IBM Business Process Manager
Verze 8, vydání 0

Přehled produktu IBM Business Process Manager

Knihy ve formátu PDF a Informační centrum

Jsou poskytovány Knihy ve formátu PDF pro tisk a čtení offline. Nejnovější informace naleznete v online Informačním centru.

Jako sada obsahují knihy ve formátu PDF stejný obsah jako Informační centrum. Některé odkazy v PDF knihách byly upraveny pro použití v Informačních centrech a nemusí správně fungovat.

Dokumentace ve formátu PDF je k dispozici během čtvrt roku po vydání hlavní verze Informačního centra, například verze 7.0 nebo verze 7.5.

Dokumentace PDF je aktualizována méně často než Informační centrum, ale častěji než publikace Redbooks. Obecně jsou knihy ve formátu PDF aktualizovány po nahromadění dostatečného množství změn.

Obsah

Kapitola 2. Další informace o klíčových konceptech 167

Kapitola 1. Začínáme s produktem IBM Business Process Manager

Porozumějte tomu, jaké schopnosti může produkt IBM® Business Process Manager nabídnout pro řízení BPM a jak mezi sebou souvisí různé fáze řízení BPM, jako např. vytváření a implementace komponent Process Application.

Komponenta Process Application je základním kontejnerem procesů a jejich komponent v produktu IBM Business Process Manager. Návrháři procesů vytváří komponenty Process Application v redakčních prostředích. Mohou do nich zahrnovat služby, úlohy a artefakty, které jsou potřeba pro podporu vlastního provádění.

Rozšířené služby integrace jsou implementovány do prostředí IBM Integration Designer a přidruženy ke komponentám Process Application. Z centra Process Center se komponenty Process Application implementují na server Process Server, který je součástí běhového prostředí procesů produktu IBM Business Process Manager.

Podobně mohou automatické procesy vytvořené v produktu Integration Designer používat toky lidských aktivit vyvinuté v prostředí IBM Process Designer.

Souhrn informací o verzi

Seznamte se s novinkami v produktu IBM Business Process Manager verze 8.0 a získejte přístup k užitečným prostředkům, které vám pomohou začít s různými oblastmi produktu.

Novinky

Produkt IBM BPM verze 8.0 poskytuje mnoho nových funkcí a různých rozšíření a zlepšení stávajících schopností. Viz úplný seznam nových funkcí a rozšířených schopností produktu IBM BPM verze 8.0.

"Novinky v produktu IBM Business Process Manager V8.0" na stránce 12

Vylepšení

Nejvýznamnější vylepšení a opravy stávajících funkcí v produktu IBM BPM verze 8.0.

Vysoký výkon a rozšiřitelnost v operačním systému z/OS

Prostředí zEnterprise nabízí vysoký výkon aplikací díky shromažďování aplikací WebSphere a správců transakcí v blízkosti dat (DB2 z/OS) a transakcí (CICS, IMS nebo WebSphere MQ). Výhody využití aplikací pod kontrolou systému z/OS přináší klientským aplikacím i podnikům značnou přidanou hodnotu. Vysoký výkon v systémech z/OS podporují vylepšení v těchto oblastech:

- Konzistence databází v oblasti konvencí pojmenování.
- Podpora identity podprocesů pro zdroje dat generované konfigurací.
- Odebrání konfigurace databáze infrastruktury CEI (Common Event Infrastructure) v samostatných profilech a scénářích síťové implementace.
- Podpora konfigurace databáze produktu Business Process Choreographer pro samostatné profily v jiných databázích než DB2.
- Generování souborů DDL (Data Definition Language) infrastruktury CEI nástrojem pro návrh databází.
- Schopnost zpracovat SQL a databázové seskupení souborů SQL.

Aspekty konfigurace síťové implementace

Rozšířená podpora WCT (WebSphere Customization Toolbox) pro scénáře instalace síťové implementace

Před úpravou prostředí mohou administrátoři a systémoví programátoři z/OS využít zjednodušený proces počáteční instalace a konfigurace. Zjednodušené konfigurace je dosaženo prostřednictvím těchto vylepšení:

- Podpora z/OS pro aplikace a data.
- Zvýrazněné a pečlivě zdokumentované role.
- Na stránkách zákazníka lze používat dokumentaci a skripty, které instalují klastrované prostředí BPM.
- Méně úloh rozepsaných v instalačních tabulkách.

Konfigurace síťové implementace pro operační systém z/OS

Zlepšená administrativní podpora pro spuštění a zastavení prostředí implementace

S pomocí akcí administrativní konzoly a příkazů wsadmin teď můžete spustit a zastavit klastrovanou topologii ve správném pořadí a bez chyb. V průběhu operací spuštění a zastavení je sdělován stav. Když jsou rozpoznány chyby, nabídne vylepšená tvorba sestav o chybách přístup k souborům protokolu, kde je uveden zasažený klastr nebo uzel a specifická chyba.

Spouštění a zastavování prostředí implementace

Vylepšené řízení opakovaných pokusů v systému

K dispozici jsou opakované automatické pokusy systému pro asynchronní vyvolání architektury SCA (Service Component Architecture) a přerušitelné procesy BPEL, které mají odstranit nekontrolovatelné opakované pokusy, jež vedou k problémům integrity dat. Byly přidány tyto schopnosti:

- Schopnost zakázat opakované pokusy pro celý systém BPM (SCA, BPC, vazby).
- Deklarativní schopnost v době návrhu označit na úrovni operací, kde má docházet k opakování, určením počtu opakovaných pokusů a intervalů mezi nimi.
- Administrativní schopnost upravit deklarovaná opakování (počet a interval) za běhu bez zastavení, spuštění, odinstalování nebo instalace.

Řízení systémových opakování

Nezbytné kontroly během instalace DB2 Express

Zlepšená instalace pomáhá vyhledat a napravit potíže během instalace vestavěné databáze DB2 Express.

- Výběr hesla odpovídá zásadám na úrovni operačního systému.
- U typické instalace se ověřuje výchozí ID uživatele a heslo používané pro instalaci produktu DB2 Express.
- U vlastní instalace není pro ID uživatele databáze DB2 žádné výchozí heslo stanoveno. Musíte zadat heslo a odpovídající potvrzení hesla pro dané ID uživatele.

- Instalační program ověří, zda již existuje instance databáze BPMINST. Pokud existuje, budete vyzváni k odstranění existující instance databáze, než budete pokračovat. Automatická kontrola také ověří, zda je volný port 50000, a pokud tomu tak není, vyzve vás k nápravě situace.

Příprava instalace a konfigurace softwaru

Interaktivní odinstalování DB2 Express
nyní můžete zrušit odebrání vestavěné databáze DB2 Express, pokud na ní závisí jiné instalace. Případně se můžete rozhodnout v rámci odinstalování produktu DB2 Express zcela odinstalovat instalaci databáze BPMINST.

Interaktivní odinstalování produktu IBM Business Process Manager

Vylepšená tvorba služeb a ošetření chyb
Další úroveň granularity umožňuje rozlišit mezi různými typy a závažností chyb a efektivněji spravovat cesty k výjimkám. Nyní je možné identifikovat a rozlišit deklarované výjimky a vygenerovat z nich poruchy, když jsou definovány pro službu AIS (Advanced Integration Service). Pojistná událost chyby identifikuje chyby, které nejsou deklarovány. Konzistentnější správa chyb nabízí tyto výhody:

- Zvyšuje úroveň schopnosti reagovat za provozu.
- Usnadňuje určování problémů.
- Zrychluje obnovu.
- Zvyšuje produktivitu vývoje.

Ošetřování chyb v BPD

Zamítnuté funkce

Produkt IBM BPM rozšiřuje schopnosti z předchozích verzí produktů WebSphere® Integration Developer, WebSphere Lombardi Edition, IBM Business Process Manager, WebSphere Process Server, WebSphere Enterprise Service Bus a dalších produktů IBM BPM.

Viz souhrn vlastností produktů, které jsou v produktu IBM BPM verze 8.0 zamítnuty a odebrány:

Zamítnuté a odebrané funkce produktu IBM Business Process Manager

Systémové požadavky

Před instalací jednotlivých produktů v sadě produktu IBM BPM verze 8.0 zkontrolujte systémové požadavky a ujistěte se, že jsou splněny.

IBM Business Process Manager Advanced
Podrobné systémové požadavky produktu IBM Business Process Manager Advanced

IBM Business Process Manager Standard
Podrobné systémové požadavky produktu IBM Business Process Manager Standard

IBM Business Process Manager Express
Podrobné systémové požadavky produktu IBM Business Process Manager Express

IBM Business Process Manager Tools & Add-Ons
Podrobné systémové požadavky produktu IBM Business Process Manager Tools & Add-Ons

IBM Integration Designer
Podrobné systémové požadavky produktu IBM Integration Designer

IBM Business Monitor
Podrobné systémové požadavky produktů IBM Business Monitor a WebSphere Business Monitor

Poznámky k verzi

Pročtěte si poznámky k verzi na webu podpory, kde najdete informace o omezeních a náhradních řešeních.

IBM Business Process Manager Advanced
http://www.ibm.com/support/search.wss?q=ibpma80relnotes

IBM Business Process Manager Standard
http://www.ibm.com/support/search.wss?q=ibpms80relnotes

IBM Business Process Manager Express
http://www.ibm.com/support/search.wss?q=ibpme80relnotes

IBM Integration Designer
http://www.ibm.com/support/search.wss?q=iid80relnotes

IBM Business Monitor
http://www.ibm.com/support/search.wss?q=mon80relnotes

Process Designer
http://www.ibm.com/support/search.wss?q=pd80relnotes

Business Space
http://www.ibm.com/support/search.wss?q=bsp80relnotes

Další prostředky

Použijte následující prostředky pro přístup ke komunitě IBM BPM a ke sdílení znalostí a prostředků.

Wikiweb komunity IBM Business Process Manager
Vyhledejte a sdílejte znalosti o produktu IBM Business Process and Decision Management s jinými odborníky a uživateli. Na wikiwebu IBM Business Process Manager Community najdete také obsah vytvořený komunitou kolem produktů WebSphere Lombardi Edition a Lombardi Teamworks.

Komunita IBM BPM

Výměna ukázek
Vyhledejte a sdílejte ukázkové aplikace, komponenty Toolkit a další kódy, které můžete použít ve svých řešeních IBM Business Process and Decision Management.

Domovská stránka pro výměnu ukázek

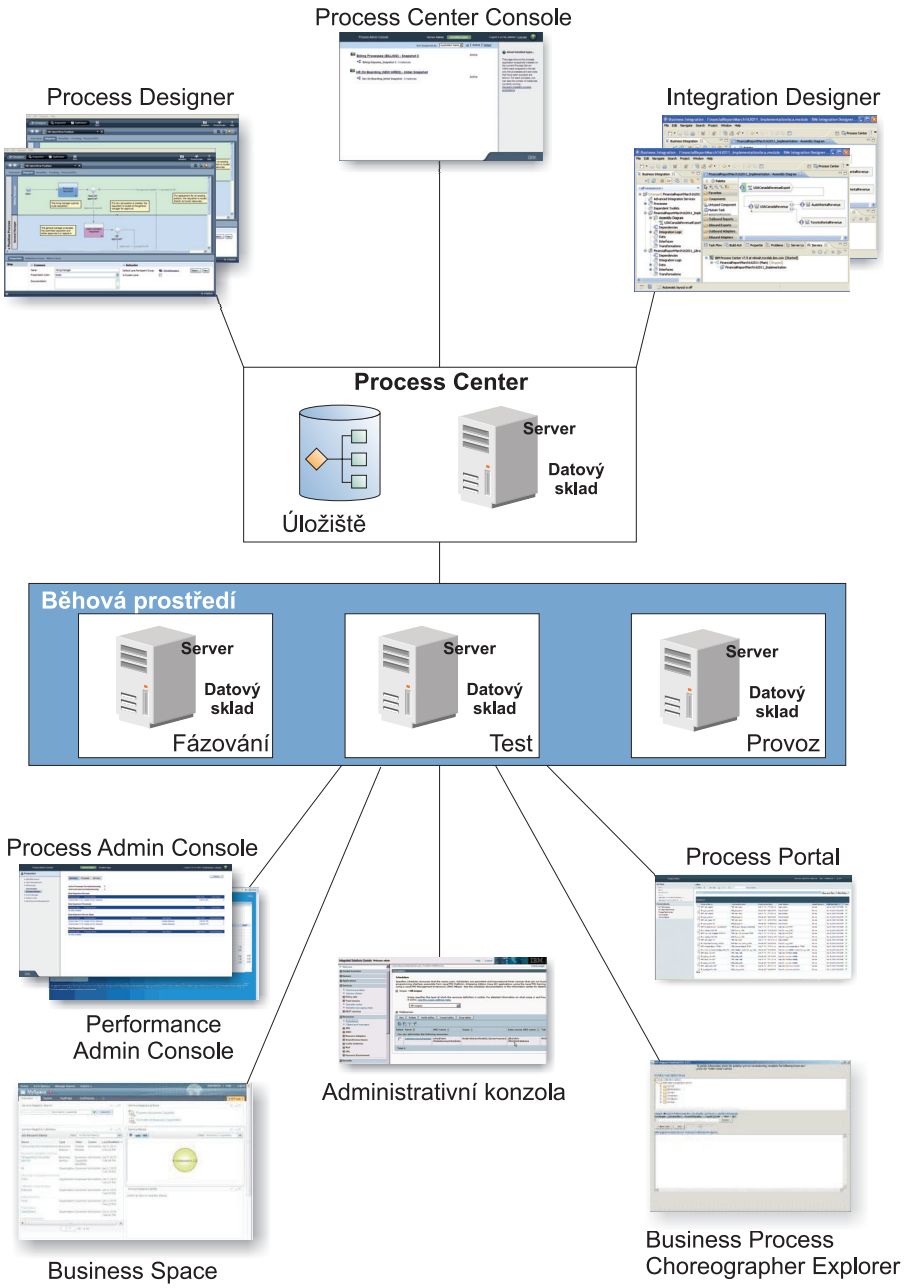
Verze služby

Není.

Přehled produktu

Komponenty produktu IBM Business Process Manager poskytují unifikované úložiště BPM, nástroje pro autory, administrátory a uživatele, a běhovou platformu. Různé konfigurace produktu podporují různé úrovně složitosti a zapojení do řízení BPM.

Následující diagram ilustruje typickou konfiguraci produktu IBM Business Process Manager:



- Z redakčních prostředí IBM Process Designer a IBM Integration Designer se může ke konzole Process Center připojovat více uživatelů.
- V redakčních prostředích Process Designer a Integration Designer vytváří návrháři procesů a služeb aplikace implementovatelných procesů a znovupoužitelné komponenty Toolkit. Komponenty Process Application obsahují modely procesů a implementace služeb, včetně všech potřebných podpůrných souborů. Ukládají se do úložiště komponenty Process Center, kde je možné je sdílet.
- Komponenta Process Center obsahuje dva servery – server Process Center Server a Performance Data Warehouse. Tyto servery umožňují uživatelům, kteří v produktu IBM Process Designer pracují, spouštět komponenty Process Application a ukládat údaje o výkonu pro účely testování a přehrávání při vývoji.
- Z konzoly Process Center Console mohou administrátoři instalovat komponenty Process Application, které jsou připraveny k fázování, testování nebo provozu na serverech Process Server v těchto prostředích.
- Z konzoly Process Center Console mohou administrátoři spravovat spuštěné instance komponent Process Application ve všech konfigurovaných prostředích.

- V produktu IBM Process Portal mohou provádět koncoví uživatelé své přiřazené úlohy. Servery Process Center a Process Server mohou v nakonfigurovaných běhových prostředích spouštět komponenty Process Application, které vytváří přiřazené úlohy.
- Pomocí produktu Process Portal se účastníci procesu mohou připojit k serveru Process Center nebo k serveru Process Server v libovolném nakonfigurovaném běhovém prostředí, v závislosti na tom, zda se proces vyvíjí, testuje nebo zda již byl uvolněn do produkčního prostředí.
- Komponenta Performance Data Warehouse načítá sledovaná data ze serveru Process Server nebo serveru Process Center v pravidelných intervalech. Uživatelé mohou vytvářet a zobrazovat sestavy, které využívají tato data v redakčních prostředích a v produktu IBM Process Portal.
- Na konzolách Process Admin Console a Performance Admin Console mohou administrátoři spravovat a udržovat všechny běhové servery.

Konfigurace produktu IBM Business Process Manager V8.0

Různé konfigurace produktu IBM Business Process Manager korelují s typickými vstupními body nebo fázemi programu řízení BPM v podniku.

Tabulka 1. Konfigurace produktu IBM Business Process Manager

Konfigurace	**Fáze**
Rozšířené	**Transformace** Úplná sada schopností řízení BPM. • Rozšířená podpora automatizace procesů s vysokým objemem. • Vestavěné komponenty SOA pro rozsáhlou integraci a koordinaci služeb v celém podniku.
Standardní	**Program.** Nakonfigurovaný pro typické projekty řízení BPM. • Pro programy zlepšení více projektů, s vysokým zastoupením obchodu. • Základní podpora systémové integrace. • Rychlý převod času na hodnotu a zlepšená produktivita uživatelů.
Expresní	**Projekt** Nakonfigurovaný pro první projekt řízení BPM. • Rychlý převod času na hodnotu: zlepšená produktivita uživatelů. • Nízká počáteční cena. • Jednoduchá instalace a konfigurace.

