/TR/SOAP-attachments>).

Atașamentele reprezentate sub formă de părți de nivel înalt pot fi de asemenea configurate pentru a vă asigura că mesajele și documentele WSDL produse de legare sunt conforme cu *WS-I Attachments Profile Version 1.0* și *WS-I Basic Profile Version 1.1* (<http://www.ws-i.org/Profiles/BasicProfile-1.1.html>).

Un atașament către care nu se face referire este purtat într-un mesaj SOAP fără nicio reprezentare în schema mesajului.

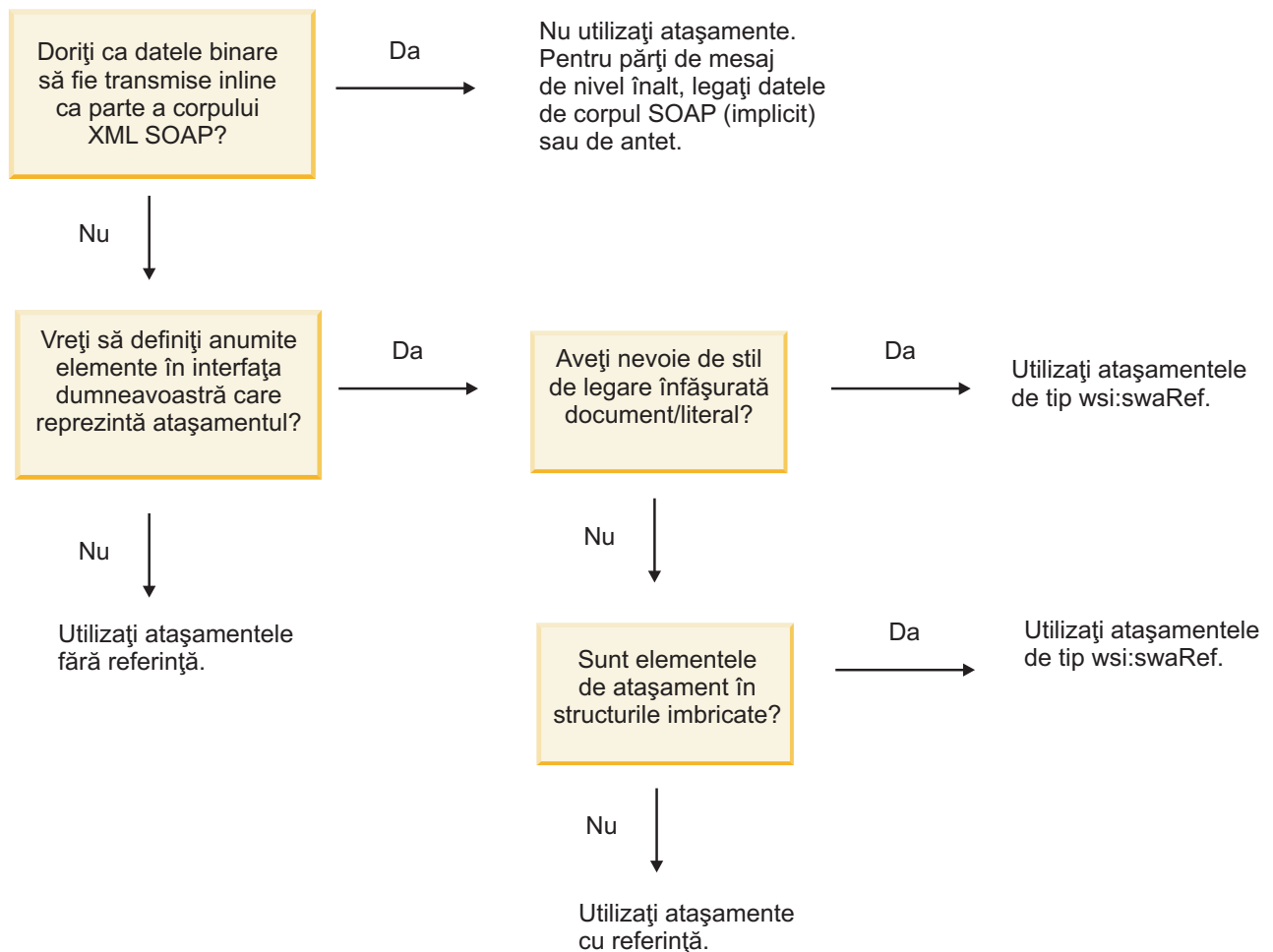
În toate cazurile, cu excepția atașamentelor MTOM, legarea WDSL SOAP trebuie să includă o legare MIME pentru atașamente de folosit, iar dimensiunea maximă a atașamentelor nu trebuie să depășească 20 MB.

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

Cum alegeți stilul corespunzător de atașamente: **Advanced**

Când proiectați o interfață nouă pentru serviciu care include date binare, luați în considerare modul în care datele binare sunt transportate în mesajele SOAP care sunt trimise și primite de către serviciu.

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



Atașamente MTOM: părți de mesaj la nivel de top:

Puteți trimite și primi mesaje de servicii web care includ atașamente MTOM (Message Transmission Optimization Mechanism - Mecanism de optimizare a transmisiei mesajului) SOAP. Într-un mesaj SOAP cu mai multe părți MIME, corpul SOAP este prima parte a mesajului, iar atașamentul sau atașamentele sunt în părțile ulterioare.

Trimițând sau primind un atașament cu referință într-un mesaj SOAP, datele binare care alcătuiesc atașamentul (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 ținute într-un element XML.

Iată un eșantion 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:0FE43E4D025F0BF3DC11582467646812" />
          </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--

```

Rețineți că în eșantionul MTOM, tipul de conținut pentru anvelopa SOAP este **application/xop+xml** și 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@apache.org" />
```

Procesarea la intrare a atașamentelor cu referință

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

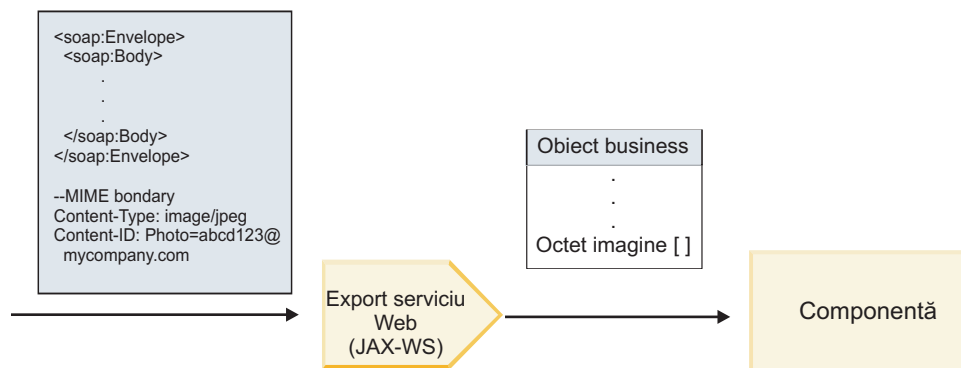


Figura 13. Cum procesează legarea export (JAX-WS) a serviciului web un mesaj SOAP cu un atașament referențiat

Atribute atașament MTOM

- MTOM poate suporta elemente atașament în structuri imbricate.
- MTOM este disponibil numai pentru tipul base64Binary.
- MTOM poate suporta elemente atașament în structuri imbricate, ceea ce înseamnă că **bodyPath** pentru atașamentele MTOM este locația **xpath** pentru elementul unde este ținut atașamentul MTOM. Logica de calcul pentru **bodyPath** urmează strict schema de generare a locației **xpath** așa 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 atașament amestecate nu sunt suportate, ceea ce înseamnă că, dacă MTOM este activat pe legarea de import, se va genera atașamentul 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.

Atașamente către care se face referință: atașamente de tip `swaRef`: **Advanced**

Aveți posibilitatea să trimiteți și să primiți mesaje SOAP care includ atașamente care sunt reprezentate în interfața serviciului sub formă de elemente de tip `swaRef`.

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

În mesajul SOAP, corpul conține un element de tip swaRef care identifică ID-ul conținutului atașamentului.

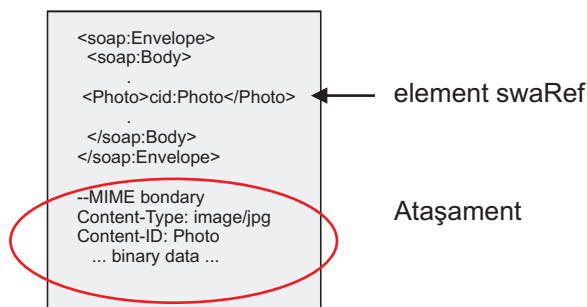


Figura 14. Un mesaj SOAP cu un element swaRef

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

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

WSDL-ul ar trebui să conțină de asemenea o legare MIME care să indice mesajele MIME compuse din mai multe părți 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 părților componente de nivel înalt ale mesajului.

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

Procesarea la intrare a atașamentelor

Folosiți Integration Designer pentru a configura o legare de export, astfel încât să recepționeze atașamentul. Creați un modul și interfața și operațiile sale asociate, incluzând un element de tip swaRef. Puteți crea apoi o legare (JAX-WS) pentru serviciul web.

Notă: Vedeți subiectul “Lucrul cu atașamente” din Centrul de informare Integration Designer pentru informații mai detaliate.

Atunci când un client transmite un mesaj SOAP cu un atașament swaRef către o componentă SCA (Service Component Architecture), legarea de export (JAX-WS) a serviciului web înlătură mai întâi atașamentul. Apoi transmite partea de tip SOAP a mesajului și creează un obiect business. În sfârșit, legarea setează Id-ul conținutului atașamentului în obiectul business.

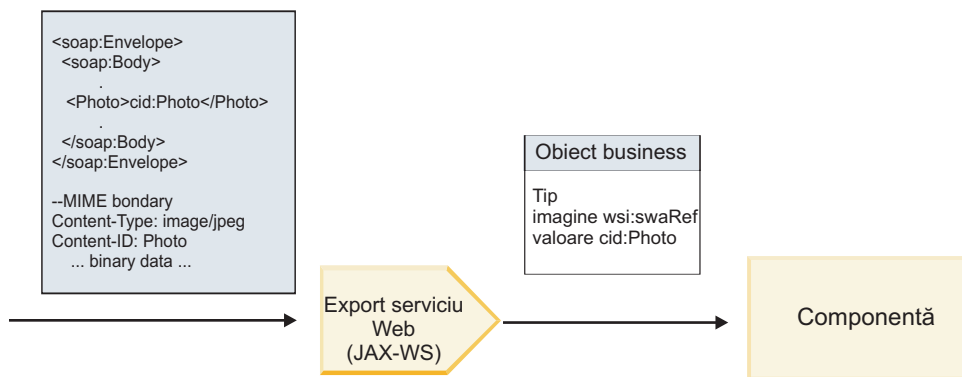


Figura 15. Modul în care o legare externă (JAX-WS) a serviciului web procesează un mesaj SOAP cu un atașament swaRef

Accesarea metadatelor atașamentului într-o componentă a fluxului de mediere

Așa cum se arată în Figura 16, atunci când atașamentele de tip swaRef sunt accesate de componente, identificatorul conținutului atașamentului apare sub forma unui element de tip swaRef.

Fiecare atașament 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 conținutului atașamentului și ID-ul conținutului, precum și datele binare reale ale atașamentului.

Pentru a putea obține valoarea unui atașament de tip swaRef, este necesară obținerea valorii elementului de tip swaRef, iar apoi localizarea elementului **attachments** cu valoarea **contentID** corespunzătoare. Rețineți că valoarea **contentID** are de obicei prefixul **cid:** înlăturat din valoarea swaRef.

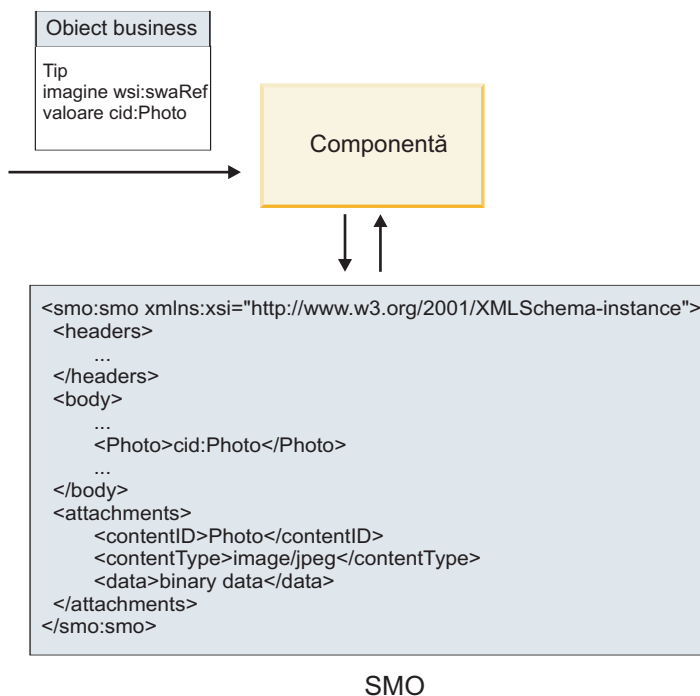


Figura 16. Modul în care atașamentele swaRef apar în SMO

Procesarea la ieșire

Utilizați 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 și definește atașamentul care va fi trimis către serviciul web.

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

Pentru procesarea la ieșire, elementele de tip swaRef sunt trimise întotdeauna împreună cu valorile lor din ID-ul conținutului; totuși, modulul de mediere trebuie să se asigure că există un element **attachments** corespunzător valorii **contentID**.

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

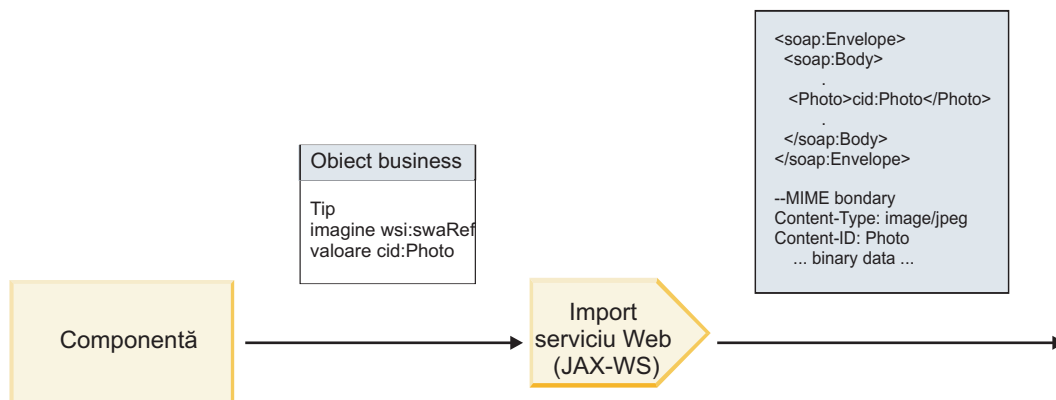


Figura 17. Modul în care o legare de import (JAX-WS) a serviciului web generează un mesaj SOAP cu un atașament swaRef

Setarea metadatelor atașamentului într-o componentă a fluxului de mediere

În cazul în care, în SMO există o valoare pentru un element de tip swaRef și un element **attachments**, legarea pregătește mesajul SOAP (cu atașament) și îl trimite către un destinatar.

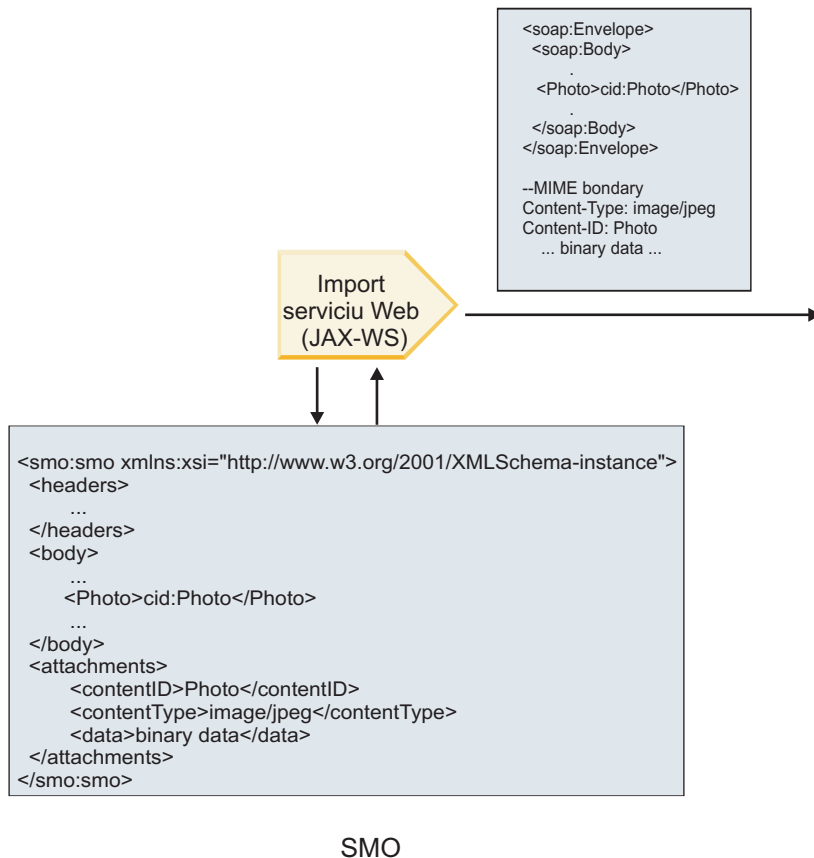


Figura 18. Modul în care un atașament de tip swaRef din SMO este accesat pentru a crea un mesaj de tip SOAP

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 conține alte tipuri de componente, ar trebui să fie utilizată o componentă a fluxului de mediere pentru a copia valorile într-o locație în care pot fi accesate din modul, și o altă componentă a fluxului de mediere utilizată pentru a seta valorile corecte înainte de o invocare de ieșire prin intermediul unui serviciu web de import.

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

Atunci când trimiteți mesaje SOAP cu atașamente, elementul rădăcină pe care îl alegeți determină modul în care atașamentele sunt propagate.

- Dacă folosiți “/body” drept rădăcină a mapării XML, toate atașamentele sunt propagate implicit de-a lungul mapării.
- Dacă folosiți “/” drept rădăcină pentru mapare, puteți controla propagarea atașamentelor.

Atașamente către care se face referință: părți componente de nivel înalt ale mesajului: **Advanced**

Aveți posibilitatea să trimiteți și să primiți mesaje SOAP care includ atașamente binare ce sunt declarate ca părți componente în interfața serviciului dumneavoastră.

Într-un mesaj SOAP cu mai multe părți MIME, corpul SOAP este prima parte a mesajului, iar atașamentul sau atașamentele se află în părțile următoare.

Care este avantajul de a trimite sau de a primi un atașament cu referință într-un mesaj SOAP? Datele binare care alcătuiesc atașamentul (care este adesea destul de mare), sunt ținute 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 ținute într-un element XML.

Tipuri de mesaje SOAP cu atașamente către care se face referință

Începând cu versiunea 7.0.0.3 a IBM Business Process Manager, aveți 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* și 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 atașamente, se utilizează codarea părții legate de id-ul conținutului (așa cum este descris în *WS-I Attachments Profile Version 1.0*) pentru a raporta atașamentul de partea componentă a mesajului.

- **Mesaje care nu sunt conforme cu WS-I**

Runtime-ul poate genera mesaje SOAP care nu sunt conforme cu profilurile WS-I, dar care sunt compatibile 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 reține atașamentul **content-id**, dar codarea părții content-id nu este folosită (așa cum este descrisă în *WS-I Attachments Profile Version 1.0*).

Selectarea compatibilității WS-I pentru exporturile de servicii web

Folosiți Integration Designer pentru a configura o legare de export. Creați un modul și interfața și operațiile sale asociate. Puteți crea apoi o legare (JAX-WS) pentru serviciul web. Pagina Atașamente cu referință afișează toate părțile binare care aparțin de operațiile create, apoi selectați care părți vor fi atașate. Apoi specificați, 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 selectați această opțiune, specificați de asemenea și care parte a mesajului ar trebui să fie legată de corpul SOAP.

Notă: Această opțiune poate fi utilizată doar atunci când fișierul WSDL corespunzător este de asemenea conform cu WS-I.

Un fișier WSDL care este generat de Integration Designer Versiunea 7.0.0.3 este în conformitate cu WS-I. Totuși, dacă importați un fișier WSDL care nu este în conformitate cu WS-I, nu puteți selecta această opțiune.

- **Utilizare mesaje SOAP care nu sunt conforme cu WS-I AP 1.0**

Dacă selectați această opțiune, care este implicită, prima parte a mesajului este legată de corpul SOAP.

Notă: Doar părțile de nivel înalt ale mesajului (adică, elementele definite în portType WSDL drept părți din cadrul mesajului de intrare sau de ieșire) care au un tip binar (fie base64Binary, fie hexBinary) pot fi trimise sau primite sub formă de atașamente cu referință.

Vedeți subiectul “Lucrul cu atașamente” 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 **name** din elementul **wsdl:part** către care se face referință prin **mime:content**
- Caracterul **=**
- O valoare unică globală, precum un UUID
- Caracterul **@**
- Un nume de domeniu valid

Procesarea la intrare a atașamentelor cu referință

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

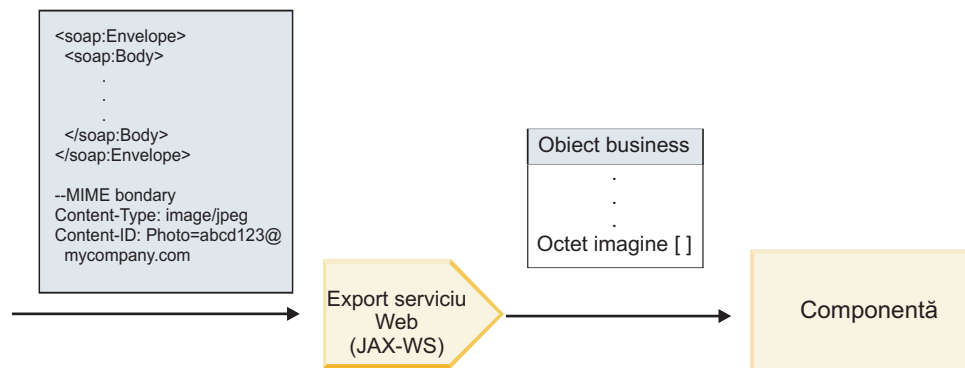


Figura 19. Modul în care o legare externă (JAX-WS) a serviciului web procesează un mesaj SOAP conform WS-I cu un atașament către care se face o referință

Accesarea metadatelor atașamentului într-o componentă a fluxului de mediere

Așa cum se arată în Figura 19, atunci când atașamentele cu referință sunt accesate de componente, datele atașamentului apar sub forma unei matrice cu date de tip octet.

Fiecare atașament cu referință al unui mesaj SOAP are, de asemenea, un element **attachments** corespunzător în SMO. Elementul **attachments** include tipul conținutului atașamentului și calea către corpul mesajului în care este păstrat atașamentul.

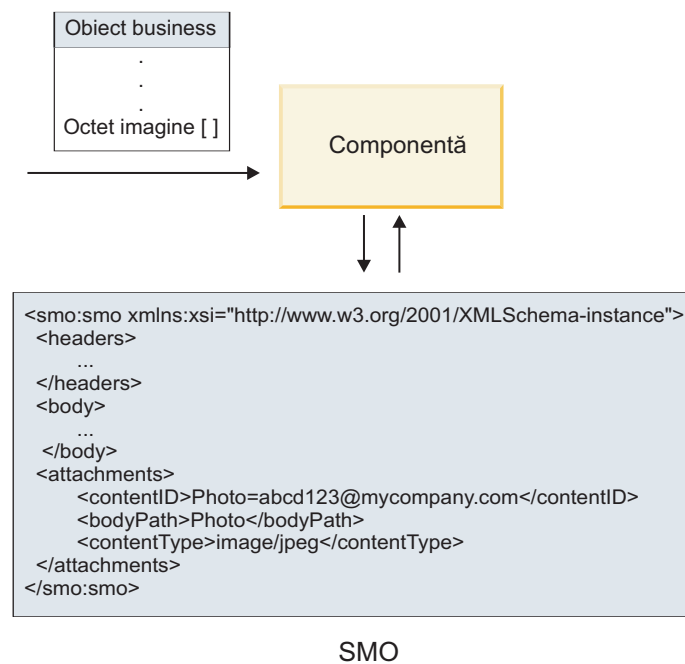


Figura 20. Modul în care atașamentele către care se face referință apar în SMO

Important: Calea către elementul corp al mesajului nu este actualizată automat în cazul în care mesajul este transformat și atașamentul mutat. Puteți 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 ieșire

Utilizați 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 și definește ce părți din mesaj ar trebui să fie trimise ca atașamente. De asemenea, puteți indica, în pagina Specificare compatibilității cu WS-I AP 1.0 a Integration Designer, una dintre următoarele alegeri:

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

Dacă selectați această opțiune, specificați de asemenea și care parte a mesajului ar trebui să fie legată de corpul SOAP; toate celelalte sunt legate de atașamente sau anteturi. Mesajele trimise de legare nu includ elemente în corpul SOAP care se referă la atașamente; relația este exprimată prin intermediul ID-ului de conținut al atașamentului, inclusiv numele părții mesajului.

- **Utilizare mesaje SOAP care nu sunt conforme cu WS-I AP 1.0**

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

Notă: Partea care reprezintă un atașament, așa 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 atașament.

Procesarea la intrare a atașamentelor cu referință

Legarea de import folosește informațiile din SMO pentru a determina modul în care părțile componente de nivel înalt ale mesajului sunt trimise ca atașamente.

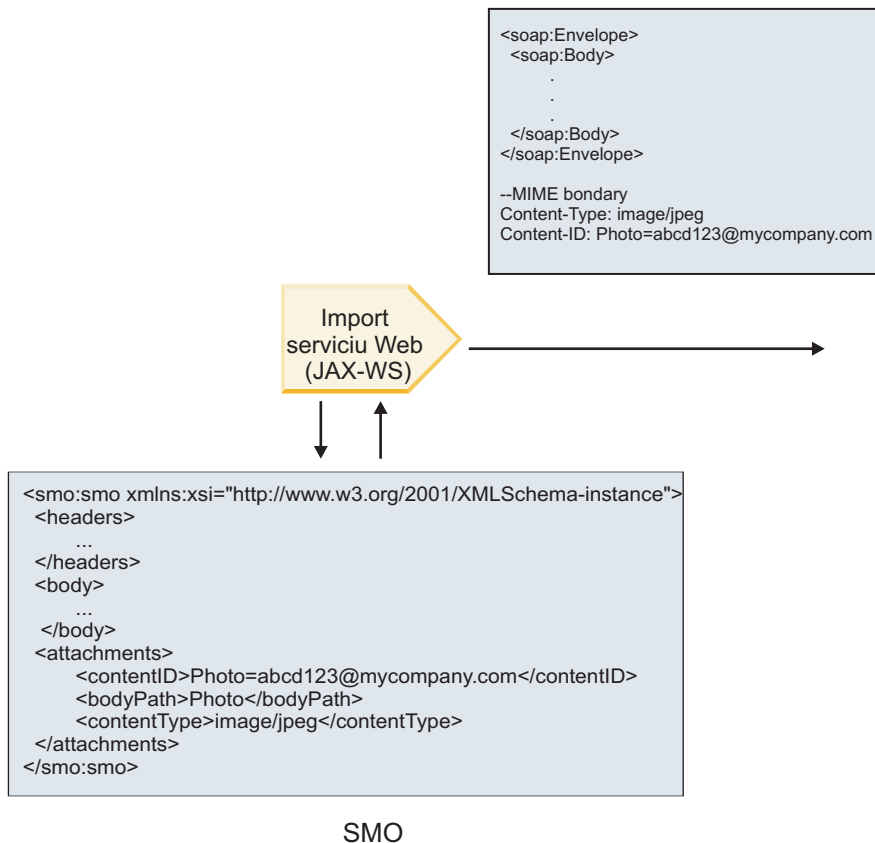


Figura 21. Modul în care atașamentul cu referință din SMO este accesat pentru a crea mesajul SOAP

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 conține alte tipuri de componente, ar trebui să fie utilizată o componentă a fluxului de mediere pentru a copia valorile într-o locație în care pot fi accesate din modul, și o altă componentă a fluxului de mediere utilizată pentru a seta valorile corecte înainte de o invocare de ieșire prin intermediul unui serviciu web de import.

Legarea folosește o combinație între următoarele condiții 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 conținutului
- În cazul în care există un element **attachments** în SMO a cărui valoare **bodyPath** face referință către o parte binară de nivel înalt

Modul în care sunt create atașamentele atunci când există niciun element attachment în SMO

Tabelul următor arată modul în care este creat și trimis un atașament în cazul în care SMO conține un element **attachment** cu un **bodyPath** care se potrivește cu o parte a numelui mesajului:

Tabela 32. Modul în care este generat atașamentul

Starea legării WSDL MIME pentru partea componentă binară de nivel înalt a mesajului	Modul în care mesajul este creat și transmis
Prezent cu una dintre următoarele: <ul style="list-style-type: none"> Nu este definit un tip de conținut pentru partea componentă a mesajului Sunt definite mai multe tipuri de conținut Este definit tipul conținutului metacaracterului 	<p>Partea componentă a mesajului este trimisă ca un atașament.</p> <p>Content-Id este setat cu o valoare în elementul attachments, în cazul în care acesta este prezent; altfel, se generează una.</p> <p>Content-Type este setat cu o valoare în elementul attachments în cazul în care acesta este prezent; altfel, este setat cu application/octet-stream.</p>
Prezent cu conținut unic, fără metacaractere pentru partea componentă a mesajului	<p>Partea componentă a mesajului este trimisă ca un atașament.</p> <p>Content-Id este setat cu o valoare în elementul attachments, în cazul în care acesta este prezent; altfel, se generează una.</p> <p>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 conținut WSDL MIME.</p>
Nu este prezent	<p>Partea componentă a mesajului este trimisă ca un atașament.</p> <p>Content-Id este setat cu o valoare în elementul attachments, în cazul în care acesta este prezent; altfel, se generează una.</p> <p>Content-Type este setat cu o valoare în elementul attachments în cazul în care acesta este prezent; altfel, este setat cu application/octet-stream.</p> <p>Notă: Trimițând părți componente ale sub formă de atașamente 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.</p>

Modul în care sunt create atașamentele atunci când nu există niciun element attachment în SMO

Tabelul următor arată modul în care este creat și trimis n atașament în cazul în care SMO nu conține un element **attachment** cu un **bodyPath** care se potrivește cu o parte din numele mesajului:

Tabela 33. Modul în care este generat atașamentul

Starea legării WSDL MIME pentru partea componentă binară de nivel înalt a mesajului	Modul în care mesajul este creat și transmis
Prezent cu una dintre următoarele: <ul style="list-style-type: none"> Nu este definit un tip de conținut pentru partea componentă a mesajului Sunt definite mai multe tipuri de conținut Este definit tipul conținutului metacaracterului 	<p>Partea componentă a mesajului este trimisă ca un atașament.</p> <p>Content-Id este generat.</p> <p>Content-Type este setat cu application/octet-stream.</p>
Prezent cu conținut unic, fără metacaractere pentru partea componentă a mesajului	<p>Partea componentă a mesajului este trimisă ca un atașament.</p> <p>Content-Id este generat.</p> <p>Content-Type este setat cu tipul definit în elementul conținut WSDL MIME.</p>
Nu este prezent	Partea componentă a mesajului nu este trimisă ca un atașament.

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

Atunci când trimiteți mesaje SOAP cu atașamente, elementul rădăcină pe care îl alegeți determină modul în care atașamentele sunt propagate.

- Dacă folosiți “/body” drept rădăcină a mapării XML, toate atașamentele sunt propagate implicit de-a lungul mapării.
- Dacă folosiți “/” drept rădăcină pentru mapare, puteți controla propagarea atașamentelor.

Atașamente fără referință: **Advanced**

Aveți posibilitatea să trimiteți și să primiți atașamente *fără referință* care nu sunt declarate în interfața serviciului.

Într-un mesaj SOAP cu mai multe părți MIME, corpul SOAP este prima parte componentă a mesajului, iar atașamentele se află în părțile componente următoare. Nu este inclusă nicio referință către atașament în corpul SOAP.

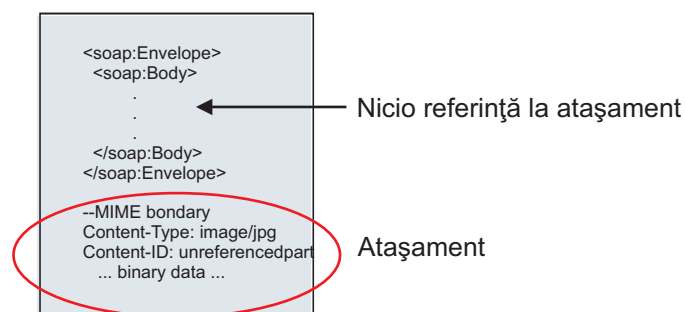


Figura 22. Un mesaj SOAP cu un atașament către care nu se face referință

Aveți posibilitatea să trimiteți un mesaj SOAP cu un atașament fără referință printr-un export de serviciu web către importul unui serviciu web. Mesajul de ieșire, care este trimis la serviciul web țintă, conține atașamentul.

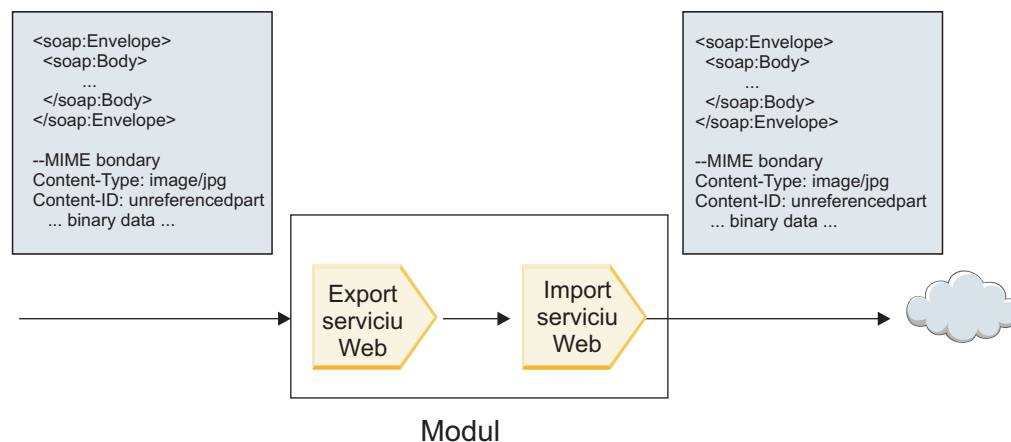


Figura 23. Un atașament care trece printr-un modul SCA

În Figura 23, mesajul SOAP, cu atașament, este transmis fără modificări.

De asemenea, puteți modifica mesajul SOAP prin utilizarea unei componente de tip flux de mediere. De exemplu, aveți posibilitatea să utilizați componenta de tip flux de mediere pentru a extrage datele din mesajul SOAP (în acest caz, date binare din corpul mesajului) și pentru a crea un mesaj SOAP cu atașamente. Datele sunt prelucrate ca parte componentă

a elementului ce conține atașamentele dintr-un SMO (service message object).

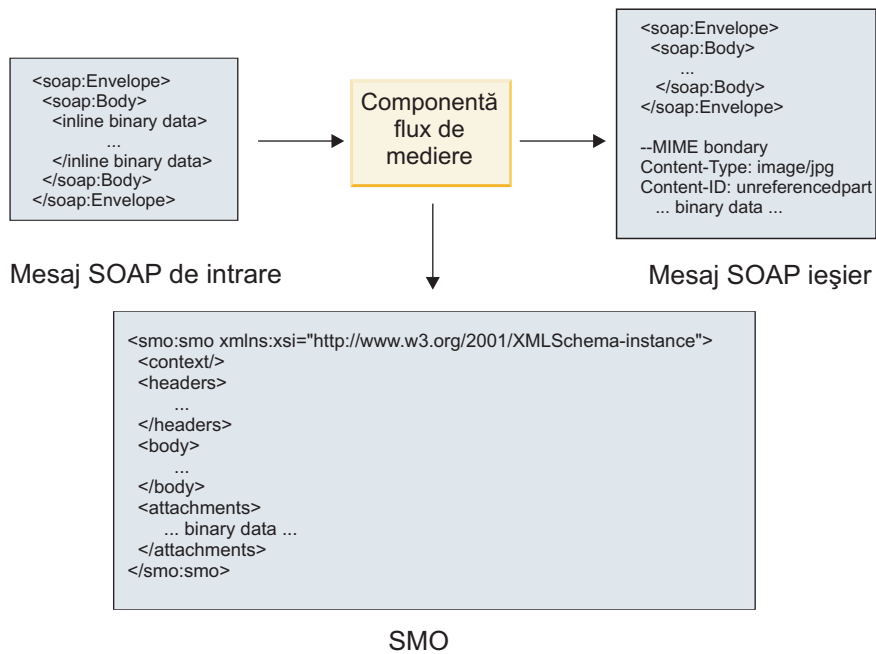


Figura 24. Un mesaj procesat de o componentă a fluxului de mediere

În schimb, componenta de flux de mediere se poate transforma mesajul de intrare prin extragerea și codificarea atașamentului, iar apoi transmite mesajul fără niciun atașament.

În loc de extragerea datelor dintr-un mesaj SOAP de intrare pentru a forma un mesaj SOAP cu atașamente, puteți obține datele atașamentului la o sursă externă, cum ar fi o bază de date.

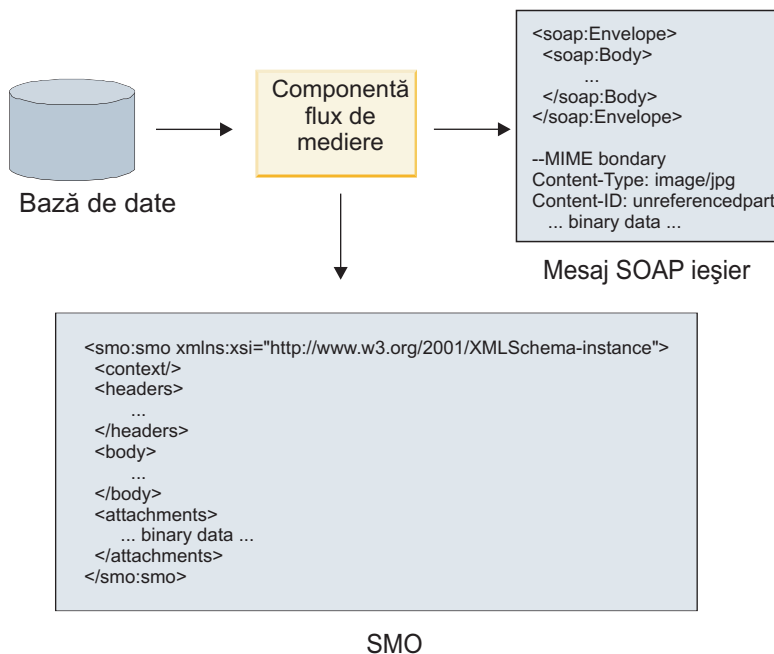


Figura 25. Un atașament obținut dintr-o bază de date și adăugat la mesajul SOAP

În schimb, componenta fluxului de mediere poate extrage atașamentul dintr-un mesaj SOAP de intrare și poate procesa mesajul (de exemplu, păstrează atașamentul într-o bază de date).

Atașamentele fără referință pot fi transmise doar peste componentele fluxului de mediere. În cazul în care un atașament trebuie să fie accesat sau transmis către o componentă de un alt tip, folosiți o componentă a fluxului de mediere pentru a muta atașamentul la o locație care este accesibilă acelei componente.

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

Atunci când trimiteți mesaje SOAP cu atașamente, elementul rădăcină pe care îl alegeți determină modul în care atașamentele sunt propagate.

- Dacă folosiți `"/body"` drept rădăcină a mapării XML, toate atașamentele sunt propagate implicit de-a lungul mapării.
- Dacă folosiți `"/"` drept rădăcină pentru mapare, puteți controla propagarea atașamentelor.

Utilizarea legării stilului documentului WSDL cu mesaje multiple: Advanced

Organizația 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 și modul în care mesajele SOAP corespunzătoare ar trebui să fie formate, în scopul asigurării interoperabilității.

Aceste reguli sunt specificate în *WS-I Basic Profile Version 1.1* (<http://www.ws-i.org/Profiles/BasicProfile-1.1.html>).

În particular, WS-I Basic Profile 1.1 R2712 stabilește: "O legare document-literal TREBUIE serializată ca o ENVELOPE cu un soap:Body a cărui element copil este o instanță a declarației de element global referit de partea wsdl:message corespondentă."

Aceasta înseamnă că, atunci când utilizați o legare SOAP de stil document pentru o operație cu mesaje (intrare, ieșire sau greșală) care sunt definite cu părți componente multiple, doar una dintre acele părți ar trebui să fie legată de corpul SOAP pentru a fi compatibilă cu WS-I Basic Profile 1.1.

În continuare, WS-I Attachments Profile 1.0 R2941 stabilește: "wsdl:binding într-o DESCRIPTION TREBUIE să lege fiecare wsdl:part a unui wsdl:message în wsdl:portType la care se referă din soapbind:body, soapbind:header, soapbind:fault, soapbind:headerfault sau mime:content."

Aceasta înseamnă că, atunci când utilizați o legare SOAP de stil document pentru o operație cu mesaje (intrare, ieșire sau greșală) care sunt definite cu părți componente multiple, toate părțile componente în afară de cea selectată care vor fi legate la corpul SOAP trebuie să fie legate ca atașamente sau anteturi.

Următoarea abordare este utilizată atunci când descrierile WSDL sunt generate pentru exporturi cu legări de servicii web (JAX-WS și JAX-RPC) în acest caz:

- Puteți 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 părți componente ale mesajului de tip "hexBinary" sau "base64Binary" sunt legate ca atașamente referite. Vedeți "Atașamente către care se face referință: părți componente de nivel înalt ale mesajului" la pagina 82.
- Toate celelalte părți componente ale mesajului sunt legate ca anteturi SOAP.

Legările de import JAX-RPC și JAX-WS respectă legarea SOAP într-un document WSDL existent cu mesaje compuse de tip document chiar dacă nu leagă mai multe părți la corpul SOAP; totuși, nu sunteți în măsură să generați clienți pentru serviciile web pentru astfel de documente WSDL în Rational Application Developer.

Notă: Legarea JAX-RPC nu suportă atașamente.

Tiparul recomandat când utilizați mesaje multiple cu o operație care are legare SOAP de stil document este prin urmare:

1. Utilizați stilul înfășurat document/literal. În acest caz, mesajele au mereu o singură parte componentă; totuși, atașamentele trebuie să fie nereferite (după cum este descris în "Atașamente fără referință" la pagina 88) sau swaRef-typed (după cum este descris în "Atașamente către care se face referință: atașamente de tip swaRef" la pagina 78) în acest caz.
2. Utilizați stilul RPC/literal. În acest caz, nu există restricții asupra legării WSDL în ceea ce privește numărul de părți componente legate de corpul SOAP; mesajul SOAP care rezultă are întotdeauna un singur copil care reprezintă operația care este invocată, cu părțile componente ale mesajului fiind copiii acelui element.
3. Pentru legarea JAX-WS, trebuie să aveți cel mult o parte componentă mesaj care nu este de tip "hexBinary" sau "base64Binary", doar dacă nu este acceptabil să legați celelalte părți componente non-binare la anteturi SOAP.
4. Orice alte cazuri sunt supuse comportamentului descris.

Notă: Există restricții suplimentare când utilizați 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 recepționați mesaje SOAP stil document multiple cu atașamente referite, legarea JAX-WS așteaptă ca fiecare atașament referit să fie reprezentat de un element copil al corpului SOAP cu o valoare atribut href care identifică atașamentul după ID-ul conținutului său. Legarea JAX-WS trimite atașamente referite pentru astfel de mesaje în același mod. Acest comportament nu este compatibil cu *WS-I Basic Profile*.

Pentru a vă asigura că mesajele dumneavoastră sunt compatibile cu Basic Profile, urmați abordarea 1 sau 2 din lista anterioară sau evitați utilizarea atașamentelor referite pentru astfel de mesaje și utilizați atașamente nereferite sau swaRef-typed în loc.

Legări HTTP:

Legarea HTTP este proiectată să furnizeze conectivitate SCA (Service Component) la HTTP. În consecință, aplicații 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 informațiilor pe web. Când lucrați cu o aplicație externă ce folosește protocolul HTTP, este necesară o legare HTTP. Legarea HTTP transformă datele transmise ca un mesaj într-un format nativ unui obiect business într-o aplicație SCA. Legarea HTTP poate de asemenea transforma datele transmise în afară ca un obiect business în formatul nativ așteptat de aplicația externă.

Notă: Dacă doriți să interacționați cu clienții și serviciile care utilizează protocolul SOAP/HTTP pentru serviciile web, luați în considerare utilizarea uneia dintre legările serviciilor web, care oferă funcționalități suplimentare cu privire la manipulare standardului calității serviciilor pentru serviciile web.

Câteva scenarii comune pentru utilizarea legării HTTP sunt descrise în următoarea listă:

- Serviciile găzduite de SCA pot invoca aplicații HTTP folosind un import HTTP.
- Serviciile găzduite de SCA se pot expune ca aplicații cu HTTP activat, pentru a putea fi folosite de clienți HTTP, folosind un export HTTP.
- IBM Business Process Manager și Process Server pot comunica între ei peste o infrastructură HTTP, în consecință utilizatorii își pot gestiona comunicările conform standardelor de corporație.
- IBM Business Process Manager și Process Server pot acționa ca mediatori de comunicații HTTP, transformând și direcționând mesaje, ce îmbunătățește integrarea aplicațiilor folosind o rețea HTTP.
- IBM Business Process Manager și Process Server pot fi folosite ca o punte între HTTP și alte protocoale, precum servicii Web SOAP/HTTP, adaptoare de resurse bazate pe JCA (Java Connector Architecture), JMS, și așa mai departe.

Informații detaliate despre crearea legărilor de import și export HTTP pot fi găsite în centrul de informare Integration Designer. Vedeți subiectele **Dezvoltarea aplicațiilor de integrare > Accesarea serviciilor externe cu HTTP**.

Privire generală asupra legărilor HTTP:

Legarea HTTP oferă conectivitate la aplicații găzduite-HTTP. Mediază comunicarea dintre aplicații HTTP și permite aplicațiilor bazate pe HTTP existente să fie apelate dintr-un modul.

Legări de import HTTP

Legarea de import HTTP oferă conectivitate de ieșire de la aplicații SCA (Service Component Architecture) la un server sau aplicații 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 țintă SMO.
- URL-ul poate fi specificat ca o proprietate de configurare pe import.

Această invocare este întotdeauna sincronă în natură.

Deși invocările HTTP sunt întotdeauna cerere-răspuns, importul HTTP suportă și operații cu sens unic și cu sens dublu și ignoră răspunsul în cazul unei operații cu sens unic.

Legări de export HTTP

Legarea de export HTTP oferă conectivitate de intrare de la aplicații HTTP la o aplicație SCA.

Un URL este definit pe exportul HTTP. Aplicațiile 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 aplicația SCA către serviciul web extern. Cererea se face din aplicația SCA către serviciul web extern, după cum se arată în Figura 26.

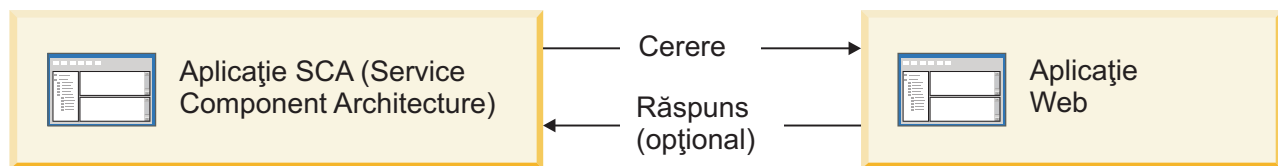


Figura 26. Fluxul unei cereri de la aplicația SCA către aplicația web

Opțional, importul cu legarea HTTP poate primi date înapoi de la aplicația web într-un răspuns la cerere.

Cu un export, cererea se face de către o aplicație client către un serviciu web, după cum se arată în Figura 27.

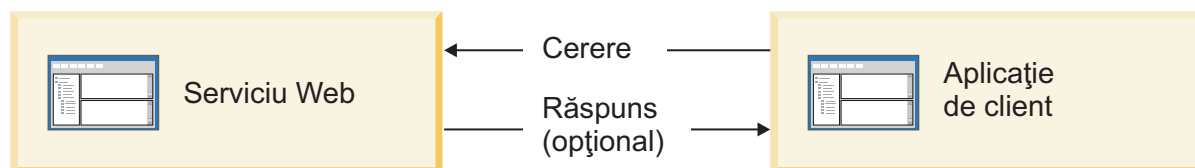


Figura 27. Fluxul unei cereri de la aplicația client la serviciul web.

Serviciul web este o aplicație web care rulează pe server. Exportul este implementat în aplicația web ca un servlet, astfel încât clientul trimite cererea către o adresă URL. Servletul transmite cererea aplicației SCA în timpul rulării.

Opțional, exportul poate trimite date aplicației client ca răspuns la cerere.

Anteturi HTTP:

Legările de import și export HTTP permit ca configurarea anteturilor HTTP și a valorilor lor să fie folosită pentru mesaje de ieșire. Importul HTTP folosește aceste anteturi pentru cereri și exportul HTTP le folosește pentru răspunsuri.

Anteturile configurate static și informațiile de control prevalează asupra valorilor setate dinamic în timpul rulării. Totuși, valorile de control ale URL-ului de înlocuire dinamic, Versiunii și Metodei înlocuiesc valorile statice, ce sunt altfel considerate implicite.

Legarea suportă natura dinamică a URL-ului importului HTTP prin determinarea valorii URL-ului țintă HTTP, Versiunii și Metodei la momentul rulării. Aceste valori sunt determinate prin extragerea valorii Referinței de punct final, URL-ului de înlocuire dinamic, Versiunii și Metodei.

- Pentru Referința punctului final, folosiți API-uri `com.ibm.websphere.sca.addressing.EndpointReference` APIs sau setați câmpul `/headers/SMOHeader/Target/address` în antetul SMO.
- Pentru URL de înlocuire dinamic, Versiune și Metodă, folosiți secțiunea parametrilor de control HTTP a mesajului SCA (Service Component Architecture). Rețineți că URL-ul de înlocuire dinamic are precedență în fața Referinței de punct final țintă; totuși, Referința punctului final se aplică peste legări, deci este abordarea preferată și ar trebui folosită unde este posibil.

Informațiile de control și antet pentru mesaje de ieșire sub legările de export și import HTTP sunt procesate în următoarea ordine:

1. Informații de antet și control excluzând URL de înlocuire dinamic HTTP, Versiune și 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. Adresă țintă specificată prin calea Referinței punctului final sau antetului SMO
5. URL de înlocuire dinamic, Versiune și Metodă din mesajul SCA
6. Informații de antet și control de la handler-ul de date sau legarea de date (cea mai înaltă prioritate)

Exportul și importul HTTP vor popula anteturile de direcție de intrare și parametri de control cu date din mesajul de intrare (`HTTPExportRequest` și `HTTPImportResponse`) doar dacă propagarea antetului de protocol este setată la **True**. Invers, exportul și importul HTTP vor citi și procesa anteturile de ieșire și parametri de control (`HTTPExportResponse` și `HTTPImportRequest`) doar dacă propagarea antetului de protocol este setată la **True**.

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 instrucțiunile de procesare ale mesajului din interiorul legării de import sau export și 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 și contextul de utilizator, precum ID cont) de-a lungul unei căi de invocare SCA. În timpul dezvoltării în IBM Integration Designer, puteți controla propagarea contextului prin proprietățile de import și export. Pentru detalii suplimentare, vedeți informațiile legărilor de import și export din centrul de informare al IBM Integration Designer.

Structuri de antet HTTP și suport furnizate

Tabela 34 detaliază parametri de cerere/răspuns pentru cereri și răspunsuri Import HTTP și Export HTTP.

Tabela 34. Informații 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: Șirul 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, înlocuiește URL-ul de import HTTP. Scris în mesaj în linia de cerere. Notă: Șirul de interogare face de asemenea parte din parametrul de control URL.	Nesetat	Nesetat	Ignorat
Versiune de înlocuire dinamică	Dacă este setat, înlocuiește Versiunea de import HTTP. Scris în mesaj în linia de cerere.	Nesetat	Nesetat	Ignorat
Metodă de înlocuire dinamică	Dacă este setat, înlocuiește 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-conținut.)	Dacă este prezent, scris în mesaj ca parte din antetul Tip-conținut. 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-conținut	Citit din mesajul cerere, antetul Tip-conținut	Dacă este prezent, scris în mesaj ca parte din antetul Tip-conținut. 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-conținut. Notă: Această valoare a elementului de control ar trebui furnizată de legarea de date.	Citit din mesajul răspuns, antetul Tip-conținut	Citit din mesajul cerere, antetul Tip-conținut	Suportat; scris în mesaj ca parte din antetul Tip-conținut. Notă: Această valoare a elementului de control ar trebui furnizată de legarea de date.

Tabela 34. Informații 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; implicitul este identity)	Dacă este prezent, scris în mesaj ca un antet și controlează cum este codată transformarea mesajului.	Citit din mesajul răspuns	Citit din mesajul de cerere	Dacă este prezent, scris în mesaj ca un antet și controlează cum este codată transformarea mesajului.
Codare de conținut (Valori posibile: gzip, x-gzip, deflate, identity; implicitul este identity)	Dacă este prezent, scris în mesaj ca un antet și 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 și controlează cum sunt codate datele utile (payload).
Lungime-Conținut	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 (conține proprietăți 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 și transmisă ca text în clar.	Nu se aplică	Citit din antetul Autentificare de bază al mesajului răspuns. Prezența acestui antet nu indică faptul că utilizatorul a fost autentificat. Autentificarea ar trebui controlată în configurarea servlet-ului. Notă: Valoarea pentru acest antet va fi codată doar pe protocolul HTTP. În SCA, va fi decodată și transmisă ca text în clar.	Nu se aplică
Proxy (conține proprietăți multiple: Gazdă, Port, Autentificare)	Dacă este prezent, folosit pentru a stabili conexiunea prin proxy.	Nu se aplică	Nu se aplică	Nu se aplică
SSL (conține proprietăți multiple: Keystore, Keystore Password, Trustore, Trustore Password, ClientAuth)	Dacă este populat și url-ul destinație 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) și un mesaj de protocol HTTP, trebuie configurate un handler de date sau o legare de date HTTP. Handler-ele de date oferă o interfață neutră-legării ce

permite reutilizarea peste legări de transport și reprezintă abordarea recomandată; legările de date sunt specifice unei anumite legări de transport. Sunt furnizate clase de legări de date specifice-HTTP; puteți 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 și HTTPServiceGatewayDataBinding) sunt depreciate începând cu IBM Business Process Manager Versiunea 7.0. În loc de a folosi legările de date descrise în acest subiect, considerați următoarele handler-e de date:

- Folosiți SOAPDataHandler în loc de HTTPStreamDataBindingSOAP.
- Folosiți UTF8XMLDataHandler în loc de HTTPStreamDataBindingXML
- Folosiți GatewayTextDataHandler în loc de HTTPServiceGatewayDataBinding

Legările de date sunt furnizate pentru utilizare cu importuri HTTP și exporturi HTTP: legare de date binare, legare de date XML și legare de date SOAP. O legare de date răspuns nu este necesară pentru operații cu sens unic. O legare de date este reprezentată de numele unei clase Java ale cărei instanțe pot converti atât de la HTTP la ServiceDataObject și vice-versa. Un selector de funcții trebuie folosit pe un export care, în conjuncție cu legări de metode, poate determina ce legare de date este folosită și ce operație 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 și nu are restricții pe schema interfeței.
- Legări de date SOAP, ce suportă corpul ca date SOAP. Implementarea legării de date SOAP nu are restricții pe schema interfeței.

Implementarea legărilor de date HTTP personalizate

Această secțiune descrie cum se implementează o legare de date HTTP personalizată.

Notă: Abordarea recomandată este să implementați un handler de date personalizat deoarece poate fi reutilizat peste legări de transport.

HTTPStreamDataBinding este principala interfață pentru tratarea mesajelor HTTP personalizate. Interfața este proiectată să permită tratarea datelor utile mari. Totuși, pentru ca astfel de implementări să funcționeze, această legare de date trebuie să returneze informațiile de control și anteturile înainte de a scrie mesajul în flux.

Metodele și ordinea lor de execuție, listate mai jos, trebuie implementate de legarea de date personalizată.

Pentru a personaliza o legare de date, scrieți o clasă ce implementează HTTPStreamDataBinding. Legarea de date ar trebui să aibă patru proprietăți 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 ieșire (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 invocați 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;
}
/*
 * Add http header "IsBusinessException" in pHeaders.
 * Two steps:
 * 1.Remove all the header with name IsBusinessException (case-insensitive) first.
 * This is to make sure only one header is present.
 * 2.Add the new header "IsBusinessException"
 */
public void setBusinessException(boolean isBusinessException) {
    //remove all the header with name IsBusinessException (case-insensitive) first.
    //This is to make sure only one header is present.
    //add the new header "IsBusinessException", code example:
    HTTPHeader header=HeadersFactory.eINSTANCE.createHTTPHeader();
    header.setName("IsBusinessException");
    header.setValue(Boolean.toString(isBusinessException));
    this.pHeaders.getHeader().add(header);
}
public HTTPControl getControlParameters() {
```

```

    return pCtrl;
}
public HTTPHeaders getHeaders() {
    return pHeaders;
}
public DataObject getDataObject() throws DataBindingException {
    return pDataObject;
}
/*
 * Get header "IsBusinessException" from pHeaders, return its boolean value
 */
public boolean isBusinessException() {
    String headerValue = getHeaderValue(pHeaders,"IsBusinessException");
    boolean result=Boolean.parseBoolean(headerValue);
    return result;
}
public void convertToNativeData() throws DataBindingException {
    DataObject dataObject = getDataObject();
    this.nativeData=realConvertWorkFromSD0ToNativeData(dataObject);
}
public void convertFromNativeData(HTTPInputStream arg0){
    //Customer-developed method to
    //Read data from HTTPInputStream
    //Convert it to DataObject
    DataObject dataobject=realConvertWorkFromNativeDataToSD0(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 operațională Java EE ce rulează pe un server Java EE. Legările de export EJB permit componentelor SCA să fie afișate sub formă de Enterprise JavaBeans, astfel încât logica operațională 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 își conecteze logica operațională în mediul IBM Business Process Manager și să participe într-un proces operațional.

Folosiți Integration Designer pentru a crea legări de import EJB. Puteți utiliza oricare dintre următoarele proceduri pentru a genera legările:

- Creare import EJB folosind vrăjitorul de servicii externe
Puteți 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, interfețe și importă fișiere în baza serviciilor descoperite.
- Creare legări de import EJB folosind editorul de asamblare
Puteți să creați un import EJB într-o diagramă de asamblare utilizând editorul de asamblare Integration Designer. Din paletă, puteți 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. Aveți posibilitatea să legați în mod direct o componentă cu o referință WSDL (Web Services Description Language) la importul EJB care comunică cu un serviciu bazat pe EJB folosind o interfață Java.

Importul EJB poate interacționa cu logica operațională Java EE folosind fie modelul de programare EJB 2.1, fie modelul de programare EJB 3.0.

Invocarea logicii operaționale Java EE poate fi locală (doar pentru EJB 3.0) sau de la distanță.

- Invocarea locală este utilizată atunci când doriți să apelați logica operațională Java EE care se află pe același server ca și importul.

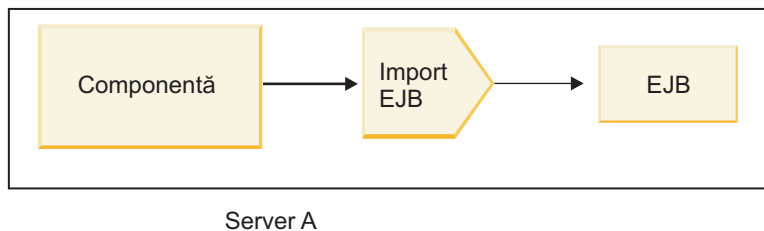


Figura 28. Invocarea locală a unui EJB (doar EJB 3.0)

- Invocarea de la distanță este utilizată atunci când doriți să apelați logica operațională Java EE care nu se află pe același server ca și 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.

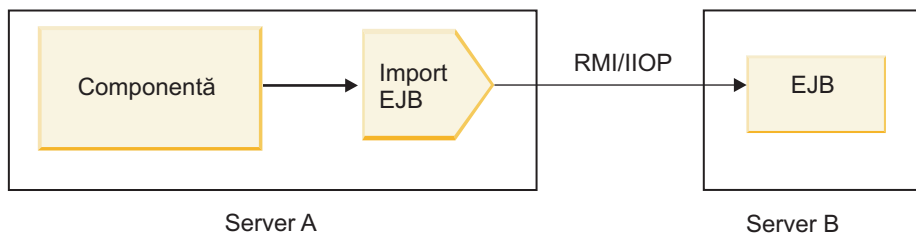


Figura 29. Invocare de la distanță a unui EJB

Atunci când configurează legarea EJB, Integration Designer folosește numele JNDI pentru a determina nivelul modelului de programare EJB și tipul de invocare (ocal sau de la distanță).

Legările de export EJB conțin următoarele componente majore:

- Handler date JAX-WS
- Selector de defectare EJB
- Selector pentru funcția de import EJB

Dacă scenariul dvs. de utilizator nu se bazează pe maparea JAX-WS, s-ar putea să aveți nevoie de un handler de date personalizat, un selectorul de funcție și 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:

Aplicațiile Java EE externe pot invoca o componentă SCA prin intermediul unei legări de export EJB. Folosind un export EJB puteți expune componente SCA, astfel încât aplicațiile Java EE externe pot invoca aceste componente folosind modelul de programare EJB.

Notă: Exportul EJB este un bean stateless.

Folosiți Integration Designer pentru a crea legări EJB. Puteți utiliza oricare dintre următoarele proceduri pentru a genera legările:

- Crearea legărilor de export EJB folosind vrăjitorul pentru servicii externe
Puteți 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, interfețe și exportă fișiere pe baza serviciilor descoperite.
- Crearea legărilor de export EJB folosind editorul de asamblare
Puteți să creați 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.

Puteți genera legarea dintr-o componentă SCA existentă, sau puteți genera un export cu o legare EJB pentru o interfață Java.

- Atunci când generați un export dintr-o componentă SCA existentă care are o interfață WSDL existentă, exportulului i se alocă o interfață Java.
- Atunci când generați un export dintr-o interfață Java, puteți selecta fie un WSDL, fie o interfață Java pentru export.

Notă: O interfață Java folosită pentru a crea un export EJB are următoarele limitări în ceea ce privește obiectele (parametrii de ieșire și intrare și excepții) transmise ca parametrii la un apel de la distanță:

- Trebuie să fie de un tip concret (în loc de un tip interfață sau abstract).
- Acestea trebuie să fie în concordanță cu specificația Enterprise JavaBeans. Trebuie să fie serializabili și să aibă un constructor implicit fără argumente, iar toate proprietățile trebuie să fie accesibile prin intermediul metodelor getter și setter.

Consultați site-ul web Sun Microsystems, Inc., <http://java.sun.com> pentru informații legate de specificațiile JavaBeans Enterprise.

În plus, excepția trebuie să fie o excepție verificată, moștenită din `java.lang.Exception`, și trebuie să fie unică (adică, nu suportă să arunce mai multe tipuri de excepții verificate).

De asemenea, rețineți că interfața afacerii pentru un EnterpriseBean Java este o interfață Java simplă și nu trebuie să extindă `javax.ejb.EJBObject` sau `javax.ejb.EJBLocalObject`. Metodele interfeței de afaceri nu ar trebui să arunce `java.rmi.Remote.Exception`.

Legările de export EJB pot interacționa cu logica operațională 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 distanță.

- Invocarea locală este utilizată atunci când logica operațională Java EE apelează o componentă SCA care se află pe același server ca și exportul.
- Invocarea de la distanță este utilizată atunci când logica operațională Java EE nu se află pe același server ca și exportul.

De exemplu, în figura următoare, un EJB folosește RMI/IIOP pentru a apela o componentă SCA aflată pe un server diferit.

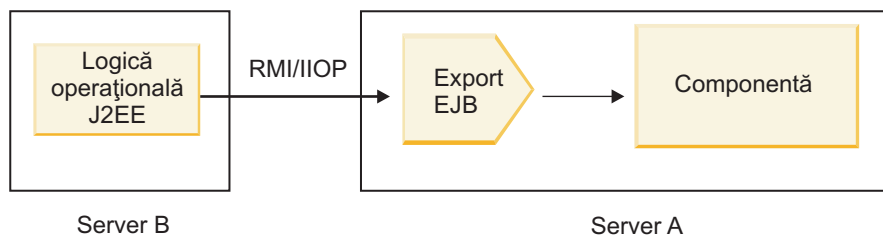


Figura 30. Un apel la distanță de la un client către o componentă SCA prin intermediul unui export EJB

Atunci când configurează legarea EJB, Integration Designer folosește numele JNDI pentru a determina nivelul modelului de programare EJB și tipul de invocare (ocal sau de la distanță).

Legările de export EJB conțin următoarele componente majore:

- Handler date JAX-WS
- Selector pentru funcția de export EJB

Dacă scenariul dvs. de utilizator nu se bazează pe maparea JAX-WS, s-ar putea să aveți nevoie de un handler de date personalizat și de un selectorul de funcție 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.

Proprietățile legărilor EJB:

Legările pentru import EJB își folosesc numele JNDI configurate pentru a determina nivelul modelului de programare EJB și tipul de invocare (local sau de la distanță). Legările de import și export EJB utilizează handler-ul de date JAX-WS pentru transformarea datelor. Legarea de import EJB folosește un selector pentru funcția de import EJB și un selector pentru defectul EJB, iar legarea de export EJB folosește un selector pentru funcția de export EJB.

Nume JNDI și legările de import EJB:

Atunci când configurează legarea EJB pentru un import, Integration Designer folosește numele JNDI pentru a determina nivelul modelului de programare EJB și tipul de invocare (local sau de la distanță).

În cazul în care nu este specificat niciun nume JNDI, se folosește legarea interfeței EJB implicite. Numele implicite care sunt create depind de faptul că invocați EJB 2.1 JavaBeans sau EJB 3.0 JavaBeans.

Notă: Consultați subiectul "Privire generală asupra legărilor de aplicație EJB 3.0" din Centrul de informare WebSphere Application Server pentru informații suplimentare detaliate despre convențiile 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 interfața home, separate prin linii oblice.

De exemplu, pentru interfața acasă a EJB 2.1 JavaBeans pentru `com.mycompany.myremotebusinesshome`, legarea implicită este:

```
ejb/com/mycompany/myremotebusinesshome
```

Pentru EJB 2.1, este suportată doar în vocarea EJB de la distanță.

- EJB 3.0 JavaBeans

Numele JNDI implicit preselectat de Integration Designer pentru JNDI-ul local este numele clasei complet calificate al interfeței locale precedate de **ejblocal:**. De exemplu, pentru interfața complet calificată a interfeței `com.mycompany.mylocalbusiness locale`, EJB-ul 3.0 JNDI preselectat este:

```
ejblocal:com.mycompany.mylocalbusiness
```

Pentru interfața `com.mycompany.myremotebusiness` de la distanță, EJB-ul 3.0 JNDI preselectat este interfața complet calificată:

com.mycompany.myremotebusiness

Legările aplicației implicite EJB 3.0 sunt descrise la următoarea locație: Privire generală asupra legărilor de aplicație EJB 3.0.

Integration Designer va folosi numele "scurt" drept locația JNDI implicită pentru EJB-uri folosind versiunea 3.0 a modelului de programare.

Notă: În cazul în care referința JNDI implementată a țintei EJB este diferită de locația legării JNDI implicite deoarece a fost utilizată sau configurată o mapare personalizată, numele JNDI-ului țintă trebuie să fie specificat în mod corespunzător. Aveți posibilitatea să specificați numele în Integration Designer înainte de implementare, sau, pentru legarea de import, puteți modifica numele în consola administrativă (după implementare) pentru a potrivi numele JNDI pentru EJB țintă.

Pentru informații suplimentare despre crearea legărilor EJB, vedeți secțiunea 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 și pentru a transforma valoarea returnată a obiectului Java în obiecte business răspuns. Legarea EJB de import folosește handler-ul de date JAX-WS pentru a transforma EJB-urile cerere în obiecte business cerere și pentru a transforma obiectul business răspuns într-o valoare returnată.

Acest handler de date mapează datele de la interfața WSDL specificată în SCA cu interfața Java EJB țintă (și vice versa) folosind specificațiile pentru JAX-WS (Java API for XML Web Services) și specificațiile pentru JAXB (Java Architecture for XML Binding).

Notă: Suportul actual este limitat la specificațiile pentru JAX-WS 2.1.1 și JAXB 2.1.3.

Handler-ul de date specificat la nivelul legării EJB este folosit pentru a realiza procesarea cererilor, răspunsurilor, defectelor și excepțiilor apărute în timpul rulării.

Notă: Pentru defecte, poate fi specificat un anumit handler de date pentru fiecare dintre ele prin specificarea proprietății de configurare `faultBindingType`. Acesta înlocuiește valoarea specificată la nivelul legării EJB.

Handler-ul de date JAX-WS este folosit în mod implicit atunci când legarea EJB are o interfață 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 folosește un handler de date pentru a transforma un obiect de date într-o matrice cu elemente de tip `Object` Java (`Object[]`). În timpul comunicațiilor ce au loc la ieșire în, are loc următoarea procesare:

1. Legarea EJB setează tipul așteptat, elementul așteptat și numele metodei țintă 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 reprezentați parametrii metodei (în ordinea definirii lor în cadrul metodei).
4. Legarea EJB folosește `Object[]` pentru a invoca metoda în interfața EJB țintă.

Legarea pregătește 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 necesar pentru a suporta parametrii de tip `In/Out` și `Out`.

Pentru parametrii de tip `Out`, valorile trebuie să fie returnate în obiectul datelor de răspuns.

Handler-ul de date procesează și transformă valorile găsite în Object[], iar apoi returnează un răspuns către obiectul de date.

Handler-ul de date suportă xs:AnyType, xs:AnySimpleType și xs:Any împreună cu alte tipuri de date XSD. Pentru a activa suportul pentru xs:Any, utilizați **@XmlAnyElement (lax=true)** pentru proprietatea JavaBeans în codul Java așa cum este afișat 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 specificația JAX-WS, modificați adnotările JAXB, astfel încât să se potrivească nevoilor afacerii dvs. Handler-ul de date JAX-WS suportă xs:any, xs:anyType și xs:anySimpleType.

Pentru handler-ul de date JAX-WS se aplică următoarele restricții:

- Handler-ul de date nu include suport pentru adnotarea **@WebParam** din atributul antetului.
- Spațiul de nume pentru fișierele schemei obiectului business (fișiere XSD) nu include maparea implicită din numele pachetului Java. Adnotarea **@XMLSchema** din package-info.java, de asemenea, nu funcționează. Singura modalitate de a crea un XSD cu un spațiu de nume este de a utiliza adnotările **@XmlType** și **@XmlRootElement**. **@XmlRootElement** definește spațiul nume țintă pentru elementul global în tipuri JavaBeans.
- Vrăjitorul pentru importul EJB nu creează fișiere XSD pentru clasele fără legătură. Versiunea 2.0 nu suportă adnotare **@XmlSeeAlso**, așa că, dacă nu se face referire în mod direct de la clasa părinte către clasa copil, nu se creează un XSD. Soluția la această problemă este să rulați 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 fișiere XSD pentru un bean dat. Aceste XSD-uri trebuie să fie copiate manual în modulul pentru ca soluția să funcționeze.

Selector defect EJB:

Selectorul defectului EJB determină dacă o invocare EJB a rezultat într-un defect, o excepție 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 excepție într-un obiect business defect.

Într-un răspuns cu succes (fără defect), legarea de import EJB assemblează 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 și Out.

Pentru scenariile de excepție, legarea assemblează un Object[], iar primul element reprezintă excepția aruncată de metodă.

Selectorul de defect poate returna oricare dintre următoarele valori:

Tabela 35. Valori returnate

Tip	Valoare returnată	Descrierea
Defect	ResponseType.FAULT	Returnată atunci când Object[] transmis conține un obiect de tip excepție.
Excepție runtime	ResponseType.RUNTIME	Returnată în cazul în care obiectul de tip excepție nu se potrivește cu niciunul dintre tipurile de excepții 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 și pentru a invoca handler-ul corect pentru datele defecte.

Selector funcție EJB:

Legările EJB folosesc un selector pentru funcția de import (pentru procesarea la ieșire) sau un selector pentru funcția de export (pentru procesarea la intrare) pentru a determina ce metodă EJB să apeleze.

Selector pentru funcția de import

Pentru procesarea la ieșire, selectorul pentru funcția de import derivă tipul metodei EJB în funcție de numele operației invocate de componenta SCA care este legată de importul EJB. Selectorul funcției caută adnotarea @WebMethod în clasa Java adnotată JAX-WS generată de Integration Designer pentru a determina numele operației țintă asociate.

- În cazul în care adnotarea @WebMethod există, selectorul funcției folosește această adnotare pentru a determina mapearea corectă a metodei Java pentru metoda WSDL.
- În cazul în care adnotarea @WebMethod lipsește, selectorul funcției presupune că numele metodei Java este același cu cel al operației invocate.

Notă: Acest selector de funcție este valid doar pentru o interfață de tip WSDL într-un import EJB, și nu pentru o interfață de tip Java dintr-un import EJB.

Selectorul funcției returnează un obiect java.lang.reflect.Method care reprezintă metoda interfeței EJB.

Selectorul funcției folosește un Obiect Java de tip matrice (Object[]) pentru a reține răspunsul de la metoda țintă. 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 funcția de export

Pentru procesarea la intrare, selectorul pentru funcția de export derivă metoda țintă, astfel încât să fie invocată din metoda Java.

Selectorul funcției de export mapează numele operației Java invocate de clientul EJB în numele operației din interfața componentei țintă. Numele metodei este returnat sub forma unui șir de caractere și este rezolvat de runtime-ul SCA în funcție de tipul interfeței al componentei țintă.

Legări EIS:

Legările EIS (Enterprise information system) asigură conectivitatea între componente SCA și un EIS extern. Această comunicație este realizată folosind exporturile EIS și importurile EIS care suportă adaptoarele de resurse JCA 1.5 și Websphere Adapters.

Componentele dumneavoastră SCA ar putea impune ca datele să fie transferate către sau de la un EIS extern. Atunci când creați un modul SCA care necesită o astfel de conectivitate, veți include (în plus față de componenta dvs. SCA) un import sau un export cu o legare EIS pentru comunicația cu un anumit EIS extern.

Adaptoarele de resurse din IBM Integration Designer sunt utilizate în contextul unui import sau export. Dezvoltați un import sau un export cu ajutorul vrăjitorului de servicii externe, iar în timpul dezvoltării includeți adaptorul de resurse. Un import EIS care permite aplicației dumneavoastră să invoce un serviciu într-un sistem EIS sau un export EIS care permite unei aplicații dintr-un sistem EIS să invoce un serviciu dezvoltat în IBM Integration Designer sunt create cu un adaptor de resurse. De exemplu, veți crea un import cu adaptorul JD Edwards pentru a invoca un serviciu în sistemul JD Edwards.

Atunci când utilizați vrăjitorul pentru servicii externe, informațiile legate de legarea EIS sunt create pentru dvs. De asemenea, puteți utiliza o altă unealtă, editorul de asamblare, pentru a adăuga sau modifica informațiile legate de legare. Vedeți Accesarea serviciilor externe cu adaptoare pentru informații suplimentare.

După ce modulul care conține legarea EIS este implementat pe server, puteți utiliza consola administrativă pentru a vizualiza informații 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 accesați servicii de pe un sistem de informații de întreprindere sau să vă faceți serviciile disponibile EIS-ului.

Următorul exemplu arată cum un modul SCA numit ContactSyncModule sincronizează informații de contact între un sistem Siebel și un sistem SAP.

1. Componenta SCA numită ContactSync ascultă (prin intermediul unui export de aplicație EIS numit Contact Siebel) modificări asupra contactelor Siebel.
2. Însăși componenta SCA ContactSync folosește o aplicație SAP (printr-un import de aplicație EIS) pentru a actualiza informațiile de contact SAP corespunzător.

Deoarece structurile de date folosite pentru memorarea contactelor sunt diferite în sistemele Siebel și SAP, componenta SCA ContactSync trebuie să ofere mapare.

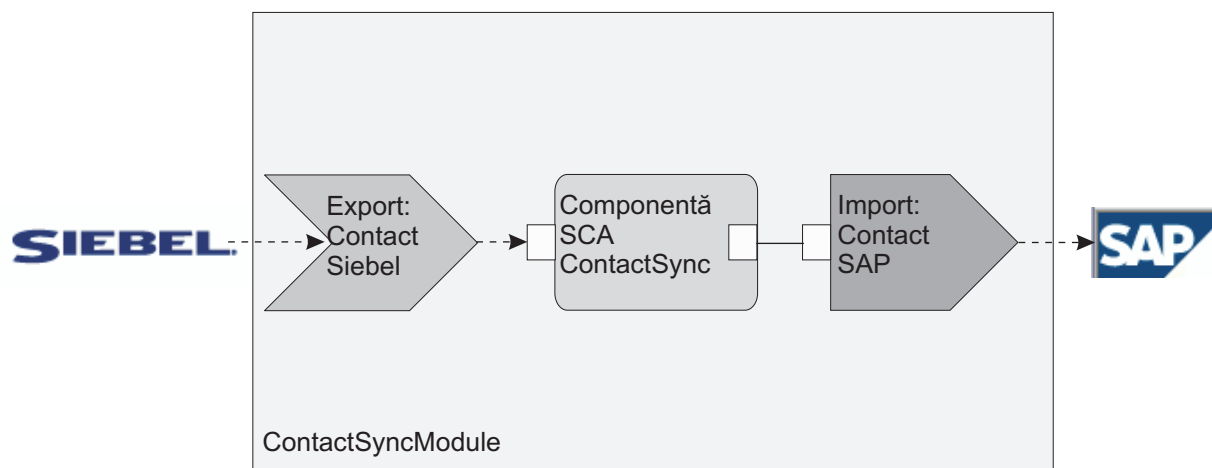


Figura 31. Flux de la un sistem Siebel la un sistem SAP

Exportul Contact Siebel și 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 aplicații 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 și sistemele EIS externe. Aplicațiile externe pot fi tratate ca un import EIS. În acest caz, importul EIS trimite date către EIS extern și primește opțional date în răspuns.

Importul EIS furnizează componentelor SCA cu o vizualizare uniformă a aplicațiilor 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 interfață, expusă de aplicația de import EIS, cu una sau mai multe metode, fiecare luând obiecte de date ca argumente și returnează valori. Pe partea de implementare, există un CCI implementat de un adaptor de resurse.

Implementarea runtime a importului EIS conectează interfața de pe partea clientului și CCI-ul. Importul mapează invocarea metodei pe interfață cu invocarea de pe CCI.

Legările sunt create la trei niveluri: legarea de interfață, ce apoi utilizează legările de metode conținute, ce mai apoi utilizează legările de date.

Legarea de interfață leagă interfața importului cu conexiunea la sistemul EIS ce furnizează aplicația. Acest lucru reflectă faptul că setul de aplicații, reprezentat de interfață, este furnizat de instanța specifică a EIS, iar conexiunea furnizează accesul la această instanță. Elementul de legare conține proprietăți cu destule informații pentru a crea conexiunea (aceste proprietăți fac parte din instanța `javax.resource.spi.ManagedConnectionFactory`).

Legarea de metodă asociază metoda cu interacțiunea specifică cu sistemul EIS. Pentru JCA, interacțiunea este caracterizată de setul de proprietăți al implementării de interfață `javax.resource.cci.InteractionSpec`. Elementul de interacțiune al legării de metodă conține aceste proprietăți, împreună cu numele clasei, astfel furnizând destule informații pentru a realiza interacțiunea. Legarea de metodă utilizează legări de date ce descriu mapearea argumentului și rezultatului metodei de interfață la reprezentarea EIS.