Funkce konfigurace produktu IBM Business Process Manager V8.0

Poznejte, jaké produkty a funkce nabízí IBM pro řízení BPM, a vyberte si to pravé řešení pro svůj podnik.

Produkt IBM Business Process Manager je jedinou platformou BPM, která do unifikovaného produktu kombinuje jak schopnosti zaměřené na člověka, tak schopnosti zaměřené na integraci. Pro různé uživatele si lze vybrat z různých konfigurací, a splnit tak rozdílné požadavky v podniku. Konfigurace produktu lze kombinovat, a zajistit tak redakční spolupráci a běhová prostředí s implementací po síti.

Tabulka 2. Schopnosti konfigurace produktu IBM Business Process Manager

Schopnost	**Adv.**	**Std.**	**Express**
Provedení kompatibilní s WebSphere Lombardi Edition	X	X	X
Process Designer (BPMN)	X	X	X
Redakční spolupráce/okamžité přehrání	X	X	X
Interaktivní uživatelské rozhraní "coache procesů"	X	X	X

Tabulka 2. Schopnosti konfigurace produktu IBM Business Process Manager (pokračování)

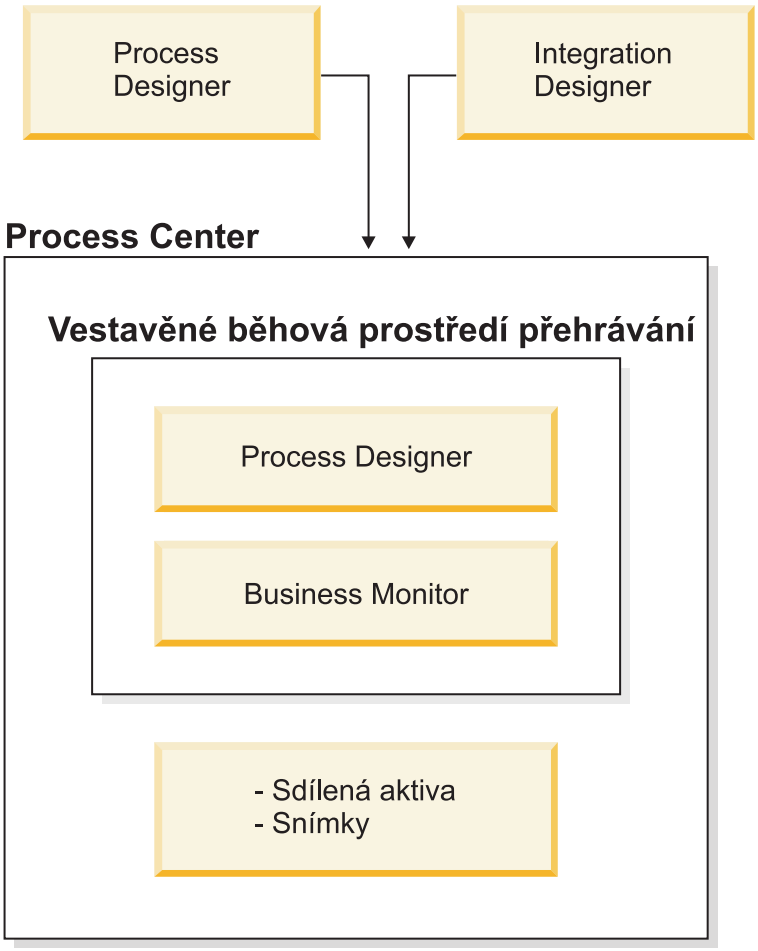
Schopnost	Adv.	Std.	Express
Pravidla procesu založená na ILOG	X	X	X
Process Portal	X	X	X
Monitorování a vytváření sestav v reálném čase	X	X	X
Analýzy & optimalizátor výkonu	X	X	X
Performance Data Warehouse	X	X	X
Process Center/sdílené úložiště aktiv	X	X	X
Autoři a koncoví uživatelé neomezených procesů	X	X	200 uživatelů/3 autoři
Vysoká dostupnost: klastrování a neomezená jádra	X	X	• 4 jádra produkce • 2 jádra vývoje • Bez klastru
Provedení kompatibilní s WebSphere Process Server	X		
Integration Designer (BPEL/SOA)	X		
Vestavěná sběrnice Enterprise Service Bus (ESB)	X		
Podpora transakcí	X		
Integrační adaptéry	X		
Flexibilní uživatelské rozhraní prostoru Business Space	X		
Rozšířená podpora platformy (Linux on System z, IBM AIX Solaris)	X	X	

Úložiště centra Process Center

Centrum Process Center zahrnuje úložiště pro všechny procesy, služby a další aktiva vytvořená v redakčních prostředích produktu IBM Business Process Manager, produktu Process Designer a Integration Designer.

Process Center je běhové prostředí, ve kterém komponenty Process Designer a Integration Designer sdílejí aktiva, což jim umožňuje vysoce interaktivní vývoj obchodních procesů založený na spolupráci. Tyto obchodní procesy mohou používat body monitorování vytvořené pomocí komponenty Toolkit Business Monitor Development Toolkit. Výsledkem je obchodní proces, jehož efektivitu lze kontrolovat za běhu v rámci reálných pracovních podmínek. Produkt Business Monitor poskytuje pohled panelu dashboard s měřidly a záznamy s vyhodnocením. Můžete přidávat výstrahy a oznámení, které vám v každém okamžiku poskytují informace o stavu obchodního procesu. Kritická místa, neefektivnosti a chyby v přidělení prostředků spuštěného obchodního procesu tak můžete zpozorovat a opravit, což vede ke zlepšení výkonu vašich obchodních procesů.

V následujícím diagramu vidíte několik souvisejících komponent, které vám společně umožní sestavit komplexní obchodní procesy.



Konzola Process Center Console poskytuje nástroje potřebné k údržbě úložiště.

- V konzole Process Center Console můžete vytvářet komponenty Process Application a Toolkit a udělovat k nim ostatním uživatelům přístup.
- V redakčních prostředích můžete vytvářet modely procesu, služby a další aktiva v rámci komponent Process Application.
- Komponenta Process Center zahrnuje server komponenty Process Center a datový sklad výkonu, které uživatelům umožňují pracovat s redakčními prostředími, a spouštět tak procesy a ukládat data o výkonu pro účely testování a přehrání.
- V konzole Process Center Console administrátoři instalují komponenty Process Application, které jsou připraveny k testování nebo výrobě na serverech procesů v těchto prostředích.
- V konzole Process Center Console mohou administrátoři spravovat spuštěné instance komponent Process Application v konfigurovaných prostředích.

Konzola Process Center Console poskytuje vhodné umístění, ve kterém lze vytvářet a udržovat kontejnery vysoké úrovně, jako jsou komponenty Process Application a Toolkit. Administrátoři, kteří v pohledu Návrhář aktivně nepracují, mohou konzolu Process Center Console použít k poskytnutí rámce, ve kterém mohou analytici řízení BPM a vývojáři sestavovat vlastní procesy a jejich základní implementace. Další úlohou primárně určenou pro administrátory je správa přístupu k úložišti Process Center pomocí nastavení příslušné autorizace pro uživatele a skupiny.

Tito uživatelé s příslušnou autorizací mohou provádět některé administrativní úlohy přímo v komponentách Process Designer a Integration Designer. Pokud chce například vývojář s přístupem pro zápis do komponenty Process Application zachytit stav všech aktiv projektů ke konkrétnímu milníku, může při práci v pohledu Návrhář vytvořit snímek.

Server Process Server a běhová prostředí

Server Process Server nabízí jedno běhové prostředí BPM, které podporuje celou řadu obchodních procesů, koordinací služeb a schopností integrace.

V redakčních prostředích vám umožňuje server procesů integrovaný do centra Process Center spouštět procesy při jejich sestavování. Až budete připraveni, můžete nainstalovat a spustit tyto procesy na serverech procesů v běhových prostředích. Komponenta Business Performance Data Warehouse shromažďuje a agreguje data procesů z procesů spuštěných na serverech procesů. Pomocí těchto dat můžete zlepšit své obchodní procesy.

Konzola Process Admin Console vám umožňuje spravovat servery procesů v běhových prostředích, např. ve fázovacím, testovacím, produkčním, ale i ty servery procesů, které jsou součástí komponenty Process Center.

Redakční prostředí

Produkt IBM Business Process Manager Advanced nabízí dvě redakční prostředí. V prostředí IBM Process Designer můžete efektivně modelovat obchodní procesy, které zahrnují lidské úlohy. V prostředí IBM Integration Designer můžete vybudovat takové služby, které budou samostatné, nebo které budou spouštět jiné existující služby, jako např. webové služby, aplikace podnikových prostředků nebo aplikace provozované v prostředích CICS a IMS.

- "Process Designer"
- "Integration Designer" na stránce 10

Process Designer

Prostředí Process Designer je k dispozici ve všech vydáních produktu. Produkt IBM Business Process Manager Advanced nabízí také prostředí Integration Designer včetně přidružených editorů a adaptérů.

Hlavními jednotkami logické struktury jsou v produktu IBM Business Process Manager procesy. Tímto pojmem se označuje kontejner obsahující všechny komponenty definice procesu: služby, aktivity a brány; události časovače, události zpráv a události výjimek; čáry posloupnosti, pravidla a proměnné. Při modelování procesu vytváříte opakovaně použitelnou definici obchodního procesu (BPD). V produktu IBM Process Designer můžete vytvářet modely procesů, které mohou obsahovat i lidské úlohy.

Process Designer vám pomáhá vyvíjet obchodní procesy. Díky graficky orientovanému nástroji, který není těžké používat, můžete vytvořit posloupnost akcí, které spoluvytváří obchodní proces, a tento proces můžete v případě, že se okolnosti změní, překreslit. Pokud některá z aktivit vyžaduje přístup k velkým back-endovým systémům nebo službám, které poskytují data obchodnímu procesu, např. potřebujete-li získat informace o zákaznících, pomůže vám produkt Integration Designer. Díky jednoduchému rozhraní může aktivita z produktu Process Designer volat službu z produktu Integration Designer. Tato služba pak pomocí mediačního toku převede, přesměruje a rozšíří data a adaptéry tak, aby bylo možné dorazit k řadě back-endových systémů standardním způsobem. Stručně lze říci, že se produkt Process Designer soustředí na obchodní proces, a produkt Integration Designer se soustředí na automatické služby, aby tento obchodní proces doplnil. Viz Začínáme s produktem IBM Process Designer.

Všechny projekty produktu Process Designer jsou součástí komponent Process Application. Tyto komponenty Process Application a jejich přidružené artefakty ukládáte do úložiště komponenty Process Center. Komponenty Process Application mohou sdílet aktiva, která byla umístěna do komponent Toolkit.

Produkt IBM Business Process Manager obsahuje několik uživatelských rozhraní, která umožňují modelovat, implementovat, simulovat a prohlížet obchodní procesy. V konzole Process Center Console vytváříte a spravujete komponenty Process Application, Toolkit, sledování a snímky. V produktu Process Designer můžete vytvářet modely procesů, sestavy a jednoduché služby. V komponentě Inspector můžete spouštět a ladit procesy. A v komponentě Optimizer můžete spouštět simulace.

Komponenty Process Application vyvinuté v produktu Process Designer lze kdykoliv spustit na serveru Process Center nebo uložit do snímku a naimplementovat na server Process Server. To samé platí pro služby vyvinuté v produktu Integration Designer a přidružené ke komponentám Process Application.

Integration Designer

Prostředí Process Designer je k dispozici ve všech vydáních produktu. Produkt IBM Business Process Manager Advanced nabízí také prostředí Integration Designer včetně přidružených editorů a adaptérů.

Produkt Integration Designer nabízí také editory a prostředky, které pomáhají vývojářům vytvářet komplexní automatické procesy a služby. Tento produkt je dostupný jako komponenta produktu IBM Business Process Manager Advanced, ale i jako samostatná sada nástrojů.

Produkt IBM Integration Designer byl navržen jako plnohodnotné vývojové prostředí integrace pro vývojáře integrovaných aplikací. Integrované aplikace nejsou jednoduché. Mohou volat aplikace v systémech EIS (Enterprise Information System), zahrnovat obchodní procesy mezi odděleními nebo společnostmi, vyvolávat lokálně nebo vzdáleně aplikace napsané v řadě jazyků a lze je spouštět v řadě operačních systémů. Komponenty jsou vytvářeny a sestavovány do jiných integrovaných aplikací (tedy aplikací vytvořených ze sady komponent) prostřednictvím vizuálních editorů. Vizuální editory prezentují vrstvu abstrakce mezi komponentami a jejich implementacemi. Vývojář může pomocí těchto nástrojů vytvořit integrovanou aplikaci, aniž by podrobně znal vlastní implementaci každé z komponent.

Nástroje produktu Integration Designer vychází z architektury SOA. Komponenty jsou službami a integrované aplikace zahrnující řadu komponent jsou také službou. Vytvářené služby splňují hlavní průmyslové standardy. Procesy BPEL, které se také stávají komponentami, se vytváří podobně pomocí jednoduchých vizuálních nástrojů, které odpovídají standardnímu jazyku BPEL (Business Process Execution Language).

V paradigmatu produktu Integration Designer se komponenty sestavují do modulů. Data mezi moduly se sdílí pomocí importů a exportů. Artefakty umístěné do knihovny mohou moduly sdílet.

Moduly a knihovny lze přidružit ke komponentě Process Application, aby je bylo možné použít v komponentě Process Center. Jako služby je mohou také používat procesy vytvořené v produktu Process Designer. V takovém případě je možné je také naimplementovat spolu s komponentou Process Application.

Moduly a knihovny lze případně také naimplementovat přímo do testovacího prostředí nebo do komponenty Process Server. Pomocí mediačních modulů můžete vytvořit mediační toky, které pak můžete naimplementovat do sběrnice WebSphere Enterprise Service Bus nebo na server Process Server.

Produkt IBM Integration Designer dále nabízí schopnosti pro vytváření datových typů a map XML, které je možné následně naimplementovat na zařízení WebSphere DataPower. Dále můžete přenášet soubory na zařízení WebSphere DataPower i nazpět.

Nástroje pro administraci

Produkt IBM Business Process Manager obsahuje komponentu Toolkit pro administraci, které vám pomohou s úlohami od instalace a správy snímků po administraci procesů a práci s prostředky ve vašem IT prostředí.

Nástroje příkazového řádku

Produkt IBM Business Process Manager poskytuje nástroje příkazového řádku, skriptovací rozhraní a programovací rozhraní pro správu běhového prostředí.

- Nástroje příkazového řádku jsou jednoduché programy, které můžete spustit z výzvy příkazového řádku operačního systému a použít je k provedení specifických úloh. Pomocí těchto nástrojů můžete spustit a zastavit aplikační servery, zkontrolovat stav serveru, přidat nebo odebrat uzly a provádět další úlohy.
- Administrativní skriptovací program WebSphere (wsadmin) je negrafické prostředí interpretu příkazů, které umožňuje spouštět administrativní volby ve skriptovacím jazyce a odesílat ke spuštění programy ve skriptovacím jazyce. Podporuje stejné úlohy jako administrativní konzola a také řadu úloh konzoly komponenty Process Center. Nástroj wsadmin je určen pro provozní prostředí a bezobslužné operace.
- Administrativní programovací rozhraní jsou sadou tříd a metod Java podle specifikace JMX (Java Management Extensions), které poskytují podporu pro administraci architektury SCA (Service Component

Architecture) a obchodních objektů. Každé programovací rozhraní obsahuje popis svého účelu, příklad, který demonstruje, jak rozhraní nebo třídu použít, a odkazy na popisy jednotlivých metod.

Konzola komponenty Process Center

Konzola komponenty Process Center poskytuje vhodné umístění, kde mohou uživatelé vytvářet a udržovat položky knihoven na vysoké úrovni, jako jsou komponenty Process Application a Toolkit. Pomáhá poskytnout rámec, ve kterém mohou analytici řízení BPM a vývojáři sestavovat vlastní procesy a jejich základní implementace. Konzola komponenty Process Center navíc poskytuje nástroje pro údržbu úložiště, včetně nastavení vhodných autorizací pro uživatele a skupiny.

Prostřednictvím webového prohlížeče otevřete konzolu centra Process Center (např. http://host:9080/ProcessCenter).

Process Admin Console

Konzola Process Admin Console slouží k administraci serverů procesů ve vašem prostředí, včetně uživatelů a instalovaných snímků na jednotlivých serverech. Navíc poskytuje nástroje, které usnadňují správu front a mezipamětí.

Konzola Process Admin Console obsahuje komponentu Process Inspector - nástroj pro zobrazení a správu instancí procesů pro komponenty Process Application, které jsou spuštěny na specifickém serveru procesů.

Prostřednictvím webového prohlížeče otevřete administrativní konzolu Process Admin Console (např. http://host:9080/ProcessAdmin).

Business Performance Admin Console

Konzola Business Performance Admin Console obsahuje nástroje pro správu komponent Performance Data Warehouse ve vašem prostředí. Pomocí tohoto nástroje můžete spravovat fronty serverů a monitorovat výkon serverů.

Prostřednictvím webového prohlížeče otevřete administrativní konzolu Business Performance Admin Console (např. http://host:9080/PerformanceAdmin).

Administrativní konzola serveru WebSphere Application Server

Administrativní konzola slouží k administraci aplikací, služeb a dalších prostředků v oboru buňky, uzlu, serveru nebo klastru. Konzolu můžete používat pro samostatné servery i pro správce implementace, jež spravují všechny servery v buňce v síťovém prostředí.

Pokud jste nainstalovali samostatný profil, získali jste jeden uzel v samostatné administrativní doméně, která se nazývá buňka. Pomocí administrativní konzoly můžete spravovat aplikace, sběrnice, servery a prostředky v rámci této administrativní domény. Podobně, pokud jste nainstalovali a nakonfigurovali buňku implementace sítě, získali jste uzel správce implementace a jeden nebo více spravovaných uzlů v téže buňce. Pomocí administrativní konzoly můžete spravovat aplikace, nastavit spravované uzly v buňce a monitorovat a řídit tyto uzly a jejich prostředky.

Otevřete tuto konzolu prostřednictvím webového prohlížeče (např. http://host:9043/ibm/console).

Business Process Choreographer Explorer a Business Process Archive Explorer

V závislosti na své uživatelské roli můžete pomocí těchto rozhraní klienta spravovat procesy BPEL a lidské úlohy vytvořené v produktu , pracovat se svými přiřazenými úlohami, nechat si zobrazit dokončené procesy BPEL a lidské úlohy z archivní databáze, nebo odstranit procesy a úlohy z archivu.

Business Space powered by WebSphere

Business Space powered by WebSphere poskytuje firemním uživatelům jednotné uživatelské ovládání napříč portfoliem produktů IBM pro řízení BPM. Prostor Business Space poskytuje upravitelné prostředí pro spolupráci určené k monitorování, kontrolování a administraci běžných obchodních procesů, jako jsou toky lidských úloh, modelování a indikátory výkonu.

Prostor Business Space je grafické uživatelské rozhraní založené na prohlížeči, v němž je možné prohlížet obsah z různých produktů v portfoliu řízení BPM a pracovat s ním. Prostor Business Space nejenže nabízí jeden webový bod pro přístup k obsahu, ale v prostoru Business Space můžete také kombinovat obsah užitečnými a zajímavými způsoby. Díky těmto kombinacím proniknete do podstaty svého obchodního podnikání a schopností, jak reagovat na jeho změny.

Business Process Rules Manager

Komponenta Business Process Rules Manager je webový nástroj, který obchodnímu analytikovi pomáhá při procházení a úpravách hodnot obchodních pravidel. Nástroj je volbou serveru IBM Process Server, jejíž instalaci můžete zvolit při vytvoření profilu nebo po instalaci serveru.

Novinky v produktu IBM Business Process Manager V8.0

Produkt IBM Business Process Manager verze V8.0 přináší novou, přestavěnou komponentu Process Portal, integraci se systémy pro správu podnikového obsahu, vyhledávání a sdílení obsahu mezi komponentami Process Center, rozšířené schopnosti řízení a řadu dalších nových funkcí produktu IBM Business Process Manager verze V8.0.

- "Process Portal"
- "Process Designer"
- "Process Center" na stránce 13
- "Process Server" na stránce 15
- "Instalace a konfigurace" na stránce 15
- "Integration Designer" na stránce 16

Process Portal

Přepracovaná komponenta Process Portal přináší účastníkům procesu pracovní zkušenost s vysokou mírou spolupráce za využití rozšířených sociálních schopností. Komponenta Process Portal nyní obsahuje tyto nové funkce:

- Schopnost vyžádat si pomoc od odborníků a spolupráce s nimi, stejně jako s dalšími uživateli, v reálném čase za účelem dokončení práce na úloze.
- Schopnost přidávat komentáře a připojovat dokumenty ke specifickým procesům či úlohám.
- Provést jediným klepnutím odběr instancí procesů, o které má uživatel zájem, poskytující vypsání oznámení týkajících se daného procesu na obrazovku, stejně jako vypsání aktualizací aktivit z proudu aktivit odebírajícího uživatele.
- Proudy aktivit zobrazující aktualizace aktivit, jako např. vytvoření a dokončení úlohy, akce a komentáře uživatelů, stejně jako oznámení související s úlohami, které vlastní uživatel, nebo které souvisí s určitými instancemi procesu následovanými uživatelem.
- Rozšíření informací o profilu, včetně avatarů a konfigurace oznámení.

Process Designer

Vylepšení návrhu procesů

Následující nové vlastnosti zlepšují funkčnost nabízenou účastníkům procesů v komponentě Process Portal:

- Automatické spuštění další úlohy - můžete nakonfigurovat jednotlivé aktivity pro automatické spuštění v případě, že jsou přiřazeny stejné osobě jako předchozí úloha. Pokud je v rámci komponenty Process Portal vlastník aktuální úlohy totožný s vlastníkem další úlohy, následující úloha se spustí automaticky po dokončení aktuální úlohy.
- Omezení jednorázových akcí na milník či skupinu účastníků - můžete nakonfigurovat jednorázové akce, nazývané také *akce zahájené uživatelem*, které budou dostupné pouze pro určitou fázi procesu či pro určitou skupinu uživatelů, což učiníte omezením viditelnosti přidružené jednorázové události na určitou dráhu či milník definice BPD.
- Konfigurace aktivit pro vložené dokončení - můžete nakonfigurovat úlohy uživatelů, což zahrnuje jednoduché rozhodnutí, například pro schválení nebo zamítnutí požadavku nebo pro volbu mezi sadou možností tak, aby ji mohl firemní uživatel dokončit v komponentě Process Portal bez nutnosti otevírání modulu Coach. Místo toho uživatel klepne na tlačítko nebo si jedním klepnutím vybere příslušnou volbu.

Vytvoření znovupoužitelných uživatelských rozhraní a chování pro moduly Coach

V produktu IBM BPM verze V8.0 byly moduly Coach kompletně přepracovány a obsahují teď pohledy modulu Coach. Pohledy modulu Coach obsahují znovupoužitelná uživatelská rozhraní, která můžete vytvářet a upravovat. Pohledy modulů Coach se skládají z minimálně jednoho dalšího pohledu modulu Coach, vázání

dat, informací o rozvržení a z chování. Protože jsou pohledy modulů Coach znovupoužitelné, můžete vytvořit knihovnu společných uživatelských rozhraní a chování, pomocí kterých můžete rychle vyvíjet nové moduly Coach.

¨Pro vyšší flexibilitu při vytváření toku služeb mohou pohledy modulů Coach vysílat hraniční události, pomocí kterých propojujete uzly ve službě.

Chcete-li zachovat zpětnou kompatibilitu, moduly Coach z předchozích verzí se nyní nazývají *Heritage Coach*. Můžete nadále používat a spravovat moduly Heritage Coach, ale při vytváření uživatelských rozhraní pro služby používejte moduly Coach.

BPMN 2.0: Rozšířená podpora ošetřování chyb a ukončení

Verze V8.0 nyní nabízí více možností generování a zachytávání chyb pomocí chybových událostí v definicích BPD, podprocesech a službách (včetně služeb rozšířené integrace). Můžete generovat specifický chybový objekt pomocí výběru proměnné a můžete zachytávat specifické chyby a následně je mapovat na proměnnou. Zlepšené schopnosti ošetřování chyb zahrnují prostřednictvím události ukončení chyby možnost určení kódu chyby a mapování na typ chyby pro chyby generované v rámci toku definice BPD či služby. Použitím intermediační události typu chyba si můžete v rámci zachytávání chyb výběrem ze seznamu všech generovaných chyb pro daný propojený proces, podproces či službu zvolit, zda chcete zachycené chyby filtrovat. Výběrem dříve definované proměnné můžete data chyby mapovat také na proměnnou. Pokud zachycujete specifické chyby, můžete vybrat kód chyby, mapovat data chyby, nebo obojí. Modely vytvořené v předchozích verzích následují chování dané verze.

V případě instancí procesu máte k dispozici větší flexibilitu při definování rozsahu události ukončení typu ukončení. Můžete určit, zda byly všechny aktivity dané instance procesu ukončeny, dokonce i nadřízené procesy. V předchozích verzích bylo možné ukončit pouze celou instanci procesu. Při navrhování modelů s událostmi ukončení typu ukončení nebylo chování viditelné a nebylo možné jej změnit. Ve verzi V8.0 je pro nové modely standardně zrušeno zaškrtnutí políčka pro ukončení celé instance procesu. Událost ukončení typu ukončení tedy ukončuje aktivity na té úrovni procesu, na které jste je přidali, včetně úrovně podprocesů s aktivitami nižší úrovně. Pro účely modelů procesu vytvořených ve straších verzích a migrovaných do verze V8.0 bylo předchozí chování ukončení všech aktivit dané instance procesu zachováno, pokud nezrušíte zaškrtnutí zmiňovaného políčka. Podle vlastních potřeb můžete toto zaškrtávací políčko vybrat či jeho výběr zrušit.

Integrace se systémy pro správu podnikového obsahu

Systémy pro správu podnikového obsahu (ECM) vám pomáhají se správou dokumentů všech typů, jako jsou záznamy, obrázky a webové stránky. Začlenění nové služby systému pro správu podnikového obsahu (ECM) do vašich obchodních procesů v produktu IBM Business Process Manager vám umožní vyhledávat, zobrazovat a ukládat dokumenty v systémech pro správu podnikového obsahu (ECM).

- Chcete-li rychle sestavit uživatelské rozhraní umožňující vypisování, zobrazování a ukládání dokumentů, můžete použít ovládací prvky modulu Coach.
- Pomocí grafického uživatelského rozhraní můžete vytvářet dotazy na systém správy podnikového obsahu bez nutnosti znalostí syntaxe dotazovacího jazyka CMIS (Content Management Interoperability Service).
- Díky tomu, že je systém pro správu podnikového obsahu (ECM) založený na rozhraní CMIS, které je odvětvovým standardem, produkt IBM Business Process Manager se může připojit ke kterémukoli z produktů systému pro správu podnikového obsahu (ECM), který rozhraní CMIS podporuje.

Viditelnost dat

Obchodní objekt lze identifikovat jako sdílený obchodní objekt, což jej a jeho hodnoty za běhu zpřístupňuje pro ostatní instance.

Process Center

Řízení instalace snímků komponenty Process Application s procesy řízení

Můžete použít proces řízení, který umožňuje řízení instalace snímků komponenty Process Application. Probíhá-li řízení v rámci komponenty Process Application, všechny požadavky z produktu IBM Process

Center na instalaci snímku komponenty Process Application prochází tímto procesem řízení. Snímek komponenty Process Application se na server Process server nainstaluje až po dokončení schvalování definovaného v procesu řízení.

Můžete také vytvořit proces řízení, který bude reagovat na změnu stavu snímku.

Referenční odkazy

Dokumentace procesu nyní obsahuje obsah a referenční odkazy ve formátovaném textu, abyste mohli připojovat odkazy na zdroje obsahu a další zdroje. Následují příklady možných referenčních odkazů:

- Web nebo wikiweb.
- Požadavek na změnu uložený v systému správy změn.
- Testovací případ uložený v systému řízení kvality.
- Artefakty spravované poskytovateli obsahu využívajícími standard OSCL (Open Services for Lifecycle Collaboration).

Tato schopnost odkazování vám pomáhá dosáhnout sledovatelnosti nebo poskytnout podrobnosti o změnách v obchodním objektu či rozhraní služby.

Porovnat a zkopírovat

Pomocí nové funkce porovnání a zkopírování se můžete dozvědět podrobnosti o vašich měnících se aplikacích obchodního procesu:

- Porozumíte tomu, které komponenty jsou nové, aktualizované či v konfliktu při porovnávání snímku s vrcholem stopy.
- Porovnáte položky knihovny ze snímku komponenty Process Application s položkami knihovny na vrcholu stopy.
- Vyberete změněné komponenty z jednoho snímku a zkopírujte je na vrchol stopy s přidruženými závislostmi.
- V případě importu komponenty Process Application začleníte volbu pro vytvoření stopy .

Vyhledávání a sdílení obsahu mezi centry Process Center

Nyní lze za použití vyhledávacího pole vyhledat aktiva, jako např. komponenty Toolkit, Process Application, služby či obchodní objekty rychleji pomocí specifické syntaxe či značek.

- Vyhledávání komponent Process Application, Toolkit a položek knihovny za základě určených klíčových slov.
- Filtrování výsledků podle typu pomocí schopnosti přímého vyhledávání.
- Správa indexu komponenty Process Center, používané k vyhledávání v úložišti komponenty Process Center. Index se vytváří a udržuje automaticky. Index můžete nyní znovu vytvořit nebo aktualizovat ručně. Index můžete také nakonfigurovat pro automatické zpracování.
- Náhled jednotlivých výsledků a zobrazení přidružené dokumentace k procesu.
- Registrace pro účely sdílení obsahu procesu s jinou komponentou Process Center. Když vzájemně zaregistrujete dvě komponenty Process Center, můžete sdílet komponenty Toolkit s jinými uživateli nebo se přihlásit k odběru komponent Toolkit sdílených jinými uživateli.
- Sdílení komponent Toolkit poskytujících společný či ukázkový obsah.
- Označování klíčových položek knihovny při jejich uvolňování, aby odběratelé věděli, co mají používat.
- Publikování jednotlivých snímků a oznámení uživatelům o dostupnosti nové verze.
- Odebírání sdíleného obsahu procesu (Toolkit) z jiné komponenty Process Center.
- Přijímání oznámení na dostupnost nových verzí (snímků).

Porovnání snímků před migrováním instancí za účelem identifikace potenciálních umístění osiřelých tokenů

Nyní můžete před migrací instancí porovnat snímky pomocí souboru zásad. Tento soubor použijte k identifikaci potenciálního umístění osiřelých tokenů, tokenů přidružených k aktivitám odebraným z definice BPD a určení, zda byl každý osiřelý prvek během migrace instance odstraněn nebo přesunut.

Process Server

Schopnosti sběrnice ESB v produktu IBM Process Server

Produkt IBM Business Process Manager Advanced poskytuje stejné schopnosti sběrnice ESB, které jsou dostupné v produktu WebSphere Enterprise Service Bus. Do komponenty mediačního toku bylo přidáno několik funkcí, které jsou dostupné pouze po implementaci do produktu IBM Process Server verze V8.0.

Komponenta mediačního toku byla aktualizována a zahrnuje teď tyto nové funkce:

- Primitiva WebSphere eXtreme Scale

  Tato primitiva mediačního toku poskytují elastickou rozšiřitelnost produktu WebSphere eXtreme Scale, a přinášejí vám tak konektivitu rozšiřující obsah mezipaměti pro účely back-endových systémů se sníženou prioritou a velkých binárních dat. Mezi klíčové oblasti použití patří ukládání odezev a zásad do mezipaměti nebo perzistence požadavků. Mediační primitiva eXtreme Scale lze používat pouze v případě nainstalování produktu WebSphere eXtreme Scale.
- Zjednodušení stylu vyvolání služby

  K dispozici jsou další volby stylu vyvolání, které umožňují řídit styl vyvolání služby bez nutnosti určení dalších parametrů, a obecně řečeno také bez nutnosti zavažování stylu vyvolání, který vyvolal mediační tok. Nové styly vyvolání jsou Asynchronní s odloženou odezvou, Asynchronní se zpětným voláním a Jako cíl.
- Optimalizované transformace XSLT

  Mediační primitivum Transformace XSL bylo přejmenováno na mediační primitivum Mapování. Za účelem dosažení zlepšené funkčnosti a výkonu lze přepínat mezi transformačními stroji transformace XSLT a modulu mapování obchodních objektů.
- Podpora produktu WebSphere Service Registry and Repository verze 8.0

Instalace a konfigurace

Zlepšená instalace

- V případě typické či vlastní instalace za použití vestavěné databáze IBM DB2 Express a vlastní instalace za použití lokálního databázového serveru DB2 instalace časněji zachytí více problémů a poskytuje informace o jejich odstranění před zahájením instalačního procesu.
- Z vlastní instalace byla odebrána výchozí hesla. V případě typické instalace byla tato hesla změněna tak, aby odpovídala zásadám hesel na všech operačních systémech.
- Instalujete-li server Process Server pro produkční účely, vyberte volbu **Produkce**, pro účely testování, fázování či vývoje vyberte volbu **Neprodukční**. Pro neprodukční využití serveru Process Server je nyní k dispozici oddělené licencování.
- Pomocí datového kolektoru produktu IBM Support Assistant, který je součástí instalace produktu IBM Business Process Manager, můžete vyhledávat informace, prozkoumávat problémy či odesílat zprávy o problému IBM.
- Interaktivní průvodce instalací a konfigurací je nový formulář, s jehož pomocí můžete vygenerovat sadu témat týkajících se instalace a konfigurace přizpůsobených přesně vašim instalačním potřebám. Ve formuláři Interaktivní průvodce instalací a konfigurací vyberte volby potřebné pro váš scénář instalace. Během výběru voleb nástroj automaticky odebírá volby vyřazené na základě vašeho předchozího výběru. Pokud například označíte, že plánujete instalovat konfiguraci Express, je odebrán produkt Network Deployment jako potenciální topologie. Tento formulář je k dispozici v Informačním centru.

Zlepšená funkčnost a spolehlivost databáze

- Produkt IBM Business Process Manager nyní podporuje produkt Oracle Data Guard, který poskytuje mechanizmus s vysokou dostupností, zotavením z havárie a ochranou dat, používaný k vytváření, správě a monitorování jedné či více rezervních databází v zájmu odolnosti produkčních databází Oracle vůči haváriím a poškozením dat.
- Své databázové řešení můžete nyní rozšířit pomocí funkce IBM DB2 pureScale. Příchozí požadavky na databázi zpracovává více databázových serverů, známých jako *členové*; tito členové operují v klastrovaném systému a sdílejí data.