Scenariul runtime pentru un import EIS este după cum urmează:

1. Metoda de pe interfața de import este invocată utilizând modelul de programare SCA.
2. Cererea, ce ajunge la importul EIS, conține numele metodei și argumentele acesteia.
3. Importul întâi creează o implementare legare de interfață; apoi, utilizând date din legarea de import, acesta creează o `ConnectionFactory` și le asociază pe cele două. Adică, importul apelează `setConnectionFactory` pe legarea de interfață.
4. este creată implementarea legării de metodă ce se potrivește cu metoda invocată.
5. Instanța `javax.resource.cci.InteractionSpec` este creată și populată; apoi, legările de date sunt utilizate pentru a lega argumentele metodei la un format înțeles de adaptorul de resurse.
6. Interfața CCI este utilizată pentru a realiza interacțiunea.
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 și un EIS extern. Aplicațiile externe pot fi tratate ca un export EIS. În acest caz, aplicația externă trimite datele sale în formă de notificări periodice. Un export EIS poate fi gândit ca o aplicație de abonare ce ascultă o cerere externă de la EIS. Componenta SCA ce utilizează exportul EIS o vizualizează ca o aplicație locală.

Exportul EIS furnizează componentelor SCA o vizualizare uniformă a aplicațiilor 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 primește cereri de la EIS. Ascultătorul implementează o interfață ascultător specifică adaptorului de resurse. Exportul conține de asemenea o interfață ce implementează componente, expusă la EIS prin export.

Implementarea runtime a unui export EIS conectează ascultătorul cu interfața ce implementează componente. Exportul mapează cererea EIS cu invocarea operațiilor corespunzătoare de pe componente. Legările sunt create la trei niveluri: o legare ascultător, ce apoi utilizează o metodă nativă conținută , ce apoi utilizează o legare de date .

Legarea ascultător leagă ascultătorul ce recepționează cererile cu componenta expusă prin export. Definiția de export conține numele componentei; runtime-ul o localizează și înaintează cererile către aceasta.

Legarea de metodă nativă asociază metoda nativă sau tipul de eveniment recepționat de către ascultător cu operația implementată de componenta expusă prin calea exportului. Nu există nicio relație între metoda invocată pe ascultător și tipul de eveniment; toate evenimentele ajung prin una sau mai multe metode ale ascultătorului. Legarea de metodă nativă utilizează selectorul de funcții definit în export pentru a extrage numele metodei native din datele de intrare și legările de date pentru a lega formatul de date al EIS cu un format înțeles de componentă.

Scenariul runtime pentru un export EIS este după cum urmează:

1. Cererea EIS declanșează invocarea metodei pe implementarea ascultătorului.
2. Ascultătorul localizează și invocă exportul, pasându-i toate argumentele invocării.
3. Exportul creează implementarea legării ascultătorului.
4. Exportul instanțiază selectorul de funcții și îl setează pe legarea ascultătorului.
5. Exportul inițializează legările metodelor native și le adaugă la legarea ascultătorului. Pentru fiecare legare de metodă nativă, legările de date sunt de asemenea inițializate.
6. Exportul invocă legarea ascultătorului.
7. Legarea ascultătorului localizează componentele exportate și utilizează selectorul de funcții pentru a extrage numele metodei native.
8. Acest nume este utilizat pentru a localiza legarea metodei native, ce apoi invocă componente destinație.

Stilul de interacțiune al adaptorului permite legării de export EIS să invoce componenta destinație fie asincron (implicit) fie sincron.

Adaptoare de resurse

Dezvoltați un import sau un export cu vrăjitorul de servicii externe și, în dezvoltarea acestuia, includeți un adaptor de resurse. Adaptoarele ce vin cu IBM Integration Designer folosite pentru a accesa sisteme CICS, IMS, JD Edwards, PeopleSoft, SAP și Siebel sunt intenționate doar pentru scopuri de dezvoltare și testare. Acest lucru presupune că le utilizați pentru a dezvolta și a testa aplicațiile dumneavoastră.

Odată ce ați implementat aplicația, veți avea nevoie de adaptoare runtime licențiate pentru a o rula. Totuși, atunci când construiți serviciul dumneavoastră, puteți îngloba adaptorul împreună cu serviciul. Licențierea adaptorului vă poate permite să utilizați adaptorul înglobat ca adaptor runtime licențiat. 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 aplicații, care oferă gestiunea și securitatea ciclului de viață pentru tranzacții. Acestea se potrivesc de asemenea cu specificațiile Enterprise Metadata Discovery cu excepția adaptorului de resurse IBM CICS ECI și 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 aplicație care este gata de început, împachetată ca fișier EAR și implementată pe server. Toate artefactele Java EE și resursele sunt create; aplicația este configurată și gata de rulare.

Proprietăți dinamice JCA Interaction Spec și Connection Spec:

Legarea EIS poate accepta intrare pentru InteractionSpec și ConnectionSpec specificate prin utilizarea unui obiect de date copil bine-specificat ce acompaniază datele utile. Aceasta permite interacțiuni cerere-răspuns dinamice cu un adaptor de resurse prin InteractionSpec și autentificare de componentă prin ConnectionSpec.

javax.cci.InteractionSpec conține informații despre cum ar trebui tratată cererea de interacțiune cu adaptorul de resurse. Poate de asemenea conține informații despre cum a fost obținută interacțiunea după cerere. Aceste comunicări pe două căi prin interacțiuni sunt uneori referite ca *conversații*.

Legarea EIS așteaptă datele utile ce vor fi un argument pentru adaptorul de resurse să conțină un obiect de date copil numit **properties**. Acest obiect de date proprietăți va conține perechi nume/valoare, cu numele proprietăților Interaction Spec într-un anumit format. Regulile de formatare sunt:

- Numele trebuie să înceapă cu prefixul **IS**, urmat de numele proprietății. De exemplu, o interactionSpec cu o proprietate JavaBeans numită **InteractionId** ar specifica numele proprietății ca **ISInteractionId**.
- Perechea nume/valoare reprezintă numele și valoarea al tipului simplu al proprietății Interaction Spec.

În acest exemplu, o interfață specifică faptul că intrarea unei operații este un obiect de date **Account**. Această interfață invocă o aplicație de legare de import EIS cu intenția de a trimite și recepționa o proprietate InteractionSpec dinamică numită **workingSet** cu valoarea **xyz**.

Graficul operațional sau obiectele business de pe server conțin un obiect business **properties** de bază ce permite trimiterea datelor specifice-protocolului cu datele utile (payload). Acest obiect business **properties** este încorporat și nu trebuie specificat în schema XML când construiți un obiect business. Trebuie doar creat și utilizat. Dacă aveți definite propriile tipuri de date bazat pe o schemă XML, trebuie să specificați un element **properties** ce vă conține perechile nume/valoare așteptate.

```
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");
```

Creați 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`, setați valoarea așteptată (**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");
```

Puteți folosi proprietăți `ConnectionSpec` pentru autentificarea componentelor dinamice. Se aplică aceleași reguli ca mai sus, exceptând faptul că prefixul numelui proprietății trebuie să fie **CS** (în loc de **IS**). Proprietățile `ConnectionSpec` nu sunt cu sens dublu. Același obiect de date de **properties** poate conține atât proprietăți **IS**, cât și **CS**.

Pentru a utiliza proprietăți `ConnectionSpec`, setați **resAuth** specificat pe legarea de import la **Application**. De asemenea, asigurați-vă că adaptorul de resurse suportă autorizarea componentelor. Vedeți capitolul 8 din J2EE Connector Architecture Specification pentru detalii suplimentare.

Clienți externi cu legări EIS:

Serverul poate trimite mesaje către, sau primi mesaje de la, clienți 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 trebuie să fie invocat de o componentă din cadrul serverului.

Clientul invocă importul EIS la fel ca și cu orice altă aplicație, folosind fie DII (Dynamic Invocation Interface) fie interfața Java.

1. Clientul extern creează o instanță a `ServiceManager` și caută importul EIS folosindu-i numele de referință. Rezultatul căutării este o implementare a interfeței 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 interfeței `Service Data Object DataFactory`.
3. Clientul extern invocă EIS-ul și obține rezultatele necesare.

Alternativ, clientul poate invoca importul EIS folosind interfața Java.

1. Clientul creează o instanță a `ServiceManager` și caută importul EIS folosindu-i numele de referință. Rezultatul căutării este o interfață Java a importului EIS.
2. Clientul creează un argument de intrare și un obiect de date tastat.
3. Clientul invocă EIS și obține rezultatele necesare.

Interfața exportului EIS definește interfața componentei SCA exportate ce este disponibilă aplicațiilor EIS externe. Vă puteți gândi la această interfață ca la interfața pe care o va invoca o aplicație externă (precum SAP sau PeopleSoft) prin implementarea runtime-ului aplicației exportului EIS.

Exportul folosește `EISExportBinding` pentru a lega servicii exportate de aplicația EIS externă. Vă permite să abonați o aplicație conținută în modulul dumneavoastră SCA să asculte cereri de servicii EIS. Legarea de export EIS specifică maparea dintre definiția evenimentelor de intrare așa cum este înțeleasă de adaptorul de resurse (folosind interfețe Java EE Connector Architecture) și invocarea operațiilor 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 și pe modelul de programare. Acesta asigură fabrici de conexiune JMS pentru a crea conexiuni pentru destinațiile JMS și pentru a trimite și primi mesaje.

Legările JMS pot fi folosite atunci când interacționați cu legarea furnizorului SIB (Service Integration Bus) și sunt conforme cu JMS și JCA 1.5.

Puteți utiliza legările de export și import JMS ale unui modul SCA (Service Component Architecture) pentru a apela și primi mesaje de la sisteme JMS externe.

Legările de import și export JMS asigură integrarea în aplicațiile JMS folosind furnizorul SIB JMS bazat pe JCA 1.5 care face parte din WebSphere Application Server. Alte adaptoare de resursă JMS bazate pe JCA 1.5 nu sunt suportate.

Privire generală asupra legărilor JMS:

Legările JMS asigură conectivitatea între mediul SCA (Service Component Architecture) și sistemele JMS.

Legări JMS

Componentele majore ale ambelor legări JMS pentru import și export sunt:

- Adaptor resursă: activează conectivitatea bidirecțională, gestionată între un modul SCA și sistemele JMS externe
- Conexiuni: încapsulează o conexiune virtuală între un client și o aplicație furnizor
- Destinații: sunt folosite de un client pentru a specifica ținta mesajelor pe care le produce sau sursa mesajelor pe care le primește
- Date de autentificare: sunt folosite pentru a securiza accesul la legare

Caracteristicile cheie ale legărilor JMS

Anteturi speciale

Proprietățile anteturilor speciale sunt utilizate în importurile și exporturile JMS pentru a-i spune destinației cum să trateze mesajul.

De exemplu, TargetFunctionName mapează de la metoda nativă la metoda de funcționare.

Resurse Java EE

Un număr de resurse Java EE este creat la implementarea importurilor și exporturilor JMS într-un mediu Java EE.

ConnectionFactory

Utilizată de clienți pentru a crea o conexiune la furnizorul JMS.

ActivationSpec

Importurile utilizează acest lucru pentru a recepționa răspunsul la o cerere; exporturile o utilizează la configurarea punctelor finale ale mesajelor ce reprezintă ascultători de mesaje în interacțiunea acestora cu sistemul de mesagerie.

Destinații

- Destinație de trimitere: pe un import, aceasta este unde cererea sau mesajul de ieșire este trimis; pe un export, aceasta este destinația unde mesajul de răspuns va fi trimis, dacă nu este depreciată de câmpul antetului JMSReplyTo din mesajul de intrare.
- Destinație de recepționare: unde mesajul de intrare va fi plasat; cu importuri, acesta este un răspuns; cu exporturi, este o cerere.
- Destinație de callback: destinația sistemului SCA JMS utilizată pentru a stoca informații de corelare. Nu citați sau scrieți la această destinație.

Operația de instalare creează ConnectionFactory și trei destinații. De asemenea creează ActivationSpec pentru a permite ascultătorului de mesaje runtime să asculte pentru replici pe destinația de recepționare. Proprietățile ale acestor resurse sunt specificate în fișierul de import sau de export.

Integrare JMS și adaptoare 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).

Utilizați un furnizor JMS pentru adaptorul de resurse JCA 1.5 atunci când doriți integrarea într-un sistem extern JMS conform cu JCA 1.5. Serviciile externe conforme cu JCA 1.5 pot primi și 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 și export JMS:

Puteți face ca modulele SCA să interacționeze cu serviciile furnizate de aplicațiile JMS externe utilizând legări de import și export JMS.

Legări de import JMS

Conexiunile către furnizorul JMS asociat destinațiilor JMS sunt create folosind o fabrică de conexiuni JMS. Utilizați obiecte administrative pentru fabrica de conexiuni pentru a gestiona fabricile de conexiuni JMS pentru furnizorul implicit de mesaje.

Interacțiunea cu sistemele JMS externe include utilizarea destinațiilor pentru trimiterea cererilor sau primirea răspunsurilor.

Se oferă suport pentru două tipuri de scenarii de utilizare pentru legările de import JMS, în funcție de tipul operației invocate:

- Primul tip: Importul JMS pune un mesaj în destinația 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 destinația de trimitere, iar apoi persistă răspunsul pe care îl primește de la componenta SCA.

Legarea de import poate fi configurată (folosind câmpul **chema de corelare a răspunsului** în Integration Designer), astfel încât să aștepte 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 utilizeze o destinație de răspuns dinamică temporară pentru a corela răspunsurile cu cererile. Pentru fiecare cerere se creează o destinație temporară, iar importul folosește această destinație pentru a primi răspunsul.

Destinația de receive este setată în proprietatea antetului replyTo din mesajul de ieșire. Un ascultător de mesaje este implementat pentru a asculta la destinația de recepționare, iar în momentul în care este primit un răspuns, acesta transmite răspunsul înapoi către componentă.

Pentru ambele scenarii de utilizare pot fi specificate atât proprietățile dinamice, cât și cele statice ale antetului. Proprietățile statice pot fi setate din legarea metodei de import JMS. Unele dintre aceste proprietăți au semnificații speciale pentru runtime-ul JMS SCA.

Este important de menționat că JMS este o legare asincronă. Dacă o componentă apelantă invocă un import JMS în mod sincron (pentru o operație bidirecțională), atunci aceasta este blocată până când răspunsul este returnat de serviciul JMS.

Figura 32 la pagina 112 ilustrează modul în care importul este legat de serviciul extern.

Import JMS

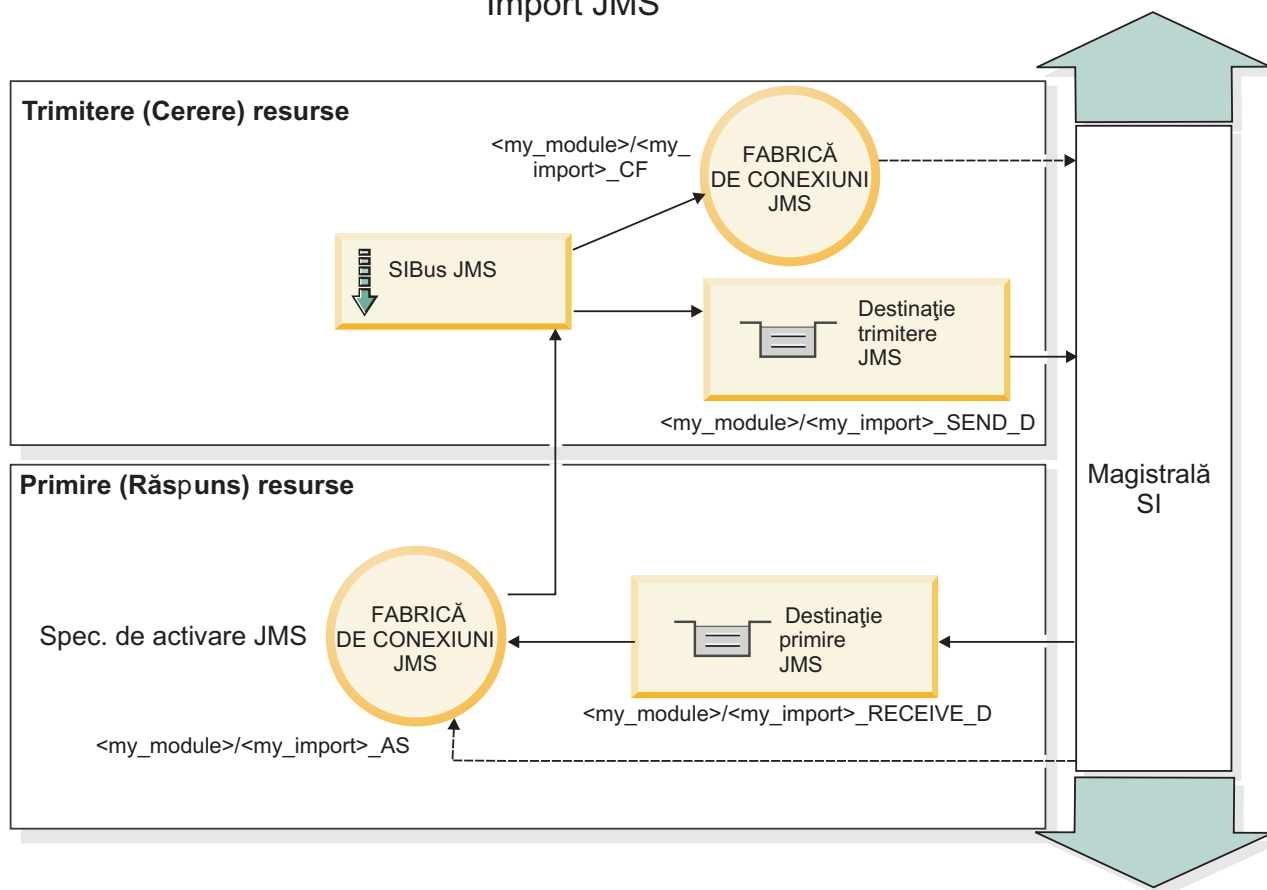


Figura 32. Resurse pentru legarea de import JMS

Legări de export JMS

Legări de export JMS pun la dispoziție mijloacele prin care modulele SCA asigură servicii către aplicațiile JMS externe.

Conexiunea care face parte dintr-un export JMS este o specificație de activare configurabilă.

Un export JMS a trimis și a primit destinațiile.

- Destinația receive este acolo unde ar trebui să fie pus mesajul de intrare pentru componenta țintă.
- Destinația sent este acolo unde va fi trimis răspunsul, cu excepția 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 destinația receive specificată în legarea de export. Destinația specificată în câmpul send este utilizat pentru a trimite răspunsul către cererea de intrare în cazul în care componenta invocată furnizează un răspuns. Destinația specificată în câmpul replyTo din mesajul de intrare înlocuiește destinația specificată în send.

Figura 33 la pagina 113 ilustrează modul în care solicitantul extern este legat de export.

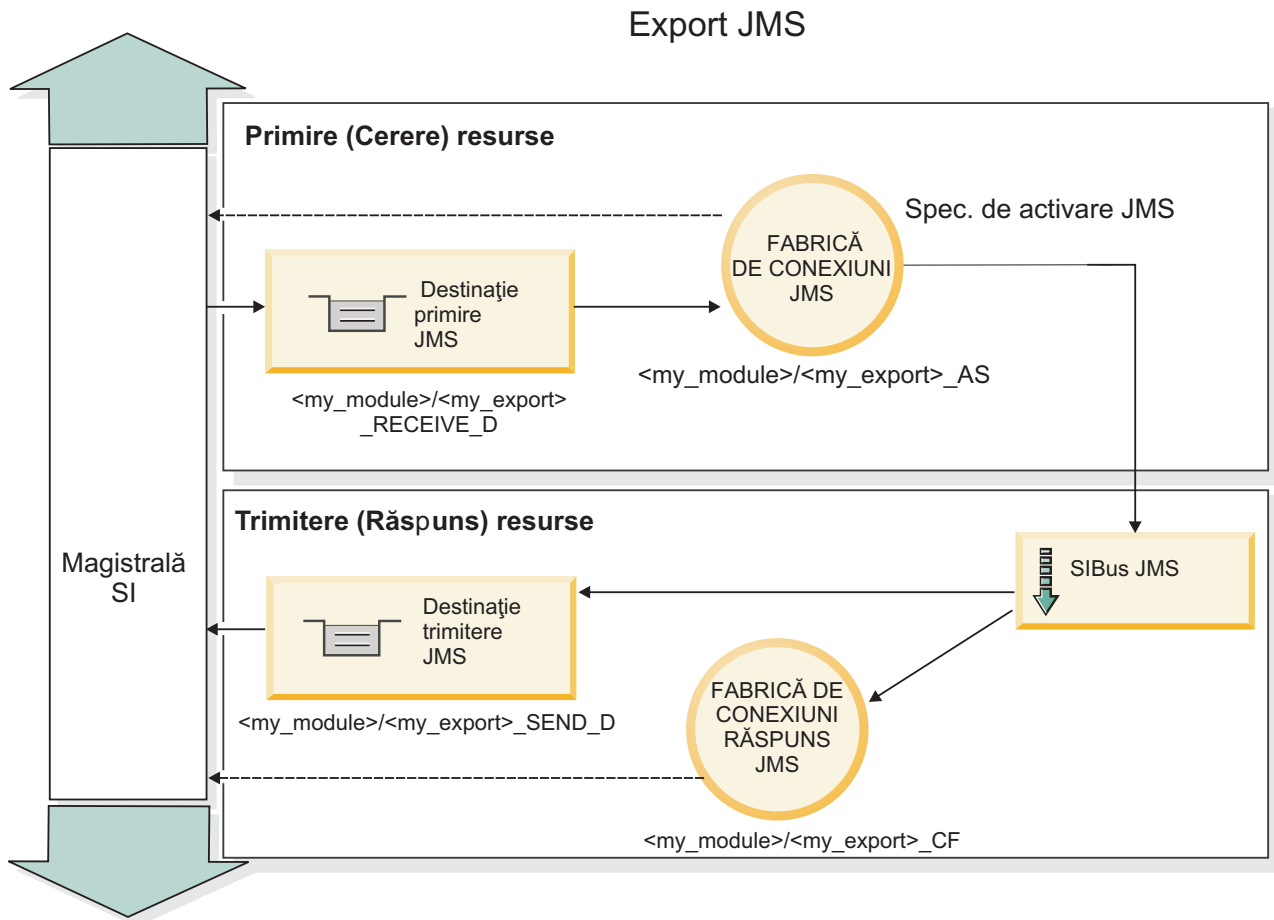


Figura 33. Resurse pentru legarea de export JMS

Anteturi JMS:

Un mesaj JMS conține două tipuri de anteturi - antet sistem JMS și proprietăți JMS multiple. 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 conține toate câmpurile care se găsesc de obicei într-un antet JMS. Deși 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 ieșire 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** și **JMSCorrelationID** - valori ale proprietăților antetului de mesaj predefinit
- **JMSDeliveryMode** - valori pentru modul de livrare (persistent sau nepersistent; implicit este persistent)
- **JMSPriority** - valoare de prioritate (de la 0 la 9; implicit este JMS_Default_Priority)

Proprietăți JMS

Proprietățile JMS sunt reprezentate în SMO sub formă de intrări în lista Proprietăți. Proprietățile pot fi adăugate, actualizate sau șterse într-o mediere sau folosind API-ul ContextService.

De asemenea, proprietățile pot fi setate în legarea JMS. Proprietățile care sunt setate în mod static înlocuiesc setările (cu același nume) care sunt setate în mod dinamic.

Proprietățile utilizatorului propagate din alte legări (de exemplu, o legare HTTP) va fi pusă în legarea JMS sub formă de proprietăți JMS.

Setările propagării anteturilor

Propagarea antetului și proprietăților sistemului JMS fie de la mesajul JMS de intrare către componentele următoare, fie de la componentele anterioare către mesajul JMS de ieșire poate fi controlată prin stegulețul Propagate Protocol Header din legare.

Atunci când Propagate Protocol Header este setat, informațiilor de antet le este permis să circule către mesaj sau către componenta țintă, așa cum este descris în următoarea listă:

- Cerere de export JMS
Antetul JMS primit în mesaj va fi propagat către o componentă țintă prin intermediul serviciului de context. Proprietățile JMS primite în mesaj vor fi propagate către o componentă țintă prin intermediul serviciului de context.
- Răspuns la exportul JMS
Oricare dintre câmpurile din antetul JMS din serviciul de context va fi utilizat în mesajul de ieșire, dacă nu este înlocuită de proprietățile statice stabilite în legarea de export JMS. Oricare dintre proprietățile setate în serviciul de context va fi utilizată în mesajul de ieșire, dacă nu este înlocuită de proprietățile 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 ieșire, dacă nu este înlocuit de proprietățile statice stabilite în legarea de import JMS. Oricare dintre proprietățile setate în serviciul de context va fi utilizată în mesajul de ieșire, dacă nu este înlocuită de proprietățile statice stabilite în legarea de import JMS.
- Răspuns import JMS
Antetul JMS primit în mesaj va fi propagat către o componentă țintă prin intermediul serviciului de context. Proprietățile JMS primite în mesaj vor fi propagate către o componentă țintă prin intermediul serviciului de context.

Schema de corelare a destinațiilor de răspuns dinamice temporare JMS:

Schema de corelare a destinațiilor de răspuns dinamice temporare determină o coadă dinamică unică sau un subiect care urmează să fie creat pentru fiecare cerere trimisă.

Destinația statică de răspuns menționată în import este utilizată pentru a obține natura cozii dinamice temporare de destinație sau subiectul. Acest lucru este setat în câmpul **ReplyTo** al cererii, iar importul JMS ascultă pentru răspunsuri pe această destinație. În cazul în care răspunsul este primit este cerut de destinația statică a răspunsului pentru prelucrarea asincronă. Câmpul **CorrelationID** din răspuns nu este utilizat și nu este nevoie să fie setat.

Probleme legate de Tranzacții

Atunci când este folosită o destinație dinamică temporară, cererea trebuie să fie prelucrat în același fir de execuție ca răspunsul trimis. Cererea trebuie să fie expediată în afara tranzacției globale, și trebuie să fie comisă înainte să fie primită de serviciul de backend, și ca un răspuns să fie returnat.

Persistență

Cozile temporare dinamice sunt entități de scurtă durată și nu garantează același nivel de persistență asociat cu o coadă sau subiect static. O coadă dinamică temporară sau subiect nu va mai exista după repornirea serverului și nici mesajele. După ce mesajul a fost cerut la destinația statică de răspuns, acesta își păstrează persistența definită în mesaj.

Timeout

Importul așteaptă să primească răspunsul la destinația 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 așteptare este depășit, importul aruncă o excepție de tip `ServiceTimeoutRuntimeException`.

Clienți externi:

Serverul poate trimite mesaje către, sau primi mesaje de la, clienți externi folosind legări JMS.

Un client extern (de exemplu, un portal web sau un sistem de informații 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 destinația de primire specificată în legarea de export. Destinația specificată în câmpul de trimitere este folosită pentru a trimite răspunsul către cererea de intrare dacă aplicația invocată furnizează un răspuns. Astfel, un client extern este capabil să invoce aplicații cu legarea de export.

Importurile JMS interacționează cu clienți externi prin trimiterea de mesaje și primirea de mesaje de la cozi JMS.

Lucrul cu clienți externi:

Un client extern (ce este în afara serverului) ar putea avea nevoie să interacționeze cu o aplicație instalată pe server.

Considerați un scenariu foarte simplu în care un client extern vrea să interacționeze cu o aplicație de pe server. Figura descrie un scenariu tipic simplu.

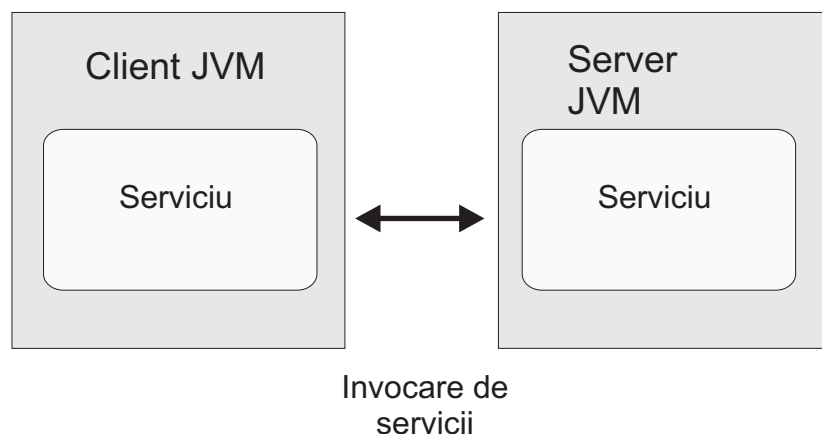


Figura 34. Scenariu simplu utilizare-caz: client extern interacționează cu aplicația serverului

Aplicația SCA include un export cu o legare JMS; aceasta face aplicația disponibilă pentru clienți externi.

Atunci când aveți un client extern în mașina virtuală Java (JVM) separat de serverul dumneavoastră, există mai mult pași pe care trebuie să-i realizați pentru a realiza o conexiune și pentru a interacționa cu un export JMS. Clientul obține un `InitialContext` cu valorile corecte și apoi caută resursele prin JNDI. Apoi clientul utilizează clientul specificație JMS 1.1 pentru a accesa destinațiile și mesajele de trimitere și primire a destinațiilor.

Numele implicite JNDI ale resurselor create automat prin runtime sunt listate în subiectul de configurare al acestei secțiuni. Totuși, dacă aveți resurse create anterior, utilizați acele nume JNDI.

1. Configurați destinațiile JMS și fabrica de conexiuni pentru a trimite mesajul.

2. Asigurați-vă de faptul că contextul JNDI, portul pentru adaptorul de resurse SIB și portul de mesagerie bootstrapping 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 instanțe de server. Puteți utiliza consola administrativă pentru a determina ce porturi ale serverului dumneavoastră sunt implementate. Mergeți la **Servere > servere de aplicații > nume_server > Configurae** și apăsați pe **Porturi** din **Comunicație**. Puteți edita portul ce este utilizat.
3. Clientul obține un context inițial cu valorile corecte și apoi caută resurse prin JNDI.
4. Utilizând specificațiile JMS 1.1, clientul accesează destinațiile și mesajele de trimitere și recepționare de pe destinații.

Depanare legări JMS:

Puteți diagnostica și repara problemele cu legările JMS.

Excepții de implementare

Ca răspuns la condiții de eroare, implementarea JMS import și export poate returna una din cele două tipuri de excepții:

- Service Business Exception: această excepție este returnată dacă a survenit eroarea specificată pe interfața serviciului business (tip port WSDL).
- Service Runtime Exception: apărută în toate celelalte cazuri. În cele mai multe cazuri, excepția cauze va conține excepția originală (JMSException).

De exemplu, un import așteaptă numai un mesaj de răspuns pentru fiecare mesaj de cerere. Dacă mai mult de un răspuns sosește, sau dacă un răspuns întârziat (unul pentru care expirarea răspunsului SCA are loc) sosește, o excepție Service Runtime Exception este aruncată. Tranzacția este repetată, și mesajul de răspuns este retras din coadă sau este manevrat de către managerul de eveniment.

Condiții de eșec primar

Condițiile de eșec primar ale legărilor JMS sunt determinate de către sensurile tranzacționale, de către configurația JMS provider, sau prin referire la comportamente existente în alte componente. Condițiile de eșec primar includ:

- Eșec la conexiunea la furnizorul sau destinația JMS.
Un eșec în conexiunea la furnizorul JMS pentru a primi mesaje va avea ca rezultat începutul căderii ascultătorului de mesaje. Această condiție va fi jurnalizată în istoricul WebSphere Application Server. Mesajele persistente vor rămâne la destinație până când vor fi retrase cu succes (sau expirate).
Un eșec la conectarea la furnizorul JMS pentru a trimite mesaje în exterior va cauza derularea înapoi a tranzacției ce controlează trimiterea.
- Eșecul în analiza unui mesaj de intrare sau în construirea unui mesaj de ieșire.
Un eșec în legarea de date sau în handler-ul de date cauzează derularea înapoi a tranzacției care controlează lucrul.
- Eșec la trimiterea mesajului de ieșire.
Un eșec la trimiterea unui mesaj cauzează derularea înapoi a tranzacției relevante.
- Mutiple sau neașteptate mesaje de răspuns întârziate.
Importul așteaptă 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 calificativu SCA Response Expiration qualifier la cerere. Când un răspuns sosește sau timpul de expirare este depășit, înregistrarea de corelare este ștearsă. Dacă mesajele de răspuns ajung pe neașteptat sau sosesc târziu, o excepție Service Runtime Exception este aruncată.
- Excepție runtime timeout serviciu cauzată de către răspunsul întârziat când se folosește schema de corelare a destinației de răspuns dinamic temporară.

JMS import va fi timeout după o perioadă de timp determinată de către calificativul SCA response expiration, 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 își au originea într-o cădere a JMS, ar trebui să vă așteptați să găsiți aceste mesaje în managerul de evenimente eşuate. Dacă astfel de mesaje nu apar în managerul de evenimente eşuate, asigurați-vă că destinația de bază SIB a destinației JMS are o valoare maximă de livrări nereușite mai mare decât 1. Setarea acestei valori la 2 sau mai mult activează interacțiunea cu managerul de evenimente eşuate în timpul invocărilor SCA pentru legările JMS.

Tratarea excepțiilor:

Modul în care este configurată legarea determină cum excepțiile 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 excepție.

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 excepție este determinat de cum implementați handler-ul de date sau legarea de date. Comportamentul recomandat este să vă proiectați legarea de date să arunce o **DataBindingException**.

Când orice excepție runtime, inclusiv o **DataBindingException**, este aruncată:

- Dacă fluxul de mediere este configurat să fie tranzacțional, mesajul JMS , implicit, este memorat în Managerul de evenimente eşuate pentru reluare sau ștergere manuală.

Notă: Puteți 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 tranzacțional, excepția este înregistrată în istoric și mesajul este pierdut.

Situația este similiară pentru un handler de date. Din moment ce handler-ul de date este invocat de legarea de date, orice excepție a handler-ului de date este înfășurată într-o excepție 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 terță parte care sunt conformi cu JMS 1.1. Operarea legărilor Generic JMS este similară cu cea a legărilor JMS.

Serviciul pus la dispoziție 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 și care suportă JMS 1.1 și implementează facilitatea opțională JMS Application Server. Legarea JMS Generic suportă acei furnizori JMS (inclusiv Oracle AQ, TIBCO, SonicMQ, WebMethods și BEA WebLogic) care nu suportă JCA 1.5, dar suportă Application Server Facility a specificației 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, utilizați “Legări JMS” la pagina 110.

Utilizați această legare generică la integrarea unui sistem JMS conform cu care nu este de tip JCA 1.5 în cadrul unui mediu SCA. Aplicațiile externe țintă pot primi și trimite apoi mesaje pentru a se integra într-o componentă SCA.

Privire generală asupra legărilor Generic JMS:

Legările Generic JMS sunt legări ne-JCA JMS care asigură conectivitate între SCA mediul (Service Component Architecture) și sistemele JMS care sunt în conformitate cu JMS 1.1 și care implementează facilitatea opțională a serverului de aplicații JMS.

Legări Generic JMS

Aspectele majore ale legărilor generice pentru import și export JMS includ următoarele:

- Port ascultător: permite furnizorilor JMS ce nu se bazează pe JCA să primească mesaje și să le trimită către un MDB (Message Driven Bean)
- Conexiuni: încapsulează o conexiune virtuală între un client și o aplicație furnizor
- Destinații: sunt folosite de un client pentru a specifica ținta mesajelor pe care le produce sau sursa mesajelor pe care le primește
- Date de autentificare: sunt folosite pentru a securiza accesul la legare

Legări de import Generic JMS

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 definiți de către un administrator. Fiecare fabrică de conexiuni este o instanță a uneia dintre interfețele `ConnectionFactory`, `QueueConnectionFactory` sau `TopicConnectionFactory`.

Interacțiunea cu sistemele JMS externe include utilizarea destinațiilor pentru trimiterea cererilor sau primirea răspunsurilor.

Se oferă suport pentru două tipuri de scenarii de utilizare pentru legările generice de import JMS, în funcție de tipul operației invocate:

- Primul tip: Importul generic JMS pune un mesaj în destinația de trimitere configurată în legarea de import. Nu se trimite nimic către câmpul `replyTo` din antetul JMS.
- Al doilea tip (cerere-răspuns): Importul Generic JMS pune un mesaj în destinația de trimitere, iar apoi persistă răspunsul pe care îl primește de la componenta SCA.

Destinația de primire este setată în proprietatea antetului `replyTo` din mesajul de ieșire. Un MDB (message driven bean) este implementat pentru a asculta la destinația de recepționare, 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 aștepte 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 proprietățile dinamice, cât și cele statice ale antetului. Proprietățile statice pot fi setate din legarea metodei de import generic JMS. Unele dintre aceste proprietăți au semnificații speciale pentru runtime-ul JMS SCA.

Este important de menționat faptul că JMS este o legare asincronă. Dacă o componentă apelantă invocă un import Generic JMS în mod sincron (pentru o operație bidirecțională), atunci aceasta este blocată până când răspunsul este returnat de serviciul JMS.

Figura 35 la pagina 119 ilustrează modul în care importul este legat de serviciul extern.

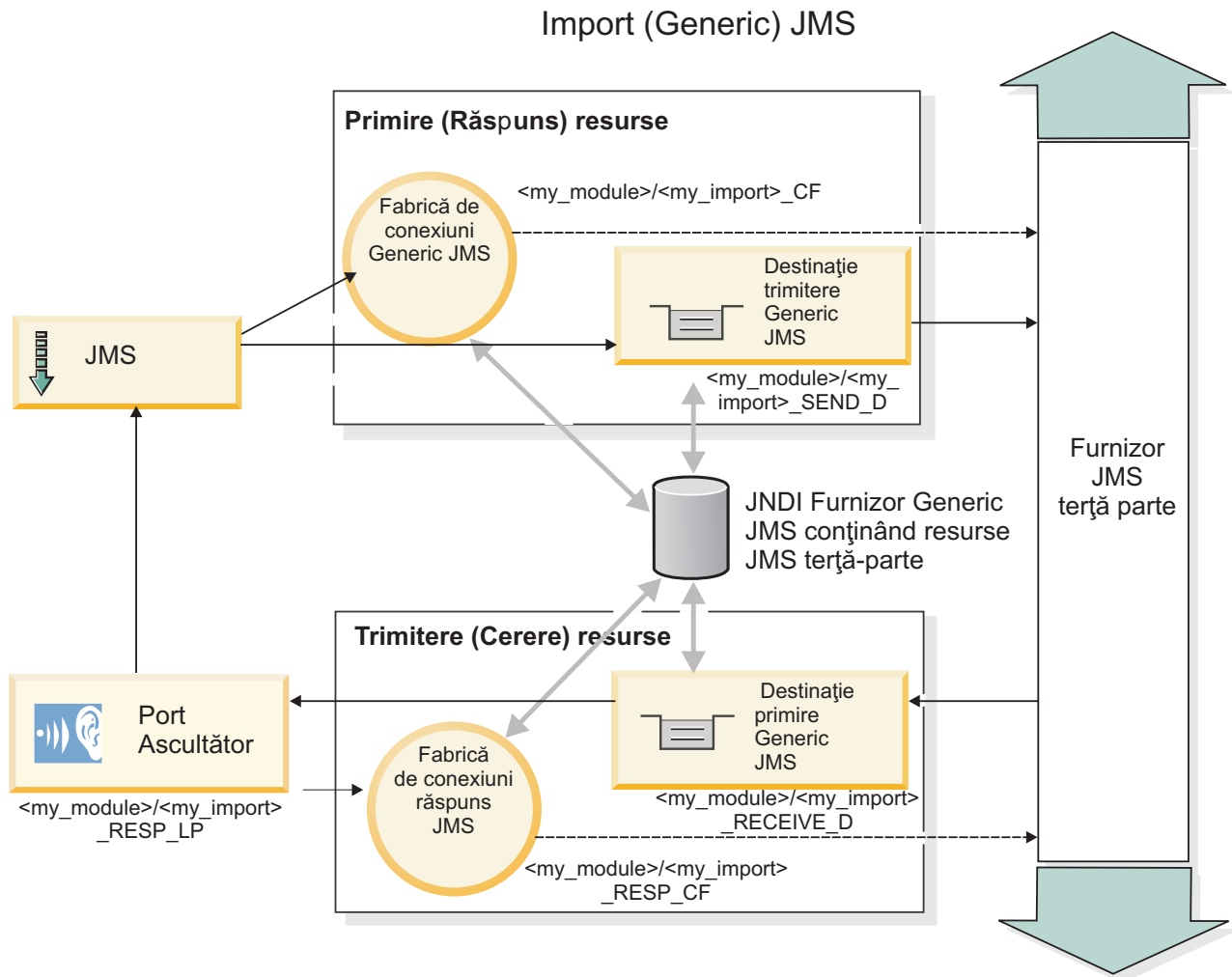


Figura 35. Resurse pentru legarea generică de import JMS

Legări de export Generic JMS

Legările de export Generic JMS pun la dispoziție mijloacele prin care modulele SCA asigură servicii către aplicațiile JMS externe.

Partea de conexiuni a unui export JMS este compus dintr-un ConnectionFactory și un ListenerPort.

Un export generic JMS a trimis și a primit destinațiile.

- Destinația primire este acolo unde ar trebui să fie pus mesajul de intrare pentru componenta țintă.
- Destinația trimitere este acolo unde va fi trimis răspunsul, cu excepția cazului în mesajul de intrare a înlocuit-o folosind proprietatea antetului replyTo.

Este implementat un MDB pentru a asculta cererile ce intră prin destinația primire specificată în legarea de export.

- Destinația specificată în câmpul trimitere este folosită pentru a trimite răspunsul către cererea de intrare în cazul în care aplicația invocată oferă un răspuns.
- Destinația specificată în câmpul replyTo din mesajul de intrare înlocuiește destinația 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 aștepte 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.

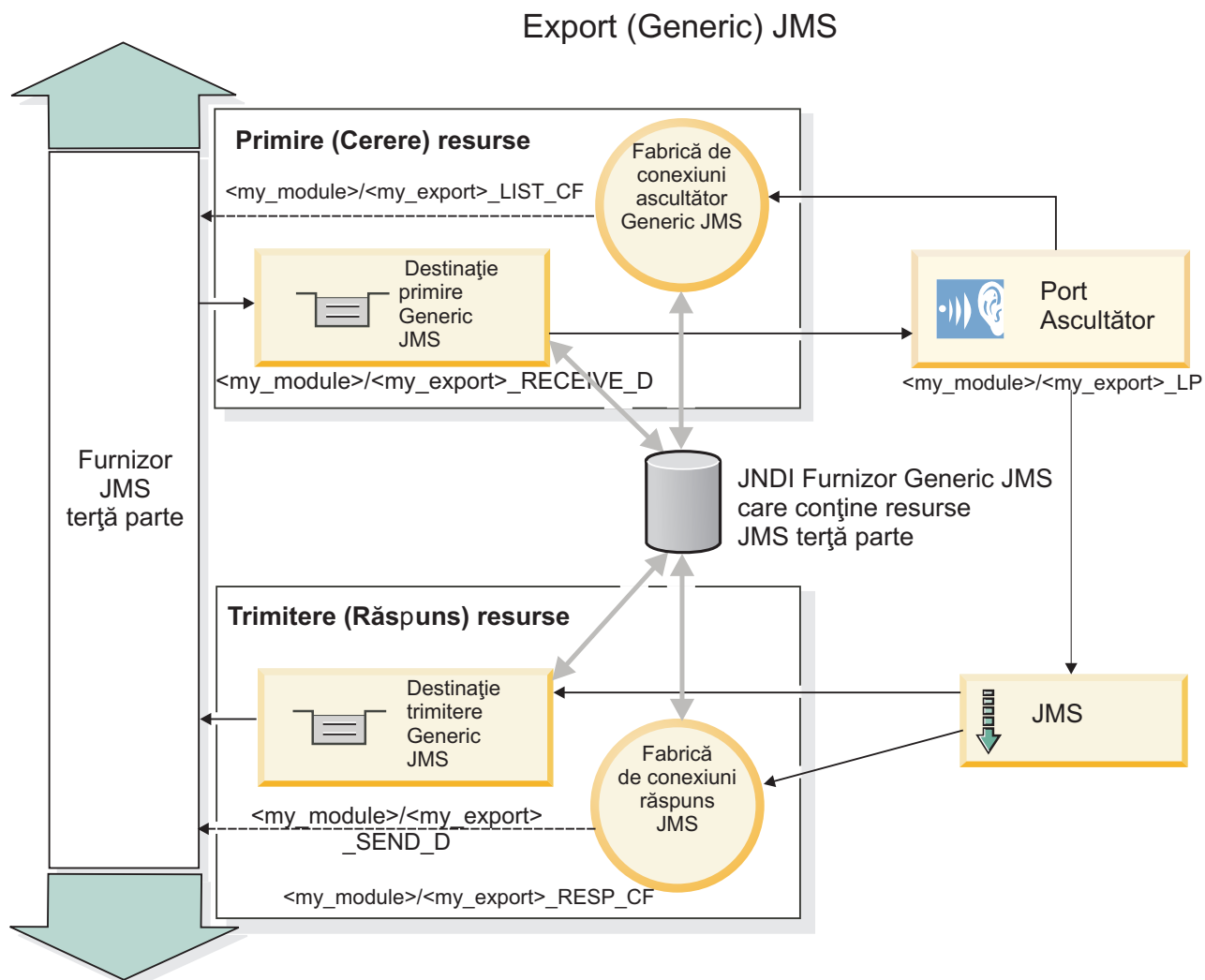


Figura 36. Resurse pentru legarea generică pentru exportul JMS

Caracteristici cheie ale legărilor JMS generic:

Caracteristicile legărilor de import și export JMS generic sunt consistente cu acelea ale legărilor de import MQ JMS și cele JMS imbricate WebSphere. Caracteristicile cheie includ definiții de antet și acces la resurse Java EE existente. Totuși, din cauza naturii sale generice, nu există nicio opțiune de conectivitate specifică furnizorilor JMS, iar această legare are capacitate limitată pentru a genera resurse la implementare și instalare.

Importuri generice

La fel ca aplicația de import MQ JMS, implementarea Generic JMS este asincronă și suportă trei invocări: o cale, două căi (de asemenea cunoscută ca cerere-răspuns), și 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) destinația trimisă cu cerere în câmpul de antetreplyTo al mesajului JMS.

Exporturi generice

Legările de exporturi Generic JMS diferă de legările de export EIS în cazul tratării returnării rezultatului. Un export Generic JMS trimite explicit răspunsul la destinația replyTo specificată în mesajul de intrare. Dacă niciuna nu este specificată, destinația 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 destinația de recepționare și apoi dispececează cererile pentru a fi procesate de runtime-ul SCA.

Anteturi speciale

Proprietățile anteturilor speciale sunt utilizate în importurile și exporturile Generic JMS pentru a informa legarea destinație cum să fie tratat mesajul.

De exemplu, proprietatea TargetFunctionName este utilizată de către selectorul de funcții implicit pentru a identifica numele operației din interfața export ce este invocată.

Notă: Legarea de import poate fi configurată pentru a seta antetul TargetFunctionName la numele de operație al fiecărei operații.

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 destinația de recepționare (răspuns) (doar două căi) pentru importuri și pe destinația de recepționare (cerere) pentru exporturi
- Fabrica de conexiuni Generic JMS pentru outboundConnection (import) și inboundConnection (export)
- Destinația Generic JMS pentru destinațiile de trimitere (import) și recepționare (export; doar două căi)
- Fabrica de conexiuni Generic JMS pentru responseConnection (doar două căi și opțional; altfel, outboundConnection este utilizat pentru importuri, iar inboundConnection este utilizat pentru exporturi)
- Destinația Generic JMS pentru destinațiile de recepționare (import) și trimitere (export) (doar două căi)
- Destinația JMS de callback a furnizorului de mesagerie implicit utilizată pentru a accesa destinația coadă de callback SIB (doar două căi)
- Fabrica de conexiuni JMS de callback a furnizorului de mesagerie implicit utilizată pentru a accesa destinația JMS de callback (doar două căi)
- Destinația coadă de callback SIB utilizată pentru a memora informațiile despre mesajul de cerere pentru utilizarea în timpul procesării răspunsului (doar două căi)

Taskul de instalare creeazăConnectionFactory, cele trei destinații, și ActivationSpec din informațiile din fișierele de import și export.

Anteturi Generic JMS:

Anteturile Generic JMS sunt SDO-uri (Service Data Objects) ce conțin toate proprietățile proprietăților mesajului JMS Generic. Aceste proprietăți pot fi de la mesajul de intrare sau pot fi proprietățile ce vor fi aplicate mesajului de ieșire.

Un mesaj JMS conține două tipuri de anteturi - antet sistem JMS și proprietăți JMS multiple. Ambele tipuri de anteturi pot fi accesate fie într-un modul de mediere din SMO (Service Message Object), fie folosind API-ul ContextService.

Următoarele proprietăți sunt setate static pe methodBinding:

- JMSType
- JMSCorrelationID
- JMSDeliveryMode
- JMSPriority

Legarea JMS Generic suportă modificarea dinamică a anteturilor și proprietăților JMS în același mod ca legările JMS și MQ JMS.

Unii furnizori de JMS Generic pun restricții pe ce proprietăți pot fi setate de aplicație și în ce combinații. Trebuie să vă consultați documentația produsului de terță parte pentru informații suplimentare. Totuși, o proprietate suplimentară a fost adăugată lui `methodBinding`, `ignoreInvalidOutboundJMSProperties`, ce permite să fie propagate orice excepții.

Anteturile și proprietățile mesajelor JMS Generic sunt folosite doar când comutatorul de bază al legării SCDDL a arhitecturii componente de servicii este pornit. Când comutatorul este pornit, informațiile de context sunt propagate. Implicit, comutatorul este pornit. Pentru a preveni propagarea informațiilor de context, modificați valoarea în **false**.

Când propagarea contextelor este activată, informațiile de antet sunt permise să curgă către mesaj sau către componenta țintă. Pentru a porni și opri propagarea contextelor, specificați **true** sau **false** pentru atributul `contextPropagationEnabled` al legărilor de import și export. De exemplu:

```
<esbBinding xsi:type="eis:JMSImportBinding" contextProgagationEnabled="true">
```

Valoarea implicită este **true**.

Depanarea legărilor JMS generic:

Puteți diagnostica și rezolva probleme cu legarea JMS generic.

Excepții de implementare

Ca răspuns la diversele condiții de eroare, implementarea Generic JMS import și export poate returna una din cele două tipuri de excepții:

- Service Business Exception: această excepție este returnată dacă fault specificată pe interfața serviciului business (WSDL port type) a survenit.
- Service Runtime Exception: apărută în toate celelalte cazuri. În cele mai multe cazuri, excepția cauze va conține excepția (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 și 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 ieșire. Notificarea expirării mesajului de apel invers va infica faptul că mesajul de cerere a expirat și clientul ar trebui notificat printr-o excepție operațională.

Dacă destinația de apel invers este mutată pe un furnizor terță parte, totuși, 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. Totuși, puteți verifica dacă răspunsul este expirat atunci când este primit.

Pentru mesaje de cerere de ieșire, valoarea JMSExpiration va fi calculată în timpul așteptat și din valorile requestExpiration efectuate în `asyncHeader`, dacă este setat.

Depanarea erorilor fabricii de conexiuni Generic JMS

Atunci când definiția anumite tipuri de fabrici de conexiuni în furnizorul dumneavoastră Generic JMS, s-ar putea să primiți un mesaj de eroare atunci când încercați să porniți aplicația. Puteți modifica fabrica de conexiuni externă pentru a evita această problemă.

Atunci când lansați o aplicație, este posibil să primiți următorul mesaj de eroare:

Tipul MDB Listener Port JMSConnectionFactory nu corespunde tipului JMSDestination.

Această problemă poate apărea atunci când definiți fabrici de conexiuni externe. Specific, excepția poate fi aruncată atunci când creați 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 și cea de publicare/anonare).

Pentru a rezolva această problemă, realizați următorii pași:

1. Accesați furnizorul Generic JMS pe care îl utilizați.
2. Înlocuiți fabrica de conexiuni subiect JMS 1.0.2 pe care ați definit-o cu fabrica de conexiuni (unificate) JMS 1.1.

Când lansați aplicația cu fabrica de conexiuni JMS 1.1 definită mai nou, ar trebui să nu mai primiți un mesaj de eroare.

Mesajele SCA bazate pe Generic JMS nu apar în managerul de evenimente eșuate

Dacă mesajele SCA au originea într-o eșuare a interacțiunii JMS, v-ați aștepta să găsiți aceste mesaje în managerul de evenimente eșuate. Dacă asemenea mesaje nu apar în managerul de evenimente eșuate, asigurați-vă că valoarea proprietății 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 interacțiunea cu managerul de evenimente eșuate în timpul invocării SCA pentru legările Generic JMS.

Tratarea excepțiilor:

Modul în care este configurată legarea determină cum sunt tratate excepțiile 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 excepție.

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 excepție este determinat de cum implementați handler-ul de date sau legarea de date. Comportamentul recomandat este să vă proiectați legarea de date să arunce o

DataBindingException.

Situația este similiară pentru un handler de date. Din moment ce handler-ul de date este invocat de legarea de date, orice excepție a handler-ului de date este înfășurată într-o excepție a legării de date. Prin urmare o

DataHandlerException vă este raportată ca o **DataBindingException.**

Când orice excepție runtime, inclusiv o excepție **DataBindingException**, este aruncată:

- Dacă fluxul de mediere este configurat să fie tranzacțional, mesajul JMS este memorat în Managerul de evenimente eșuate implicit pentru reluare sau ștergere manuală.

Notă: Puteți modifica modul de recuperare de pe legare, astfel încât mesajul să fie derulat înapoi în loc să fie stocat în managerul de evenimente eșuate.

- Dacă fluxul de mediere nu este tranzacțional, excepția este înregistrată în istoric și mesajul este pierdut.

Situația este similiară pentru un handler de date. Deoarece handler-ul de date este apelat de legarea de date, o excepție a handler-ului de date este produsă înăuntrul unei excepții 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 aplicațiile externe care folosesc un furnizor bazat pe WebSphere MQ JMS.

Utilizați WebSphere MQ JMS legările de export și 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 Service Integration Bus.

Atunci când o componentă interacționează cu un serviciu bazat pe WebSphere MQ JMS prin intermediul unui import, legarea de import WebSphere MQ JMS utilizează o destinație către care datele vor fi trimise și o destinație unde va fi primit răspunsul. Conversia datelor către și 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 clienților WebSphere MQ JMS, legarea de export WebSphere MQ JMS utilizează o destinație la care vor fi recepționate datele și către care poate fi trimis răspunsul. Conversia datelor către și de la un mesaj JMS se realizează prin intermediul handler-ului de date JMS sau prin legarea datelor.

Selectorul funcției asigură o mapare operației în cadrul componentei țintă care va fi invocată.

Privire generală asupra legărilor WebSphere MQ JMS:

Legarea WebSphere MQ JMS asigură integrarea în aplicațiile externe care folosesc furnizorul WebSphere MQ JMS.

Taskuri administrative WebSphere MQ

Se așteaptă 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 aplicație care conține 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 bazați pe WebSphere MQ JMS. Trebuie să utilizați o versiune suportată de WebSphere MQ. Cerințe detaliate despre hardware și 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 funcție de tipul de operație care este invocat:

- Primul tip: Importul WebSphere MQ JMS pune un mesaj în destinația 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 destinația de trimitere. Destinația de primire este setată în câmpul replyTo din antet. Un MDB (message-driven bean) este implementat pentru a asculta la destinația de recepționare, 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 aștepte 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 proprietățile dinamice, cât și cele statice ale antetului. Proprietățile statice pot fi setate din legarea metodei de import JMS. Unele dintre aceste proprietăți au semnificații speciale pentru runtime-ul JMS SCA.

Este important de menționat faptul că WebSphere MQ JMS este o legare asincronă. Dacă o componentă apelantă invocă un import WebSphere MQ JMS în mod sincron (pentru o operație bidirecțională), atunci aceasta este blocată până când răspunsul este returnat de serviciul JMS.

Figura 37 ilustrează modul în care importul este legat de serviciul extern.

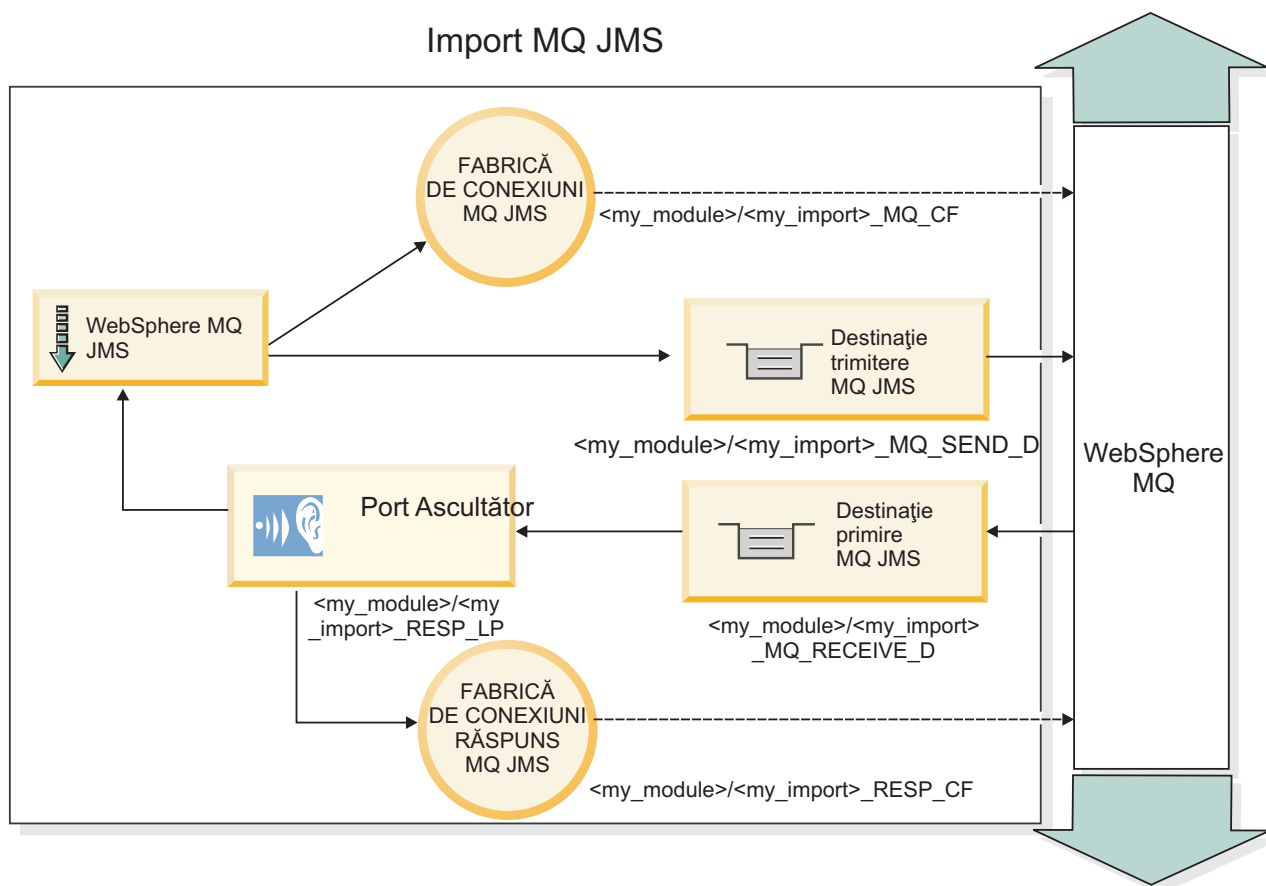


Figura 37. Resurse pentru legarea de import WebSphere MQ JMS

Legări pentru export WebSphere MQ JMS

Legarea de export WebSphere MQ JMS pune la dispoziție mijloacele prin care modulele SCA asigură servicii către aplicațiile externe bazate pe WebSphere MQ.

Este implementat un MDB pentru a asculta cererile ce intră prin destinația primire specificată în legarea de export. Destinația specificată în câmpul trimite este utilizat pentru a trimite răspunsul către cererea de intrare în cazul în care componenta invocată furnizează un răspuns. Destinația specificată în câmpul replyTo din mesajul de răspuns înlocuiește destinația specificată în câmpul trimite.

Figura 38 la pagina 126 ilustrează modul în care solicitantul extern este legat de export.

Export MQ JMS

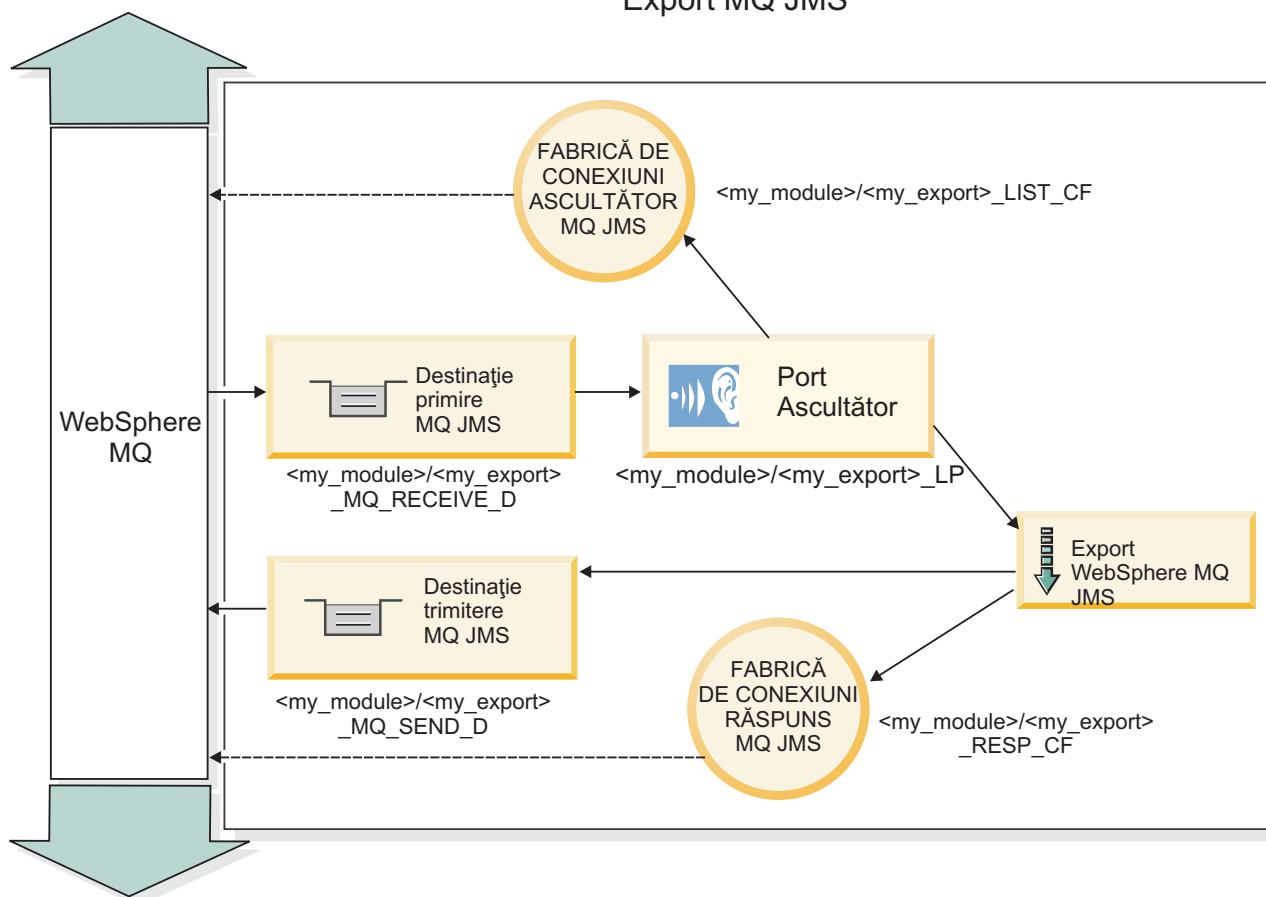


Figura 38. Resurse pentru legarea de export WebSphere MQ JMS

Notă: Figura 37 la pagina 125 și Figura 38 ilustrează modul în care o aplicație dintr-o versiune anterioară a IBM Business Process Manager este legată de un serviciu extern. Pentru aplicațiile dezvoltate pentru IBM Business Process Manager Versiunea 7.0, se folosește Specificare activare în loc de Port ascultător și Fabrică de conexiuni.

Caracteristici cheie ale legărilor WebSphere MQ JMS:

Caracteristicile cheie ale legărilor WebSphere MQ JMS includ anteturi, artefacte Java EE, și resurse create Java EE.

Anteturile

Un antet de mesaj JMS conține un număr de câmpuri predefinite ce conțin valori utilizate de și de clienți și de furnizori pentru a identifica și a ruta mesajele. Puteți folosi proprietățile de legare pentru a configura aceste anteturi cu valori fixe, sau anteturile pot fi specificate dinamic la rulare.

JMSCorrelationID

Legări la un mesaj înrudit. În mod tipic, acest câmp este setat la șirul identificatorului mesajului căruia îi este dat un răspuns.

TargetFunctionName

Acest antet este utilizat de unul din selectorii de funcții furnizați pentru a identifica operația ce este invocată. Setarea proprietății de antet JMS TargetFunctionName în mesajele trimise la un export JMS permite acestui selector de funcții să fie utilizat. Proprietatea poate fi setată direct în aplicațiile 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 operație din interfață 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 creere 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.

Parametrii

Fabrica de conexiuni MQ

Utilizată de clienți pentru a crea o conexiune la furnizorul MQ JMS.

Fabrica de conexiuni răspuns

Folosit de runtime-ul SCA MQ JMS atunci când destinația de trimitere este pe un Manager coadă diferit de cel al destinației de primire.

Specificații de activare

O specificație de activare MQ JMS este asociată cu una sau mai multe bean-uri bazate pe mesaje și furnizează configurația necesară pentru a recepționa mesajele.

Destinații

- Trimitere destinație:
 - Importuri: Unde sunt trimise mesajele de cerere sau de ieșire.
 - Exporturi: Unde vor fi trimise mesajele de răspuns dacă nu sunt permise de câmpul de antet JMSReplyTo al mesajului de intrare.
- Destinație de recepționare:
 - Importuri: Unde vor fi plasate mesajele de intrare și de răspuns.
 - Exporturi: Unde vor fi plasate mesajele de intrare și de cerere.

Anteturi JMS:

Un mesaj JMS conține două tipuri de anteturi - antet sistem JMS și proprietăți JMS multiple. 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 conține toate câmpurile care se găsesc de obicei într-un antet JMS. Deși 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 ieșire 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** și **JMSCorrelationID** - valori ale proprietăților antetului de mesaj predefinit
- **JMSDeliveryMode** - valori pentru modul de livrare (persistent sau nepersistent; implicit este persistent)
- **JMSPriority** - valoare de prioritate (de la 0 la 9; implicit este JMS_Default_Priority)

Proprietăți JMS

Proprietățile JMS sunt reprezentate în SMO sub formă de intrări în lista Proprietăți. Proprietățile pot fi adăugate, actualizate sau șterse într-o mediere sau folosind API-ul ContextService.

De asemenea, proprietățile pot fi setate în legarea JMS. Proprietățile care sunt setate în mod static înlocuiesc setările (cu același nume) care sunt setate în mod dinamic.

Proprietățile utilizatorului propagate din alte legări (de exemplu, o legare HTTP) va fi pusă în legarea JMS sub formă de proprietăți JMS.

Setările propagării anteturilor

Propagarea antetului și proprietăților sistemului JMS fie de la mesajul JMS de intrare către componentele următoare, fie de la componentele anterioare către mesajul JMS de ieșire poate fi controlată prin stegulețul Propagate Protocol Header din legare.

Atunci când Propagate Protocol Header este setat, informațiilor de antet le este permis să circule către mesaj sau către componenta țintă, așa cum este descris în următoarea listă:

- Cerere de export JMS
Antetul JMS primit în mesaj va fi propagat către o componentă țintă prin intermediul serviciului de context. Proprietățile JMS primite în mesaj vor fi propagate către o componentă țintă prin intermediul serviciului de context.
- Răspuns la exportul JMS
Oricare dintre câmpurile din antetul JMS din serviciul de context va fi utilizat în mesajul de ieșire, dacă nu este înlocuită de proprietățile statice stabilite în legarea de export JMS. Oricare dintre proprietățile setate în serviciul de context va fi utilizată în mesajul de ieșire, dacă nu este înlocuită de proprietățile 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 ieșire, dacă nu este înlocuit de proprietățile statice stabilite în legarea de import JMS. Oricare dintre proprietățile setate în serviciul de context va fi utilizată în mesajul de ieșire, dacă nu este înlocuită de proprietățile statice stabilite în legarea de import JMS.
- Răspuns import JMS
Antetul JMS primit în mesaj va fi propagat către o componentă țintă prin intermediul serviciului de context. Proprietățile JMS primite în mesaj vor fi propagate către o componentă țintă prin intermediul serviciului de context.

Clienți externi:

Serverul poate trimite mesaje către, sau să primească mesaje de la, clienți externi folosind legări JMS WebSphere MQ.

Un client extern (de exemplu, un portal web sau un sistem de informații pentru întreprindere) poate trimite un mesaj către o componentă SCA din aplicație prin intermediul unui export sau poate fi invocat de o componentă SCA din aplicație prin intermediul unui import.

Legarea de export WebSphere MQ JMS implementează MDB-uri (message driven beans) cu scopul de a asculta ceririle ce intră prin destinația primire specificată în legarea de export. Destinația specificată în câmpul trimite este folosită pentru a trimite răspunsul către cererea de intrare în cazul în care aplicația invocată oferă un răspuns. Cu toate acestea, un client extern poate invoca aplicații prin intermediul legării de export.

WebSphere MQ JMS importă legarea în, și poate trimite un mesaj către, clienții externi. Acest mesaj ar putea sau nu să solicite un răspuns de la clientul extern.

Informații suplimentare despre modul de interacționare cu clienții externi folosind WebSphere MQ pot fi găsite în Centrul de informare WebSphere MQ.

Depanarea legărilor WebSphere MQ JMS:

Puteți diagnostica și rezolva probleme cu legările WebSphere MQ JMS.

Excepții de implementare

Ca răspuns la diferite condiții de eroare, implementarea MQ JMS import și export poate returna unul din cele două tipuri de excepții:

- Service Business Exception: această excepție este returnată dacă fault specificată pe interfața serviciului business (WSDL port type) a survenit.
- Service Runtime Exception: apărută în toate celelalte cazuri. În cele mai multe cazuri, cause exception va conține excepția originală (JMSEException).

De exemplu, un import așteaptă numai un mesaj de răspuns pentru fiecare mesaj de cerere. Dacă mai mult de un răspuns sosește, sau dacă un răspuns întârziat (unul pentru care expirarea răspunsului SCA are loc) sosește, o excepție Service Runtime Exception este aruncată. Tranzacția este repetată, și 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 își au originea într-un eșec de interacțiune WebSphere MQ JMS, ar trebui să vă așteptați să găsiți aceste mesaje în managerul de evenimente eşuate. Dacă astfel de mesaje nu apar în managerul de evenimente eşuate, asigurați-vă că valoarea proprietății 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 interacțiunea cu managerul de evenimente eşuate în timpul invocarilor SCA pentru legările MQ JMS.

Scenarii de folosire greșită: comparație cu legările WebSphere MQ

Legările WebSphere MQ JMS sunt proiectate să interacționeze cu aplicațiile JMS implementate în comparație cu WebSphere MQ, care expune mesajele în concordanță cu modelul de mesaje JMS. Totuși, importul și exportul WebSphere MQ, sunt în principal proiectate să interacționeze cu aplicațiile native WebSphere MQ și să expună întregul conținut al corpului mesajului WebSphere MQ la mediere.

Următoarele scenarii ar trebui construite folosind legarea WebSphere MQ JMS, și 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.
- Permișiunea 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 conținutului 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 situații în care legarea WebSphere MQ și legarea WebSphere MQ JMS se așteaptă să interacționeze. În particular, când se face o punte între aplicații Java EE și non-Java EE WebSphere MQ, se va folosi export WebSphere MQ și import WebSphere MQ JMS (sau invers) în concordanță cu legările de date potrivite sau modulele de mediere potrivite (sau ambele).

Tratarea excepțiilor:

Modul în care este configurată legarea determină cum sunt tratate excepțiile 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 excepție.

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 excepție este determinat de cum implementați handler-ul de date sau legarea de date. Comportamentul recomandat este să vă proiectați legarea de date să arunce o **DataBindingException**.

Situația este similiară pentru un handler de date. Din moment ce handler-ul de date este invocat de legarea de date, orice excepție a handler-ului de date este înfășurată într-o excepție a legării de date. Prin urmare o **DataHandlerException** vă este raportată ca o **DataBindingException**.

Când orice excepție runtime, inclusiv o excepție **DataBindingException**, este aruncată:

- Dacă fluxul de mediere este configurat să fie tranzacțional, mesajul JMS este memorat în Managerul de evenimente eşuate implicit pentru reluare sau ștergere manuală.

Notă: Puteți 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 tranzacțional, excepția este înregistrată în istoric și mesajul este pierdut.

Situația este similiară pentru un handler de date. Deoarece handler-ul de date este apelat de legarea de date, o excepție a handler-ului de date este produsă înăuntrul unei excepții 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 aplicațiile WebSphere MQ.

Utilizați legările de export și 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 Service Integration Bus.

Atunci când o componentă interacționează cu un serviciu WebSphere MQ prin intermediul unui import, legarea de import WebSphere MQ utilizează o coadă către care vor fi trimise datele și o coadă unde poate fi primit răspunsul.

Atunci când un modul SCA oferă un serviciu clienților WebSphere MQ, legarea de export WebSphere MQ folosește o coadă în care pot fi primite cererile și către care pot fi trimise răspunsurile. Selectorul funcției asigură o mapare operației în cadrul componentei țintă care va fi invocată.

Conversia datelor de sarcină utilă către și 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 și de la un mesaj MQ se realizează prin intermediul legării datelor din antetul MQ.

Pentru informații despre versiunile WebSphere MQ suportate, vedeți pagina web cu cerințe detaliate de sistem.

Privire generală asupra legărilor WebSphere MQ:

Legarea WebSphere MQ asigură integrarea în aplicațiile native bazate pe MQ.

Taskuri administrative WebSphere MQ

Se așteaptă 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 aplicație care conține aceste legări.

Taskuri administrativ WebSphere

Trebuie să setați proprietatea **Cale bibliotecă nativă** din adaptorul pentru resurse MQ în Websphere cu versiunea WebSphere MQ suportată de server, și să reporniți serverul. Acest lucru asigură faptul că sunt utilizate bibliotecile unei versiuni acceptate a WebSphere MQ. Cerințe detaliate despre hardware și software se pot găsi pe Paginile de suport IBM .

Legări de import WebSphere MQ

Legarea de import WebSphere MQ permite componentelor din modul dumneavoastră SCA să comunice cu serviciile oferite de aplicațiile externe bazate pe WebSphere MQ. Trebuie să utilizați o versiune suportată de WebSphere MQ. Cerințe detaliate despre hardware și software se pot găsi pe Paginile de suport IBM .

Interacțiunea 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 funcție de tipul de operație invocată:

- Într-o direcție: Importul WebSphere MQ pune un mesaj pe coada configurată în câmpul **Coadă destinație 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 destinație de trimitere**

Coadă 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 aștepte 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 menționat faptul că WebSphere MQ este o legare asincronă. Dacă o componentă apelantă invocă un import în mod sincron WebSphere MQ (pentru o operație bidirecțională), 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.

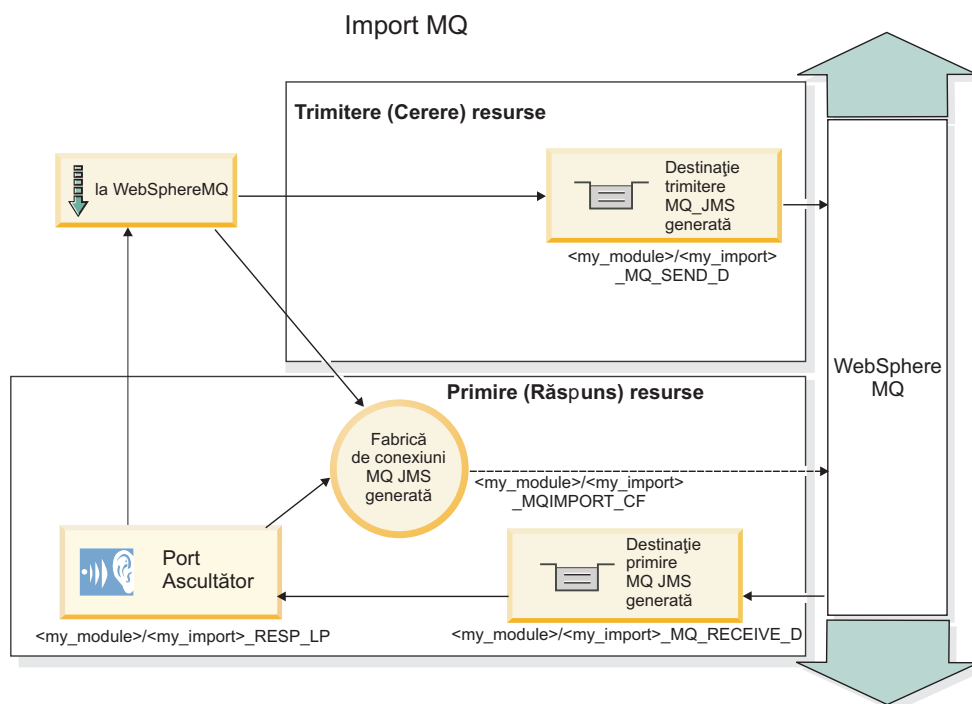


Figura 39. Resurse pentru legarea de import WebSphere MQ

Legări pentru export WebSphere MQ

Legarea de export WebSphere MQ pune la dispoziție mijloacele prin care modulele SCA asigură servicii către aplicațiile externe bazate pe WebSphere MQ.

Este implementat un MDB pentru a asculta cererile de intrare în **Coadă destinații de primire** specificată în legarea de export. Coada specificată în câmpul **Coadă destinații 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 înlocuiește coada specificată în câmpul **Coadă destinații de trimitere** field.

Figura 40 ilustrează modul în care solicitantul extern este legat de export.

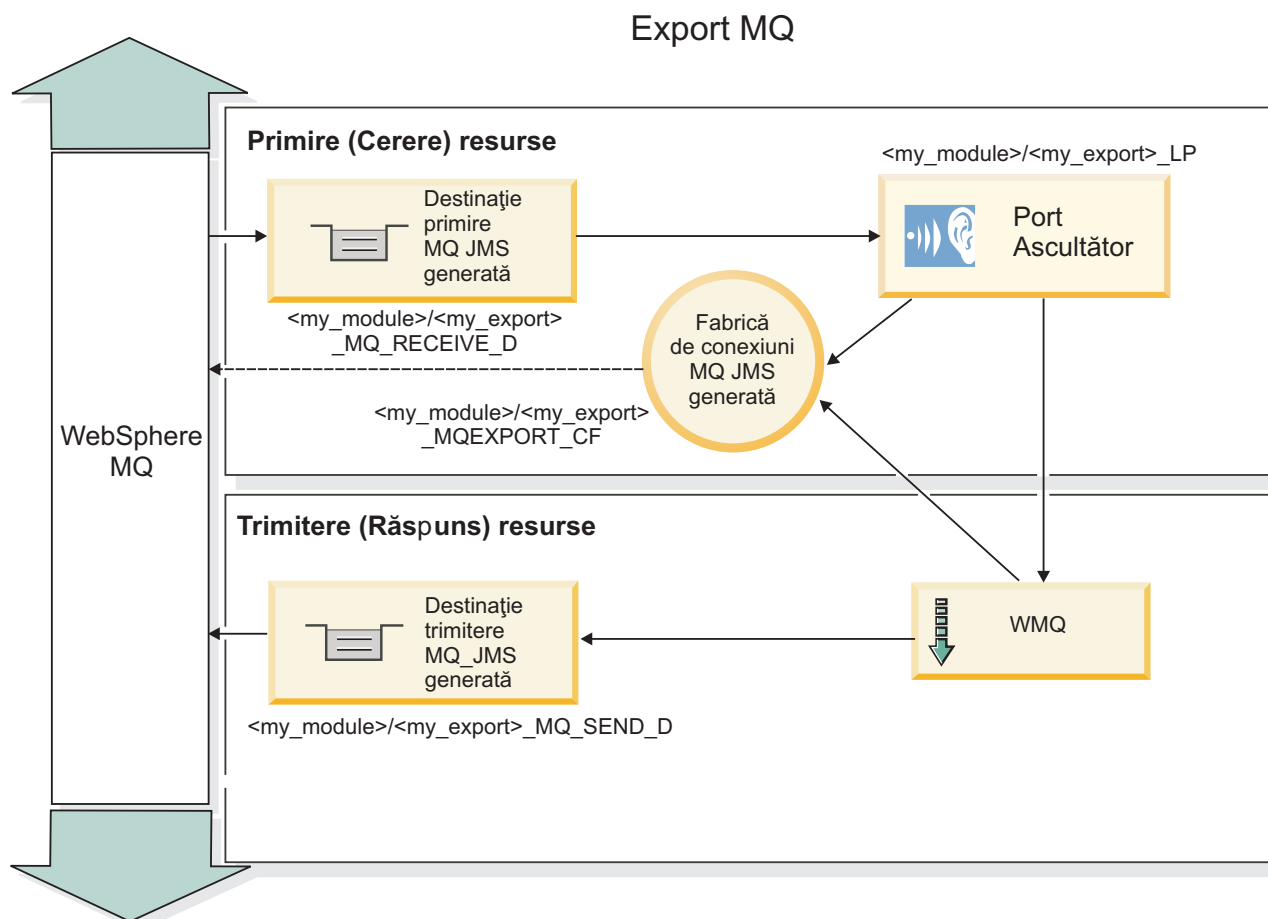


Figura 40. Resurse pentru legarea de export WebSphere MQ

Notă: Figura 39 la pagina 131 și Figura 40 ilustrează modul în care o aplicație dintr-o versiune anterioară a IBM Business Process Manager este legată de un serviciu extern. Pentru aplicațiile dezvoltate pentru IBM Business Process Manager Versiunea 7, se folosește Specificare activare în loc de Port ascultător și Fabrică de conexiuni.

Caracteristici cheie ale unei legări WebSphere MQ:

Caracteristicile cheie ale unei legări WebSphere MQ includ anteturi, artefacte Java EE, și resurse create Java EE.

Scheme de corelare

O aplicație 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 și CorrelID. În marea majoritate a cazurilor, solicitantul lasă

managerul de cozi să selecteze un MessageID și așteaptă ca aplicația de răspuns să copieze aceasta în CorrelID al răspunsului. În majoritatea cazurilor, solicitantul și aplicația de răspuns știu implicit ce tehnică de corelare este în uz. Ocazional aplicația de răspuns va onora diverse stegulețe în câmpulRaport al cererii ce descriu cum vor fi tratate aceste câmpuri.

Legările de export pentru mesajele WebSphere MQ pot fi configurate cu următoarele opțiuni:

Opțiuni 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 opțiuni de raport

Inspectează câmpul Raport al MQMD din cerere pentru o sugestie despre cum să fie tratat MsgId. Opțiunile MQRO_NEW_MSG_ID și MQRO_PASS_MSG_ID sunt suportate și se comportă ca New MsgId și Copy from Request MsgID.

Opțiuni CorrelId răspuns:

Copiere de la MsgID cerere

Copiază câmpul CorrelId de la câmpul MsgId din cerere (implicit).

Copiere de la CorrelID cerere

Copiază câmpul CorrelId de la câmpul CorrelId din cerere.

Copiere de la mesaj SCA

Setează CorrelId pentru a fi transportat în anteturile WebSphere MQ din mesajele de răspuns SCA sau îl lasă gol dacă valoarea nu există.

Ca opțiuni de raport

Inspectează câmpul Raport al MQMD din cerere pentru o sugestie despre cum să fie tratat CorrelId. Opțiunile MQRO_COPY_MSG_ID_TO_CORREL_ID și MQRO_PASS_CORREL_ID sunt suportate și se comportă ca Copy fromRequest MsgID și Copy from Request CorrelID.

Legările de import pentru mesajele WebSphere MQ pot fi configurate cu următoarele opțiuni:

Opțiuni MsgId cerere:

MsgID nou

Permite managerului cozii să selecteze un MsgId unic pentru cerere (implicit).

Copiere de la mesaj SCA

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.

Opțiuni corelare răspuns:

Răspunsul are CorrelID copiat de la MsgId

Se așteaptă ca mesajul de răspuns să aibă un câmp CorrelId setat, per MsgId al cererii (implicit).

Răspunsul are MsgID copiat de la MsgId

Se așteaptă ca mesajul de răspuns să aibă un câmp MsgId setat, per MsgId al cererii.

Răspunsul are CorrelID copiat de la CorrelId

Se așteaptă 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.

Parametrii

Fabrica de conexiuni MQ

Utilizată de clienți pentru a crea o conexiune la furnizorul WebSphere MQ.

Fabrica de conexiuni răspuns

Utilizată de runtime-ul SCA MQ atunci când destinația de trimitere este pe un Manager de coadă diferit ca destinația de recepționare.

Specificații de activare

O specificație de activare MQ JMS este asociată cu una sau mai multe bean-uri bazate pe mesaje și furnizează configurația necesară pentru a recepționa mesajele.

Destinații

- Destinație de trimitere: unde cererea sau mesajul de ieșire este trimis (import); unde mesajul de răspuns va fi trimis (export), dacă nu este permiat de câmpul de antet MQMD ReplyTo din mesajul de intrare.
- Destinație de recepționare: unde răspunsul/ceerea sau mesajul de intrare ar trebui plasat.

Anteturi WebSphere MQ:

Anteturile WebSphere MQ încorporează anumite convenții pentru conversie la mesaje SCA (Service Component Architecture).

Mesajele WebSphere MQ conțin un antet sistem (MQMD), zero sau mai multe alte anteturi MQ (sistem sau personalizate) și un corp mesaj. Dacă există mai multe anteturi în mesaj, ordinea anteturilor este semnificativă.

Fiecare antet conține informații 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 parse 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 octeți neparsați.

Pentru alte anteturi MQ, puteți scrie legări de date antet MQ pentru a parse 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 conține un element valoare de orice tip. Conține MQMD, MQControl (informații de control corp mesaj MQ) și o listă de alte anteturi MQ.

- MQMD reprezintă conținuturile descrierii mesajului WebSphere MQ, cu excepția informațiilor care determină structura și codarea corpului.
- MQControl conține informații care determină structura și codarea unui corp de mesaj.
- MQHeaders conține o listă de obiecte MQHeader.

Lanțul antet MQ este desfăcut astfel încât, în interiorul SMO, fiecare antet MQ are propriile sale informații de control (CCSID, Codare și Format). Anteturile pot fi adăugate sau șterse cu ușurință, fără a modifica alte date ale antetului.

Setare câmpuri în MQMD

Puteți 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 ieșire:

- Codare
- CodedCharacterSet
- Format
- Raport
- Expirare
- Reacție
- Prioritate
- Persistență
- CorrelId
- MsgFlags

Configurați legarea MQ pe un Import sau Export pentru a propaga următoarele proprietăți la mesajul MQ de ieșire:

MsgID

Setare **ID mesaj de cerere** la copiere din mesaj SCA.

MsgType

Curățați caseta de bifare **Setare tip mesaj la MQMT_DATAGRAM sau MQMT_REQUEST pentru operații cerere-răspuns**.

ReplyToQ

Curățați caseta de bifare **Înlocuire răspuns la coadă de mesaje de cerere**.

ReplyToQMGr

Curățați 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 definiția destinației JNDI. Setati proprietatea personalizată MDCTX cu valoarea SET_IDENTITY_CONTEXT pe destinația de trimitere pentru a propaga câmpurile următoare la mesajul MQ de ieșire:

- UserIdentifier
- AppIdentityData

Setati proprietatea personalizată MDCTX cu valoarea SET_ALL_CONTEXT pe destinația de trimitere pentru a propaga proprietățile următoare la mesajul MQ de ieșire:

- UserIdentifier
- AppIdentityData
- PutApplType
- PutApplName
- ApplOriginData

Unele câmpuri nu sunt la mesajul MQ de ieșire. Câmpurile următoare sunt înlocuite în timpul trimerii mesajului:

- BackoutCount
- AccountingToken
- PutDate
- PutTime
- Offset
- OriginalLength

Adăugarea statică a MQCIH într-o legare WebSphere MQ:

IBM Business Process Manager suportă adăugarea informațiilor de antet MQCIH static, fără folosirea unui modul de mediere.

Există diferite moduri de a adăuga informații de antet MQCIH unui mesaj (de exemplu, folosind primitiva de mediere Header Setter). Poate fi util să adăugați aceste informații de antet static, fără folosirea unui modul de mediere suplimentar. Informații de antet statice, inclusiv numele de program CICS, ID-ul tranzacției și alte detalii de antet de format de date, pot fi definite ca și parte a legării WebSphere MQ.

WebSphere MQ, MQ CICS Bridge și CICS trebuie să fie configurat pentru ca informațiile de antet MQCIH să fie adăugate static.

Puteți folosi Integration Designer pentru a configura importul WebSphere MQ cu valori statice care sunt necesare pentru informațiile de antet MQCIH.

Atunci când un mesaj ajunge și este procesat de către importul WebSphere MQ, este efectuată o verificare pentru vedeți dacă informația 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 și valorile statice definite în importul WebSphere MQ sunt adăugate.

Valorile statice definite în importul WebSphere MQ sunt specifice unei metode. Puteți specifica valori statice MQCIH diferite pentru diferite metode din același import WebSphere MQ.

Această facilitate nu este folosită pentru a furniza valori implicite dacă MQCIH nu conține informații de antet specifice deoarece o valoare statică definită în importul WebSphere MQ va înlocui o valoare corespunzătoare furnizată în mesajul de intrare.

Clienți externi:

IBM Business Process Manager poate trimite mesaje către, sau să primească mesaje de la, clienți externi folosind legări WebSphere MQ.

Un client extern (de exemplu, un portal web sau un sistem de informații pentru întreprindere) poate trimite un mesaj către o componentă SCA din aplicație prin intermediul unui export sau poate fi invocat de o componentă SCA din aplicație 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 destinația receive specificată în legarea de export. Destinația specificată în câmpul trimite este folosită pentru a trimite răspunsul către cererea de intrare în cazul în care aplicația invocată oferă un răspuns. Cu toate acestea, un client extern poate invoca aplicații prin intermediul legării de export.

WebSphere MQ importă legarea la, și poate trimite un mesaj către, clienții externi. Acest mesaj ar putea sau nu să solicite un răspuns de la clientul extern.

Informații suplimentare despre modul de interacționare cu clienții externi folosind WebSphere MQ pot fi găsite în Centrul de informare WebSphere MQ.

Depanarea legărilor WebSphere MQ:

Puteți diagnostica și rezolva condițiile de eșec ce apar în legările WebSphere MQ.

Condiții de eșec primar

Condițiile de eșec primar ale legărilor WebSphere MQ sunt determinate de către sensurile tranzacționale, de către configurația WebSphere MQ sau prin referire la comportamente existente în alte componente. Condițiile de eșec primar includ:

- Eșec în conexiunea la managerul de coadă sau la coada WebSphere MQ.
Un eșec la conectarea la WebSphere MQ pentru a primi mesaje va avea ca rezultat eșecul în pornirea portului MDB Listener. Această condiție 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 eșec la conectarea la WebSphere MQ pentru a trimite mesaje de ieșire va duce la derularea înapoi a operației de control a trimiterii.
- Eșecul în analiza unui mesaj de intrare sau în construirea unui mesaj de ieșire.
Un eșec la legarea de date sau la handler-ul de date cauzează derularea înapoi a tranzacției care controlează lucrul.
- Eșec la trimiterea mesajului de ieșire.
Un eșec la trimiterea unui mesaj cauzează derularea înapoi a tranzacției relevante.
- Multiple sau neașteptate mesaje de răspuns.
Importul așteaptă numai un mesaj de răspuns pentru fiecare mesaj de cerere. Dacă mai mult de un răspuns sosește, sau dacă un răspuns întârziat (unul pentru care expirarea răspunsului SCA are loc) sosește, o excepție Service Runtime Exception este aruncată. Tranzacția este repetată, și mesajul de răspuns este retras din coadă sau este manevrat de către managerul de eveniment.

Utilizarea greșită a scenariilor: comparație cu legările WebSphere MQ JMS

WebSphere MQ import și export sunt în principal proiectate să interacționeze cu aplicațiile native WebSphere MQ și să expună întregul conținut al corpului mesajului WebSphere MQ la medieri. Legările WebSphere MQ JMS sunt proiectate să interacționeze cu aplicațiile JMS implementate în comparație cu WebSphere MQ, care expune mesajele în concordanță cu modelul de mesaje JMS.

Următoarele scenarii ar trebui construite folosind legarea WebSphere MQ JMS, și 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.
- Permișiunea modulului SCA de a fi apelat dintr-un servlet componentă Java EE sau EJB prin intermediul căii JMS. Utilizați un export WebSphere MQ JMS.
- Medierea conținutului unui JMS MapMessage, la traversare de-a lungul WebSphere MQ. Utilizați un export și import WebSphere MQ JMS împreună cu legarea de date potrivită.

Există situații în care legarea WebSphere MQ și legarea WebSphere MQ JMS se așteaptă să interacționeze. În particular, când se face o punte între aplicații Java EE și non-Java EE WebSphere MQ, utilizați export WebSphere MQ și import WebSphere MQ JMS (sau invers) în concordanță cu legările de date corespunzătoare sau modulele de mediere potrivite (sau ambele).

Mesaje netrimise

Dacă WebSphere MQ nu poate livra un mesaj la destinația sa intenționată (de exemplu, din cauza erorilor de configurație), 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 conține motivele de eșec, destinația originală și alte informații.

Mesajele SCA bazate pe MQ nu apar în managerul de evenimente eşuate

Dacă mesajele SCA își au originea într-un eșec de interacțiune WebSphere, ar trebui să vă așteptați să găsiți aceste mesaje în managerul de evenimente eşuate. Dacă astfel de mesaje nu apar în managerul de evenimente eşuate, asigurați-vă că destinația 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 interacțiunea cu managerul de evenimente eşuate în timpul invocarilor SCA pentru legările WebSphere MQ.

Evenimentele eşuate MQ sunt reluate pe managerul de coadă greșit

Atunci când o fabrică de conexiuni predefinită trebuie să fie utilizată pentru conexiuni de ieșire, proprietățile de conexiune trebuie să se potrivească cu acelea definite în specificația de activare utilizată pentru conexiunile de intrare.

Fabrica de conexiuni predefinită este utilizată pentru a crea o conexiune la reluarea unui eveniment eşuat și, prin urmare, trebuie configurată pentru a utiliza același manager de coadă ca cel din care a fost primit inițial mesajul.

Tratarea excepțiilor:

Modul în care este configurată legarea determină cum sunt tratate excepțiile 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 excepție.

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 excepție este determinat de cum implementați handler-ul de date sau legarea de date. Comportamentul recomandat este să vă proiectați legarea de date să arunce o

DataBindingException.

Situația este similiară pentru un handler de date. Din moment ce handler-ul de date este invocat de legarea de date, orice excepție a handler-ului de date este înfășurată într-o excepție a legării de date. Prin urmare o

DataHandlerException vă este raportată ca o **DataBindingException.**

Când orice excepție runtime, inclusiv o excepție **DataBindingException**, este aruncată:

- Dacă fluxul de mediere este configurat să fie tranzacțional, mesajul JMS este memorat în Managerul de evenimente eşuate implicit pentru reluare sau ștergere manuală.

Notă: Puteți 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 tranzacțional, excepția este înregistrată în istoric și mesajul este pierdut.

Situația este similiară pentru un handler de date. Deoarece handler-ul de date este apelat de legarea de date, o excepție a handler-ului de date este produsă înăuntrul unei excepții 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 menționate aici.

Limitările legării MQ:

Legarea MQ are unele limitări în utilizarea sa. Acestea sunt menționate aici.

Distribuția mesajelor din abonamentele nepublicate

Metoda de abonare fără publicare a mesajelor distribuite nu este suportată în prezent de legarea MQ prin intermediul propriului WMQ. Totuși, legarea MQ JMS nu suportă această metodă de distribuție.

Cozi de primire partajate

Mai multe legări de export și import WebSphere MQ așteaptă ca orice mesaj prezent în coada lor configurată pentru recepție să fie intenționate pentru acel export sau import. Legările pentru importuri și exporturi ar trebui să fie configurate ținând cont de următoarele considerente:

- Fiecare import MQ trebuie să aibă o coadă diferită de recepționare deoarece legarea de import MQ presupune că toate mesajele din coada de recepționare sunt răspunsuri la cererile pe care le trimite. În cazul în care coada de recepționare este partajată între mai multe importuri, răspunsurile ar putea fi recepționate de un import greșit și nu vor putea fi corelate cu mesajul de cerere inițial.
- Fiecare export MQ ar trebui să aibă o coadă diferită de recepționare, deoarece altfel nu se poate anticipa care dintre exporturi va primi oricare dintre mesajele de cerere individuale.
- Importurile și exporturile MQ pot face referire către aceeași coadă de trimitere.

Limitările legărilor JMS, MQ JMS și Generic JMS:

Legările JMS și MQ JMS au unele limitări.

Implicațiile generării legărilor implicite

Limitările utilizării legărilor JMS, MQ JMS și JMS sunt discutate în următoarele secțiuni:

- Implicațiile generării legărilor implicite
- Schema de corelare a răspunsurilor
- Suport bidirecțional

Atunci când generați o legare, vor fi completate în mod implicit mai multe câmpuri în cazul în care nu alegeți să introduceți singur valorile. De exemplu, numele unui fabrici de conexiuni va fi creat pentru dvs. Dacă știți că veți pune aplicația dumneavoastră pe un server și o veți accesa de la distanță cu un client, ar trebui ca în momentul creării legării să introduceți numele JNDI în locul celor implicite, deoarece probabil veți dori să controlați aceste valori prin intermediul consolei administrative în momentul rulării.

Totuși, dacă ați acceptat valorile implicite, iar apoi ați aflat mai târziu că nu vă puteți accesa aplicația de la un client aflat la distanță, aveți posibilitatea să utilizați consola administrativă pentru a seta în mod explicit valoarea fabricii de conexiuni. Localizați câmpul pentru punctele finale care aparțin de furnizor în setările fabricii de conexiuni și adăugați valoarea în forma <nume_server>:7276 (dacă se utilizează numărul de port implicit).

Schema de corelare a răspunsurilor

În cazul în care folosiți schema de corelare a răspunsurilor CorrelationId To CorrelationId, care este folosită pentru a corela mesaje într-o operație de tip răspuns-răspuns, trebuie să aveți 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, adăugați o primitivă de mediere Mapare înainte de importul cu legarea JMS. Deschideți editorul de mapare. Anteturile arhitecturii componente serviciului cunoscut vor fi disponibile în mesajul țintă. Trageți un câmp care conține un ID unic în mesajul sursă în ID-ul de corelare din antetul JMS din mesajul țintă.

Suportul bidirecțional

Doar caracterele ASCII sunt acceptate pentru numele JNDI (Java Naming and Directory Interface) la momentul execuției.

Cozi de primire partajate

Mai multe legări de export și import așteaptă ca orice mesaj prezent în coada lor configurată pentru recepție să fie intenționate pentru acel export sau import. Legările pentru importuri și exporturi ar trebui să fie configurate ținând cont de următoarele considerente:

- Fiecare legare pentru import trebuie să aibă o coadă diferită de recepționare deoarece aceasta presupune că toate mesajele din coada de recepționare sunt răspunsuri la cererile pe care le trimite. În cazul în care coada de recepționare este partajată între mai multe importuri, răspunsurile ar putea fi recepționate de un import greșit și nu vor putea fi corelate cu mesajul de cerere inițial.
- 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 și exporturile pot face referire la aceeași coadă de trimitere.

Obiectele business

Industria de software de calculator a dezvoltat mai multe modele de programare și cadre de lucru în care *obiectele business* furnizează o reprezentare naturală a datelor de afaceri pentru procesarea aplicației.

În general, aceste obiecte business:

- Sunt definite cu ajutorul standardelor din industrie
- Mapează transparent date la tabele de baze de date sau sisteme de informații de întreprindere
- Suportă protocoale de invocare la distanță
- Furnizează fundația modelului de programare de date pentru programare aplicație

Process Designer and Integration Designer le furnizează dezvoltatorilor un astfel de model de obiect business comun pentru reprezentarea diferitelor tipuri de entități de afaceri din diferite domenii.

În Process Designer, obiectele business sunt concentrate pe reprezentarea tipului de date. Obiectele business de bază (tipurile de variabile) sunt furnizate în seturile de unelte ale sistemului și puteți crea tipuri de variabile personalizate numite obiecte business personalizate. Consultați *Obiecte business și variabile*.

În Integration Designer, care este disponibil doar cu IBM Business Process Manager Advanced, obiectele business pot reprezenta mai multe construcții XSD. În Integration Designer, obiectele business seamănă foarte bine cu schemele XML. Pentru informațiile de luat în considerare atunci când integrați obiecte business objects care sunt definite în Integration Designer cu obiecte business care sunt definite în Process Designer, consultați *Oglindire bibliotecă și construcții XML care nu sunt suportate*.

La momentul dezvoltării Integration Designer, modelul obiectului business permite dezvoltatorilor să definească obiecte ca definiții de schemă XML. La momentul rulării, datele de afaceri care sunt definite prin definițiile din schema XML sunt reprezentate sub formă de Java Business Objects. În acest model, obiectele business sunt bazate pe ciorne anterioare ale specificației SDO (Service Data Object) și furnizează setul complet de interfețe aplicație model de programare necesare pentru manipularea datelor operaționale. Următoarele subsubiecte descriu mai departe modul în care puteți utiliza obiectele business împreună cu Integration Designer.

Definirea obiectelor business

Definiți obiecte business utilizând editorul de obiecte business din Integration Designer. Editorul de obiecte business memorează obiectele business ca definiții schemă XML.

Utilizarea schemei XML pentru a defini obiecte business furnizează mai multe avantaje:

- Schema XML furnizează un model de definiție de date bazat pe standarde și o fundație pentru interoperabilitatea dintre aplicații și sisteme eterogene incompatibile. Schemele XML sunt utilizate împreună cu WSDL (Web Services Description Language) pentru a furniza contracte de interfață bazate pe standarde de-a lungul componentelor, aplicațiilor și sistemelor.

- Schemele XML definesc un model de definiție date bogat pentru reprezentarea datelor operaționale. Acest model include tipuri complexe, tipuri simple, tipuri definite de utilizator, moștenire tip și cardinalitate printre alte caracteristici.
- Obiectele business pot fi definite de interfețe operaționale și date definite din Web Services Description Language, precum și de schema XML din organizații de standarde industriale sau din alte sisteme și aplicații. Integration Designer poate importa aceste obiecte business direct.

Integration Designer furnizează de asemenea suport pentru descoperirea datelor operaționale din baze de date și sisteme de informații de întreprindere și generarea definiției obiectului business al schemei XML bazată pe standarde a acelor date operaționale. Obiectele business generate în acest fel sunt adesea adresate ca *obiecte business specifice aplicației* deoarece imită structura datelor operaționale definite în sistemul de informații de întreprindere.

Când un proces manipulează date din mai multe sisteme de informații diferite, poate fi prețios să transformăm reprezentarea dispartă a datelor operaționale (de exemplu, CustomerEIS1 și CustomerEIS2 sau OrderEIS1 și OrderEIS2) într-o singură reprezentare canonică (de exemplu, Customer sau Order). Reprezentarea canonică este adesea adresată ca *obiectul business generic*.

Definițiile de obiecte business, în special pentru obiecte business generice, sunt utilizate frecvent de mai multe aplicații. Pentru a susține această reutilizare, Integration Designer permite ca obiectele business să fie create în biblioteci care pot fi apoi asociate cu module de aplicații multiple.

Web Services Description Language (WSDL) definește contractele pentru serviciile furnizate și consumate de un modul de aplicații SCA (Service Component Architecture) precum și contractele utilizate pentru a crea componentele dintr-un modul de aplicații. Într-un contract, un WSDL poate reprezenta atât operații, cât și obiecte business (care sunt definite de scheme XML pentru a reprezenta datele operaționale).

Lucrul cu obiecte business

SCA (Service Component Architecture) furnizează cadrul de lucru pentru definirea unui modul de aplicații, serviciile pe care le furnizează, serviciile pe care le consumă și alcătuirea componentelor care furnizează logica operațională a modulului de aplicații. Obiectele business joacă un rol important în aplicație, definind datele operaționale care sunt utilizate pentru a descrie contractele componentei și datele operaționale pe care le manipulează componentele.

Diagrama următoare descrie un modul de aplicații SCA și ilustrează multe dintre locurile în care dezvoltatorul lucrează cu obiecte business.

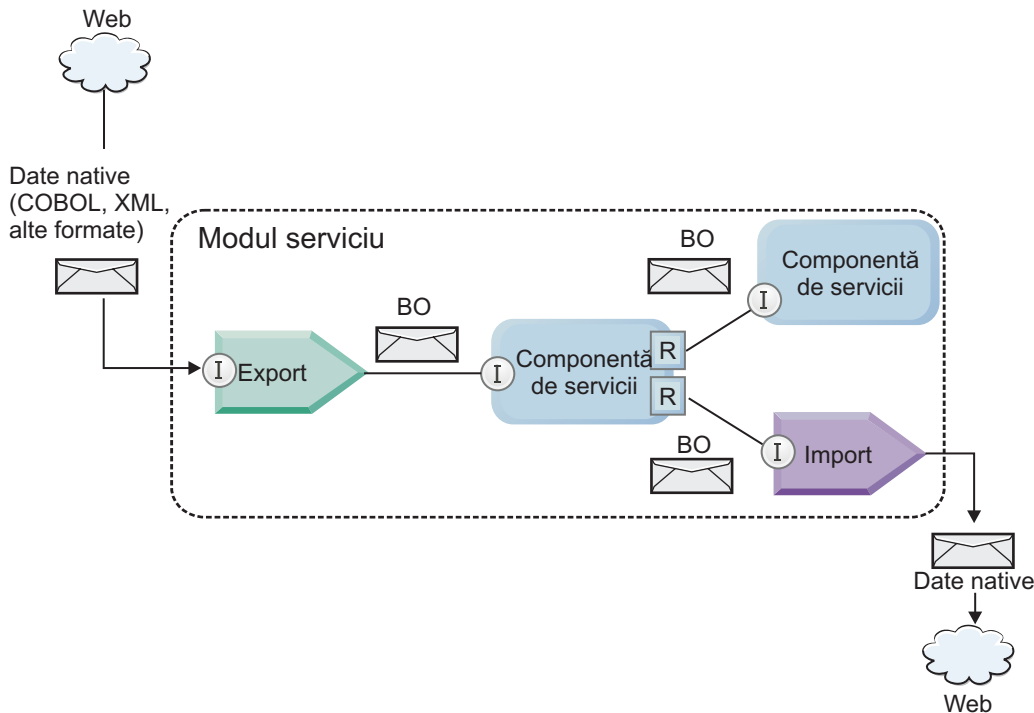


Figura 41. Obiectele business reprezintă datele care curg între servicii într-o aplicație

Notă: Acest subiect descrie cum sunt utilizate obiectele business de către modulele de aplicații SCA. Dacă utilizați interfețe Java, modulele de aplicații SCA pot procesa de asemenea obiecte Java.

Model de programare obiect business

Modelul de programare obiect business conține un set de interfețe Java care reprezintă:

- Definiția obiectului business și date instanță
- Un set de servicii care suportă operațiile de pe obiectele business

Definițiile tipului de obiect business sunt reprezentate de interfețele `commonj.sdo.Type` și `commonj.sdo.Property`. Modelul de programare obiect business furnizează un set de reguli pentru maparea informațiilor de tip complex ale schemei XML la interfața `Tip` și fiecare dintre elementele din definiția tipului complex la interfața `Proprietate`.

Instanțele obiectului business sunt reprezentate de interfața `commonj.sdo.DataObject`. Modelul de programare obiect business nu are tip, ceea ce înseamnă că aceeași interfață `commonj.sdo.DataObject` poate fi utilizată pentru a reprezenta diferite definiții de obiect business, cum ar fi `Client` sau `Comandă`. Definiția căror proprietăți pot fi setate și extrase din fiecare obiect business este determinată de informațiile de tip definite în schema XML asociată cu fiecare obiect business.

Comportamentul modelului de programare obiect business este bazat pe specificația `Service Data Object 2.1`. Pentru informații suplimentare, consultați `SDO 2.1` pentru specificațiile Java, îndrumare și `javadocs` pe web:

<http://www.oasis-open.org/oca/specs/2005/06/sdo/>

Serviciile obiectului business suportă diverse operații de ciclu de viață (cum ar fi creare, egalitate, parsare și serializare) pe obiecte business.

Pentru lucruri specifice despre modelul de programare al obiectului business, vedeți `Programare utilizând servicii de obiecte business` și `Documentație API și SPI generată despre obiecte business`.

Legări, legări de date și handler-e de date

După cum a fost afișat în Figura 41 la pagina 142, datele operaționale care sunt utilizate pentru a invoca servicii furnizate de module de aplicații SCA sunt transformate în obiecte business, astfel încât componentele SCA să poată manipula datele operaționale. 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 și 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 convertește formatele nenative în obiecte business reprezentate de interfața DataObject.

Pentru informații suplimentare despre dezvoltarea legărilor de date și handler-elor de date, consultați “Handler-e de date” la pagina 54 și “Legări de date” la pagina 55.

Componente

Componentele SCA definesc contractele lor de servicii de provizionare și consum utilizând o combinație de Web Services Description Language și schemă XML. Datele operaționale pe care SCA le transmite între componente sunt reprezentate ca obiecte business utilizând interfața DataObject. SCA verifică dacă aceste tipuri de obiecte business sunt compatibile cu contractul interfeței definit de componenta care va fi invocată.

Teoriile modelului de programare pentru manipularea obiectelor business variază de la componentă la componentă. Componenta POJO și primitiva personalizată a componentei flux de mediere furnizează manipulare directă a obiectelor business prin activarea programării Java direct utilizând interfețele și 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 interfețele și serviciile obiectului business.

Obiectele business pot fi transformate utilizând fie combinația componentei Interface Flow Mediation și Business Object Map, fie componenta flux de mediere și primitiva sa XML Map. Aceste capacități de transformare obiecte business sunt utile pentru convertirea obiectelor business specifice aplicației la și de la obiecte business generice.

Obiectele business speciale

Obiectele de mesaje de servicii și graficele operaționale sunt două tipuri specializate de obiecte business care sunt utilizate în scopuri de aplicații 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 colecția de date asociată cu o invocare de servicii.

Un SMO are o structură de nivel înalt corectată alcătuită din anteturi, context, corp și atașamente (dacă sunt prezente).

- Anteturile transmit informații înrudite cu invocarea de servicii peste un anumit protocol sau legare. Exemple sunt anteturile SOAP și anteturile JMS.
- Datele de context transmit informații logice suplimentare asociate cu invocarea în timp ce sunt procesate de componenta fluxului de mediere. Aceste informații nu fac parte în mod tipic din datele de aplicație trimise sau recepționate de clienți.
- Corpul SMO-ului transmite datele operaționale ale datelor utile, care reprezintă mesajul de aplicație nucleu sau datele de invocare sub forma unui obiect business standard.

SMO poate transporta de asemenea date în atașament pentru invocările serviciilor web folosind SOAP cu atașamente.

Fluxurile de mediere realizează astfel de taskuri precum dirijare cerere și transformare de date, și SMO-ul furnizează vizualizarea combinată de conținut antet și date utile (payload) într-o singură structură unificată.

Grafic operațional

Un grafic operațional este un obiect business special utilizat pentru a furniza suport pentru sincronizare date în scenarii de integrare.

Considerați un exemplu în care două sisteme de informații 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 operaționale suportă noțiunea de trimitere doar a porțiunii de comandă care s-a modificat la celălalt sistem și adnotarea ei cu informații de modificare sumar pentru a defini tipul de modificare.

În acest exemplu, un Grafic operațional de comenzi va transmite celuilalt sistem că unul dintre articolele de linie din comandă a fost șters și că proprietatea dată de livrare proiectată a comenzii a fost actualizată.

Graficele operaționale pot fi adăugate cu ușurință la obiecte business existente din Integration Designer. Sunt găsite de cele mai multe ori în scenarii în care sunt utilizate adaptoarele WebSphere și pentru a ajuta migrarea aplicațiilor WebSphere InterChange Server.

Modul de parsare a obiectelor business

Integration Designer furnizează o proprietate pe module și biblioteci pe care le puteți vedea pentru a configura modul de parsare XML pentru obiecte business fie la vioi, fie la leneș.

- Dacă opțiunea este setată la *vioi*, fluxurile de octeți XML sunt parsate cu nerăbdare pentru a crea obiectul business.
- Dacă opțiunea este setată la *leneș*, obiectul business este creat normal, dar parsarea reală a fluxului de octeți XML este amânată și parsată parțial doar când sunt accesate proprietățile obiectului business.

În oricare dintre modurile de parsare XML, datele non-XML sunt întotdeauna parsate vioi pentru a crea obiectul business.

Considerații 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). Puteți modifica modul de parsare a modulului sau bibliotecii, totuși ar trebui să fiți conștient de implicații.

Modul de parsare a obiectelor business este setat la nivelul modulului sau al bibliotecii. Module 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 și bibliotecile care sunt create în IBM Integration Designer versiunea 7 și ulterioare vor primi cel mai adecvat mod de parsare în funcție de numărul de factori, cum ar fi modul de parsare al proiectelor existente în spațiul dumneavoastră de lucru sau modul de parsare al proiectelor dependente sau a altor proiecte din aceeași soluție și așa mai departe. Puteți modifica modul de parsare de obiect business al unui mod sau bibliotecă pentru a se potrivi cu implementarea dumneavoastră, totuși ar trebui să fiți conștienți de următoarele considerente.

Considerente

- Modul de parsare obiect business leneș procesează datele XML mai repede; totuși există diferențe de compatibilitate între modul vioi și modul leneș de care trebuie să fiți conștient înainte de modificarea configurației unui modul sau a unei biblioteci. Aceste diferențe vor afecta comportamentul din timpul rulării modulelor. Pentru informații despre modul optim de parsare pentru aplicația dumneavoastră, vedeți "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 pentru suportul, fie a 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, și de un modul utilizând modul de parsare leneș. Modul de parsare al unei biblioteci la momentul rulării este determinat de modulele care fac referire la bibliotecă. La momentul execuției, un modul își declară modul de parsare, iar acel mod de parsare este utilizat de modul și de orice bibliotecă folosită de modul.
- Modulele și bibliotecile care sunt configurate pentru moduri de parsare diferite sunt compatibile în următoarele cazuri:

- Modulele și bibliotecile configurate cu modul de parsare leneș sunt compatibile cu bibliotecile care utilizează fie modul de parsare leneș, fie ambele moduri de parsare, leneș și vioi.
 - Modulele și bibliotecile configurate cu modul de parsare vioi sunt compatibile cu bibliotecile care utilizează fie modul de parsare vioi, fie ambele moduri de parsare, leneș și vioi.
 - Bibliotecile configurate cu modurile de parsare leneș și vioi sunt compatibile doar cu bibliotecile care utilizează ambele moduri de parsare, leneș și vioi.
- Utilizați același mod de parsare pentru module interactive care comunică utilizând legarea SCA. Dacă modulele comunică utilizând diferite moduri de parsare, ar putea rezulta probleme de performanță.

Concepte înrudite:

“Beneficii ale utilizării modului de parsare leneș față de vioi”

Unele aplicații beneficiază de modul leneș de parsare XML în timp ce altele văd o performanță îmbunătățită cu mod de parsare vioi. Este recomandat să măsurați aplicația dumneavoastră în ambele moduri de parsare pentru a determina ce mod se potrivește cel mai bine caracteristicilor specifice ale aplicației dumneavoastră.

Beneficii ale utilizării modului de parsare leneș față de vioi:

Unele aplicații beneficiază de modul leneș de parsare XML în timp ce altele văd o performanță îmbunătățită cu mod de parsare vioi. Este recomandat să măsurați aplicația dumneavoastră în ambele moduri de parsare pentru a determina ce mod se potrivește cel mai bine caracteristicilor specifice ale aplicației dumneavoastră.

Este posibil ca aplicațiile care parsează fluxuri de date XML mari să vadă îmbunătățiri de performanță când este utilizat modul de parsare XML leneș. Beneficiile performanței cresc pe măsură ce dimensiunea șirului de octeți XML creșteși cantitatea de date de la șirul de octeți care este accesat de către aplicație scade.

Este posibil ca aplicațiile următoare să funcționeze mai bine folosind un mod de parsare vioi:

- Aplicații care parsează fluxuri de date non-XML
- Aplicații care folosesc mesajele care sunt create folosind serviciul BOFactory
- Aplicații care parsează mesaje XML foarte mici

Referințe înrudite:

“Considerații 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). Puteți modifica modul de parsare a modulului sau bibliotecii, totuși ar trebui să fiți conștient de implicații.

Considerații de migrare și de dezvoltare de aplicații:

Dacă configurați o aplicație care a fost dezvoltată inițial utilizând un mod de parsare vioi pentru a utiliza acum un mod de parsare leneș, sau dacă plănuiți să comutați o aplicație între modul de parsare leneș și cel vioi, fiți conștient de diferențele dintre moduri și de considerente când comutați moduri.

Tratarea erorilor

Dacă fluxul de octeți XML care este parsat este format greșit, apar excepții de parsare.

- În modul de parsare XML vioi, acele excepții apar de îndată ce este parsat obiectul business din fluxul XML de intrare.
- Dacă este configurat modul de parsare XML leneș, excepțiile de parsare apar cu întârziere când sunt accesate proprietățile obiectului business și este parsată porțiunea XML-ului care este format greșit.

Pentru a vă ocupa de XML format greșit, selectați una dintre următoarele opțiuni:

- Implementarea unei magistrale de servicii de întreprindere pe margini pentru a valida XML de intrare
- Logica de detecție eroare autor în punctul în care sunt accesate proprietățile de obiect business

Mesaje și stive de excepții

Deoarece modurile de parsare XML leneș și vioi au implementări fundamentale diferite, urmele de stivă aruncate de interfețele și serviciile de programare ale obiectului business au același nume de clasă de excepții, dar ar putea să nu conțină același mesaj de excepție sau set înfășurat de clase de excepții specifice implementării.

Format serializare XML

Modul de parsare XML leneș furnizează o optimizare de performanță care încearcă să copieze XML nemodificat de la fluxul de octeți de intrare la fluxul de octeți de ieșire pe serializare. Rezultatul este performanță crescută, dar formatul de serializare al sirului de octeți XML de ieșire poate fi diferit dacă întregul obiect business a fost actualizat în mod de parsare XML leneș sau dacă rulează în mod de parsare XML vioi.

Deși 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 și XML poate fi transmis în siguranță între aplicații care rulează în moduri de parsare diferite cu echivalență semantică.

Validator instanță obiect business

Validatorul instanței modului obiectului business de parsare XML leneș furnizează o validare de fidelitate mai înaltă a obiectelor business, în special validare fațetă de valori proprietate. Din cauza acestor îmbunătățiri, validatorul instanței modului de parsare leneș prinde probleme suplimentare care nu sunt prinse în modul de parsare vioi și furnizează mesaje de eroare mai detaliate.

Mapări XML versiune 602

Fluxurile de mediere dezvoltate inițial înainte de WebSphere Integration Developer Version 6.1 pot conține primitive Mapare care utilizează o hartă sau o foaie de stil care nu poate rula direct în modul de parsare XML mai lent. La migrarea unei aplicații pentru utilizarea în modul leneș de parsare XML, fișierele hartă asociate cu primitivele Mapare pot fi actualizate în mod automat de către vrăjitorul de migrare pentru a rula în noul mod. Totuși, în cazul în care o primitivă Mapare se referă în mod direct la o foaie de stil care a fost editată manual, foaia de stil nu este migrată și nu poate rula în modul XML mai lent.

API-uri nepublicate private

Dacă o aplicație profită de interfețe de programare obiecte business specifice implementării, private, nepublicate, aplicația are toate șansele să eșueze compilarea când este schimbat modul de parsare. În mod de parsare vioi, aceste interfețe 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 aplicație.

API-uri EMF Obiect mesaj serviciu

O componentă de mediere din IBM Integration Designer furnizează abilitatea de a manipula conținut mesaj utilizând clase și interfețe Java furnizate în pachetul `com.ibm.websphere.sibx.smobo`. În mod de parsare XML leneș, interfețele Java din pachetul `com.ibm.websphere.sibx.smobo` pot fi încă utilizate, dar metodele care se referă direct la clasele și interfețele EMF (Eclipse Modeling Framework) sau care sunt moștenite de la interfețele EMF au toate șansele să eșueze.

`ServiceMessageObject` și conținuturile sale nu pot fi atribuite obiectelor EMF în mod de parsare XML leneș.

Serviciu BOMode

Serviciul BOMode este utilizat pentru a determina dacă modul de parsare XML care se execută momentan este vioi sau leneș.

Migrare

Toate aplicațiile 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 aplicații anterioare versiunii 7.0.0.0 să fie configurată pentru a utiliza modul de parsare XML, utilizați întâi Integration Designer pentru a migra artefactele aplicației. După migrare, configurați aplicația pentru a utiliza parsare XML leneșă.

Vedeți Migrare artefacte sursă pentru informații despre migrarea artefactelor din Integration Designer și vedeți Configurarea modului de parsare obiect business al modulelor și bibliotecilor pentru informații despre setarea modului de parsare.

Relații

O relație este o asociere între două sau mai multe entități de date, în special obiecte business. În IBM Business Process Manager Advanced, relațiile pot fi utilizate pentru a transforma date care sunt echivalente peste obiecte business și alte date care sunt reprezentate diferit, sau pe care le pot utiliza pentru a trasa aplicații peste diferite obiecte găsite în aplicații diferite. Acestea pot fi partajate între aplicații, soluții și chiar produse.

Serviciul de relații din IBM Business Process Manager Advanced furnizează infrastructura și operațiile pentru gestionarea relațiilor. Deoarece vă permite să tratați obiecte business indiferent de locul în care acestea se află, acesta vă poate oferi o vizualizare de asamblare unificată a tuturor aplicațiilor dintr-o întreprindere și poate fi folosit ca un bloc de construire pentru soluții BPM. Deoarece relațiile pot fi extinse și gestionate, ele pot fi utilizate în soluții complexe de integrare.

Ce sunt relațiile?

O relație este o asociere între obiecte business. Fiecare obiect business dintr-o relație este numit *participant* la relație. Fiecare participant din relație este distins de alți participanți pe baza funcției sau a *rolului* pe care îl are în acea relație. O relație conține o listă de roluri.

Definiția relației descrie fiecare rol și specifică modul în care sunt înrudite rolurile. De asemenea, aceasta descrie "forma" generală a relației. De exemplu, acest rol poate avea doar un participant, dar acest alt rol poate avea câți participanți sunt necesari. Puteți defini o relație *car-owner*, de exemplu, unde un posesor poate avea mai multe mașini. De exemplu, o instanță poate avea următorii participanți pentru fiecare dintre aceste roluri:

- Mașină (Ferrari)
- Posesor (John)

Definiția relației este un șablon pentru *instanța* relației. Instanța este instanțierea relației în timpul rulării. În exemplul cu *proprietarii de mașini*, o instanță ar putea descrie oricare din următoarele asocieri:

- John deține Ferrari
- Sara deține Mazda
- Bob deține Ferrari

Utilizarea relațiilor vă eliberează de necesitatea de a construi personalizat persistența de urmărire a relației din logica dumneavoastră operațională. Pentru anumite scenarii, serviciul de relații face toată munca în locul dumneavoastră. Vedeți exemplul descris în secțiunea din Relații de identitate.

Scenarii

Aici este un exemplu tipic de situație în care o soluție de integrare poate folosi relații. O corporație mare cumpără mai multe companii sau unități de afaceri. Fiecare unitate operațională folosește software diferit pentru a monitoriza personalul și caietele. Compania are nevoie de o cale de a-și monitoriza angajații și carnetele lor. Aceasta vrea o soluție care le permite:

- Vizualizarea tuturor angajaților din diverse unități de afaceri ca și cum ar fi într-o singură bază de date
- O singură vizualizare a tuturor carnetelor
- Permite angajaților să se logheze în sistem și să cumpere un carnet.
- Acomodarea diferitelor sisteme de aplicații de întreprindere din diverse unități de afaceri

Pentru a realiza acest lucru, compania are nevoie de o cale pentru a se asigura, de exemplu, că John Smith și John A. Smith din aplicații diferite sunt văzuți ca același angajat. De exemplu, aceștia au nevoie de o cale pentru a consolida o singură entitate în mai multe spații ale aplicației.

Scenariile mai complexe de relații implică construirea proceselor BPEL care stabilesc relații între obiecte diferite găsite în mai multe aplicații. Cu scenarii mai complexe de relații, obiectele business se află în soluția de integrare, ci nu în aplicații. Serviciul de relații asigură o platformă pentru gestionarea în mod persistent a relațiilor. Înainte de serviciul de relații, trebuie să construiești propriul serviciu de persistență al obiectului. Două exemple de scenarii complexe de relații sunt:

- Aveți un obiect business **car** cu un număr VIN într-o aplicație SAP și vreți să urmăriți că această mașină este deținută de altcineva. Totuși, relația de posesiune este cu cineva într-o aplicație PeopleSoft. În acest tipar de relații, aveți două soluții și este nevoie să construiești o punte laterală între ele.
- O companie mare de comerț cu amănuntul vrea să fie capabilă să monitorizeze marfa returnată pentru primirea banilor înapoi sau pentru credit. Există două aplicații diferite implicate: OMS (order management system) pentru achiziții și RMS (returns management system) pentru returnări. Obiectele business se află în mai multe aplicații și aveți nevoie de o cale de a afișa relațiile care există între ele.

Modele de utilizare comune

Cele mai comune tipare de relații sunt tiparele *echivalență*. Acestea sunt bazate pe referința încrucișată sau pe corelare. Există două tipuri de relații care se potrivesc cu acest tipar: *non-identitate* și *identitate*.

- **Relații non-identitate** stabilesc asocieri între obiecte business sau alte date pe o bază unu-la-mulți sau mulți-la-mulți. Pentru fiecare instanță a relației, pot exista una sau mai multe instanțe pentru fiecare participant. Un tip de relație non-identitate este o relație de căutare statică. Un astfel de exemplu este o relație în care **CA** dintr-o aplicație SAP este înrudită cu **California** dintr-o aplicație Siebel.

•

Relații de identitate stabilesc asocieri între obiecte business sau alte date pe o bază unu-la-unu. Pentru fiecare instanță a relației, poate exista doar o instanță a fiecărui participant. Relațiile de identitate capturează referințe încrucișate între obiecte business care sunt echivalente semantic, dar care sunt identificate în mod diferit în diferite aplicații. Fiecare participant la relație este asociat cu un obiect business care are o valoare (sau combinație de valori) care identifică obiectul în mod unic. În mod obișnuit, relațiile de identitate transformă atributele cheie ale obiectelor business precum numere ID și coduri de produs.

De exemplu, dacă aveți obiecte business **car** în aplicații SAP, PeopleSoft și Siebel și vreți să construiești o soluție care să le sincronizeze, va fi nevoie, în mod normal, să introduceți logica de sincronizare a relației construite manual în șase mapări:

SAP -> generic

generic -> SAP

PeopleSoft-> generic

generic-> PeopleSoft

Siebel-> generic

generic-> Siebel

Totuși, dacă utilizați relații în soluția dumneavoastră, serviciul de relații asigură implementări de tipare preconstruite care mențin toate aceste relații pentru dumneavoastră.

Unelte pentru lucrul cu relații

Editorul de relații din Integration Designer este unelta pe care o utilizați pentru a modela și proiecta relații și roluri de integrare business. Pentru informații detaliate despre experiența necesară și despre taskuri la crearea relațiilor și utilizarea editorului de relații, vedeți *Creare relații*.

Serviciul de relații este un serviciu de infrastructură din IBM Business Process Manager care menține relații și roluri în sistem și oferă operații pentru gestiunea relației și a rolului.

Managerul de relații este interfața administrativă pentru gestionarea relațiilor. Acesta este accesat prin pagina Manager de relații a consolei administrative.

Relațiile pot fi invocate programatic prin API-urile serviciului de relații.

Serviciu de relații

Serviciul de relații memorează date ale relației în tabele de relații unde păstrează pista valorilor specifice aplicației din aplicații și soluții. Serviciul de relații oferă operații pentru gestiunea relațiilor și a rolului.

Cum funcționează relațiile

Relațiile și rolurile sunt descrise folosind interfața grafică a unelei editor de relații din Integration Designer. Serviciul de relații memorează date de corelare în tabele din baza de date a relației în sursa implicită de date pe care o specificați la configurarea serviciului de relații. O tabelă separată (uneori numită tabelă participantă) memorează informații pentru fiecare participant la relație. Serviciul de relații utilizează aceste tabele ale relației pentru a păstra pista valorilor înrudite specifice aplicației și pentru a propaga informații actualizate în toate soluțiile.

Relațiile, care sunt artefacte business, sunt implementate într-un proiect sau într-o bibliotecă partajată. La prima implementare, serviciul de relații populează datele.

În timpul rulării, când mapările sau alte componente IBM Business Process Manager necesită o instanță de relație, instanțele relației sunt fie actualizate, fie extrase în funcție de scenariu.

Datele instanței de relație sau rol pot fi manipulate prin trei mijloace:

- Invocări componentă IBM Business Process Manager snippet Java ale API-urilor serviciului de relații
- Transformări ale relației în serviciul de mapare a obiectului business IBM Business Process Manager
- Unealta managerului de relații

Pentru informații detaliate despre experiența necesară și despre taskuri la crearea relațiilor, identificarea tipurilor de relații și utilizarea editorului de relații, vedeți subiectul *Creare relații*.

Manager de relații

Managerul de relații este interfața administrativă pentru gestionarea relațiilor. Acesta este accesat prin pagina Manager de relații a consolei administrative.

Managerul de relații asigură o interfață grafică cu utilizatorul pentru crearea și manipularea datelor relației și ale rolului în timpul rulării. Puteți gestiona entități ale relației la toate nivelurile: instanță de relație, instanță de rol și niveluri ale datelor atribut și ale datelor proprietate. Cu managerul de relații, puteți:

- Vizualiza o listă a relațiilor din sistem și a informațiilor detaliate despre relații individuale
- Gestiona instanțe de relații:
 - Interoga date ale relației pentru a vizualiza subseturi de date ale instanței
 - Interoga date ale relației pentru a vizualiza subseturi de date ale instanței utilizând vizualizări ale bazei de date
 - Vizualiza o listă a instanțelor relației care se potrivesc unei interogări a relației și informații detaliate despre o instanță
 - Edita valorile proprietății pentru o instanță a relației

- Crea și șterge instanțe de relație
- Gestionarea rolurilor și instanțelor de roluri:
 - Vizualizarea detaliilor despre un rol sau o instanță de rol
 - Editarea proprietăților instanței de rol
 - Crea și șterge instanțe de rol pentru o relație
 - Derularea înapoi a datelor instanței de relație la un punct în care știți că datele sunt de încredere
- Importarea datelor dintr-o relație statică existentă în sistemul dumneavoastră sau exportarea datelor dintr-o relație statică existentă într-un fișier RI sau CSV
- Înlăturarea schemei și datelor relației din magazie când aplicația care le folosește este dezinstalată

Relații în medii Network Deployment

Relațiile pot fi utilizate în medii ND (Network Deployment) fără nicio configurație suplimentară.

În medii ND (Network Deployment), relațiile sunt instalate într-un cluster de aplicații. Atunci relațiile sunt vizibile în cluster și toate serverele din cluster au acces la datele instanței memorate în baza de date a relației. Abilitatea de rula serviciului de relații într-un mediu ND îl face scalabil și cu disponibilitate bună.

Managerul de relații permite relațiilor să fie gestionate în cluster-e diferite printr-o interfață administrativă centralizată. Conectați managerul de relații la un server dintr-un cluster selectând MBean-ul relației sale.

API-uri ale serviciului de relații

Relațiile pot fi invocate programatic prin API-urile serviciului de relații în sau în afara mapărilor de obiecte business.

Sunt disponibile trei tipuri de API-uri:

- API-uri de manipulare a instanței de relație (inclusiv crearea, actualizarea, ștergerea directă a datelor instanței)
- API-uri suport ale tiparului de relații (inclusiv `correlate()`, `correlateforeignKeyLookup`)
- Tipare de căutare relație (API-uri de căutare)

Magistrala ESB (Enterprise Service Bus) din IBM Business Process Manager

IBM Business Process Manager suportă integrarea serviciilor aplicație, inclusiv aceleași capacități ca WebSphere Enterprise Service Bus.

Conectarea serviciilor printr-o magistrală pentru serviciile întreprinderii

Cu ajutorul unui ESB (Enterprise Service Bus), puteți maximiza flexibilitatea unui SOA. Participanții din interacțiunea cu serviciul sunt conectați mai degrabă la ESB, decât direct unul cu altul.

În cazul în care solicitantul serviciului se conectează la ESB, atunci ESB-ul își asumă răspunderea pentru furnizarea cererilor sale, folosind mesaje, către un furnizor de servicii oferind funcția necesară și calitatea serviciilor. ESB ușurează interacțiunile solicitant-furnizor și adresează protocoalele nepotrivite, tiparele de interacțiune sau capacitățile serviciului. De asemenea, un ESB poate permite sau îmbunătăți monitorizarea și gestiunea. ESB oferă caracteristici de virtualizare și management care pun în aplicare și extind capacitățile de bază pentru SOA.

ESB prezintă pe scurt următoarele caracteristici:

Locație și identitate

Participanții nu trebuie să știe locația sau identitatea altor participanți. De exemplu, solicitanții nu trebuie să fie conștienți de faptul că o cerere ar putea fi deservită de oricare dintre furnizori; furnizorii de servicii pot fi adăugați sau înlăturați fără întrerupere.

Protocol de interacțiune

Participanții nu trebuie să partajeze același protocol de comunicație sau stil de interacțiune. De exemplu, o cerere exprimată ca SOAP peste HTTP poate fi deservită de un furnizor care înțelege doar SOAP peste JMS (Java Message Service).

Interfață

Solicitanții și furnizorii nu trebuie să fie de acord cu o interfață comună. Un ESB împacă diferențele prin transformarea mesajelor de cerere și de răspuns într-o formă așteptată de furnizor.

Calitățile unui serviciu (de interacțiune)

Participanții, sau administratorii de sisteme, își declară cerințele legate de calitatea serviciilor, inclusiv autorizația cererilor, criptarea și decriptarea conținutului mesajelor, auditarea automată a interacțiunilor dintre servicii, și modul în care cererile lor ar trebui să fie rutate (de exemplu optimizarea vitezei sau a costului).

Interpunerea ESB între participanți vă dă posibilitatea să modulați interacțiunea acestora prin intermediul unei construcții logice numite *mediere*. Medierile se aplică mesajelor aflate în zbor între solicitanți și furnizori. De exemplu, medierile pot fi folosite pentru a găsi servicii cu caracteristici specifice cerute de un solicitant, sau pentru a rezolva diferențele interfeței apărute între solicitanți și furnizori. Pentru interacțiuni complexe, medierile pot fi legate secvențial.

Folosind medieri, o magistrală pentru serviciile întreprinderii efectuează următoarele acțiuni între solicitant și serviciu:

- *Rutarea* mesajelor între servicii. O magistrală pentru serviciile întreprinderii oferă o infrastructură de comunicație comună care poate fi folosită pentru a conecta servicii, și, prin urmare funcțiile business pe care acestea le reprezintă, fără a fi nevoie ca programatorii să scrie și să mențină logica complexă de conectivitate.
- *Convertirea* protocoalelor de transport între solicitant și serviciu. O magistrală pentru serviciile de întreprindere oferă o cale consistentă bazată pe standarde de integrare a funcțiilor business care folosesc diferite standarde IT. Acest lucru permite integrarea funcțiilor business care nu au putut comunica în mod normal, cum ar fi conectarea aplicațiilor în silozurile departamentale sau activarea aplicațiilor în companii diferite care participă în interacțiunile serviciului.
- *Transformarea* formatelor mesajelor între solicitant și serviciu. O magistrală pentru serviciile de întreprindere permite funcțiilor business să schimbe informații în formate diferite, iar magistrala se asigură că informațiile livrate funcției business sunt în formatul cerut de acea aplicație.
- *Tratarea* evenimentelor business din surse incompatibile. O magistrală pentru serviciile de întreprindere suportă interacțiuni bazate pe evenimente în plus față de schimbul de mesaje ce tratează cererile serviciului.

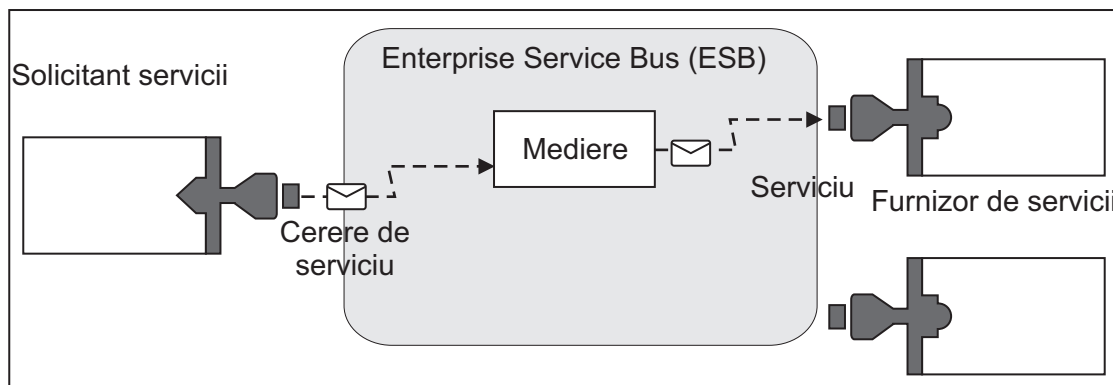


Figura 42. O magistrală pentru serviciile întreprinderii. Magistrală pentru serviciile de întreprindere rutează mesajele între aplicații, care pot fi solicitanți sau furnizori de servicii. Magistrala convertește protocoalele de transport și transformă formatul mesajelor între solicitanți și furnizori. În această figură, fiecare aplicație utilizează un protocol diferit (reprezentate prin formele geometrice diferite ale conectorilor lor) și folosește diferite formate de mesaje.

Folosind magistrala pentru serviciile de întreprindere vă puteți concentra pe afacerea dumneavoastră de bază mai degrabă decât pe sistemele dumneavoastră computer. Puteți modifica sau adăuga la servicii, dacă este necesar; de exemplu, pentru a răspunde la modificările din cerința de afaceri, pentru a adăuga capacitate de serviciu suplimentar sau pentru a adăuga noi capacități. Aveți posibilitatea să efectuați modificările necesare prin reconfigurarea magistralei, cu impact redus sau nul asupra serviciilor și aplicațiilor existente care folosesc magistrala.

Infrastructura de mesagerie ESB (Enterprise Service Bus)

IBM Business Process Manager include capacități ale magistralei ESB. IBM Business Process Manager suportă integrarea tehnologiilor orientate spre servicii, orientate pe mesaj și conduse de eveniment pentru a asigura o infrastructură a mesageriei bazată pe standarde într-o magistrală de servicii de întreprindere integrate.

Capabilitățile serviciului de întreprindere pe care le puteți utiliza pentru aplicațiile dumneavoastră de întreprindere asigură nu doar un nivel de transport dar și suport de mediere pentru a facilita interacțiunile serviciului. Magistrala ESB (Enterprise Service Bus) este construită în jurul standardelor deschise și a SOA (service-oriented architecture). Este bazată pe infrastructura Java EE robustă și serviciile platformei asociate furnizate de IBM WebSphere Application Server Network Deployment.

IBM Business Process Manager este alimentat de aceeași tehnologie disponibilă cu IBM WebSphere Enterprise Service Bus. Această capacitate este parte componentă a funcționalității de bază a IBM Business Process Manager și nu este necesară nicio licență suplimentară pentru WebSphere Enterprise Service Bus pentru a beneficia de aceste capacități.

Totuși, puteți implementa licențe autonome suplimentare ale WebSphere Enterprise Service Bus în jurul întreprinderii dumneavoastră pentru a extinde întinderea conectivității la soluții de integrare a procesului alimentate de IBM Business Process Manager. De exemplu, WebSphere Enterprise Service Bus poate fi instalat mai aproape de o aplicație SAP pentru a găzdui un IBM WebSphere Adapter for SAP și pentru a transforma mesajele SAP înainte de a trimite acele informații în rețea într-un proces operațional condus de IBM Business Process Manager.

Gazde destinație mesagerie sau coadă:

O gazdă destinație mesagerie sau coadă furnizează funcția mesagerie într-un server. Un server devine gazda destinației mesageriei la configurarea lui ca destinație mesagerie.

Un motor de mesagerie rulează într-un server. Motorul mesagerie furnizează funcții mesagerie și un punct de conexiune pentru ca aplicațiile să se conecteze la magistrală. Comunicația asincronă Service Component Architecture (SCA), importurile și exporturile JMS, procesarea internă asincronă utilizează cozi de mesaje în motorul de mesagerie.

Mediul de implementare conectează sursa mesajului la destinația mesajului prin magistrală la implementarea modulelor de aplicații. Cunoașterea sursei mesajului și a destinației mesajului vă ajută să determinați de ce tip de mediu de implementare aveți nevoie.

Aplicațiile 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 fișiere. Motorul mesagerie utilizează o instanță a unei surse de date JDBC pentru a interacționa cu acea bază de date.

Configurați gazda destinație a mesageriei la definirea mediului de implementare utilizând **Server** din consola administrativă sau desemnați serverul ca gazdă destinație 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 reținute în aceeași schemă a bazei de date. Puteți crea fiecare depozit de date într-o bază de date separată. Alternativ, puteți crea mai multe depozite de date în aceeași bază de date, fiecare depozit de date utilizând o schemă diferită.

Un motor de mesagerie utilizează o instanță a unei surse de date JDBC pentru a interacționa cu baza de date care conține depozitul de date pentru acel motor de mesagerie.

Furnizori JDBC:

Puteți folosi furnizorii JDBC pentru interacționarea aplicațiilor cu bazele de date relaționale.

Aplicațiile folosesc furnizori JDBC pentru interacționarea cu bazele de date relaționale. 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, asociați sursa de date cu furnizorul JDBC. Împreună, furnizorul JDBC și obiectele sursei de date sunt echivalente funcțional cu fabrica de conexiuni Java EE Connector Architecture (JCA), care furnizează conexiunea cu o bază de date non-relațională.

Referiți-vă la exemplele din Setări pentru mediul tipic autonom și Setări pentru mediul tipic de implementare din subiectul anterior.

Pentru informații suplimentare despre furnizorii JDBC, vedeți “Furnizori JDBC” în Centrul de informare WebSphere Application Server.

Magistrala de integrare servicii pentru IBM Business Process Manager:

O magistrală de integrare a serviciului este un mecanism de comunicare gestionat care suportă integrarea serviciului prin mesageria sincronă și asincronă. O magistrală conține motoare de mesagerie interconectate care gestionează resursele magistralei. Este una din tehnologiile WebSphere Application Server pe care este bazat IBM Business Process Manager.

O singură magistrală de integrare servicii și un singur motor de mesagerie folosesc implicit aceeași schemă de bază de date ca și baza de date a produsului. Fiecare mediu de implementare are magistrala sa proprie. Magistrala singulară se numește **BPM.nume_mediu_implementare.Bus**.

O destinație magistrală este o adresă logică la care aplicațiile se pot atașa ca producător, consumator sau ambele. O destinație coadă este o destinație magistrală care este utilizată pentru mesagerie punct-la-punct.

Fiecare magistrală poate avea unul sau mai mulți membri magistrală, fiecare dintre ei fiind un server sau un cluster.

Topologie magistrală este aranjarea fizică a serverelor de aplicații, motoarelor de mesagerie și a managerilor de cozi WebSphere MQ și tiparul conexiunilor magistralei și al legăturilor dintre ele care alcătuiesc magistrala de servicii întreprindere.

Aplicații de servicii și module de servicii

Un modul serviciu este un modul Service Component Architecture (SCA) care oferă servicii în mediul runtime. Atunci când instalați un modul service pe IBM Business Process Manager, construiți o aplicație servicii asociată, care este împachetată ca un fișier EAR (enterprise archive).

Modulele de servicii sunt unități de bază ale implementării și pot conține componente, biblioteci și module de intermediere folosite de aplicația pentru servicii asociată. Modulele de servicii au exporturi și, în mod opțional, importuri pentru a defini relațiile între module și solicitanții și furnizorii serviciului. WebSphere Process Server suportă module pentru serviciile business și module de mediere. Atât modulele, cât și modulele de mediere sunt tipuri de module SCA. Un modul de mediere permite comunicația între aplicații prin transformarea invocării serviciului într-un format înțeles de țintă, transmițând cererea către țintă și returnând rezultatul originatorului. Un modul pentru un serviciu operațional implementează logica unui proces operațional. Totuși, un modul poate de asemenea să includă aceeași logică de mediere care poate fi împachetată într-un modul de mediere.

Implementarea unei aplicații de servicii

Procesul de implementare a unui fișier EAR care conține o aplicație de servicii este același cu cel de implementarea a oricărui fișier EAR. Puteți modifica valorile pentru parametrii de mediere în momentul implementării. După ce au implementat un fișier EAR care conține un modul SCA, puteți vizualiza detaliile legate de aplicația de servicii și modulele sale asociate. Puteți vedea modul în care un modul de servicii este conectat la solicitanții serviciului (prin exporturi) și la furnizorii de servicii (prin importuri).

Vizualizarea detaliilor modului SCA

Detaliile legate de modulul de servicii pe care le puteți vizualiza depind de modulul SCA. Acestea includ următoarele atribute.

- Nume modul SCA
- Descriere modul SCA
- Nume asociat aplicației
- Informații legate de versiunea modului SCA, dacă modulul are mai multe versiuni
- Importuri module SCA:
 - Interfețele de import sunt definiții prezentare pe scurt care descriu modul în care un modul SCA accesează un serviciu.
 - Legările de import sunt definiții concrete care specifică mecanismul fizic prin care un modul SCA accesează un serviciu. De exemplu, folosirea SOAP/HTTP.
- Exporturi modul SCA:
 - Interfețele de export sunt definiții prezentare pe scurt care descriu modul în care solicitantii serviciului accesează un modul SCA.
 - Legările de export sunt definiții concrete care specifică mecanismul fizic prin care un solicitant al serviciului accesează un modul SCA, și în mod indirect, un serviciu.
- Proprietățile modului SCA

Importurile și legările de import:

Importurile definesc interacțiuni între modulele SCA și furnizorii de servicii. Modulele SCA folosesc importuri pentru a permite componentelor să acceseze servicii externe (servicii care se află în afara modului 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 țintele lor intenționate.

Interfețele și legările

Importul unui modul SCA are nevoie de cel puțin o interfață și are o singură legare.

- Interfețele de import sunt definiții abstracte care definesc un set de operațiuni folosind Web Services Description Language (WSDL), și limbaj XML pentru descrierea serviciilor Web. Un modul SCA poate avea mai multe interfețe de import.
- Legările de import sunt definiții 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 legări de import:

- Legările SCA conectează modulele SCA cu alte module SCA. De asemenea, legările SCA sunt menționate și ca legări implicite.
- Legările Serviciu Web permit componentelor să invoce servicii web. Protocoalele suportate sunt SOAP1.1/HTTP, SOAP1.2/HTTP, și SOAP1.1/JMS.

Puteți 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 interacțiunea cu serviciile folosind documente sau legări literale RPC și care folosesc handler-e JAX-WS pentru a personaliza invocarea. Este furnizată o legare SOAP1.1/HTTP separată pentru a permite interacțiunea cu serviciile care folosesc o legare codată RPC sau acolo unde există o cerință pentru folosirea handler-elor JAX-RPC pentru a personaliza invocările.
- Legările HTTP vă permit să accesați aplicații folosind protocolul HTTP.

- Legările de import EJP (Enterprise JavaBeans) permit componentelor SCA să invoce servicii oferite de logica operațională Java EE care rulează pe un server Java EE.
- Legările EIS (Enterprise information system) asigură conectivitatea între componente SCA și un EIS extern. Această comunicație 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 și HTTP sau HTTPS. Clasa JMS Message și cele cinci subtipuri ale sale (Text, Bytes, Object, Stream și Map) sunt acceptate automat.
- Legările JMS interoperabilitate cu furnizori JMS terță parte care se integrează cu WebSphere Application Server folosind JMS ASF (Application Server Facility).
- Legările JMS MQ WebSphere permit interoperabilitatea cu furnizorii JMS bazați pe MQ WebSphere. Clasa JMS Message și cele cinci subtipuri ale sale (Text, Bytes, Object, Stream și Map) sunt acceptate automat. Dacă doriți să utilizați WebSphere MQ pe post de furnizor JMS, folosiți legările JMS MQ WebSphere.
- Legările MQ WebSphere permit interoperabilitatea cu furnizorii MQ WebSphere. Puteți folosi legări WebSphere MQ doar împreună cu managerii cozii aflați la distanță prin intermediul unei conexiuni client WebSphere MQ; nu le puteți folosi cu managerii locali ai cozii. Folosiți legări WebSphere MQ dacă doriți să comunicați cu aplicațiile 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 invocați un serviciu la un alt punct final, prin orice legare de import suportată. Invocarea dinamică a serviciilor vă permite de asemenea să invocați 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 și configurația sa pentru invocarea dinamică. Importul folosit pentru invocarea dinamică poate fi legat de componenta de apelare, sau poate fi selectat dinamic la runtime.

Pentru serviciul Web și invocații SCA, se poate face o invocare dinamică fără import, cu protocolul și configurarea deduse din URL-ul punct final. Tipul țintă 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 URL HTTP sau JMS indică în mod implicit invocări ale serviciului web; pentru aceste URL-uri este posibilă furnizarea unor valori de tip de legare suplimentare care să indice faptul că URL-ul reprezintă o invocare a modului de legare a unui HTTP sau JMS.
- Pentru un URL HTTP al unui serviciu web, implicit este folosirea SOAP 1.1, și se poate specifica o valoare tip de legare care să indice folosirea SOAP 1.2.

Exporturile și legările de export:

Exporturile definesc interacțiuni între modulele SCA și solicitanții 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 solicitanții serviciului.

Interfețele și legările

Un export pentru un modul SCA are nevoie de cel puțin o interfață.

- Interfețele de export sunt definiții abstracte care definesc un set de operațiuni folosind Web Services Description Language (WSDL), și limbaj XML pentru descrierea serviciilor Web. Un modul SCA poate avea mai multe interfețe de export.
- Legările de export sunt definiții concrete care specifică mecanismul fizic folosit de solicitanții serviciului pentru a accesa un serviciu. De obicei, exportul unui modul SCA are specificată o singură legare. Un export fără legări specificate este interpretat de mediul runtime ca un export cu o legare SCA.

Legările de export suportate

IBM Business Process Manager suportă următoarele legări de export:

- Legările SCA conectează modulele SCA cu alte module SCA. De asemenea, legările SCA sunt menționate și ca legări implicite.
- Legările Serviciu web permit exporturi care pot fi invocate ca și servicii web. Protocoalele suportate sunt SOAP1.1/HTTP, SOAP1.2/HTTP, și SOAP1.1/JMS.
Puteți folosi o legare SOAP1.1/HTTP sau SOAP1.2/HTTP bazată pe JAX-WS (Java API for XML Web Services), ceea ce permite interacțiunea cu serviciile folosind documente sau legări de literal RPC și care folosesc handler-e JAX-WS pentru a personaliza invocările. Este furnizată o legare SOAP1.1/HTTP separată pentru a permite interacțiunea cu serviciile care folosesc o legare codată RPC sau acolo unde există o cerință 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 operațională Java EE să poată invoca componente SCA care altfel le-ar fi indisponibile.
- Legările EIS (Enterprise information system) asigură conectivitatea între componente SCA și un EIS extern. Această comunicație 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 și HTTP sau HTTPS. Clasa JMS Message și cele cinci subtipuri ale sale (Text, Bytes, Object, Stream și Map) sunt acceptate automat.
- Legările JMS interoperabilitate cu furnizori JMS terță parte care se integrează cu WebSphere Application Server folosind JMS ASF (Application Server Facility).
- Legările JMS MQ WebSphere permit interoperabilitatea cu furnizorii JMS bazați pe MQ WebSphere. Clasa JMS Message și cele cinci subtipuri ale sale (Text, Bytes, Object, Stream și Map) sunt acceptate automat. Dacă doriți să utilizați WebSphere MQ pe post de furnizor JMS, folosiți legările JMS MQ WebSphere.
- Legările MQ WebSphere permit interoperabilitatea cu furnizorii MQ WebSphere. Puteți utiliza o conexiune de la distanță (sau client) pentru a vă conecta la un manager de coadă MQ aflat pe o mașină la distanță. O conexiune locală (sau legări) este o conexiune directă la WebSphere MQ. Aceasta poate fi utilizată doar pentru o conexiune la un manager al cozii MQ de pe aceeași mașină. WebSphere MQ va permite ambele tipuri de conexiuni, dar legările MQ suportă doar conexiunea "de la distanță" (sau "client").

Module de mediere:

Modulele de mediere sunt module SCA (Service Component Architecture) care pot modifica formatul, conținutul sau ținta cererilor serviciului.

Modulele de mediere operează asupra mesajelor care sunt în zbor între solicitanții și furnizorii serviciului. Puteți ruta mesaje către diferiți furnizori de servicii și puteți modifica conținutul sau forma mesajului. Modulele de mediere pot oferi funcții cum ar fi înregistrarea mesajului și procesarea erorii care sunt adaptate cerințelor dumneavoastră.

Puteți modifica anumite aspecte ale modulelor de mediere din consola administrativă fără a fi nevoie să reimplementați modulul .

Componentele modulelor de mediere

Modulele de mediere cuprind următoarele elemente:

- Importuri, care definesc interacțiunea între modulele SCA și furnizorii de servicii. Acestea permit modulelor SCA să apeleze servicii externe ca și cum acestea ar fi locale. Puteți vizualiza importurile modulelor de mediere și să modificați legarea.
- Exporturi, care definesc interacțiunea între modulele SCA și solicitanții serviciului. Acestea permit unui modul SCA să ofere un serviciu și să definească interfețele externe (puncte de acces) pentru un modul SCA. Puteți vizualiza exporturile modulelor de mediere.

- Componente SCA, care construiesc blocuri pentru modulele SCA precum module de mediere. Aveți posibilitatea să creați și să particularizați module SCA și componente în mod grafic, folosind Integration Designer. După ce ați implementat un modul de mediere puteți personaliza anumite aspecte ale acestuia din consola administrativă fără să fiți nevoiți să reimplementați modulul.

De obicei, module de mediere conține un anumit tip de componentă SCA numită *componentă a fluxului de mediere*. Componentele fluxului de mediere definesc fluxurile de mediere.

Componenta unui flux de mediere poate conține una, un anumit număr de primitive de mediere sau niciuna. IBM Business Process Manager suportă un set livrat de primitive de mediere care oferă funcționalitate pentru rutarea și transformarea mesajelor. Pentru flexibilitate suplimentară a primitivei de mediere, utilizați primitiva Mediere personalizată pentru apelarea logicii personalizate.

Scopul unui modul de mediere care nu conține 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 transformată în SOAP/HTTP înainte de a fi trimisă mai departe.

Notă: Aveți posibilitatea să vizualizați și să faceți anumite modificări asupra modulelor de mediere din IBM Business Process Manager. Totuși, nu puteți vizualiza sau modifica componentele SCA din interiorul unui modul din IBM Business Process Manager. Utilizați Integration Designer pentru a personaliza componentele SCA.

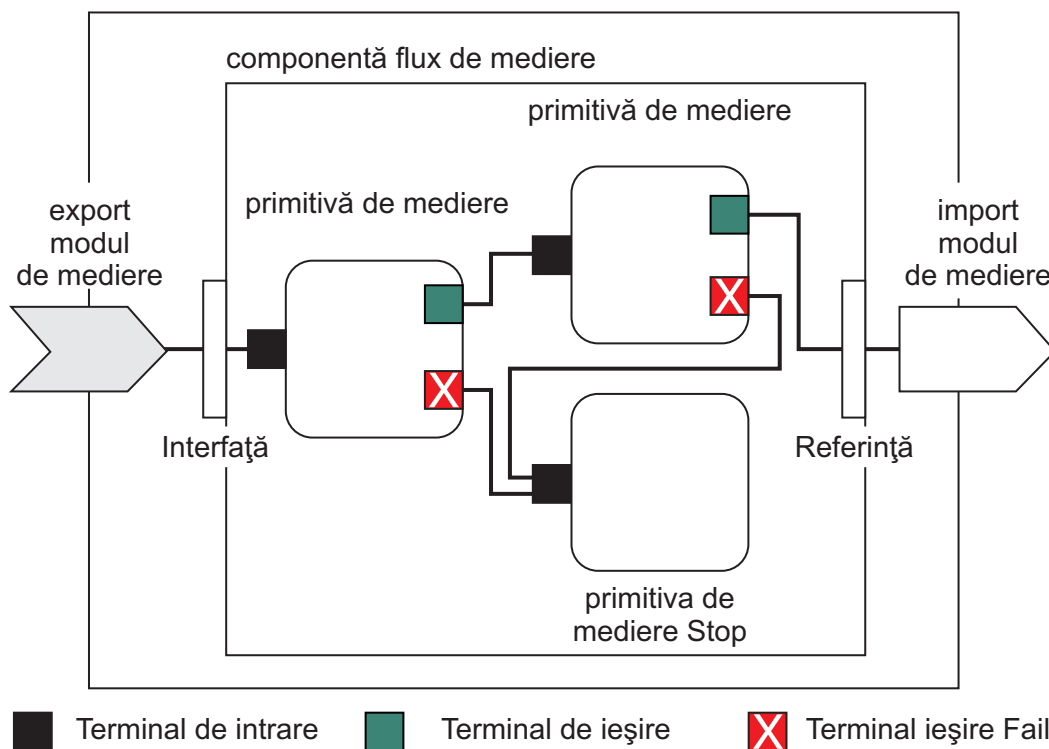


Figura 43. Exemplu simplificat de modul mediere. Modulul de mediere conține o singură componentă a fluxului de mediere, care conține primitive de mediere.

- Proprietăți

Primitivele de mediere au proprietăți, unele dintre ele pot fi afișate în consola administrativă sub formă de proprietăți suplimentare ale unui modul SCA.

Pentru ca proprietățile primitivelor de mediere să fie vizibile din consola administrativă IBM Business Process Manager, dezvoltatorul care se ocupă cu integrarea trebuie să le promoveze. Anumite proprietăți se pretează a fi configurate administrativ și Integration Designer le descrie ca proprietăți ce pot fi promovate deoarece acestea pot fi promovate din cercul de integrare în ciclul administrativ. Alte proprietăți nu sunt potrivite pentru configurare

administrativă deoarece modificarea acestora poate afecta fluxul de mediere în așa fel încât modulul de mediere va trebui să fie reimplementat. Integration Designer listează proprietățile pe care le puteți alege pentru a promova în proprietățile promovate ale unei primitive de mediere.

Puteți folosi consola administrativă IBM Business Process Manager pentru a modifica valoarea proprietăților promovate fără a trebui să reimplementați un modul de mediere sau să reporniți serverul sau modulul.

În general, fluxurile de mediere utilizează imediat modificările proprietății. Cu toate acestea, dacă apar modificări asupra proprietății într-o celulă pentru managerul de implementare, acestea au efect în fiecare nod pe măsură ce acestea sunt sincronizate. De asemenea, fluxurile de mediere care sunt în zbor continua să utilizeze valorile anterioare.

Notă: Din consola administrativă, aveți posibilitatea să modificați doar valorile proprietății, nu grupurile acesteia, numele sau tipurile. Dacă doriți să modificați grupurile de proprietăți, nume sau tipuri, trebuie să utilizați Integration Designer.

- Un modul de mediere sau biblioteca dependentă poate defini, de asemenea, subfluxuri. Un subflux 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, și sunt implementate în general pe IBM Business Process Manager înăuntrul unui fișier EAR (enterprise archive).

Aveți posibilitatea să modificați valoarea proprietăților promovate în momentul implementării.

Puteți exporta un modul de mediere din Integration Designer, și să determinați ca Integration Designer să împacheteze modulul de mediere într-un fișier JAR (Java archive), iar acest fișier JAR într-un fișier EAR. Apoi, aveți posibilitatea să implementați apoi fișierul EAR prin instalarea unei noi aplicații din consola administrativă.

Modulele de mediere pot fi gândite ca o singură entitate. Totuși, modulele SCA sunt definite de un număr de fișiere XML memorate într-un fișier JAR.

Exemplu de fișier EAR, care conține un modul de mediere

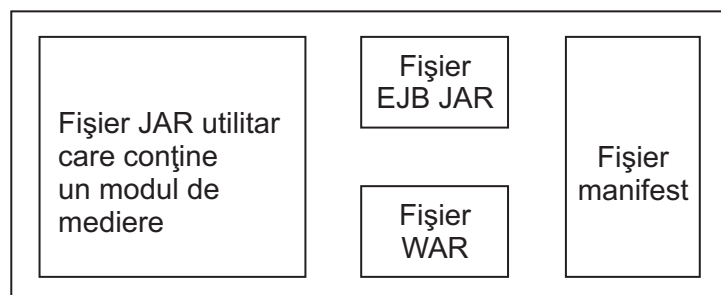


Figura 44. Un exemplu simplificat de fișier EAR care conține un modul de mediere. Fișierul EAR conține JAR-uri. Fișierul JAR utilitar conține un modul de mediere.

Primitive de mediere:

Componentele fluxului de mediere operează asupra fluxurilor de mesaje între componentelor serviciului. Capabilitățile componente 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 și unul pentru răspuns.

IBM Business Process Manager suportă un set livrat de primitive de mediere, care implementează capabilitățile standard de mediere pentru modulele de mediere sau pentru modulele implementate în IBM Business Process Manager. Dacă aveți nevoie de anumite capabilități de mediere, puteți să vă dezvoltați propriile primitive personalizate de mediere.

O primitivă de mediere definește o operație 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 operație de tip “out” care trimite mesaje la o altă componentă sau la un alt modul.

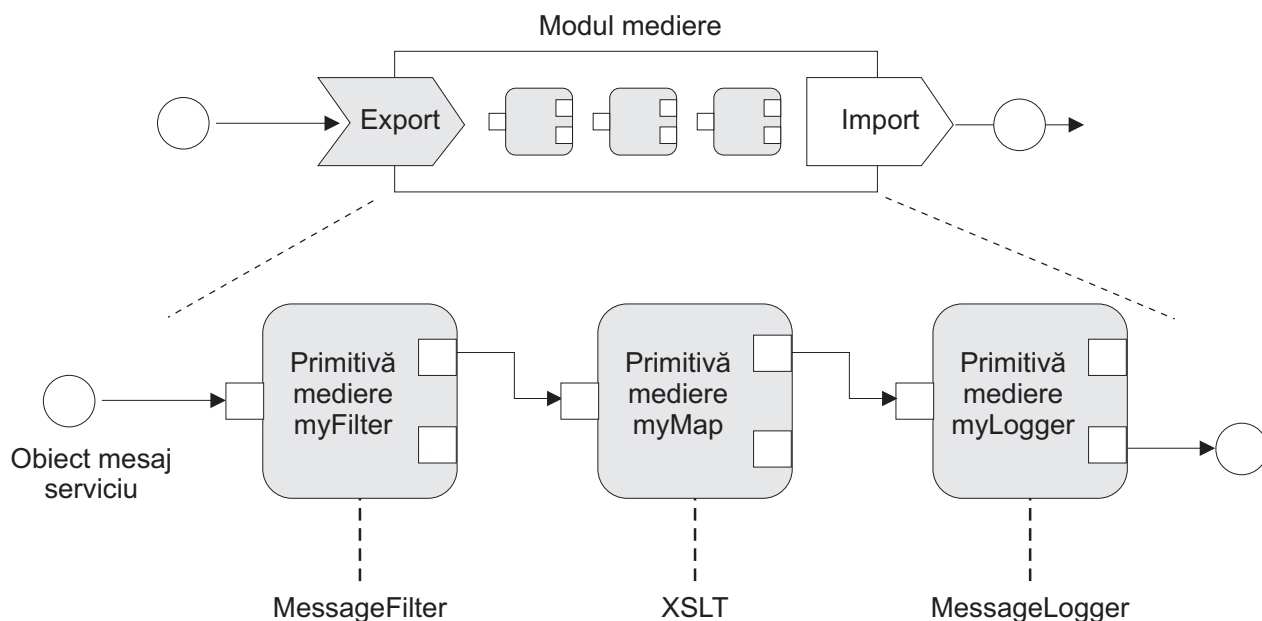


Figura 45. Modulul de mediere care conține trei primitive de mediere

Puteți folosi Integration Designer pentru a configura primitivele de mediere și pentru a le seta proprietățile. Unele dintre aceste proprietăți pot fi făcute vizibile pentru administratorul de runtime prin promovarea acestora. Oricare dintre proprietățile primitivelor de mediere care pot fi promovate pot fi de asemenea proprietăți dinamice. O proprietate dinamică poate fi înlocuită în momentul execuției prin folosirea unui fișier de politică.

Integration Designer vă permite de asemenea să modelați grafic și să asamblați componentele fluxului de mediere din cadrul primitivei de mediere, și să asamblați module de mediere sau module din cadrul componentelor fluxului de mediere. Consola administrativă se referă la modulele de mediere și la modulele ca la module SCA.

Integration Designer permite de asemenea definirea de subfluxuri în module sau a bibliotecilor lor dependente. Un subflux poate conține orice primitivă de mediere, cu excepția primitivei Policy Resolution. Un subflux este invocat din fluxul de cerere sau de răspuns, sau dintr-un alt subflux folosind primitiva de mediere Subflux. Proprietățile promovate într-un subflux prin primitivele de mediere sunt expuse ca proprietăți 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.

- Definește transformări pentru mesaje folosind o mapare pentru obiecte business care poate fi refolosită.
- Vă permite să definiți transformări pentru mesaje în mod grafic folosind editorul pentru maparea cu obiecte business.
- Poate modifica conținutului unui mesaj.
- Poate transforma tipul unui mesaj de intrare într-un tip diferit de mesaj de ieșire.

Custom Mediation

Vă permite să implementați 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. Puteți crea transformări complexe și tipare de rutare astfel:

- Crearea de cod Java.
- Crearea propriilor proprietăți.
- Adăugarea de terminale noi.

Puteți apela un serviciu dintr-o primitivă de mediere personalizată, dar primitiva de mediere Service Invoke este proiectată să apeleze servicii și să asigure funcționalități suplimentare, precum reîncercarea.

Data Handler

Vă permite să modificați 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-o formă fizică. Utilizarea primară a primitivei este de a converti un format fizic, precum un șir de text dintr-un obiect JMS Text Message, într-o structură de tip Obiect business și din nou înapoi. Această mediere este frecvent utilizată pentru a:

- Transformarea unei secțiuni a mesajului de intrare dintr-o structură definită în alta - un exemplu de acest fel ar fi cazul în care SMO include o valoare șir care este delimitată prin virgulă și vreți să o parșați într-un Obiect business specific.
- Modificarea tipului de mesaj - un exemplu ar fi când un export JMS a fost configurat să folosească o legătură de date tipizată de bază JMS și în modulul de mediere dezvoltatorul de integrare decide extinderea conținutului la o structură BO (obiect business) specifică.

Database Lookup

Modifică mesaje, utilizând informații dintr-o bază de date ce sunt furnizate de utilizator.

- Trebuie să configurați o bază de date, o sursă de date, precum și orice setări de autentificare necesare serverului utilizate de primitiva de mediere Database Lookup. Pentru a face acest lucru, ajutați-vă de consola administrativă.
- Primitiva de mediere Database Lookup poate citi dintr-o singură tabelă.
- Coloana index specificată trebuie să conțină o valoare unică.
- Datele din coloanele ce conțin 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.

- Informațiile 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 distanță.
- Faceți modificările asupra registrului din consola administrativă WSRR.
- IBM Business Process Manager trebuie să știe ce registru să folosească, și de aceea, trebuie să creați definiții de acces WSRR folosind consola administrativă IBM Business Process Manager.

Event Emitter

Îmbunătățește monitorizarea prin permiterea trimiterii evenimentelor din interiorul unei componente a fluxului de mediere.

- Puteți suspenda acțiunea de mediere prin debifarea casetei de bifare.

- Puteți vizualiza evenimente Event Emitter utilizând browser-ul Common Base Events în IBM Business Process Manager.
- Ar trebui să trimiteți evenimente doar către un singur punct important într-un flux de mediere, din motive de performanță.
- Aveți posibilitatea să definiți părțile mesajului conținut de eveniment.
- Evenimentele sunt trimise în formatul Common Base Events și sunt trimise la un server Common Event Infrastructure.
- Pentru a utiliza pe deplin informațiile legate de Event Emitter, consumatorii de evenimente trebuie să înțeleagă structura evenimentelor CBE (Common Base Events). Evenimentele CBE au o schemă generală, dar acest lucru nu modelează datele specifice aplicației care sunt conținute în elementele extinse de date. Pentru a modela elementele extinse de date, unelele Integration Designer generează un fișier cu definiții pentru catalogul de evenimente Common Event Infrastructure pentru fiecare dintre primitivele de mediere Event Emitter configurate. Fișierele 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 faceți referire la fișierele cu definiții pentru catalogul de evenimente atunci când creați aplicații care consumă evenimente Event Emitter.
- Puteți specifica o altă monitorizare din IBM Business Process Manager. De exemplu, puteți controla ca evenimentele să fie emise din importuri și exporturi.

Fail Oprește o anumită cale din flux și generează o excepție.

Fan In Ajută la agregarea (combinarea) mesajelor.

- Poate fi folosită doar în combinație cu primitiva de mediere Fan Out.
- Împreună, primitivele de mediere Fan Out și Fan In permit agregarea datelor într-un singur mesaj de ieșire.
- Primitiva de mediere Fan In primește mesaje până când se ajunge la un punct de decizie, moment în care este trimis la ieșire un mesaj.
- Contextul partajat ar trebui folosit pentru reținerea datelor agregate.

Fan Out

Ajută la divizarea și agregarea (combinarea) mesajelor.

- Împreună, primitivele de mediere Fan Out și Fan In permit agregarea datelor într-un singur mesaj de ieșire.
- În modul iterativ, primitiva de mediere Fan Out vă permite să iterați printr-un printr-un singur mesaj de intrare care conține un element repetitiv. Pentru fiecare apariție a elementului repetitiv este trimis un mesaj.
- Contextul partajat ar trebui folosit pentru reținerea datelor agregate.

HTTP Header Setter

Oferă un mecanism pentru gestionarea anteturilor în mesajele HTTP.

- Poate crea, seta, copia sau șterge anteturile mesajelor HTTP.
- Poate seta mai multe acțiuni pentru a modifica mai multe anteturi HTTP.

Mapare

Transformă mesaje.

- Vă permite să realizați transformări XSL (Extensible Stylesheet Language) sau transformări Business Object Map.
- Puteți 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 conținutului mesajelor.

- Poate modifica, adăuga sau șterge elemente din mesaj.
- Nu modifică tipul de mesaj.
- Datele din coloanele ce conțin 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 funcție de conținutul acestora.

- Puteți suspenda acțiunea de mediere prin debifarea casetei de bifare.

Message Logger

Înregistrează mesajele într-o bază de date relațională sau prin intermediul propriului jurnalizator personalizat. Mesajele sunt memorate sub formă de XML, și de aceea, datele pot fi procesate după aceea de aplicațiile care suportă XML.

- Puteți suspenda acțiunea de mediere prin debifarea casetei de bifare.
- Schema bazei de date relaționale (structura tabeli) este definită de IBM.
- În mod implicit, primitiva de mediere Message Logger folosește baza de date Common. Mediul runtime mapează sursa de date la **jdbc/mediation/messageLog** pe baza de date Comună.
- Puteți seta clasele de implementare Handler pentru a personaliza comportamentul jurnalizatorului personalizat. Opțional, puteți 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 șterge anteturile mesajelor MQ.
- Poate seta mai multe acțiuni pentru a modifica mai multe anteturi MQ.

Policy Resolution

Permite configurarea dinamică a cererilor, prin căutarea punctelor finale ale serviciului și a fișierelor de politică asociate, într-o magazie.

- Aveți posibilitatea să utilizați un fișier de politică pentru a înlocui dinamic proprietățile promovate ale altor primitive de mediere.
- Informațiile punctului final al serviciului și informațiile legate de politică sunt extrase dintr-un WSRR (WebSphere Service Registry and Repository). Registrul WSRR poate fi local sau poate fi aflat la distanță.
- Faceți modificările asupra registrului din consola administrativă WSRR.
- IBM Business Process Manager trebuie să știe ce registru să folosească, și de aceea, trebuie să creați definiții 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 aștepte până la sfârșitul unui flux de mediere și folosirea mecanismului callout.

- Dacă serviciul returnează un defect, puteți reîncerca același serviciu sau să apălați 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 combinație cu alte primitive de mediere pentru medieri mai complexe.

Set Message Type

În timpul dezvoltării de integrare, vă permite să tratați câmpurile mesajului care sunt tipizate slab, ca și pe cele care sunt tipizate tare. Un câmp este tipizat slab dacă poate conține mai multe tipuri de date. Un câmp este tipizat tare dacă tipul și structura sa internă sunt cunoscute.

- La runtime, medierea Setare tip mesaj permite verificarea potrivirii între conținutul unui mesaj și tipurile de date pe care le așteptați.

SOAP Header Setter

Oferă un mecanism pentru gestionarea anteturilor în mesajele SOAP.