Zlepšená flexibilita změny hesla databáze

Nyní můžete po dokončení konfigurace databáze heslo databáze podle potřeby překonfigurovat. Překonfigurování hesla poskytuje flexibilitu v případě, kdy roli administrátora databáze převezmou noví uživatelé, nebo když vaše společnost používá zásadu pravidelné změny hesel.

Integration Designer

Zlepšená viditelnost a kontrola nad opakovanými pokusy systému pro komponenty SCA ve vaší aplikaci

Nyní máte větší kontrolu nad navržením opakování systému v rámci plánování běhového prostředí. Počet opakování můžete nastavit na stránce vlastností svého modulu, nebo můžete pomocí průvodce Konfigurovat asynchronní počet opakování změnit počet opakování pro více modulů. Při výskytu systémové chyby je opakováno asynchronní vyvolání, dokud není dosaženo určeného počtu opakování. V předchozích verzích se moduly vytvářely s počtem opakování 4. Nové moduly se nyní vytvoří s počtem opakování rovným nule. Moduly z předchozích verzí si při migraci ponechají stávající nastavení opakování.

Na rozdíl od předchozích verzí chování opakování mediačních primitiv potlačí asynchronní počet opakování, a to i když opakování neurčíte. Před touto verzí nebyla logika opakování těchto primitiv integrována do základní asynchronní logiky opakování, takže mohlo k opakování dojít i v případě, že jste opakování nenadefinovali, nebo mohlo dojít najednou k opakování jak na základě mediačního primitiva, tak i na základě cíle sběrnice pro integraci služeb. Nyní je uplatněno chování definované v těchto mediačních primitivech, které tak potlačuje logiku opakování cíle sběrnice pro integraci služeb.

Vzhledem k tomu, že mediační primitivum potlačuje asynchronní počet opakování, mohou situace překonání selhání, jako např. problém s aplikačním serverem či strojem systému zpráv, způsobit výskyt zpráv ve správci událostí se selháním. V předchozích verzích mohl tyto zprávy zpracovávat cíl sběrnice pro integraci služeb.

Generování mapy obchodních objektů pro účely zvýšení rychlosti a flexibility

Za účelem generování mapy obchodních objektů lze nyní kromě mapy XSLT nastavit také typ generování map.

Zlepšené ošetření chyb s podporou poruch v rámci služeb rozšířené integrace

Služby rozšířené integrace podporují operace rozhraní s poruchami.

Usnadnění přístupu v produktu IBM Business Process Manager

Funkce usnadnění přístupu napomáhají uživatelům s fyzickým postižením (například s pohybovým omezením či vadou zraku) při úspěšném využívání produktů z oblasti informačních technologií.

IBM se snaží nabízet produkty s použitelným usnadněním přístupu pro každého, bez rozdílu věku či schopností. Asistenční technologie, jako je softwarové zařízení pro čtení obrazovky nebo digitální syntetizátor řeči, můžete využívat k práci s funkcemi a prvky zobrazenými na obrazovce. Podrobnosti o použití těchto technologií s tímto produktem najdete v dokumentaci k podpůrné technologii.

Funkce lze ovládat pomocí klávesnice namísto myši.

Můžete si upravit atributy, jako např. barvy, kontrast a velikost fontu.

Informace prezentované v grafických zobrazeních lze detailněji přiblížit.

Dokument U.S. Section 508 Voluntary Product Accessibility Template (VPAT) si můžete vyžádat na webovém serveru IBM na adrese http://www.ibm.com/able/product_accessibility/index.html.

Modul dokumentace Informačního centra je vybaven následujícími doplňkovými funkcemi a pomůckami pro usnadnění přístupu:

- Dokumentace je k dispozici ve formátech HTML s cílem umožnit přístup uživatelům, kteří chtějí používat softwarovou technologii čtecího zařízení.

- Obrázky v dokumentaci jsou poskytovány s alternativním textem, takže s obsahem obrázků se mohou seznámit i uživatelé s vadou zraku.

Dostupnost národních jazyků v produktu IBM Business Process Manager

Produkt IBM Business Process Manager podporuje následující jazyky. Dokumentace nemusí být plně přeložena.

- Čínština - zjednodušená.
- Čínština - tradiční.
- Čeština.
- Americká angličtina.
- Francouzština.
- Němčina.
- Maďarština.
- Italština.
- Japonština.
- Korejština.
- Polština.
- Brazilská portugalština.
- Ruština.
- Španělština.

Produkt IBM Business Process Manager částečně podporuje následující jazyky. Dokumentace nemusí být plně přeložena.

- Arabština (přeloženy moduly widget lidských úloh BPEL, moduly widget komponenty Business Process Choreographer Explorer, moduly widget monitorování a rámec prostoru Business Space).
- Dánština (přeloženy moduly widget monitorování prostoru Business Space).
- Nizozemština (přeloženy komponenty Process Designer, Process Center a rámec prostoru Business Space).
- Finština (přeloženy moduly widget monitorování prostoru Business Space).
- Řečtina (přeloženy komponenty Process Designer, Process Center a rámec prostoru Business Space).
- Hebrejština (přeloženy lidské úlohy BPEL, komponenta Business Process Choreographer Explorer a rámec prostoru Business Space).
- Norština (přeloženy moduly widget monitorování prostoru Business Space).
- Portugalská portugalština (komponenty Process Designer, Process Center).
- Rumunština (přeloženy běhové operace).
- Slovenština (přeložen prostor Business Space).
- Švédština (přeloženy moduly widget monitorování prostoru Business Space).
- Turečtina (přeložen prostor Business Space).

  Poznámka: V případě tureckého národního prostředí musíte nastavit položku **case-insensitive-security-cache** v souboru 60Database.xml na hodnotu **false**, aby mohla jména uživatelů a hesla obsahovat písmeno **"i"** (např. **tw_admin**). Soubor 60Database.xml se nachází v adresáři *kořenový_adresář_instalace*\profiles*název_profilu*\config\cells*název_buňky*\nodes*název_uzlu*\servers*název_serveru*\process-center\config\system\.

Produkt IBM Business Process Manager poskytuje uživatelům podporu pro zadávání obousměrných řetězců v prostředí komponenty Process Designer, v modulech coach a v portálu Process Portal. Poskytuje rozhraní API JavaScript pro manipulaci s testy v obousměrných jazycích.

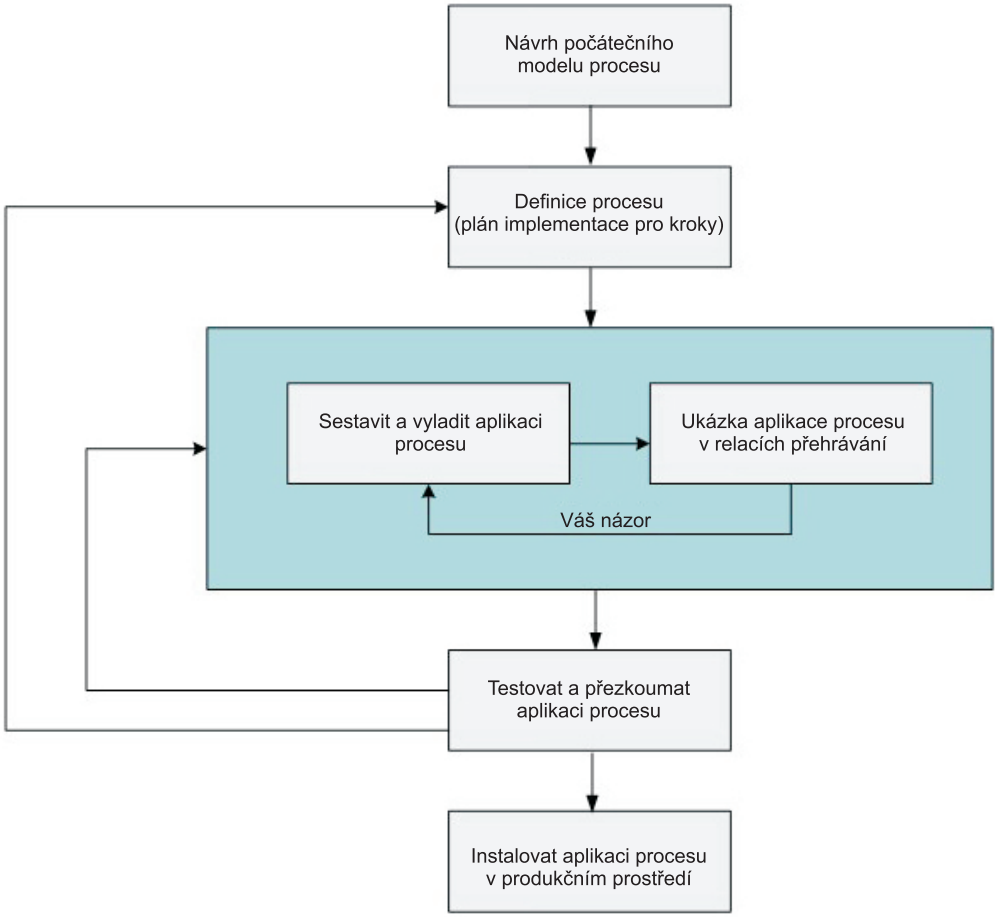
Moduly coach a portál Process Portal podporují používání hebrejských a arabských kalendářů.

Přehled řízení BPM

Vyvíjíte-li procesy v prostředí Process Designer, musíte naplánovat případnou instalaci komponent Process Application na serverech v testovacím a produkčním prostředí.

Následující diagram ukazuje životní cyklus typického průběhu vývoje procesu. Zahrnuje kroky pro sestavení a vyladění instalační služby, abyste mohli komponenty Process Application nainstalovat do produkčního prostředí.

Jak tento diagram ukazuje, můžete pracovat výlučně ve svém vývojovém prostředí. Musíte ale server Process Server nakonfigurovat jak v testovacím, tak i v produkčním prostředí.



Přehled modelování procesů

Hlavními jednotkami logické struktury jsou v produktu IBM Business Process Manager procesy. Tímto pojmem se označuje kontejner obsahující všechny komponenty definice procesu: služby, aktivity a brány; události časovače, události zpráv a události výjimek; čáry posloupnosti, pravidla a proměnné. Při modelování procesu vytváříte opakovaně použitelnou definici obchodního procesu (BPD).

Komponenty procesu umožňují definovat sled prací v procesu pro koncové uživatele, přičemž se vytvoří vnitřní logická struktura procesu a integrační propojení s dalšími aplikacemi a zdroji dat. Chcete-li pochopit, co se při běhu v rámci procesu odehrává, je důležité porozumět funkci jednotlivých komponent, z nichž je proces sestaven v průběhu návrhu.

Sestavování procesů v produktu IBM BPM

Na vývoji procesů pomocí produktu IBM BPM se obvykle podílí mnoho různých jednotlivců z různých organizací. Prvořadým úkolem je vždy zajistit, aby bylo vytvářeno nejlepší možné řešení pro stanovené cíle daného projektu. Úspěšných výsledků lze dosáhnout vzájemnou spoluprací členů týmu, kteří nejprve shromáždí požadavky na procesy a poté postupně vytvoří model a jeho implementace.

Opětovné použití položek v produktu Process Designer

Produkt Process Designer umožňuje vývojářům procesů opakovaně používat existující položky uvnitř jednotlivých komponent Process Application i napříč více komponentami. Víte-li například, že již existuje více služeb obsahujících moduly Coach a další sdílené položky, které potřebujete vy i další vývojáři, můžete k těmto položkám získat přístup a používat je opakovaně, zahrnete-li je do komponenty Toolkit. Poté můžete v komponentě Process Application přidat závislost na komponentě Toolkit, v níž jsou sdílené položky umístěny. Tak budete moci při volbě implementace aktivity vybrat některou z existujících služeb. Položky v komponentě Toolkit mohou využívat i jiní vývojáři, kteří pracují s jinými komponentami Process Application.

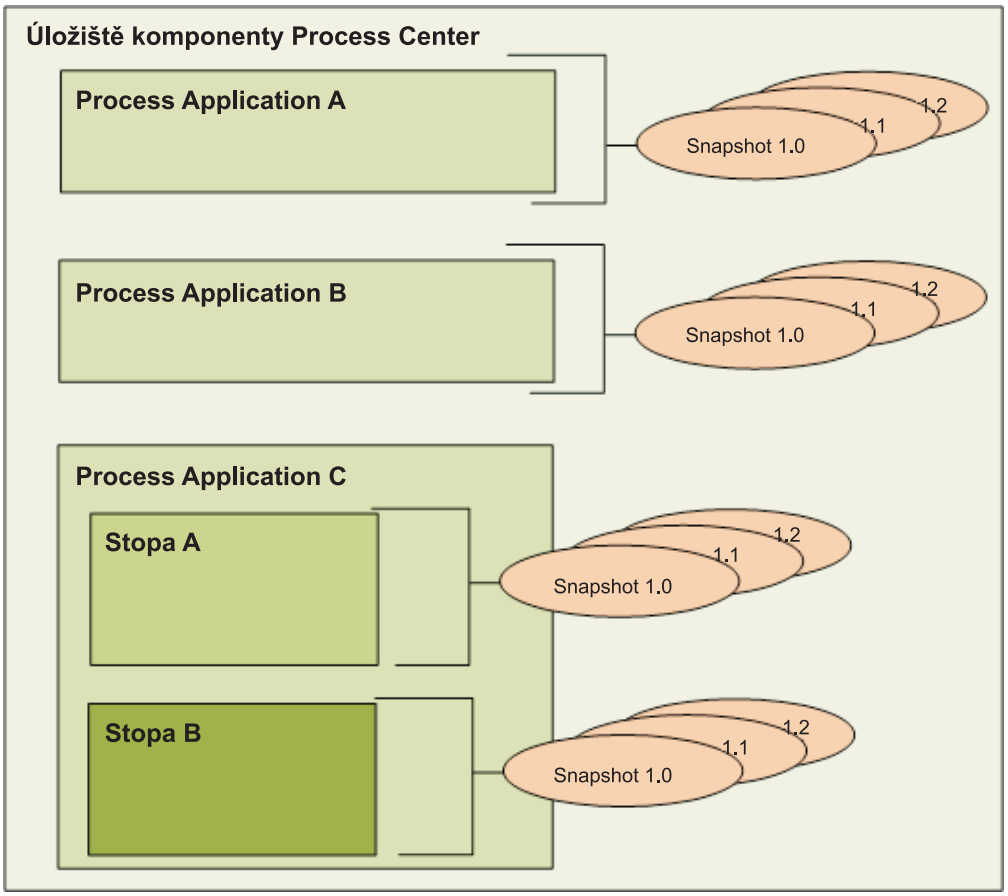
Použití komponenty Designer v prostředí IBM Process Designer

Rozhraní komponenty Designer poskytuje nástroje potřebné k modelování procesů v produktu IBM BPM.

Vývoj procesů s komponentou Process Center

Komponenta IBM Process Center slouží jako centrální úložiště pro všechna projektová aktiva vytvořená v produktu Process Designer. Připojuje-li se ke komponentě Process Center více klientů produktu Process Designer, mohou uživatelé sdílet položky, např. procesy a služby, a rovněž sledovat změny provedené jinými uživateli ihned, jakmile k nim dojde. Komponentu Process Center můžete použít rovněž jako úložiště pro aktiva vytvořená v produktu IBM Integration Designer.

Vyvíjíte-li procesy v produktu Process Designer, máte v úložišti komponenty Process Center k dispozici hierarchii navrženou tak, aby vám pomohla se správou projektů. Následující obrázek přehledně znázorňuje koncepci hierarchie úložiště:



Jak vidíte na předchozím diagramu, úložiště komponenty Process Center obsahuje následující artefakty:

Typ obsahu	Popis
Komponenta Process Application	Kontejnery pro modely procesů a pomocné implementace, které analytici řízení BPM a vývojáři vytvářejí v komponentě Designer v produktu IBM Process Designer.
Stopy	Volitelné podružné rozdělení komponenty Process Application podle týmových úloh nebo verzí komponenty Process Application. Jsou-li stopy povoleny, umožňují, aby probíhal paralelní vývoj. Administrátoři určují, zda jsou nezbytné další stopy, a povolují je pro jednotlivé komponenty Process Application.
Snímky	Záznamy o stavu položek v komponentě Process Application nebo ve stopě ke specifickému časovému bodu. Snímky obvykle reprezentují milníky nebo se používají při demonstracích a instalacích. Snímky lze porovnávat a vracet se k předchozím snímkům. Pokud administrátor u určité komponenty Process Application povolí stopy, jako základ nové stopy se použije snímek.

Komponenty Process Application: Přehled

Komponenta Process Application je kontejner modelů procesu a jejich podpůrných implementací. Je uložen v úložišti. Po vytvoření nebo jiném vygenerování artefaktů je z nich sestavena komponenta Process Application.

Komponenty Process Application obsahují některé nebo všechny následující artefakty:

- Jeden či více modelů procesu, nazývaných také Definice obchodního procesu (BPD).
- Odkazy na komponenty Toolkit.
- Služby nezbytné pro implementaci aktivit nebo integraci s jinými systémy, včetně služeb rozšířené integrace.
- Jednu či více stop.
- Moduly a knihovny SCA (Service Component Architecture) (vytvořené v produktu IBM Integration Designer).

- Model produktu IBM Business Monitor pro monitorování obchodní výkonnosti.
- Libovolné další položky nezbytné pro spuštění procesu.