- Poate crea, seta, copia sau șterge anteturile mesajelor SOAP.
- Poate seta mai multe acțiuni pentru a modifica mai multe anteturi SOAP.

Stop Oprește o anumită cale din flux fără a genera o excepție.

Type Filter

Vă permite să direcționați mesajele pe o cale diferită a fluxului, în funcție de tipul acestora.

Extragere WebSphere eXtreme Scale

Puteți extrage informații dintr-un mediu cache server eXtreme Scale.

- Puteți căuta valori în cache și le puteți păstra ca elemente în mesaj folosind un index.
- Combinând primitivele de mediere Stocare și Extragere eXtreme Scale, puteți memora în cache răspunsul de la un sistem back-end. Cererile viitoare nu vor necesita acces la acel sistem de back-end.
- Trebuie să creați definiții eXtreme Scale utilizând consola administrativă WebSphere ESB, astfel încât puteți specifica serverul eXtreme Scale de utilizat.

Stocare WebSphere eXtreme Scale

Puteți stoca informații într-un mediu cache server eXtreme Scale.

- Puteți stoca informații într-un cache eXtreme Scale utilizând un index sau un obiect.
- Combinând primitivele de mediere eXtreme Scale Stocare și Extragere, puteți folosi primitiva de mediere Stocare pentru stocarea datelor în memoria cache și puteți folosi primitiva de mediere Extragere pentru a extrage date stocate anterior în memoria cache.
- Trebuie să creați definiții eXtreme Scale utilizând consola administrativă WebSphere ESB, astfel încât puteți specifica serverul eXtreme Scale de utilizat.

Rutarea dinamică:

Puteți 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 și selecția 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 execuției are capacitatea de a ruta mesajele de cerere și 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 informații dintr-un registru, o bază de date, sau cu informații 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 execuției 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 execuției), astfel încât administratorul din timpul rulării să o poată seta. Aveți posibilitatea să vizualizați proprietățile modulului în fereastra **Proprietăți modul**. Pentru a vedea fereastra, faceți clic pe **Aplicații > Module SCA > Proprietăți Modul**. Dezvoltatorul care se ocupă cu integrarea oferă proprietăților promovate aliasuri, iar acestea sunt numele afișate în consola administrativă.

Registru

Puteți folosi WSRR (IBM WebSphere Service Registry and Repository) pentru a memora informațiile legate de punctul final al serviciului, iar apoi să creați module SCA pentru a extrage puncte finale din magazia WSRR.

Atunci când dezvoltați module SCA, folosiți primitiva de mediere Endpoint Lookup pentru a permite unui flux de mediere să interogheze un registru WSRR pentru a obține un punct final pentru serviciu sau un set de puncte finale pentru serviciu. Dacă un modul SCA 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:

Puteți utiliza politici de mediere pentru a controla fluxurile de mediere dintre solicitanții serviciului și furnizorii de servicii.

Puteți 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 aveți module SCA (Service Component Architecture) dorite și documente ale politicii de mediere în registrul dumneavoastră WSRR.

Cum se face atașarea unei politici de mediere la o cerere de serviciu

Atunci când dezvoltați un modul SCA care trebuie să facă uz de o politică de mediere, trebuie să includeți în fluxul de mediere o primitivă de mediere Policy Resolution. În timpul rulării, primitiva de mediere Policy Resolution obține informații despre politica de mediere din registru. Prin urmare, un modul SCA trebuie să conțină o componentă flux de mediere pentru a suporta controlul politicii de mediere a cererilor de servicii.

În registru, puteți atașa una sau mai multe politici de mediere la un modul SCA sau la un serviciu țintă utilizat de către modulul SCA. Politicile de mediere atașate pot fi utilizate (sunt în domeniu) pentru toate mesajele serviciului procesate de către acel modul SCA. Politicile de mediere pot avea atașamente de politică care definesc condiții. Condițiile 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 memorați, accesați și să gestionați informațiile legate de punctele finale ale serviciului și de politicile de mediere. Puteți utiliza WSRR pentru a face ca aplicațiile serviciului dvs. să fie mult mai dinamice și mult mai adaptabile la modificările aduse condițiilor de business.

Introducere

Fluxurile de mediere pot folosi WSRR pe post de mașină de căutare dinamică, oferind informații legate de punctele finale sau de politicile de mediere ale serviciului.

Pentru a configura accesul la WSRR, creați documente de definiție WSRR folosind consola administrativă. Alternativ, aveți posibilitatea să utilizați comenzile de administrare WSRR din clientul script wsadmin. Definițiile WSRR și proprietățile acestora de conexiune reprezintă mecanismul de conectare la o instanță a unui registru, și de extragere a punctului final sau a politicii de mediere pentru un serviciu.

Puncte finale pentru serviciu

Puteți folosi WSRR pentru a memora informații despre serviciile pe care le folosiți deja, pentru cele pe care plănuți să le folosiți sau pentru cele de care vreți să fiți conștient. Aceste servicii pot fi în sistemele dvs, sau în ale sistemelor. De exemplu, aplicația ar putea folosi WSRR la localizarea celui mai corespunzător serviciu care îi satisface nevoile funcționale și de performanță.

Atunci când dezvoltați un modul SCA care are nevoie să acceseze punctele finale ale serviciului din WSRR, trebuie să includeți o primitivă de mediere Endpoint Lookup în fluxul de mediere. În momentul rulării, primitiva de mediere Endpoint Lookup obține punctele finale ale serviciului din registru.

Politici de mediere

De asemenea, puteți folosi WSRR pentru a memora informații legate de politica de mediere. Politicile de mediere vă pot ajuta să controlați cererile de servicii prin suprascrisura dinamică a proprietăților modulului. Dacă WSRR conține

politici de mediere care sunt atașate la un obiect care reprezintă fie modulul dvs. SCA, fie serviciul dvs. țintă, atunci politicile de mediere ar putea suprascrie proprietățile modulului. Dacă doriți ca în diferite contexte să se aplice politici de mediere diferite, puteți crea condiții pentru aceste politici.

Notă: Politicile de mediere sunt preocupate de control fluxurilor de mediere, nu de securitate.

Atunci când dezvoltați un modul SCA care trebuie să facă uz de o politică de mediere, trebuie să includeți în fluxul de mediere o primitivă de mediere Policy Resolution. În momentul rulării, primitiva de mediere Policy Resolution obține informațiile legate de politica de mediere din registru.

WebSphere eXtreme Scale:

Utilizând produsul WebSphere eXtreme Scale (eXtreme Scale) puteți asigura un sistem de memorare în cache pe care îl puteți integra cu o aplicație IBM Business Process Manager. Utilizând eXtreme Scale cu IBM Business Process Manager puteți îmbunătăți timpul de răspuns și fiabilitatea serviciului și asigura funcționalitate de integrare suplimentară.

eXtreme Scale acționează ca o grilă de date elastică, scalabilă, în memorie. Grila de date memorează în cache, partiționează, replică și gestionează dinamic datele aplicației și logica operațională pe mai multe servere. Cu eXtreme Scale, puteți, de asemenea, să obțineți calitate a serviciilor precum integritate tranzacțională, disponibilitate înaltă și timpi de răspuns predictabili.

Puteți utiliza fluxuri de mediere pentru a accesa funcția de memorare în cacheeXtreme Scale inclusiv primitivele de mediere WebSphere eXtreme Scale din fluxul dumneavoastră. La dezvoltarea unui modul SCA (Service Component Architecture) care are nevoie să stocheze informații într-o memorie cacheeXtreme Scale, trebuie să includeți primitiva de mediere WebSphere eXtreme Scale Store în fluxul de mediere. Dacă vreți să extrageți informații dintr-o memorie cache eXtreme Scale, trebuie să includeți primitiva de mediere WebSphere eXtreme Scale Retrieve. Combinând cele două primitive de mediere într-un flux de mediere, puteți 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 creați o definiție WebSphere eXtreme Scale utilizând consola administrativă. Alternativ, puteți utiliza comenzile de administrare WebSphere eXtreme Scale din clientul de scriptare wsadmin. O definiție eXtreme Scale este mecanismul utilizat de primitivele de mediere WebSphere eXtreme Scale Retrieve și Store pentru conectarea la un servereXtreme Scale.

Clienți ai serviciului de mesaje

Clienții pentru serviciul de mesaje sunt disponibili pentru C/C++ și .NET pentru a permite aplicațiilor 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 același set de interfețe ca și API-ul JMS (Java Message Service). Clieții serviciului de mesaje pentru C/C++ conține două implementări ale XMS, una pentru aplicațiile C și una pentru aplicațiile by C++. Message Service Client for .NET conține o implementare complet gestionată a XMS, care poate fi folosită de orice limbaj compatibil cu .NET.

Puteți obține clienți ai serviciului de mesaje (Message Service Clients) 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

Puteți obține clienți ai serviciului de mesaje (Message Service Clients) 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.

Puteți instala și folosi și suportul pentru clientul Java EE de la WebSphere Application Server Network Deployment, inclusiv client servicii web, EJB Client și JMS Client.

Capitolul 2. Aflați mai multe despre conceptele cheie

Utilizați această secțiune ca un punct de plecare pentru investigarea tehnologiilor utilizate în și de către IBM Business Process Manager.

Crearea scenariilor

Folosiți scenariii pentru a înțelege și lucra cu componentele și produsele din familia gestionarea proceselor operaționale.

Versionarea

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

Modul în care funcționează această versionare în IBM Business Process Manager depinde de ce anume implementați - o aplicație de proces, implementată din magazin în IBM Process Center sau o aplicație de întreprindere implementată direct din IBM Integration Designer.

Aplicațiile de proces și trusele de unelte pe care le implementați într-un mediu de runtime din Process Center sunt versionate în mod implicit. Pentru aplicațiile de întreprindere, puteți alege să versionați modulele și bibliotecile în IBM Integration Designer.

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

Versionarea aplicațiilor de proces

Versionarea furnizează abilitatea, pentru mediul runtime, de a identifica instanțele din ciclul de viață al unei aplicații de proces și de a fi capabil să ruleze simultan mai multe instanțe pe un Process Server.

Pentru a înțelege modul în care sunt versionate aplicațiile de proces, este important să reținem că o aplicație de procesare este un container care reține diverse artefacte folosite în sau de aplicația de proces (de exemplu, modelele de procesare sau BPD-urile, referințele seturilor de instrumente, serviciile, or tracks). Orice versionare este făcută la acest nivel de container, nu la nivelul artefactelor individuale. Pentru aplicații proces, aceasta înseamnă că versionarea apare când realizați un instantaneu.

Puteți compara instanțele pentru a determina diferențele dintre versiuni. De exemplu, dacă un dezvoltator a corectat o problemă cu un serviciu și a realizat un instantaneu al aplicației de proces sau al trusei sale de unelte în acel moment, și apoi un alt dezvoltator a făcut mai multe modificări suplimentare la același serviciu și a realizat un nou instantaneu, managerul de proiect poate compara cele două instanțe pentru a determina ce modificări au fost făcute, unde și de către cine. Dacă managerul de proiect a decis că modificările suplimentare la serviciu nu au valoare, managerul de proiect poate reveni la instantaneul corecției inițiale.

Puteți rula versiuni diferite (instantanee) ale unei aplicații de proces simultan pe un server; când instalați un instantaneu nou, fie înlăturați originalul fie lăsați-l să ruleze.

Context de versiune

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

IBM Business Process Manager asignează un spațiu de nume global pentru fiecare aplicație de proces. Spațiul de nume global este în mod specific fie sugestia aplicației de proces, fie un instanțaneu de aplicație de proces particular. Numele de versiune utilizat de către server nu poate fi mai lung de șapte caractere, astfel încât numele alocat este un acronim care utilizează caractere din numele instanțaneului pe care l-ați alocat. Acronimele instanțaneului sunt identice cu numele instanțaneului conform stilului IBM VRM recomandat și nu sunt mai lungi de șapte caractere. De exemplu, un nume de instanțaneu de 1.0.0 va avea acronimul 1.0.0 și un nume de instanțaneu of 10.3.0 va avea acronimul 10.3.0. Acronimul instanțaneului va fi garantat ca și unicitate în contextul aplicației de proces din domeniul serverului Process Center. Pentru acest motiv, nu puteți edita acronimul instanțaneului.

Considerente de versionare pentru aplicații proces în clustere multiple

Puteți instala aceeași versiune de aplicație de proces la clustere multiple din aceeași celulă. Pentru a face diferența între aceste instalări multiple ale aceleiași versiuni a aplicației de proces, creați un instanțaneu pentru fiecare instalare și includeți un ID unic de celulă în numele instanțaneului (de exemplu, v1.0_cell1_1 și v1.0_cell1_2). Fiecare instanțaneu este o versiune nouă a aplicației de proces (dintr-o perspectivă de gestiune ciclului de viață pură), dar conținutul și funcția sunt la fel.

Când instalați o aplicație de proces într-un cluster, este realizată o sincronizare automată a nodurilor.

Considerente de versionare pentru truse de unelte Process Designer

Amintiți-vă că instanțaneele de aplicații proces sunt realizate în mod tipic când sunteți pregătit să testați sau să instalați. Instanțaneele de truse de unelte sunt însă realizate în mod tipic când sunteți pregătit ca acea trusă de unelte să fie utilizată de aplicații proces. După, dacă doriți să actualizați trusa de unelte, trebuie să faceți un alt instanțaneu al "sugestiei" atunci când sunteți pregătit și apoi deținătorii aplicațiilor de proces și ai truselor de unelte pot decide dacă doresc să treacă la noul instanțaneu.

Versionarea modulelor și bibliotecilor

Dacă un modul sau o bibliotecă este într-o aplicație de proces sau într-o trusă de unelte, acesta continuă ciclul de viață al aplicației de proces sau trusei de unelte (versiuni, instanțaneee, urme și așa mai departe). Numele modulelor și bibliotecilor trebuie să fie unice în cadrul domeniului unei aplicații de proces sau a unei truse de unelte.

Acest subiect descrie versionarea modulelor și a bibliotecilor care sunt folosite împreună cu aplicațiile de proces. Rețineți totuși că în cazul în care implementați modulele direct din IBM Integration Designer pe Process Server, puteți continua să urmați procedura de asignare a numerelor de versiune modulelor în timpul implementării, așa cum este descris în "Crearea modulelor și bibliotecilor cu număr de versiune".

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

Următorul tabel listează selecțiile pe care le puteți face în editorul de dependențe din IBM Integration Designer, atunci când o bibliotecă este asociată cu o aplicație de proces sau o trusă de unelte:

Tabela 36. Dependențele pentru Modul, Aplicație de proces sau Trusa de unelte și bibliotecile globale

Domeniu bibliotecă	Descriere	Poate depinde de . . .
Modul	Pe server există câte o copie a acestei biblioteci pentru fiecare modul care o folosește.	O bibliotecă din domeniul modului poate depinde de toate tipurile de biblioteci.

Tabela 36. Dependențele pentru Modul, Aplicație de proces sau Trusa de unelte și bibliotecile globale (continuare)

Domeniu bibliotecă	Descriere	Poate depinde de . . .
Aplicație de proces sau Trusă de unelte	Biblioteca este partajată între toate modulele din domeniul aplicației 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 Aplicație 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 configurați o bibliotecă partajată WebSphere pentru a putea implementa biblioteca globală. Vedeți “Dependențele modulelor și bibliotecilor ” pentru informații suplimentare.

Module și biblioteci asociate cu aplicații de proces sau cu truse de unelte

Nu aveți nevoie să versionați module și biblioteci asociate cu aplicații de proces sau cu truse de unelte.

Modulele și bibliotecile asociate cu aplicații de proces sau cu truse de unelte nu au nevoie să fie versionate. De fapt, nu puteți crea o versiune a unui modul sau bibliotecă asociate cu aplicații de proces sau cu truse de unelte în editorul de dependențe. Modulele și bibliotecile asociate cu aplicații de proces sau cu truse de unelte utilizează instanțane, o funcție în Process Center, pentru a realiza același rezultat ca o versiune.

Bibliotecile, asociate cu aplicații de proces sau cu truse de unelte, nu vor avea un număr de versiune necesar în secțiunea Bibliotecii a editorului de dependențe, deoarece nu este necesară nicio versiune.

Convenții de numire

O convenție de numire este folosită pentru a diferenția diversele versiuni ale unei aplicații de proces pe măsură ce se mută prin ciclul de viață de actualizare, implementarea, co-implementare, dezimplementare și arhivare.

Această secțiune vă oferă convențiile care sunt utilizate pentru a identifica în mod unic versiunile unei aplicații de proces.

Un *context de versiune* este o combinație de acronime care descriu în mod unic o aplicație de proces sau o trusă de unelte. Fiecare tip de acronim are o convenție de numire. Acronimul este limitat la o lungime maximă de șapte caractere din setul de caractere [A-Z0-9_], cu excepția acronimului instantaneului, care poate include un punct.

- Acronimul aplicației de proces este creat o dată cu aplicația. Acesta poate avea o lungime maximă de șapte caractere.
- Acronimul instantaneului este creat în mod automat la crearea instantaneului. Acesta poate avea o lungime maximă de șapte caractere.

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

Notă: Atunci când utilizați funcția de rutare dependentă de versiune pentru componenta fluxului de mediere, denumiți-vă instantaneul, astfel încât să fie în concordanță cu schema <versiune>.<ediție>.<modificare> (de exemplu, **1.0.0**). Deoarece acronimul instantaneului este limitat la șapte caractere, valorile cifrelor sunt limitate la un maxim de cinci cifre (cinci cifre plus două perioade). Prin urmare, trebuie să aveți grijă atunci când câmpurile cifre sunt incrementate deoarece orice oricare depășește primele șapte caractere este trunchiat.

De exemplu, un nume de instanțaneu **11.22.33** rezultă într-un acronim de instanțaneu 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 și acronimul implicit pentru pistă 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 definiție de proces operațional într-o aplicație de proces este identificată de obicei prin acronimul numelui procesului operațional, acronimul instanțaneului și numele definiției procesului operațional. Alegeți nume unice pentru definițiile procesului operațional ori de câte ori este posibil. Atunci când există nume duplicate, este posibil să întâmpinați următoarele probleme:

- S-ar putea să fiți în imposibilitatea de a expune definițiile procesului operațional sub formă de servicii web fără o anumită formă de mediere.
- S-ar putea să fiți în imposibilitatea de a invoca o definiție a procesului operațional creat în IBM Process Designer dintr-un proces BPEL creat în IBM Integration Designer.

Contextul de versionare variază în funcție de cum este implementată aplicația de proces.

Convenții de numire pentru implementarea pe servere Process Center

Pe serverul IBM Process Center, puteți implementa un instanțaneu al unei aplicații de proces, precum și un instanțaneu al unei truse de unelte. În plus, puteți implementa sugestia unei aplicații de proces sau sugestia unei truse de unelte. (O *sugestie* reprezintă versiunea curentă funcțională a aplicației dumneavoastră de proces sau a trusei de unelte.) Contextul de versionare variază în funcție de tipul implementării.

Pentru aplicațiile de proces, sugestia aplicației de proces sau instanțaneul specific aplicației de proces este utilizat pentru a identifica versiunea în mod unic.

Trusele de unelte pot fi implementate cu una sau mai multe aplicații de proces, dar ciclul de viață al fiecărei truse de unelte este legat de ciclul de viață al aplicației de proces. Fiecare aplicație 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 aplicațiile de proces.

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

Pentru mai multe informații, vedeți secțiunea “Exemple” la pagina 31, mai târziu în acest subiect.

Instanțanele ale aplicației de proces

Pentru implementarea instanțaneelor aplicațiilor de proces, contextul de versiune este o combinație între articolele următoare:

- Acronim pentru numele aplicației de proces
- Acronim pentru pista aplicației de proces (în cazul în care se folosește altă pistă decât **Main**)
- Acronim pentru instanțaneul aplicației de proces

Truse de unelte autonome

Pentru implementarea instanțaneelor truselor de unelte, contextul de versionare este o combinație între articolele următoare:

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

Sugestii

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

Pentru implementările sugestiilor pentru aplicația de proces, contextul de versionare este o combinație între articolele următoare:

- Acronim pentru numele aplicației de proces
- Acronim pentru pista aplicației de proces (în cazul în care se folosește 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 producție.

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

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

Exemple

Resursele ar trebui să fie numite în mod unic și 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 aplicației de proces folosește numele de pistă implicit (**Main**):

Tabela 37. Sugestie pentru aplicația de proces cu nume implicit pentru pistă

Tip de nume	Exemplu
Nume aplicație de proces	Process Application 1
Acronim pentru numele aplicației de proces	PA1
Pistă aplicație de proces	Main
Acronim pentru urma aplicației de proces	"" (când pista este Main)
Instantaneu aplicație de proces	
Acronim pentru instantaneul aplicației de proces	Type

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

Tabela 38. Module SCA și fișiere EAR dependentă de versiune

Nume modul SCA	Nume dependent de versiune	EAR dependent de versiune/nume aplicație
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 aplicație de proces care folosește un nume neimplicit pentru pistă:

Tabela 39. Sugestie pentru aplicația de proces cu nume neimplicit pentru pistă

Tip de nume	Exemplu
Nume aplicație de proces	Process Application 1
Acronim pentru numele aplicației de proces	PA1

Tabela 39. Sugestie pentru aplicația de proces cu nume neimplicit pentru pistă (continuare)

Tip de nume	Exemplu
Pistă aplicație de proces	Track1
Acronim pentru urma aplicației de proces	T1
Instantaneu aplicație de proces	
Acronim pentru instantaneul aplicației de proces	Type

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

Tabela 40. Module SCA și fișiere EAR dependentă de versiune

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

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

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

Tabela 41. Instantaneu pentru aplicația de proces cu nume implicit pentru pistă

Tip de nume	Exemplu
Nume aplicație de proces	Process Application 1
Acronim pentru numele aplicației de proces	PA1
Pistă aplicație de proces	Main
Acronim pentru urma aplicației de proces	"" (când pista este Main)
Instantaneu aplicație de proces	Process Shapshot V1
Acronim pentru instantaneul aplicației de proces	PSV1

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

Tabela 42. Module SCA și fișiere EAR dependentă de versiune

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

- Următorul tabel arată un exemplu de instantaneu de aplicație de proces care folosește un nume de pistă neimplicit:

Tabela 43. Instantaneu pentru aplicația de proces cu nume neimplicit pentru pistă

Tip de nume	Exemplu
Nume aplicație de proces	Process Application 1
Acronim pentru numele aplicației de proces	PA1
Pistă aplicație de proces	Track1
Acronim pentru urma aplicației de proces	T1
Instantaneu aplicație de proces	Process Snapshot V1
Acronim pentru instantaneul aplicației de proces	PSV1

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

Tabela 44. Module SCA și fișiere EAR dependentă de versiune

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

Convenții de numire pentru implementarea Process Server

Pe un Process Server, puteți implementa instanțele unei aplicații de proces. Acronimul instanțelor aplicației de proces este folosit pentru a identifica versiunea în mod unic.

Pentru implementarea instanțelor aplicațiilor de proces, contextul de versionare este o combinație între articolele următoare:

- Acronim pentru numele aplicației de proces
- Acronim instanțelor pentru aplicația de proces

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

Tabela 45. Exemple de nume și acronime

Tip de nume	Exemplu
Nume aplicație de proces	Process Application 1
Acronim pentru numele aplicației de proces	PA1
Instanțelor aplicație de proces	1.0.0
Acronim instanțelor pentru aplicația 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 și modul în care fișierele EAR asociate includ contextul de versiune:

Tabela 46. Module SCA și fișiere EAR dependente de versiune

Nume modul SCA	Nume dependent de versiune	EAR dependent de versiune/nume aplicație
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 aplicației de proces și modul în care fișierele JAR asociate includ contextul versiunii:

Tabela 47. Biblioteci din domeniul aplicației de proces și fișiere JAR dependente de versiune

Nume bibliotecă din domeniul aplicației 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

Legările dependente de versiune

Aplicațiile de proces pot conține module SCA care includ legări de import și export. Atunci când co-implementați aplicații, legarea pentru fiecare versiune a aplicației 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 actualizați legarea după implementare pentru a îi asigura unicitatea.

O legare *dependentă de versiune* este pusă în domeniul unei anumite versiuni ale aplicației de proces, lucru care garantează unicitatea între aplicațiile de proces. Următoarele secțiuni descriu legările care sunt actualizate automat pentru a fi dependente de versiune, precum și orice acțiune pe care aveți nevoie să o efectuați la momentul rulării atunci când o legare nu este dependentă de versiune. Pentru mai multe informații despre lucrurile care se iau în considerare atunci când creați module, vedeți “Considerente atunci când se utilizează legări”.

SCA

Ținta unei legări SCA este redenumită în mod automat pentru a fi dependentă de versiune în timpul implementării în cazul în care legările de import și export ale modulului sunt definite în același domeniu al aplicației de proces.

În cazul în care legările nu sunt definite în același domeniu al aplicației de proces, se înregistrează un mesaj de informare. Trebuie să modificați legarea de import după implementare pentru ca adresa țintă a punctului final să se modifice. Puteți utiliza consola administrativă pentru a modifica adresa țintă a punctului final.

Serviciu Web (JAX-WS sau JAX-RPC)

Adresa țintă 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:

- Ați urmat convenția implicită de numire pentru adresa:
`http://ip:port/ModuleNameWeb/sca/ExportName`
- Punctul final al adresei este SOAP/HTTP.
- Legările de import și export ale modulului sunt definite în același domeniu al aplicației de proces.

Dacă aceste condiții nu sunt adevărate, este înregistrat un mesaj de informare. Acțiunea pe care o întreprindeți apoi depinde de modul în care vă implementați aplicația de proces:

- Dacă vă co-implementați aplicația de proces, trebuie să redenumiți manual URL-ul punctului final pentru SOAP/HTTP sau coada de destinație SOAP/JMS, astfel încât să fie unic între versiunile aplicației de proces. Puteți utiliza consola administrativă după implementare pentru a modifica adresa țintă a punctului final.
- Dacă implementați doar o singură versiune a aplicației de proces, aveți posibilitatea să ignorați acest mesaj

Pentru co-implementarea instantaneului legării serviciului Web SOAP/ JMS, acțiunea pe care o întreprindeți depinde de modul în care vă implementați aplicația de proces:

- În cazul în care importul și exportul țintă sunt în aceeași aplicație de proces, realizați pașii următori înainte de a publica aplicația de proces pe Process Center și de a crea instantaneul:
 1. Modificați URL-ul punctului final pentru export. Asigurați-vă că destinația și fabrica de conexiuni sunt unice.
 2. Modificați URL-ul punctului final pentru import, astfel încât să fie același cu cel pe care l-ați specificat pentru export la pasul anterior.
- În cazul în care importul și exportul țintă sunt în aplicații de proces diferite, realizați pașii următori:
 1. Modificați URL-ul punctului final pentru export. Asigurați-vă că destinația și fabrica de conexiuni sunt unice.
 2. Publicați aplicația de proces la Process Center.
 3. Creați instantaneul.
 4. Implementați aplicația de proces pe Process Server.
 5. Utilizați consola administrativă WebSphere pentru a modifica URL-ul punctului final pentru importul corespunzător, astfel încât să fie același cu cel specificat pentru export.

HTTP

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

- Ați urmat convenția implicită de numire pentru adresa:
`http(s)://ip:port/ModuleNameWeb/contextPathInExport`
- Legările de import și export ale modulului sunt definite în același domeniu al aplicației de proces.

Dacă aceste condiții nu sunt adevărate, este înregistrat un mesaj de informare. Acțiunea pe care o întreprindeți apoi depinde de modul în care vă implementați aplicația de proces:

- Dacă co-implementați aplicația de proces, trebuie să redenumiți manual URL-ul punctului final, astfel încât să fie unic între versiunile aplicației de proces. Puteți utiliza consola administrativă după implementare pentru a modifica adresa țintă a punctului final.
- Dacă implementați doar o singură versiune a aplicației de proces, aveți posibilitatea să ignorați acest mesaj

JMS și Generic JMS

Legările JMS generate de sistem și legările generice JMS sunt conștiente de versiune în mod automat.

Notă: Pentru legările JMS definite de utilizator și legările Generic JMS, nu are loc redenumirea automată în timpul implementării pentru ca legările să devină conștiente de versiune. În cazul în care legarea este definită de utilizator, trebuie să redenumiți următoarele atribute, astfel încât să fie unice între versiunile aplicațiilor de proces:

- Configurarea punctului final
- Coadă destinație de recepție
- Nume port ascultător (dacă este definit)

Setați destinația Trimite corespunzătoare în cazul în care modificați punctul final al modulului țintă.

MQ/JMS și MQ

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

Trebuie să redenumiți atributele următoare, astfel încât să fie unice între versiunile aplicațiilor de proces:

- Configurarea punctului final
- Coadă destinație de recepție

Setați destinația Trimite corespunzătoare în cazul în care modificați punctul final al modulului țintă.

EJB

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

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

Rețineți că aplicațiile 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 (**`ModuleNameApp: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 aplicației de proces.

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

Invocarea dinamică dependentă de versiune

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

În cazul în care folosiți stilul IBM_VRM (<versiune>.<ediție>.<modificare>) pentru instantaneu, puteți exporta fișierul EAR al aplicației de proces în WSRR (WebSphere Service Registry and Repository). Atunci când creați modulul de mediere, configurați atunci și căutarea punctelor finale, astfel încât să folosească rutarea dependentă de versiune. De exemplu, selectați **Returnare punct final care se potrivește cu cea mai recentă versiune compatibilă pentru serviciile bazate pe modulul SCA** în câmpul **Politică potrivire**, apoi selectați **SCA** pentru **Tip legare**.

Versiunile viitoare ale aplicației de proces sunt implementate pe server și 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.

Rețineți ca o alternativă, că puteți seta ținta în SMOHeader, iar valoarea poate fi purtată de către mesajul de cerere.

Implementarea aplicațiilor de proces cu module și proiecte Java

Aplicațiile de proces pot conține module Java EE și proiecte Java personalizate. Atunci când co-implementați aplicații, modulul Java EE personalizat pentru fiecare versiune a aplicației trebuie să fie unic.

Rețineți că modulele Java EE și proiectele Java personalizate sunt implementate pe un server în cazul în care acestea sunt implementate cu un modul SCA care are o dependență declarată în acestea. Dacă nu selectați **Implementare cu modul** (care este implicit) atunci când declarați dependența, trebuie să implementați modulul sau proiectul în mod manual.

Implementarea aplicațiilor de proces cu ajutorul regulilor operaționale și a selectoarelor

În cazul în care implementați mai multe versiuni ale unei aplicații de proces care include o regulă operațională sau o componentă de tip selector, fiți atenți la modul în care versiunile folosesc metadatele asociate.

Metadatele dinamice pentru o regulă operațională sau pentru o componentă de tip selector sunt definite în momentul rulării prin numele componentei, numele spațiului pentru componenta țintă și tipul componentei. În cazul în care două sau mai multe versiuni ale aplicației de proces care conțin o regulă operațională sau un selector sunt implementate în același mediu de runtime, acestea vor partaja aceeași logică pentru reguli (regulă operațională) sau aceleași metadate (selector) pentru rutare.

Pentru a permite fiecărei versiuni a regulii operaționale sau a componentei de tip selector ce aparține de aplicația de proces să își utilizeze propriile metadate dinamice (logică pentru reguli sau rutare), modificați codul (refactorizați) pentru spațiul de nume țintă, astfel încât acesta să fie unic pentru fiecare versiune a aplicației de proces.

Obiecte de configurație

Puteți folosi comenzile AdminConfig ale uneltei de administrare în linie de comandă WebSphere (wsadmin) pentru a accesa și modifica baza de date și proprietățile de securitate din IBM Business Process Manager.

Termenul *obiect configurație* se referă la un obiect care este accesat prin folosirea comenzilor wsadmin AdminConfig. Pentru mai multe informații, vedeți Proprietățile configurației de securitate.

Arhitectura de implementare

Arhitectura de implementare IBM Business Process Manager conține procese software numite servere, unități topologice referite ca noduri și celule și magazia de configurare utilizată pentru memorarea informațiilor de configurare.

Celulele

În IBM Business Process Manager, *celulele* sunt grupări logice de unu sau mai multe noduri dintr-o rețea 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 funcție de criteriile specifice care au sens în mediile lor organizaționale.

Datele de configurare administrative sunt memorate în fișiere XML. O celulă reține fișiere de configurare master pentru fiecare server din fiecare nod din celulă. Fiecare nod și server are, de asemenea, propriile fișiere locale de configurare. Modificările fișierului local de configurare a unui nod sau a unui server sunt temporare dacă serverul aparține celulei. Când sunt efective, modificările locale înlocuiesc configurațiile celulei. Modificările fișierelor de configurare ale serverului master și ale nodului master făcute la nivelul celulei înlocuiesc orice modificare temporară făcută asupra nodului când documentele configurației celulei sunt sincronizate cu nodurile. Sincronizarea apare la evenimente desemnate, cum ar fi pornirea unui server.

Serverele

Serverele furnizează funcționalitatea de bază a IBM Business Process Manager. Servere Process extind sau măresc abilitatea unui server de aplicații de a manipula module SCA (Service Component Architecture). Alte servere (manageri de implementare și agenți 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 colecție de servere gestionate, cluster-e de servere și alte middleware-uri se numește un *mediu de implementare*. Într-un mediu de implementare, fiecare server sau cluster gestionat este configurat pentru o anumită funcție în mediul de implementare (de exemplu, gazdă destinație, gazdă modul de aplicație sau server Common Event Infrastructure). Un server autonom este configurat să furnizeze toate funcțiile cerute.

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

Un *agent de nod* este un agent administrativ ce reprezintă un nod din sistemul dvs. și gestionează serverele aceluși nod. Agenții de nod monitorizează serverele de pe un sistem de gazdă și 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 și cluster-e.

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

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 țintă de implementare, suportul de mesaje, managerul de reguli de proces operațional și un server Common Event Infrastructure.

Un server autonom este ușor de setat și are o consolă Pornire rapidă de unde puteți porni și opri serverul și puteți deschide galerie de eșantioane și consola administrativă. Dacă instalați eșantioanele IBM Business Process Manager și apoi deschideți galeria de eșantioane, o soluție exemplu este implementată în serverul autonom. Puteți explora resursele utilizate pentru acest eșantion din consola administrativă.

Puteți implementa propriile dumneavoastră soluții într-un server autonom, dar un server autonom nu poate furniza capacitatea, scalabilitatea sau robustețea care este cerută de un mediu de producție. Pentru mediul dumneavoastră de producție, este mai bine să utilizați un mediu Network Deployment.

Este posibil să porniți cu un server autonom și mai târziu să îl includeți într-un mediu Network Deployment, federalizându-l la o celulă a managerului de implementare, *asigurat că niciun alt nod nu a fost federalizat la aceea celulă*. Nu este posibil să federalizați mai multe servere autonome într-o celulă. Pentru a federaliza serverul autonom, utilizați consola administrativă a managerului de implementare sau comanda **addNode**. Serverul autonom nu trebuie să ruleze când îl federalizați 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 și care participă la gestiunea încărcării de lucru.

Un cluster poate conține noduri sau servere individuale de aplicații. Un nod este de obicei un calculator fizic cu o adresă IP a gazdei distinctă care rulează unu sau mai multe servere de aplicații. Cluster-ele pot fi grupate sub configurația unei celule care asociază logic multe servere și cluster-e cu configurații și aplicații diferite unul cu celălalt în funcție de discreția administratorului și de ceea ce are sens în mediile lor organizaționale.

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

Deoarece fiecare membru al cluster-ului conține aceleași aplicații, puteți distribui taskuri client în funcție de capacitățile diferitelor mașini prin asignarea de ponderi fiecărui server.

Asignarea de ponderi serverelor dintr-un cluster îmbunătățește performanța și preluarea la defect. Taskurile sunt asignate serverelor care au capacitatea de a realiza operațiile 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 aplicații care poate deveni supraîncărcat dacă sunt făcute prea multe cereri.

Profilurile

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

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

Utilizați Utilitarul de linie de comandă BPMConfig pentru a crea profiluri IBM BPM. Utilitarul linie de comandă **manageprofiles** sau interfața lui grafică de utilizator, Profile Management Tool (PMT), poate fi utilizat ca mod alternativ de a crea managerul de implementare sau profiluri de noduri gestionate. PMT nu mai este suportate pentru crearea unui profil autonom.

Notă: În platformele distribuite, fiecare profil are un nume unic. Pe z/OS, toate profilurile sunt numite “default”; nu puteți redenumi, edita, copia și șterge profiluri pe z/OS.

Tipuri de profiluri

Sunt disponibile următoarele tipuri de profiluri IBM BPM cu IBM Business Process Manager V8.5:

Profil autonom IBM BPM

Profilul autonom definește servere autonome cu capacități și funcționalitate specifice în configurațiile IBM BPM Express. Puteți crea profilul autonom utilizând șablonul de profil BPM/BpmServer, care este suportat numai în IBM BPM Express.

Profil de manager de implementare IBM BPM

Profilul de manager de implementare definește un manager de implementare care asigură o interfață administrativă pentru un grup logic de servere pe una sau mai multe spații de lucru. Puteți crea profilul de manager de implementare utilizând șablonul de profile BPM/BpmDmgr, care este suportat numai în IBM BPM Standard și IBM BPM Advanced.

Profil de nod gestionat IBM BPM

Profilul de nod gestionat definește un nod gestionat când nodul este federalizat la un manager de implementare. Puteți crea profilul de nod gestionat utilizând șablonul de profiluri BPM/BpmNode, care este suportat numai în IBM BPM Standard și IBM BPM Advanced.

Tipurile specifice de profiluri sunt disponibile cu fiecare configurație IBM BPM.

Tabela 48. Tipuri disponibile de profiluri IBM BPM

Configurarea IBM BPM	Tipuri de profiluri		
	Autonom	Managerul de implementare	Nod gestionat
IBM BPM Express	Da	Nu	Nu
IBM BPM Standard	Nu	Da	Da
IBM BPM Advanced	Nu	Da	Da
Unit Test Environment (UTE) for IBM Integration Designer	Da	Opțional	Opțional

Directorul profilului

Fiecare profil din sistem are propriul său director care conține toate fișierele sale. Specificați locația 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 Dmgr (manager de implementare) este în C:\Program Files\IBM\WebSphere\ProcServer\profiles\Dmgr.

Profilul implicit

Primul profil pe care îl puteți crea într-o instalare a IBM Business Process Manager este *profilul implicit*. Profilul implicit este destinația 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 creați un alt profil, puteți să îl faceți implicit.

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

Augmentare profiluri

Dacă aveți deja un profil de manager de implementare, un profil de nod gestionat sau un profil de server autonom creat pentru WebSphere Application Server Network Deployment, îl puteți *augmenta* pentru a suporta IBM Business Process Manager în plus față de funcția existentă. Pentru a augmenta un profil, instalați mai întâi IBM Business Process Manager. Apoi, utilizați utilitarul linie de comandă **manageprofiles** pentru a augmenta un profil autonom, sau utilizați fie Profile Management Tool fie utilitarul linie de comandă **manageprofiles** pentru a augmenta un profil de manager de implementare sau de nod gestionat.

Important: Într-un mediu Network Deployment, trebuie să augmentați mai întâi profilul de manager de implementare, și apoi augmentați profilurile de nod gestionat.

Restricție: Nu puteți augmenta un profil de manager de implementare sau autonom unde registrul de utilizatori WebSphere VMM implicit a fost modificat, de exemplu, să folosească LDAP.

Managerii de implementare

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

La crearea unui mediu de implementare, profilul managerului de implementare este primul profil pe care îl creați. Fiecare mediu de implementare pe care-l creați are o consolă Pornire rapidă (Quick Start), din care puteți porni și opri managerul de implementare și puteți porni consola sa administrativă. Folosiți consola administrativă a managerului de implementare pentru a gestiona serverele și cluster-ele din celulă. Aceasta include configurarea serverelor și a cluster-elor, adăugarea serverelor la cluster-e, pornirea și oprirea serverelor și a cluster-elor și implementarea modulelor SCA.

Deși managerul de implementare este un tip de server, nu puteți implementa module pe însuși 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 configurația sa și 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 și care conține agent nod și poate conține servere gestionate. Într-un nod gestionat, puteți configura și rula serverele gestionate.

Serverele care sunt configurate pe un nod gestionate formează resursele mediului dumneavoastră de implementare. Aceste servere sunt create, configurate, pornite, oprite, gestionate și șterse 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 configurația profilului. De exemplu, când realizați următoarele taskuri:

- Porniți și opriți procesele serverului.
- Sincronizați datele de configurare de pe managerul de implementare cu copia de pe nod.

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

Un nod gestionat poate conține unul sau mai multe servere, care sunt gestionate de un manager de implementare. Puteți implementa soluții pentru servere într-un nod gestionat, dar nodul gestionat nu conține o galerie de eșantioane de aplicații. Nodul gestionat este definit de un profil de nod gestionat și are o consolă Pornire rapidă (Quick Start).

Nodurile negestionate

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

Nodurile negestionate din topologia Network Deployment pot avea definiții de servere cum ar fi servere web, dar nu definiții de servere de aplicații. 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.

Agenții de nod

Agenții de nod sunt agenți administrativi care rutează cereri administrative la servere.

Un agent de nod este un server care rulează pe fiecare calculator gazdă care participă la configurația Network Deployment. Este pur și simplu un agent administrativ și nu este implicat în funcții care deserveșc aplicații. Un agent de nod găzduiește, de asemenea, alte funcții administrative importante precum servicii de transfer de fișiere, sincronizare de configurare și monitor de performanță.

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

Acest subiect discută termeni rezervați și probleme pe care trebuie să le luați în considerare la numirea profilului, nodului, serverului, gazdei și 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 restricții. Nu folosiți niciunul din următoarele caractere când vă denumiți profilul:

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

Sunt permise caractere pe doi octeți.

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: Folosiți o convenție de numire cale scurtă când creați 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 și celulă

Nume rezervate: Evitați 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 și nume de gazde și nod, Gazdă și Nume celule: Utilizați indicațiile corespunzătoare pentru a denumi atunci când creați profiluri.

- Profilurile serverului autonom
- Profilurile managerului de implementare
- Profile de noduri gestionate

Tabela 49. Indicații de denumire pentru profiluri de server autonome

Nume câmp	Valoare implicită	Constrângeri	Descriere
Nume nod	<p>Linux UNIX</p> <p>Windows <i>shortHostName</i> Node <i>NodeNumber</i> unde:</p> <ul style="list-style-type: none"> • <i>shortHost Name</i> este numele scurt al gazdei. • <i>NodeNumber</i> este un număr secvențial care pornește de la 01. 	Evitați folosirea numelor rezervate.	Selectați orice nume doriți. Pentru a ajuta organizarea instalației dumneavoastră, folosiți un nume unic dacă aveți de gând să creați mai mult de un server pe sistem.
Nume server	<p>Linux UNIX</p> <p>Windows server1</p>	Folosiți un nume unic pentru server.	Numele logic pentru server.
Nume de gazdă	<p>Linux UNIX</p> <p>Windows Forma lungă a numelui serverului de nume domeniu (DNS - domain name server).</p>	Folosiți un nume de gazdă complet calificat care este adresabilă prin rețea.	Folosiți numele DNS actual sau adresa IP a stației dumneavoastră de lucru pentru a permite comunicația cu ea. Vedeți informații suplimentare despre numele gazdă urmând această tabelă.

Tabela 50. Indicații de denumire pentru profiluri de manager de implementare

Nume câmp	Valoare implicită	Constrângeri	Descriere
Nume nod	<p>Linux UNIX</p> <p>Windows <i>shortHostName</i> Cell <i>ManagerNode Number</i> unde:</p> <ul style="list-style-type: none"> • <i>shortHost Name</i> este numele scurt al gazdei. • <i>NodeNumber</i> este un număr secvențial care pornește de la 01. 	Folosiți un nume unic pentru managerul de implementare. Evitați folosirea numelor rezervate.	Numele este folosit pentru administrarea în cadrul celei manager de implementare.
Nume de gazdă	<p>Linux UNIX</p> <p>Windows Forma lungă a numelui serverului de nume domeniu (DNS - domain name server).</p>	Folosiți un nume de gazdă complet calificat care este adresabilă prin rețea. Evitați folosirea numelor rezervate.	Folosiți numele DNS actual sau adresa IP a stației dumneavoastră de lucru pentru a permite comunicația cu ea. Vedeți informații suplimentare despre numele gazdă urmând această tabelă.

Tabela 50. Indicații de denumire pentru profiluri de manager de implementare (continuare)

Nume câmp	Valoare implicită	Constrângeri	Descriere
Nume celulă	<p>Linux UNIX</p> <p>Windows <i>shortHostName</i> Cell <i>CellNumber</i> unde:</p> <ul style="list-style-type: none"> <i>shortHost Name</i> este numele scurt al gazdei. <i>CellNumber</i> este un număr secvențial care pornește de la 01. 	<p>Folosiți un nume unic pentru celula manager de implementare. Un nume de celulă trebuie să fie unic în orice circumstanțe în care rulează produsul pe aceeași stație de lucru fizică sau cluster de stații de lucru, precum un Sysplex. În plus, un nume de celulă trebuie să fie unic în orice circumstanțe în care conectarea la rețea între entități 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 spațiile lor de nume vor fi federalizate. Altfel, ați putea întâlni excepții precum <code>javax.naming.NameNotFoundException</code>, caz în care trebuie să creați celule unic numite.</p>	<p>Toate nodurile federalizate devin membri ai celulei manager de implementare, pe care ați numit-o în pagina Nume de noduri, gazde și celule a Profile Management Tool.</p>

Tabela 51. Indicații de denumire pentru profiluri de noduri gestionate

Nume câmp	Valoare implicită	Constrângeri	Descriere
Nume nod	<p>Linux UNIX</p> <p>Windows <i>shortHostName</i> Node <i>NodeNumber</i> unde:</p> <ul style="list-style-type: none"> <i>shortHost Name</i> este numele scurt al gazdei. <i>NodeNumber</i> este un număr secvențial care pornește de la 01. 	<p>Evitați folosirea numelor rezervate.</p> <p>Folosiți un nume unic în cadrul celulei manager de implementare.</p>	<p>Numele este folosit pentru administrarea în cadrul celulei manager de implementare la care este adăugat profilul de nod gestionat. Folosiți un nume unic în cadrul celulei manager de implementare.</p>
Nume de gazdă	<p>Linux UNIX</p> <p>Windows Forma lungă a numelui serverului de nume domeniu (DNS - domain name server).</p>	<p>Folosiți un nume de gazdă complet calificat care este adresabilă prin rețea.</p>	<p>Folosiți numele DNS actual sau adresa IP a stației dumneavoastră de lucru pentru a permite comunicația cu ea. Vedeți informații suplimentare despre numele gazdă urmând această tabelă.</p>

Considerente privind numele gazdă:

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

IBM Business Process Manager este compatibil cu ambele versiuni ale Internet Protocol versiunea 4 (IPv4) și versiunea 6 (IPv6). Oricând puteți introduce adresele IP în consola administrativă sau în altă parte, puteți face aceasta în orice format. Rețineți că dacă IPv6 este implementat pe sistemul dumneavoastră, trebuie să introduceți adresa IP în formatul

IPv6 și, invers dacă IPv6 nu vă este încă disponibilă, introduceți adresele IP în formatul IPv4. Pentru mai multe informații despre IPv6, referiți-vă la descrierea următoare: IPv6.

Următoarele indicații vă pot ajuta la determinarea numelui de gazdă corespunzător pentru stația dumneavoastră de lucru:

- Selectați un nume de gazdă la care pot ajunge alte stații de lucru din rețeaua dumneavoastră.
- Nu folosiți identificatorul generic, localhost, pentru această valoare.
- Nu încercați să instalați produse IBM Business Process Manager pe un server cu un nume de gazdă care folosește caractere din setul de caractere pe doi octeți (DBCS - double-byte character set). Caracterele DBCS nu sunt suportate când sunt folosite în numele gazdă.
- Evitați folosirea caracterului liniuță de subliniere (_) în numele de server. Standardele Internet obligă ca numele de domenii să se conformeze cerințelor numelor gazdă descrise în Internet Official Protocol Standards RFC 952 și RFC 1123. Numele de domenii trebuie să conțină doar litere (majuscule sau minuscule) și cifre. Numele de domenii pot conține de asemenea caractere liniuță (-), atâta timp cât liniuțele nu se află la sfârșitul numelui. Caracterele liniuță de subliniere (_) nu sunt suportate în numele gazdă. Dacă ați instalat IBM Business Process Manager pe un server cu un caracter liniuță de subliniere în numele său, accesați serverul prin adresa sa IP până îl redenumiți.

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

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

- Șirul numelui gazdă DNS complet calificat, precum xmachine.manhattan.ibm.com
- Șirul 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 și flexibil. Aveți flexibilitatea de a schimba adresa IP actuală pentru sistemul gazdă, fără a schimba configurația serverului. Această valoare pentru numele gazdă este folosită în special dacă doriți să schimbați frecvent adresa IP când folosiți 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 fișierul gazdelor locale, pentru ca sistemul să poată rula serverul chiar când este deconectat de la rețea. Definiți numele scurt la 127.0.0.1 (loopback-ul local) în fișierul 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 distanță. 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 distanță se poate conecta la nodul pe care îl numiți 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 modificați setările proprietății hostName în documentele de configurare de fiecare dată când modificați adresa IP a stației de lucru. Prin urmare, nu folosiți o adresă IP numerică dacă utilizați DHCP sau dacă modificați regulat adresele IP. Un alt dezavantaj al acestui format este acela că nu puteți folosi nodul dacă gazda este deconectată de la rețea.

BPMN 2.0

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

BPMN (Business Process Model and Notation) este standardul de bază pentru procese din IBM Process Designer și IBM Process Center. Diagramele BPD (Business process definition) sunt bazate pe specificația BPMN. Acest subiect

introduce unele dintre modalitățile în care BPMN 2.0 este aplicat în IBM Business Process Manager. Pentru informații detaliate despre BPMN, vedeți pagina Specificații BPMN la <http://www.bpmn.org/>.

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

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

Evenimentele mesaj intermediar IBM BPM oferă funcții asemănătoare cu taskul de trimitere BPMN și 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 și afișate în diagrame de proces. Pictogramele afișează dacă activitatea dumneavoastră este un task de sistem, un task utilizator, un task de decizie, script sau proces legat. Activitățile din modelele care au fost create în versiuni anterioare afișează, de asemenea, tipuri de taskuri BPMN 2.0 corespunzătoare și pictograme de taskuri la vizualizarea lor în versiunea 7.5.1 sau mai târziu.

Activități și taskuri

Există câteva modificări de terminologie de la versiunile anterioare ale Process Designer. Un număr al acestor modificări implică tipuri de activități care au fost redenumite.

- Activitățile serviciu (automate) sunt acum taskuri sistem.
- Activitățile serviciu (task) într-un culoar non-sistem sunt acum taskuri utilizator.
- Activitățile serviciu (task) într-un culoar sistem sunt acum taskuri de decizie dacă facă referire la un serviciu de decizie.
- Activitățile serviciu (task) într-un culoar sistem sunt acum taskuri sistem dacă fac referire la orice tip de serviciu altul decât serviciu de decizie.
- Activitățile Javascript sunt acum taskuri script.
- Activitățile proces imbricat sunt acum procese legate.
- Activitățile 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 notație în gateway-urile din versiunile anterioare. Totuși, 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 condițional 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 graniță întrerupe activitatea de care este atașată. Atunci când evenimentul este declanșat, activitatea se oprește iar jetonul continuă secvența ieșire a evenimentului. Dacă evenimentul este setat ca non-întrerupere atunci când evenimentul este declanșat

activitatea atașată continuă în paralel și un nou jeton este generat și este transmis în fluxul secvență de ieșire al evenimentului. Granița evenimentului se modifică într-o linie întreruptă pentru evenimente care nu întrerup.

Evenimentele intermediare care sunt atașate activităților sunt evenimente intermediare de întrerupere dacă închid activitățile atașate lor sau evenimente intermediare non-întrerupere dacă nu închid activitățile atașate lor.

Eveniment de start

Specificația BPMN permite modelor de proces pentru să omită simbolurile de evenimente de start și de sfârșit. Process Designer necesită ca modelele de proces să folosească evenimente de start și de sfârșit.

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

proces

- fără
- mesaj
- ad hoc

subproces

- fără

subproces eveniment

- eroare
- mesaj
- cronometru

Puteți modifica tipul unui eveniment de start prin editarea proprietăților evenimentului. Puteți avea numeroase evenimente de start mesaj într-un proces, dar puteți folosi doar un eveniment fără start.

Oprire evenimente

Sunt disponibile patru tipuri de evenimente: *mesaj*, *terminare*, *eroare* și *none*. Puteți modifica tipul unui eveniment de sfârșit.

Când un proces părinte apelează un proces copil și procesul copil o acțiune de terminare eveniment, procesul copil se oprește și procesul părinte continuă apoi cu următorii pași.

Subproces

Specificația BPMN definește două tipuri de subproces, înglobate și refolosibile. Puteți crea ambele tipuri în Process Designer. Subprocesele înglobate sunt numite doar *subproces* în Process Designer și 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 conține și este o modalitate de grupare a pașilor procesului pentru a reduce complexitatea diagramei și confuzia. Subprocesele restrâng mai mulți pași într-o singură activitate. Subprocesul poate fi văzut doar de către procesul în care este definit. Un subproces există în domeniul apelantului său și are acces la toate variabilele din acel mediu. Nu există niciun parametru care să fie transmis în și în afara subprocesului înglobat.

Separat de subproces și 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 activități prin flux de secvențe, și apare doar dacă evenimentul de start este declanșat.

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 definiții de proces

pot de asemenea apela acest proces. Procesul legat definește parametrii de intrare și de ieșire și nu are niciun acces la domeniul sau mediul apelantului. Procesul legat este similar cu procesul imbricat disponibil în versiunile anterioare; nu există nicio modificare în comportamentul activității. Procesele anterioare imbricate sunt migrate la procese legate. Procesul legat arată ca un subproces cu o limită compactă și este evidențiat în fereastra Inspector.

Bucle

BPMN oferă noțiunea de activitate care poate fi repetată. Activitatea poate fi atomică, însemnând că activitatea se repetă, sau poate fi un subproces, încapsulând o serie de pași care se repetă. Dacă expandați activitatea repetată, vedeți activitățile conținute care urmează să fie rulate în mod repetat. Condiția este evaluată întotdeauna la începutul fiecărei iterații buclă. Nu există nicio abilitate de a evalua la sfârșitul fiecărei iterații buclă.

IBM Business Process Manager are o *buclă cu mai multe instanțe* care este rulată de un număr finit de ori cu activitățile conținute în cadrul rulării secvențiale sau în paralel.

Import procese non-BPMN

Puteți importa modele care au fost create în IBM WebSphere Business Modeler și le puteți utiliza în Process Designer. Pentru informații despre importul BPMN 2.0, vedeți elementele Mapare IBM WebSphere Business Modeler în construcțiile IBM Business Process Manager. De asemenea, puteți importa modele BPMN 2.0 care au fost create în IBM WebSphere Business Compass, Rational Software Architect sau alte medii de modelare.

Definiții de proces operațional (BPD-uri)

Pentru a modela un proces în IBM Process Designer, trebuie să creați o definiție a procesului operațional (BPD). Definiția de proces operațional (BPD) poate fi bazată pe un model BPMN importat.

Un BPD este un model reutilizabil al unui proces, definind ceea ce este comun tuturor instanțelor runtime ale aceluși model de proces. Un BPD trebuie să conțină un eveniment de start, un eveniment sfârșit, cel puțin un strat, și una sau mai multe activități. Consultați "Convențiile de numire IBM Process Designer" în legăturile înrudite pentru detalii despre limitarea caracterului care se aplică BPD-urilor.

Definiția de proces operațional (BPD) trebuie să includă un culoar pentru fiecare sistem sau grup de utilizatori care participă într-un proces. Un culoar poate fi un culoar participant sau un culoar sistem. Totuși puteți crea un BPD care grupează activitățile unui grup și a unui sistem într-un singur culoar dacă acea este preferința dumneavoastră. Consultați "Crearea unei definiții a procesului operațional (BPD)" în legăturile înrudite pentru informații despre cum se creează un BPD.

Puteți desemna orice persoană sau grup specific pentru a fi responsabil pentru activități într-un strat participant. Fiecare culoar pe care-l creați este atribuit implicit grupului Toți utilizatorii. Puteți folosi acest grup pentru a rula și testa definiția de proces operațional (BPD) în Inspector. Grupul Toți utilizatorii include toți utilizatorii care sunt membri grupului de securitate **tw_allusers**, care este un grup de securitate special care include automat toți utilizatorii din sistem.

Un strat sistem conține activități manipulate de un sistem IBM Process Center specific. Fiecare activitate necesită o implementare, care definește activitatea și seturile de proprietăți pentru task. În timpul implementării, un dezvoltator creează un serviciu sau scrie JavaScript necesar pentru a finaliza activitățile din stratul sistem. Consultați "Înțelegerea tipurilor de serviciu" în legăturile înrudite pentru informații despre servicii.

Pentru fiecare BPD creat, trebuie să declarați variabile pentru a captura datele de afaceri care sunt transmise de la o activitate la altă activitate în procesul dumneavoastră. Consultați "Gestionarea și maparea variabilelor" în legăturile înrudite pentru a învăța despre implementarea variabilelor.

Puteți de asemenea să adăugați evenimente unui BPD. Evenimentele IBM BPM pot fi determinate de trecerea unei date de scadență, de o excepție sau primirea unui mesaj. Declanșatorul dorit determină tipul de eveniment pe care îl alegeți să-l implementați. Pentru informații detaliate despre tipurile de evenimente disponibile și despre declanșatorii lor, consultați "Modelarea evenimentelor".

Legările

La nucleul unei arhitecturi orientată-pe-servicii este conceptul unui *serviciu*, o unitate de funcționalitate realizată de o interacțiune între dispozitive de calcul. Un *export* definește interfața externă (sau punctul de acces) a unui modul, astfel încât componentele SCA (Service Component Architecture) din modul să-și poată oferi serviciile clienților externi. Un *import* definește o interfață către serviciile din afara modulului, astfel încât serviciile să poată fi apelate din modul. Utilizați *legări* specifice-protocolului cu exporturi și importuri pentru a specifica mijloacele de a transporta datele în și din modul.

Exporturile

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

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

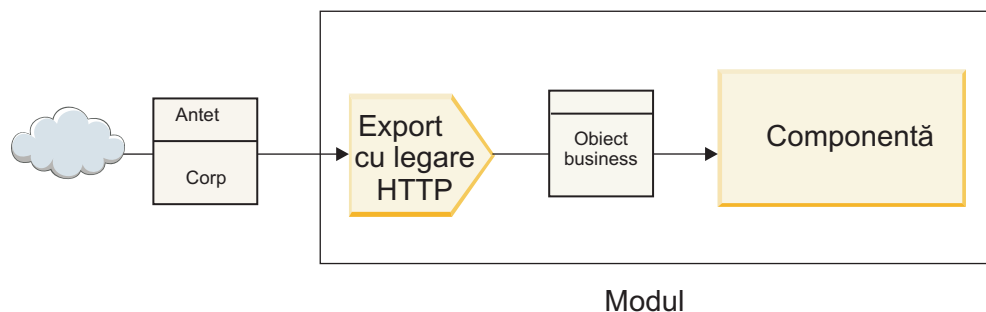


Figura 46. Un export cu legare HTTP

Importuri

O componentă SCA ar putea dori să invoce un serviciu extern non-SCA ce așteaptă 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 țintă în modul așteptat 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 așteptat de serviciu și serviciul este invocată.

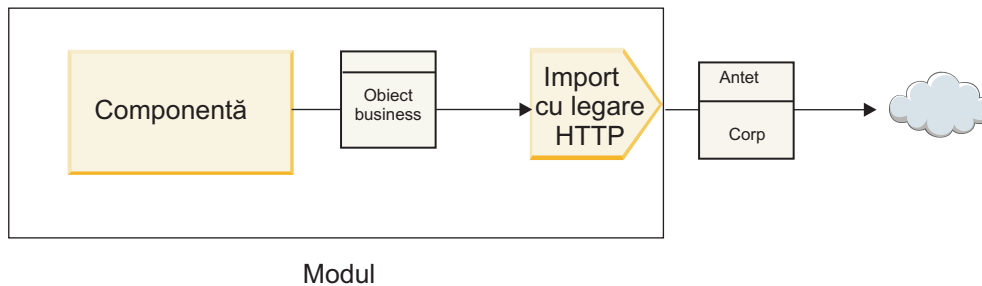


Figura 47. Un import cu legare HTTP

Listă cu legări

Utilizați Integration Designer pentru a genera o legare pentru un import sau un export și 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. Utilizați un import cu legare SCA pentru a accesa un serviciu dintr-un alt modul SCA. Utilizați un export cu legare SCA pentru a oferi un serviciu altor module SCA.
- Serviciu Web

O legare de serviciu web vă permite să accesați un serviciu extern folosind mesajele SOAP interoperabile și calitățile de serviciu. Puteți folosi, de asemenea, legările serviciului web pentru a include atașamente 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 transmite mesajele SOAP, legările serviciului manipulează întotdeauna interacțiunile cerere/răspuns în mod sincron.
- HTTP

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

Legările EJB permit componentelor SCA să interacționeze cu servicii furnizate de logica operațională 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 accesați servicii de pe un sistem de informații de întreprindere sau să vă faceți serviciile disponibile EIS-ului.
- Legări JMS

Legările Java Message Service (JMS), Generic JMS și WebSphere MQ JMS (MQ JMS) sunt folosite pentru interacțiuni cu sisteme de mesagerie, unde comunicarea asincronă prin cozi de mesaje este critică pentru fiabilitate. Un export cu una din legările JMS urmărește coada pentru sosirea unui mesaj și trimite asincron răspunsul, dacă există, cozii de răspunsuri. Un import cu una din legările JMS construiește și trimite un mesaj unei cozi JMS și urmărește o coadă pentru sosirea răspunsului, dacă există.

 - JMS

Legarea JMS vă lasă să accesați furnizorul JMS încorporat în WebSphere.
 - Generic JMS

Legarea Generic JMS vă lasă să accesați un sistem de mesagerie a unui vendor non-IBM.
 - MQ JMS

Legarea MQ JMS vă lasă să accesați subsetul JMS al unui sistem de mesagerie WebSphere MQ. Veți folosi această legare când subsetul JMS de funcții este suficient pentru aplicația dumneavoastră.

- MQ

Legarea WebSphere MQ vă lasă să comunicați cu aplicații native MQ, aducându-le în cadrul de lucru al arhitecturii orientate spre servicii și furnizând acces la informații de antet specifice MQ. Veți folosi această legare când aveți nevoie să folosiți funcții native MQ.

Privire generală asupra legării de export și import

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

Fluxul de informații printr-un export

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

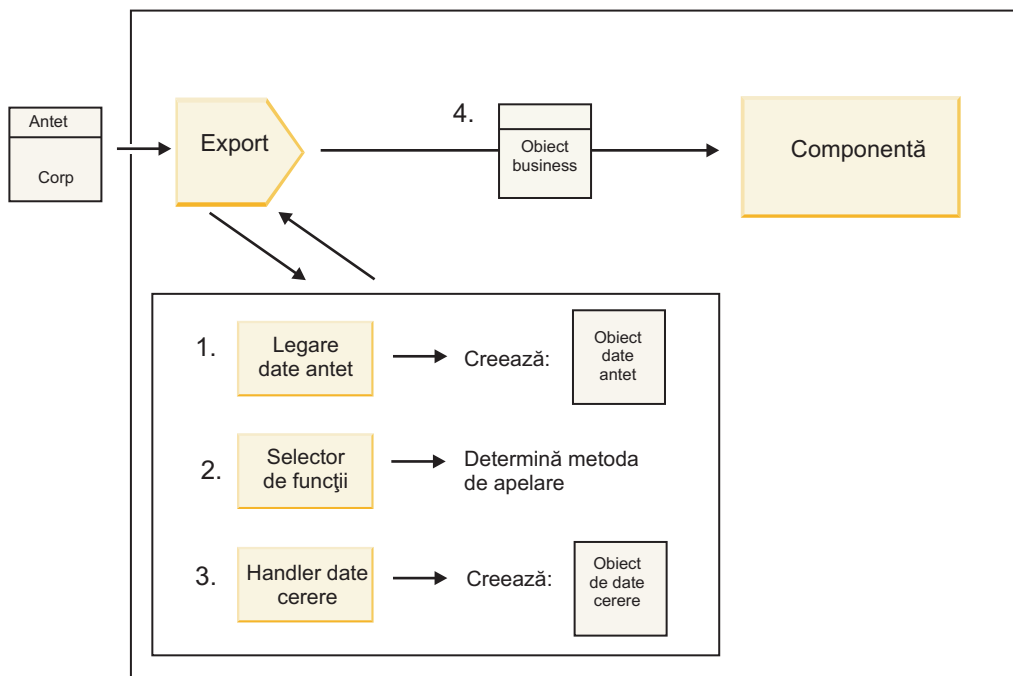


Figura 48. Fluxul unei cereri prin export către o componentă

Când exportul primește cererea, are loc următoarea secvență 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 funcții determină numele metodei native din mesajul de protocol. Numele metodei native este mapat de către configurația exportului de numele unei operații de pe interfața 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 și legarea de export EJB invocă componenta SCA în mod sincron.
 - Legările de export JMS, Generic JMS, MQ JMS și WebSphere MQ invocă componeta SCA asincron.

Notăți că un export poate propaga anteturile și proprietățile de utilizator pe care le primește prin protocol, dacă propagarea contextului este activată. Componentele cablate de export pot accesa apoi aceste anteturi și proprietăți de utilizator. Vedeți subiectul “Propagare” din centrul de informare WebSphere Integration Developer pentru informații suplimentare.

Dacă aceasta este o operație pe două direcții, componenta returnează un răspuns.

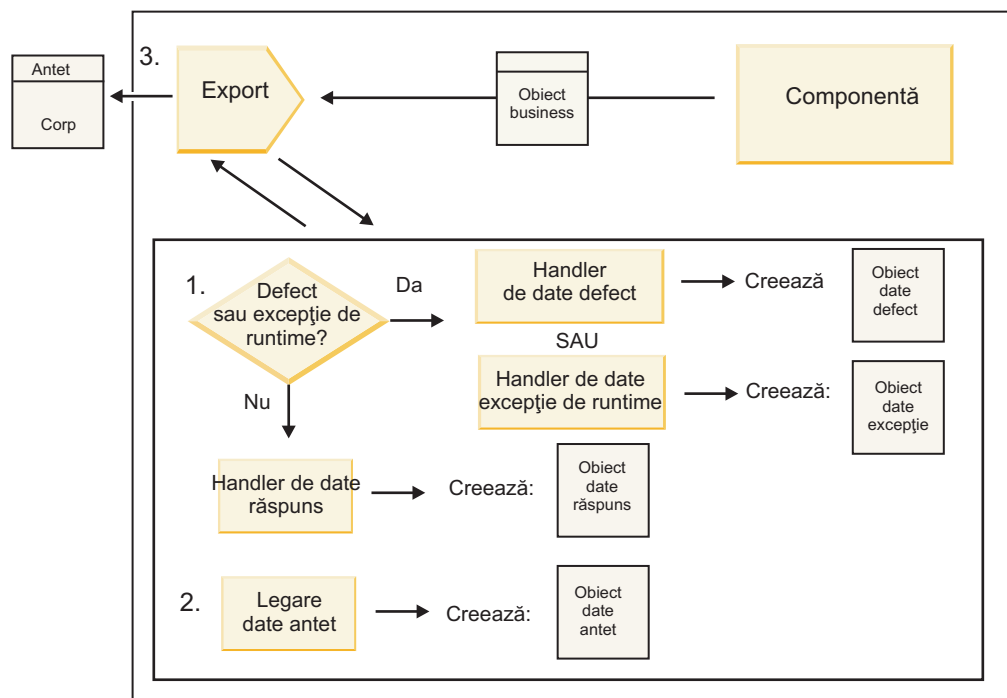


Figura 49. Fluxul unui răspuns înapoi prin export

Are loc următoarea secvență de pași:

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 excepție din timpul rulării, handler-ul de date pentru excepții 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 informațiilor printr-un import

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

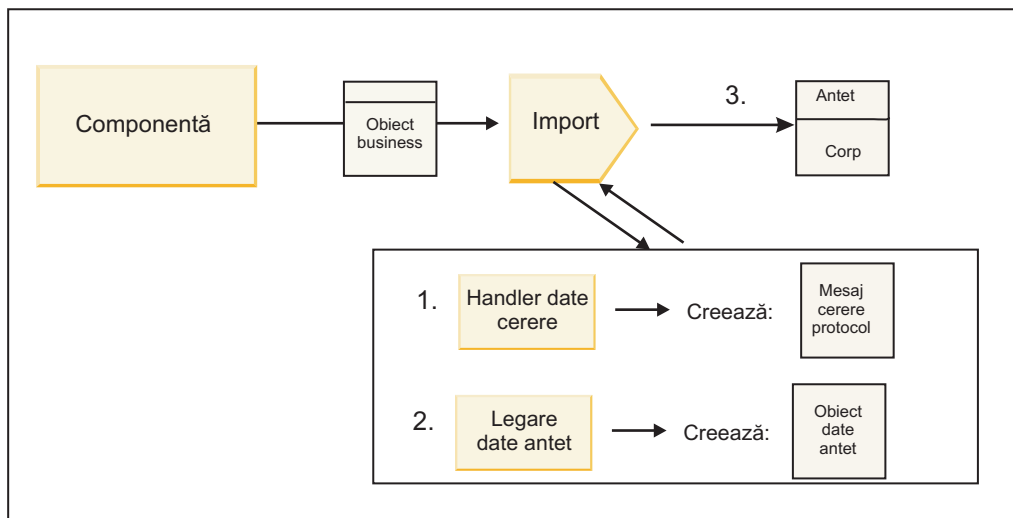


Figura 50. Fluxul de la o componentă prin import către un serviciu

Componenta invocă importul cu un obiect business de cerere.

Notă:

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

După ce componenta invocă importul, are loc următoarea secvență 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 operație pe două-căi, serviciul returnează un răspuns și are loc următoarea secvență de pași.

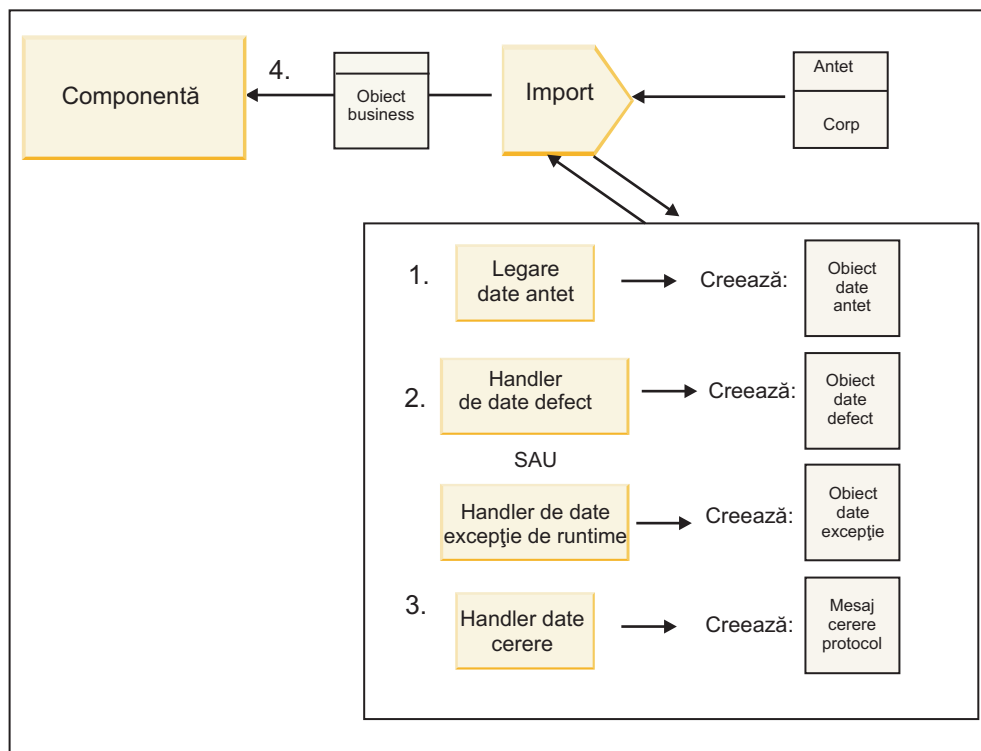


Figura 51. Fluxul unui răspuns înapoi prin import

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 excepție din timpul rulării, handler-ul de date pentru excepții 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 și import

Unul din aspectele cheie ale legărilor de export și 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, puteți de asemenea specifica un selector de funcții care să indice ce operație ar trebui efectuată pe date. Pentru legări asociate cu exporturi sau importuri, puteți indica cum ar trebui tratate defectele ce au loc în timpul procesării.

În plus, specificați informații specifice transportului pe legări. De exemplu, pentru o legare HTTP, specificați URL-ul punctului final. Pentru legarea HTTP, informațiile specifice transportului sunt descrise în subiectele “Generarea unei legări de import HTTP” și “Generarea unei legări de export HTTP”. Puteți de asemenea găsi informații despre alte legări în centrul de informare.

Transformarea formatului de date în importuri și exporturi

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

- Pentru legări de export, unde o aplicație client trimite cereri către și primește răspunsuri de la o componentă SCA, indicați formatul datelor native. În funcție 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) și invers pentru a transforma obiectul business în date native (care este răspunsul către aplicația client).

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

IBM Business Process Manager oferă un set de formate de date predefinite și handler-e de date sau legări de date corespunzătoare ce suportă formatele. Vă puteți de asemenea crea propriile handler-e de date personalizate și înregistrați formatul de date pentru acele handler-e de date. Pentru informații suplimentare, vedeți 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 și 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 și COBOL) într-un obiect business și un obiect business în date native. Deoarece sunt neutre din punct de vedere al protocolului, puteți reutiliza același handler de date cu o varietate de legări de export și import. De exemplu, puteți folosi același 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 (și 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 și HTTPServiceGatewayDataBinding) sunt depreciate începând cu IBM Business Process Manager Versiunea 7.0. Folosiți handler-e de date oricând este posibil.

După cum s-a menționat mai devreme, puteți crea handler-e de date personalizate, dacă este necesar. Puteți de asemenea legări de date personalizate; totuși, este recomandat să creați handler-e de date personalizate deoarece pot fi folosite peste multiple legări.

Handler-e de date:

Handler-ele de date sunt configurate funcție de legările de export și 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 puteți de asemenea crea propriul handler de date, dacă este necesar. Puteți asocia un handler de date cu o legare de export sau import la unul din două niveluri: îl puteți asocia cu toate operațiile din interfața exportului sau importului sau îl puteți asocia cu o anumită operație pentru cerere sau răspuns.

Handler-e de date predefinite

Utilizați IBM Integration Designer pentru a specifica handler-ul de date pe care vreți să îl utilizați.

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

Notă: Cu excepția notificărilor, aceste handler-e de date pot fi utilizate cu legări JMS, Generic JMS, MQ JMS, WebSphere MQ și HTTP.

Vedeți subiectul “Handler-e de date” din centrul de informare Integration Designer pentru informații mai detaliate.

Tabela 52. Handler-e de date predefinite

Handler de date	Date native în obiect business	Obiect business în 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 52. Handler-e de date predefinite (continuare)

Handler de date	Date native în obiect business	Obiect business în date native
Lățime fixată	Parsează date cu lățime fixată într-un obiect business.	Serializează un obiect business în date cu lățime 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 specificația Java Architecture for XML Binding (JAXB).	Deserializează un obiect de business în bean-uri Java folosind regulile de mapare definite de specificația 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 excepție într-un obiect business răspuns folosind regulile de mapare definite de specificația Java API for XML Web Services (JAX-WS).	Utilizat de o legare EJB pentru a transforma un obiect business în parametri Java de ieșire folosind regulile de mapare definite de specificația JAX-WS.
JSON	Parsează date JSON într-un obiect business.	Serializează un obiect business în date JSON.
Corp nativ	Parsează octeții, textul, maparea, fluxul sau obiectul nativ în unul din cinci obiecte business de bază (text, octeți, mapare, flux sau obiect).	Transformă cele cinci obiecte business de bază în octet, text, mapare, flux sau obiect.
SOAP	Parsează mesajul SOAP (și 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

Informații 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 și 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 puteți de asemenea crea propria legare de date, dacă este necesar. Puteți asocia o legare de date cu o legare de export sau import la unul din două niveluri, o puteți asocia cu toate operațiile din interfața exportului sau importului, sau o puteți asocia cu o operație specifică pentru cerere sau răspuns.

Utilizați IBM Integration Designer pentru a specifica legarea de date pe care vreți să o utilizați sau pentru a crea propria dumneavoastră legare de date. O discuție despre crearea legărilor de date poate fi găsită în secțiunea “Privire generală asupra JMS, MQ JMS și legările Generic JMS” 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 53. Legări de date predefinite pentru legări JMS

Legare de date	Date native în obiect business	Obiect business în date native
Obiect Java serializat	Transformă obiectul serializat Java într-un obiect business (care este mapat ca tipul de intrare sau ieșire în WSDL).	Serializează un obiect business în obiectul serializat Java în mesajul obiect JMS.
Octeți înfășurați	Extrage octeții din mesajul de octeți JMS de intrare și îi înfășoară în obiectul business JMSBytesBody.	Extrage octeții din obiectul business JMSBytesBody și îi înfășoară în mesajul de octeți JMS de ieșire
Intrare mapare înfășurată	Extrage informațiile despre nume, valoare și tip pentru fiecare intrare din mesajul de mapare JMS de intrare și creează o listă cu obiecte business MapEntry. Înfășoară apoi lista în obiectul business JMSMapBody	Extrage informațiile despre nume, valoare și tip din lista MapEntry din obiectul business JMSMapBody și creează intrările corespunzătoare în mesajul de mapare JMS de ieșire.
Obiect înfășurat	Extrage obiectul din mesajul obiect JMS de intrare și îl înfășoară în obiectul business JMSTextBody.	Extrage obiectul din obiectul business JMSTextBody și îl înfășoară în mesajul obiect JMS de ieșire.
Text înfășurat	Extrage textul din mesajul text JMS de intrare și îl înfășoară în obiectul business JMSTextBody.	Extrage textul din obiectul business JMSTextBody și îl înfășoară în mesajul text JMS de ieșire.

Legări WebSphere MQ

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

Tabela 54. Legări de date predefinite pentru legări WebSphere MQ

Legare de date	Date native în obiect business	Obiect business în 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 ieșire în WSDL).	Transformă un obiect business în obiectul serializat Java din mesajul de ieșire
Octeți înfășurați	Extrage octeții din mesajul de octeți MQ nestructurat și îi înfășoară în obiectul business JMSBytesBody.	Extrage octeții dintr-un obiect business JMSBytesBody și înfășoară octeții în mesajul de octeți MQ nestructurat de ieșire.
Text înfășurat	Extrage textul dintr-un mesaj text MQ nestructurat și îl înfășoară într-un obiect business JMSTextBody.	Extrage textul dintr-un obiect business JMSTextBody și îl înfășoară într-un mesaj text MQ nestructurat.
Intrare flux înfășurată	Extrage informațiile despre nume și tip pentru fiecare intrare din mesajul flux JMS de intrare și creează o listă cu obiectele business StreamEntry. Înfășoară apoi lista în obiectul business JMSSStreamBody.	Extrage informațiile despre nume și tip din lista StreamEntry din obiectul business JMSSStreamBody și creează intrări corespunzătoare în JMSSStreamMessage de ieșire.

În plus față de legările de date menționate în Tabela 25 la pagina 56, WebSphere MQ folosește de asemenea legări de date de antet. Vedeți Centrul de informare IBM Integration Designer pentru detalii.

Legări HTTP

Următorul tabel listează legările de date ce pot fi folosite cu HTTP și descrie taskurile realizate de legările de date.

Tabela 55. Legări de date predefinite pentru legări HTTP

Legare de date	Date native în obiect business	Obiect business în date native
Octeți înfășurați	Extrage octeții din corpul mesajului HTTP de intrare și îi înfășoară în obiectul business HTTPBytes.	Extrage octeții din obiectul business HTTPBytes și îi adaugă la corpul mesajului HTTP de ieșire.
Text înfășurat	Extrage textul din corpul mesajului HTTP de intrare și îl înfășoară în obiectul business HTTPText.	Extrage textul din obiectul business HTTPText și îl adaugă la corpul mesajului HTTP de ieșire.

Selectorii de funcții în legări de export

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

Considerați un export SCA ce expune o interfață. Interfața conține două operații - Creare și Actualizare. Exportul are o legare JMS ce citește dintr-o coadă.

Când un mesaj ajunge în coadă, exportului îi sunt transmise datele asociate, dar ce operație din interfața exportului ar trebui invocată pe componenta cablată? Operația este determinată de selectorul de funcții și configurația legării de export.

Selectorul de funcții returnează numele funcției native (numele funcției din sistemul client ce a trimis mesajul). Numele funcției native este apoi mapat de numele operației sau funcției de pe interfața asociată cu exportul. De exemplu, în următoarea figură, selectorul de funcții returnează numele funcției native (CRT) din mesajul de intrare, numele funcției native este mapat de operația Creare și obiectul business este trimis componentei SCA cu operația Creare.

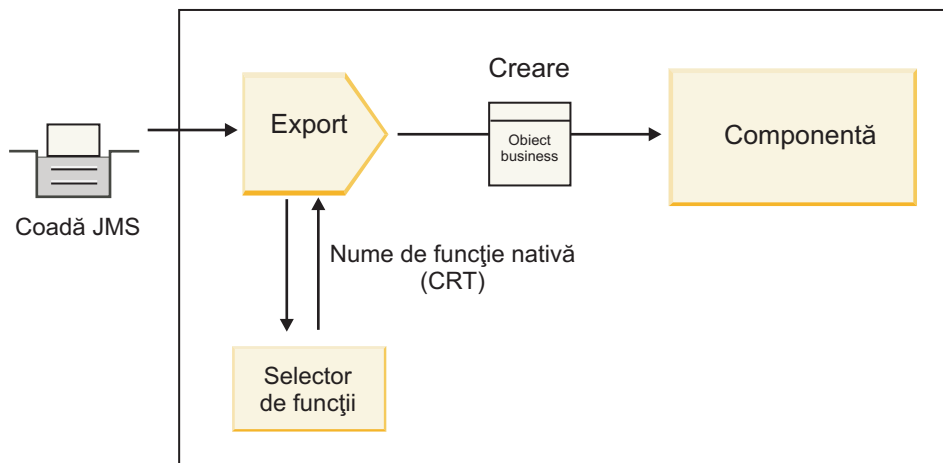


Figura 52. Selectorul de funcții

Dacă interfața are o singură operație, nu este nevoie să specificați un selector de funcții.

Mai mulți selectori de funcții preambalați sunt disponibili și sunt menționați în secțiunile ce urmează.

Legări JMS

Următorul tabel listează selectorii de funcții ce pot fi folosiți cu:

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

Tabela 56. Selectorii de funcții predefiniți pentru legări JMS

Selector de funcții	Descriere
Selector de funcții JMS pentru legări simple de date JMS	Folosește proprietatea JMSType a mesajului pentru a selecta numele operației.
Selector de funcții a proprietății antetului JMS	Returnează valoarea Proprietății șir JSM, TargetFunctionName, din antet.
Selector funcție gateway de servicii JMS	Determină dacă cererea este o operație cu un singur sens sau cu sens dublu prin examinarea proprietății JMSReplyTo setată de client.

Legări WebSphere MQ

Următorul tabel listează selectorii de funcții ce pot fi utilizați cu legări WebSphere MQ.

Tabela 57. Selectorii de funcții predefiniți pentru legări WebSphere MQ

Selector de funcții	Descriere
Selector de funcții MQ handleMessage	Returnează handleMessage ca o valoare, ce este mapată folosind legările metodei de export de numele unei operații de pe interfață.
MQ folosește selector de funcții implicit JMS	Citește operația nativă din proprietatea TargetFunctionName din folderul unui antet MQRFH2.
MQ folosește formatul corpului mesajului ca funcție nativă	Găsește câmpul Format al ultimului antet și returnează acel câmp ca un Șir.
Selector de funcții de tip MQ	Crează o metodă în legarea dumneavoastră de export prin extragerea unui URL ce conține proprietățile Msd, Set, Type și Format aflate în antetul MQRFH2.
Selector funcție gateway de servicii MQ	Folosește proprietatea MsgType din antetul MQMD pentru a determina numele proprietății.

Legări HTTP

Următorul tabel listează selectorii de funcții ce pot fi utilizați cu legări HTTP.

Tabela 58. Selectorii de funcții predefiniți pentru legări HTTP

Selector de funcții	Descriere
Selector de funcții HTTP bazat pe antetul TargetFunctionName	Utilizează proprietatea antetului HTTP TargetFunctionName de la client pentru a determina ce operație să invoce în momentul rulării din export.
Selector de funcții HTTP bazat pe metoda URL și HTTP	Folosește calea relativă de la URL adăugată la sfârșitul metodei HTTP de la client pentru a determina operația nativă definită pe export.
Selector funcție gateway de servicii HTTP bazat pe URL cu un nume de operație	Determină metoda de invocare în funcție de URL dacă "operationMode = oneway" a fost adăugat la URL-ul cererii.

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

Tratarea defectelor

Vă puteți configura legările de import și export să trateze defecte (de exemplu, excepții operaționale) ce au loc în timpul procesării prin specificarea handler-elor de date ale defectelor. Puteți un handler de date de defecte la trei niveluri - puteți asocia un handler de date de defecte cu un defect, cu o operație sau pentru toate operațiile cu o legare.

Un handler de date al defectelor procesează date ale defectelor și 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 excepției trimis de la componentă într-un mesaj răspuns ce poate fi folosit de aplicația 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 excepției 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 operațional sau o excepție runtime.

Puteți specifica un handler de date ale defectelor pentru un anumit defect, pentru o operație și pentru toate operațiile 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-urile de date ale defectelor sunt setate la nivelurile de operație și legare, este apelat handler-ul de date asociat cu operația.

Sunt folosite două editoare în IBM Integration Designer pentru specificarea tratării defectelor. Editorul de interfețe este folosit pentru a indica dacă va fi un defect pe o operație. După ce este generată o legare cu această interfață, editorul din vizualizarea proprietăților vă dă voie să configurați cum va fi tratat defectul. Pentru informații suplimentare, vedeți subiectul “Selectorii de defecte” din centrul de informare IBM Integration Designer.

Cum sunt tratate defectele în legări de export:

Când apare un defect în timpul procesării cererii de la o aplicație client, legarea de export poate returna informațiile despre defect clientului. Configurați legarea de export să specifice cum ar trebui procesat defectul și returnat clientului.

Configurați legarea de export folosind IBM Integration Designer.

În timpul procesării cererii, un client invocă un export cu o cerere și exportul invocă componenta SCA. În timpul procesării cererii, componenta SCA poate fie returna un răspuns operațional fie poate arunca o excepție operațională de serviciu sau o excepție runtime de serviciu. Când apare aceasta, legarea de export transformă excepția într-un mesaj de defect și îl trimite clientului, după cum este arătat în figura următoare și descris în secțiunile ce urmează.



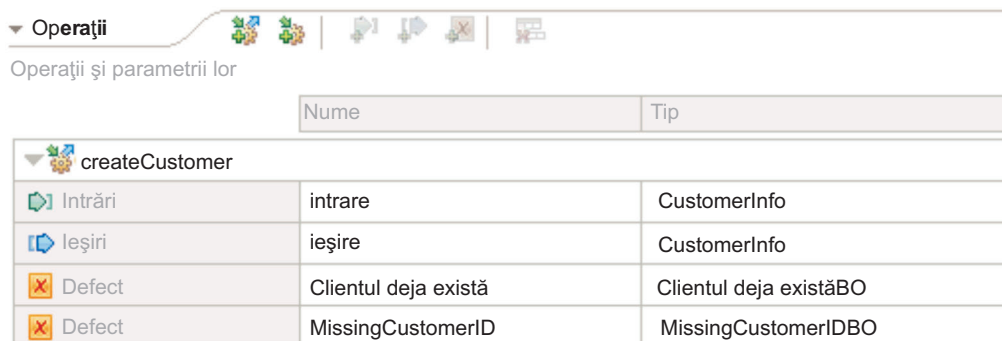
Figura 53. Cum sunt trimise informațiile despre defect de la componentă prin legarea de export clientului

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

Defecte operaționale

Defectele operaționale sunt erori sau excepții operaționale ce au loc în timpul procesării.

Considerați următoarea interfață, ce are o operație createCustomer pe ea. Această operație are două defecte operaționale definite: CustomerAlreadyExists și MissingCustomerId.



The screenshot shows a software interface with a menu bar containing 'Operații' and several icons. Below the menu bar, the text 'Operații și parametrii lor' is displayed. A table with two columns, 'Nume' and 'Tip', lists the following data:

	Nume	Tip
▼ createCustomer		
Intrări	intrare	CustomerInfo
Ieșiri	ieșire	CustomerInfo
Defect	Clientul deja există	Clientul deja existăBO
Defect	MissingCustomerId	MissingCustomerIdBO

Figura 54. Interfață cu două defecte

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

Când un defect operațional 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 excepția operațională de servicii conține numele defectului, este apelat handler-ul de date ce este configurat pe defect. Dacă excepția operațională de servicii nu conține 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 excepția operațională de servicii.
3. Handler-ul de date de defecte transformă obiectul de date de defect într-un mesaj răspuns și îl returnează legării de export.
4. Exportul returnează mesajul răspuns clientului.

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

Excepții în timpul rulării

O excepție runtime este o excepție ce are loc în aplicația SCA în timpul procesării unei cereri ce nu corespunde unui defect operațional. Spre deosebire de defectele operaționale, excepțiile runtime nu sunt definite pe interfață.

În anumite scenarii, ați putea dori să propagați aceste excepții runtime către aplicația client, astfel încât aplicația client să poată lua măsurile corespunzătoare.

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

Notă: Puteți configura un handler de date al excepției runtime doar pe legări HTTP.

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

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

Tratarea defectelor și tratarea excepțiilor runtime sunt opționale. Dacă nu doriți să propagați defecte sau excepții runtime către clientul apelant, nu configurați handler-ul de date al defectelor sau handler-ul de date al excepției runtime.

Modul în care defectele sunt tratate în legările de import:

O componentă folosește 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 configurați legarea de import puteți specifica modul în care defectul ar trebui procesat și returnat către componentă.

Configurați legarea de import folosind IBM Integration Designer. Puteți specifica un handler pentru datele ce conțin defecte (sau o legare de date), și, de asemenea, puteți specifica un selector pentru defect.

Handler-e pentru datele ce conțin defecte

Serviciul care procesează cererea trimite, către legarea de import, informații despre defect sub forma unei excepții sau a unui mesaj de răspuns care conține datele de defect.

Legarea de import transformă excepția serviciului sau mesajul de răspuns într-un excepție a serviciului operațional sau o excepție de runtime serviciu, după cum se arată în figura de mai jos și cum este descris în secțiunile care urmează.

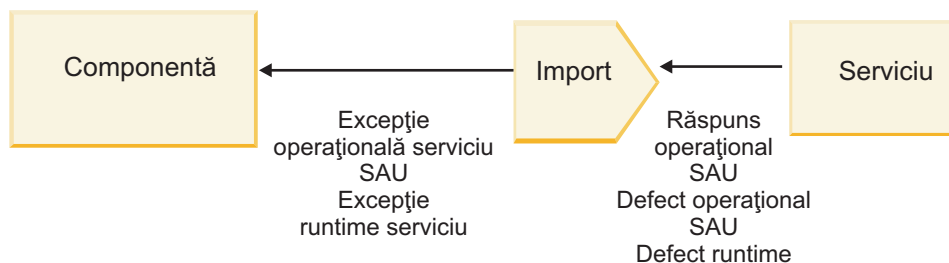


Figura 55. Modul în care informațiile despre eroare sunt transmise cu ajutorul importului de la serviciu către componentă

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

Selectorii defect

Atunci când configurați o legare de import, aveți posibilitatea să specificați un selector de defect. Selectorul de defect determină dacă răspunsul de import este un răspuns real, o excepție operațională sau un defect apărut în timpul rulării. De asemenea, determină din corpul sau antetul răspunsului, numele nativ al defectului, care este mapat prin configurația legării de numele defectului din interfața asociată.

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

Tabela 59. Selectoare pentru defecte împachetate

Tip selector defect	Descriere
Bazat pe antet	Stabilește dacă un mesaj de răspuns este un defect operațional, o excepție 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 excepție în timpul rulării.

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

- Selector de defect bazat pe antet

Dacă o aplicație dorește să indice că mesajul primit este un defect operațional, atunci trebuie să existe două anteturi în mesajul de intrare pentru defectele operaționale, așa cum este prezentat în continuare:

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

Dacă o aplicație dorește să indice că mesajul primit este o excepție apărută în timpul rulării, atunci trebuie să existe un antet în mesajul de intrare, așa cum este prezentat în continuare:

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

- Selector de defect SOAP

Un defect operațional poate fi trimis ca parte componentă a mesajului SOAP care are în compoziție 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 opțional. Dacă nu specificați un selector pentru defecare, legarea de import nu poate determina tipul răspunsului. Prin urmare, legarea îl tratează ca un răspuns operațional (business) și apelează handler-ul datelor de răspuns sau legarea de date.

Puteți crea un selector personalizat pentru defect. Pașii 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 operaționale

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

Atunci când legarea primește o excepție operațională, etapele de prelucrare depind de faptul dacă un selector de defect a fost setat pentru legare.

- Dacă nu a fost setat niciun 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 operațional (business) sau un defect apărut în timpul rulării.
 2. În cazul în care răspunsul este un defect operațional, 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 de date pentru defect care este configurat pentru acest defect WSDL.
 5. Legarea de import apelează acest handler-ul de date pentru defect cu datele defectului.
 6. Handler-ul de date pentru defect transformă datele defectului într-un obiect de date și îl returnează către legarea de import.

7. Legarea de import construiește un obiect de tip excepție pentru serviciul operațional cu obiectul de date și numele defectului.
8. Importul returnează obiectul de tip excepție pentru serviciul operațional către componentă.

Excepții din timpul rulării

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

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

Interoperabilitatea între modulele SCA și 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 aplicațiilor bazate pe specificațiile Open SCA. Modulele SCA din IBM Business Process Manager folosesc legările de import și export pentru a interopera cu serviciile Open SCA dezvoltate într-un mediu Rational Application Developer și găzduite de WebSphere Application Server Feature Pack for Service Component Architecture.

O aplicație SCA invocă o aplicație Open SCA prin intermediul unei legări de import. O aplicație SCA primește un apel de la aplicație Open SCA prin intermediul unei legări de export. O listă de legări acceptate este prezentată în “Invocarea serviciilor peste legările interoperabile” la pagina 65.

Invocarea serviciilor Open SCA din module SCA

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

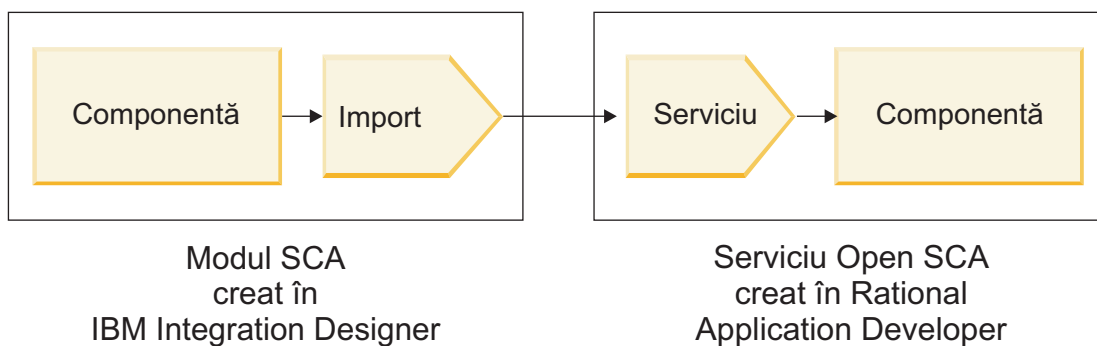


Figura 56. Componenta din modulul SCA invocă un serviciu Open SCA

Nu este nevoie de nicio configurație specială pentru a invoca un serviciu Open SCA.

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

1. Pentru a obține numele componentei țintă și serviciul din Open SCA compus, efectuați următorii pași:
 - a. Asigurați-vă că fila **Proprietăți** este deschisă făcând clic pe **Fereastră > Afișare vizualizare > Proprietăți**.

- b. Deschideți editorul pentru compus făcând dublu-clic pe diagrama compusă care conține componenta și serviciul. De exemplu, pentru o componentă ce are numele **customer**, diagrama compusă este **customer.composite_diagram**.
 - c. Faceți clic pe componenta țintă.
 - d. În câmpul **Nume** din fila **Proprietăți**, notați numele componentei țintă.
 - e. Faceți clic pe pictograma serviciului asociată cu componenta.
 - f. În câmpul **Nume** din fila **Proprietăți**, notați numele serviciului.
2. Pentru a configura importul IBM Business Process Manager, astfel încât să se conecteze la serviciul Open SCA, efectuați pașii următori:
 - a. În IBM Integration Designer, mergeți la fila **Proprietăți** din importul SCA pe care vreți să îl conectați la serviciul Open SCA.
 - b. În câmpul **Nume modul**, introduceți numele componentei de la pasul 1d la pagina 64.
 - c. În câmpul **Nume export**, introduceți numele serviciului de la pasul 1f la pagina 64.
 - d. Salvați-vă munca apăsând Ctrl+S.

Invocarea modulelor SCA din serviciile Open SCA

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

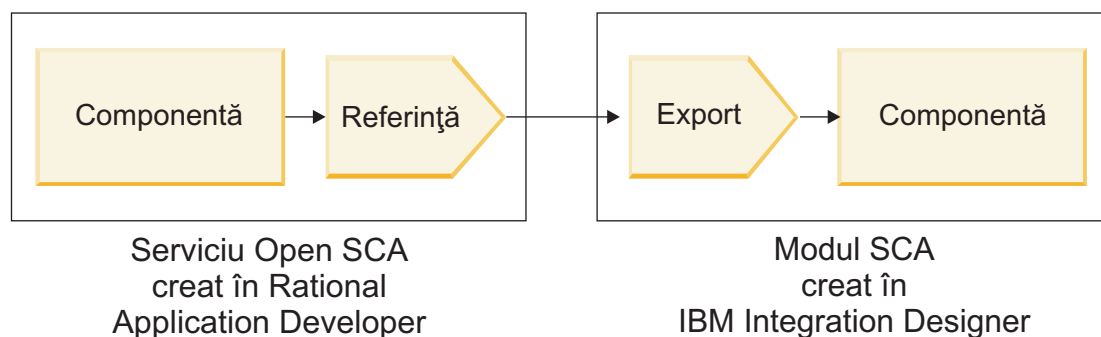


Figura 57. Deschiderea serviciului SCA prin invocarea componentei în modulul SCA

Pentru a vă conecta la o componentă SCA printr-o legare de referință a unui Open SCA, furnizați numele modulului și numele de export.

1. Pentru a obține numele modulului și exportului țintă, efectuați următorii pași:
 - a. În IBM Integration Designer, deschideți modulul în editorul de asamblare făcând dublu-clic pe modul.
 - b. Faceți clic pe export.
 - c. În câmpul **Nume** din fila **Proprietăți**, notați numele exportului.
2. Configurați referința Open SCA pe care doriți să o conectați la modulul IBM Business Process Manager și exportați:
 - a. În Rational Application Developer, deschideți editorul pentru compus făcând dublu-clic pe diagrama compusă care conține componenta și serviciul.
 - b. Faceți clic pe pictograma referinței componentei pentru a afișa aceste proprietăți de referință în fila **Proprietăți**.
 - c. Faceți clic pe fila **Legare** din partea stânga a paginii.
 - d. Faceți clic pe **Legări**, iar apoi faceți clic pe **Adăugare**.
 - e. Selectați legarea **SCA**.

- f. În câmpul **Uri**, introduceți numele modulului IBM Business Process Manager, urmat de o bară oblică (“/”), urmat de numele de export (pe care îl stabiliți la pasul 1c la pagina 64).
- g. Apăsați **OK**.
- h. Salvați-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 legări de import SCA, sunt acceptate următoarele stiluri de invocare:

- Asincron (într-un singur sens)
- Sincron (cerere/răspuns)

Interfața de import SCA și interfața serviciului Open SCA trebuie să utilizeze o interfață WSDL conformă WS-I. Rețineți că legarea SCA suportă propagarea contextului de tranzacție și de securitate.

- Legare (JAX-WS) a serviciului Web fie cu protocol SOAP1.1/HTTP, fie cu protocol SOAP1.2/HTTP

Interfața de import SCA și interfața serviciului Open SCA trebuie să utilizeze o interfață WSDL conformă WS-I.

În plus, sunt suportate următoarele calități ale serviciului:

- Tranzacția atomică a serviciilor Web
- Securitatea serviciilor Web

- Legare EJB

Pentru a defini interacțiunea dintre un modul SCA și un serviciu Open SCA se folosește o interfață Java atunci când este utilizată legarea EJB.

Rețineți că legarea EJB suportă propagarea contextului de tranzacție și de securitate.

- Legări JMS

Interfața de import SCA și interfața serviciului Open SCA trebuie să utilizeze o interfață WSDL conformă WS-I.

Sunt suportați următorii furnizori JMS:

- Platforma de mesagerie WebSphere (Legarea JMS)
- WebSphere MQ (Legarea MQ JMS)

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

Tipuri de legări

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

Selectarea legărilor corespunzătoare

Atunci când creați o aplicație, trebuie să știți cum să selectați legarea care corespunde cel mai bine necesităților aplicației dumneavoastră.

Legările disponibile din IBM Integration Designer furnizează o gamă de alegeri. În această listă puteți determina care tip de legare este mai potrivită nevoilor aplicației dumneavoastră.

Luăți în considerare o legare *SCA (Service Component Architecture)* atunci când se pot aplica acești factori:

- Toate serviciile sunt conținute în module; ceea ce înseamnă că nu există servicii externe.
- Vreți să separați funcția în module SCA diferite care interacționează direct unul cu celălalt.
- Modulele sunt cuplate strâns.

Luăți în considerare o legare *serviciu web* atunci când se aplică acești factori:

- Trebuie să accesați un serviciu extern de pe Internet sau să furnizați un serviciu pe Internet.
- Serviciile sunt cuplate slab.
- Comunicația sincronă este preferabilă; adică o cerere de la un serviciu poate aștepta pentru un răspuns de la altul.
- Protocolul serviciului extern pe care îl accesați sau al serviciului pe care vreți să îl furnizați este SOAP/HTTP sau SOAP/JMS.

Considerați o legare *HTTP* când acești factori sunt aplicabili:

- Este nevoie să accesați un serviciu extern de pe Internet sau să furnizați un serviciu de pe Internet și lucrați cu alte servicii web cum ar fi GET, PUT și DELETE.
- Serviciile sunt cuplate slab.
- Comunicația sincronă este preferabilă; adică o cerere de la un serviciu poate aștepta pentru un răspuns de la altul.

Luați în considerare o legare *EJB (Enterprise JavaBeans)* atunci când se pot aplica acești factori:

- Legarea este pentru un serviciu importat care este el însuși un EJB sau care trebuie să fie accesat de clienți EJB.
- Serviciul importat este cuplat slab.
- Interacțiunile EJB stateful nu sunt necesare.
- Comunicația sincronă este preferabilă; adică o cerere de la un serviciu poate aștepta pentru un răspuns de la altul.

Luați în considerare o legătură *EIS (Enterprise Information Systems)* atunci când se pot aplica acești factori:

- Trebuie să accesați un serviciu de pe un sistem EIS utilizând un adaptor de resurse.
- Transmisia de date sincronă este preferată în locul celei asincrone.

Luați în considerare o legare *JMS (Java Message Service)* atunci când se pot aplica acești factori:

Important: Există mai multe tipuri de legări JMS. Dacă vă așteptați să schimbați mesajele SOAP folosind JMS, luați în considerare legarea de servicii web cu protocolul SOAP/JMS. Vedeți “Legări de serviciu Web” la pagina 67.

- Trebuie să accesați un sistem de mesagerie.
- Serviciile sunt cuplate slab.
- Transmisia de date asincron este preferată în locul celei sincron.

Luați în considerare o legare *JMS (Java Message Service) generic* atunci când se pot aplica acești factori:

- Trebuie să accesați sistem de mesagerie a unui vendor non-IBM.
- Serviciile sunt cuplate slab.
- Fiabilitatea este mai importantă decât performanța; adică, transmisia de date asincronă este preferată în locul celei sincron.

Luați în considerare o legare *MQ (Message Queue)* atunci când se pot aplica acești factori:

- Trebuie să accesați un sistem de mesagerie WebSphere MQ și să utilizați funcțiile native MQ.
- Serviciile sunt cuplate slab.
- Fiabilitatea este mai importantă decât performanța; adică, transmisia de date asincronă este preferată în locul celei sincron.

Luați în considerare o legare *MQ JMS* când se aplică acești factori:

- Trebuie să accesați un sistem de mesagerie WebSphere MQ dar puteți face astfel într-un context JMS; adică, subsetul JMS al funcțiilor este suficient pentru aplicația dumneavoastră.
- Serviciile sunt cuplate slab.
- Fiabilitatea este mai importantă decât performanța; adică, transmisia de date asincronă este preferată în locul celei sincron.

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 accesați un serviciu dintr-un alt modul SCA. Un export cu o legare SCA vă permite să oferiți un serviciu către alte module.

Utilizați IBM Integration Designer pentru a genera și configura legări SCA în importuri și exporturi din module SCA.

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

După ce modulul care conține legarea SCA este implementat pe server, puteți utiliza consola administrativă pentru a vizualiza informații despre legare sau, în cazul unei legări de import, pentru a modifica proprietățile 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 (și vice versa).

Privire generală asupra legărilor de serviciu Web: Advanced

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

Cu o legare de serviciu web, puteți accesa serviciile externe folosind mesaje SOAP interoperabile și calitățile de serviciu (QoS).

Utilizați Integration Designer pentru a genera și configura legările serviciului web la importuri și exporturi în module SCA. Sunt disponibile următoarele tipuri de legări serviciu web:

- SOAP1.2/HTTP și SOAP1.1/HTTP

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

- Utilizați SOAP1.2/HTTP dacă serviciul dvs. web corespunde specificației SOAP 1.2.
- Utilizați SOAP1.1/HTTP dacă serviciul dvs. web corespunde specificației SOAP 1.1.

Important: Când implementați o aplicație cu o legare de serviciu web (JAX-WS), serverul țintă nu trebuie să aibă selectată opțiunea **Pornire componente după cum este necesar**. Pentru detalii, vedeți “Verificarea configurației serverului” la pagina 75.

Când selectați una din aceste legări, puteți trimite atașamente cu mesajele dvs. SOAP.

Legările serviciului web funcționează cu mesaje SOAP standard. Folosind una dintre legările serviciului web JAX-WS, totuși, aveți posibilitatea să personalizați modul în care sunt parsate sau scrise mesajele SOAP. De exemplu, puteți manipula elementele care nu sunt standard în mesajele SOAP sau puteți aplica prelucrări suplimentare mesajelor SOAP. Când configurați legarea specificați un handler personalizat pentru date, care efectuează această prelucrare într-un mesaj SOAP.

Puteți folosi seturi de politici cu o legare a serviciului web (JAX-WS). Un set de politici este o colecție 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 și a mesajelor. Utilizați Integration Designer pentru a selecta setul de politici pentru legare.

Notă: Dacă doriți să utilizați un set de politici de tip SAML (Security Assertion Markup Language), trebuie să realizați câteva configurații suplimentare, așa cum este descris în “Importul seturilor de politici” la pagina 73.

- SOAP1.1/HTTP

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

- SOAP1.1/JMS

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

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

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

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

După ce modulul SCA care conține legarea serviciului web este implementată pe server, aveți posibilitatea să utilizați consola administrativă pentru a vizualiza informații legate de legare sau să modificați proprietățile selectate a legării.

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

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

Propagarea antetului SOAP: Advanced

În momentul în care manipulați mesaje SOAP, este posibil să aveți nevoie să accesați informații din anumite anteturi SOAP din mesaje ce sunt primite, ca să vă asigurați că mesajele cu anteturi SOAP sunt trimise cu anumite valori sau ca să permiteți anteturilor SOAP să parcurgă un modul.

Când configurați legarea unui serviciu web în Integration Designer, puteți indica faptul că doriți ca anteturile SOAP să fie propagate.

- Atunci când cererile sunt primite într-un export sau răspunsurile sunt primite la un import, informațiile din antetul SOAP pot fi accesate, permițând ca logica din modul să se bazeze pe valorile antetului și permițând 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 și prezența anteturilor SOAP transmise pot fi afectate de seturile de politici configurate la import sau export, așa cum este explicat în Tabela 31 la pagina 70.

Pentru a configura transmiterea anteturilor SOAP pentru un import sau export, selectați (din modul de vizualizare Proprietăți al Integration Designer) fila **Antet Protocol de Transmitere** și selectați opțiunile de care aveți nevoie.

Anteturi de Adresare WS

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

Atunci când transmiteți antetul de adresare WS, fiți atenți la următoarele informații:

- Dacă activați transmiterea pentru antetul de adresare WS, antetul va fi transmis prin modul în următoarele circumstanțe:
 - 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 ieșire 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 WS-Security

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

Specificațiile WS-Security ale serviciilor web descriu îmbunătățirile aduse mesajelor SOAP pentru a oferi calitatea protecției prin integritatea mesajelor, confidențialitatea mesajelor și autentificarea unică a mesajului. Aceste mecanisme pot fi folosite pentru a acomoda o mare varietate de modele de securitate și tehnologii de criptare.

Atunci când transmiteți antetul WS-Security, fiți atenți la următoarele informații:

- Dacă activați transmiterea pentru antetul de WS-Security, antetul va fi transmis prin modul în următoarele circumstanțe:
 - 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. Totuși, dacă setați 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 inițial. 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 includeți schema WS-Security în modulul aplicației. Vedeți “Includerea schemei WS-Security într-un modul de aplicație” la pagina 70 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, așa cum este arătat în Tabela 31 la pagina 70:

Tabela 60. 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	<p>Anteturile de securitate sunt transmise așa cum sunt prin modul. Ele nu sunt decriptate.</p> <p>Anteturile sunt trimise către ieșire în aceeași formă în care au fost primite.</p> <p>Semnătura digitală ar putea deveni nevalidă.</p>	<p>Anteturile de securitate sunt decriptate și transmise prin modul cu verificarea semnăturii și autentificare.</p> <p>Anteturile decriptate sunt trimise către ieșire.</p> <p>Semnătura digitală ar putea deveni nevalidă.</p>
Legare de import cu politică de securitate	<p>Anteturile de securitate sunt transmise așa cum sunt prin modul. Ele nu sunt decriptate.</p> <p>Anteturile nu ar trebui să fie propagate către import. În caz contrar, din cauza duplicării apare o eroare.</p>	<p>Anteturile de securitate sunt decriptate și transmise prin modul cu verificarea semnăturii și autentificare.</p> <p>Anteturile nu ar trebui să fie propagate către import. În caz contrar, din cauza duplicării apare o eroare.</p>

Configurați seturile de politici corespunzătoare pentru legările de export și import, deoarece acestea izolează solicitantul serviciului de modificările aduse configurației sau cerințelor QoS ale furnizorului de servicii. Anteturile SOAP standard vizibile într-un modul pot fi utilizate pentru a influența prelucrarea (de exemplu, înregistrarea în istoric și urmărirea) în modulul. Propagarea anteturilor SOAP printr-un modul ce aparțin de un mesaj recepționat 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 așa fel încât să prevină propagarea anteturilor protocolului.

Accesarea anteturilor SOAP

Atunci când este primit un mesaj care conține anteturi SOAP printr-un import sau export de serviciu web, anteturile sunt puse în secțiunea anteturi a obiectului mesaj al serviciului (SMO - service message object). Puteți accesa informațiile din antet, așa cum este descris în "Accesarea informațiilor din antetul SOAP în SMOO".

Includerea schemei WS-Security într-un modul de aplicație

Procedura următoare definește pașii pentru includerea schemei în modulul aplicației:

- În cazul în care computerul pe care rulează Integration Designer are acces la Internet, efectuați următorii pași:
 1. În perspectiva Business Integration, selectați **Dependente** pentru proiectul dumneavoastră.
 2. Extindeți **Resurse predefinite** și selectați fie **Fișiere schemă WS-Security 1.0**, fie **Fișiere schemă WS-Security 1.1** pentru a importa schema în modulul dumneavoastră.
 3. Curățați și reconstruiți proiectul.
- În cazul în care un computer pe care rulează Integration Designer nu are acces la Internet, puteți descărca schema pe un al doilea computer care are acces la Internet. Apoi o puteți copia pe computerul pe care rulează Integration Designer.
 1. De pe computerul care are acces la Internet, descărcați schema de la distanță:
 - a. Faceți clic pe **Fișier > Import > Business Integration > WSDL și XSD**.
 - b. Selectați **WSDL la distanță** sau **Fișier XSD**.
 - c. Importați 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/xmlsig-core/xmlsig-core-schema.xsd

2. Copiați schemele pe un computer care nu are acces la Internet.
3. De pe computerul care nu are acces la Internet, importați schema:
 - a. Faceți clic pe **Fișier > Import > Business Integration > WSDL și XSD**.
 - b. Selectați **WSDL Local** sau **Fișier XSD**.
4. Modificați locațiile schemei pentru oasis-wss-wssecurity-secext-1.1.xsd:
 - a. Deschideți schema în *locație spațiu de lucru/nume_modul/StandardImportFilesGen/oasis-wss-wssecurity-secext-1.1.xsd*.
 - b. Modificați:

```
<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. Modificați:

```
<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. Modificați locația schemei pentru oasis-200401-wss-wssecurity-secext-1.0.xsd:
 - a. Deschideți schema în *locație spațiu de lucru/nume_modul/StandardImportFilesGen/oasis-200401-wss-wssecurity-secext-1.0.xsd*.
 - b. Modificați:

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

în:

```
<xsd:import namespace="http://www.w3.org/2000/09/xmlsig#"  
schemaLocation="../w3/tr/_2002/rec_xmlsig_core_20020212/xmlsig-core-schema.xsd"/>
```
6. Curățați și reconstruiți proiectul.

Propagarea antetului Transport: **Advanced**

În momentul în care manipulați mesaje SOAP, este posibil să aveți nevoie să accesați informații din anumite anteturi de transport din mesaje ce sunt primite, asigurați-vă că mesajele cu anteturi de transport sunt trimise cu anumite valori, sau permiteți anteturilor de transport să parcurgă un modul.

Când configurați legarea unui serviciu web în Integration Designer, puteți indica faptul că doriți ca anteturile transport să fie propagate.

- Atunci când cererile sunt primite într-un export sau răspunsurile sunt primite la un import, informațiile din antetul de transport pot fi accesate, permițând ca logica din modul să se bazeze pe valorile antetului și permițând 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, realizați pașii următori:

1. Din vizualizarea Proprietăți Integration Designer, selectați **Legare > Propagare**.

2. Setați opțiunea de propagare a antetului de transport de care aveți nevoie.

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

Dacă activați propagarea anteturilor de transport, anteturile din mesajele primite vor fi propagate printr-un modul, iar dacă nu le eliminați în mod explicit aceste vor fi utilizate în invocațiile ulterioare prin același fir de execuție.

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

Accesarea informațiilor 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 object). Puteți seta anteturile cu valori diferite sau puteți crea altele noi. Cu toate acestea, rețineți că nu există nicio verificare sau validare a valorilor setate de dvs., și orice antet necorespunzător sau incorect poate cauza probleme în timpul execuției serviciului web.

Luați în considerare următoarele informații cu privire la setarea anteturilor HTTP:

- Orice modificare adusă anteturilor rezervate pentru motorul de servicii web nu vor fi onorate în mesajul de ieșire. De exemplu, versiunea sau metoda HTTP, anteturile Content-Type, Content-Length și SOAPAction sunt rezervate pentru motorul serviciului web.
- În cazul în care valoarea antetului este un număr, numărul (mai degrabă decât șirul) ar trebui stabilit în mod direct. De exemplu, utilizați **Max-Forwards = 5** (în loc de **Max-Forwards = Max-Forwards: 5**) și **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 și setează în loc antetul Content-Length pentru dimensiune fixată a mesajului.
- Conținutul 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 sfârșit valorile rezervate de motorul serviciilor web la setările clientului:
 - User-Agent
 - Cache-Control
 - Pragma
 - Accept
 - Connection

Puteți accesa informațiile din antet într-unul din următoarele moduri:

- Utilizarea unei primitive de mediere pentru accesarea structurilor de tip SMO
Vedeți legăturile “Informații înrudite” pentru a găsi informații despre utilizarea primitivelor de mediere.
- Utilizarea serviciului de context SPI

Următorul exemplu de cod citește 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 întâlniți probleme în momentul în care trimiteți anteturi revizuite, puteți intercepta mesajul TCP/IP folosind unelte precum TCP/IP Monitor din Integration Designer. Puteți accesa TCP/IP Monitor selectând **Rulare/Depanare > TCP/IP Monitor** în pagina Preferințe.

De asemenea puteți 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): **Advanced**

Atunci când utilizați legări de servicii web (JAX-WS) cu aplicațiile dvs., aveți posibilitatea să adăugați o calitate SAML (Security Assertion Markup Language) a serviciului (QOS) la legare. În primul rând trebuie să utilizați consola administrativă pentru a importa setul de politici. De asemenea, puteți 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: **Advanced**

Security Assertion Markup Language (SAML) este un standard OASIS bazat pe XML pentru schimbarea identității și atributelor de securitate ale informației. Atunci când configurați un serviciu web (JAX-WS) legat cu Integration Designer, puteți specifica un set de politici SAML. Se folosește 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 a profilului:

profile_root/config/templates/PolicySets

Înainte de a începe această procedură, se verifică dacă următoarele directoare (ce conțin seturile de politici) sunt localizate în directorul de configurare a 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 a profilului, copiați-le în acel director din următoarea locație:

app_server_root/profileTemplates/default/documents/config/templates/PolicySets

Importați seturile de politici în consola administrativă, selectați una pe care doriți să o faceți disponibilă pentru Integration Designer și apoi salvați un fișier .zip pentru fiecare dintre aceste seturi de politici într-o locație care este accesibilă prin Integration Designer.

1. Se va importa seturile de politici prin urmărirea pașilor următori:

- a. Din consola administrativă, se face clic pe **Servicii > Seturi de politici > Seturi de politici aplicație**.
 - b. Se face clic pe **Import > Din magazia implicită**.
 - c. Se selectează seturile de politici implicite SAML și 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 aplicație, se selectează setul de politici SAML care se vrea a fi exportat și se face clic pe **Export**.

Notă: Dacă pagina Seturi de politici aplicație nu este momentan afișată, se face clic pe **Servicii > Seturi de politici > Seturi de politici aplicație** din consola administrativă.

- b. Pe pagina următoare, se face clic pe legătura la fișierului .zip pentru setul de politici.
- c. În fereastra Descărcare fișier, se face clic pe **Salvare** și se indică o locație ce este accesibilă prin Integration Designer.
- d. Se face clic pe **Înapoi**.
- e. Finalizați pașii de la 2a la pagina 74 la 2d la pagina 74 pentru fiecare set de politici care se vrea a fi exportat.

Seturile de politici SAML sunt salvate în fișierul .zip și sunt gata să fie importate în Integration Designer.

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

Invocarea serviciilor web ce necesită autentificare de bază HTTP: 

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

Setați autentificarea de bază HTTP pentru recepționarea cererilor de servicii web prin configurarea Java API pentru XML Web Services (JAX-WS) legări de export, așa cum este descris în Crearea și asignarea rolurilor de securitate pentru exporturi 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 modalități:

- La configurarea legării de import într-un modul SCA, puteți 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, puteți utiliza capabilități de mediere a fluxului pentru a crea dinamic un nou antet de autentificate HTTP și specifica informații cu numele și parola de utilizator în antet.

Notă: Setul de politică are precedență peste valoarea specificată în antet. Dacă doriți să utilizați setul de valori în antetul de autentificare HTTP la momentul rulării, nu potriviți un set de politică ce include politica HTTPTransport. Specific, nu utilizați setul de politică BPMHTTPBasicAuthentication și dacă aveți un set de politică definit, asigurați-vă că include politica HTTPTransport.

Pentru informații suplimentare despre seturile de politici pentru servicii web și legări de politici și cum sunt utilizate acestea, vedeți Seturi de politici de servicii web din WebSphere Application Server Information Center.

- Pentru a utiliza setul de politică furnizat, realizați următorii pași:
 1. Opțional: În consola administrativă, creați o legare de politică generală sau editați una existentă care include politica HTTPTransport cu valorile ID-ului de utilizator și parolei necesare.
 2. În IBM Integration Designer, generați o legare de import a serviciului web (JAX-WS) și atașați setul de politică BPMHTTPBasicAuthentication.
 3. Realizați *unul* din următorii pași:
 - În IBM Integration Designer, în proprietățile de legare de import al serviciului web (JAX-WS), specificați numele unei legări a politicii existente de client generale ce include politica HTTPTransport.

- După implementarea modului SCA, utilizați consola administrativă fie pentru a selecta o legare de politică pentru un client existent, fie creați o nouă legare de politică de client și apoi o asociați cu legarea de import.
- 4. Opțional: În consola administrativă a serverului Process editați legarea setului de politici selectat pentru a specifica ID-ul și parola necesare.
- Pentru a specifica numele și parola de utilizator în antetul de autentificare HTTP, realizați unul din următorul set de pași:
 - Utilizați medierea simplă HTTP Header Setter în IBM Integration Designer pentru a crea antetul de autentificare HTTP și specificați numele și parola de utilizator.
 - Dacă o logică suplimentară este necesară, utilizați codul Java într-o mediere simplă personalizată (asemenea următorului exemplu) pentru a:
 1. Crea un antet de autentificare HTTP.
 2. Specifica informații privind numele și parola de utilizator.
 3. Adăuga un nou antet de autentificare HTTP la HTTPControl.
 4. Setă HTTPControl actualizat înapoi în serviciul Context.

```
//Get the HeaderInfoType from 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();
}
//Create new HTTPAuthentication and set the 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);
// Set header info back to the current execution context.
contextService.setHeaderInfo(headers);
```

Verificarea configurației serverului: Advanced

Atunci când implementați o aplicație cu legarea unui serviciu web (JAX-WS), trebuie să vă asigurați că serverul pe care este implementată aplicația nu are selectată opțiunea **Pornire componente după cum este necesar**.

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

1. Se face clic pe **Servere > Tipuri de servere > Servere de aplicații WebSphere**.
2. Se face clic pe nume server.
3. Din fișa Configurație, se determină dacă este selectat **Pornire componente după cum este necesar**.
4. Se vor realiza următorii pași:
 - Dacă **Pornire componente după cum este necesar** este selectat, se va șterge caseta de bifare și apoi se va face clic pe **Aplicare**.
 - Dacă **Pornire componente după cum este necesar** nu este selectat, se va faceți clic pe **Anulare**.

Atașamente în mesaje SOAP: **Advanced**

Aveți posibilitatea să trimiteți și să primiți mesaje SOAP care includ date binare (precum fișiere PDF sau imagini JPEG) sub formă de atașamente. Atașamentele pot fi cu *referință* (adică sunt reprezentate în mod explicit ca părți componente ale mesajului în interfața serviciului) sau *fără referință* (în care pot fi incluse numere arbitrare și tipuri de atașamente).

Un atașament către care se face referință poate fi reprezentat în unul dintre următoarele moduri:

- Atașamentele MTOM folosesc codarea specifică SOAP Message Transmission Optimization Mechanism (<http://www.w3.org/TR/soap12-mtom/>). Atașamentele MTOM sunt activate prin opțiuni de configurare în legările de import și de export și sunt modul recomandat de încodare a atașamentelor pentru noi aplicații.

- Sub forma unui element de tip `wsi:swaRef`-typed în schema mesajului

Atașamentele 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 definește modul în care elementele mesajului sunt asociate cu părțile MIME.

- Ca o parte de nivel înalt a mesajului, folosind un tip de schemă binar

Atașamentele reprezentate sub formă de părți de nivel înalt ale mesajului în concordanță cu specificațiile *SOAP Messages with Attachments* (<http://www.w3.org/TR/SOAP-attachments>).

Atașamentele reprezentate sub formă de părți de nivel înalt pot fi de asemenea configurate pentru a vă asigura că mesajele și documentele WSDL produse de legare sunt conforme cu WS-I *Attachments Profile Version 1.0* și WS-I *Basic Profile Version 1.1* (<http://www.ws-i.org/Profiles/BasicProfile-1.1.html>).

Un atașament către care nu se face referire este purtat într-un mesaj SOAP fără nicio reprezentare în schema mesajului.

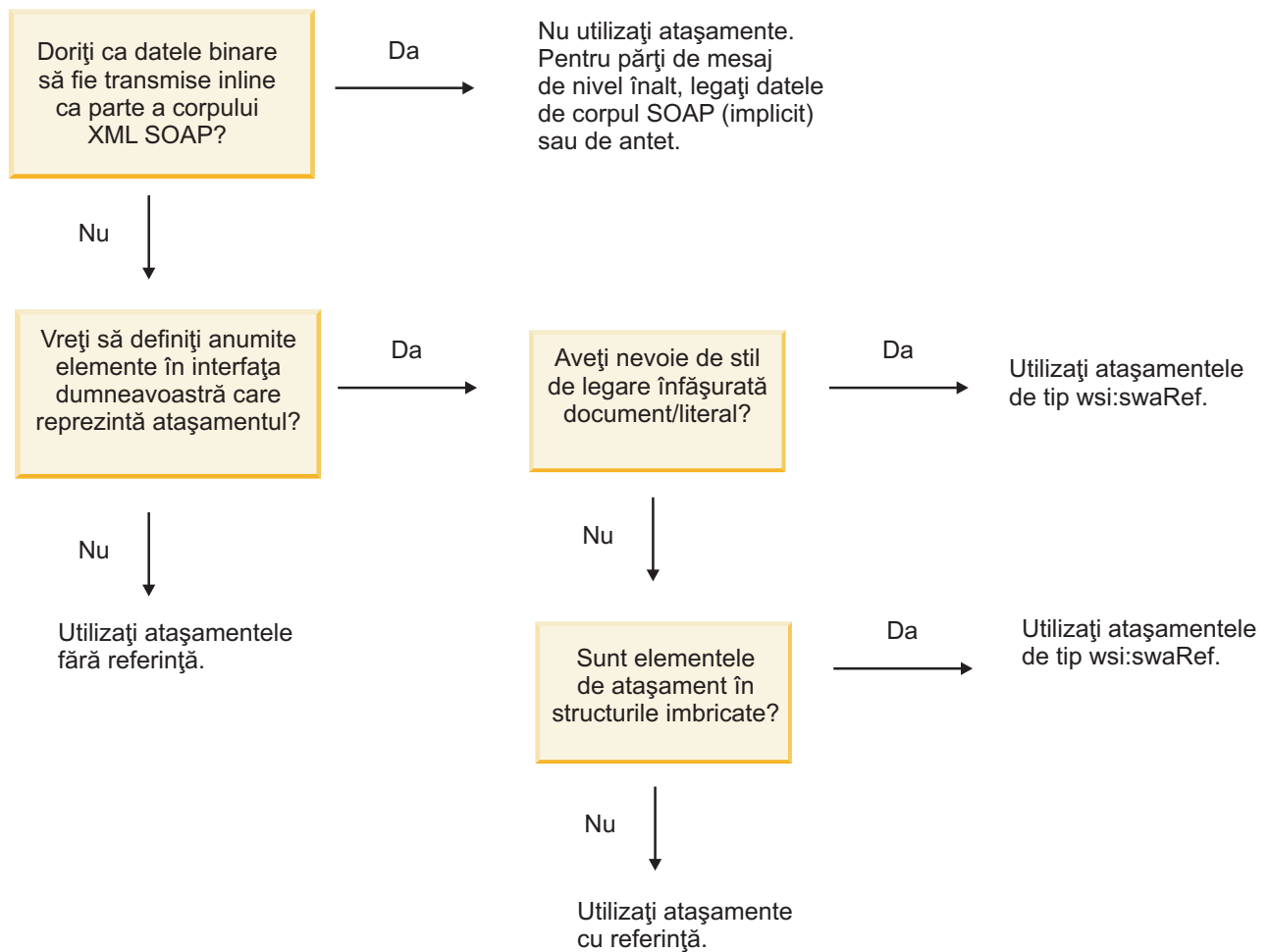
În toate cazurile, cu excepția atașamentelor MTOM, legarea WSDL SOAP trebuie să includă o legare MIME pentru atașamente de folosit, iar dimensiunea maximă a atașamentelor nu trebuie să depășească 20 MB.

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

Cum alegeți stilul corespunzător de atașamente: **Advanced**

Când proiectați o interfață nouă pentru serviciu care include date binare, luați în considerare modul în care datele binare sunt transportate în mesaje SOAP care sunt trimise și primite de către serviciu.

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



Atașamente MTOM: părți de mesaj la nivel de top:

Puteți trimite și primi mesaje de servicii web care includ atașamente MTOM (Message Transmission Optimization Mechanism - Mecanism de optimizare a transmisiei mesajului) SOAP. Într-un mesaj SOAP cu mai multe părți MIME, corpul SOAP este prima parte a mesajului, iar atașamentul sau atașamentele sunt în părțile ulterioare.

Trimițând sau primind un atașament cu referință într-un mesaj SOAP, datele binare care alcătuiesc atașamentul (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 ținute într-un element XML.

Iată un eșantion 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:0FE43E4D025F0BF3DC11582467646812" />
          </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--

```

Rețineți că în eșantionul MTOM, tipul de conținut pentru anvelopa SOAP este **application/xop+xml** și 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@apache.org" />
```

Procesarea la intrare a atașamentelor cu referință

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

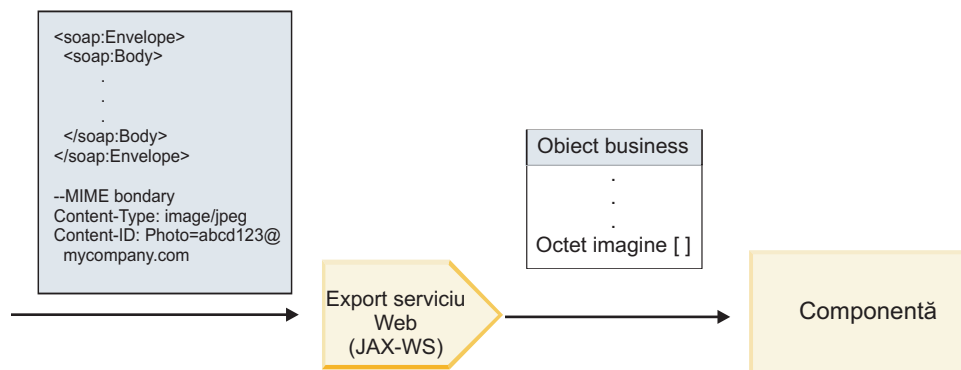


Figura 58. Cum procesează legarea export (JAX-WS) a serviciului web un mesaj SOAP cu un atașament referențiat

Atribute atașament MTOM

- MTOM poate suporta elemente atașament în structuri imbricate.
- MTOM este disponibil numai pentru tipul base64Binary.
- MTOM poate suporta elemente atașament în structuri imbricate, ceea ce înseamnă că **bodyPath** pentru atașamentele MTOM este locația **xpath** pentru elementul unde este ținut atașamentul MTOM. Logica de calcul pentru **bodyPath** urmează strict schema de generare a locației **xpath** așa 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 atașament amestecate nu sunt suportate, ceea ce înseamnă că, dacă MTOM este activat pe legarea de import, se va genera atașamentul 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.

Atașamente către care se face referință: atașamente de tip swaRef: **Advanced**

Aveți posibilitatea să trimiteți și să primiți mesaje SOAP care includ atașamente care sunt reprezentate în interfața serviciului sub formă de elemente de tip swaRef.

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

În mesajul SOAP, corpul conține un element de tip swaRef care identifică ID-ul conținutului atașamentului.

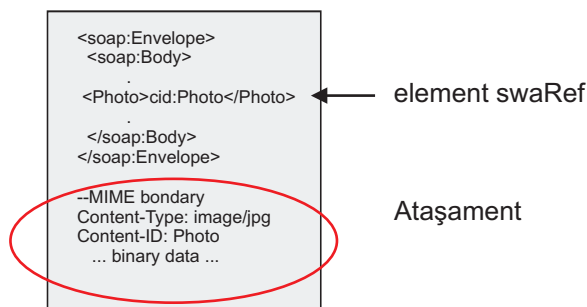


Figura 59. Un mesaj SOAP cu un element swaRef

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

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

WSDL-ul ar trebui să conțină de asemenea o legare MIME care să indice mesajele MIME compuse din mai multe părți 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 părților componente de nivel înalt ale mesajului.

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

Procesarea la intrare a atașamentelor

Folosiți Integration Designer pentru a configura o legare de export, astfel încât să recepționeze atașamentul. Creați un modul și interfața și operațiile sale asociate, incluzând un element de tip swaRef. Puteți crea apoi o legare (JAX-WS) pentru serviciul web.

Notă: Vedeți subiectul “Lucrul cu atașamente” din Centrul de informare Integration Designer pentru informații mai detaliate.

Atunci când un client transmite un mesaj SOAP cu un atașament swaRef către o componentă SCA (Service Component Architecture), legarea de export (JAX-WS) a serviciului web înlătură mai întâi atașamentul. Apoi transmite partea de tip SOAP a mesajului și creează un obiect business. În sfârșit, legarea setează Id-ul conținutului atașamentului în obiectul business.

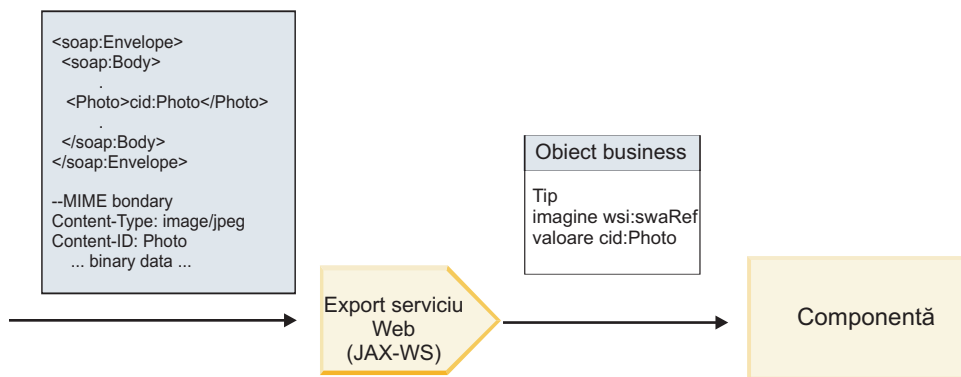


Figura 60. Modul în care o legare externă (JAX-WS) a serviciului web procesează un mesaj SOAP cu un atașament swaRef

Accesarea metadatelor atașamentului într-o componentă a fluxului de mediere

Așa cum se arată în Figura 16 la pagina 80, atunci când atașamentele de tip swaRef sunt accesate de componente, identificatorul conținutului atașamentului apare sub forma unui element de tip swaRef.

Fiecare atașament 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 conținutului atașamentului și ID-ul conținutului, precum și datele binare reale ale atașamentului.

Pentru a putea obține valoarea unui atașament de tip swaRef, este necesară obținerea valorii elementului de tip swaRef, iar apoi localizarea elementului **attachments** cu valoarea **contentID** corespunzătoare. Rețineți că valoarea **contentID** are de obicei prefixul **cid:** înlăturat din valoarea swaRef.

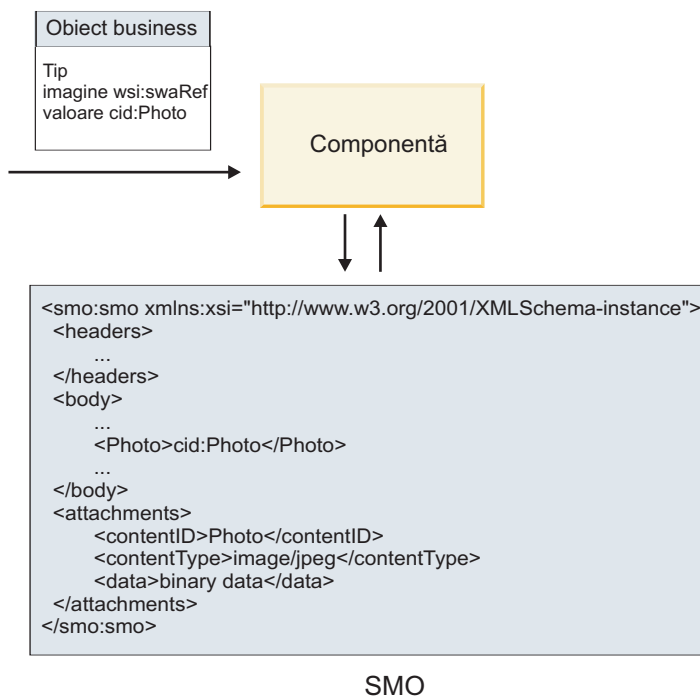


Figura 61. Modul în care atașamentele swaRef apar în SMO

Procesarea la ieșire

Utilizați 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 și definește atașamentul care va fi trimis către serviciul web.

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

Pentru procesarea la ieșire, elementele de tip swaRef sunt trimise întotdeauna împreună cu valorile lor din ID-ul conținutului; totuși, modulul de mediere trebuie să se asigure că există un element **attachments** corespunzător valorii **contentID**.

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

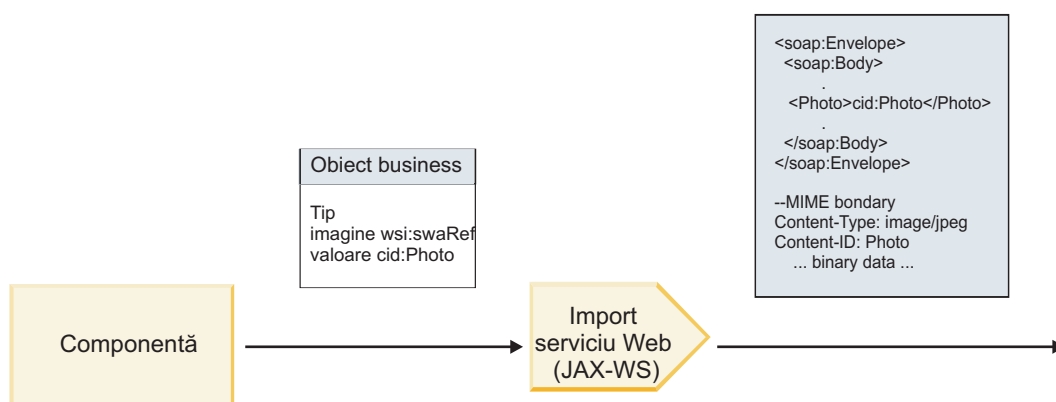


Figura 62. Modul în care o legare de import (JAX-WS) a serviciului web generează un mesaj SOAP cu un atașament swaRef

Setarea metadatelor atașamentului într-o componentă a fluxului de mediere

În cazul în care, în SMO există o valoare pentru un element de tip swaRef și un element **attachments**, legarea pregătește mesajul SOAP (cu atașament) și îl trimite către un destinatar.

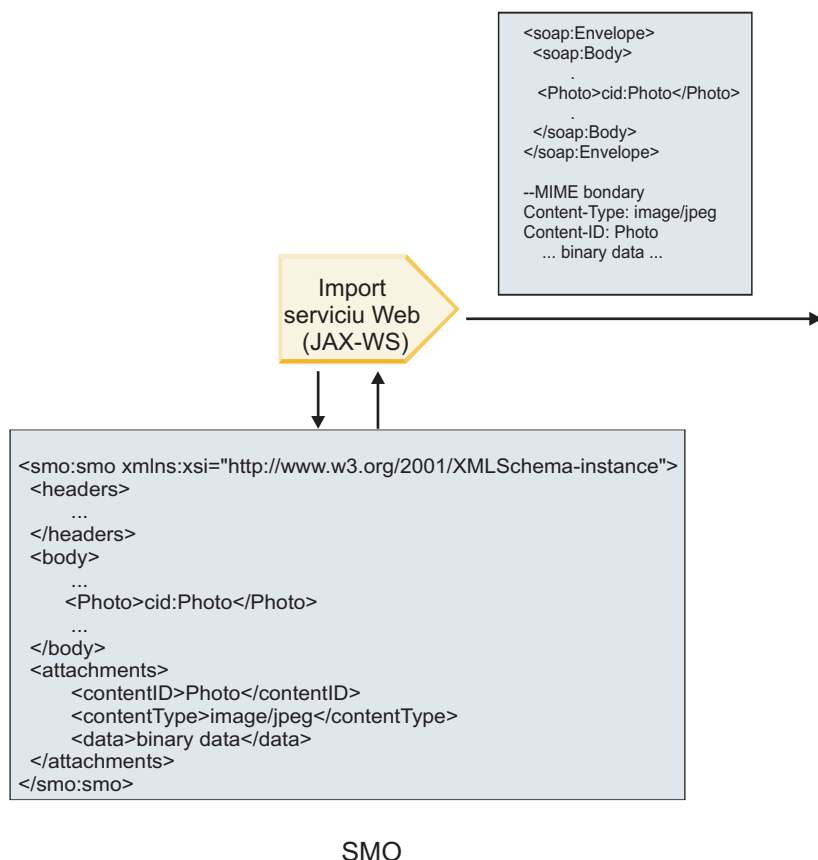


Figura 63. Modul în care un atașament de tip swaRef din SMO este accesat pentru a crea un mesaj de tip SOAP

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 conține alte tipuri de componente, ar trebui să fie utilizată o componentă a fluxului de mediere pentru a copia valorile într-o locație în care pot fi accesate din modul, și o altă componentă a fluxului de mediere utilizată pentru a seta valorile corecte înainte de o invocare de ieșire prin intermediul unui serviciu web de import.

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

Atunci când trimiteți mesaje SOAP cu atașamente, elementul rădăcină pe care îl alegeți determină modul în care atașamentele sunt propagate.

- Dacă folosiți “/body” drept rădăcină a mapării XML, toate atașamentele sunt propagate implicit de-a lungul mapării.
- Dacă folosiți “/” drept rădăcină pentru mapare, puteți controla propagarea atașamentelor.

Atașamente către care se face referință: părți componente de nivel înalt ale mesajului: **Advanced**

Aveți posibilitatea să trimiteți și să primiți mesaje SOAP care includ atașamente binare ce sunt declarate ca părți componente în interfața serviciului dumneavoastră.

Într-un mesaj SOAP cu mai multe părți MIME, corpul SOAP este prima parte a mesajului, iar atașamentul sau atașamentele se află în părțile următoare.

Care este avantajul de a trimite sau de a primi un atașament cu referință într-un mesaj SOAP? Datele binare care alcătuiesc atașamentul (care este adesea destul de mare), sunt ținute 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 ținute într-un element XML.

Tipuri de mesaje SOAP cu atașamente către care se face referință

Începând cu versiunea 7.0.0.3 a IBM Business Process Manager, aveți 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* și 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 atașamente, se utilizează codarea părții legate de id-ul conținutului (așa cum este descris în *WS-I Attachments Profile Version 1.0*) pentru a raporta atașamentul de partea componentă a mesajului.

- **Mesaje care nu sunt conforme cu WS-I**

Runtime-ul poate genera mesaje SOAP care nu sunt conforme cu profilurile WS-I, dar care sunt compatibile 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 reține atașamentul **content-id**, dar codarea părții content-id nu este folosită (așa cum este descrisă în *WS-I Attachments Profile Version 1.0*).

Selectarea compatibilității WS-I pentru exporturile de servicii web

Folosiți Integration Designer pentru a configura o legare de export. Creați un modul și interfața și operațiile sale asociate. Puteți crea apoi o legare (JAX-WS) pentru serviciul web. Pagina Atașamente cu referință afișează toate părțile binare care aparțin de operațiile create, apoi selectați care părți vor fi atașate. Apoi specificați, 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 selectați această opțiune, specificați de asemenea și care parte a mesajului ar trebui să fie legată de corpul SOAP.

Notă: Această opțiune poate fi utilizată doar atunci când fișierul WSDL corespunzător este de asemenea conform cu WS-I.

Un fișier WSDL care este generat de Integration Designer Versiunea 7.0.0.3 este în conformitate cu WS-I. Totuși, dacă importați un fișier WSDL care nu este în conformitate cu WS-I, nu puteți selecta această opțiune.

- **Utilizare mesaje SOAP care nu sunt conforme cu WS-I AP 1.0**

Dacă selectați această opțiune, care este implicită, prima parte a mesajului este legată de corpul SOAP.

Notă: Doar părțile de nivel înalt ale mesajului (adică, elementele definite în portType WSDL drept părți din cadrul mesajului de intrare sau de ieșire) care au un tip binar (fie base64Binary, fie hexBinary) pot fi trimise sau primite sub formă de atașamente cu referință.

Vedeți subiectul “Lucrul cu atașamente” 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 **name** din elementul **wsdl:part** către care se face referință prin **mime:content**
- Caracterul **=**
- O valoare unică globală, precum un UUID
- Caracterul **@**
- Un nume de domeniu valid

Procesarea la intrare a atașamentelor cu referință

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

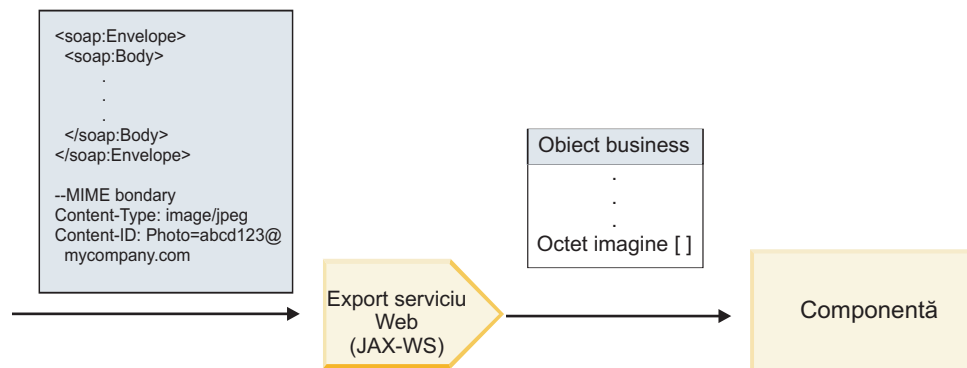


Figura 64. Modul în care o legare externă (JAX-WS) a serviciului web procesează un mesaj SOAP conform WS-I cu un atașament către care se face o referință

Accesarea metadatelor atașamentului într-o componentă a fluxului de mediere

Așa cum se arată în Figura 19 la pagina 84, atunci când atașamentele cu referință sunt accesate de componente, datele atașamentului apar sub forma unei matrice cu date de tip octet.

Fiecare atașament cu referință al unui mesaj SOAP are, de asemenea, un element **attachments** corespunzător în SMO. Elementul **attachments** include tipul conținutului atașamentului și calea către corpul mesajului în care este păstrat atașamentul.

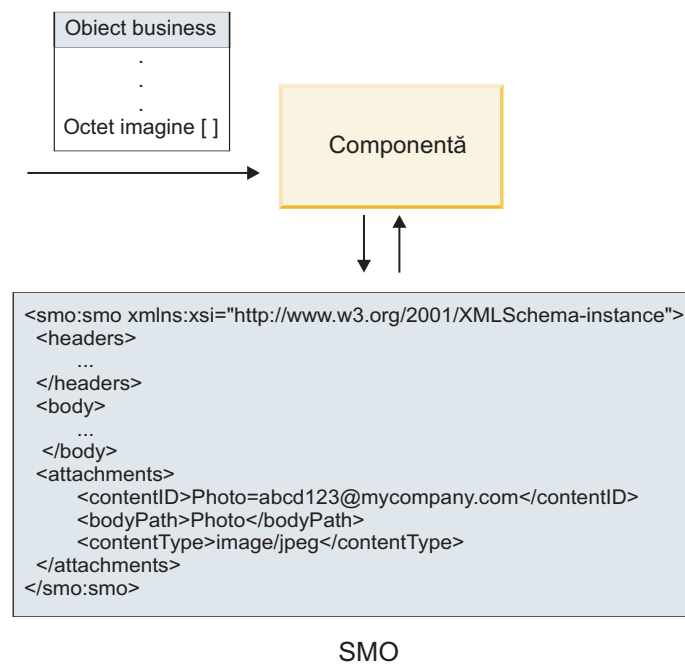


Figura 65. Modul în care atașamentele către care se face referință apar în SMO

Important: Calea către elementul corp al mesajului nu este actualizată automat în cazul în care mesajul este transformat și atașamentul mutat. Puteți 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 ieșire

Utilizați 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 și definește ce părți din mesaj ar trebui să fie trimise ca atașamente. De asemenea, puteți indica, în pagina Specificare compatibilității cu WS-I AP 1.0 a Integration Designer, una dintre următoarele alegeri:

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

Dacă selectați această opțiune, specificați de asemenea și care parte a mesajului ar trebui să fie legată de corpul SOAP; toate celelalte sunt legate de atașamente sau anteturi. Mesajele trimise de legare nu includ elemente în corpul SOAP care se referă la atașamente; relația este exprimată prin intermediul ID-ului de conținut al atașamentului, inclusiv numele părții mesajului.

- **Utilizare mesaje SOAP care nu sunt conforme cu WS-I AP 1.0**

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

Notă: Partea care reprezintă un atașament, așa 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 atașament.

Procesarea la intrare a atașamentelor cu referință

Legarea de import folosește informațiile din SMO pentru a determina modul în care părțile componente de nivel înalt ale mesajului sunt trimise ca atașamente.

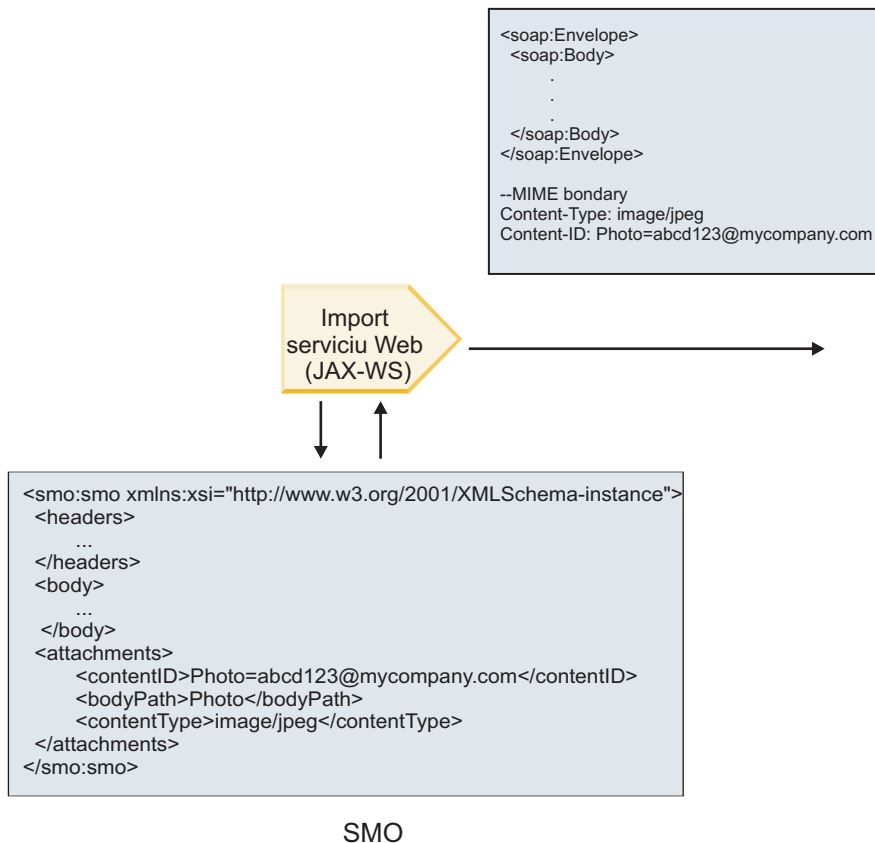


Figura 66. Modul în care atașamentul cu referință din SMO este accesat pentru a crea mesajul SOAP

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 conține alte tipuri de componente, ar trebui să fie utilizată o componentă a fluxului de mediere pentru a copia valorile într-o locație în care pot fi accesate din modul, și o altă componentă a fluxului de mediere utilizată pentru a seta valorile corecte înainte de o invocare de ieșire prin intermediul unui serviciu web de import.

Legarea folosește o combinație între următoarele condiții 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 conținutului
- În cazul în care există un element **attachments** în SMO a cărui valoare **bodyPath** face referință către o parte binară de nivel înalt

Modul în care sunt create atașamentele atunci când există niciun element attachment în SMO

Tabelul următor arată modul în care este creat și trimis un atașament în cazul în care SMO conține un element **attachment** cu un **bodyPath** care se potrivește cu o parte a numelui mesajului:

Tabela 61. Modul în care este generat atașamentul

Starea legării WSDL MIME pentru partea componentă binară de nivel înalt a mesajului	Modul în care mesajul este creat și transmis
Prezent cu una dintre următoarele: <ul style="list-style-type: none"> Nu este definit un tip de conținut pentru partea componentă a mesajului Sunt definite mai multe tipuri de conținut Este definit tipul conținutului metacaracterului 	<p>Partea componentă a mesajului este trimisă ca un atașament.</p> <p>Content-Id este setat cu o valoare în elementul attachments, în cazul în care acesta este prezent; altfel, se generează una.</p> <p>Content-Type este setat cu o valoare în elementul attachments în cazul în care acesta este prezent; altfel, este setat cu application/octet-stream.</p>
Prezent cu conținut unic, fără metacaractere pentru partea componentă a mesajului	<p>Partea componentă a mesajului este trimisă ca un atașament.</p> <p>Content-Id este setat cu o valoare în elementul attachments, în cazul în care acesta este prezent; altfel, se generează una.</p> <p>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 conținut WSDL MIME.</p>
Nu este prezent	<p>Partea componentă a mesajului este trimisă ca un atașament.</p> <p>Content-Id este setat cu o valoare în elementul attachments, în cazul în care acesta este prezent; altfel, se generează una.</p> <p>Content-Type este setat cu o valoare în elementul attachments în cazul în care acesta este prezent; altfel, este setat cu application/octet-stream.</p> <p>Notă: Trimițând părți componente ale sub formă de atașamente 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.</p>

Modul în care sunt create atașamentele atunci când nu există niciun element attachment în SMO

Tabelul următor arată modul în care este creat și trimis n atașament în cazul în care SMO nu conține un element **attachment** cu un **bodyPath** care se potrivește cu o parte din numele mesajului:

Tabela 62. Modul în care este generat atașamentul

Starea legării WSDL MIME pentru partea componentă binară de nivel înalt a mesajului	Modul în care mesajul este creat și transmis
Prezent cu una dintre următoarele: <ul style="list-style-type: none"> Nu este definit un tip de conținut pentru partea componentă a mesajului Sunt definite mai multe tipuri de conținut Este definit tipul conținutului metacaracterului 	<p>Partea componentă a mesajului este trimisă ca un atașament.</p> <p>Content-Id este generat.</p> <p>Content-Type este setat cu application/octet-stream.</p>
Prezent cu conținut unic, fără metacaractere pentru partea componentă a mesajului	<p>Partea componentă a mesajului este trimisă ca un atașament.</p> <p>Content-Id este generat.</p> <p>Content-Type este setat cu tipul definit în elementul conținut WSDL MIME.</p>
Nu este prezent	Partea componentă a mesajului nu este trimisă ca un atașament.

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

Atunci când trimiteți mesaje SOAP cu atașamente, elementul rădăcină pe care îl alegeți determină modul în care atașamentele sunt propagate.

- Dacă folosiți “/body” drept rădăcină a mapării XML, toate atașamentele sunt propagate implicit de-a lungul mapării.
- Dacă folosiți “/” drept rădăcină pentru mapare, puteți controla propagarea atașamentelor.

Atașamente fără referință: **Advanced**

Aveți posibilitatea să trimiteți și să primiți atașamente *fără referință* care nu sunt declarate în interfața serviciului.

Într-un mesaj SOAP cu mai multe părți MIME, corpul SOAP este prima parte componentă a mesajului, iar atașamentele se află în părțile componente următoare. Nu este inclusă nicio referință către atașament în corpul SOAP.

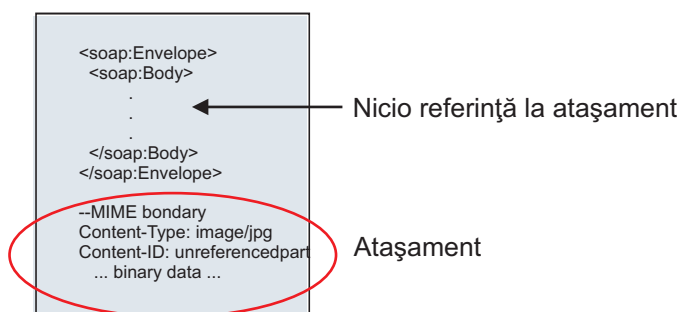


Figura 67. Un mesaj SOAP cu un atașament către care nu se face referință

Aveți posibilitatea să trimiteți un mesaj SOAP cu un atașament fără referință printr-un export de serviciu web către importul unui serviciu web. Mesajul de ieșire, care este trimis la serviciul web țintă, conține atașamentul.

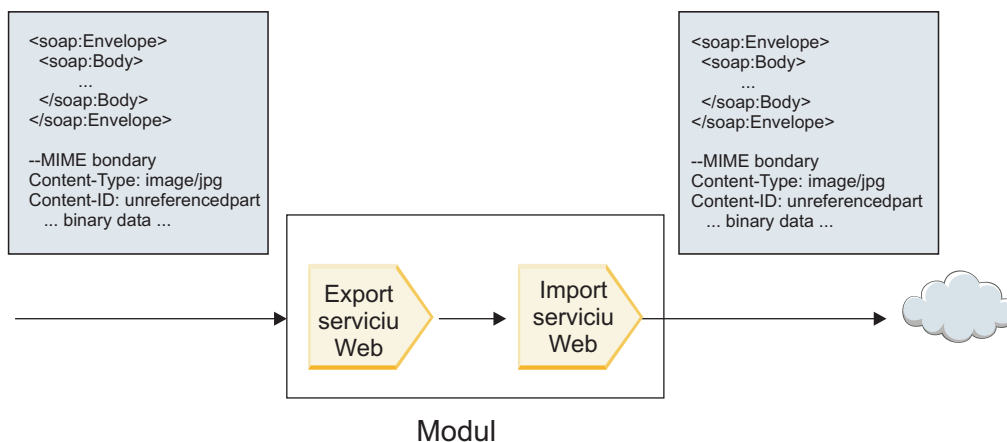


Figura 68. Un atașament care trece printr-un modul SCA

În Figura 23 la pagina 88, mesajul SOAP, cu atașament, este transmis fără modificări.

De asemenea, puteți modifica mesajul SOAP prin utilizarea unei componente de tip flux de mediere. De exemplu, aveți posibilitatea să utilizați componenta de tip flux de mediere pentru a extrage datele din mesajul SOAP (în acest caz, date binare din corpul mesajului) și pentru a crea un mesaj SOAP cu atașamente. Datele sunt prelucrate ca parte componentă

a elementului ce conține atașamentele dintr-un SMO (service message object).

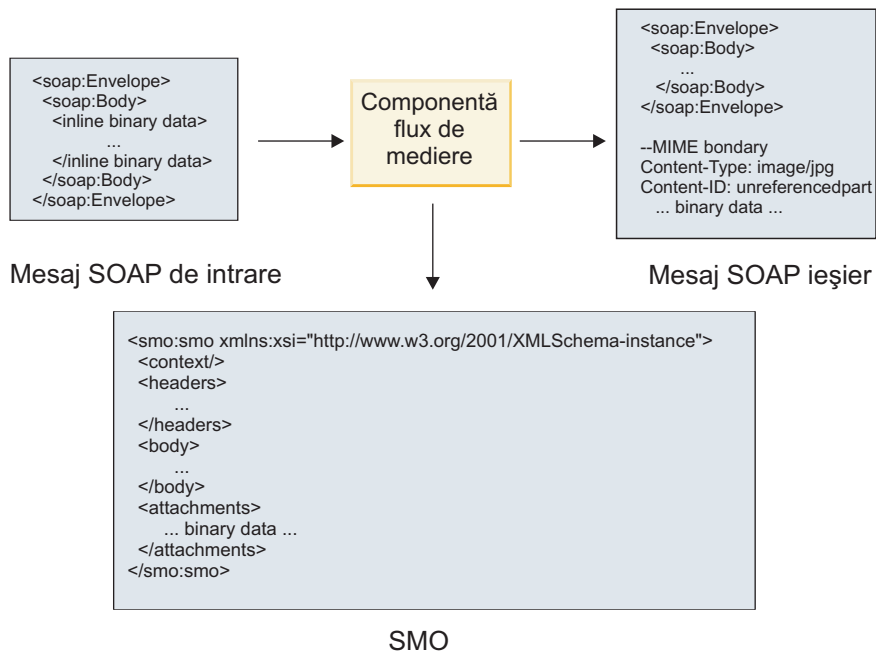


Figura 69. Un mesaj procesat de o componentă a fluxului de mediere

În schimb, componenta de flux de mediere se poate transforma mesajul de intrare prin extragerea și codificarea atașamentului, iar apoi transmite mesajul fără niciun atașament.

În loc de extragerea datelor dintr-un mesaj SOAP de intrare pentru a forma un mesaj SOAP cu atașamente, puteți obține datele atașamentului la o sursă externă, cum ar fi o bază de date.

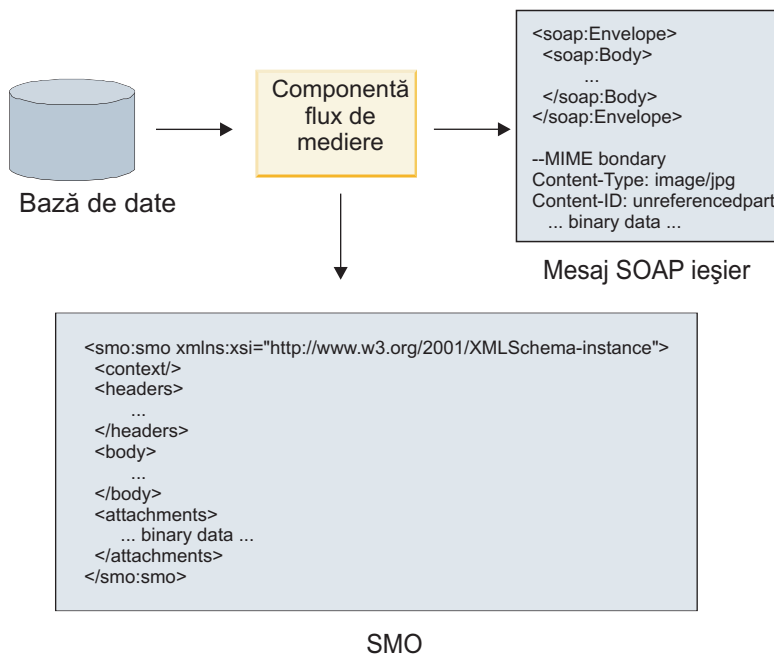


Figura 70. Un atașament obținut dintr-o bază de date și adăugat la mesajul SOAP

În schimb, componenta fluxului de mediere poate extrage atașamentul dintr-un mesaj SOAP de intrare și poate procesa mesajul (de exemplu, păstrează atașamentul într-o bază de date).

Atașamentele fără referință pot fi transmise doar peste componentele fluxului de mediere. În cazul în care un atașament trebuie să fie accesat sau transmis către o componentă de un alt tip, folosiți o componentă a fluxului de mediere pentru a muta atașamentul la o locație care este accesibilă acelei componente.

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

Atunci când trimiteți mesaje SOAP cu atașamente, elementul rădăcină pe care îl alegeți determină modul în care atașamentele sunt propagate.

- Dacă folosiți `"/body"` drept rădăcină a mapării XML, toate atașamentele sunt propagate implicit de-a lungul mapării.
- Dacă folosiți `"/"` drept rădăcină pentru mapare, puteți controla propagarea atașamentelor.

Utilizarea legării stilului documentului WSDL cu mesaje multiple: Advanced

Organizația 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 și modul în care mesajele SOAP corespunzătoare ar trebui să fie formate, în scopul asigurării interoperabilității.

Aceste reguli sunt specificate în *WS-I Basic Profile Version 1.1* (<http://www.ws-i.org/Profiles/BasicProfile-1.1.html>). În particular, WS-I Basic Profile 1.1 R2712 stabilește: "O legare document-literal TREBUIE serializată ca o ENVELOPE cu un soap:Body a cărui element copil este o instanță a declarației de element global referit de partea `wSDL:message` corespondentă."

Aceasta înseamnă că, atunci când utilizați o legare SOAP de stil document pentru o operație cu mesaje (intrare, ieșire sau greșală) care sunt definite cu părți componente multiple, doar una dintre acele părți ar trebui să fie legată de corpul SOAP pentru a fi compatibilă cu WS-I Basic Profile 1.1.

În continuare, WS-I Attachments Profile 1.0 R2941 stabilește: "`wSDL:binding` într-o DESCRIPTION TREBUIE să lege fiecare `wSDL:part` a unui `wSDL:message` în `wSDL:portType` la care se referă din `soapbind:body`, `soapbind:header`, `soapbind:fault`, `soapbind:headerfault` sau `mime:content`".

Aceasta înseamnă că, atunci când utilizați o legare SOAP de stil document pentru o operație cu mesaje (intrare, ieșire sau greșală) care sunt definite cu părți componente multiple, toate părțile componente în afară de cea selectată care vor fi legate la corpul SOAP trebuie să fie legate ca atașamente sau anteturi.

Următoarea abordare este utilizată atunci când descrierile WSDL sunt generate pentru exporturi cu legări de servicii web (JAX-WS și JAX-RPC) în acest caz:

- Puteți 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 părți componente ale mesajului de tip "hexBinary" sau "base64Binary" sunt legate ca atașamente referite. Vedeți "Atașamente către care se face referință: părți componente de nivel înalt ale mesajului" la pagina 82.
- Toate celelalte părți componente ale mesajului sunt legate ca anteturi SOAP.

Legările de import JAX-RPC și JAX-WS respectă legarea SOAP într-un document WSDL existent cu mesaje compuse de tip document chiar dacă nu leagă mai multe părți la corpul SOAP; totuși, nu sunteți în măsură să generați clienți pentru serviciile web pentru astfel de documente WSDL în Rational Application Developer.

Notă: Legarea JAX-RPC nu suportă atașamente.

Tiparul recomandat când utilizați mesaje multiple cu o operație care are legare SOAP de stil document este prin urmare:

1. Utilizați stilul înfășurat document/literal. În acest caz, mesajele au mereu o singură parte componentă; totuși, atașamentele trebuie să fie nereferite (după cum este descris în "Atașamente fără referință" la pagina 88) sau swaRef-typed (după cum este descris în "Atașamente către care se face referință: atașamente de tip swaRef" la pagina 78) în acest caz.
2. Utilizați stilul RPC/literal. În acest caz, nu există restricții asupra legării WSDL în ceea ce privește numărul de părți componente legate de corpul SOAP; mesajul SOAP care rezultă are întotdeauna un singur copil care reprezintă operația care este invocată, cu părțile componente ale mesajului fiind copiii acelui element.
3. Pentru legarea JAX-WS, trebuie să aveți cel mult o parte componentă mesaj care nu este de tip "hexBinary" sau "base64Binary", doar dacă nu este acceptabil să legați celelalte părți componente non-binare la anteturi SOAP.
4. Orice alte cazuri sunt supuse comportamentului descris.

Notă: Există restricții suplimentare când utilizați 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 recepționați mesaje SOAP stil document multiple cu atașamente referite, legarea JAX-WS așteaptă ca fiecare atașament referit să fie reprezentat de un element copil al corpului SOAP cu o valoare atribut href care identifică atașamentul după ID-ul conținutului său. Legarea JAX-WS trimite atașamente referite pentru astfel de mesaje în același mod. Acest comportament nu este compatibil cu *WS-I Basic Profile*.

Pentru a vă asigura că mesajele dumneavoastră sunt compatibile cu Basic Profile, urmați abordarea 1 la pagina 91 sau 2 la pagina 91 din lista anterioară sau evitați utilizarea atașamentelor referite pentru astfel de mesaje și utilizați atașamente nereferite sau swaRef-typed în loc.

Legări HTTP

Legarea HTTP este proiectată să furnizeze conectivitate SCA (Service Component) la HTTP. În consecință, aplicații 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 informațiilor pe web. Când lucrați cu o aplicație externă ce folosește protocolul HTTP, este necesară o legare HTTP. Legarea HTTP transformă datele transmise ca un mesaj într-un format nativ unui obiect business într-o aplicație SCA. Legarea HTTP poate de asemenea transforma datele transmise în afară ca un obiect business în formatul nativ așteptat de aplicația externă.

Notă: Dacă doriți să interacționați cu clienții și serviciile care utilizează protocolul SOAP/HTTP pentru serviciile web, luați în considerare utilizarea uneia dintre legările serviciilor web, care oferă funcționalități suplimentare cu privire la manipulare standardului calității serviciilor pentru serviciile web.

Câteva scenarii comune pentru utilizarea legării HTTP sunt descrise în următoarea listă:

- Serviciile găzduite de SCA pot invoca aplicații HTTP folosind un import HTTP.
- Serviciile găzduite de SCA se pot expune ca aplicații cu HTTP activat, pentru a putea fi folosite de clienți HTTP, folosind un export HTTP.
- IBM Business Process Manager și Process Server pot comunica între ei peste o infrastructură HTTP, în consecință utilizatorii își pot gestiona comunicările conform standardelor de corporație.
- IBM Business Process Manager și Process Server pot acționa ca mediatori de comunicații HTTP, transformând și direcționând mesaje, ce îmbunătățește integrarea aplicațiilor folosind o rețea HTTP.
- IBM Business Process Manager și Process Server pot fi folosite ca o punte între HTTP și alte protocoale, precum servicii Web SOAP/HTTP, adaptoare de resurse bazate pe JCA (Java Connector Architecture), JMS, și așa mai departe.

Informații detaliate despre crearea legărilor de import și export HTTP pot fi găsite în centrul de informare Integration Designer. Vedeți subiectele **Dezvoltarea aplicațiilor de integrare > Accesarea serviciilor externe cu HTTP**.

Privire generală asupra legărilor HTTP:

Legarea HTTP oferă conectivitate la aplicații găzduite-HTTP. Mediază comunicarea dintre aplicații HTTP și permite aplicațiilor bazate pe HTTP existente să fie apelate dintr-un modul.

Legări de import HTTP

Legarea de import HTTP oferă conectivitate de ieșire de la aplicații SCA (Service Component Architecture) la un server sau aplicații 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 țintă SMO.
- URL-ul poate fi specificat ca o proprietate de configurare pe import.

Această invocare este întotdeauna sincronă în natură.

Deși invocările HTTP sunt întotdeauna cerere-răspuns, importul HTTP suportă și operații cu sens unic și cu sens dublu și ignoră răspunsul în cazul unei operații cu sens unic.

Legări de export HTTP

Legarea de export HTTP oferă conectivitate de intrare de la aplicații HTTP la o aplicație SCA.

Un URL este definit pe exportul HTTP. Aplicațiile 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 aplicația SCA către serviciul web extern. Cererea se face din aplicația SCA către serviciul web extern, după cum se arată în Figura 26 la pagina 92.

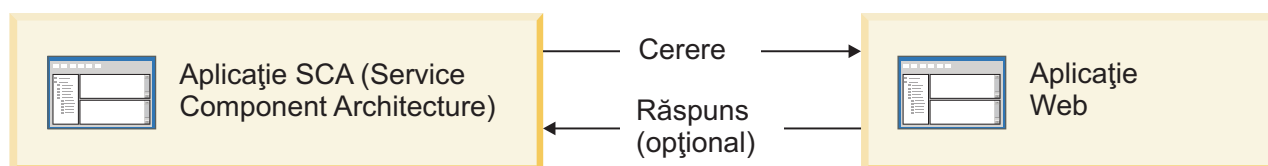


Figura 71. Fluxul unei cereri de la aplicația SCA către aplicația web

Opțional, importul cu legarea HTTP poate primi date înapoi de la aplicația web într-un răspuns la cerere.

Cu un export, cererea se face de către o aplicație client către un serviciu web, după cum se arată în Figura 27 la pagina 92.

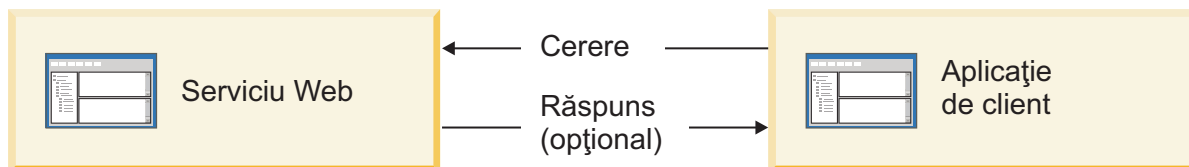


Figura 72. Fluxul unei cereri de la aplicația client la serviciul web.

Serviciul web este o aplicație web care rulează pe server. Exportul este implementat în aplicația web ca un servlet, astfel încât clientul trimite cererea către o adresă URL. Servletul transmite cererea aplicației SCA în timpul rulării.

Opțional, exportul poate trimite date aplicației client ca răspuns la cerere.

Anteturi HTTP:

Legările de import și export HTTP permit ca configurarea anteturilor HTTP și a valorilor lor să fie folosită pentru mesaje de ieșire. Importul HTTP folosește aceste anteturi pentru cereri și exportul HTTP le folosește pentru răspunsuri.

Anteturile configurate static și informațiile de control prevalează asupra valorilor setate dinamic în timpul rulării. Totuși, valorile de control ale URL-ului de înlocuire dinamic, Versiunii și Metodei înlocuiesc valorile statice, ce sunt altfel considerate implicite.

Legarea suportă natura dinamică a URL-ului importului HTTP prin determinarea valorii URL-ului țintă HTTP, Versiunii și Metodei la momentul rulării. Aceste valori sunt determinate prin extragerea valorii Referinței de punct final, URL-ului de înlocuire dinamic, Versiunii și Metodei.

- Pentru Referința punctului final, folosiți API-uri `com.ibm.websphere.sca.addressing.EndpointReference` APIs sau setați câmpul `/headers/SMOHeader/Target/address` în antetul SMO.
- Pentru URL de înlocuire dinamic, Versiune și Metodă, folosiți secțiunea parametrilor de control HTTP a mesajului SCA (Service Component Architecture). Rețineți că URL-ul de înlocuire dinamic are precedență în fața Referinței de punct final țintă; totuși, Referința punctului final se aplică peste legări, deci este abordarea preferată și ar trebui folosită unde este posibil.

Informațiile de control și antet pentru mesaje de ieșire sub legările de export și import HTTP sunt procesate în următoarea ordine:

1. Informații de antet și control excluzând URL de înlocuire dinamic HTTP, Versiune și 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. Adresă țintă specificată prin calea Referinței punctului final sau antetului SMO
5. URL de înlocuire dinamic, Versiune și Metodă din mesajul SCA
6. Informații de antet și control de la handler-ul de date sau legarea de date (cea mai înaltă prioritate)

Exportul și importul HTTP vor popula anteturile de direcție de intrare și parametri de control cu date din mesajul de intrare (`HTTPExportRequest` și `HTTPImportResponse`) doar dacă propagarea antetului de protocol este setată la **True**. Invers, exportul și importul HTTP vor citi și procesa anteturile de ieșire și parametri de control (`HTTPExportResponse` și `HTTPImportRequest`) doar dacă propagarea antetului de protocol este setată la **True**.

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 instrucțiunile de procesare ale mesajului din interiorul legării de import sau export și 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 și contextul de utilizator, precum ID cont) de-a lungul unei căi de invocare SCA. În timpul dezvoltării în IBM

Integration Designer, puteți controla propagarea contextului prin proprietățile de import și export. Pentru detalii suplimentare, vedeți informațiile legărilor de import și export din centrul de informare al IBM Integration Designer.

Structuri de antet HTTP și suport furnizate

Tabela 34 la pagina 94 detaliază parametri de cerere/răspuns pentru cereri și răspunsuri Import HTTP și Export HTTP.

Tabela 63. Informații 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: Șirul 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, înlocuiește URL-ul de import HTTP. Scris în mesaj în linia de cerere. Notă: Șirul de interogare face de asemenea parte din parametrul de control URL.	Nesetat	Nesetat	Ignorat
Versiune de înlocuire dinamică	Dacă este setat, înlocuiește Versiunea de import HTTP. Scris în mesaj în linia de cerere.	Nesetat	Nesetat	Ignorat
Metodă de înlocuire dinamică	Dacă este setat, înlocuiește 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-conținut.)	Dacă este prezent, scris în mesaj ca parte din antetul Tip-conținut. 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-conținut	Citit din mesajul cerere, antetul Tip-conținut	Dacă este prezent, scris în mesaj ca parte din antetul Tip-conținut. 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-conținut. Notă: Această valoare a elementului de control ar trebui furnizată de legarea de date.	Citit din mesajul răspuns, antetul Tip-conținut	Citit din mesajul cerere, antetul Tip-conținut	Suportat; scris în mesaj ca parte din antetul Tip-conținut. Notă: Această valoare a elementului de control ar trebui furnizată de legarea de date.

Tabela 63. Informații 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; implicitul este identity)	Dacă este prezent, scris în mesaj ca un antet și controlează cum este codată transformarea mesajului.	Citit din mesajul răspuns	Citit din mesajul de cerere	Dacă este prezent, scris în mesaj ca un antet și controlează cum este codată transformarea mesajului.
Codare de conținut (Valori posibile: gzip, x-gzip, deflate, identity; implicitul este identity)	Dacă este prezent, scris în mesaj ca un antet și 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 și controlează cum sunt codate datele utile (payload).
Lungime-Conținut	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 (conține proprietăți 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 și transmisă ca text în clar.	Nu se aplică	Citit din antetul Autentificare de bază al mesajului răspuns. Prezența acestui antet nu indică faptul că utilizatorul a fost autentificat. Autentificarea ar trebui controlată în configurarea servlet-ului. Notă: Valoarea pentru acest antet va fi codată doar pe protocolul HTTP. În SCA, va fi decodată și transmisă ca text în clar.	Nu se aplică
Proxy (conține proprietăți multiple: Gazdă, Port, Autentificare)	Dacă este prezent, folosit pentru a stabili conexiunea prin proxy.	Nu se aplică	Nu se aplică	Nu se aplică
SSL (conține proprietăți multiple: Keystore, Keystore Password, Trustore, Trustore Password, ClientAuth)	Dacă este populat și url-ul destinație 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) și un mesaj de protocol HTTP, trebuie configurate un handler de date sau o legare de date HTTP. Handler-ele de date oferă o interfață neutră-legării ce

permite reutilizarea peste legări de transport și reprezintă abordarea recomandată; legările de date sunt specifice unei anumite legări de transport. Sunt furnizate clase de legări de date specifice-HTTP; puteți 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 și HTTPServiceGatewayDataBinding) sunt depreciate începând cu IBM Business Process Manager Versiunea 7.0. În loc de a folosi legările de date descrise în acest subiect, considerați următoarele handler-e de date:

- Folosiți SOAPDataHandler în loc de HTTPStreamDataBindingSOAP.
- Folosiți UTF8XMLDataHandler în loc de HTTPStreamDataBindingXML
- Folosiți GatewayTextDataHandler în loc de HTTPServiceGatewayDataBinding

Legările de date sunt furnizate pentru utilizare cu importuri HTTP și exporturi HTTP: legare de date binare, legare de date XML și legare de date SOAP. O legare de date răspuns nu este necesară pentru operații cu sens unic. O legare de date este reprezentată de numele unei clase Java ale cărei instanțe pot converti atât de la HTTP la ServiceDataObject și vice-versa. Un selector de funcții trebuie folosit pe un export care, în conjuncție cu legări de metode, poate determina ce legare de date este folosită și ce operație 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 și nu are restricții pe schema interfeței.
- Legări de date SOAP, ce suportă corpul ca date SOAP. Implementarea legării de date SOAP nu are restricții pe schema interfeței.

Implementarea legărilor de date HTTP personalizate

Această secțiune descrie cum se implementează o legare de date HTTP personalizată.

Notă: Abordarea recomandată este să implementați un handler de date personalizat deoarece poate fi reutilizat peste legări de transport.

HTTPStreamDataBinding este principala interfață pentru tratarea mesajelor HTTP personalizate. Interfața este proiectată să permită tratarea datelor utile mari. Totuși, pentru ca astfel de implementări să funcționeze, această legare de date trebuie să returneze informațiile de control și anteturile înainte de a scrie mesajul în flux.

Metodele și ordinea lor de execuție, listate mai jos, trebuie implementate de legarea de date personalizată.

Pentru a personaliza o legare de date, scrieți o clasă ce implementează HTTPStreamDataBinding. Legarea de date ar trebui să aibă patru proprietăți 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 ieșire (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 invocați 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;
}
/*
 * Add http header "IsBusinessException" in pHeaders.
 * Two steps:
 * 1.Remove all the header with name IsBusinessException (case-insensitive) first.
 * This is to make sure only one header is present.
 * 2.Add the new header "IsBusinessException"
 */
public void setBusinessException(boolean isBusinessException) {
    //remove all the header with name IsBusinessException (case-insensitive) first.
    //This is to make sure only one header is present.
    //add the new header "IsBusinessException", code example:
    HTTPHeader header=HeadersFactory.eINSTANCE.createHTTPHeader();
    header.setName("IsBusinessException");
    header.setValue(Boolean.toString(isBusinessException));
    this.pHeaders.getHeader().add(header);
}
public HTTPControl getControlParameters() {
```

```

    return pCtrl;
}
public HTTPHeaders getHeaders() {
    return pHeaders;
}
public DataObject getDataObject() throws DataBindingException {
    return pDataObject;
}
/*
 * Get header "IsBusinessException" from pHeaders, return its boolean value
 */
public boolean isBusinessException() {
    String headerValue = getHeaderValue(pHeaders, "IsBusinessException");
    boolean result=Boolean.parseBoolean(headerValue);
    return result;
}
public void convertToNativeData() throws DataBindingException {
    DataObject dataObject = getDataObject();
    this.nativeData=realConvertWorkFromSD0ToNativeData(dataObject);
}
public void convertFromNativeData(HTTPInputStream arg0){
    //Customer-developed method to
    //Read data from HTTPInputStream
    //Convert it to DataObject
    DataObject dataobject=realConvertWorkFromNativeDataToSD0(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 operațională Java EE ce rulează pe un server Java EE. Legările de export EJB permit componentelor SCA să fie afișate sub formă de Enterprise JavaBeans, astfel încât logica operațională Java EE să poată invoce 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 își conecteze logica operațională în mediul IBM Business Process Manager și să participe într-un proces operațional.

Folosiți Integration Designer pentru a crea legări de import EJB. Puteți utiliza oricare dintre următoarele proceduri pentru a genera legările:

- Creare import EJB folosind vrăjitorul de servicii externe
Puteți 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, interfețe și importă fișiere în baza serviciilor descoperite.
- Creare legări de import EJB folosind editorul de asamblare
Puteți să creați un import EJB într-o diagramă de asamblare utilizând editorul de asamblare Integration Designer. Din paletă, puteți 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. Aveți posibilitatea să legați în mod direct o componentă cu o referință WSDL (Web Services Description Language) la importul EJB care comunică cu un serviciu bazat pe EJB folosind o interfață Java.

Importul EJB poate interacționa cu logica operațională Java EE folosind fie modelul de programare EJB 2.1, fie modelul de programare EJB 3.0.

Invocarea logicii operaționale Java EE poate fi locală (doar pentru EJB 3.0) sau de la distanță.

- Invocarea locală este utilizată atunci când doriți să apelați logica operațională Java EE care se află pe același server ca și importul.

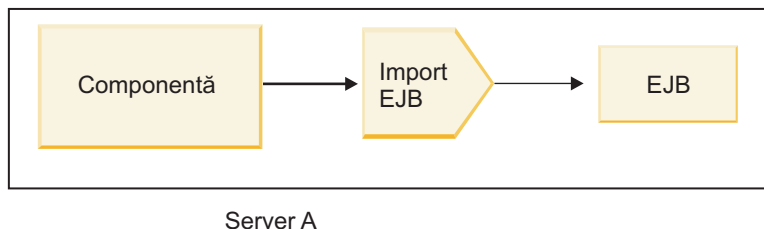


Figura 73. Invocarea locală a unui EJB (doar EJB 3.0)

- Invocarea de la distanță este utilizată atunci când doriți să apelați logica operațională Java EE care nu se află pe același server ca și 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.

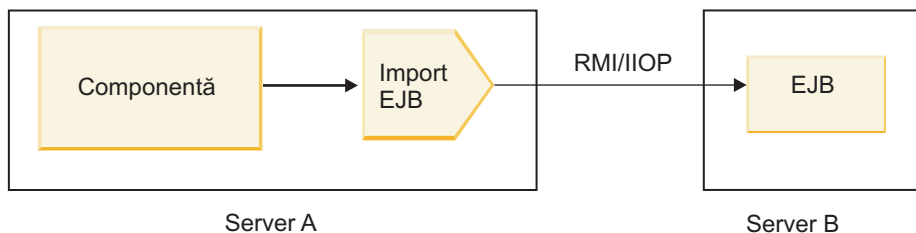


Figura 74. Invocare de la distanță a unui EJB

Atunci când configurează legarea EJB, Integration Designer folosește numele JNDI pentru a determina nivelul modelului de programare EJB și tipul de invocare (ocal sau de la distanță).

Legările de export EJB conțin următoarele componente majore:

- Handler date JAX-WS
- Selector de defectare EJB
- Selector pentru funcția de import EJB

Dacă scenariul dvs. de utilizator nu se bazează pe maparea JAX-WS, s-ar putea să aveți nevoie de un handler de date personalizat, un selectorul de funcție și 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:

Aplicațiile Java EE externe pot invoca o componentă SCA prin intermediul unei legări de export EJB. Folosind un export EJB puteți expune componente SCA, astfel încât aplicațiile Java EE externe pot invoca aceste componente folosind modelul de programare EJB.

Notă: Exportul EJB este un bean stateless.

Folosiți Integration Designer pentru a crea legări EJB. Puteți utiliza oricare dintre următoarele proceduri pentru a genera legările:

- Crearea legărilor de export EJB folosind vrăjitorul pentru servicii externe
Puteți 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, interfețe și exportă fișiere pe baza serviciilor descoperite.
- Crearea legărilor de export EJB folosind editorul de asamblare
Puteți să creați 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.

Puteți genera legarea dintr-o componentă SCA existentă, sau puteți genera un export cu o legare EJB pentru o interfață Java.

- Atunci când generați un export dintr-o componentă SCA existentă care are o interfață WSDL existentă, exportulului i se alocă o interfață Java.
- Atunci când generați un export dintr-o interfață Java, puteți selecta fie un WSDL, fie o interfață Java pentru export.

Notă: O interfață Java folosită pentru a crea un export EJB are următoarele limitări în ceea ce privește obiectele (parametrii de ieșire și intrare și excepții) transmise ca parametrii la un apel de la distanță:

- Trebuie să fie de un tip concret (în loc de un tip interfață sau abstract).
- Acestea trebuie să fie în concordanță cu specificația Enterprise JavaBeans. Trebuie să fie serializabili și să aibă un constructor implicit fără argumente, iar toate proprietățile trebuie să fie accesibile prin intermediul metodelor getter și setter.

Consultați site-ul web Sun Microsystems, Inc., <http://java.sun.com> pentru informații legate de specificațiile JavaBeans Enterprise.

În plus, excepția trebuie să fie o excepție verificată, moștenită din `java.lang.Exception`, și trebuie să fie unică (adică, nu suportă să arunce mai multe tipuri de excepții verificate).

De asemenea, rețineți că interfața afacerii pentru un `EnterpriseBean` Java este o interfață Java simplă și nu trebuie să extindă `javax.ejb.EJBObject` sau `javax.ejb.EJBLocalObject`. Metodele interfeței de afaceri nu ar trebui să arunce `java.rmi.Remote.Exception`.

Legările de export EJB pot interacționa cu logica operațională 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 distanță.

- Invocarea locală este utilizată atunci când logica operațională Java EE apelează o componentă SCA care se află pe același server ca și exportul.
- Invocarea de la distanță este utilizată atunci când logica operațională Java EE nu se află pe același server ca și exportul.

De exemplu, în figura următoare, un EJB folosește RMI/IIOP pentru a apela o componentă SCA aflată pe un server diferit.

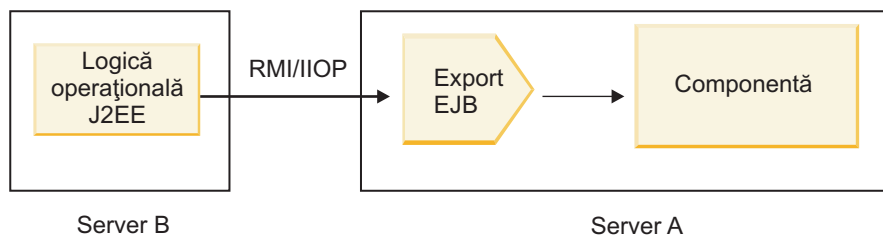


Figura 75. Un apel la distanță de la un client către o componentă SCA prin intermediul unui export EJB

Atunci când configurează legarea EJB, Integration Designer folosește numele JNDI pentru a determina nivelul modelului de programare EJB și tipul de invocare (ocal sau de la distanță).

Legările de export EJB conțin următoarele componente majore:

- Handler date JAX-WS
- Selector pentru funcția de export EJB

Dacă scenariul dvs. de utilizator nu se bazează pe maparea JAX-WS, s-ar putea să aveți nevoie de un handler de date personalizat și de un selectorul de funcție 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.

Proprietățile legărilor EJB:

Legările pentru import EJB își folosesc numele JNDI configurate pentru a determina nivelul modelului de programare EJB și tipul de invocare (local sau de la distanță). Legările de import și export EJB utilizează handler-ul de date JAX-WS pentru transformarea datelor. Legarea de import EJB folosește un selector pentru funcția de import EJB și un selector pentru defectul EJB, iar legarea de export EJB folosește un selector pentru funcția de export EJB.

Nume JNDI și legările de import EJB:

Atunci când configurează legarea EJB pentru un import, Integration Designer folosește numele JNDI pentru a determina nivelul modelului de programare EJB și tipul de invocare (local sau de la distanță).

În cazul în care nu este specificat niciun nume JNDI, se folosește legarea interfeței EJB implicite. Numele implicite care sunt create depind de faptul că invocați EJB 2.1 JavaBeans sau EJB 3.0 JavaBeans.

Notă: Consultați subiectul "Privire generală asupra legărilor de aplicație EJB 3.0" din Centrul de informare WebSphere Application Server pentru informații suplimentare detaliate despre convențiile 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 interfața home, separate prin linii oblice.

De exemplu, pentru interfața acasă a EJB 2.1 JavaBeans pentru `com.mycompany.myremotebusinesshome`, legarea implicită este:

```
ejb/com/mycompany/myremotebusinesshome
```

Pentru EJB 2.1, este suportată doar în vocarea EJB de la distanță.

- EJB 3.0 JavaBeans

Numele JNDI implicit preselectat de Integration Designer pentru JNDI-ul local este numele clasei complet calificate al interfeței locale precedate de **ejblocal:**. De exemplu, pentru interfața complet calificată a interfeței `com.mycompany.mylocalbusiness` locale, EJB-ul 3.0 JNDI preselectat este:

```
ejblocal:com.mycompany.mylocalbusiness
```

Pentru interfața `com.mycompany.myremotebusiness` de la distanță, EJB-ul 3.0 JNDI preselectat este interfața complet calificată:

com.mycompany.myremotebusiness

Legările aplicației implicite EJB 3.0 sunt descrise la următoarea locație: Privire generală asupra legărilor de aplicație EJB 3.0.

Integration Designer va folosi numele "scurt" drept locația JNDI implicită pentru EJB-uri folosind versiunea 3.0 a modelului de programare.

Notă: În cazul în care referința JNDI implementată a țintei EJB este diferită de locația legării JNDI implicite deoarece a fost utilizată sau configurată o mapare personalizată, numele JNDI-ului țintă trebuie să fie specificat în mod corespunzător. Aveți posibilitatea să specificați numele în Integration Designer înainte de implementare, sau, pentru legarea de import, puteți modifica numele în consola administrativă (după implementare) pentru a potrivi numele JNDI pentru EJB țintă.

Pentru informații suplimentare despre crearea legărilor EJB, vedeți secțiunea 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 și pentru a transforma valoarea returnată a obiectului Java în obiecte business răspuns. Legarea EJB de import folosește handler-ul de date JAX-WS pentru a transforma EJB-urile cerere în obiecte business cerere și pentru a transforma obiectul business răspuns într-o valoare returnată.

Acest handler de date mapează datele de la interfața WSDL specificată în SCA cu interfața Java EJB țintă (și vice versa) folosind specificațiile pentru JAX-WS (Java API for XML Web Services) și specificațiile pentru JAXB (Java Architecture for XML Binding).

Notă: Suportul actual este limitat la specificațiile pentru JAX-WS 2.1.1 și JAXB 2.1.3.

Handler-ul de date specificat la nivelul legării EJB este folosit pentru a realiza procesarea cererilor, răspunsurilor, defectelor și excepțiilor apărute în timpul rulării.

Notă: Pentru defecte, poate fi specificat un anumit handler de date pentru fiecare dintre ele prin specificarea proprietății de configurare `faultBindingType`. Acesta înlocuiește valoarea specificată la nivelul legării EJB.

Handler-ul de date JAX-WS este folosit în mod implicit atunci când legarea EJB are o interfață 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 folosește un handler de date pentru a transforma un obiect de date într-o matrice cu elemente de tip `Object` Java (`Object[]`). În timpul comunicațiilor ce au loc la ieșire în, are loc următoarea procesare:

1. Legarea EJB setează tipul așteptat, elementul așteptat și numele metodei țintă 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 reprezentați parametrii metodei (în ordinea definirii lor în cadrul metodei).
4. Legarea EJB folosește `Object[]` pentru a invoca metoda în interfața EJB țintă.

Legarea pregătește 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 necesar pentru a suporta parametrii de tip `In/Out` și `Out`.

Pentru parametrii de tip `Out`, valorile trebuie să fie returnate în obiectul datelor de răspuns.

Handler-ul de date procesează și transformă valorile găsite în Object[], iar apoi returnează un răspuns către obiectul de date.

Handler-ul de date suportă xs:AnyType, xs:AnySimpleType și xs:Any împreună cu alte tipuri de date XSD. Pentru a activa suportul pentru xs:Any, utilizați **@XmlAnyElement (lax=true)** pentru proprietatea JavaBeans în codul Java așa cum este afișat 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 specificația JAX-WS, modificați adnotările JAXB, astfel încât să se potrivească nevoilor afacerii dvs. Handler-ul de date JAX-WS suportă xs:any, xs:anyType și xs:anySimpleType.

Pentru handler-ul de date JAX-WS se aplică următoarele restricții:

- Handler-ul de date nu include suport pentru adnotarea **@WebParam** din atributul antetului.
- Spațiul de nume pentru fișierele schemei obiectului business (fișiere XSD) nu include maparea implicită din numele pachetului Java. Adnotarea **@XMLSchema** din package-info.java, de asemenea, nu funcționează. Singura modalitate de a crea un XSD cu un spațiu de nume este de a utiliza adnotările **@XmlType** și **@XmlRootElement**. **@XmlRootElement** definește spațiul nume țintă pentru elementul global în tipuri JavaBeans.
- Vrăjitorul pentru importul EJB nu creează fișiere XSD pentru clasele fără legătură. Versiunea 2.0 nu suportă adnotare **@XmlSeeAlso**, așa că, dacă nu se face referire în mod direct de la clasa părinte către clasa copil, nu se creează un XSD. Soluția la această problemă este să rulați 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 fișiere XSD pentru un bean dat. Aceste XSD-uri trebuie să fie copiate manual în modulul pentru ca soluția să funcționeze.

Selector defect EJB:

Selectorul defectului EJB determină dacă o invocare EJB a rezultat într-un defect, o excepție 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 excepție într-un obiect business defect.

Într-un răspuns cu succes (fără defect), legarea de import EJB assemblează 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 și Out.

Pentru scenariile de excepție, legarea assemblează un Object[], iar primul element reprezintă excepția aruncată de metodă.

Selectorul de defect poate returna oricare dintre următoarele valori:

Tabela 64. Valori returnate

Tip	Valoare returnată	Descrierea
Defect	ResponseType.FAULT	Returnată atunci când Object[] transmis conține un obiect de tip excepție.
Excepție runtime	ResponseType.RUNTIME	Returnată în cazul în care obiectul de tip excepție nu se potrivește cu niciunul dintre tipurile de excepții 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 și pentru a invoca handler-ul corect pentru datele defecte.

Selector funcție EJB:

Legările EJB folosesc un selector pentru funcția de import (pentru procesarea la ieșire) sau un selector pentru funcția de export (pentru procesarea la intrare) pentru a determina ce metodă EJB să apeleze.

Selector pentru funcția de import

Pentru procesarea la ieșire, selectorul pentru funcția de import derivă tipul metodei EJB în funcție de numele operației invocate de componenta SCA care este legată de importul EJB. Selectorul funcției caută adnotarea @WebMethod în clasa Java adnotată JAX-WS generată de Integration Designer pentru a determina numele operației țintă asociate.

- În cazul în care adnotarea @WebMethod există, selectorul funcției folosește această adnotare pentru a determina mapearea corectă a metodei Java pentru metoda WSDL.
- În cazul în care adnotarea @WebMethod lipsește, selectorul funcției presupune că numele metodei Java este același cu cel al operației invocate.

Notă: Acest selector de funcție este valid doar pentru o interfață de tip WSDL într-un import EJB, și nu pentru o interfață de tip Java dintr-un import EJB.

Selectorul funcției returnează un obiect java.lang.reflect.Method care reprezintă metoda interfeței EJB.

Selectorul funcției folosește un Obiect Java de tip matrice (Object[]) pentru a reține răspunsul de la metoda țintă. 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 funcția de export

Pentru procesarea la intrare, selectorul pentru funcția de export derivă metoda țintă, astfel încât să fie invocată din metoda Java.

Selectorul funcției de export mapează numele operației Java invocate de clientul EJB în numele operației din interfața componentei țintă. Numele metodei este returnat sub forma unui șir de caractere și este rezolvat de runtime-ul SCA în funcție de tipul interfeței al componentei țintă.

Legări EIS

Legările EIS (Enterprise information system) asigură conectivitatea între componente SCA și un EIS extern. Această comunicație este realizată folosind exporturile EIS și importurile EIS care suportă adaptoarele de resurse JCA 1.5 și Websphere Adapters.

Componentele dumneavoastră SCA ar putea impune ca datele să fie transferate către sau de la un EIS extern. Atunci când creați un modul SCA care necesită o astfel de conectivitate, veți include (în plus față de componenta dvs. SCA) un import sau un export cu o legare EIS pentru comunicația cu un anumit EIS extern.

Adaptoarele de resurse din IBM Integration Designer sunt utilizate în contextul unui import sau export. Dezvoltați un import sau un export cu ajutorul vrăjitorului de servicii externe, iar în timpul dezvoltării includeți adaptorul de resurse. Un import EIS care permite aplicației dumneavoastră să invoce un serviciu într-un sistem EIS sau un export EIS care permite unei aplicații dintr-un sistem EIS să invoce un serviciu dezvoltat în IBM Integration Designer sunt create cu un adaptor de resurse. De exemplu, veți crea un import cu adaptorul JD Edwards pentru a invoca un serviciu în sistemul JD Edwards.

Atunci când utilizați vrăjitorul pentru servicii externe, informațiile legate de legarea EIS sunt create pentru dvs. De asemenea, puteți utiliza o altă unealtă, editorul de asamblare, pentru a adăuga sau modifica informațiile legate de legare. Vedeți Accesarea serviciilor externe cu adaptoare pentru informații suplimentare.

După ce modulul care conține legarea EIS este implementat pe server, puteți utiliza consola administrativă pentru a vizualiza informații 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 accesați servicii de pe un sistem de informații de întreprindere sau să vă faceți serviciile disponibile EIS-ului.

Următorul exemplu arată cum un modul SCA numit ContactSyncModule sincronizează informații de contact între un sistem Siebel și un sistem SAP.

1. Componenta SCA numită ContactSync ascultă (prin intermediul unui export de aplicație EIS numit Contact Siebel) modificări asupra contactelor Siebel.
2. Însăși componenta SCA ContactSync folosește o aplicație SAP (printr-un import de aplicație EIS) pentru a actualiza informațiile de contact SAP corespunzător.

Deoarece structurile de date folosite pentru memorarea contactelor sunt diferite în sistemele Siebel și SAP, componenta SCA ContactSync trebuie să ofere mapare.

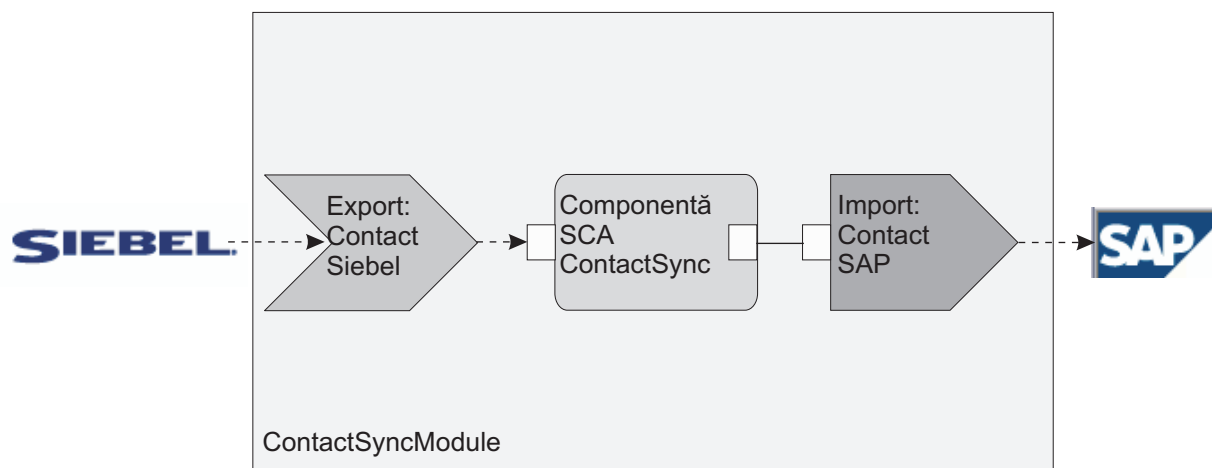


Figura 76. Flux de la un sistem Siebel la un sistem SAP

Exportul Contact Siebel și 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 aplicații 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 și sistemele EIS externe. Aplicațiile externe pot fi tratate ca un import EIS. În acest caz, importul EIS trimite date către EIS extern și primește opțional date în răspuns.

Importul EIS furnizează componentelor SCA cu o vizualizare uniformă a aplicațiilor 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 interfață, expusă de aplicația de import EIS, cu una sau mai multe metode, fiecare luând obiecte de date ca argumente și returnează valori. Pe partea de implementare, există un CCI implementat de un adaptor de resurse.

Implementarea runtime a importului EIS conectează interfața de pe partea clientului și CCI-ul. Importul mapează invocarea metodei pe interfață cu invocarea de pe CCI.

Legările sunt create la trei niveluri: legarea de interfață, ce apoi utilizează legările de metode conținute, ce mai apoi utilizează legările de date.

Legarea de interfață leagă interfața importului cu conexiunea la sistemul EIS ce furnizează aplicația. Acest lucru reflectă faptul că setul de aplicații, reprezentat de interfață, este furnizat de instanța specifică a EIS, iar conexiunea furnizează accesul la această instanță. Elementul de legare conține proprietăți cu destule informații pentru a crea conexiunea (aceste proprietăți fac parte din instanța `javax.resource.spi.ManagedConnectionFactory`).

Legarea de metodă asociază metoda cu interacțiunea specifică cu sistemul EIS. Pentru JCA, interacțiunea este caracterizată de setul de proprietăți al implementării de interfață `javax.resource.cci.InteractionSpec`. Elementul de interacțiune al legării de metodă conține aceste proprietăți, împreună cu numele clasei, astfel furnizând destule informații pentru a realiza interacțiunea. Legarea de metodă utilizează legări de date ce descriu mapearea argumentului și rezultatului metodei de interfață la reprezentarea EIS.

Scenariul runtime pentru un import EIS este după cum urmează:

1. Metoda de pe interfața de import este invocată utilizând modelul de programare SCA.
2. Cererea, ce ajunge la importul EIS, conține numele metodei și argumentele acesteia.
3. Importul întâi creează o implementare legare de interfață; apoi, utilizând date din legarea de import, acesta creează o `ConnectionFactory` și le asociază pe cele două. Adică, importul apelează `setConnectionFactory` pe legarea de interfață.
4. este creată implementarea legării de metodă ce se potrivește cu metoda invocată.
5. Instanța `javax.resource.cci.InteractionSpec` este creată și populată; apoi, legările de date sunt utilizate pentru a lega argumentele metodei la un format înțeles de adaptorul de resurse.
6. Interfața CCI este utilizată pentru a realiza interacțiunea.
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 și un EIS extern. Aplicațiile externe pot fi tratate ca un export EIS. În acest caz, aplicația externă trimite datele sale în formă de notificări periodice. Un export EIS poate fi gândit ca o aplicație de abonare ce ascultă o cerere externă de la EIS. Componenta SCA ce utilizează exportul EIS o vizualizează ca o aplicație locală.

Exportul EIS furnizează componentelor SCA o vizualizare uniformă a aplicațiilor 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 primește cereri de la EIS. Ascultătorul implementează o interfață ascultător specifică adaptorului de resurse. Exportul conține de asemenea o interfață ce implementează componente, expusă la EIS prin export.

Implementarea runtime a unui export EIS conectează ascultătorul cu interfața ce implementează componente. Exportul mapează cererea EIS cu invocarea operațiilor corespunzătoare de pe componente. Legările sunt create la trei niveluri: o legare ascultător, ce apoi utilizează o metodă nativă conținută , ce apoi utilizează o legare de date .

Legarea ascultător leagă ascultătorul ce recepționează cererile cu componenta expusă prin export. Definiția de export conține numele componentei; runtime-ul o localizează și înaintează cererile către aceasta.

Legarea de metodă nativă asociază metoda nativă sau tipul de eveniment recepționat de către ascultător cu operația implementată de componenta expusă prin calea exportului. Nu există nicio relație între metoda invocată pe ascultător și tipul de eveniment; toate evenimentele ajung prin una sau mai multe metode ale ascultătorului. Legarea de metodă nativă utilizează selectorul de funcții definit în export pentru a extrage numele metodei native din datele de intrare și legările de date pentru a lega formatul de date al EIS cu un format înțeles de componentă.

Scenariul runtime pentru un export EIS este după cum urmează:

1. Cererea EIS declanșează invocarea metodei pe implementarea ascultătorului.
2. Ascultătorul localizează și invocă exportul, pasându-i toate argumentele invocării.
3. Exportul creează implementarea legării ascultătorului.
4. Exportul instanțiază selectorul de funcții și îl setează pe legarea ascultătorului.
5. Exportul inițializează legările metodelor native și le adaugă la legarea ascultătorului. Pentru fiecare legare de metodă nativă, legările de date sunt de asemenea inițializate.
6. Exportul invocă legarea ascultătorului.
7. Legarea ascultătorului localizează componentele exportate și utilizează selectorul de funcții pentru a extrage numele metodei native.
8. Acest nume este utilizat pentru a localiza legarea metodei native, ce apoi invocă componente destinație.

Stilul de interacțiune al adaptorului permite legării de export EIS să invoce componenta destinație fie asincron (implicit) fie sincron.

Adaptoare de resurse

Dezvoltați un import sau un export cu vrăjitorul de servicii externe și, în dezvoltarea acestuia, includeți un adaptor de resurse. Adaptoarele ce vin cu IBM Integration Designer folosite pentru a accesa sisteme CICS, IMS, JD Edwards, PeopleSoft, SAP și Siebel sunt intenționate doar pentru scopuri de dezvoltare și testare. Acest lucru presupune că le utilizați pentru a dezvolta și a testa aplicațiile dumneavoastră.

Odată ce ați implementat aplicația, veți avea nevoie de adaptoare runtime licențiate pentru a o rula. Totuși, atunci când construiți serviciul dumneavoastră, puteți îngloba adaptorul împreună cu serviciul. Licențierea adaptorului vă poate permite să utilizați adaptorul înglobat ca adaptor runtime licențiat. 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 aplicații, care oferă gestiunea și securitatea ciclului de viață pentru tranzacții. Acestea se potrivesc de asemenea cu specificațiile Enterprise Metadata Discovery cu excepția adaptorului de resurse IBM CICS ECI și 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 aplicație care este gata de început, împachetată ca fișier EAR și implementată pe server. Toate artefactele Java EE și resursele sunt create; aplicația este configurată și gata de rulare.

Proprietăți dinamice JCA Interaction Spec și Connection Spec:

Legarea EIS poate accepta intrare pentru InteractionSpec și ConnectionSpec specificate prin utilizarea unui obiect de date copil bine-specificat ce acompaniază datele utile. Aceasta permite interacțiuni cerere-răspuns dinamice cu un adaptor de resurse prin InteractionSpec și autentificare de componentă prin ConnectionSpec.

`javax.cci.InteractionSpec` conține informații despre cum ar trebui tratată cererea de interacțiune cu adaptorul de resurse. Poate de asemenea conține informații despre cum a fost obținută interacțiunea după cerere. Aceste comunicări pe două căi prin interacțiuni sunt uneori referite ca *conversații*.

Legarea EIS așteaptă datele utile ce vor fi un argument pentru adaptorul de resurse să conțină un obiect de date copil numit **properties**. Acest obiect de date proprietăți va conține perechi nume/valoare, cu numele proprietăților Interaction Spec într-un anumit format. Regulile de formatare sunt:

- Numele trebuie să înceapă cu prefixul **IS**, urmat de numele proprietății. De exemplu, o `interactionSpec` cu o proprietate JavaBeans numită **InteractionId** ar specifica numele proprietății ca **ISInteractionId**.
- Perechea nume/valoare reprezintă numele și valoarea al tipului simplu al proprietății Interaction Spec.

În acest exemplu, o interfață specifică faptul că intrarea unei operații este un obiect de date **Account**. Această interfață invocă o aplicație de legare de import EIS cu intenția de a trimite și recepționa o proprietate InteractionSpec dinamică numită **workingSet** cu valoarea **xyz**.

Graficul operațional sau obiectele business de pe server conțin un obiect business **properties** de bază ce permite trimiterea datelor specifice-protocolului cu datele utile (payload). Acest obiect business **properties** este încorporat și nu trebuie specificat în schema XML când construiți un obiect business. Trebuie doar creat și utilizat. Dacă aveți definite propriile tipuri de date bazat pe o schemă XML, trebuie să specificați un element **properties** ce vă conține perechile nume/valoare așteptate.

```
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.createElement /
("http://mytest/eis/Account", "AccountWrapper");
```

Creați 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`, setați valoarea așteptată (**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");
```

Puteți folosi proprietăți `ConnectionSpec` pentru autentificarea componentelor dinamice. Se aplică aceleași reguli ca mai sus, exceptând faptul că prefixul numelui proprietății trebuie să fie **CS** (în loc de **IS**). Proprietățile `ConnectionSpec` nu sunt cu sens dublu. Același obiect de date de **properties** poate conține atât proprietăți **IS**, cât și **CS**.

Pentru a utiliza proprietăți `ConnectionSpec`, setați **resAuth** specificat pe legarea de import la **Application**. De asemenea, asigurați-vă că adaptorul de resurse suportă autorizarea componentelor. Vedeți capitolul 8 din J2EE Connector Architecture Specification pentru detalii suplimentare.

Clienți externi cu legări EIS:

Serverul poate trimite mesaje către, sau primi mesaje de la, clienți 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 trebuie să fie invocat de o componentă din cadrul serverului.

Clientul invocă importul EIS la fel ca și cu orice altă aplicație, folosind fie DII (Dynamic Invocation Interface) fie interfața Java.

1. Clientul extern creează o instanță a `ServiceManager` și caută importul EIS folosindu-i numele de referință. Rezultatul căutării este o implementare a interfeței 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 interfeței `Service Data Object DataFactory`.
3. Clientul extern invocă EIS-ul și obține rezultatele necesare.

Alternativ, clientul poate invoca importul EIS folosind interfața Java.

1. Clientul creează o instanță a `ServiceManager` și caută importul EIS folosindu-i numele de referință. Rezultatul căutării este o interfață Java a importului EIS.
2. Clientul creează un argument de intrare și un obiect de date tastat.
3. Clientul invocă EIS și obține rezultatele necesare.

Interfața exportului EIS definește interfața componentei SCA exportate ce este disponibilă aplicațiilor EIS externe. Vă puteți gândi la această interfață ca la interfața pe care o va invoca o aplicație externă (precum SAP sau PeopleSoft) prin implementarea runtime-ului aplicației exportului EIS.

Exportul folosește `EISExportBinding` pentru a lega servicii exportate de aplicația EIS externă. Vă permite să abonați o aplicație conținută în modulul dumneavoastră SCA să asculte cereri de servicii EIS. Legarea de export EIS specifică maparea dintre definiția evenimentelor de intrare așa cum este înțeleasă de adaptorul de resurse (folosind interfețe Java EE Connector Architecture) și invocarea operațiilor 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 și pe modelul de programare. Acesta asigură fabrici de conexiune JMS pentru a crea conexiuni pentru destinațiile JMS și pentru a trimite și primi mesaje.

Legările JMS pot fi folosite atunci când interacționați cu legarea furnizorului SIB (Service Integration Bus) și sunt conforme cu JMS și JCA 1.5.

Puteți utiliza legările de export și import JMS ale unui modul SCA (Service Component Architecture) pentru a apela și primi mesaje de la sisteme JMS externe.

Legările de import și export JMS asigură integrarea în aplicațiile JMS folosind furnizorul SIB JMS bazat pe JCA 1.5 care face parte din WebSphere Application Server. Alte adaptoare de resursă JMS bazate pe JCA 1.5 nu sunt suportate.

Privire generală asupra legărilor JMS:

Legările JMS asigură conectivitatea între mediul SCA (Service Component Architecture) și sistemele JMS.

Legări JMS

Componentele majore ale ambelor legări JMS pentru import și export sunt:

- Adaptor resursă: activează conectivitatea bidirecțională, gestionată între un modul SCA și sistemele JMS externe
- Conexiuni: încapsulează o conexiune virtuală între un client și o aplicație furnizor
- Destinații: sunt folosite de un client pentru a specifica ținta mesajelor pe care le produce sau sursa mesajelor pe care le primește
- Date de autentificare: sunt folosite pentru a securiza accesul la legare

Caracteristicile cheie ale legărilor JMS

Anteturi speciale

Proprietățile anteturilor speciale sunt utilizate în importurile și exporturile JMS pentru a-i spune destinației cum să trateze mesajul.

De exemplu, `TargetFunctionName` mapează de la metoda nativă la metoda de funcționare.

Resurse Java EE

Un număr de resurse Java EE este creat la implementarea importurilor și exporturilor JMS într-un mediu Java EE.

ConnectionFactory

Utilizată de clienți pentru a crea o conexiune la furnizorul JMS.

ActivationSpec

Importurile utilizează acest lucru pentru a recepționa răspunsul la o cerere; exporturile o utilizează la configurarea punctelor finale ale mesajelor ce reprezintă ascultători de mesaje în interacțiunea acestora cu sistemul de mesagerie.

Destinații

- Destinație de trimitere: pe un import, aceasta este unde cererea sau mesajul de ieșire este trimis; pe un export, aceasta este destinația unde mesajul de răspuns va fi trimis, dacă nu este depreciată de câmpul antetului `JMSReplyTo` din mesajul de intrare.
- Destinație de recepționare: unde mesajul de intrare va fi plasat; cu importuri, acesta este un răspuns; cu exporturi, este o cerere.
- Destinație de callback: destinația sistemului SCA JMS utilizată pentru a stoca informații de corelare. Nu citați sau scrieți la această destinație.

Operația de instalare creează ConnectionFactory și trei destinații. De asemenea creează ActivationSpec pentru a permite ascultătorului de mesaje runtime să asculte pentru replici pe destinația de recepționare. Proprietățile ale acestor resurse sunt specificate în fișierul de import sau de export.

Integrare JMS și adaptoare 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).

Utilizați un furnizor JMS pentru adaptorul de resurse JCA 1.5 atunci când doriți integrarea într-un sistem extern JMS conform cu JCA 1.5. Serviciile externe conforme cu JCA 1.5 pot primi și 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 și export JMS:

Puteți face ca modulele SCA să interacționeze cu serviciile furnizate de aplicațiile JMS externe utilizând legări de import și export JMS.

Legări de import JMS

Conexiunile către furnizorul JMS asociat destinațiilor JMS sunt create folosind o fabrică de conexiuni JMS. Utilizați obiecte administrative pentru fabrica de conexiuni pentru a gestiona fabricile de conexiuni JMS pentru furnizorul implicit de mesaje.

Interacțiunea cu sistemele JMS externe include utilizarea destinațiilor pentru trimiterea cererilor sau primirea răspunsurilor.

Se oferă suport pentru două tipuri de scenarii de utilizare pentru legările de import JMS, în funcție de tipul operației invocate:

- Primul tip: Importul JMS pune un mesaj în destinația 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 destinația de trimitere, iar apoi persistă răspunsul pe care îl primește de la componenta SCA.

Legarea de import poate fi configurată (folosind câmpul **chema de corelare a răspunsului** în Integration Designer), astfel încât să aștepte 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 utilizeze o destinație de răspuns dinamică temporară pentru a corela răspunsurile cu cererile. Pentru fiecare cerere se creează o destinație temporară, iar importul folosește această destinație pentru a primi răspunsul.

Destinația de receive este setată în proprietatea antetului replyTo din mesajul de ieșire. Un ascultător de mesaje este implementat pentru a asculta la destinația de recepționare, iar în momentul în care este primit un răspuns, acesta transmite răspunsul înapoi către componentă.

Pentru ambele scenarii de utilizare pot fi specificate atât proprietățile dinamice, cât și cele statice ale antetului. Proprietățile statice pot fi setate din legarea metodei de import JMS. Unele dintre aceste proprietăți au semnificații speciale pentru runtime-ul JMS SCA.

Este important de menționat că JMS este o legare asincronă. Dacă o componentă apelantă invocă un import JMS în mod sincron (pentru o operație bidirecțională), atunci aceasta este blocată până când răspunsul este returnat de serviciul JMS.

Figura 32 la pagina 112 ilustrează modul în care importul este legat de serviciul extern.

Import JMS

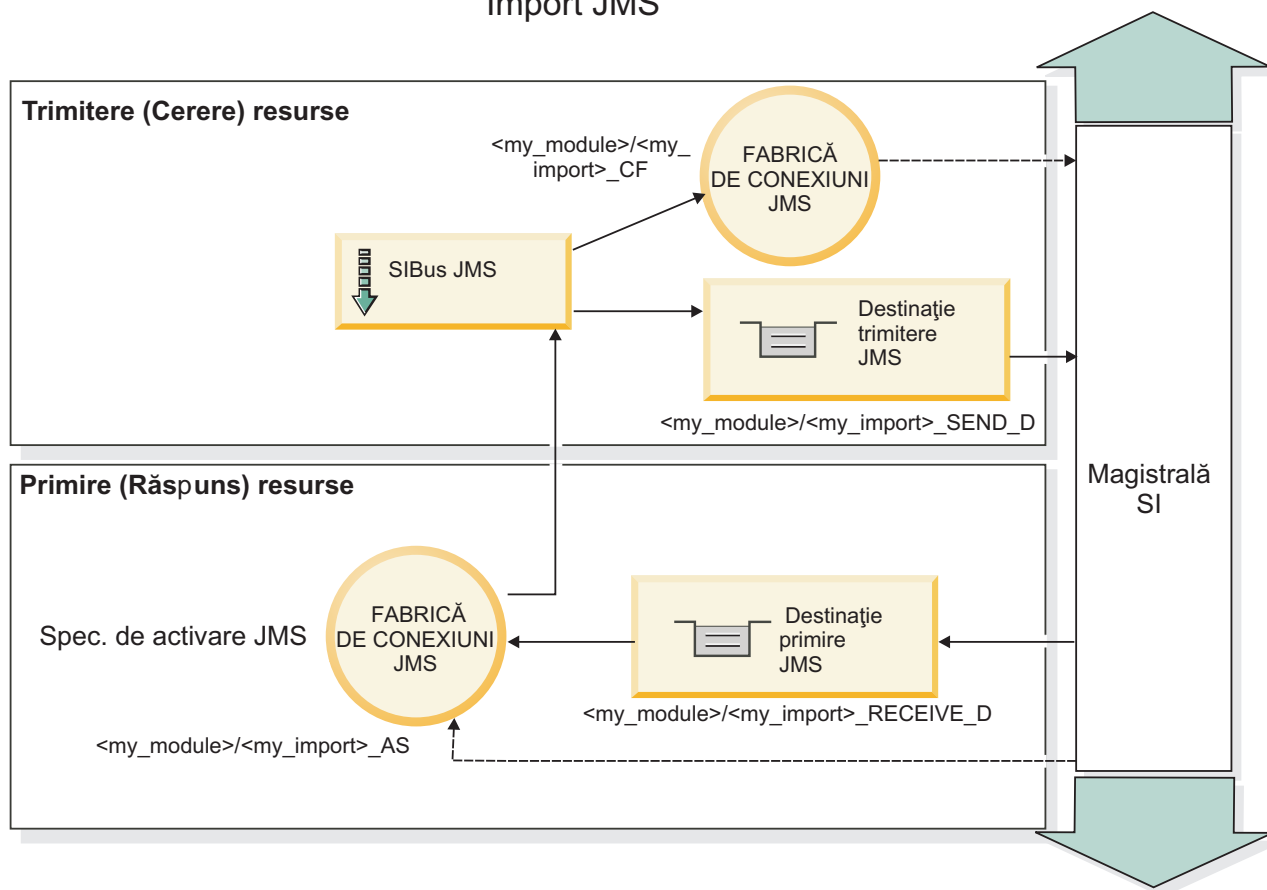


Figura 77. Resurse pentru legarea de import JMS

Legări de export JMS

Legări de export JMS pun la dispoziție mijloacele prin care modulele SCA asigură servicii către aplicațiile JMS externe.

Conexiunea care face parte dintr-un export JMS este o specificație de activare configurabilă.

Un export JMS a trimis și a primit destinațiile.

- Destinația receive este acolo unde ar trebui să fie pus mesajul de intrare pentru componenta țintă.
- Destinația sent este acolo unde va fi trimis răspunsul, cu excepția 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 destinația receive specificată în legarea de export. Destinația specificată în câmpul send este utilizat pentru a trimite răspunsul către cererea de intrare în cazul în care componenta invocată furnizează un răspuns. Destinația specificată în câmpul replyTo din mesajul de intrare înlocuiește destinația specificată în send.

Figura 33 la pagina 113 ilustrează modul în care solicitantul extern este legat de export.

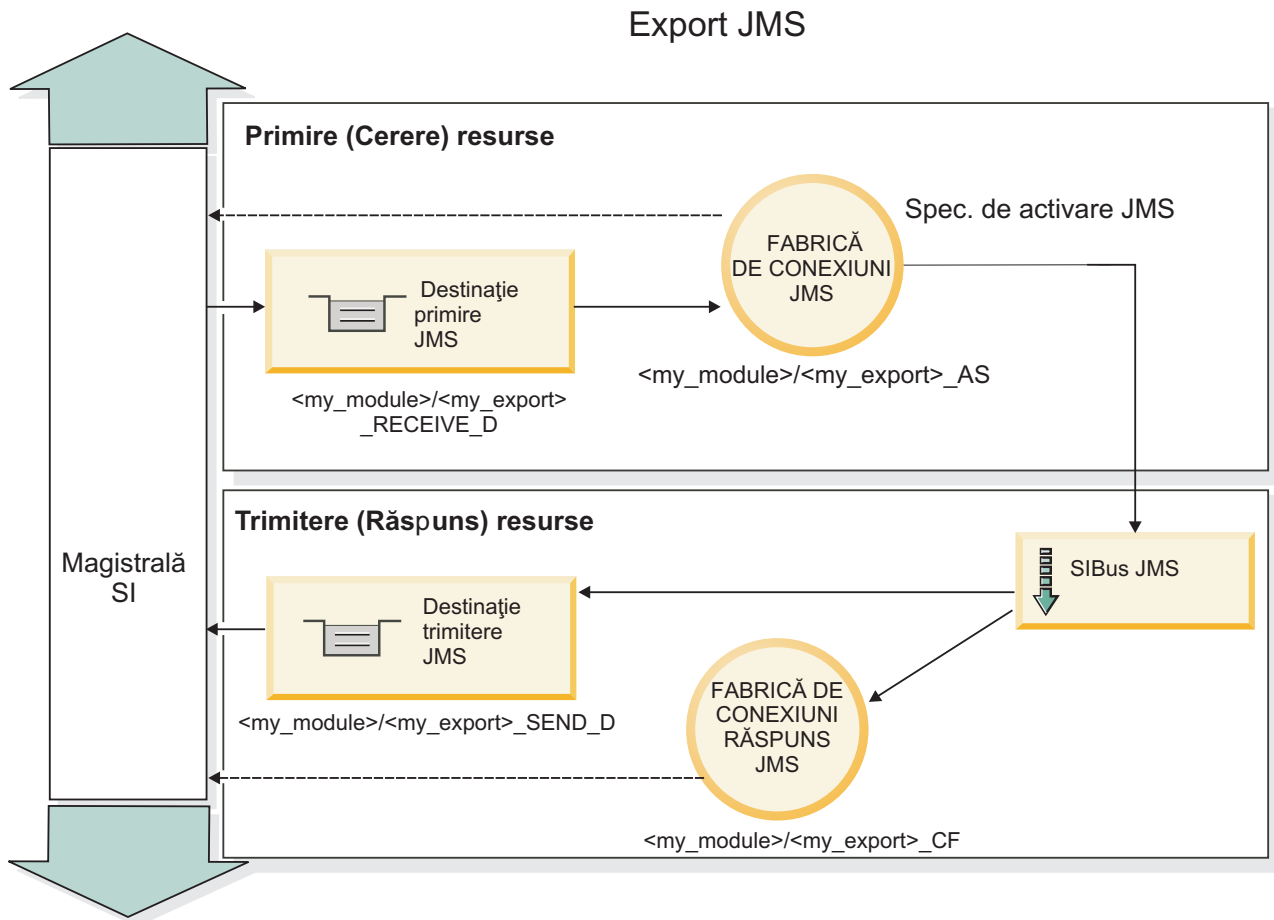


Figura 78. Resurse pentru legarea de export JMS

Anteturi JMS:

Un mesaj JMS conține două tipuri de anteturi - antet sistem JMS și proprietăți JMS multiple. 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 conține toate câmpurile care se găsesc de obicei într-un antet JMS. Deși 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 ieșire 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** și **JMSCorrelationID** - valori ale proprietăților antetului de mesaj predefinit
- **JMSDeliveryMode** - valori pentru modul de livrare (persistent sau nepersistent; implicit este persistent)
- **JMSPriority** - valoare de prioritate (de la 0 la 9; implicit este JMS_Default_Priority)

Proprietăți JMS

Proprietățile JMS sunt reprezentate în SMO sub formă de intrări în lista Proprietăți. Proprietățile pot fi adăugate, actualizate sau șterse într-o mediere sau folosind API-ul ContextService.