Pomocí relací přehrání je možné okamžitě testovat a spravovat aktuální pracovní verzi komponenty Process Application (zvanou vrchol). Také je možné vytvořit snímek vrcholu, který zaznamená stav položek knihovny v rámci komponenty Process Application, nebo stopu ve specifickém okamžiku. Snímek komponenty Process Application lze testovat, instalovat a spravovat.

Komponenty Process Application a aplikace BLA

Každá nainstalovaná komponenta Process Application má aplikaci BLA, která se chová jako kontejner komponenty Process Application a jejích aktiv (aktiva zahrnují věci jako modely monitorování, moduly SCA, komponenty Toolkit a knihovny). Navíc má svou aplikaci BLA i každý snímek komponenty Process Application. Mnoho úloh administrace snímku (například jeho zastavení nebo spuštění na produkčním serveru) se provádí na úrovni aplikace BLA, což umožňuje rychlejší a jednodušší administraci snímku a všech jeho aktiv.

Spuštění a ladění procesů

S pomocí komponenty Inspector mohou jednotliví vývojáři spouštět procesy a služby na serveru Process Center Server nebo vzdáleném běhovém serveru Process Server.

Komponenta Inspector v produktu IBM Process Designer je klíčovou komponentou iterativního přístupu k vývoji procesů. Celý vývojový tým může pomocí komponenty Inspector demonstrovat aktuální návrh procesu a implementaci v relacích přehrání. Relace přehrání usnadňují zachycení důležitých informací od různých účastníků procesu, jako je například management, koncoví uživatelé a obchodní analytici. Použití iterativního přístupu k vývoji procesů zajišťuje, že komponenty Process Application splní cíle a požadavky všech zúčastněných stran.

Komponenta Inspector v produktu IBM Process Designer zahrnuje několik nástrojů, které umožňují provádět úlohy, jako jsou například následující úlohy, v každém konfigurovaném prostředí:

Úloha	Popis
Spravovat instance procesů	Při spuštění procesu můžete zobrazit všechny dříve spuštěné a aktuálně spuštěné instance na serverech produktu IBM Business Process Manager v daném prostředí. Spuštěné instance můžete spravovat například pomocí zastavení a následného obnovení. Dále můžete dříve spuštěné instance spravovat pomocí filtrování nebo odstranění specifických záznamů.
Projít a vyladit proces	Pro vybranou instanci zobrazte aktuálně prováděný krok a poté daný proces projděte a vyhodnoťte spuštění procesu po jednotlivých krocích. Zobrazení stromu daného procesu sloučené s indikátory nazvanými tokeny v diagramu procesu usnadňuje zjišťování, kde se v daném procesu nacházíte. Dále můžete využít výhody zobrazení proměnných používaných v jednotlivých krocích a jejich příslušných hodnot (pokud existují).

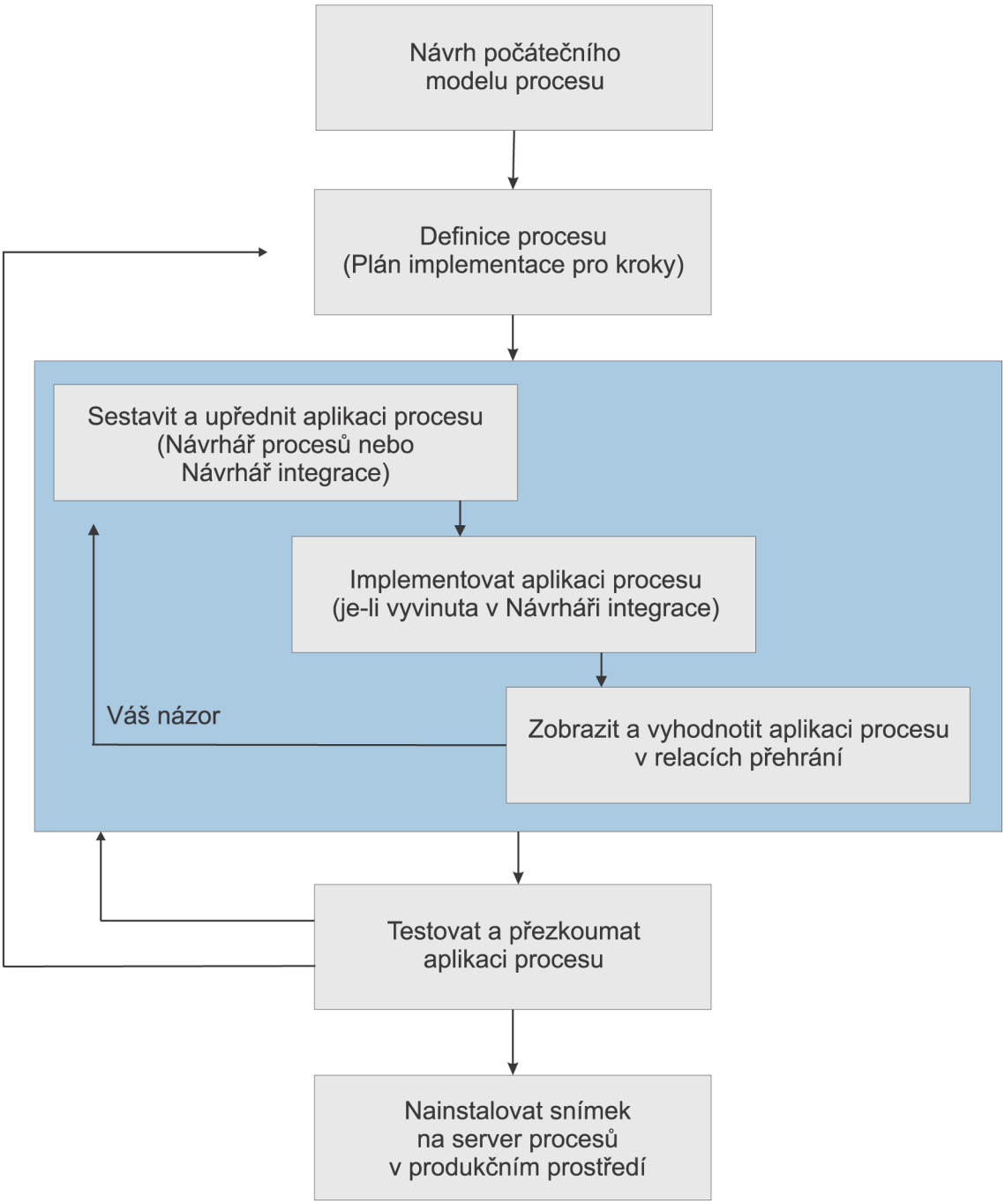
Pokud pracujete v produktu IBM Integration Designer, můžete komponentu Inspector využít, pokud je váš projekt přidružen ke komponentě Process Application. K dispozici jsou i jiné nástroje pro ladění a testování. Další informace o těchto nástrojích produktu Integration Designer viz související odkazy "Testovací moduly" a "Určování problémů pomocí ladicího programu integrace".

Instalace a správa komponent Process Application

Životní cyklus komponenty Process Application zahrnuje instalaci, administraci a odstranění implementací snímků. Součástí životního cyklu jsou i aspekty správy verzí.

Při vývoji procesů můžete plně využít iterativní přístup podporovaný nástroji v rámci produktu Process Designer. Procesy se v průběhu času vyvíjejí, a to od stavu vývoje přes testování do provozního stavu. I ve výrobě se mohou vaše procesy dál vyvíjet kvůli změnám potřeb. Příprava na životní cyklus procesů je důležitá a usnadní efektivní vytváření návrhů od úplného začátku.

Následující obrázek ilustruje iterativní přístup k vývoji procesů.



Typická konfigurace produktu Business Process Manager zahrnuje tři prostředí pro podporu vývoje a případné instalace procesů.

Prostředí	Popis
Vývoj	Sestavení a rozpracování komponent Process Application v produktu IBM Process Designer. Vytvořte modely procesů a implementujte kroky v těchto modelech pomocí komponenty Designer. Pomocí komponenty Inspector demonstrujte průběh vývoje v relacích přehrání, což umožňuje rychlé vyhodnocení a zdokonalení prototypu. Pomocí konzoly Process Center Console můžete instalovat své komponenty Process Application na testovací nebo produkční servery procesů.

Prostředí	Popis
Test	Pomocí konzoly Process Center Console můžete instalovat komponenty Process Application na server Process Server ve vašem testovacím prostředí, a implementovat tak formální testy kontroly kvality. Komponenta Inspector pomáhá ověřit a vyřešit problémy.
Provoz	Po vyřešení všech problémů ohlášených při formálním testování implementujte komponenty Process Application pomocí konzoly Process Center Console na server Process Server v produkčním prostředí. Komponenta Inspector slouží ke zjištění a vyřešení problémů ohlášených v provozním prostředí.

Chcete-li otestovat, nainstalovat, nebo spravovat snímek komponenty Process Application, který obsahuje obsah produktu IBM Business Process Manager Advanced nebo model produktu IBM Business Monitor, musí mít uživatel nebo skupina, k níž patří, přiřazeny role zabezpečení pro administraci Konfigurátor, Obsluha *a* Implementátor. Pokud právě nemáte přiřazeny všechny tyto role, klepněte v administrativní konzole WebSphere Administrative Console na volbu **Uživatelé a skupiny** a upravte role uživatelů nebo skupin. Viz "Role zabezpečení pro administraci" v souvisejících odkazech.

Strategie vydání a instalace

Chcete-li zajistit, aby implementované a instalované komponenty Process Application odpovídaly kvalitativním normám vaší organizace, zvažte možnost definování strategie vydání a instalace. Po identifikaci cílů a požadavků na vydání a instalaci nových a aktualizovaných komponent Process Application můžete automatizovat procesy nezbytné ke schválení a spuštění programů.

Můžete například směrovat proces do více různých správců v rámci různých struktur vytváření sestav v organizaci. Nový nebo aktualizovaný proces lze instalovat do provozního prostředí a poskytnout koncovým uživatelům až po podepsání všemi manažery. Kroky zahrnuté v příslušné kontrole můžete vytvořit a implementovat v produktu IBM Business Process Manager Advanced, čímž zajistíte, že budou splněny všechny podnikové směrnice a že budete mít nezbytné podpisy. Posledním krokem kontroly může být oznámení týmu IT, že schválená komponenta Process Application je připravena k instalaci.

Vytváření, přístup a začlenění služeb

Přístup k externím službám vzhledem k aplikaci

Tento scénář pojednává o různých způsobech přístupu ke službám, které jsou vzhledem k aplikaci externí, a nabízí obecné úlohy pro přístup k těmto externím službám.

Poznámka: Tento scénář se týká produktů WebSphere Enterprise Service Bus a IBM Business Process Manager. Mediační moduly lze implementovat v produktech WebSphere Enterprise Service Bus a IBM Business Process Manager. Moduly lze implementovat v produktu IBM Business Process Manager.

V integrované obchodní aplikaci vstupují *obchodní služby* do vzájemných interakcí, aby tak zajistily požadovanou funkci. Obchodní služba zajišťuje opakovatelnou funkci nebo úlohu, která přispívá k dosažení obchodního cíle. Ale práce s vyhledáním služby a připojením k ní se obchodní funkce netýká. Oddělení obchodní funkce od úlohy správy připojení ke službám zajišťuje flexibilitu řešení.

Interakce se službou začíná, když *klient služby* odešle *poskytovateli služby* požadavek na provedení obchodní funkce. Tento požadavek je odeslán ve formě *zprávy*, která definuje funkci, jež se má provést. Poskytovatel služby provede požadovanou funkci a výsledek odešle ve zprávě klientovi služby. Zpravidla je třeba zprávy zpracovat, a umožnit tak službám výměnu dat a implementaci dalších funkcí IT nízké úrovně, které jsou nezávislé na obchodních funkcích a datech. Například směrování, převodu protokolů, transformace, opakování nezdařeného vyvolání a vyvolání dynamických služeb. Toto zpracování je známé jako *mediace*.

V produktu IBM Integration Designer existují dva typy modulů: moduly (nebo moduly obchodní integrace), které jsou určeny primárně pro obchodní logiku (např. obchodní procesy, obchodní pravidla a obchodní stavové automaty), a mediační moduly, které implementují mediační toky. Ačkoli se funkce těchto dvou typů modulů zčásti překrývají, obecně se doporučuje izolovat obchodní logiku v obchodních modulech a mediační logiku provádět pomocí mediačních modulů.

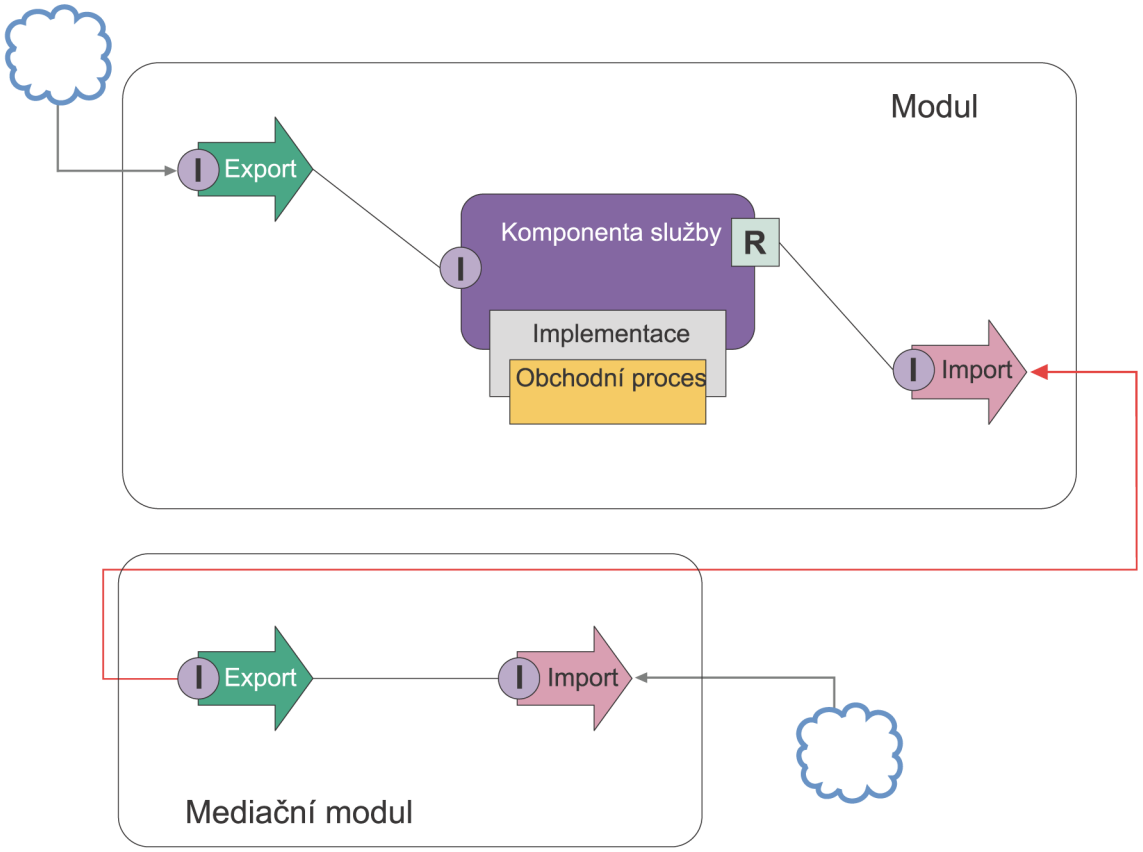
Obchodní a mediační logika však nejsou vždy jasně odděleny. V těchto případech zvažte, kolik *stavů* nebo dat v proměnných bude třeba zpracovat mezi vyvoláním různých služeb. Obecně, pokud je vyžadováno jen minimum zpracování stavů, zvažte použití komponenty mediačního toku. Pokud je třeba mezi vyvoláním různých služeb uložit stav nebo máte data, která bude nutné uložit v proměnných a zpracovat, zvažte použití komponenty obchodního procesu. Pokud například voláte více služeb a zaznamenáváte informace vracené jednotlivými službami, protože po vyvolání všech služeb chcete provést nějaké další zpracování vrácených dat, použijte obchodní proces, kde můžete vracené informace snadno přiřadit k proměnným. Jinými slovy, když máte příliš stavů, už jste v oblasti obchodní logiky.

Neexistuje jediný správný scénář integrace a neexistuje technicky chybná odpověď. Zde uvedené pokyny představují osvědčené postupy umožňující flexibilitu a opakované použití a jsou vám předkládány ke zvážení. Jako obvykle byste měli pečlivě zvážit výhody a nevýhody implementace těchto vzorků u vaší aplikace obchodní integrace. Podívejme se na některé situace.

Přístup ke komponentě SCA

Základním příkladem přístupu ke službě je situace, kdy import volá jinou komponentu SCA, aniž by vyžadoval nějaké transformace dat. I v této situaci můžete získat přístup k externí službě z mediačního modulu a nemusíte přistupovat přímo z modulu obchodního. To by v budoucnu umožnilo flexibilitu pro změny koncového bodu služby nebo kvality služby či řízení (například přidání protokolování) bez dopadů na obchodní komponenty, které danou službu přijímají. Tato architektonická šablona je známá jako "oddělení zájmů".

Než se rozhodnete tuto šablonu implementovat, zvažte její výhody oproti potenciálním účinkům zatížení, které přinese další modul. Pokud je vaším hlavním požadavkem flexibilita a chcete často provádět změny ve službě, k níž se přistupuje, zvažte použití samostatného modulu, jak je ukázáno zde. Pokud je nejdůležitější výkon a obchodní logiku jste ochotni aktualizovat a implementovat znovu, zvažte použití jednoho modulu.