De asemenea, proprietățile pot fi setate în legarea JMS. Proprietățile care sunt setate în mod static înlocuiesc setările (cu același nume) care sunt setate în mod dinamic.

Proprietățile utilizatorului propagate din alte legări (de exemplu, o legare HTTP) va fi pusă în legarea JMS sub formă de proprietăți JMS.

Setările propagării anteturilor

Propagarea antetului și proprietăților sistemului JMS fie de la mesajul JMS de intrare către componentele următoare, fie de la componentele anterioare către mesajul JMS de ieșire poate fi controlată prin stegulețul Propagate Protocol Header din legare.

Atunci când Propagate Protocol Header este setat, informațiilor de antet le este permis să circule către mesaj sau către componenta țintă, așa cum este descris în următoarea listă:

- Cerere de export JMS
Antetul JMS primit în mesaj va fi propagat către o componentă țintă prin intermediul serviciului de context. Proprietățile JMS primite în mesaj vor fi propagate către o componentă țintă prin intermediul serviciului de context.
- Răspuns la exportul JMS
Oricare dintre câmpurile din antetul JMS din serviciul de context va fi utilizat în mesajul de ieșire, dacă nu este înlocuită de proprietățile statice stabilite în legarea de export JMS. Oricare dintre proprietățile setate în serviciul de context va fi utilizată în mesajul de ieșire, dacă nu este înlocuită de proprietățile 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 ieșire, dacă nu este înlocuit de proprietățile statice stabilite în legarea de import JMS. Oricare dintre proprietățile setate în serviciul de context va fi utilizată în mesajul de ieșire, dacă nu este înlocuită de proprietățile statice stabilite în legarea de import JMS.
- Răspuns import JMS
Antetul JMS primit în mesaj va fi propagat către o componentă țintă prin intermediul serviciului de context. Proprietățile JMS primite în mesaj vor fi propagate către o componentă țintă prin intermediul serviciului de context.

Schema de corelare a destinațiilor de răspuns dinamice temporare JMS:

Schema de corelare a destinațiilor de răspuns dinamice temporare determină o coadă dinamică unică sau un subiect care urmează să fie creat pentru fiecare cerere trimisă.

Destinația statică de răspuns menționată în import este utilizată pentru a obține natura cozii dinamice temporare de destinație sau subiectul. Acest lucru este setat în câmpul **ReplyTo** al cererii, iar importul JMS ascultă pentru răspunsuri pe această destinație. În cazul în care răspunsul este primit este cerut de destinația statică a răspunsului pentru prelucrarea asincronă. Câmpul **CorrelationID** din răspuns nu este utilizat și nu este nevoie să fie setat.

Probleme legate de Tranzacții

Atunci când este folosită o destinație dinamică temporară, cererea trebuie să fie prelucrat în același fir de execuție ca răspunsul trimis. Cererea trebuie să fie expediată în afara tranzacției globale, și trebuie să fie comisă înainte să fie primită de serviciul de backend, și ca un răspuns să fie returnat.

Persistență

Cozile temporare dinamice sunt entități de scurtă durată și nu garantează același nivel de persistență asociat cu o coadă sau subiect static. O coadă dinamică temporară sau subiect nu va mai exista după repornirea serverului și nici mesajele. După ce mesajul a fost cerut la destinația statică de răspuns, acesta își păstrează persistența definită în mesaj.

Timeout

Importul așteaptă să primească răspunsul la destinația 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 așteptare este depășit, importul aruncă o excepție de tip `ServiceTimeoutRuntimeException`.

Clienți externi:

Serverul poate trimite mesaje către, sau primi mesaje de la, clienți externi folosind legări JMS.

Un client extern (de exemplu, un portal web sau un sistem de informații 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 destinația de primire specificată în legarea de export. Destinația specificată în câmpul de trimitere este folosită pentru a trimite răspunsul către cererea de intrare dacă aplicația invocată furnizează un răspuns. Astfel, un client extern este capabil să invoce aplicații cu legarea de export.

Importurile JMS interacționează cu clienți externi prin trimiterea de mesaje și primirea de mesaje de la cozi JMS.

Lucrul cu clienți externi:

Un client extern (ce este în afara serverului) ar putea avea nevoie să interacționeze cu o aplicație instalată pe server.

Considerați un scenariu foarte simplu în care un client extern vrea să interacționeze cu o aplicație de pe server. Figura descrie un scenariu tipic simplu.

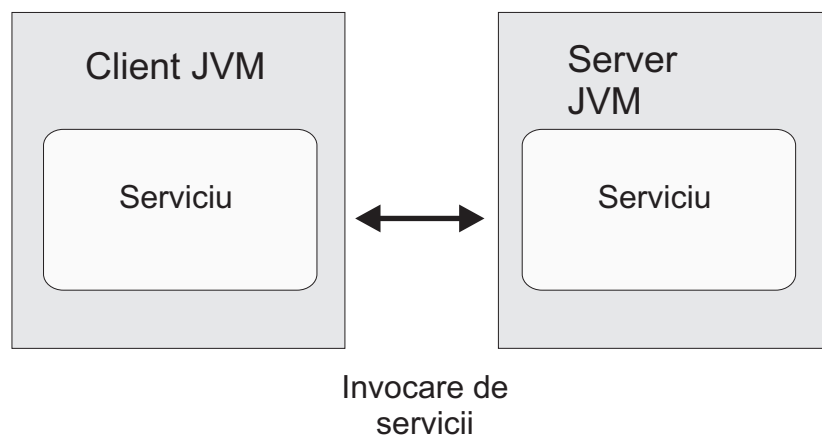


Figura 79. Scenariu simplu utilizare-caz: client extern interacționează cu aplicația serverului

Aplicația SCA include un export cu o legare JMS; aceasta face aplicația disponibilă pentru clienți externi.

Atunci când aveți un client extern în mașina virtuală Java (JVM) separat de serverul dumneavoastră, există mai mult pași pe care trebuie să-i realizați pentru a realiza o conexiune și pentru a interacționa cu un export JMS. Clientul obține un `InitialContext` cu valorile corecte și apoi caută resursele prin JNDI. Apoi clientul utilizează clientul specificație JMS 1.1 pentru a accesa destinațiile și mesajele de trimitere și primire a destinațiilor.

Numele implicite JNDI ale resurselor create automat prin runtime sunt listate în subiectul de configurare al acestei secțiuni. Totuși, dacă aveți resurse create anterior, utilizați acele nume JNDI.

1. Configurați destinațiile JMS și fabrica de conexiuni pentru a trimite mesajul.

2. Asigurați-vă de faptul că contextul JNDI, portul pentru adaptorul de resurse SIB și portul de mesagerie bootstrapping 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 instanțe de server. Puteți utiliza consola administrativă pentru a determina ce porturi ale serverului dumneavoastră sunt implementate. Mergeți la **Servere > servere de aplicații > nume_server > Configurae** și apăsați pe **Porturi** din **Comunicație**. Puteți edita portul ce este utilizat.
3. Clientul obține un context inițial cu valorile corecte și apoi caută resurse prin JNDI.
4. Utilizând specificațiile JMS 1.1, clientul accesează destinațiile și mesajele de trimitere și recepționare de pe destinații.

Depanare legări JMS:

Puteți diagnostica și repara problemele cu legările JMS.

Excepții de implementare

Ca răspuns la condiții de eroare, implementarea JMS import și export poate returna una din cele două tipuri de excepții:

- Service Business Exception: această excepție este returnată dacă a survenit eroarea specificată pe interfața serviciului business (tip port WSDL).
- Service Runtime Exception: apărută în toate celelalte cazuri. În cele mai multe cazuri, excepția cauze va conține excepția originală (JMSException).

De exemplu, un import așteaptă numai un mesaj de răspuns pentru fiecare mesaj de cerere. Dacă mai mult de un răspuns sosește, sau dacă un răspuns întârziat (unul pentru care expirarea răspunsului SCA are loc) sosește, o excepție Service Runtime Exception este aruncată. Tranzacția este repetată, și mesajul de răspuns este retras din coadă sau este manevrat de către managerul de eveniment.

Condiții de eșec primar

Condițiile de eșec primar ale legărilor JMS sunt determinate de către sensurile tranzacționale, de către configurația JMS provider, sau prin referire la comportamente existente în alte componente. Condițiile de eșec primar includ:

- Eșec la conexiunea la furnizorul sau destinația JMS.
Un eșec în conexiunea la furnizorul JMS pentru a primi mesaje va avea ca rezultat începutul căderii ascultătorului de mesaje. Această condiție va fi jurnalizată în istoricul WebSphere Application Server. Mesajele persistente vor rămâne la destinație până când vor fi retrase cu succes (sau expirate).
Un eșec la conectarea la furnizorul JMS pentru a trimite mesaje în exterior va cauza derularea înapoi a tranzacției ce controlează trimiterea.
- Eșecul în analiza unui mesaj de intrare sau în construirea unui mesaj de ieșire.
Un eșec în legarea de date sau în handler-ul de date cauzează derularea înapoi a tranzacției care controlează lucrul.
- Eșec la trimiterea mesajului de ieșire.
Un eșec la trimiterea unui mesaj cauzează derularea înapoi a tranzacției relevante.
- Mutiple sau neașteptate mesaje de răspuns întârziate.
Importul așteaptă 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 calificativu SCA Response Expiration qualifier la cerere. Când un răspuns sosește sau timpul de expirare este depășit, înregistrarea de corelare este ștearsă. Dacă mesajele de răspuns ajung pe neașteptat sau sosesc târziu, o excepție Service Runtime Exception este aruncată.
- Excepție runtime timeout serviciu cauzată de către răspunsul întârziat când se folosește schema de corelare a destinației de răspuns dinamic temporară.

JMS import va fi timeout după o perioadă de timp determinată de către calificativul SCA response expiration, 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 își au originea într-o cădere a JMS, ar trebui să vă așteptați să găsiți aceste mesaje în managerul de evenimente eşuate. Dacă astfel de mesaje nu apar în managerul de evenimente eşuate, asigurați-vă că destinația de bază SIB a destinației JMS are o valoare maximă de livrări nereușite mai mare decât 1. Setarea acestei valori la 2 sau mai mult activează interacțiunea cu managerul de evenimente eşuate în timpul invocărilor SCA pentru legările JMS.

Tratarea excepțiilor:

Modul în care este configurată legarea determină cum excepțiile 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 excepție.

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 excepție este determinat de cum implementați handler-ul de date sau legarea de date. Comportamentul recomandat este să vă proiectați legarea de date să arunce o **DataBindingException**.

Când orice excepție runtime, inclusiv o **DataBindingException**, este aruncată:

- Dacă fluxul de mediere este configurat să fie tranzacțional, mesajul JMS, implicit, este memorat în Managerul de evenimente eşuate pentru reluare sau ștergere manuală.

Notă: Puteți 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 tranzacțional, excepția este înregistrată în istoric și mesajul este pierdut.

Situația este similiară pentru un handler de date. Din moment ce handler-ul de date este invocat de legarea de date, orice excepție a handler-ului de date este înfășurată într-o excepție 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 terță parte care sunt conformi cu JMS 1.1. Operarea legărilor Generic JMS este similară cu cea a legărilor JMS.

Serviciul pus la dispoziție 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 și care suportă JMS 1.1 și implementează facilitatea opțională JMS Application Server. Legarea JMS Generic suportă acei furnizori JMS (inclusiv Oracle AQ, TIBCO, SonicMQ, WebMethods și BEA WebLogic) care nu suportă JCA 1.5, dar suportă Application Server Facility a specificației 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, utilizați “Legări JMS” la pagina 110.

Utilizați această legare generică la integrarea unui sistem JMS conform cu care nu este de tip JCA 1.5 în cadrul unui mediu SCA. Aplicațiile externe țintă pot primi și trimite apoi mesaje pentru a se integra într-o componentă SCA.

Privire generală asupra legărilor Generic JMS:

Legările Generic JMS sunt legări ne-JCA JMS care asigură conectivitate între SCA mediul (Service Component Architecture) și sistemele JMS care sunt în conformitate cu JMS 1.1 și care implementează facilitatea opțională a serverului de aplicații JMS.

Legări Generic JMS

Aspectele majore ale legărilor generice pentru import și export JMS includ următoarele:

- Port ascultător: permite furnizorilor JMS ce nu se bazează pe JCA să primească mesaje și să le trimită către un MDB (Message Driven Bean)
- Conexiuni: încapsulează o conexiune virtuală între un client și o aplicație furnizor
- Destinații: sunt folosite de un client pentru a specifica ținta mesajelor pe care le produce sau sursa mesajelor pe care le primește
- Date de autentificare: sunt folosite pentru a securiza accesul la legare

Legări de import Generic JMS

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 definiți de către un administrator. Fiecare fabrică de conexiuni este o instanță a uneia dintre interfețele `ConnectionFactory`, `QueueConnectionFactory` sau `TopicConnectionFactory`.

Interacțiunea cu sistemele JMS externe include utilizarea destinațiilor pentru trimiterea cererilor sau primirea răspunsurilor.

Se oferă suport pentru două tipuri de scenarii de utilizare pentru legările generice de import JMS, în funcție de tipul operației invocate:

- Primul tip: Importul generic JMS pune un mesaj în destinația de trimitere configurată în legarea de import. Nu se trimite nimic către câmpul `replyTo` din antetul JMS.
- Al doilea tip (cerere-răspuns): Importul Generic JMS pune un mesaj în destinația de trimitere, iar apoi persistă răspunsul pe care îl primește de la componenta SCA.

Destinația de primire este setată în proprietatea antetului `replyTo` din mesajul de ieșire. Un MDB (message driven bean) este implementat pentru a asculta la destinația de recepționare, 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 aștepte 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 proprietățile dinamice, cât și cele statice ale antetului. Proprietățile statice pot fi setate din legarea metodei de import generic JMS. Unele dintre aceste proprietăți au semnificații speciale pentru runtime-ul JMS SCA.

Este important de menționat faptul că JMS este o legare asincronă. Dacă o componentă apelantă invocă un import Generic JMS în mod sincron (pentru o operație bidirecțională), atunci aceasta este blocată până când răspunsul este returnat de serviciul JMS.

Figura 35 la pagina 119 ilustrează modul în care importul este legat de serviciul extern.

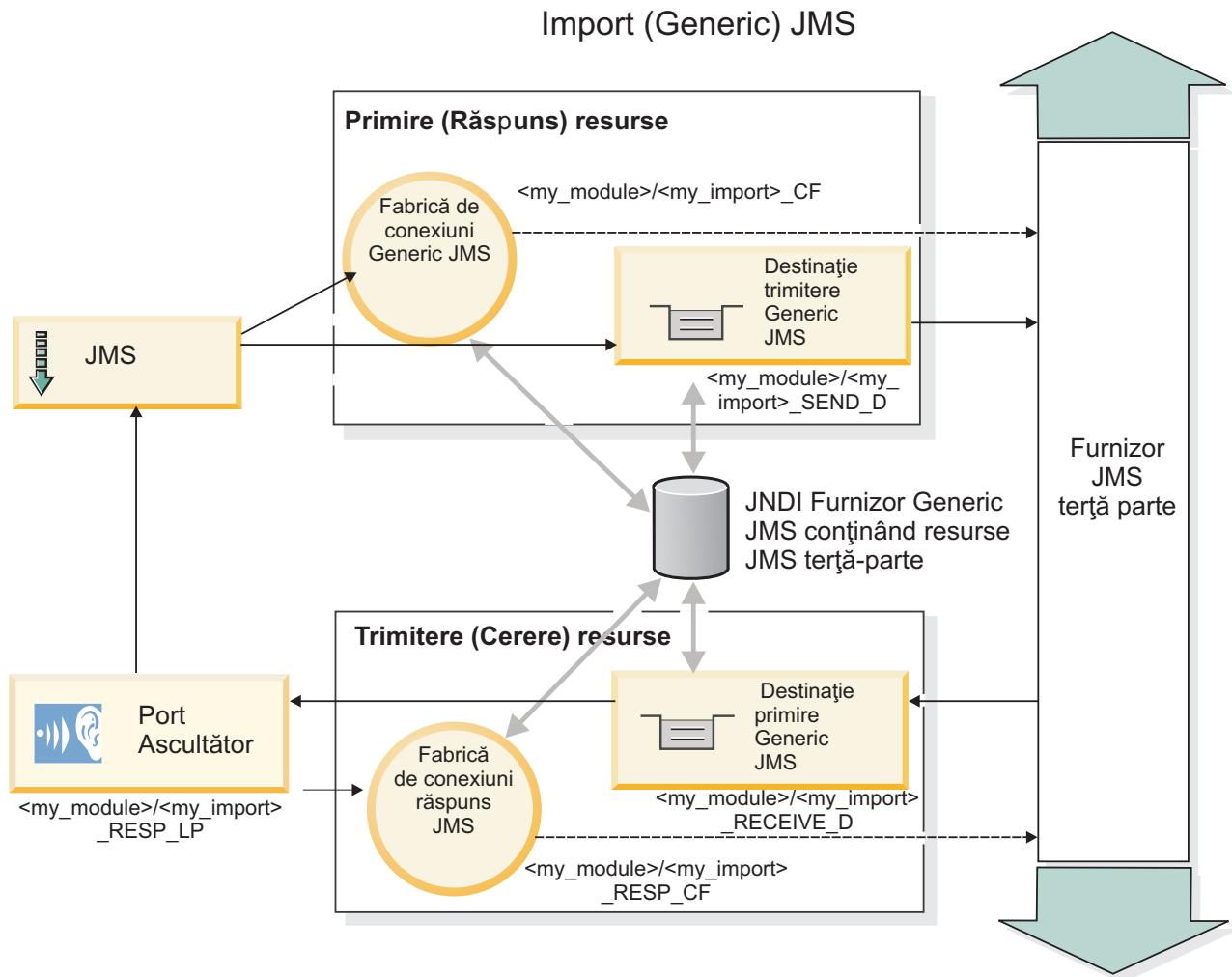


Figura 80. Resurse pentru legarea generică de import JMS

Legări de export Generic JMS

Legările de export Generic JMS pun la dispoziție mijloacele prin care modulele SCA asigură servicii către aplicațiile JMS externe.

Partea de conexiuni a unui export JMS este compus dintr-un ConnectionFactory și un ListenerPort.

Un export generic JMS a trimis și a primit destinațiile.

- Destinația primire este acolo unde ar trebui să fie pus mesajul de intrare pentru componenta țintă.
- Destinația trimitere este acolo unde va fi trimis răspunsul, cu excepția cazului în mesajul de intrare a înlocuit-o folosind proprietatea antetului replyTo.

Este implementat un MDB pentru a asculta cererile ce intră prin destinația primire specificată în legarea de export.

- Destinația specificată în câmpul trimitere este folosită pentru a trimite răspunsul către cererea de intrare în cazul în care aplicația invocată oferă un răspuns.
- Destinația specificată în câmpul replyTo din mesajul de intrare înlocuiește destinația 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 aștepte 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 120 ilustrează modul în care solicitantul extern este legat de export.

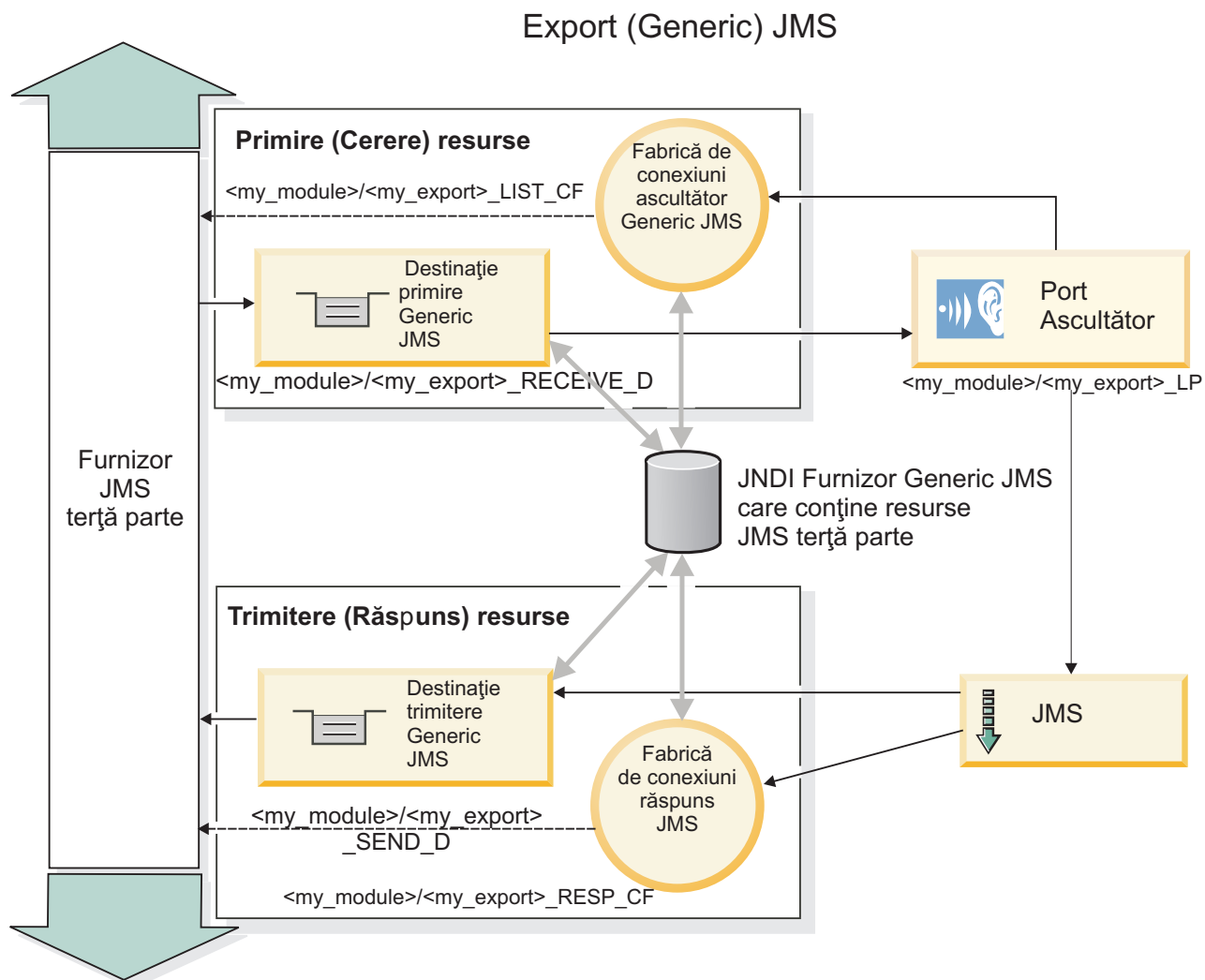


Figura 81. Resurse pentru legarea generică pentru exportul JMS

Caracteristici cheie ale legărilor JMS generic:

Caracteristicile legărilor de import și export JMS generic sunt consistente cu acelea ale legărilor de import MQ JMS și cele JMS imbricate WebSphere. Caracteristicile cheie includ definiții de antet și acces la resurse Java EE existente. Totuși, din cauza naturii sale generice, nu există nicio opțiune de conectivitate specifică furnizorilor JMS, iar această legare are capacitate limitată pentru a genera resurse la implementare și instalare.

Importuri generice

La fel ca aplicația de import MQ JMS, implementarea Generic JMS este asincronă și suportă trei invocări: o cale, două căi (de asemenea cunoscută ca cerere-răspuns), și 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) destinația trimisă cu cerere în câmpul de antetreplyTo al mesajului JMS.

Exporturi generice

Legările de exporturi Generic JMS diferă de legările de export EIS în cazul tratării returnării rezultatului. Un export Generic JMS trimite explicit răspunsul la destinația replyTo specificată în mesajul de intrare. Dacă niciuna nu este specificată, destinația 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 destinația de recepționare și apoi dispeceerizează cererile pentru a fi procesate de runtime-ul SCA.

Anteturi speciale

Proprietățile anteturilor speciale sunt utilizate în importurile și exporturile Generic JMS pentru a informa legarea destinație cum să fie tratat mesajul.

De exemplu, proprietatea TargetFunctionName este utilizată de către selectorul de funcții implicit pentru a identifica numele operației din interfața export ce este invocată.

Notă: Legarea de import poate fi configurată pentru a seta antetul TargetFunctionName la numele de operație al fiecărei operații.

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 destinația de recepționare (răspuns) (doar două căi) pentru importuri și pe destinația de recepționare (cerere) pentru exporturi
- Fabrica de conexiuni Generic JMS pentru outboundConnection (import) și inboundConnection (export)
- Destinația Generic JMS pentru destinațiile de trimitere (import) și recepționare (export; doar două căi)
- Fabrica de conexiuni Generic JMS pentru responseConnection (doar două căi și opțional; altfel, outboundConnection este utilizat pentru importuri, iar inboundConnection este utilizat pentru exporturi)
- Destinația Generic JMS pentru destinațiile de recepționare (import) și trimitere (export) (doar două căi)
- Destinația JMS de callback a furnizorului de mesagerie implicit utilizată pentru a accesa destinația coadă de callback SIB (doar două căi)
- Fabrica de conexiuni JMS de callback a furnizorului de mesagerie implicit utilizată pentru a accesa destinația JMS de callback (doar două căi)
- Destinația coadă de callback SIB utilizată pentru a memora informațiile despre mesajul de cerere pentru utilizarea în timpul procesării răspunsului (doar două căi)

Taskul de instalare creeazăConnectionFactory, cele trei destinații, și ActivationSpec din informațiile din fișierele de import și export.

Anteturi Generic JMS:

Anteturile Generic JMS sunt SDO-uri (Service Data Objects) ce conțin toate proprietățile proprietăților mesajului JMS Generic. Aceste proprietăți pot fi de la mesajul de intrare sau pot fi proprietățile ce vor fi aplicate mesajului de ieșire.

Un mesaj JMS conține două tipuri de anteturi - antet sistem JMS și proprietăți JMS multiple. Ambele tipuri de anteturi pot fi accesate fie într-un modul de mediere din SMO (Service Message Object), fie folosind API-ul ContextService.

Următoarele proprietăți sunt setate static pe methodBinding:

- JMSType
- JMSCorrelationID
- JMSDeliveryMode
- JMSPriority

Legarea JMS Generic suportă modificarea dinamică a anteturilor și proprietăților JMS în același mod ca legările JMS și MQ JMS.

Unii furnizori de JMS Generic pun restricții pe ce proprietăți pot fi setate de aplicație și în ce combinații. Trebuie să vă consultați documentația produsului de terță parte pentru informații suplimentare. Totuși, o proprietate suplimentară a fost adăugată lui `methodBinding`, `ignoreInvalidOutboundJMSProperties`, ce permite să fie propagate orice excepții.

Anteturile și proprietățile mesajelor JMS Generic sunt folosite doar când comutatorul de bază al legării SCDL a arhitecturii componente de servicii este pornit. Când comutatorul este pornit, informațiile de context sunt propagate. Implicit, comutatorul este pornit. Pentru a preveni propagarea informațiilor de context, modificați valoarea în **false**.

Când propagarea contextelor este activată, informațiile de antet sunt permise să curgă către mesaj sau către componenta țintă. Pentru a porni și opri propagarea contextelor, specificați **true** sau **false** pentru atributul `contextPropagationEnabled` al legărilor de import și export. De exemplu:

```
<esbBinding xsi:type="eis:JMSImportBinding" contextPropagationEnabled="true">
```

Valoarea implicită este **true**.

Depanarea legărilor JMS generic:

Puteți diagnostica și rezolva probleme cu legarea JMS generic.

Excepții de implementare

Ca răspuns la diversele condiții de eroare, implementarea Generic JMS import și export poate returna una din cele două tipuri de excepții:

- **Service Business Exception:** această excepție este returnată dacă fault specificată pe interfața serviciului business (WSDL port type) a survenit.
- **Service Runtime Exception:** apărută în toate celelalte cazuri. În cele mai multe cazuri, excepția cauze va conține excepția (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 și 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 ieșire. Notificarea expirării mesajului de apel invers va indica faptul că mesajul de cerere a expirat și clientul ar trebui notificat printr-o excepție operațională.

Dacă destinația de apel invers este mutată pe un furnizor terță parte, totuși, 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. Totuși, puteți verifica dacă răspunsul este expirat atunci când este primit.

Pentru mesaje de cerere de ieșire, valoarea JMSExpiration va fi calculată în timpul așteptat și din valorile requestExpiration efectuate în `asyncHeader`, dacă este setat.

Depanarea erorilor fabricii de conexiuni Generic JMS

Atunci când definiția anumite tipuri de fabrici de conexiuni în furnizorul dumneavoastră Generic JMS, s-ar putea să primiți un mesaj de eroare atunci când încercați să porniți aplicația. Puteți modifica fabrica de conexiuni externă pentru a evita această problemă.

Atunci când lansați o aplicație, este posibil să primiți următorul mesaj de eroare:

Tipul MDB Listener Port JMSConnectionFactory nu corespunde tipului JMSDestination.

Această problemă poate apărea atunci când definiți fabrici de conexiuni externe. Specific, excepția poate fi aruncată atunci când creați 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 și cea de publicare/anonare).

Pentru a rezolva această problemă, realizați următorii pași:

1. Accesați furnizorul Generic JMS pe care îl utilizați.
2. Înlocuiți fabrica de conexiuni subiect JMS 1.0.2 pe care ați definit-o cu fabrica de conexiuni (unificate) JMS 1.1.

Când lansați aplicația cu fabrica de conexiuni JMS 1.1 definită mai nou, ar trebui să nu mai primiți un mesaj de eroare.

Mesajele SCA bazate pe Generic JMS nu apar în managerul de evenimente eșuate

Dacă mesajele SCA au originea într-o eșuare a interacțiunii JMS, v-ați aștepta să găsiți aceste mesaje în managerul de evenimente eșuate. Dacă asemenea mesaje nu apar în managerul de evenimente eșuate, asigurați-vă că valoarea proprietății 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 interacțiunea cu managerul de evenimente eșuate în timpul invocării SCA pentru legările Generic JMS.

Tratarea excepțiilor:

Modul în care este configurată legarea determină cum sunt tratate excepțiile 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 excepție.

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 excepție este determinat de cum implementați handler-ul de date sau legarea de date. Comportamentul recomandat este să vă proiectați legarea de date să arunce o

DataBindingException.

Situația este similiară pentru un handler de date. Din moment ce handler-ul de date este invocat de legarea de date, orice excepție a handler-ului de date este înfășurată într-o excepție a legării de date. Prin urmare o

DataHandlerException vă este raportată ca o **DataBindingException.**

Când orice excepție runtime, inclusiv o excepție **DataBindingException**, este aruncată:

- Dacă fluxul de mediere este configurat să fie tranzacțional, mesajul JMS este memorat în Managerul de evenimente eșuate implicit pentru reluare sau ștergere manuală.

Notă: Puteți modifica modul de recuperare de pe legare, astfel încât mesajul să fie derulat înapoi în loc să fie stocat în managerul de evenimente eșuate.

- Dacă fluxul de mediere nu este tranzacțional, excepția este înregistrată în istoric și mesajul este pierdut.

Situația este similiară pentru un handler de date. Deoarece handler-ul de date este apelat de legarea de date, o excepție a handler-ului de date este produsă înăuntrul unei excepții 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 aplicațiile externe care folosesc un furnizor bazat pe WebSphere MQ JMS.

Utilizați WebSphere MQ JMS legările de export și 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 Service Integration Bus.

Atunci când o componentă interacționează cu un serviciu bazat pe WebSphere MQ JMS prin intermediul unui import, legarea de import WebSphere MQ JMS utilizează o destinație către care datele vor fi trimise și o destinație unde va fi primit răspunsul. Conversia datelor către și 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 clienților WebSphere MQ JMS, legarea de export WebSphere MQ JMS utilizează o destinație la care vor fi recepționate datele și către care poate fi trimis răspunsul. Conversia datelor către și de la un mesaj JMS se realizează prin intermediul handler-ului de date JMS sau prin legarea datelor.

Selectorul funcției asigură o mapare operației în cadrul componentei țintă care va fi invocată.

Privire generală asupra legărilor WebSphere MQ JMS:

Legarea WebSphere MQ JMS asigură integrarea în aplicațiile externe care folosesc furnizorul WebSphere MQ JMS.

Taskuri administrative WebSphere MQ

Se așteaptă 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 aplicație care conține 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 bazați pe WebSphere MQ JMS. Trebuie să utilizați o versiune suportată de WebSphere MQ. Cerințe detaliate despre hardware și 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 funcție de tipul de operație care este invocat:

- Primul tip: Importul WebSphere MQ JMS pune un mesaj în destinația 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 destinația de trimitere. Destinația de primire este setată în câmpul replyTo din antet. Un MDB (message-driven bean) este implementat pentru a asculta la destinația de recepționare, 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 aștepte 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 proprietățile dinamice, cât și cele statice ale antetului. Proprietățile statice pot fi setate din legarea metodei de import JMS. Unele dintre aceste proprietăți au semnificații speciale pentru runtime-ul JMS SCA.

Este important de menționat faptul că WebSphere MQ JMS este o legare asincronă. Dacă o componentă apelantă invocă un import WebSphere MQ JMS în mod sincron (pentru o operație bidirecțională), atunci aceasta este blocată până când răspunsul este returnat de serviciul JMS.

Figura 37 la pagina 125 ilustrează modul în care importul este legat de serviciul extern.

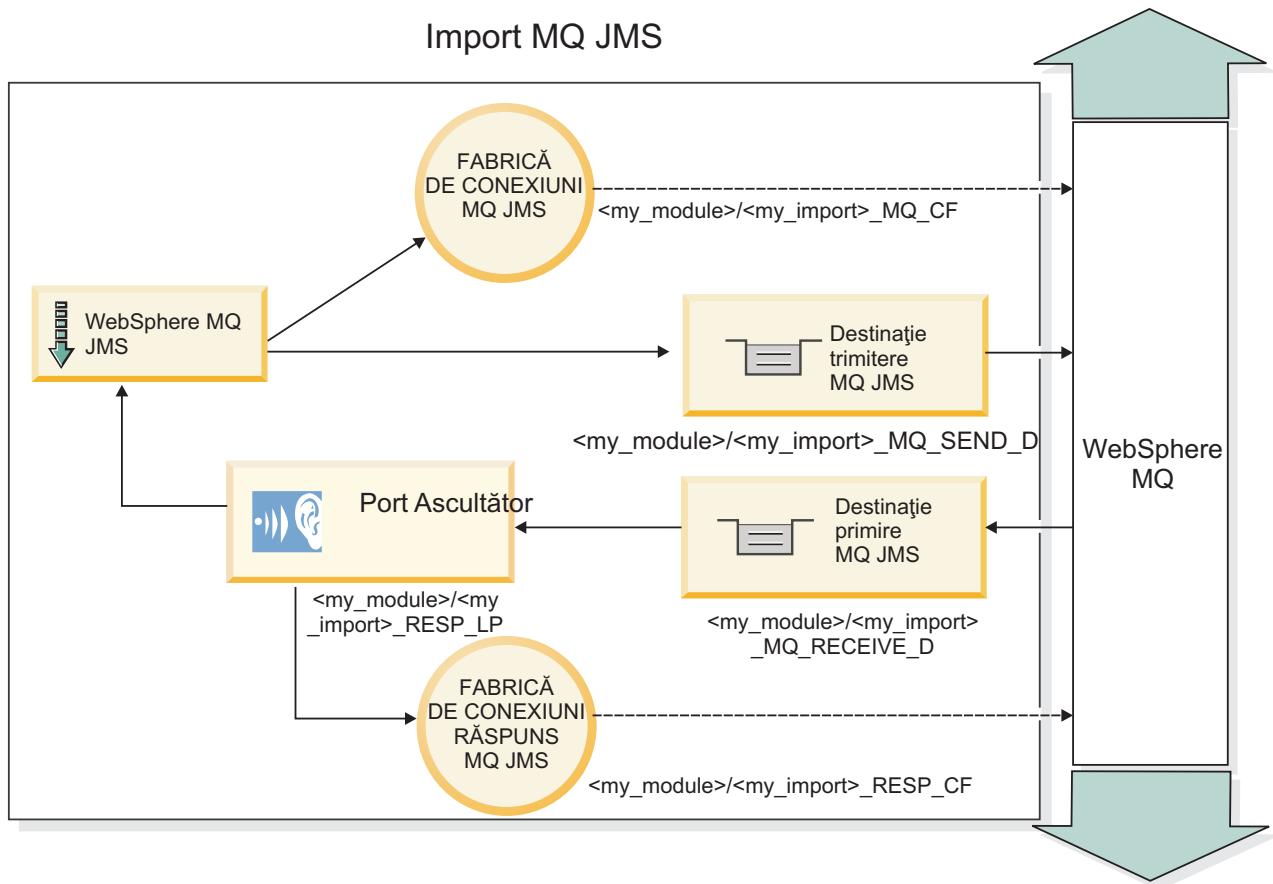


Figura 82. Resurse pentru legarea de import WebSphere MQ JMS

Legări pentru export WebSphere MQ JMS

Legarea de export WebSphere MQ JMS pune la dispoziție mijloacele prin care modulele SCA asigură servicii către aplicațiile externe bazate pe WebSphere MQ.

Este implementat un MDB pentru a asculta cererile ce intră prin destinația primire specificată în legarea de export. Destinația specificată în câmpul trimite este utilizat pentru a trimite răspunsul către cererea de intrare în cazul în care componenta invocată furnizează un răspuns. Destinația specificată în câmpul replyTo din mesajul de răspuns înlocuiește destinația specificată în câmpul trimite.

Figura 38 la pagina 126 ilustrează modul în care solicitantul extern este legat de export.

Export MQ JMS

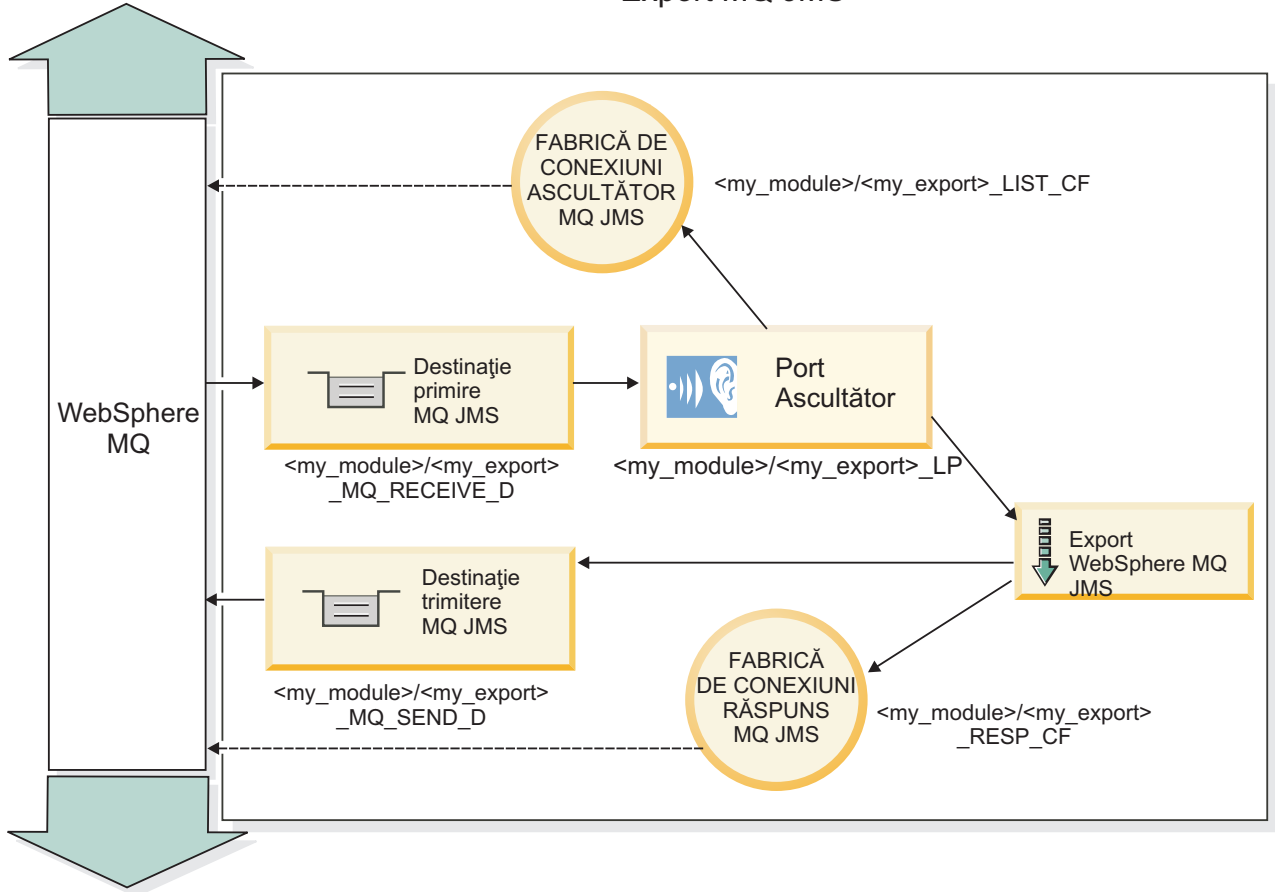


Figura 83. Resurse pentru legarea de export WebSphere MQ JMS

Notă: Figura 37 la pagina 125 și Figura 38 la pagina 126 ilustrează modul în care o aplicație dintr-o versiune anterioară a IBM Business Process Manager este legată de un serviciu extern. Pentru aplicațiile dezvoltate pentru IBM Business Process Manager Versiunea 7.0, se folosește Specificare activare în loc de Port ascultător și Fabrică de conexiuni.

Caracteristici cheie ale legărilor WebSphere MQ JMS:

Caracteristicile cheie ale legărilor WebSphere MQ JMS includ anteturi, artefacte Java EE, și resurse create Java EE.

Anteturile

Un antet de mesaj JMS conține un număr de câmpuri predefinite ce conțin valori utilizate de și de clienți și de furnizori pentru a identifica și a ruta mesajele. Puteți folosi proprietățile de legare pentru a configura aceste anteturi cu valori fixe, sau anteturile pot fi specificate dinamic la rulare.

JMSCorrelationID

Legări la un mesaj înrudit. În mod tipic, acest câmp este setat la șirul identificatorului mesajului căruia îi este dat un răspuns.

TargetFunctionName

Acest antet este utilizat de unul din selectorii de funcții furnizați pentru a identifica operația ce este invocată. Setarea proprietății de antet JMS TargetFunctionName în mesajele trimise la un export JMS permite acestui selector de funcții să fie utilizat. Proprietatea poate fi setată direct în aplicațiile 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 operație din interfață 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 creere 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.

Parametrii

Fabrica de conexiuni MQ

Utilizată de clienți pentru a crea o conexiune la furnizorul MQ JMS.

Fabrica de conexiuni răspuns

Folosit de runtime-ul SCA MQ JMS atunci când destinația de trimitere este pe un Manager coadă diferit de cel al destinației de primire.

Specificații de activare

O specificație de activare MQ JMS este asociată cu una sau mai multe bean-uri bazate pe mesaje și furnizează configurația necesară pentru a recepționa mesajele.

Destinații

- Trimitere destinație:
 - Importuri: Unde sunt trimise mesajele de cerere sau de ieșire.
 - Exporturi: Unde vor fi trimise mesajele de răspuns dacă nu sunt permise de câmpul de antet JMSReplyTo al mesajului de intrare.
- Destinație de recepționare:
 - Importuri: Unde vor fi plasate mesajele de intrare și de răspuns.
 - Exporturi: Unde vor fi plasate mesajele de intrare și de cerere.

Anteturi JMS:

Un mesaj JMS conține două tipuri de anteturi - antet sistem JMS și proprietăți JMS multiple. 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 conține toate câmpurile care se găsesc de obicei într-un antet JMS. Deși 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 ieșire 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** și **JMSCorrelationID** - valori ale proprietăților antetului de mesaj predefinit
- **JMSDeliveryMode** - valori pentru modul de livrare (persistent sau nepersistent; implicit este persistent)
- **JMSPriority** - valoare de prioritate (de la 0 la 9; implicit este JMS_Default_Priority)

Proprietăți JMS

Proprietățile JMS sunt reprezentate în SMO sub formă de intrări în lista Proprietăți. Proprietățile pot fi adăugate, actualizate sau șterse într-o mediere sau folosind API-ul ContextService.

De asemenea, proprietățile pot fi setate în legarea JMS. Proprietățile care sunt setate în mod static înlocuiesc setările (cu același nume) care sunt setate în mod dinamic.

Proprietățile utilizatorului propagate din alte legări (de exemplu, o legare HTTP) va fi pusă în legarea JMS sub formă de proprietăți JMS.

Setările propagării anteturilor

Propagarea antetului și proprietăților sistemului JMS fie de la mesajul JMS de intrare către componentele următoare, fie de la componentele anterioare către mesajul JMS de ieșire poate fi controlată prin stegulețul Propagate Protocol Header din legare.

Atunci când Propagate Protocol Header este setat, informațiilor de antet le este permis să circule către mesaj sau către componenta țintă, așa cum este descris în următoarea listă:

- Cerere de export JMS
Antetul JMS primit în mesaj va fi propagat către o componentă țintă prin intermediul serviciului de context. Proprietățile JMS primite în mesaj vor fi propagate către o componentă țintă prin intermediul serviciului de context.
- Răspuns la exportul JMS
Oricare dintre câmpurile din antetul JMS din serviciul de context va fi utilizat în mesajul de ieșire, dacă nu este înlocuită de proprietățile statice stabilite în legarea de export JMS. Oricare dintre proprietățile setate în serviciul de context va fi utilizată în mesajul de ieșire, dacă nu este înlocuită de proprietățile 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 ieșire, dacă nu este înlocuit de proprietățile statice stabilite în legarea de import JMS. Oricare dintre proprietățile setate în serviciul de context va fi utilizată în mesajul de ieșire, dacă nu este înlocuită de proprietățile statice stabilite în legarea de import JMS.
- Răspuns import JMS
Antetul JMS primit în mesaj va fi propagat către o componentă țintă prin intermediul serviciului de context. Proprietățile JMS primite în mesaj vor fi propagate către o componentă țintă prin intermediul serviciului de context.

Clienți externi:

Serverul poate trimite mesaje către, sau să primească mesaje de la, clienți externi folosind legări JMS WebSphere MQ.

Un client extern (de exemplu, un portal web sau un sistem de informații pentru întreprindere) poate trimite un mesaj către o componentă SCA din aplicație prin intermediul unui export sau poate fi invocat de o componentă SCA din aplicație prin intermediul unui import.

Legarea de export WebSphere MQ JMS implementează MDB-uri (message driven beans) cu scopul de a asculta ceririle ce intră prin destinația primire specificată în legarea de export. Destinația specificată în câmpul trimite este folosită pentru a trimite răspunsul către cererea de intrare în cazul în care aplicația invocată oferă un răspuns. Cu toate acestea, un client extern poate invoca aplicații prin intermediul legării de export.

WebSphere MQ JMS importă legarea în, și poate trimite un mesaj către, clienții externi. Acest mesaj ar putea sau nu să solicite un răspuns de la clientul extern.

Informații suplimentare despre modul de interacționare cu clienții externi folosind WebSphere MQ pot fi găsite în Centrul de informare WebSphere MQ.

Depanarea legărilor WebSphere MQ JMS:

Puteți diagnostica și rezolva probleme cu legările WebSphere MQ JMS.

Excepții de implementare

Ca răspuns la diferite condiții de eroare, implementarea MQ JMS import și export poate returna unul din cele două tipuri de excepții:

- Service Business Exception: această excepție este returnată dacă fault specificată pe interfața serviciului business (WSDL port type) a survenit.
- Service Runtime Exception: apărută în toate celelalte cazuri. În cele mai multe cazuri, cause exception va conține excepția originală (JMSEException).

De exemplu, un import așteaptă numai un mesaj de răspuns pentru fiecare mesaj de cerere. Dacă mai mult de un răspuns sosește, sau dacă un răspuns întârziat (unul pentru care expirarea răspunsului SCA are loc) sosește, o excepție Service Runtime Exception este aruncată. Tranzacția este repetată, și 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 își au originea într-un eșec de interacțiune WebSphere MQ JMS, ar trebui să vă așteptați să găsiți aceste mesaje în managerul de evenimente eşuate. Dacă astfel de mesaje nu apar în managerul de evenimente eşuate, asigurați-vă că valoarea proprietății 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 interacțiunea cu managerul de evenimente eşuate în timpul invocărilor SCA pentru legările MQ JMS.

Scenarii de folosire greșită: comparație cu legările WebSphere MQ

Legările WebSphere MQ JMS sunt proiectate să interacționeze cu aplicațiile JMS implementate în comparație cu WebSphere MQ, care expune mesajele în concordanță cu modelul de mesaje JMS. Totuși, importul și exportul WebSphere MQ, sunt în principal proiectate să interacționeze cu aplicațiile native WebSphere MQ și să expună întregul conținut al corpului mesajului WebSphere MQ la mediere.

Următoarele scenarii ar trebui construite folosind legarea WebSphere MQ JMS, și 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.
- Permișiunea 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.
- Mediarea conținutului 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 situații în care legarea WebSphere MQ și legarea WebSphere MQ JMS se așteaptă să interacționeze. În particular, când se face o punte între aplicații Java EE și non-Java EE WebSphere MQ, se va folosi export WebSphere MQ și import WebSphere MQ JMS (sau invers) în concordanță cu legările de date potrivite sau modulele de mediere potrivite (sau ambele).

Tratarea excepțiilor:

Modul în care este configurată legarea determină cum sunt tratate excepțiile 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 excepție.

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 excepție este determinat de cum implementați handler-ul de date sau legarea de date. Comportamentul recomandat este să vă proiectați legarea de date să arunce o **DataBindingException**.

Situația este similiară pentru un handler de date. Din moment ce handler-ul de date este invocat de legarea de date, orice excepție a handler-ului de date este înfășurată într-o excepție a legării de date. Prin urmare o **DataHandlerException** vă este raportată ca o **DataBindingException**.

Când orice excepție runtime, inclusiv o excepție **DataBindingException**, este aruncată:

- Dacă fluxul de mediere este configurat să fie tranzacțional, mesajul JMS este memorat în Managerul de evenimente eşuate implicit pentru reluare sau ștergere manuală.

Notă: Puteți 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 tranzacțional, excepția este înregistrată în istoric și mesajul este pierdut.

Situația este similiară pentru un handler de date. Deoarece handler-ul de date este apelat de legarea de date, o excepție a handler-ului de date este produsă înăuntrul unei excepții 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 aplicațiile WebSphere MQ.

Utilizați legările de export și 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 Service Integration Bus.

Atunci când o componentă interacționează cu un serviciu WebSphere MQ prin intermediul unui import, legarea de import WebSphere MQ utilizează o coadă către care vor fi trimise datele și o coadă unde poate fi primit răspunsul.

Atunci când un modul SCA oferă un serviciu clienților WebSphere MQ, legarea de export WebSphere MQ folosește o coadă în care pot fi primite cererile și către care pot fi trimise răspunsurile. Selectorul funcției asigură o mapare operației în cadrul componentei țintă care va fi invocată.

Conversia datelor de sarcină utilă către și 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 și de la un mesaj MQ se realizează prin intermediul legării datelor din antetul MQ.

Pentru informații despre versiunile WebSphere MQ suportate, vedeți pagina web cu cerințe detaliate de sistem.

Privire generală asupra legărilor WebSphere MQ:

Legarea WebSphere MQ asigură integrarea în aplicațiile native bazate pe MQ.

Taskuri administrative WebSphere MQ

Se așteaptă 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 aplicație care conține aceste legări.

Taskuri administrativ WebSphere

Trebuie să setați proprietatea **Cale bibliotecă nativă** din adaptorul pentru resurse MQ în Websphere cu versiunea WebSphere MQ suportată de server, și să reporniți serverul. Acest lucru asigură faptul că sunt utilizate bibliotecile unei versiuni acceptate a WebSphere MQ. Cerințe detaliate despre hardware și software se pot găsi pe Paginile de suport IBM .

Legări de import WebSphere MQ

Legarea de import WebSphere MQ permite componentelor din modul dumneavoastră SCA să comunice cu serviciile oferite de aplicațiile externe bazate pe WebSphere MQ. Trebuie să utilizați o versiune suportată de WebSphere MQ. Cerințe detaliate despre hardware și software se pot găsi pe Paginile de suport IBM .

Interacțiunea 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 funcție de tipul de operație invocată:

- Într-o direcție: Importul WebSphere MQ pune un mesaj pe coada configurată în câmpul **Coadă destinație 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 destinație de trimitere**

Coadă 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 aștepte 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 menționat faptul că WebSphere MQ este o legare asincronă. Dacă o componentă apelantă invocă un import în mod sincron WebSphere MQ (pentru o operație bidirecțională), atunci aceasta este blocată până când răspunsul este returnat de serviciul WebSphere MQ.

Figura 39 la pagina 131 ilustrează modul în care importul este legat de serviciul extern.

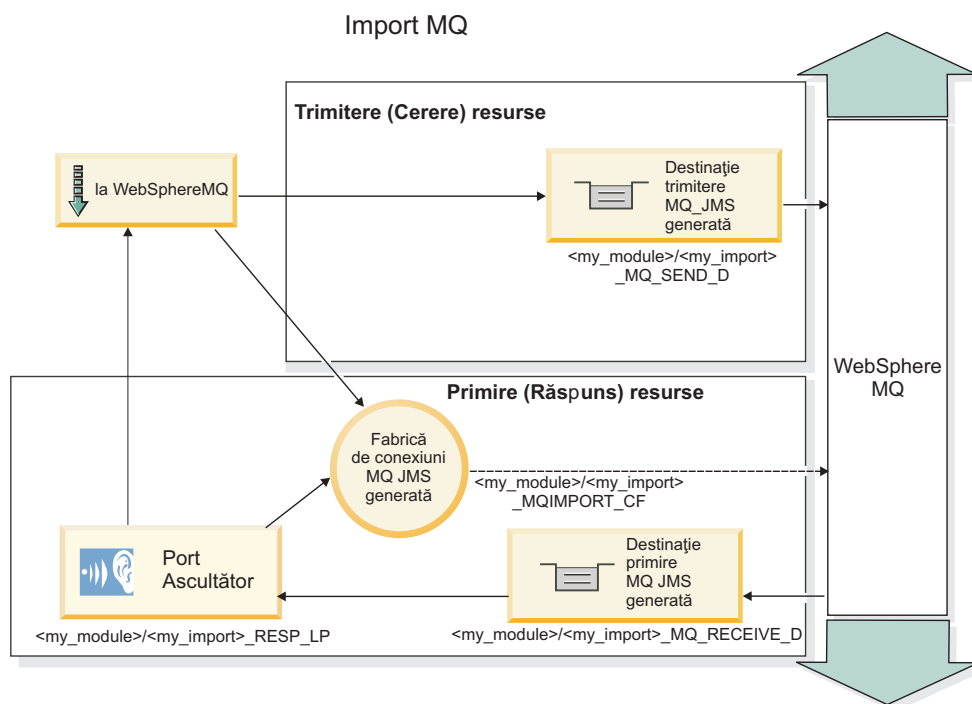


Figura 84. Resurse pentru legarea de import WebSphere MQ

Legări pentru export WebSphere MQ

Legarea de export WebSphere MQ pune la dispoziție mijloacele prin care modulele SCA asigură servicii către aplicațiile externe bazate pe WebSphere MQ.

Este implementat un MDB pentru a asculta cererile de intrare în **Coadă destinații de primire** specificată în legarea de export. Coada specificată în câmpul **Coadă destinații 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 înlocuiește coada specificată în câmpul **Coadă destinații de trimitere** field.

Figura 40 la pagina 132 ilustrează modul în care solicitantul extern este legat de export.

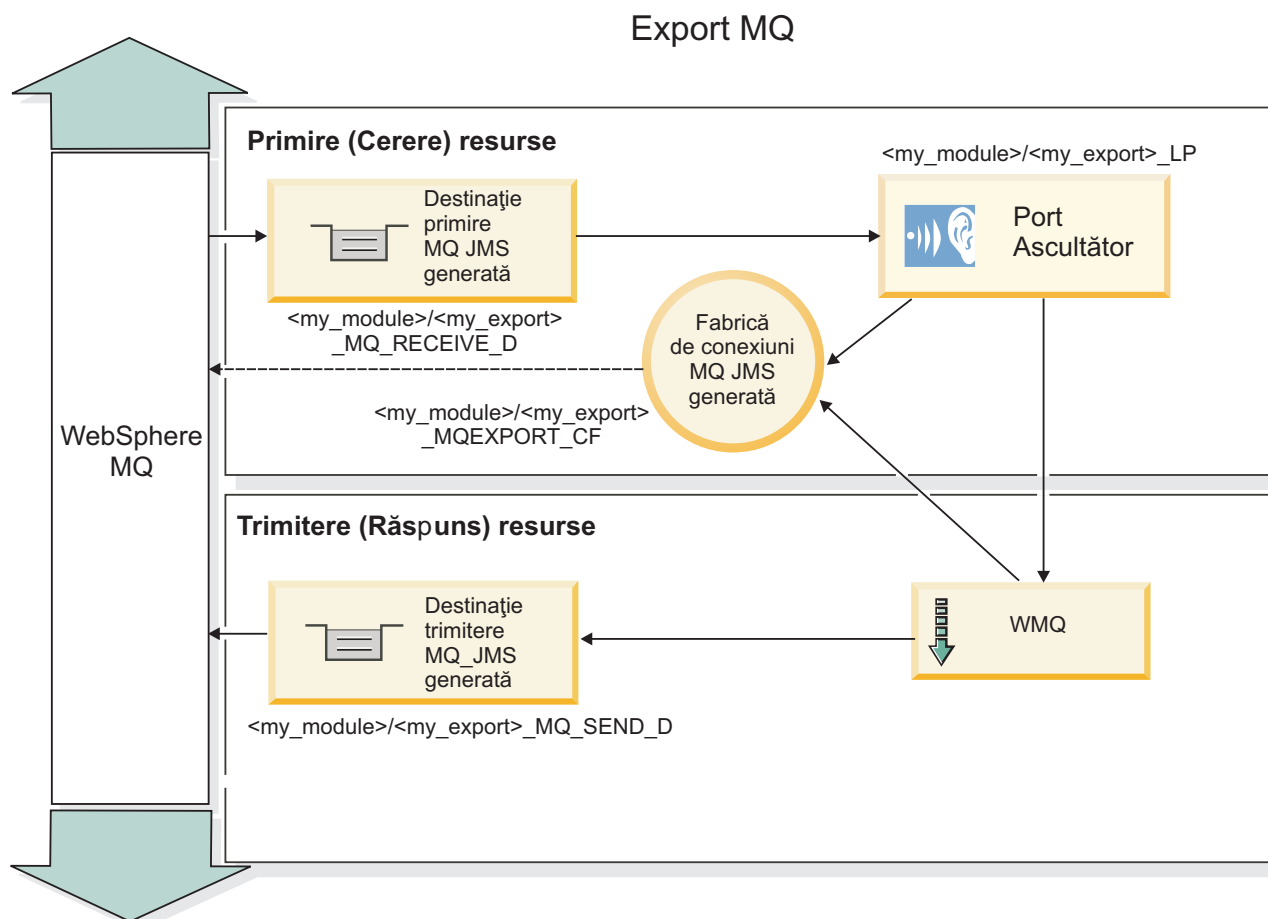


Figura 85. Resurse pentru legarea de export WebSphere MQ

Notă: Figura 39 la pagina 131 și Figura 40 la pagina 132 ilustrează modul în care o aplicație dintr-o versiune anterioară a IBM Business Process Manager este legată de un serviciu extern. Pentru aplicațiile dezvoltate pentru IBM Business Process Manager Versiunea 7, se folosește Specificare activare în loc de Port ascultător și Fabrică de conexiuni.

Caracteristici cheie ale unei legări WebSphere MQ:

Caracteristicile cheie ale unei legări WebSphere MQ includ anteturi, artefacte Java EE, și resurse create Java EE.

Scheme de corelare

O aplicație 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 și CorrelID. În marea majoritate a cazurilor, solicitantul lasă managerul de cozi să selecteze un MessageID și așteaptă ca aplicația de răspuns să copieze aceasta în CorrelID al răspunsului. În majoritatea cazurilor, solicitantul și aplicația de răspuns știu implicit ce tehnică de corelare este în uz. Ocazional aplicația de răspuns va onora diverse stegulețe în câmpul Raport al cererii ce descriu cum vor fi tratate aceste câmpuri.

Legările de export pentru mesajele WebSphere MQ pot fi configurate cu următoarele opțiuni:

Opțiuni 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 opțiuni de raport

Inspectează câmpul Raport al MQMD din cerere pentru o sugestie despre cum să fie tratat MsgId. Opțiunile MQRO_NEW_MSG_ID și MQRO_PASS_MSG_ID sunt suportate și se comportă ca New MsgId și Copy from Request MsgID.

Opțiuni CorrelId răspuns:

Copiere de la MsgID cerere

Copiază câmpul CorrelId de la câmpul MsgId din cerere (implicit).

Copiere de la CorrelID cerere

Copiază câmpul CorrelId de la câmpul CorrelId din cerere.

Copiere de la mesaj SCA

Setează CorrelId pentru a fi transportat în anteturile WebSphere MQ din mesajele de răspuns SCA sau îl lasă gol dacă valoarea nu există.

Ca opțiuni de raport

Inspectează câmpul Raport al MQMD din cerere pentru o sugestie despre cum să fie tratat CorrelId. Opțiunile MQRO_COPY_MSG_ID_TO_CORREL_ID și MQRO_PASS_CORREL_ID sunt suportate și se comportă ca Copy fromRequest MsgID și Copy from Request CorrelID.

Legările de import pentru mesajele WebSphere MQ pot fi configurate cu următoarele opțiuni:

Opțiuni MsgId cerere:

MsgID nou

Permite managerului cozii să selecteze un MsgId unic pentru cerere (implicit).

Copiere de la mesaj SCA

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.

Opțiuni corelare răspuns:

Răspunsul are CorrelID copiat de la MsgId

Se așteaptă ca mesajul de răspuns să aibă un câmp CorrelId setat, per MsgId al cererii (implicit).

Răspunsul are MsgID copiat de la MsgId

Se așteaptă ca mesajul de răspuns să aibă un câmp MsgId setat, per MsgId al cererii.

Răspunsul are CorrelID copiat de la CorrelId

Se așteaptă 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.

Parametrii

Fabrica de conexiuni MQ

Utilizată de clienți pentru a crea o conexiune la furnizorul WebSphere MQ.

Fabrica de conexiuni răspuns

Utilizată de runtime-ul SCA MQ atunci când destinația de trimitere este pe un Manager de coadă diferit ca destinația de recepționare.

Specificații de activare

O specificație de activare MQ JMS este asociată cu una sau mai multe bean-uri bazate pe mesaje și furnizează configurația necesară pentru a recepționa mesajele.

Destinații

- Destinație de trimitere: unde cererea sau mesajul de ieșire este trimis (import); unde mesajul de răspuns va fi trimis (export), dacă nu este permiat de câmpul de antet MQMD ReplyTo din mesajul de intrare.
- Destinație de recepționare: unde răspunsul/ceerea sau mesajul de intrare ar trebui plasat.

Anteturi WebSphere MQ:

Anteturile WebSphere MQ încorporează anumite convenții pentru conversie la mesaje SCA (Service Component Architecture).

Mesajele WebSphere MQ conțin un antet sistem (MQMD), zero sau mai multe alte anteturi MQ (sistem sau personalizate) și un corp mesaj. Dacă există mai multe anteturi în mesaj, ordinea anteturilor este semnificativă.

Fiecare antet conține informații 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 parsea 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 octeți neparsați.

Pentru alte anteturi MQ, puteți scrie legări de date antet MQ pentru a parsea 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 conține un element valoare de orice tip. Conține MQMD, MQControl (informații de control corp mesaj MQ) și o listă de alte anteturi MQ.

- MQMD reprezintă conținuturile descrierii mesajului WebSphere MQ, cu excepția informațiilor care determină structura și codarea corpului.
- MQControl conține informații care determină structura și codarea unui corp de mesaj.
- MQHeaders conține o listă de obiecte MQHeader.

Lațul antet MQ este desfăcut astfel încât, în interiorul SMO, fiecare antet MQ are propriile sale informații de control (CCSID, Codare și Format). Anteturile pot fi adăugate sau șterse cu ușurință, fără a modifica alte date ale antetului.

Setare câmpuri în MQMD

Puteți 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 ieșire:

- Codare
- CodedCharacterSet
- Format
- Raport
- Expirare
- Reacție
- Prioritate
- Persistență
- CorrelId
- MsgFlags

Configurați legarea MQ pe un Import sau Export pentru a propaga următoarele proprietăți la mesajul MQ de ieșire:

MsgID

Setare **ID mesaj de cerere** la copiere din mesaj SCA.

MsgType

Curățați caseta de bifare **Setare tip mesaj la MQMT_DATAGRAM sau MQMT_REQUEST pentru operații cerere-răspuns**.

ReplyToQ

Curățați caseta de bifare **Înlocuire răspuns la coadă de mesaje de cerere**.

ReplyToQMgr

Curățați 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 definiția destinației JNDI. Setați proprietatea personalizată MDCTX cu valoarea SET_IDENTITY_CONTEXT pe destinația de trimitere pentru a propaga câmpurile următoare la mesajul MQ de ieșire:

- UserIdentifier
- AppIdentityData

Setați proprietatea personalizată MDCTX cu valoarea SET_ALL_CONTEXT pe destinația de trimitere pentru a propaga proprietățile următoare la mesajul MQ de ieșire:

- UserIdentifier
- AppIdentityData
- PutApplType
- PutApplName

- ApplOriginData

Unele câmpuri nu sunt la mesajul MQ de ieșire. Câmpurile următoare sunt înlocuite în timpul trimiterii mesajului:

- BackoutCount
- AccountingToken
- PutDate
- PutTime
- Offset
- OriginalLength

Adăugarea statică a MQCIH într-o legare WebSphere MQ:

IBM Business Process Manager suportă adăugarea informațiilor de antet MQCIH static, fără folosirea unui modul de mediere.

Există diferite moduri de a adăuga informații de antet MQCIH unui mesaj (de exemplu, folosind primitiva de mediere Header Setter). Poate fi util să adăugați aceste informații de antet static, fără folosirea unui modul de mediere suplimentar. Informații de antet statice, inclusiv numele de program CICS, ID-ul tranzacției și alte detalii de antet de format de date, pot fi definite ca și parte a legării WebSphere MQ.

WebSphere MQ, MQ CICS Bridge și CICS trebuie să fie configurat pentru ca informațiile de antet MQCIH să fie adăugate static.

Puteți folosi Integration Designer pentru a configura importul WebSphere MQ cu valori statice care sunt necesare pentru informațiile de antet MQCIH.

Atunci când un mesaj ajunge și este procesat de către importul WebSphere MQ, este efectuată o verificare pentru vedeți dacă informația 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 și valorile statice definite în importul WebSphere MQ sunt adăugate.

Valorile statice definite în importul WebSphere MQ sunt specifice unei metode. Puteți specifica valori statice MQCIH diferite pentru diferite metode din același import WebSphere MQ.

Această facilitate nu este folosită pentru a furniza valori implicite dacă MQCIH nu conține informații de antet specifice deoarece o valoare statică definită în importul WebSphere MQ va înlocui o valoare corespunzătoare furnizată în mesajul de intrare.

Clienți externi:

IBM Business Process Manager poate trimite mesaje către, sau să primească mesaje de la, clienți externi folosind legări WebSphere MQ.

Un client extern (de exemplu, un portal web sau un sistem de informații pentru întreprindere) poate trimite un mesaj către o componentă SCA din aplicație prin intermediul unui export sau poate fi invocat de o componentă SCA din aplicație 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 destinația receive specificată în legarea de export. Destinația specificată în câmpul trimite este folosită pentru a trimite răspunsul către cererea de intrare în cazul în care aplicația invocată oferă un răspuns. Cu toate acestea, un client extern poate invoca aplicații prin intermediul legării de export.

WebSphere MQ importă legarea la, și poate trimite un mesaj către, clienții externi. Acest mesaj ar putea sau nu să solicite un răspuns de la clientul extern.

Informații suplimentare despre modul de interacționare cu clienții externi folosind WebSphere MQ pot fi găsite în Centrul de informare WebSphere MQ.

Depanarea legărilor WebSphere MQ:

Puteți diagnostica și rezolva condițiile de eșec ce apar în legările WebSphere MQ.

Condiții de eșec primar

Condițiile de eșec primar ale legărilor WebSphere MQ sunt determinate de către sensurile tranzacționale, de către configurația WebSphere MQ sau prin referire la comportamente existente în alte componente. Condițiile de eșec primar includ:

- Eșec în conexiunea la managerul de coadă sau la coada WebSphere MQ.
Un eșec la conectarea la WebSphere MQ pentru a primi mesaje va avea ca rezultat eșecul în pornirea portului MDB Listener. Această condiție 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 eșec la conectarea la WebSphere MQ pentru a trimite mesaje de ieșire va duce la derularea înapoi a operației de control a trimiterii.
- Eșecul în analiza unui mesaj de intrare sau în construirea unui mesaj de ieșire.
Un eșec la legarea de date sau la handler-ul de date cauzează derularea înapoi a tranzacției care controlează lucrul.
- Eșec la trimiterea mesajului de ieșire.
Un eșec la trimiterea unui mesaj cauzează derularea înapoi a tranzacției relevante.
- Multiple sau neașteptate mesaje de răspuns.
Importul așteaptă numai un mesaj de răspuns pentru fiecare mesaj de cerere. Dacă mai mult de un răspuns sosește, sau dacă un răspuns întârziat (unul pentru care expirarea răspunsului SCA are loc) sosește, o excepție Service Runtime Exception este aruncată. Tranzacția este repetată, și mesajul de răspuns este retras din coadă sau este manevrat de către managerul de eveniment.

Utilizarea greșită a scenariilor: comparație cu legările WebSphere MQ JMS

WebSphere MQ import și export sunt în principal proiectate să interacționeze cu aplicațiile native WebSphere MQ și să expună întregul conținut al corpului mesajului WebSphere MQ la medieri. Legările WebSphere MQ JMS sunt proiectate să interacționeze cu aplicațiile JMS implementate în comparație cu WebSphere MQ, care expune mesajele în concordanță cu modelul de mesaje JMS.

Următoarele scenarii ar trebui construite folosind legarea WebSphere MQ JMS, și 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.
- Permișiunea modulului SCA de a fi apelat dintr-un servlet componentă Java EE sau EJB prin intermediul căii JMS. Utilizați un export WebSphere MQ JMS.
- Medierea conținutului unui JMS MapMessage, la traversare de-a lungul WebSphere MQ. Utilizați un export și import WebSphere MQ JMS împreună cu legarea de date potrivită.

Există situații în care legarea WebSphere MQ și legarea WebSphere MQ JMS se așteaptă să interacționeze. În particular, când se face o punte între aplicații Java EE și non-Java EE WebSphere MQ, utilizați export WebSphere MQ și import WebSphere MQ JMS (sau invers) în concordanță cu legările de date corespunzătoare sau modulele de mediere potrivite (sau ambele).

Mesaje netrimise

Dacă WebSphere MQ nu poate livra un mesaj la destinația sa intenționată (de exemplu, din cauza erorilor de configurație), 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 conține motivele de eșec, destinația originală și alte informații.

Mesajele SCA bazate pe MQ nu apar în managerul de evenimente eșuate

Dacă mesajele SCA își au originea într-un eșec de interacțiune WebSphere, ar trebui să vă așteptați să găsiți aceste mesaje în managerul de evenimente eșuate. Dacă astfel de mesaje nu apar în managerul de evenimente eșuate, asigurați-vă că destinația WebSphere MQ ce stă la bază are valoarea maximă de livrări eșuate mai mare decât 1. Setarea acestei valori la 2 sau mai mult permite interacțiunea cu managerul de evenimente eșuate în timpul invocarilor SCA pentru legările WebSphere MQ.

Evenimentele eșuate MQ sunt reluate pe managerul de coadă greșit

Atunci când o fabrică de conexiuni predefinită trebuie să fie utilizată pentru conexiuni de ieșire, proprietățile de conexiune trebuie să se potrivească cu acelea definite în specificația de activare utilizată pentru conexiunile de intrare.

Fabrica de conexiuni predefinită este utilizată pentru a crea o conexiune la reluarea unui eveniment eșuat și, prin urmare, trebuie configurată pentru a utiliza același manager de coadă ca cel din care a fost primit inițial mesajul.

Tratarea excepțiilor:

Modul în care este configurată legarea determină cum sunt tratate excepțiile 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 excepție.

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 excepție este determinat de cum implementați handler-ul de date sau legarea de date. Comportamentul recomandat este să vă proiectați legarea de date să arunce o **DataBindingException**.

Situația este similiară pentru un handler de date. Din moment ce handler-ul de date este invocat de legarea de date, orice excepție a handler-ului de date este înfășurată într-o excepție a legării de date. Prin urmare o **DataHandlerException** vă este raportată ca o **DataBindingException**.

Când orice excepție runtime, inclusiv o excepție **DataBindingException**, este aruncată:

- Dacă fluxul de mediere este configurat să fie tranzacțional, mesajul JMS este memorat în Managerul de evenimente eșuate implicit pentru reluare sau ștergere manuală.

Notă: Puteți modifica modul de recuperare de pe legare, astfel încât mesajul să fie derulat înapoi în loc să fie stocat în managerul de evenimente eșuate.

- Dacă fluxul de mediere nu este tranzacțional, excepția este înregistrată în istoric și mesajul este pierdut.

Situația este similiară pentru un handler de date. Deoarece handler-ul de date este apelat de legarea de date, o excepție a handler-ului de date este produsă înăuntrul unei excepții 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 menționate aici.

Limitările legării MQ:

Legarea MQ are unele limitări în utilizarea sa. Acestea sunt menționate aici.

Distribuția mesajelor din abonamentele nepublicate

Metoda de abonare fără publicare a mesajelor distribuite nu este suportată în prezent de legarea MQ prin intermediul propriului WMQ. Totuși, legarea MQ JMS nu suportă această metodă de distribuție.

Cozi de primire partajate

Mai multe legări de export și import WebSphere MQ așteaptă ca orice mesaj prezent în coada lor configurată pentru recepție să fie intenționate pentru acel export sau import. Legările pentru importuri și exporturi ar trebui să fie configurate ținând cont de următoarele considerente:

- Fiecare import MQ trebuie să aibă o coadă diferită de recepționare deoarece legarea de import MQ presupune că toate mesajele din coada de recepționare sunt răspunsuri la cererile pe care le trimite. În cazul în care coada de recepționare este partajată între mai multe importuri, răspunsurile ar putea fi recepționate de un import greșit și nu vor putea fi corelate cu mesajul de cerere inițial.
- Fiecare export MQ ar trebui să aibă o coadă diferită de recepționare, deoarece altfel nu se poate anticipa care dintre exporturi va primi oricare dintre mesajele de cerere individuale.
- Importurile și exporturile MQ pot face referire către aceeași coadă de trimitere.

Limitările legărilor JMS, MQ JMS și Generic JMS:

Legările JMS și MQ JMS au unele limitări.

Implicațiile generării legărilor implicite

Limitările utilizării legărilor JMS, MQ JMS și JMS sunt discutate în următoarele secțiuni:

- Implicațiile generării legărilor implicite
- Schema de corelare a răspunsurilor
- Suport bidirecțional

Atunci când generați o legare, vor fi completate în mod implicit mai multe câmpuri în cazul în care nu alegeți să introduceți singur valorile. De exemplu, numele unui fabrici de conexiuni va fi creat pentru dvs. Dacă știți că veți pune aplicația dumneavoastră pe un server și o veți accesa de la distanță cu un client, ar trebui ca în momentul creării legării să introduceți numele JNDI în locul celor implicite, deoarece probabil veți dori să controlați aceste valori prin intermediul consolei administrative în momentul rulării.

Totuși, dacă ați acceptat valorile implicite, iar apoi ați aflat mai târziu că nu vă puteți accesa aplicația de la un client aflat la distanță, aveți posibilitatea să utilizați consola administrativă pentru a seta în mod explicit valoarea fabricii de conexiuni. Localizați câmpul pentru punctele finale care aparțin de furnizor în setările fabricii de conexiuni și adăugați valoarea în forma <nume_server>:7276 (dacă se utilizează numărul de port implicit).

Schema de corelare a răspunsurilor

În cazul în care folosiți schema de corelare a răspunsurilor CorrelationId To CorrelationId, care este folosită pentru a corela mesaje într-o operație de tip răspuns-răspuns, trebuie să aveți 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, adăugați o primitivă de mediere Mapare înainte de importul cu legarea JMS. Deschideți editorul de mapare. Anteturile arhitecturii componentei serviciului cunoscut vor fi disponibile în mesajul țintă. Trageți un câmp care conține un ID unic în mesajul sursă în ID-ul de corelare din antetul JMS din mesajul țintă.

Suportul bidirecțional

Doar caracterele ASCII sunt acceptate pentru numele JNDI (Java Naming and Directory Interface) la momentul execuției.

Cozi de primire partajate

Mai multe legări de export și import așteaptă ca orice mesaj prezent în coada lor configurată pentru recepție să fie intenționate pentru acel export sau import. Legările pentru importuri și exporturi ar trebui să fie configurate ținând cont de următoarele considerente:

- Fiecare legare pentru import trebuie să aibă o coadă diferită de recepționare deoarece aceasta presupune că toate mesajele din coada de recepționare sunt răspunsuri la cererile pe care le trimite. În cazul în care coada de recepționare este partajată între mai multe importuri, răspunsurile ar putea fi recepționate de un import greșit și nu vor putea fi corelate cu mesajul de cerere inițial.
- 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 și exporturile pot face referire la aceeași coadă de trimitere.

Obiectele business

Industria de software de calculator a dezvoltat mai multe modele de programare și cadre de lucru în care *obiectele business* furnizează o reprezentare naturală a datelor de afaceri pentru procesarea aplicației.

În general, aceste obiecte business:

- Sunt definite cu ajutorul standardelor din industrie
- Mapează transparent date la tabele de baze de date sau sisteme de informații de întreprindere
- Suportă protocoale de invocare la distanță
- Furnizează fundația modelului de programare de date pentru programare aplicație

Process Designer and Integration Designer le furnizează dezvoltatorilor un astfel de model de obiect business comun pentru reprezentarea diferitelor tipuri de entități de afaceri din diferite domenii.

În Process Designer, obiectele business sunt concentrate pe reprezentarea tipului de date. Obiectele business de bază (tipurile de variabile) sunt furnizate în seturile de unelte ale sistemului și puteți crea tipuri de variabile personalizate numite obiecte business personalizate. Consultați Obiecte business și variabile.

În Integration Designer, care este disponibil doar cu IBM Business Process Manager Advanced, obiectele business pot reprezenta mai multe construcții XSD. În Integration Designer, obiectele business seamănă foarte bine cu schemele XML. Pentru informațiile de luat în considerare atunci când integrați obiecte business objects care sunt definite în Integration Designer cu obiecte business care sunt definite în Process Designer, consultați Oglindire bibliotecă și construcții XML care nu sunt suportate.

La momentul dezvoltării Integration Designer, modelul obiectului business permite dezvoltatorilor să definească obiecte ca definiții de schemă XML. La momentul rulării, datele de afaceri care sunt definite prin definițiile din schema XML sunt reprezentate sub formă de Java Business Objects. În acest model, obiectele business sunt bazate pe ciorne anterioare ale specificației SDO (Service Data Object) și furnizează setul complet de interfețe aplicație model de programare necesare pentru manipularea datelor operaționale. Următoarele subsubiecte descriu mai departe modul în care puteți utiliza obiectele business împreună cu Integration Designer.

Definirea obiectelor business

Definiți obiecte business utilizând editorul de obiecte business din Integration Designer. Editorul de obiecte business memorează obiectele business ca definiții schemă XML.

Utilizarea schemei XML pentru a defini obiecte business furnizează mai multe avantaje:

- Schema XML furnizează un model de definiție de date bazat pe standarde și o fundație pentru interoperabilitatea dintre aplicații și sisteme eterogene incompatibile. Schemele XML sunt utilizate împreună cu WSDL (Web Services Description Language) pentru a furniza contracte de interfață bazate pe standarde de-a lungul componentelor, aplicațiilor și sistemelor.