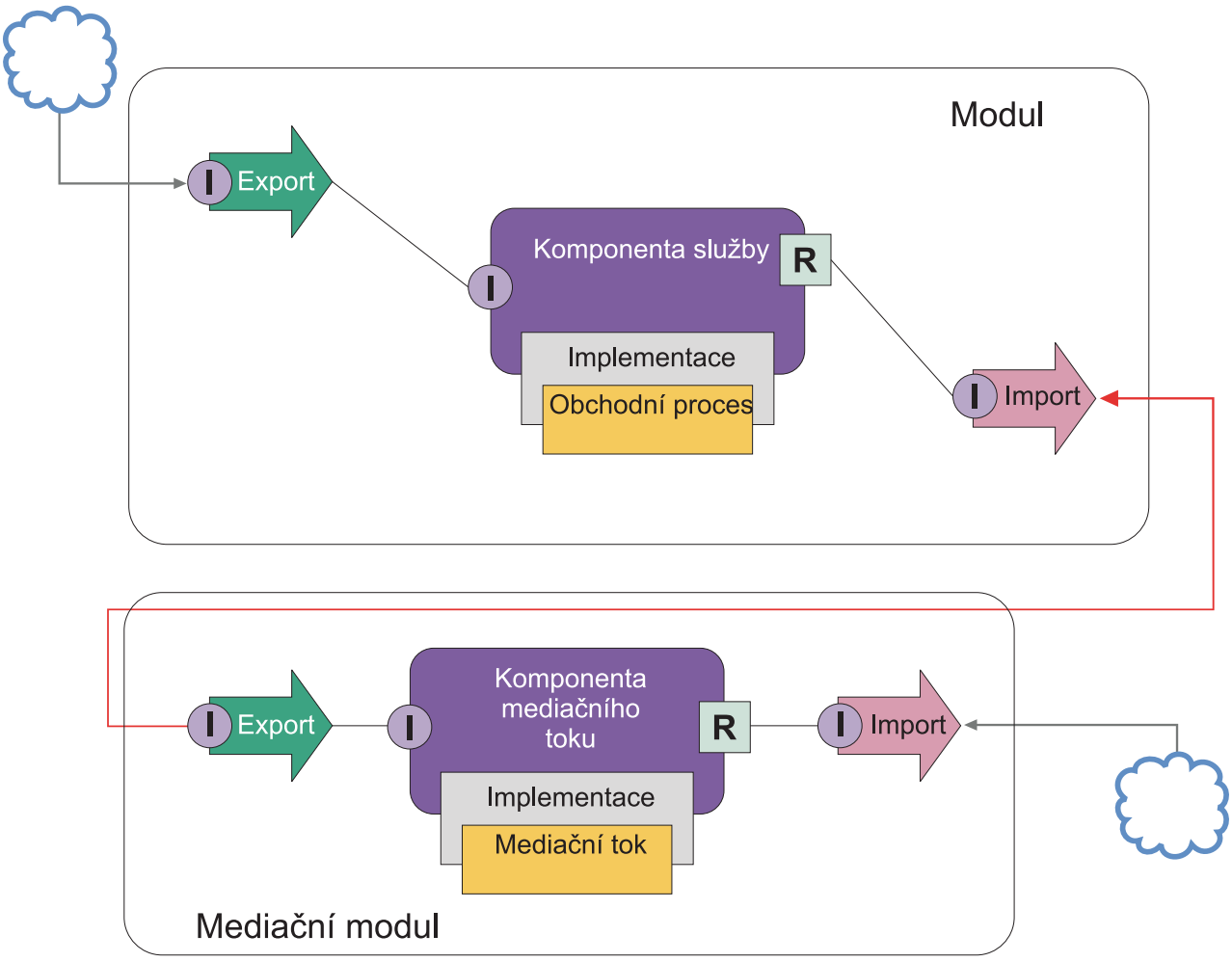
Následují obecné úlohy pro dosažení tohoto příkladu.

1. Vytvořte mediační modul. Podrobné pokyny viz téma Vytvoření mediačních modulů.
2. V mediačním modulu vytvořte import s příslušnou vazbou pro externí službu, k níž chcete přistupovat. Podrobné pokyny viz téma Vytvoření importů. Podrobnější informace o vazbách viz téma Vazby
3. Vytvořte export a dejte mu stejné rozhraní jako importu. Podrobné pokyny viz téma Vytvoření exportů.
4. Vygenerujte vazbu SCA pro export. Podrobné pokyny viz téma Generování vazeb SCA.
5. V sestavení mediačního modulu spojte export s importem. Uložte mediační modul.
6. Vytvořte modul. Podrobné pokyny viz téma Vytvoření modulu pro obchodní služby.
7. Přidejte export a komponentu.
8. V pohledu Obchodní integrace přetáhněte export, který jste vytvořili v mediačním modulu (v kroku 4), do sestavení modulu. Bude vytvořen import se stejnou vazbou jako export.
9. Spojte export s komponentou a komponentu k importu.
10. Přidejte implementaci komponenty. Informace o typech implementace viz téma Implementace.

Později můžete přidat mediační logiku, jako např. protokolování nebo směrování do mediačního modulu, aniž by to ovlivnilo obchodní modul.

Přidání mediace

Někdy nestačí jednoduše vyvolat externí službu. Někdy je třeba nejprve provést zpracování, přidáním mediačního modulu jako prostředníka mezi klientem a poskytovatelem služby.



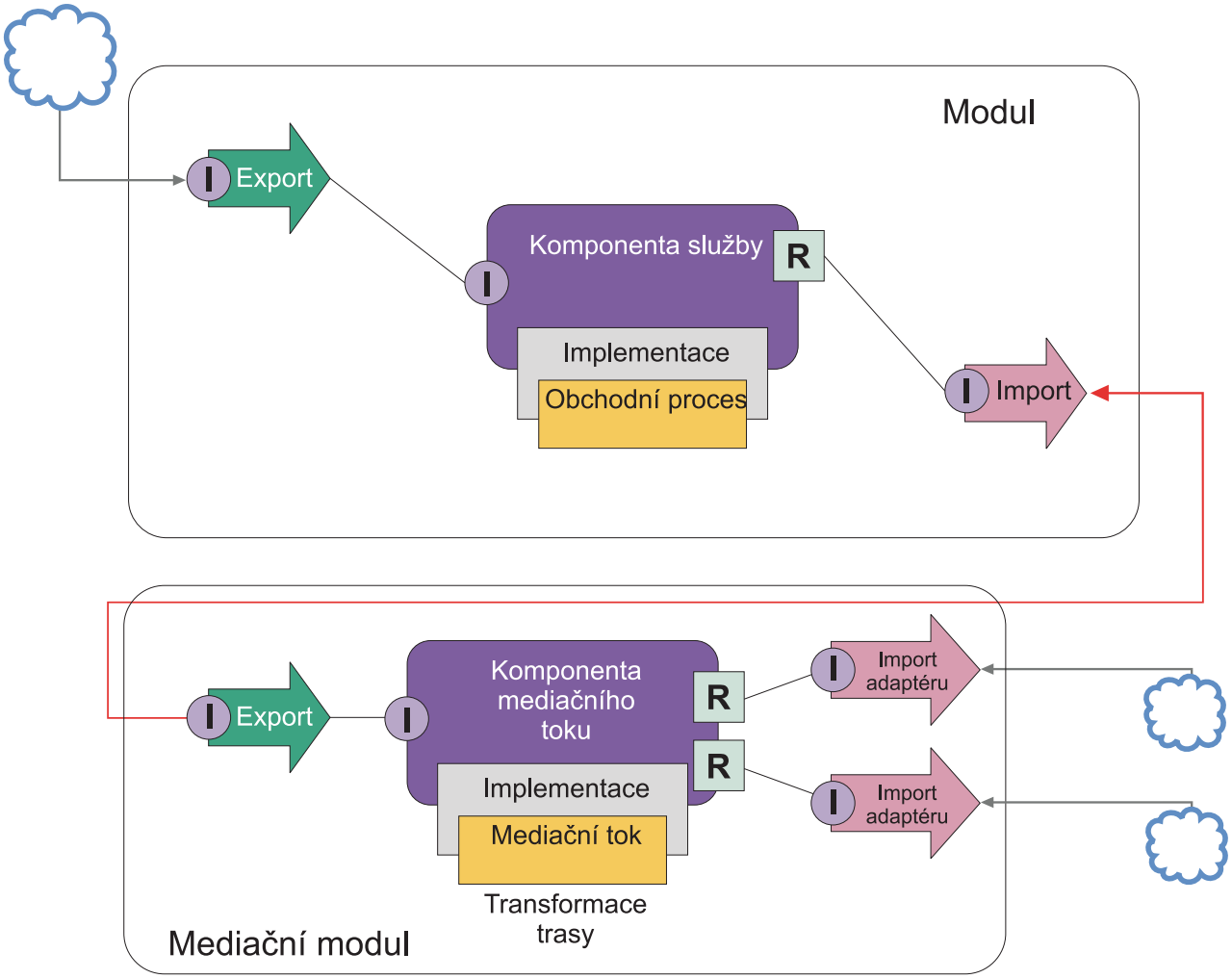
Následují některé funkce, které by prováděl mediační tok prostředníka:

- Nastavení záhlaví protokolu. Další informace viz téma Převod protokolů v Informačním centru pro sběrnici WebSphere Enterprise Service Bus.
- Transformace rozhraní nebo parametrů pomocí primitiva Mapa obchodních objektů nebo Mapování.Transformace zpráv
- Výběr konkrétní služby ze statického seznamu pomocí primitiva Filtr zpráv.Filtr zpráv
- Vyvolání více služeb pro agregaci výsledků pomocí primitiv Výstupní větvení a Vstupní větvení.Agregace a všesměrové vysílání zpráv
- Vyřešení selhání vyvolání služby pomocí opakovaného pokusu o vyvolání stejné služby nebo vyvoláním jiné služby pomocí primitiva Vyvolání služby. Opakovaný pokus o vyvolání služby po nezdaru
- Dynamické směrování výběrem služby, která se má použít, za běhu místo při integraci, což umožňuje volnější spojení služeb a rychlejší reakci podniků na změny. Nové služby lze přidávat bez nutnosti cokoli měnit v modulech, které již byly implementovány v běhovém prostředí. Dynamické směrování je nejúčinnější při použití s registrem, což vyžaduje použití mediačního primitiva Vyhledávání koncového bodu.Dynamický výběr koncového bodu

Přístup k podnikovým informačním systémům

Služby a artefakty v externích systémech lze importovat do produktu Integration Designer. Průvodce najde aplikace a data v podnikových informačních systémech (EIS) a nechá vás z nalezených aplikací a dat vygenerovat služby. Vygenerované artefakty jsou rozhraní a obchodní objekty, které mohou být využívány komponentami v modulu.

Použití mediačního modulu jako prostředníka mezi modulem a hostitelským systémem umožňuje častější opakované použití. V níže uvedeném příkladě je mediační tok použit ke směrování ke správnému hostitelskému systému a k transformaci dat do formátu, který je tímto hostitelským systémem vyžadován.



Následují obecné úlohy pro tento příklad:

1. Pomocí průvodce externí službou se připojte k hostitelskému systému. Použití průvodce externí službou pro přístup k externím službám má obdobnou strukturu bez ohledu na použitý adaptér. Informace o způsobu použití průvodce externí službou viz téma Šablona pro přístup k externím službám s pomocí adaptérů.
2. Vytvořte modul. Podrobné pokyny viz téma Vytvoření modulu pro obchodní služby.
3. Přidejte export, komponentu a import s vazbou SCA. Další informace viz téma Volání služeb.
4. Přidejte k exportu rozhraní a spojte export s komponentou.
5. Přidejte implementaci komponenty. V této implementaci nastavte vlastnost, která udává, ke které službě hostitele přistupovat. Informace o typech implementace viz téma Implementace.
6. Vytvořte mediační modul s exportem, který má vazbu SCA a stejné rozhraní jako import modulu, který jste vytvořili v kroku 2.
7. Spojte export s komponentou mediačního toku.
8. Vytvořte import pro každý hostitelský systém, k němuž chcete přistupovat, s použitím příslušného odchozího adaptéru z palety editoru sestavení.
9. Spojte komponentu mediačního toku s importy.
10. Implementujte komponentu mediačního toku. Pomocí primitiva Filtr zpráv vyberte import na základě vlastnosti nastavené v obchodní logice a použijte primitivum Mapování pro všechny importy adaptéru. Filtr zpráv.

11. V modulu vyberte export mediačního modulu jako službu, která má být do modulu importována. Podrobné informace viz téma Vyvolání služby z jiného modulu.

Později můžete provést změnu, např. přidání adaptéru nebo jeho změnu tak, aby ukazoval na jiný hostitelský systém, s minimálním dopadem na obchodní logiku.

Přístup k systémům zasílání zpráv

Aby váš modul architektury SCA (Service Component Architecture) komunikoval s existujícím klientem systému zpráv JMS, MQ nebo MQ JMS, musíte vytvořit rozhraní, obchodní objekty a vazby pro importy a exporty. Viz téma Mapování zprávy na rozhraní architektury SCA.

Mediační toky používají zprávy, které kromě obchodních objektů zajišťují přístup také k informacím o kontextu a záhlaví. Chcete-li získat přístup k informacím záhlaví JMS nebo přizpůsobené vlastnosti JMS, použijte mediační tok. Pokud provádíte integraci se systémem MQ a chcete získat přístup k informacím záhlaví MQ, použijte mediační tok.

Vytvoření nebo volání webové služby

Webové služby jsou samostatné aplikace, které provádějí obchodní funkce, od jednoduchého dotazu až po komplexní interakce obchodních procesů. Můžete volat již existující webovou službu nebo vyvinout novou webovou službu, která bude odpovídat vašim potřebám. Tento scénář popisuje příslušné kroky a odkazuje na zdroje dalších informací.

Ačkoli nemusíte všechny své služby vytvářet od základu v produktu IBM Integration Designer, některé služby tak vytvořeny budou. Při práci s editorem sestavení a editorem obchodních procesů na sestavení služeb do obchodního procesu pravděpodobně zjistíte, že některé služby chybí. Proto může být užitečné vytvořit tyto chybějící služby pomocí nástrojů IBM Integration Designer. Platí to i obráceně - po vytvoření nového procesu se můžete rozhodnout, že bude užitečné vystavit všechny operace procesu nebo některou jejich podmnožinu, jako služby, které mohou využívat ostatní.

Poznámka: Tento scénář se týká uživatelů produktů IBM Integration Designer pro produkty IBM Process Server a WebSphere Enterprise Service Bus.

Existuje několik důvodů, proč vyvíjet webové služby pomocí produktu IBM Integration Designer:

- Vytvoření služeb v produktu IBM Integration Designer umožňuje implementovat služby s pomocí obchodních pravidel.
- Vývoj v produktu IBM Integration Designer umožňuje vyvinout službu Java a vystavit ji jako webovou službu i prostřednictvím architektury SCA.
- Výhodou je mapování rozhraní bez nutnosti psaní kódu. Z kódu Java můžete vzít všechna mapování dat a nechat pro vývojáře v jazyce Java jednoduchý program Java ve formě černé skříňky.
- IBM Integration Designer ukazuje všechny služby a vztahy na jednom místě.
- Schopnost refaktorovat pomůže také při vývoji webových služeb pomocí produktu IBM Integration Designer.

Nezapomeňte, že by webové služby neměly být považovány za řešení všech vašich problémů s integrací. Ale stejně jako u všech ostatních technologických nebo architektonických přístupů existují i zde podstatné výhody, když jsou webové služby používány ve správný čas a na správném místě.

Exporty, importy a vazby

Produkt IBM Integration Designer umožňuje importovat standardní webové služby a využívat je ve složených aplikacích.

V produktu IBM Integration Designer použijte k vývoji služeb editor sestavení. Postupujte podle standardního procesu pro vytváření modulů, mediačních modulů, knihoven a komponent. Potom můžete pomocí exportů, importů a vazeb tyto služby sdílet a přistupovat k nim. Kroky těchto základních úloh jsou uvedeny níže a odkazy vedou k podrobnějším informacím o jednotlivých úlohách.

Pro webové služby můžete použít jednu ze dvou vazeb - vazbu webové služby nebo vazbu HTTP. Vazba webové služby poskytuje specifikaci pro přenos zpráv k webové službě a od ní. Vazbu webové služby vám nástroje pomohou vygenerovat automaticky. Vazba HTTP je standardní protokol požadavku a odezvy mezi klienty a serverem, jak jej definuje protokol HTTP publikovaný konsorciem W3C (World Wide Web). Pokud používáte vazbu HTTP, musíte uvést nějaké počáteční informace o konfiguraci vázání.

1. Vytvořte export pro publikování služby modulu, kterou budou moci používat jiné moduly.
2. Vygenerujte vazbu pro tento export.
   - Vygenerujte pro tento export vazbu webové služby.
   - Vygenerujte vazbu HTTP exportu.
3. Vytvořte import pro volání existující služby, která není součástí modulu, jenž sestavujete.
   - Vygenerujte pro tento import vazbu webové služby.
   - Generování vazby HTTP importu.

Přečtěte si odkazované téma, chcete-li vyvolat webovou službu ze stránek JSP.