- Schemele XML definesc un model de definiție date bogat pentru reprezentarea datelor operaționale. Acest model include tipuri complexe, tipuri simple, tipuri definite de utilizator, moștenire tip și cardinalitate printre alte caracteristici.
- Obiectele business pot fi definite de interfețe operaționale și date definite din Web Services Description Language, precum și de schema XML din organizații de standarde industriale sau din alte sisteme și aplicații. Integration Designer poate importa aceste obiecte business direct.

Integration Designer furnizează de asemenea suport pentru descoperirea datelor operaționale din baze de date și sisteme de informații de întreprindere și generarea definiției obiectului business al schemei XML bazată pe standarde a acelor date operaționale. Obiectele business generate în acest fel sunt adesea adresate ca *obiecte business specifice aplicației* deoarece imită structura datelor operaționale definite în sistemul de informații de întreprindere.

Când un proces manipulează date din mai multe sisteme de informații diferite, poate fi prețios să transformăm reprezentarea dispartă a datelor operaționale (de exemplu, CustomerEIS1 și CustomerEIS2 sau OrderEIS1 și OrderEIS2) într-o singură reprezentare canonică (de exemplu, Customer sau Order). Reprezentarea canonică este adesea adresată ca *obiectul business generic*.

Definițiile de obiecte business, în special pentru obiecte business generice, sunt utilizate frecvent de mai multe aplicații. Pentru a susține această reutilizare, Integration Designer permite ca obiectele business să fie create în biblioteci care pot fi apoi asociate cu module de aplicații multiple.

Web Services Description Language (WSDL) definește contractele pentru serviciile furnizate și consumate de un modul de aplicații SCA (Service Component Architecture) precum și contractele utilizate pentru a crea componentele dintr-un modul de aplicații. Într-un contract, un WSDL poate reprezenta atât operații, cât și obiecte business (care sunt definite de scheme XML pentru a reprezenta datele operaționale).

Lucrul cu obiecte business

SCA (Service Component Architecture) furnizează cadrul de lucru pentru definirea unui modul de aplicații, serviciile pe care le furnizează, serviciile pe care le consumă și alcătuirea componentelor care furnizează logica operațională a modulului de aplicații. Obiectele business joacă un rol important în aplicație, definind datele operaționale care sunt utilizate pentru a descrie contractele componentei și datele operaționale pe care le manipulează componentele.

Diagrama următoare descrie un modul de aplicații SCA și ilustrează multe dintre locurile în care dezvoltatorul lucrează cu obiecte business.

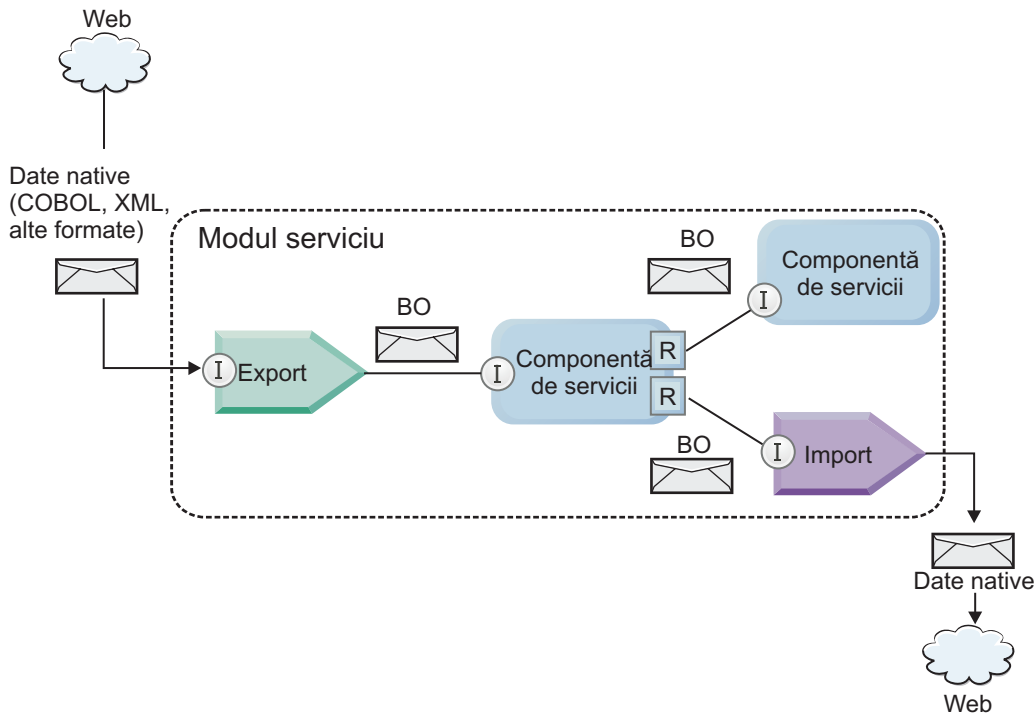


Figura 86. Obiectele business reprezintă datele care curg între servicii într-o aplicație

Notă: Acest subiect descrie cum sunt utilizate obiectele business de către modulele de aplicații SCA. Dacă utilizați interfețe Java, modulele de aplicații SCA pot procesa de asemenea obiecte Java.

Model de programare obiect business

Modelul de programare obiect business conține un set de interfețe Java care reprezintă:

- Definiția obiectului business și date instanță
- Un set de servicii care suportă operațiile de pe obiectele business

Definițiile tipului de obiect business sunt reprezentate de interfețele `commonj.sdo.Type` și `commonj.sdo.Property`. Modelul de programare obiect business furnizează un set de reguli pentru maparea informațiilor de tip complex ale schemei XML la interfața `Tip` și fiecare dintre elementele din definiția tipului complex la interfața `Proprietate`.

Instanțele obiectului business sunt reprezentate de interfața `commonj.sdo.DataObject`. Modelul de programare obiect business nu are tip, ceea ce înseamnă că aceeași interfață `commonj.sdo.DataObject` poate fi utilizată pentru a reprezenta diferite definiții de obiect business, cum ar fi `Client` sau `Comandă`. Definiția căror proprietăți pot fi setate și extrase din fiecare obiect business este determinată de informațiile de tip definite în schema XML asociată cu fiecare obiect business.

Comportamentul modelului de programare obiect business este bazat pe specificația `Service Data Object 2.1`. Pentru informații suplimentare, consultați `SDO 2.1` pentru specificațiile Java, îndrumare și `javadocs` pe web: <http://www.oasis-open.org/>.

Serviciile obiectului business suportă diverse operații de ciclu de viață (cum ar fi creare, egalitate, parsare și serializare) pe obiecte business.

Pentru lucruri specifice despre modelul de programare al obiectului business, vedeți `Programare utilizând servicii de obiecte business` și `Documentație API și SPI generată despre obiecte business`.

Legări, legări de date și handler-e de date

După cum a fost afișat în Figura 41 la pagina 142, datele operaționale care sunt utilizate pentru a invoca servicii furnizate de module de aplicații SCA sunt transformate în obiecte business, astfel încât componentele SCA să poată manipula datele operaționale. 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 și 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 convertește formatele nenative în obiecte business reprezentate de interfața DataObject.

Pentru informații suplimentare despre dezvoltarea legărilor de date și handler-elor de date, consultați “Handler-e de date” la pagina 54 și “Legări de date” la pagina 55.

Componente

Componentele SCA definesc contractele lor de servicii de provizionare și consum utilizând o combinație de Web Services Description Language și schemă XML. Datele operaționale pe care SCA le transmite între componente sunt reprezentate ca obiecte business utilizând interfața DataObject. SCA verifică dacă aceste tipuri de obiecte business sunt compatibile cu contractul interfeței definit de componenta care va fi invocată.

Teoriile modelului de programare pentru manipularea obiectelor business variază de la componentă la componentă. Componenta POJO și primitiva personalizată a componentei flux de mediere furnizează manipulare directă a obiectelor business prin activarea programării Java direct utilizând interfețele și 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 interfețele și serviciile obiectului business.

Obiectele business pot fi transformate utilizând fie combinația componentei Interface Flow Mediation și Business Object Map, fie componenta flux de mediere și primitiva sa XML Map. Aceste capacități de transformare obiecte business sunt utile pentru convertirea obiectelor business specifice aplicației la și de la obiecte business generice.

Obiectele business speciale

Obiectele de mesaje de servicii și graficele operaționale sunt două tipuri specializate de obiecte business care sunt utilizate în scopuri de aplicații 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 colecția de date asociată cu o invocare de servicii.

Un SMO are o structură de nivel înalt corectată alcătuită din anteturi, context, corp și atașamente (dacă sunt prezente).

- Anteturile transmit informații înrudite cu invocarea de servicii peste un anumit protocol sau legare. Exemple sunt anteturile SOAP și anteturile JMS.
- Datele de context transmit informații logice suplimentare asociate cu invocarea în timp ce sunt procesate de componenta fluxului de mediere. Aceste informații nu fac parte în mod tipic din datele de aplicație trimise sau recepționate de clienți.
- Corpul SMO-ului transmite datele operaționale ale datelor utile, care reprezintă mesajul de aplicație nucleu sau datele de invocare sub forma unui obiect business standard.

SMO poate transporta de asemenea date în atașament pentru invocările serviciilor web folosind SOAP cu atașamente.

Fluxurile de mediere realizează astfel de taskuri precum dirijare cerere și transformare de date, și SMO-ul furnizează vizualizarea combinată de conținut antet și date utile (payload) într-o singură structură unificată.

Grafic operațional

Un grafic operațional este un obiect business special utilizat pentru a furniza suport pentru sincronizare date în scenarii de integrare.

Considerați un exemplu în care două sisteme de informații 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 operaționale suportă noțiunea de trimitere doar a porțiunii de comandă care s-a modificat la celălalt sistem și adnotarea ei cu informații de modificare sumar pentru a defini tipul de modificare.

În acest exemplu, un Grafic operațional de comenzi va transmite celuilalt sistem că unul dintre articolele de linie din comandă a fost șters și că proprietatea dată de livrare proiectată a comenzii a fost actualizată.

Graficele operaționale pot fi adăugate cu ușurință la obiecte business existente din Integration Designer. Sunt găsite de cele mai multe ori în scenarii în care sunt utilizate adaptoarele WebSphere și pentru a ajuta migrarea aplicațiilor WebSphere InterChange Server.

Modul de parsare a obiectelor business

Integration Designer furnizează o proprietate pe module și biblioteci pe care le puteți vedea pentru a configura modul de parsare XML pentru obiecte business fie la *vioi*, fie la *leneș*.

- Dacă opțiunea este setată la *vioi*, fluxurile de octeți XML sunt parsate cu nerăbdare pentru a crea obiectul business.
- Dacă opțiunea este setată la *leneș*, obiectul business este creat normal, dar parsarea reală a fluxului de octeți XML este amânată și parsată parțial doar când sunt accesate proprietățile obiectului business.

În oricare dintre modurile de parsare XML, datele non-XML sunt întotdeauna parsate *vioi* pentru a crea obiectul business.

Considerații 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). Puteți modifica modul de parsare a modului sau bibliotecii, totuși ar trebui să fiți conștient de implicații.

Modul de parsare a obiectelor business este setat la nivelul modului sau al bibliotecii. Module 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 și bibliotecile care sunt create în IBM Integration Designer versiunea 7 și ulterioare vor primi cel mai adecvat mod de parsare în funcție de numărul de factori, cum ar fi modul de parsare al proiectelor existente în spațiul dumneavoastră de lucru sau modul de parsare al proiectelor dependente sau a altor proiecte din aceeași soluție și așa mai departe. Puteți modifica modul de parsare de obiect business al unui mod sau bibliotecă pentru a se potrivi cu implementarea dumneavoastră, totuși ar trebui să fiți conștienți de următoarele considerente.

Considerente

- Modul de parsare obiect business *leneș* procesează datele XML mai repede; totuși există diferențe de compatibilitate între modul *vioi* și modul *leneș* de care trebuie să fiți conștient înainte de modificarea configurației unui modul sau a unei biblioteci. Aceste diferențe vor afecta comportamentul din timpul rulării modulelor. Pentru informații despre modul optim de parsare pentru aplicația dumneavoastră, vedeți "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 pentru suportul, fie a 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*, și de un modul utilizând modul de parsare *leneș*. Modul de parsare al unei biblioteci la momentul rulării este determinat de modulele care fac referire la bibliotecă. La momentul execuției, un modul își declară modul de parsare, iar acel mod de parsare este utilizat de modul și de orice bibliotecă folosită de modul.
- Modulele și bibliotecile care sunt configurate pentru moduri de parsare diferite sunt compatibile în următoarele cazuri:

- Modulele și bibliotecile configurate cu modul de parsare leneș sunt compatibile cu bibliotecile care utilizează fie modul de parsare leneș, fie ambele moduri de parsare, leneș și vioi.
 - Modulele și bibliotecile configurate cu modul de parsare vioi sunt compatibile cu bibliotecile care utilizează fie modul de parsare vioi, fie ambele moduri de parsare, leneș și vioi.
 - Bibliotecile configurate cu modurile de parsare leneș și vioi sunt compatibile doar cu bibliotecile care utilizează ambele moduri de parsare, leneș și vioi.
- Utilizați același mod de parsare pentru module interactive care comunică utilizând legarea SCA. Dacă modulele comunică utilizând diferite moduri de parsare, ar putea rezulta probleme de performanță.

Concepte înrudite:

“Beneficii ale utilizării modului de parsare leneș față de vioi” la pagina 145

Unele aplicații beneficiază de modul leneș de parsare XML în timp ce altele văd o performanță îmbunătățită cu mod de parsare vioi. Este recomandat să măsurați aplicația dumneavoastră în ambele moduri de parsare pentru a determina ce mod se potrivește cel mai bine caracteristicilor specifice ale aplicației dumneavoastră.

Beneficii ale utilizării modului de parsare leneș față de vioi

Unele aplicații beneficiază de modul leneș de parsare XML în timp ce altele văd o performanță îmbunătățită cu mod de parsare vioi. Este recomandat să măsurați aplicația dumneavoastră în ambele moduri de parsare pentru a determina ce mod se potrivește cel mai bine caracteristicilor specifice ale aplicației dumneavoastră.

Este posibil ca aplicațiile care parsează fluxuri de date XML mari să vadă îmbunătățiri de performanță când este utilizat modul de parsare XML leneș. Beneficiile performanței cresc pe măsură ce dimensiunea șirului de octeți XML crește și cantitatea de date de la șirul de octeți care este accesat de către aplicație scade.

Este posibil ca aplicațiile următoare să funcționeze mai bine folosind un mod de parsare vioi:

- Aplicații care parsează fluxuri de date non-XML
- Aplicații care folosesc mesajele care sunt create folosind serviciul BOFactory
- Aplicații care parsează mesaje XML foarte mici

Referințe înrudite:

“Considerații 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). Puteți modifica modul de parsare a modulului sau bibliotecii, totuși ar trebui să fiți conștient de implicații.

Considerații de migrare și de dezvoltare de aplicații

Dacă configurați o aplicație care a fost dezvoltată inițial utilizând un mod de parsare vioi pentru a utiliza acum un mod de parsare leneș, sau dacă plănuți să comutați o aplicație între modul de parsare leneș și cel vioi, fiți conștient de diferențele dintre moduri și de considerente când comutați moduri.

Tratarea erorilor

Dacă fluxul de octeți XML care este parsat este format greșit, apar excepții de parsare.

- În modul de parsare XML vioi, acele excepții apar de îndată ce este parsat obiectul business din fluxul XML de intrare.
- Dacă este configurat modul de parsare XML leneș, excepțiile de parsare apar cu întârziere când sunt accesate proprietățile obiectului business și este parsată porțiunea XML-ului care este format greșit.

Pentru a vă ocupa de XML format greșit, selectați una dintre următoarele opțiuni:

- Implementarea unei magistrale de servicii de întreprindere pe margini pentru a valida XML de intrare
- Logica de detecție eroare autor în punctul în care sunt accesate proprietățile de obiect business

Mesaje și stive de excepții

Deoarece modurile de parsare XML leneș și vioi au implementări fundamentale diferite, urmele de stivă aruncate de interfețele și serviciile de programare ale obiectului business au același nume de clasă de excepții, dar ar putea să nu conțină același mesaj de excepție sau set înfășurat de clase de excepții specifice implementării.

Format serializare XML

Modul de parsare XML leneș furnizează o optimizare de performanță care încearcă să copieze XML nemodificat de la fluxul de oțeți de intrare la fluxul de oțeți de ieșire pe serializare. Rezultatul este performanță crescută, dar formatul de serializare al sirului de oțeți XML de ieșire poate fi diferit dacă întregul obiect business a fost actualizat în mod de parsare XML leneș sau dacă rulează în mod de parsare XML vioi.

Deși 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 și XML poate fi transmis în siguranță între aplicații care rulează în moduri de parsare diferite cu echivalență semantică.

Validator instanță obiect business

Validatorul instanței modului obiectului business de parsare XML leneș furnizează o validare de fidelitate mai înaltă a obiectelor business, în special validare fațetă de valori proprietate. Din cauza acestor îmbunătățiri, validatorul instanței modului de parsare leneș prinde probleme suplimentare care nu sunt prinse în modul de parsare vioi și furnizează mesaje de eroare mai detaliate.

Mapări XML versiune 602

Fluxurile de mediere dezvoltate inițial înainte de WebSphere Integration Developer Version 6.1 pot conține primitive Mapare care utilizează o hartă sau o foaie de stil care nu poate rula direct în modul de parsare XML mai lent. La migrarea unei aplicații pentru utilizarea în modul leneș de parsare XML, fișierele hartă asociate cu primitivele Mapare pot fi actualizate în mod automat de către vrăjitorul de migrare pentru a rula în noul mod. Totuși, în cazul în care o primitivă Mapare se referă în mod direct la o foaie de stil care a fost editată manual, foaia de stil nu este migrată și nu poate rula în modul XML mai lent.

API-uri nepublicate private

Dacă o aplicație profită de interfețe de programare obiecte business specifice implementării, private, nepublicate, aplicația are toate șansele să eșueze compilarea când este schimbat modul de parsare. În mod de parsare vioi, aceste interfețe 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 aplicație.

API-uri EMF Obiect mesaj serviciu

O componentă de mediere din IBM Integration Designer furnizează abilitatea de a manipula conținut mesaj utilizând clase și interfețe Java furnizate în pachetul `com.ibm.websphere.sibx.smobo`. În mod de parsare XML leneș, interfețele Java din pachetul `com.ibm.websphere.sibx.smobo` pot fi încă utilizate, dar metodele care se referă direct la clasele și interfețele EMF (Eclipse Modeling Framework) sau care sunt moștenite de la interfețele EMF au toate șansele să eșueze.

`ServiceMessageObject` și conținuturile sale nu pot fi atribuite obiectelor EMF în mod de parsare XML leneș.

Serviciu BOMode

Serviciul BOMode este utilizat pentru a determina dacă modul de parsare XML care se execută momentan este vioi sau leneș.

Migrare

Toate aplicațiile 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 aplicații anterioare versiunii 7.0.0.0 să fie configurată pentru a utiliza modul de parsare XML, utilizați întâi Integration Designer pentru a migra artefactele aplicației. După migrare, configurați aplicația pentru a utiliza parsare XML leneșă.

Vedeți Migrare artefacte sursă pentru informații despre migrarea artefactelor din Integration Designer și vedeți Configurarea modului de parsare obiect business al modulelor și bibliotecilor pentru informații despre setarea modului de parsare.

Relații

O relație este o asociere între două sau mai multe entități de date, în special obiecte business. În IBM Business Process Manager Advanced, relațiile pot fi utilizate pentru a transforma date care sunt echivalente peste obiecte business și alte date care sunt reprezentate diferit, sau pe care le pot utiliza pentru a trasa aplicații peste diferite obiecte găsite în aplicații diferite. Acestea pot fi partajate între aplicații, soluții și chiar produse.

Serviciul de relații din IBM Business Process Manager Advanced furnizează infrastructura și operațiile pentru gestionarea relațiilor. Deoarece vă permite să tratați obiecte business indiferent de locul în care acestea se află, acesta vă poate oferi o vizualizare de asamblare unificată a tuturor aplicațiilor dintr-o întreprindere și poate fi folosit ca un bloc de construire pentru soluții BPM. Deoarece relațiile pot fi extinse și gestionate, ele pot fi utilizate în soluții complexe de integrare.

Ce sunt relațiile?

O relație este o asociere între obiecte business. Fiecare obiect business dintr-o relație este numit *participant* la relație. Fiecare participant din relație este distins de alți participanți pe baza funcției sau a *rolului* pe care îl are în acea relație. O relație conține o listă de roluri.

Definiția relației descrie fiecare rol și specifică modul în care sunt înrudite rolurile. De asemenea, aceasta descrie "forma" generală a relației. De exemplu, acest rol poate avea doar un participant, dar acest alt rol poate avea câți participanți sunt necesari. Puteți defini o relație *car-owner*, de exemplu, unde un posesor poate avea mai multe mașini. De exemplu, o instanță poate avea următorii participanți pentru fiecare dintre aceste roluri:

- Mașină (Ferrari)
- Posesor (John)

Definiția relației este un șablon pentru *instanța* relației. Instanța este instanțierea relației în timpul rulării. În exemplul cu *proprietarii de mașini*, o instanță ar putea descrie oricare din următoarele asocieri:

- John deține Ferrari
- Sara deține Mazda
- Bob deține Ferrari

Utilizarea relațiilor vă eliberează de necesitatea de a construi personalizat persistența de urmărire a relației din logica dumneavoastră operațională. Pentru anumite scenarii, serviciul de relații face toată munca în locul dumneavoastră. Vedeți exemplul descris în secțiunea din Relații de identitate.

Scenarii

Aici este un exemplu tipic de situație în care o soluție de integrare poate folosi relații. O corporație mare cumpără mai multe companii sau unități de afaceri. Fiecare unitate operațională folosește software diferit pentru a monitoriza personalul și caietele. Compania are nevoie de o cale de a-și monitoriza angajații și carnetele lor. Aceasta vrea o soluție care le permite:

- Vizualizarea tuturor angajaților din diverse unități de afaceri ca și cum ar fi într-o singură bază de date
- O singură vizualizare a tuturor carnetelor
- Permite angajaților să se logheze în sistem și să cumpere un carnet.
- Acomodarea diferitelor sisteme de aplicații de întreprindere din diverse unități de afaceri

Pentru a realiza acest lucru, compania are nevoie de o cale pentru a se asigura, de exemplu, că John Smith și John A. Smith din aplicații diferite sunt văzuți ca același angajat. De exemplu, aceștia au nevoie de o cale pentru a consolida o singură entitate în mai multe spații ale aplicației.

Scenariile mai complexe de relații implică construirea proceselor BPEL care stabilesc relații între obiecte diferite găsite în mai multe aplicații. Cu scenarii mai complexe de relații, obiectele business se află în soluția de integrare, ci nu în aplicații. Serviciul de relații asigură o platformă pentru gestionarea în mod persistent a relațiilor. Înainte de serviciul de relații, trebuie să construiești propriul serviciu de persistență al obiectului. Două exemple de scenarii complexe de relații sunt:

- Aveți un obiect business **car** cu un număr VIN într-o aplicație SAP și vreți să urmăriți că această mașină este deținută de altcineva. Totuși, relația de posesiune este cu cineva într-o aplicație PeopleSoft. În acest tipar de relații, aveți două soluții și este nevoie să construiești o punte laterală între ele.
- O companie mare de comerț cu amănuntul vrea să fie capabilă să monitorizeze marfa returnată pentru primirea banilor înapoi sau pentru credit. Există două aplicații diferite implicate: OMS (order management system) pentru achiziții și RMS (returns management system) pentru returnări. Obiectele business se află în mai multe aplicații și aveți nevoie de o cale de a afișa relațiile care există între ele.

Modele de utilizare comune

Cele mai comune tipare de relații sunt tiparele *echivalență*. Acestea sunt bazate pe referința încrucișată sau pe corelare. Există două tipuri de relații care se potrivesc cu acest tipar: *non-identitate* și *identitate*.

- **Relații non-identitate** stabilesc asocieri între obiecte business sau alte date pe o bază unu-la-mulți sau mulți-la-mulți. Pentru fiecare instanță a relației, pot exista una sau mai multe instanțe pentru fiecare participant. Un tip de relație non-identitate este o relație de căutare statică. Un astfel de exemplu este o relație în care **CA** dintr-o aplicație SAP este înrudită cu **California** dintr-o aplicație Siebel.

•

Relații de identitate stabilesc asocieri între obiecte business sau alte date pe o bază unu-la-unu. Pentru fiecare instanță a relației, poate exista doar o instanță a fiecărui participant. Relațiile de identitate capturează referințe încrucișate între obiecte business care sunt echivalente semantic, dar care sunt identificate în mod diferit în diferite aplicații. Fiecare participant la relație este asociat cu un obiect business care are o valoare (sau combinație de valori) care identifică obiectul în mod unic. În mod obișnuit, relațiile de identitate transformă atributele cheie ale obiectelor business precum numere ID și coduri de produs.

De exemplu, dacă aveți obiecte business **car** în aplicații SAP, PeopleSoft și Siebel și vreți să construiești o soluție care să le sincronizeze, va fi nevoie, în mod normal, să introduceți logica de sincronizare a relației construite manual în șase mapări:

SAP -> generic

generic -> SAP

PeopleSoft-> generic

generic-> PeopleSoft

Siebel-> generic

generic-> Siebel

Totuși, dacă utilizați relații în soluția dumneavoastră, serviciul de relații asigură implementări de tipare preconstruite care mențin toate aceste relații pentru dumneavoastră.

Unelte pentru lucrul cu relații

Editorul de relații din Integration Designer este unelta pe care o utilizați pentru a modela și proiecta relații și roluri de integrare business. Pentru informații detaliate despre experiența necesară și despre taskuri la crearea relațiilor și utilizarea editorului de relații, vedeți Creare relații.

Serviciul de relații este un serviciu de infrastructură din IBM Business Process Manager care menține relații și roluri în sistem și oferă operații pentru gestiunea relației și a rolului.

Managerul de relații este interfața administrativă pentru gestionarea relațiilor. Acesta este accesat prin pagina Manager de relații a consolei administrative.

Relațiile pot fi invocate programatic prin API-urile serviciului de relații.

Serviciu de relații

Serviciul de relații memorează date ale relației în tabele de relații unde păstrează pista valorilor specifice aplicației din aplicații și soluții. Serviciul de relații oferă operații pentru gestiunea relațiilor și a rolului.

Cum funcționează relațiile

Relațiile și rolurile sunt descrise folosind interfața grafică a unelei editor de relații din Integration Designer. Serviciul de relații memorează date de corelare în tabele din baza de date a relației în sursa implicită de date pe care o specificați la configurarea serviciului de relații. O tabelă separată (uneori numită tabelă participantă) memorează informații pentru fiecare participant la relație. Serviciul de relații utilizează aceste tabele ale relației pentru a păstra pista valorilor înrudite specifice aplicației și pentru a propaga informații actualizate în toate soluțiile.

Relațiile, care sunt artefacte business, sunt implementate într-un proiect sau într-o bibliotecă partajată. La prima implementare, serviciul de relații populează datele.

În timpul rulării, când mapările sau alte componente IBM Business Process Manager necesită o instanță de relație, instanțele relației sunt fie actualizate, fie extrase în funcție de scenariu.

Datele instanței de relație sau rol pot fi manipulate prin trei mijloace:

- Invocări componentă IBM Business Process Manager snippet Java ale API-urilor serviciului de relații
- Transformări ale relației în serviciul de mapare a obiectului business IBM Business Process Manager
- Unealta managerului de relații

Pentru informații detaliate despre experiența necesară și despre taskuri la crearea relațiilor, identificarea tipurilor de relații și utilizarea editorului de relații, vedeți subiectul Creare relații.

Manager de relații

Managerul de relații este interfața administrativă pentru gestionarea relațiilor. Acesta este accesat prin pagina Manager de relații a consolei administrative.

Managerul de relații asigură o interfață grafică cu utilizatorul pentru crearea și manipularea datelor relației și ale rolului în timpul rulării. Puteți gestiona entități ale relației la toate nivelurile: instanță de relație, instanță de rol și niveluri ale datelor atribut și ale datelor proprietate. Cu managerul de relații, puteți:

- Vizualiza o listă a relațiilor din sistem și a informațiilor detaliate despre relații individuale
- Gestiona instanțe de relații:
 - Interoga date ale relației pentru a vizualiza subseturi de date ale instanței
 - Interoga date ale relației pentru a vizualiza subseturi de date ale instanței utilizând vizualizări ale bazei de date
 - Vizualiza o listă a instanțelor relației care se potrivesc unei interogări a relației și informații detaliate despre o instanță

- Edita valorile proprietății pentru o instanță a relației
- Crea și șterge instanțe de relație
- Gestiona roluri și instanțe de roluri:
 - Vizualiza detalii despre un rol sau o instanță de rol
 - Edita proprietățile instanței de rol
 - Crea și șterge instanțe de rol pentru o relație
 - Derula înapoi date ale instanței de relație la un punct în care știți că datele sunt de încredere
- Importa date dintr-o relație statică existentă în sistemul dumneavoastră sau exporta date dintr-o relație statică existentă într-un fișier RI sau CSV
- Înlătura schema și datele relației din magazie când aplicația care le folosește este dezinstalată

Relații în medii Network Deployment

Relațiile pot fi utilizate în medii ND (Network Deployment) fără nicio configurație suplimentară.

În medii ND (Network Deployment), relațiile sunt instalate într-un cluster de aplicații. Atunci relațiile sunt vizibile în cluster și toate serverele din cluster au acces la datele instanței memorate în baza de date a relației. Abilitatea de rula serviciul de relații într-un mediu ND îl face scalabil și cu disponibilitate bună.

Managerul de relații permite relațiilor să fie gestionate în cluster-e diferite printr-o interfață administrativă centralizată. Conectați managerul de relații la un server dintr-un cluster selectând MBean-ul relației sale.

API-uri ale serviciului de relații

Relațiile pot fi invocate programatic prin API-urile serviciului de relații în sau în afara mapărilor de obiecte business.

Sunt disponibile trei tipuri de API-uri:

- API-uri de manipulare a instanței de relație (inclusiv crearea, actualizarea, ștergerea directă a datelor instanței)
- API-uri suport ale tiparului de relații (inclusiv correlate(), correlateforeignKeyLookup)
- Tipare căutare relație (API-uri de căutare)

Magistrala ESB (Enterprise Service Bus) din IBM Business Process Manager

IBM Business Process Manager suportă integrarea serviciilor aplicație, inclusiv aceleași capabilități ca WebSphere Enterprise Service Bus.

Conectarea serviciilor printr-o magistrală pentru serviciile întreprinderii

Cu ajutorul unui ESB (Enterprise Service Bus), puteți maximiza flexibilitatea unui SOA. Participanții din interacțiunea cu serviciul sunt conectați mai degrabă la ESB, decât direct unul cu altul.

În cazul în care solicitantul serviciului se conectează la ESB, atunci ESB-ul își asumă răspunderea pentru furnizarea cererilor sale, folosind mesaje, către un furnizor de servicii oferind funcția necesară și calitatea serviciilor. ESB ușurează interacțiunile solicitant-furnizor și adresează protocoalele nepotrivite, tiparele de interacțiune sau capabilitățile serviciului. De asemenea, un ESB poate permite sau îmbunătăți monitorizarea și gestiunea. ESB oferă caracteristici de virtualizare și management care pun în aplicare și extind capacitățile de bază pentru SOA.

ESB prezintă pe scurt următoarele caracteristici:

Locație și identitate

Participanții nu trebuie să știe locația sau identitatea altor participanți. De exemplu, solicitanții nu trebuie să fie conștienți de faptul că o cerere ar putea fi deservită de oricare dintre furnizori; furnizorii de servicii pot fi adăugați sau înlăturați fără întrerupere.

Protocol de Interacțiune

Participanții nu trebuie să partajeze același protocol de comunicație sau stil de interacțiune. De exemplu, o cerere exprimată ca SOAP peste HTTP poate fi deservită de un furnizor care înțelege doar SOAP peste JMS (Java Message Service).

Interfață

Solicitanții și furnizorii nu trebuie să fie de acord cu o interfață comună. Un ESB împacă diferențele prin transformarea mesajelor de cerere și de răspuns într-o formă așteptată de furnizor.

Calitățile unui serviciu (de interacțiune)

Participanții, sau administratorii de sisteme, își declară cerințele legate de calitatea serviciilor, inclusiv autorizația cererilor, criptarea și decriptarea conținutului mesajelor, auditarea automată a interacțiunilor dintre servicii, și modul în care cererile lor ar trebui să fie rutate (de exemplu optimizarea vitezei sau a costului).

Interpunerea ESB între participanți vă dă posibilitatea să modulați interacțiunea acestora prin intermediul unei construcții logice numite *mediere*. Medierile se aplică mesajelor aflate în zbor între solicitanți și furnizori. De exemplu, medierile pot fi folosite pentru a găsi servicii cu caracteristici specifice cerute de un solicitant, sau pentru a rezolva diferențele interfeței apărute între solicitanți și furnizori. Pentru interacțiuni complexe, medierile pot fi legate secvențial.

Folosind medieri, o magistrală pentru serviciile întreprinderii efectuează următoarele acțiuni între solicitant și serviciu:

- *Rutarea* mesajelor între servicii. O magistrală pentru serviciile întreprinderii oferă o infrastructură de comunicație comună care poate fi folosită pentru a conecta servicii, și, prin urmare funcțiile business pe care acestea le reprezintă, fără a fi nevoie ca programatorii să scrie și să mențină logica complexă de conectivitate.
- *Convertirea* protocoalelor de transport între solicitant și serviciu. O magistrală pentru serviciile de întreprindere oferă o cale consistentă bazată pe standarde de integrare a funcțiilor business care folosesc diferite standarde IT. Acest lucru permite integrarea funcțiilor business care nu au putut comunica în mod normal, cum ar fi conectarea aplicațiilor în silozurile departamentale sau activarea aplicațiilor în companii diferite care participă în interacțiunile serviciului.
- *Transformarea* formatelor mesajelor între solicitant și serviciu. O magistrală pentru serviciile de întreprindere permite funcțiilor business să schimbe informații în formate diferite, iar magistrala se asigură că informațiile livrate funcției business sunt în formatul cerut de acea aplicație.
- *Tratarea* evenimentelor business din surse incompatibile. O magistrală pentru serviciile de întreprindere suportă interacțiuni bazate pe evenimente în plus față de schimbul de mesaje ce tratează cererile serviciului.

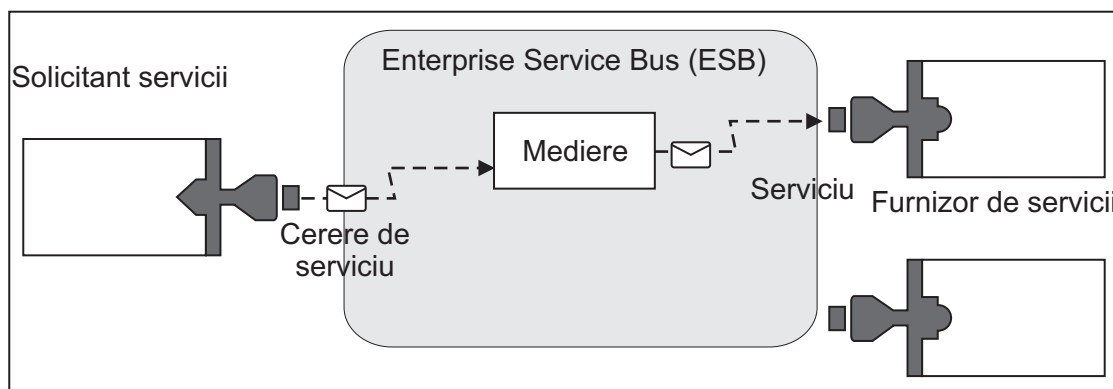


Figura 87. O magistrală pentru serviciile întreprinderii. Magistrală pentru serviciile de întreprindere rutează mesajele între aplicații, care pot fi solicitanți sau furnizori de servicii. Magistrala convertește protocoalele de transport și transformă formatul mesajelor între solicitanți și furnizori. În această figură, fiecare aplicație utilizează un protocol diferit (reprezentate prin formele geometrice diferite ale conectorilor lor) și folosește diferite formate de mesaje.

Folosind magistrala pentru serviciile de întreprindere vă puteți concentra pe afacerea dumneavoastră de bază mai degrabă decât pe sistemele dumneavoastră computer. Puteți modifica sau adăuga la servicii, dacă este necesar; de exemplu, pentru a răspunde la modificările din cerința de afaceri, pentru a adăuga capacitate de serviciu suplimentar

sau pentru a adăuga noi capabilități. Aveți posibilitatea să efectuați modificările necesare prin reconfigurarea magistralei, cu impact redus sau nul asupra serviciilor și aplicațiilor existente care folosesc magistrala.

Infrastructura de mesagerie ESB (Enterprise Service Bus)

IBM Business Process Manager include capabilități ale magistralei ESB. IBM Business Process Manager suportă integrarea tehnologiilor orientate spre servicii, orientate pe mesaj și conduse de eveniment pentru a asigura o infrastructură a mesageriei bazată pe standarde într-o magistrală de servicii de întreprindere integrate.

Capabilitățile serviciului de întreprindere pe care le puteți utiliza pentru aplicațiile dumneavoastră de întreprindere asigură nu doar un nivel de transport dar și suport de mediere pentru a facilita interacțiunile serviciului. Magistrala ESB (Enterprise Service Bus) este construită în jurul standardelor deschise și a SOA (service-oriented architecture). Este bazată pe infrastructura Java EE robustă și serviciile platformei asociate furnizate de IBM WebSphere Application Server Network Deployment.

IBM Business Process Manager este alimentat de aceeași tehnologie disponibilă cu IBM WebSphere Enterprise Service Bus. Această capabilitate este parte componentă a funcționalității de bază a IBM Business Process Manager și nu este necesară nicio licență suplimentară pentru WebSphere Enterprise Service Bus pentru a beneficia de aceste capabilități.

Totuși, puteți implementa licențe autonome suplimentare ale WebSphere Enterprise Service Bus în jurul întreprinderii dumneavoastră pentru a extinde întinderea conectivității la soluții de integrare a procesului alimentate de IBM Business Process Manager. De exemplu, WebSphere Enterprise Service Bus poate fi instalat mai aproape de o aplicație SAP pentru a găzdui un IBM WebSphere Adapter for SAP și pentru a transforma mesajele SAP înainte de a trimite acele informații în rețea într-un proces operațional condus de IBM Business Process Manager.

Gazde destinație mesagerie sau coadă

O gazdă destinație mesagerie sau coadă furnizează funcția mesagerie într-un server. Un server devine gazda destinației mesageriei la configurarea lui ca destinație mesagerie.

Un motor de mesagerie rulează într-un server. Motorul mesagerie furnizează funcții mesagerie și un punct de conexiune pentru ca aplicațiile să se conecteze la magistrală. Comunicația asincronă Service Component Architecture (SCA), importurile și exporturile JMS, procesarea internă asincronă utilizează cozi de mesaje în motorul de mesagerie.

Mediul de implementare conectează sursa mesajului la destinația mesajului prin magistrală la implementarea modulelor de aplicații. Cunoașterea sursei mesajului și a destinației mesajului vă ajută să determinați de ce tip de mediu de implementare aveți nevoie.

Aplicațiile 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 fișiere. Motorul mesagerie utilizează o instanță a unei surse de date JDBC pentru a interacționa cu acea bază de date.

Configurați gazda destinație a mesageriei la definirea mediului de implementare utilizând **Server** din consola administrativă sau desemnați serverul ca gazdă destinație 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 reținute în aceeași schemă a bazei de date. Puteți crea fiecare depozit de date într-o bază de date separată. Alternativ, puteți crea mai multe depozite de date în aceeași bază de date, fiecare depozit de date utilizând o schemă diferită.

Un motor de mesagerie utilizează o instanță a unei surse de date JDBC pentru a interacționa cu baza de date care conține depozitul de date pentru acel motor de mesagerie.

Furnizori JDBC

Puteți folosi furnizorii JDBC pentru interacționarea aplicațiilor cu bazele de date relaționale.

Aplicațiile folosesc furnizori JDBC pentru interacționarea cu bazele de date relaționale. 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, asociați sursa de date cu furnizorul JDBC. Împreună, furnizorul JDBC și obiectele sursei de date sunt echivalente funcțional cu fabrica de conexiuni Java EE Connector Architecture (JCA), care furnizează conexiunea cu o bază de date non-relațională.

Referiți-vă la exemplele din Setări pentru mediul tipic autonom și Setări pentru mediul tipic de implementare din subiectul anterior.

Pentru informații suplimentare despre furnizorii JDBC, vedeți “Furnizori JDBC” în Centrul de informare WebSphere Application Server.

Magistrala de integrare servicii pentru IBM Business Process Manager

O magistrală de integrare a serviciului este un mecanism de comunicare gestionat care suportă integrarea serviciului prin mesageria sincronă și asincronă. O magistrală conține motoare de mesagerie interconectate care gestionează resursele magistralei. Este una din tehnologiile WebSphere Application Server pe care este bazat IBM Business Process Manager.

O singură magistrală de integrare servicii și un singur motor de mesagerie folosesc implicit aceeași schemă de bază de date ca și baza de date a produsului. Fiecare mediu de implementare are magistrala sa proprie. Magistrala singulară se numește **BPM.nume_mediu_implementare.Bus**.

O destinație magistrală este o adresă logică la care aplicațiile se pot atașa ca producător, consumator sau ambele. O destinație coadă este o destinație magistrală care este utilizată pentru mesagerie punct-la-punct.

Fiecare magistrală poate avea unul sau mai mulți membri magistrală, fiecare dintre ei fiind un server sau un cluster.

Topologie magistrală este aranjarea fizică a serverelor de aplicații, motoarelor de mesagerie și a managerilor de cozi WebSphere MQ și tiparul conexiunilor magistralei și al legăturilor dintre ele care alcătuiesc magistrala de servicii întreprindere.

Aplicații de servicii și module de servicii

Un modul serviciu este un modul Service Component Architecture (SCA) care oferă servicii în mediul runtime. Atunci când instalați un modul service pe IBM Business Process Manager, construiți o aplicație servicii asociată, care este împachetată ca un fișier EAR (enterprise archive).

Modulele de servicii sunt unități de bază ale implementării și pot conține componente, biblioteci și module de intermediere folosite de aplicația pentru servicii asociată. Modulele de servicii au exporturi și, în mod opțional, importuri pentru a defini relațiile între module și solicitanții și furnizorii serviciului. WebSphere Process Server suportă module pentru serviciile business și module de mediere. Atât modulele, cât și modulele de mediere sunt tipuri de module SCA. Un modul de mediere permite comunicația între aplicații prin transformarea invocării serviciului într-un format înțeles de țintă, transmițând cererea către țintă și returnând rezultatul originatorului. Un modul pentru un serviciu operațional implementează logica unui proces operațional. Totuși, un modul poate de asemenea să includă aceeași logică de mediere care poate fi împachetată într-un modul de mediere.

Implementarea unei aplicații de servicii

Procesul de implementare a unui fișier EAR care conține o aplicație de servicii este același cu cel de implementarea a oricărui fișier EAR. Puteți modifica valorile pentru parametrii de mediere în momentul implementării. După ce au implementat un fișier EAR care conține un modul SCA, puteți vizualiza detaliile legate de aplicația de servicii și modulele sale asociate. Puteți vedea modul în care un modul de servicii este conectat la solicitanții serviciului (prin exporturi) și la furnizorii de servicii (prin importuri).

Vizualizarea detaliilor modului SCA

Detaliile legate de modulul de servicii pe care le puteți vizualiza depind de modulul SCA. Acestea includ următoarele atribute.

- Nume modul SCA
- Descriere modul SCA
- Nume asociat aplicației
- Informații legate de versiunea modului SCA, dacă modulul are mai multe versiuni
- Importuri module SCA:
 - Interfețele de import sunt definiții prezentare pe scurt care descriu modul în care un modul SCA accesează un serviciu.
 - Legările de import sunt definiții concrete care specifică mecanismul fizic prin care un modul SCA accesează un serviciu. De exemplu, folosirea SOAP/HTTP.
- Exporturi modul SCA:
 - Interfețele de export sunt definiții prezentare pe scurt care descriu modul în care solicitantii serviciului accesează un modul SCA.
 - Legările de export sunt definiții concrete care specifică mecanismul fizic prin care un solicitant al serviciului accesează un modul SCA, și în mod indirect, un serviciu.
- Proprietățile modului SCA

Importurile și legările de import

Importurile definesc interacțiuni între modulele SCA și furnizorii de servicii. Modulele SCA folosesc importuri pentru a permite componentelor să acceseze servicii externe (servicii care se află în afara modului 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 țintele lor intenționate.

Interfețele și legările

Importul unui modul SCA are nevoie de cel puțin o interfață și are o singură legare.

- Interfețele de import sunt definiții abstracte care definesc un set de operațiuni folosind Web Services Description Language (WSDL), și limbaj XML pentru descrierea serviciilor Web. Un modul SCA poate avea mai multe interfețe de import.
- Legările de import sunt definiții 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 legări de import:

- Legările SCA conectează modulele SCA cu alte module SCA. De asemenea, legările SCA sunt menționate și ca legări implicite.
- Legările Serviciu Web permit componentelor să invoce servicii web. Protocoalele suportate sunt SOAP1.1/HTTP, SOAP1.2/HTTP, și SOAP1.1/JMS.

Puteți 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 interacțiunea cu serviciile folosind documente sau legări literale RPC și care folosesc handler-e JAX-WS pentru a personaliza invocarea. Este furnizată o legare SOAP1.1/HTTP separată pentru a permite interacțiunea cu serviciile care folosesc o legare codată RPC sau acolo unde există o cerință pentru folosirea handler-elor JAX-RPC pentru a personaliza invocările.
- Legările HTTP vă permit să accesați aplicații folosind protocolul HTTP.

- Legările de import EJP (Enterprise JavaBeans) permit componentelor SCA să invoce servicii oferite de logica operațională Java EE care rulează pe un server Java EE.
- Legările EIS (Enterprise information system) asigură conectivitatea între componente SCA și un EIS extern. Această comunicație 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 și HTTP sau HTTPS. Clasa JMS Message și cele cinci subtipuri ale sale (Text, Bytes, Object, Stream și Map) sunt acceptate automat.
- Legările JMS interoperabilitate cu furnizori JMS terță parte care se integrează cu WebSphere Application Server folosind JMS ASF (Application Server Facility).
- Legările JMS MQ WebSphere permit interoperabilitatea cu furnizorii JMS bazați pe MQ WebSphere. Clasa JMS Message și cele cinci subtipuri ale sale (Text, Bytes, Object, Stream și Map) sunt acceptate automat. Dacă doriți să utilizați WebSphere MQ pe post de furnizor JMS, folosiți legările JMS MQ WebSphere.
- Legările MQ WebSphere permit interoperabilitatea cu furnizorii MQ WebSphere. Puteți folosi legări WebSphere MQ doar împreună cu managerii cozii aflați la distanță prin intermediul unei conexiuni client WebSphere MQ; nu le puteți folosi cu managerii locali ai cozii. Folosiți legări WebSphere MQ dacă doriți să comunicați cu aplicațiile 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 invocați un serviciu la un alt punct final, prin orice legare de import suportată. Invocarea dinamică a serviciilor vă permite de asemenea să invocați 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 și configurația sa pentru invocarea dinamică. Importul folosit pentru invocarea dinamică poate fi legat de componenta de apelare, sau poate fi selectat dinamic la runtime.

Pentru serviciul Web și invocații SCA, se poate face o invocare dinamică fără import, cu protocolul și configurarea deduse din URL-ul punct final. Tipul țintă 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 URL HTTP sau JMS indică în mod implicit invocări ale serviciului web; pentru aceste URL-uri este posibilă furnizarea unor valori de tip de legare suplimentare care să indice faptul că URL-ul reprezintă o invocare a modului de legare a unui HTTP sau JMS.
- Pentru un URL HTTP al unui serviciu web, implicit este folosirea SOAP 1.1, și se poate specifica o valoare tip de legare care să indice folosirea SOAP 1.2.

Exporturile și legările de export

Exporturile definesc interacțiuni între modulele SCA și solicitanții 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 solicitanții serviciului.

Interfețele și legările

Un export pentru un modul SCA are nevoie de cel puțin o interfață.

- Interfețele de export sunt definiții abstracte care definesc un set de operațiuni folosind Web Services Description Language (WSDL), și limbaj XML pentru descrierea serviciilor Web. Un modul SCA poate avea mai multe interfețe de export.
- Legările de export sunt definiții concrete care specifică mecanismul fizic folosit de solicitanții serviciului pentru a accesa un serviciu. De obicei, exportul unui modul SCA are specificată o singură legare. Un export fără legări specificate este interpretat de mediul runtime ca un export cu o legare SCA.

Legările de export suportate

IBM Business Process Manager suportă următoarele legări de export:

- Legările SCA conectează modulele SCA cu alte module SCA. De asemenea, legările SCA sunt menționate și ca legări implicite.
- Legările Serviciu web permit exporturi care pot fi invocate ca și servicii web. Protocoalele suportate sunt SOAP1.1/HTTP, SOAP1.2/HTTP, și SOAP1.1/JMS.
Puteți folosi o legare SOAP1.1/HTTP sau SOAP1.2/HTTP bazată pe JAX-WS (Java API for XML Web Services), ceea ce permite interacțiunea cu serviciile folosind documente sau legări de literal RPC și care folosesc handler-e JAX-WS pentru a personaliza invocările. Este furnizată o legare SOAP1.1/HTTP separată pentru a permite interacțiunea cu serviciile care folosesc o legare codată RPC sau acolo unde există o cerință 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 operațională Java EE să poată invoca componente SCA care altfel le-ar fi indisponibile.
- Legările EIS (Enterprise information system) asigură conectivitatea între componente SCA și un EIS extern. Această comunicație 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 și HTTP sau HTTPS. Clasa JMS Message și cele cinci subtipuri ale sale (Text, Bytes, Object, Stream și Map) sunt acceptate automat.
- Legările JMS interoperabilitate cu furnizori JMS terță parte care se integrează cu WebSphere Application Server folosind JMS ASF (Application Server Facility).
- Legările JMS MQ WebSphere permit interoperabilitatea cu furnizorii JMS bazați pe MQ WebSphere. Clasa JMS Message și cele cinci subtipuri ale sale (Text, Bytes, Object, Stream și Map) sunt acceptate automat. Dacă doriți să utilizați WebSphere MQ pe post de furnizor JMS, folosiți legările JMS MQ WebSphere.
- Legările MQ WebSphere permit interoperabilitatea cu furnizorii MQ WebSphere. Puteți utiliza o conexiune de la distanță (sau client) pentru a vă conecta la un manager de coadă MQ aflat pe o mașină la distanță. O conexiune locală (sau legări) este o conexiune directă la WebSphere MQ. Aceasta poate fi utilizată doar pentru o conexiune la un manager al cozii MQ de pe aceeași mașină. WebSphere MQ va permite ambele tipuri de conexiuni, dar legările MQ suportă doar conexiunea "de la distanță" (sau "client").

Module de mediere

Modulele de mediere sunt module SCA (Service Component Architecture) care pot modifica formatul, conținutul sau ținta cererilor serviciului.

Modulele de mediere operează asupra mesajelor care sunt în zbor între solicitanții și furnizorii serviciului. Puteți ruta mesaje către diferiți furnizori de servicii și puteți modifica conținutul sau forma mesajului. Modulele de mediere pot oferi funcții cum ar fi înregistrarea mesajului și procesarea erorii care sunt adaptate cerințelor dumneavoastră.

Puteți modifica anumite aspecte ale modulelor de mediere din consola administrativă fără a fi nevoie să reimplementați modulul .

Componentele modulelor de mediere

Modulele de mediere cuprind următoarele elemente:

- Importuri, care definesc interacțiunea între modulele SCA și furnizorii de servicii. Acestea permit modulelor SCA să apeleze servicii externe ca și cum acestea ar fi locale. Puteți vizualiza importurile modulelor de mediere și să modificați legarea.
- Exporturi, care definesc interacțiunea între modulele SCA și solicitanții serviciului. Acestea permit unui modul SCA să ofere un serviciu și să definească interfețele externe (puncte de acces) pentru un modul SCA. Puteți vizualiza exporturile modulelor de mediere.

- Componente SCA, care construiesc blocuri pentru modulele SCA precum module de mediere. Aveți posibilitatea să creați și să particularizați module SCA și componente în mod grafic, folosind Integration Designer. După ce ați implementat un modul de mediere puteți personaliza anumite aspecte ale acestuia din consola administrativă fără să fiți nevoiți să reimplementați modulul.

De obicei, module de mediere conține un anumit tip de componentă SCA numită *componentă a fluxului de mediere*. Componentele fluxului de mediere definesc fluxurile de mediere.

Componenta unui flux de mediere poate conține una, un anumit număr de primitive de mediere sau niciuna. IBM Business Process Manager suportă un set livrat de primitive de mediere care oferă funcționalitate pentru rutarea și transformarea mesajelor. Pentru flexibilitate suplimentară a primitivei de mediere, utilizați primitiva Mediere personalizată pentru apelarea logicii personalizate.

Scopul unui modul de mediere care nu conține 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 transformată în SOAP/HTTP înainte de a fi trimisă mai departe.

Notă: Aveți posibilitatea să vizualizați și să faceți anumite modificări asupra modulelor de mediere din IBM Business Process Manager. Totuși, nu puteți vizualiza sau modifica componentele SCA din interiorul unui modul din IBM Business Process Manager. Utilizați Integration Designer pentru a personaliza componentele SCA.

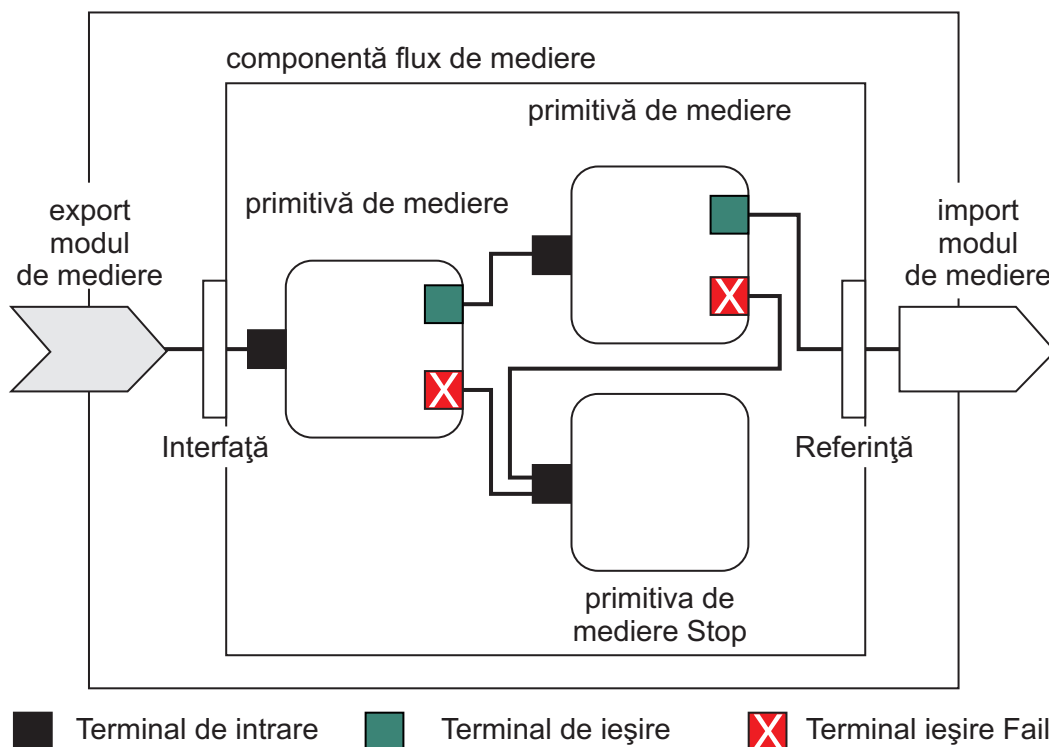


Figura 88. Exemplu simplificat de modul mediere. Modulul de mediere conține o singură componentă a fluxului de mediere, care conține primitive de mediere.

- Proprietăți

Primitivele de mediere au proprietăți, unele dintre ele pot fi afișate în consola administrativă sub formă de proprietăți suplimentare ale unui modul SCA.

Pentru ca proprietățile primitivelor de mediere să fie vizibile din consola administrativă IBM Business Process Manager, dezvoltatorul care se ocupă cu integrarea trebuie să le promoveze. Anumite proprietăți se pretează a fi configurate administrativ și Integration Designer le descrie ca proprietăți ce pot fi promovate deoarece acestea pot fi promovate din cercul de integrare în ciclul administrativ. Alte proprietăți nu sunt potrivite pentru configurare

administrativă deoarece modificarea acestora poate afecta fluxul de mediere în așa fel încât modulul de mediere va trebui să fie reimplementat. Integration Designer listează proprietățile pe care le puteți alege pentru a promova în proprietățile promovate ale unei primitive de mediere.

Puteți folosi consola administrativă IBM Business Process Manager pentru a modifica valoarea proprietăților promovate fără a trebui să reimplementați un modul de mediere sau să reporniți serverul sau modulul.

În general, fluxurile de mediere utilizează imediat modificările proprietății. Cu toate acestea, dacă apar modificări asupra proprietății într-o celulă pentru managerul de implementare, acestea au efect în fiecare nod pe măsură ce acestea sunt sincronizate. De asemenea, fluxurile de mediere care sunt în zbor continua să utilizeze valorile anterioare.

Notă: Din consola administrativă, aveți posibilitatea să modificați doar valorile proprietății, nu grupurile acesteia, numele sau tipurile. Dacă doriți să modificați grupurile de proprietăți, nume sau tipuri, trebuie să utilizați Integration Designer.

- Un modul de mediere sau biblioteca dependentă poate defini, de asemenea, subfluxuri. Un subflux 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, și sunt implementate în general pe IBM Business Process Manager înăuntrul unui fișier EAR (enterprise archive).

Aveți posibilitatea să modificați valoarea proprietăților promovate în momentul implementării.

Puteți exporta un modul de mediere din Integration Designer, și să determinați ca Integration Designer să împacheteze modulul de mediere într-un fișier JAR (Java archive), iar acest fișier JAR într-un fișier EAR. Apoi, aveți posibilitatea să implementați apoi fișierul EAR prin instalarea unei noi aplicații din consola administrativă.

Modulele de mediere pot fi gândite ca o singură entitate. Totuși, modulele SCA sunt definite de un număr de fișiere XML memorate într-un fișier JAR.

Exemplu de fișier EAR, care conține un modul de mediere

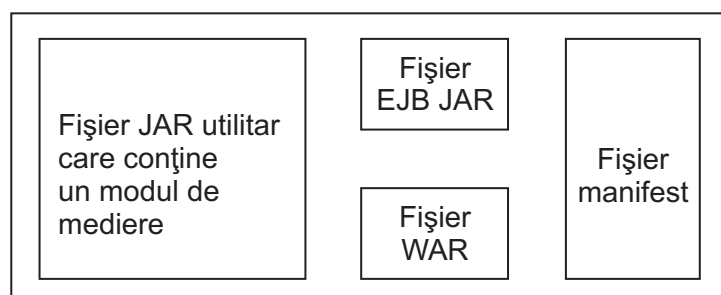


Figura 89. Un exemplu simplificat de fișier EAR care conține un modul de mediere. Fișierul EAR conține JAR-uri. Fișierul JAR utilitar conține un modul de mediere.

Primitive de mediere

Componentele fluxului de mediere operează asupra fluxurilor de mesaje între componentelor serviciului. Capabilitățile componente 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 și unul pentru răspuns.

IBM Business Process Manager suportă un set livrat de primitive de mediere, care implementează capabilitățile standard de mediere pentru modulele de mediere sau pentru modulele implementate în IBM Business Process Manager. Dacă aveți nevoie de anumite capabilități de mediere, puteți să vă dezvoltați propriile primitive personalizate de mediere.

O primitivă de mediere definește o operație 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 operație de tip “out” care trimite mesaje la o altă componentă sau la un alt modul.

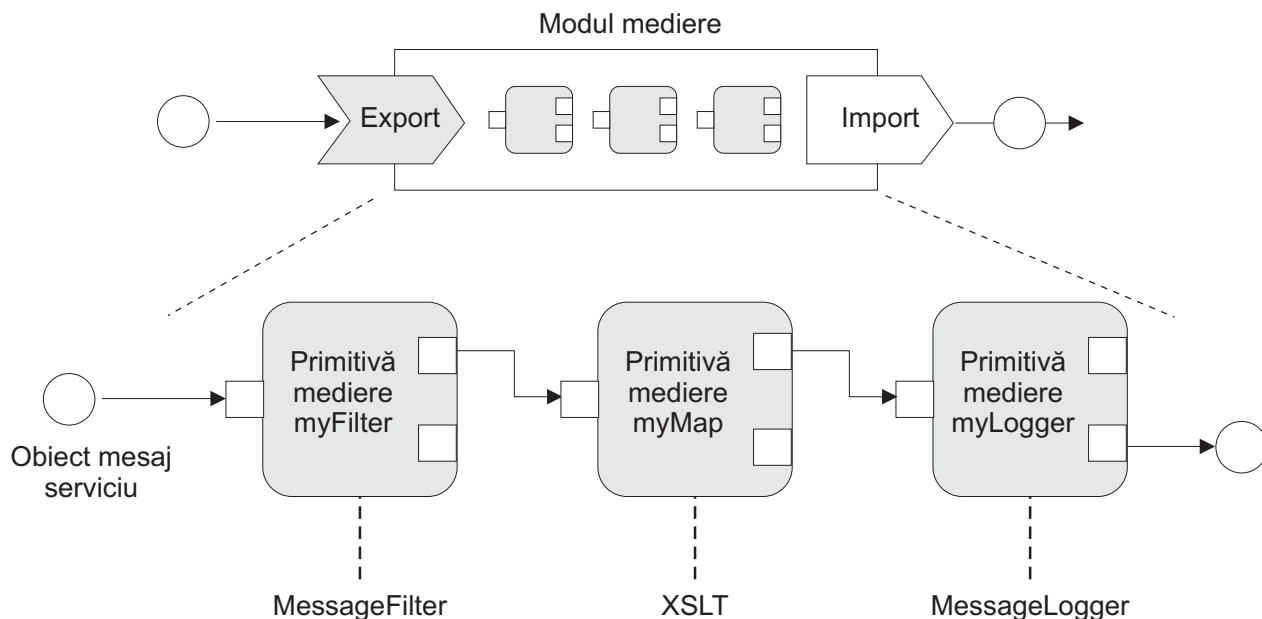


Figura 90. Modulul de mediere care conține trei primitive de mediere

Puteți folosi Integration Designer pentru a configura primitivele de mediere și pentru a le seta proprietățile. Unele dintre aceste proprietăți pot fi făcute vizibile pentru administratorul de runtime prin promovarea acestora. Oricare dintre proprietățile primitivelor de mediere care pot fi promovate pot fi de asemenea proprietăți dinamice. O proprietate dinamică poate fi înlocuită în momentul execuției prin folosirea unui fișier de politică.

Integration Designer vă permite de asemenea să modelați grafic și să asamblați componentele fluxului de mediere din cadrul primitivei de mediere, și să asamblați module de mediere sau module din cadrul componentelor fluxului de mediere. Consola administrativă se referă la modulele de mediere și la modulele ca la module SCA.

Integration Designer permite de asemenea definirea de subfluxuri în module sau a bibliotecilor lor dependente. Un subflux poate conține orice primitivă de mediere, cu excepția primitivei Policy Resolution. Un subflux este invocat din fluxul de cerere sau de răspuns, sau dintr-un alt subflux folosind primitiva de mediere Subflux. Proprietățile promovate într-un subflux prin primitivele de mediere sunt expuse ca proprietăți 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.

- Definește transformări pentru mesaje folosind o mapare pentru obiecte business care poate fi refolosită.

- Vă permite să definiți transformări pentru mesaje în mod grafic folosind editorul pentru maparea cu obiecte business.
- Poate modifica conținutului unui mesaj.
- Poate transforma tipul unui mesaj de intrare într-un tip diferit de mesaj de ieșire.

Custom Mediation

Vă permite să implementați 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. Puteți crea transformări complexe și tipare de rutare astfel:

- Crearea de cod Java.
- Crearea propriilor proprietăți.
- Adăugarea de terminale noi.

Puteți apela un serviciu dintr-o primitivă de mediere personalizată, dar primitiva de mediere Service Invoke este proiectată să apeleze servicii și să asigure funcționalități suplimentare, precum reîncercarea.

Data Handler

Vă permite să modificați 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 șir text dintr-un obiect JMS Text Message, într-o structură de tip Obiect business și din nou înapoi. Această mediere este frecvent utilizată pentru a:

- Transformarea unei secțiuni a mesajului de intrare dintr-o structură definită în alta - un exemplu de acest fel ar fi cazul în care SMO include o valoare șir care este delimitată prin virgulă și vreți să o parsați într-un Obiect business specific.
- Modificarea tipului de mesaj - un exemplu ar fi când un export JMS a fost configurat să folosească o legare de date tipizată de bază JMS și în modulul de mediere dezvoltatorul de integrare decide extinderea conținutului la o structură BO (obiect business) specifică.

Database Lookup

Modifică mesaje, utilizând informații dintr-o bază de date ce sunt furnizate de utilizator.

- Trebuie să configurați o bază de date, o sursă de date, precum și orice setări de autentificare necesare serverului utilizate de primitiva de mediere Database Lookup. Pentru a face acest lucru, ajutați-vă de consola administrativă.
- Primitiva de mediere Database Lookup poate citi dintr-o singură tabelă.
- Coloana index specificată trebuie să conțină o valoare unică.
- Datele din coloanele ce conțin 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.

- Informațiile 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 distanță.
- Faceți modificările asupra registrului din consola administrativă WSRR.
- IBM Business Process Manager trebuie să știe ce registru să folosească, și de aceea, trebuie să creați definiții de acces WSRR folosind consola administrativă IBM Business Process Manager.

Event Emitter

Îmbunătățește monitorizarea prin permiterea trimiterii evenimentelor din interiorul unei componente a fluxului de mediere.

- Puteți suspenda acțiunea de mediere prin debifarea casetei de bifare.
- Puteți vizualiza evenimente Event Emitter utilizând browser-ul Common Base Events în IBM Business Process Manager.
- Ar trebui să trimiteți evenimente doar către un singur punct important într-un flux de mediere, din motive de performanță.

- Aveți posibilitatea să definiți părțile mesajului conținut de eveniment.
- Evenimentele sunt trimise în formatul Common Base Events și sunt trimise la un server Common Event Infrastructure.
- Pentru a utiliza pe deplin informațiile legate de Event Emitter, consumatorii de evenimente trebuie să înțeleagă structura evenimentelor CBE (Common Base Events). Evenimentele CBE au o schemă generală, dar acest lucru nu modelează datele specifice aplicației care sunt conținute în elementele extinse de date. Pentru a modela elementele extinse de date, unelte Integration Designer generează un fișier cu definiții pentru catalogul de evenimente Common Event Infrastructure pentru fiecare dintre primitivele de mediere Event Emitter configurate. Fișierele 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 faceți referire la fișierele cu definiții pentru catalogul de evenimente atunci când creați aplicații care consumă evenimente Event Emitter.
- Puteți specifica o altă monitorizare din IBM Business Process Manager. De exemplu, puteți controla ca evenimentele să fie emise din importuri și exporturi.

Fail Oprește o anumită cale din flux și generează o excepție.

Fan In Ajută la agregarea (combinarea) mesajelor.

- Poate fi folosită doar în combinație cu primitiva de mediere Fan Out.
- Împreună, primitivele de mediere Fan Out și Fan In permit agregarea datelor într-un singur mesaj de ieșire.
- Primitiva de mediere Fan In primește mesaje până când se ajunge la un punct de decizie, moment în care este trimis la ieșire un mesaj.
- Contextul partajat ar trebui folosit pentru reținerea datelor agregate.

Fan Out

Ajută la divizarea și agregarea (combinarea) mesajelor.

- Împreună, primitivele de mediere Fan Out și Fan In permit agregarea datelor într-un singur mesaj de ieșire.
- În modul iterativ, primitiva de mediere Fan Out vă permite să iterați printr-un printr-un singur mesaj de intrare care conține un element repetitiv. Pentru fiecare apariție a elementului repetitiv este trimis un mesaj.
- Contextul partajat ar trebui folosit pentru reținerea datelor agregate.

HTTP Header Setter

Oferă un mecanism pentru gestionarea anteturilor în mesajele HTTP.

- Poate crea, seta, copia sau șterge anteturile mesajelor HTTP.
- Poate seta mai multe acțiuni pentru a modifica mai multe anteturi HTTP.

Mapare

Transformă mesaje.

- Vă permite să realizați transformări XSL (Extensible Stylesheet Language) sau transformări Business Object Map.
- Puteți 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 conținutului mesajelor.

- Poate modifica, adăuga sau șterge elemente din mesaj.
- Nu modifică tipul de mesaj.
- Datele din coloanele ce conțin 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 funcție de conținutul acestora.

- Puteți suspenda acțiunea de mediere prin debifarea casetei de bifare.

Message Logger

Înregistrează mesajele într-o bază de date relațională sau prin intermediul propriului jurnalizator personalizat. Mesajele sunt memorate sub formă de XML, și de aceea, datele pot fi procesate după aceea de aplicațiile care suportă XML.

- Puteți suspenda acțiunea de mediere prin debifarea casetei de bifare.
- Schema bazei de date relaționale (structura tabeli) este definită de IBM.
- În mod implicit, primitiva de mediere Message Logger folosește baza de date Common. Mediul runtime mapează sursa de date la **jdbc/mediation/messageLog** pe baza de date Comună.
- Puteți seta clasele de implementare Handler pentru a personaliza comportamentul jurnalizatorului personalizat. Opțional, puteți 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 șterge anteturile mesajelor MQ.
- Poate seta mai multe acțiuni pentru a modifica mai multe anteturi MQ.

Policy Resolution

Permite configurarea dinamică a cererilor, prin căutarea punctelor finale ale serviciului și a fișierelor de politică asociate, într-o magazie.

- Aveți posibilitatea să utilizați un fișier de politică pentru a înlocui dinamic proprietățile promovate ale altor primitive de mediere.
- Informațiile punctului final al serviciului și informațiile legate de politică sunt extrase dintr-un WSRR (WebSphere Service Registry and Repository). Registrul WSRR poate fi local sau poate fi aflat la distanță.
- Faceți modificările asupra registrului din consola administrativă WSRR.
- IBM Business Process Manager trebuie să știe ce registru să folosească, și de aceea, trebuie să creați definiții 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 aștepte până la sfârșitul unui flux de mediere și folosirea mecanismului callout.

- Dacă serviciul returnează un defect, puteți reîncerca același serviciu sau să apălați 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 combinație cu alte primitive de mediere pentru medieri mai complexe.

Set Message Type

În timpul dezvoltării de integrare, vă permite să tratați câmpurile mesajului care sunt tipizate slab, ca și pe cele care sunt tipizate tare. Un câmp este tipizat slab dacă poate conține mai multe tipuri de date. Un câmp este tipizat tare dacă tipul și structura sa internă sunt cunoscute.

- La runtime, medierea Setare tip mesaj permite verificarea potrivirii între conținutul unui mesaj și tipurile de date pe care le așteptați.

SOAP Header Setter

Oferă un mecanism pentru gestionarea anteturilor în mesajele SOAP.

- Poate crea, seta, copia sau șterge anteturile mesajelor SOAP.
- Poate seta mai multe acțiuni pentru a modifica mai multe anteturi SOAP.

Stop Oprește o anumită cale din flux fără a genera o excepție.

Type Filter

Vă permite să direcționați mesajele pe o cale diferită a fluxului, în funcție de tipul acestora.

Extragere WebSphere eXtreme Scale

Puteți extrage informații dintr-un mediu cache server eXtreme Scale.

- Puteți căuta valori în cache și le puteți păstra ca elemente în mesaj folosind un index.

- Combinând primitivele de mediere Stocare și Extragere eXtreme Scale, puteți memora în cache răspunsul de la un sistem back-end. Cererile viitoare nu vor necesita acces la acel sistem de back-end.
- Trebuie să creați definiții eXtreme Scale utilizând consola administrativă WebSphere ESB, astfel încât puteți specifica serverul eXtreme Scale de utilizat.

Stocare WebSphere eXtreme Scale

Puteți stoca informații într-un mediu cache server eXtreme Scale.

- Puteți stoca informații într-un cache eXtreme Scale utilizând un index sau un obiect.
- Combinând primitivele de mediere eXtreme Scale Stocare și Extragere, puteți folosi primitiva de mediere Stocare pentru stocarea datelor în memoria cache și puteți folosi primitiva de mediere Extragere pentru a extrage date stocate anterior în memoria cache.
- Trebuie să creați definiții eXtreme Scale utilizând consola administrativă WebSphere ESB, astfel încât puteți specifica serverul eXtreme Scale de utilizat.

Rutarea dinamică

Puteți 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 și selecția 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 execuției are capacitatea de a ruta mesajele de cerere și 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 informații dintr-un registru, o bază de date, sau cu informații 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 execuției 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. Dezvoltorii 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 execuției), astfel încât administratorul din timpul rulării să o poată seta. Aveți posibilitatea să vizualizați proprietățile modulului în fereastra **Proprietăți modul**. Pentru a vedea fereastra, faceți clic pe **Aplicații > Module SCA > Proprietăți Modul**. Dezvoltorul care se ocupă cu integrarea oferă proprietăților promovate aliasuri, iar acestea sunt numele afișate în consola administrativă.

Registru

Puteți folosi WSRR (IBM WebSphere Service Registry and Repository) pentru a memora informațiile legate de punctul final al serviciului, iar apoi să creați module SCA pentru a extrage puncte finale din magazia WSRR.

Atunci când dezvoltați module SCA, folosiți primitiva de mediere Endpoint Lookup pentru a permite unui flux de mediere să interogheze un registru WSRR pentru a obține un punct final pentru serviciu sau un set de puncte finale pentru serviciu. Dacă un modul SCA 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

Puteți utiliza politici de mediere pentru a controla fluxurile de mediere dintre solicitanții serviciului și furnizorii de servicii.

Puteți 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 aveți module SCA (Service Component Architecture) dorite și documente ale politicii de mediere în registrul dumneavoastră WSRR.

Cum se face atașarea unei politici de mediere la o cerere de serviciu

Atunci când dezvoltați un modul SCA care trebuie să facă uz de o politică de mediere, trebuie să includeți în fluxul de mediere o primitivă de mediere Policy Resolution. În timpul rulării, primitiva de mediere Policy Resolution obține informații despre politica de mediere din registru. Prin urmare, un modul SCA trebuie să conțină o componentă flux de mediere pentru a suporta controlul politicii de mediere a cererilor de servicii.

În registru, puteți atașa una sau mai multe politici de mediere la un modul SCA sau la un serviciu țintă utilizat de către modulul SCA. Politicile de mediere atașate pot fi utilizate (sunt în domeniu) pentru toate mesajele serviciului procesate de către acel modul SCA. Politicile de mediere pot avea atașamente de politică care definesc condiții. Condițiile 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 memorați, accesați și să gestionați informațiile legate de punctele finale ale serviciului și de politicile de mediere. Puteți utiliza WSRR pentru a face ca aplicațiile serviciului dvs. să fie mult mai dinamice și mult mai adaptabile la modificările aduse condițiilor de business.

Introducere

Fluxurile de mediere pot folosi WSRR pe post de mașină de căutare dinamică, oferind informații legate de punctele finale sau de politicile de mediere ale serviciului.

Pentru a configura accesul la WSRR, creați documente de definiție WSRR folosind consola administrativă. Alternativ, aveți posibilitatea să utilizați comenzile de administrare WSRR din clientul script wsadmin. Definițiile WSRR și proprietățile acestora de conexiune reprezintă mecanismul de conectare la o instanță a unui registru, și de extragere a punctului final sau a politicii de mediere pentru un serviciu.

Puncte finale pentru serviciu

Puteți folosi WSRR pentru a memora informații despre serviciile pe care le folosiți deja, pentru cele pe care plănuți să le folosiți sau pentru cele de care vreți să fiți conștient. Aceste servicii pot fi în sistemele dvs, sau în ale sistemelor. De exemplu, aplicația ar putea folosi WSRR la localizarea celui mai corespunzător serviciu care îi satisface nevoile funcționale și de performanță.

Atunci când dezvoltați un modul SCA care are nevoie să acceseze punctele finale ale serviciului din WSRR, trebuie să includeți o primitivă de mediere Endpoint Lookup în fluxul de mediere. În momentul rulării, primitiva de mediere Endpoint Lookup obține punctele finale ale serviciului din registru.

Politici de mediere

De asemenea, puteți folosi WSRR pentru a memora informații legate de politica de mediere. Politicile de mediere vă pot ajuta să controlați cererile de servicii prin suprascrierea dinamică a proprietăților modulului. Dacă WSRR conține politici de mediere care sunt atașate la un obiect care reprezintă fie modulul dvs. SCA, fie serviciul dvs. țintă, atunci politicile de mediere ar putea suprascrie proprietățile modulului. Dacă doriți ca în diferite contexte să se aplice politici de mediere diferite, puteți crea condiții pentru aceste politici.

Notă: Politicile de mediere sunt preocupate de control fluxurilor de mediere, nu de securitate.

Atunci când dezvoltați un modul SCA care trebuie să facă uz de o politică de mediere, trebuie să includeți în fluxul de mediere o primitivă de mediere Policy Resolution. În momentul rulării, primitiva de mediere Policy Resolution obține informațiile legate de politica de mediere din registru.

WebSphere eXtreme Scale

Utilizând produsul WebSphere eXtreme Scale (eXtreme Scale) puteți asigura un sistem de memorare în cache pe care îl puteți integra cu o aplicație IBM Business Process Manager. Utilizând eXtreme Scale cu IBM Business Process Manager puteți îmbunătăți timpul de răspuns și fiabilitatea serviciului și asigura funcționalitate de integrare suplimentară.

eXtreme Scale acționează ca o grilă de date elastică, scalabilă, în memorie. Grila de date memorează în cache, partiționează, replică și gestionează dinamic datele aplicației și logica operațională pe mai multe servere. Cu eXtreme Scale, puteți, de asemenea, să obțineți calitate a serviciilor precum integritate tranzacțională, disponibilitate înaltă și timpi de răspuns predictabili.

Puteți utiliza fluxuri de mediere pentru a accesa funcția de memorare în cacheeXtreme Scale inclusiv primitivile de mediere WebSphere eXtreme Scale din fluxul dumneavoastră. La dezvoltarea unui modul SCA (Service Component Architecture) care are nevoie să stocheze informații într-o memorie cacheeXtreme Scale, trebuie să includeți primitiva de mediere WebSphere eXtreme Scale Store în fluxul de mediere. Dacă vreți să extrageți informații dintr-o memorie cache eXtreme Scale, trebuie să includeți primitiva de mediere WebSphere eXtreme Scale Retrieve. Combinând cele două primitive de mediere într-un flux de mediere, puteți 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 creați o definiție WebSphere eXtreme Scale utilizând consola administrativă. Alternativ, puteți utiliza comenzile de administrare WebSphere eXtreme Scale din clientul de scriptare wsadmin. O definiție eXtreme Scale este mecanismul utilizat de primitivile de mediere WebSphere eXtreme Scale Retrieve și Store pentru conectarea la un servereXtreme Scale.

Clienți ai serviciului de mesaje

Clienții pentru serviciul de mesaje sunt disponibili pentru C/C++ și .NET pentru a permite aplicațiilor 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 același set de interfețe ca și API-ul JMS (Java Message Service). Clieții serviciului de mesaje pentru C/C++ conține două implementări ale XMS, una pentru aplicațiile C și una pentru aplicațiile by C++. Message Service Client for .NET conține o implementare complet gestionată a XMS, care poate fi folosită de orice limbaj compatibil cu .NET.

Puteți obține clienți ai serviciului de mesaje (Message Service Clients) 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

Puteți obține clienți ai serviciului de mesaje (Message Service Clients) 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.

Puteți instala și folosi și suportul pentru clientul Java EE de la WebSphere Application Server Network Deployment, inclusiv client servicii web, EJB Client și JMS Client.