Schopnosti vývoje webových služeb

Při otevření editoru přidruženého k procesu tvorby webových služeb se může zobrazit okno Potvrdit zpřístupnění s touto informací:

```
Tato akce vyžaduje zpřístupnění "Implementace webových služeb".
Chcete povolit požadovanou funkci?
```

Produkt IBM Integration Designer nabízí filtrovací funkci známou jako *schopnosti*. V nastavení Předvolby jsou funkce a nástroje kategorizovány do schopností a je možné povolit nebo zakázat kategorie schopností nebo podmnožinu funkcí z libovolné kategorie. Další informace viz Schopnosti.

Další informace o klíčových konceptech

Tento oddíl použijte jako výchozí bod pro zkoumání technologií, které se používají v produktu IBM Business Process Manager a které tento produkt využívají.

Scénáře tvorby obsahu

Pomocí scénářů lze porozumět komponentám a produktům z řady pro řízení BPM a pracovat s nimi.

Správa verzí

Životní cyklus komponenty Process Application začíná jejím vytvořením a pokračuje cyklem její aktualizace, implementace, souběžné implementace, odstranění implementace a archivace. *Správa verzí* je mechanizmus správy životního cyklu komponenty Process Application, který provede jedinečnou identifikaci jednotlivých verzí této komponenty.

Způsob fungování správy verzí v produktu IBM Business Process Manager závisí na tom, co implementujete - komponentu Process Application implementovanou z úložiště produktu IBM Process Center nebo podnikovou aplikaci implementovanou přímo z produktu IBM Integration Designer.

Komponenty Process Application a Toolkit, které implementujete do běhového prostředí z komponenty Process Center jsou standardně opatřeny verzí. V případě podnikových aplikací můžete moduly a knihovny opatřit verzí v produktu IBM Integration Designer.

Navíc můžete vytvořit verze lidské úlohy či stavového automatu, aby tak mohlo v běhovém prostředí současně existovat více verzí dané úlohy či stavového automatu.

Správa verzí komponent Process Application

Díky správě verzí dokáže běhové prostředí identifikovat snímky v životním cyklu komponenty Process Application a může souběžně spouštět více snímků ve stejný čas.

Na komponentu Process Application se můžete dívat jako na kontejner. Všechny snímky, implementace a správa verzí se řeší na úrovni kontejneru, nikoli na úrovni jednotlivých artefaktů v tomto kontejneru. Snímky se spravují prostřednictvím konzoly Process Center Console.

Změny se ukládají dynamicky do úložiště komponenty Process Center na vrchol, který je aktuální pracovní verzí komponenty Process Application. Komponenta Process Application zůstává na této úrovni vrcholu tak dlouho, dokud se nerozhodnete vytvořit snímek (sn1). Snímek komponenty Process Application lze implementovat na server komponenty Process Center nebo na Process Server pro účely testování, fázování nebo produkce.

Pokud provedete změny a chcete implementovat novou verzi, musíte vytvořit nový snímek (sn2). Při implementaci snímku sn2 můžete snímek sn1 buď odebrat, nebo jej ponechat spuštěný na serveru.

Kontext verze

Kontextem verze jsou metadata identifikující určitou verzi. Identifikátor přiřazujete vy sami, nicméně IBM doporučuje používat tříciferný způsob označování verzí ve formátu <hlavní>.<vedlejší>.<služba>. Podrobnější popis tohoto schématu číslování verzí najdete v tématech o konvencích pojmenování.

Produkt IBM Business Process Manager přiřazuje každé komponentě Process Application globální obor názvů. Globálním oborem názvů je buď vrchol komponenty Process Application, nebo konkrétní snímek komponenty Process Application. Název verze používaný serverem nesmí obsahovat více než sedm znaků, a tak přiřazený název bude akronymem s použitím znaků z názvu snímku, který jste přiřadili. Akronymy snímků se budou shodovat s příslušnými názvy snímků - za předpokladu, že názvy snímků splňují doporučený styl IBM VRM a nejsou delší než sedm znaků. Například název snímku 1.0.0 bude mít akronym 1.0.0, resp. název snímku 10.3.0 bude mít akronym 10.3.0. U akronymu snímku je zaručena jedinečnost v kontextu komponenty Process Application v rámci rozsahu serveru komponenty Process Center. Z tohoto důvodu nemůžete akronym snímku měnit.

Správa verzí komponent Process Application a Toolkit produktu Process Designer

Chcete-li spravovat verze komponent Process Application a Toolkit v úložišti komponenty Process Center, můžete ukládat snímky a přiřazovat jim názvy. Poté můžete například navzájem porovnat určitý snímek s jiným a zjistit, jaké jsou mezi nimi rozdíly. Opraví-li například vývojář určitý problém se službou, pořídí v tomto bodě snímek nadřízené komponenty Process Application nebo Toolkit a poté přijde jiný vývojář, provede několik dalších změn ve stejné službě a pořídí další snímek. Správce projektu může oba snímky porovnat a zjistit, které změny kdy kdo provedl. Zjistí-li správce projektu, že dodatečně provedené změny nevedly k požadovaným výsledkům, může se vrátit zpět ke snímku původní opravy.

Snímek komponenty Process Application zpravidla pořizujete pokaždé, když jste připraveni nebo téměř připraveni implementovat do produkčního prostředí nebo testovat integraci. Snímek komponenty Process Application musíte pořídit v případě, že chcete implementovat na samostatný Process Server. V případě komponent Toolkit je situace poněkud odlišná; snímek komponenty Toolkit pořizujete ve chvíli, kdy je daná komponenta Toolkit připravena k použití komponentami Process Application. Pokud později chcete komponentu Toolkit aktualizovat, musíte v příslušném okamžiku pořídit nový snímek "vrcholu", přičemž vlastníci komponent Process Application a Toolkit se rozhodnou, zda chtějí na nový snímek přejít. Jako vrchol označujeme speciální a zároveň jediný snímek, ve kterém lze měnit obsah, ale který můžete spouštět pouze na serveru komponenty Process Center. Na serveru Process Server nemůžete vrchol implementovat.

Komponenty Process Application ve více klastrech

Jednu verzi komponenty Process Application můžete implementovat na více klastrech v rámci jedné buňky. Chcete-li rozlišovat mezi více implementacemi stejné verze komponenty Process Application, vytvořte pro každou implementaci

snímek a použijte v názvu snímku jedinečný identifikátor v rámci buňky (například v1.0_cell1_1 a v1.0_cell1_2). Pokud bychom chtěli být zcela přesní, každý snímek je novou verzí komponenty Process Application (čistě z pohledu správy životního cyklu), ale obsah a funkce jsou stejné.

Při implementaci komponenty Process Application na klastr se provádí automatická synchronizace uzlů.

Správa verzí modulů a knihoven

Pokud se modul či knihovna nachází v komponentě Process Application nebo Toolkit, přebírá životní cyklus těchto komponent (verze, snímky, stopy atd.). Názvy modulů a knihoven musí být v rámci rozsahu komponenty Process Application či Toolkit jedinečné.

Toto téma popisuje správu verzí modulů a knihoven používaných s komponentami Process Application. Poznámka: Pokud nicméně implementujete moduly na komponentu Process Server přímo z produktu IBM Integration Designer, můžete i nadále postupovat tak, že přiřadíte modulům čísla verzí během implementace, jak je popsáno v tématu "Vytvoření modulů a knihoven s verzemi".

Modul či knihovna přidružená k produktu IBM Process Center musí mít své závislé knihovny ve stejné komponentě Process Application nebo v závislé komponentě Toolkit.

V následující tabulce jsou vypsány možné výběry v editoru závislostí z produktu IBM Integration Designer při přidružení knihovny ke komponentě Process Application či Toolkit:

Tabulka 3. Závislosti pro knihovny Modulu, komponent Process Application či Toolkit a Globální knihovnu

Rozsah knihovny	Popis	Může záviset na . . .
Modul	Na serveru existuje kopie této knihovny pro každý modul, který ji používá.	Knihovna s oborem modulu může záviset na všech typech knihoven.
komponenty Process Application či Toolkit	Tato knihovna je sdílena mezi všemi moduly v rámci oboru komponenty Process Application či Toolkit. Toto nastavení se projeví, pokud implementace probíhá prostřednictvím produktu IBM Process Center. Pokud implementace probíhá mimo produkt IBM Process Center, dojde ke zkopírování knihovny do každého modulu. **Poznámka:** Knihovny vytvořené v produktu IBM Integration Designer verze 8 mají standardně nastavenu úroveň sdílení na volbu **komponenta Process Application či Toolkit**.	Knihovna tohoto typu může záviset pouze na globálních knihovnách.
Globální	Tato knihovna je sdílena mezi všemi spuštěnými moduly.	Globální knihovna může být závislá pouze na jiných globálních knihovnách. **Poznámka:** Abyste mohli implementovat globální knihovnu, musíte nakonfigurovat sdílenou knihovnu WebSphere. Další informace viz "Závislosti modulů a knihoven".

Moduly a knihovny přidružené ke komponentám Process Application nebo Toolkit

Modulům a knihovnám přidruženým ke komponentám Process Application nebo Toolkit nemusíte opatřit verzi.

Moduly a knihovny, které jsou přidruženy ke komponentě Process Application nebo Toolkit, nemusí být opatřeny verzí. Ve skutečnosti nemůžete v editoru závislostí vytvořit verzi modulu nebo knihovny přidružené ke komponentě Process Application nebo Toolkit. Moduly a knihovny přidružené ke komponentě Process Application nebo Toolkit používají snímky a funkci v produktu Process Center, aby dosáhly stejného výsledku jako verze.

Knihovny přidružené s komponentou Process Application nebo Toolkit nebudou mít požadované číslo verze v části editoru závislostí Knihovny, protože verze není potřeba.

Konvence pojmenování

Konvence pojmenování se používá pro rozrůznění možných verzí komponenty Process Application při jejím průchodu svým životním cyklem, který zahrnuje aktualizace, implementace, souběžné implementace, zrušení implementací či archivace.

V tomto oddílu naleznete konvence používané k jedinečné identifikaci verzí komponenty Process Application.

Kontext verze je kombinace akronymů, sloužící jako jedinečný popis komponenty Process Application či Toolkit. Každý typ akronymu má svou konvenci pojmenování. Akronym je omezený maximální délkou sedmi znaků ze znakové sady [A-Z0-9_], s výjimkou akronymu snímku, který může obsahovat také tečku.

- Akronym komponenty Process Application je vytvořen při vytvoření této komponenty. Jeho délka nesmí přesáhnout sedm znaků.
- Akronym snímku je vytvořen automaticky při vytvoření snímku. Jeho délka nesmí přesáhnout sedm znaků.

  Pokud název akronymu splňuje kritéria platného akronymu snímku, název snímku a akronym jsou shodné.

  Poznámka: S použitím funkce trasování s ohledem na verzi komponenty mediačního toku pojmenujte svůj snímek tak, aby odpovídal schématu <*verze*>.<*vydání*>.<*úprava*> (např. **1.0.0**). Vzhledem k tomu, že je délka akronymu snímku omezena na sedm znaků, jsou číselné hodnoty omezeny na maximálně pět číslic (pět číslic a dvě tečky). Při zvyšování hodnot polí s číslicemi proto postupujte obezřetně, protože cokoliv, co přesáhne prvních sedm znaků, bude oříznuto.

  Například název snímku **11.22.33** povede k akronymu snímku **11.22.3**.
- Akronym stopy se automaticky generuje z prvního znaku každého slova názvu dané stopy. Například nově vytvořená stopa s názvem **Moje Nová Stopa** by vedla k hodnotě akronymu **MNS**.

  Výchozí název stopy a akronym jsou **Hlavní**. Implementace na server produktu IBM Process Center obsahuje akronym stopy v kontextu správy verzí, pokud akronym stopy není **Hlavní**.

Definici obchodního procesu z komponenty Process Application zpravidla definuje akronym názvu komponenty Process Application, akronym snímku a název definice obchodního procesu. Kdykoliv to bude možné, vybírejte pro své definice obchodních procesů jedinečné názvy. V případě výskytu duplicitních názvů můžete narazit na následující problémy:

- Definice obchodních procesů může být možné vystavit jako webové služby pouze prostřednictvím určité mediace.
- Může se stát, že nebudete moci vyvolat definici obchodního procesu vytvořenou v produktu IBM Process Designer z procesu BPEL vytvořeného v produktu IBM Integration Designer.

Kontext verze je proměnlivý v závislosti na způsobu implementace komponenty Process Application.

Konvence pojmenování pro implementace serveru Process Center:

Na server IBM Process Center můžete implementovat snímek komponenty Process Application, stejně jako snímek komponenty Toolkit. Navíc můžete také implementovat vrchol komponenty Process Application nebo vrchol komponenty Toolkit. (*Vrchol* je aktuální pracovní verze vaší komponenty Process Application či Toolkit.) Kontext verze se různí podle typu implementace.

V případě komponent Process Application se vrchol či specifický snímek používá jako jedinečný identifikátor verze.

Komponenty Toolkit lze implementovat s jednou či více komponentami Process Application, ale životní cyklus každé komponenty Toolkit je svázán s životním cyklem dané komponenty Process Application. Každá komponenta Process Application obsahuje vlastní kopii závislé komponenty Toolkit či komponent Toolkit implementovaných na serveru. Implementovaná komponenta Toolkit není sdílena mezi komponentami Process Application.

Pokud je stopa přidružená ke komponentě Process Application pojmenovaná jinak než výchozím názvem **Hlavní**, je akronym trasy také součástí kontextu dané verze.

Snímky komponenty Process Application

V případě implementací snímků komponenty Process Application je kontext verze kombinací následujících položek:
- Akronym názvu komponenty Process Application.
- Akronym stopy komponenty Process Application (v případě použití jiné stopy než **Hlavní**).
- Akronym snímku komponenty Process Application.

Samostatné komponenty Toolkit

V případě implementací snímků komponent Toolkit je kontext verze kombinací následujících položek:
- Akronym názvu komponenty Toolkit.
- Akronym stopy komponenty Toolkit (v případě použití jiné stopy než **Hlavní**).
- Akronym snímku komponenty Toolkit.

Tipy

Během iterativního testování v produktu Process Designer se používají vrcholy komponenty Process Application. Lze je implementovat pouze na servery Process Center.

V případě implementací vrcholu komponenty Process Application je kontext verze kombinací následujících položek:
- Akronym názvu komponenty Process Application.
- Akronym stopy komponenty Process Application (v případě použití jiné stopy než **Hlavní**).
- "Vrchol".

Vrcholy komponenty Toolkit se používají také během iterativního testování v produktu Process Designer. Neimplementují se na produkční server.

V případě implementací vrcholu komponenty Toolkit je kontext verze kombinací následujících položek:
- Akronym názvu komponenty Toolkit.
- Akronym stopy komponenty Toolkit (v případě použití jiné stopy než **Hlavní**).
- "Vrchol".

Příklady

Prostředky by měly být opatřeny jedinečným názvem a externě identifikovány pomocí kontextu verze.
- V následující tabulce jsou zobrazeny příklady názvů, které jsou identifikovány jedinečně. V tomto příkladu používá vrchol komponenty Process Application výchozí název stopy (**Hlavní**):

Tabulka 4. Vrchol komponenty Process Application s výchozím názvem stopy

Typ názvu	**Příklad**
Název komponenty Process Application	**Process Application 1**
Akronym názvu komponenty Process Application	**PA1**
Stopa komponenty Process Application	**hlavní**
Akronym stopy komponenty Process Application	“” (když je stopa **Hlavní**)
Snímek komponenty Process Application	
Akronym snímku komponenty Process Application.	**Rada**

Všechny moduly SCA přidružené k tomuto vrcholu komponenty Process Application zahrnují daný kontext verze, jak je uvedeno v následující tabulce:

Tabulka 5. Moduly SCA a soubory EAR s ohledem na verzi

Název modulu SCA	Název s ohledem na verzi	Název souboru EAR/aplikace s ohledem na verzi
M1	PA1-Rada-M1	PA1-Rada-M1.ear
M2	PA1-Rada-M2	PA1-Rada-M2.ear

- V následující tabulce je zobrazen příklad vrcholu komponenty Process Application, která používá jiný než výchozí název stopy:

Tabulka 6. Vrchol komponenty Process Application s jiným než výchozím názvem stopy

Typ názvu	Příklad
Název komponenty Process Application	**Process Application 1**
Akronym názvu komponenty Process Application	**PA1**
Stopa komponenty Process Application	**Stopa1**
Akronym stopy komponenty Process Application	**T1**
Snímek komponenty Process Application	
Akronym snímku komponenty Process Application.	**Rada**

Všechny moduly SCA přidružené k tomuto vrcholu komponenty Process Application zahrnují daný kontext verze, jak je uvedeno v následující tabulce:

Tabulka 7. Moduly SCA a soubory EAR s ohledem na verzi

Název modulu SCA	Název s ohledem na verzi	Název souboru EAR/aplikace s ohledem na verzi
M1	PA1-T1-Rada-M1	PA1-T1-Rada-M1.ear
M2	PA1-T1-Rada-M2	PA1-T1-Rada-M2.ear

Konvence pojmenování pro implementace komponenty Process Server:

Na komponentu Process Server můžete implementovat snímek komponenty Process Application. Akronym snímku komponenty Process Application se používá za účelem jedinečné identifikace verze.

V případě implementací snímků komponenty Process Application je kontext verze kombinací následujících položek:

- Akronym názvu komponenty Process Application.
- Akronym snímku komponenty Process Application.

Prostředky by měly být opatřeny jedinečným názvem a externě identifikovány pomocí kontextu verze. V následující tabulce jsou zobrazeny příklady názvů, které jsou identifikovány jedinečně:

Tabulka 8. Příklad názvů a akronymů

Typ názvu	Příklad
Název komponenty Process Application	**Process Application 1**
Akronym názvu komponenty Process Application	**PA1**
Snímek komponenty Process Application	**1.0.0**
Akronym snímku komponenty Process Application.	**1.0.0**

V případě prostředku, jako např. modulu či knihovny, je kontext verze součástí jeho identifikace.

V následující tabulce je zobrazen příklad dvou modulů a také zahrnutí kontextu verze do přidružených souborů EAR:

Tabulka 9. Moduly SCA a soubory EAR s ohledem na verzi

Název modulu SCA	Název s ohledem na verzi	Název souboru EAR/aplikace s ohledem na verzi
M1	PA1-1.0.0-M1	PA1-1.0.0-M1.ear
M2	PA1-1.0.0-M2	PA1-1.0.0-M2.ear

V následující tabulce je zobrazen příklad dvou knihoven s oborem komponenty Process Application a také zahrnutí kontextu verze do přidružených souborů JAR:

Tabulka 10. Knihovny s oborem komponenty Process Application a soubory JAR s ohledem na verzi

Název knihovny s oborem komponenty Process Application architektury SCA	Název s ohledem na verzi	Název souboru JAR s ohledem na verzi
Lib1	PA1-1.0.0-Lib1	PA1-1.0.0-Lib1.jar
Lib2	PA1-1.0.0-Lib2	PA1-1.0.0-Lib2.jar

Vazba s ohledem na verzi

Komponenty Process Application mohou obsahovat moduly SCA zahrnující vazby exportu a importu. V případě souběžné implementace aplikací musí být vazba pro každou verzi dané aplikace jedinečná. Některé vazby jsou během implementace automaticky aktualizovány, aby tak byla mezi verzemi zajištěna jedinečnost. Jindy je v zájmu zajištění jedinečnosti vazby nutné po implementaci tuto vazbu aktualizovat.

Rozsah vazby *s ohledem na verzi* je vymezený pro konkrétní verzi komponenty Process Application, což zaručuje její jedinečnost mezi těmito komponentami. Následující oddíl popisuje vazby, které jsou automaticky aktualizovány tak, aby braly ohled na verzi, stejně jako akce, které je třeba provést za běhu v případě, že vazba nefunguje s ohledem na verzi. Informace o bodech, které je třeba zvážit při vytváření modulů, viz “Aspekty vytváření vazeb”.

SCA

Pokud jsou vazby exportu a importu daného modulu definovány ve stejném oboru komponenty Process Application, dojde během implementace k automatickému přejmenování cíle vazby SCA na takový, který bude brát ohled na verzi.

Pokud dané vazby ve stejném oboru komponenty Process Application definovány nejsou, dojde k zaprotokolování informační zprávy. Po implementaci musíte upravit vazbu importu za účelem změny cílové adresy koncového bodu. Cílovou adresu koncového bodu můžete změnit prostřednictvím administrativní konzoly.

Webová služba (JAX-WS nebo JAX-RPC)

Cílová adresa koncového bodu vazby webové služby je během implementace automaticky přejmenována tak, aby brala ohled na verzi, pokud jsou všechny následující podmínky pravdivé:

- V případě adresy jste postupovali v souladu s výchozí konvencí pojmenování:
  http://*ip*:*port*/*NázevModulu*Web/sca/*Název Exportu*
- Adresa koncového bodu je SOAP/HTTP.
- Vazby exportu a importu daného modulu byly definovány v rámci stejného oboru komponenty Process Application.

Pokud tyto podmínky pravdivé nejsou, dojde k zaprotokolování informační zprávy. Následující akce potom závisí na způsobu implementace vaší komponenty Process Application:

- Pokud souběžně implementujete svoji komponentu Process Application, musíte ručně přejmenovat adresu URL koncového bodu SOAP/HTTP nebo cílovou frontuSOAP/JMS tak, aby byla mezi verzemi dané komponenty Process Application jedinečná. Po implementaci můžete cílovou adresu koncového bodu změnit prostřednictvím administrativní konzoly.
- Pokud implementujete pouze jedinou verzi komponenty Process Application, můžete tuto zprávu ignorovat

V případě souběžné implementace snímku vazby webové služby SOAP/JMS se typ akce, kterou budete provádět, odvíjí od způsobu implementace vaší komponenty Process Application:

- Pokud se import a cílový export nachází ve stejné komponentě Process Application, postupujte před publikováním komponenty Process Application do komponenty Process Center a vytvořením snímků takto:
  1. Změňte adresu URL koncového bodu exportu. Ujistěte se, že jsou cíl a továrna připojení jedinečné.
  2. Změňte adresu URL koncového bodu importu tak, aby se shodovala s adresou, kterou jste v předchozím kroku určili pro export.
- Pokud se import a cílový export nachází v různých komponentách Process Application, postupujte takto:
  1. Změňte adresu URL koncového bodu exportu. Ujistěte se, že jsou cíl a továrna připojení jedinečné.
  2. Publikujte komponentu Process Application do komponenty Process Center.
  3. Vytvořte snímek.
  4. Implementujte komponentu Process Application na komponentu Process Server.
  5. Pomocí administrativní konzoly WebSphere Administrative Console změňte adresu URL koncového bodu odpovídajícího importu tak, aby se shodovala s adresou určenou pro export.

HTTP

Adresa URL koncového bodu vazby HTTP je během implementace automaticky přejmenována tak, aby brala ohled na verzi, pokud jsou všechny následující podmínky pravdivé:

- V případě adresy jste postupovali v souladu s výchozí konvencí pojmenování:

  http(s)://*ip*:*port*/*NázevModulu*Web/*kontextováCestavExportu*
- Vazby exportu a importu daného modulu byly definovány v rámci stejného oboru komponenty Process Application.

Pokud tyto podmínky pravdivé nejsou, dojde k zaprotokolování informační zprávy. Následující akce potom závisí na způsobu implementace vaší komponenty Process Application:

- Pokud souběžně implementujete svoji komponentu Process Application, musíte ručně přejmenovat adresu URL koncového bodu tak, aby byla mezi verzemi dané komponenty Process Application jedinečná. Po implementaci můžete cílovou adresu koncového bodu změnit prostřednictvím administrativní konzoly.
- Pokud implementujete pouze jedinou verzi komponenty Process Application, můžete tuto zprávu ignorovat.

Platforma JMS a generická platforma JMS

Systémem generovaná služba JMS a generické vazby JMS automaticky berou ohled na verzi.

Poznámka: V případě služby JMS a generických vazeb služby JMS definovaných uživatelem nedochází během implementace k automatickému přejmenování, aby tak mohly vazby brát ohled na verzi. Pokud je vazba definovaná uživatelem, musíte následující atributy přejmenovat tak, aby byly mezi verzemi dané komponenty Process Application jedinečné:

- Konfigurace koncového bodu.
- Cílová fronta příjmu.
- Název portu modulu listener (pokud je definovaný).

Pokud změníte koncový bod cílového modulu, nastavte odpovídající cíl odeslání.

MQ/JMS a MQ

Během implementace nedochází k automatickému přejmenování, aby tak mohly vazby typu MQ/JMS či MQ brát ohled na verzi.

Následující atributy musíte přejmenovat tak, aby byly mezi verzemi dané komponenty Process Application jedinečné:

- Konfigurace koncového bodu.
- Cílová fronta příjmu.

Pokud změníte koncový bod cílového modulu, nastavte odpovídající cíl odeslání.

EJB

Během implementace nedochází k automatickému přejmenování, aby tak mohly vazby typu EJB brát ohled na verzi.

Atribut Názvy rozhraní JNDI musíte přejmenovat tak, aby byl mezi verzemi dané komponenty Process Application jedinečný.

Poznámka: Je třeba aktualizovat také klientské aplikace, aby používali nové názvy rozhraní JNDI.

EIS

Adaptér prostředku je během implementace automaticky přejmenován tak, aby bral ohled na verzi, pokud ovšem nebyl změněn výchozí název prostředku (***NázevModulu*App:Adapter Description**).

Pokud výchozí název prostředku změněn byl, musí být názvy adaptérů prostředku mezi verzemi dané komponenty Process Application jedinečné.

Pokud názvy adaptérů prostředku jedinečné nejsou, dojde během implementace k zaprotokolování informační zprávy, která vás na tuto skutečnost upozorní. Pomocí administrativní konzoly můžete adaptéry prostředku přejmenovat po dokončení implementace ručně.

Dynamické vyvolání s ohledem na verzi

Můžete definovat komponenty mediačního toku, které budou směrovat zprávy na koncové body, které jsou určovány dynamicky za běhu. Při vytváření mediačního modulu konfigurujete vyhledání koncového bodu tak, aby používalo trasování s ohledem na verzi.

Pokud pro snímek používáte styl IBM_VRM (<*verze*>.<*vydání*>.<*úprava*), můžete exportovat soubor EAR komponenty Process Application do produktu WSRR (WebSphere Service Registry and Repository). Když konfigurujete mediační modul, konfigurujete vyhledání koncového bodu tak, aby používalo směrování s ohledem na verzi. Například v poli **Zásada shody** vyberete volbu **Vrátit koncový bod odpovídající nejnovější kompatibilní verzi služeb založených na modulu SCA** a pro položku **Typ vazby** vyberete volbu **SCA**.

Budoucí verze komponenty Process Application budou implementovány na server a publikovány do produktu WSRR. Vyhledávání koncového bodu mediačního modulu dynamicky vyvolá nejnovější kompatibilní verzi koncového bodu dané služby.

Poznámka: Můžete také nastavit cíl v záhlaví SMOHeader a příslušnou hodnotu může nést zpráva požadavku.

Implementace komponent Process Application s moduly a projekty jazyka Java

Komponenty Process Application mohou obsahovat vlastní moduly Java EE a projekty Java. V případě souběžné implementace aplikací musí být vlastní modul Java EE pro každou verzi dané aplikace jedinečný.

Poznámka: Vlastní moduly Java EE a projekty Java jsou implementovány na server v případě, že jsou implementovány s modulem SCA s deklarovanou závislostí. Pokud při deklaraci dané závislosti nevyberete volbu **Implementovat s modulem** (což je výchozí nastavení), musíte modul či projekt implementovat ručně.

Implementace komponent Process Application s obchodními pravidly a selektory

Pokud implementujete více verzí komponenty Process Application, které obsahují komponentu obchodního pravidla nebo selektoru, všímejte si způsobu, jakým tyto verze používají přidružená metadata.

Dynamická metadata komponenty obchodního pravidla či selektoru se definují za běhu na základě názvu, cílového oboru názvů a typu dané komponenty. Pokud dojde k implementaci dvou či více verzí komponenty Process Application obsahujících obchodní pravidlo či selektor do stejného běhového prostředí, budou tyto sdílet stejná metadata logiky pravidla (obchodní pravidlo) nebo směrování (selektor).

Chcete-li umožnit, aby každá verze komponenty obchodního pravidla či selektoru v rámci komponenty Process Application používala vlastní dynamická metadata (logiky pravidla či směrování), refaktorujte cílový obor názvů tak, aby byl pro každou verzi komponenty Process Application jedinečný.

Architektura implementace

Architektura implementace produktu IBM Business Process Manager sestává ze softwarových procesů, nazývaných servery, topologických jednotek, na které je odkazováno jako na uzly a buňky, a z úložiště konfigurací používaného k ukládání informací o konfiguraci.

Buňky

V rámci produktu IBM Business Process Manager jsou *buňky* logická seskupení jednoho či více uzlů rozdělené sítě.

Buňka je konceptem konfigurace. Je to způsob, kterým administrátoři vzájemně logicky přidružují uzly. Na základě specifických kritérií, která dávají smysl v rámci organizačních prostředí, vytvoří administrátoři uzly, ze kterých daná buňka sestává.

Konfigurační data administrace jsou uložena v souborech XML. Buňka uchovává soubory hlavní konfigurace každého serveru v každém svém uzlu. Každý uzel a server má také vlastní lokální konfigurační soubory. Změny v lokálním uzlu nebo v konfiguračním souboru serveru jsou dočasné, pokud tento server patří k dané buňce. Aktuálně platné lokální změny potlačují konfiguraci buňky. Změny provedené v konfiguračních souborech hlavního serveru a hlavního uzlu na úrovni buňky nahrazují jakékoli dočasné změny provedené v uzlu při jeho synchronizaci s konfiguračními dokumenty buňky. K synchronizaci dochází při určitých událostech, jako např. při spuštění serveru.

Servery

Jádro funkčnosti produktu IBM Business Process Manager poskytují servery. Servery Process Server rozšiřují či zvyšují schopnost aplikačního serveru zpracovávat moduly architektury SCA (Service Component Architecture). Jiné servery (správci implementace a agenti uzlu) se používají ke správě serverů Process Server.

Server Process server může být buď *samostatný server*, nebo *spravovaný server*. Spravovaný server může volitelně být členem *klastru*. Kolekce spravovaných serverů, serverových klastrů a jiného middleware se nazývá *prostředí implementace*. V rámci prostředí implementace je každý spravovaný server nakonfigurovaný pro specifickou funkci (např. cílový hostitel, hostitel modulu aplikace, či server Common Event Infrastructure). Samostatný server je nakonfigurovaný tak, aby poskytoval všechny povinné funkce.

Servery poskytují běhové prostředí pro moduly SCA, pro prostředky, které tyto moduly používají (zdroje dat, specifikace aktivace a cíle služby JMS), a také pro prostředky dodané IBM (cíle zpráv, kontejnery komponenty Business Process Choreographer a servery Common Event Infrastructure).

Agent uzlu je administrativní agent reprezentující uzel pro váš systém a spravující servery tohoto uzlu. Agenti uzlu sledují servery hostitelského systému a trasují administrativní požadavky na servery. K vytvoření agenta uzlu dochází při sdružení uzlu v rámci správce implementace.

Správce implementace je administrativní agent poskytující centralizovaný pohled na správu více serverů a klastrů.

Samostatný server je definován samostatným profilem. Správce implementace je definován profilem správce implementace. Spravované servery se vytvářejí v rámci *spravovaného uzlu*, který je definován vlastním profilem.

Samostatné servery:

Samostatný server poskytuje prostředí pro implementaci modulů SCA v jediném procesu serveru. Tento proces serveru zahrnuje mimo jiné administrativní konzolu, cíl implementace, podporu systému zpráv, správce Business Process Rules Manager a server CEI (Common Event Infrastructure).

Samostatný server lze snadno nastavit a má konzolu První kroky, ze které můžete spustit s zastavit server a otevřít galerii ukázek a administrativní konzolu. Pokud nainstalujete ukázky produktu IBM Business Process Manager a poté otevřete galerii ukázek, provede se implementace ukázkového řešení na samostatný server. Prostředky použité v této ukázce můžete zkoumat v administrativní konzole.

Na samostatný server můžete implementovat i svá řešení, ale samostatný server nemůže zajistit kapacitu, rozšiřitelnost ani odolnost, které jsou vyžadovány po produkčním prostředí. Pro produkční prostředí je lepší použít prostředí síťové implementace.

Je možné začít samostatným serverem a později jej zahrnout do prostředí síťové implementace, a to federováním do buňky správce implementace, *pokud do této buňky nebyly federovány žádné jiné uzly*. Do jedné buňky nelze federovat více samostatných serverů. Chcete-li federovat samostatný server, použijte administrativní konzolu správce implementace nebo příkaz **addNode**. Při federování pomocí příkazu **addNode** nesmí být samostatný server spuštěný.

Samostatný server je definován profilem samostatného serveru.

Klastry:

Klastry jsou skupiny společně spravovaných serverů podílejících se na správě pracovní zátěže.

Klastr může obsahovat uzly nebo individuální aplikační servery. Uzel je zpravidla fyzický počítačový systém s jasně určenou adresou IP hostitele, na kterém je spuštěn minimálně jeden aplikační server. Klastry lze seskupit v rámci konfigurace buňky, čímž dojde k logickému vzájemnému přidružení mnoha serverů a klastrů, stejně jako aplikací s různými konfiguracemi. Výběr závisí čistě na příslušném administrátorovi a odvíjí se od toho, co je smysluplné v rámci jeho organizačního prostředí.

Klastry zodpovídají za vyrovnávání pracovní zátěže mezi servery. Servery, které jsou součástí klastru, se nazývají členy klastru. Když instalujete aplikaci na klastr, dojde k její automatické instalaci na každý člen klastru.

Vzhledem k tomu, že každý člen klastru obsahuje stejné aplikace, můžete úlohy klienta rozdělit na základě kapacity různých počítačů, a to tím, že každému serveru přiřadíte váhu.

Přiřazením váhy serverům klastru dosáhnete zlepšení výkonu a lepšího překonávání selhání. Úlohy se přiřazují serverům, které mají kapacitu na provedení operací úloh. Pokud je určitý server pro provedení dané úlohy nedostupný, přiřadí se tato úloha jinému členu klastru. Tato schopnost nového přiřazení má zjevné výhody oproti spuštění jediného aplikačního serveru, který se může v případě příliš velkého počtu požadavků přetížit.

Profily

Profil definuje jedinečné běhové prostředí se samostatnými příkazovými soubory, konfiguračními soubory a soubory protokolů. Profily definují v systémech produktu IBM Business Process Manager tři různé typy prostředí: samostatný server, správce implementace a spravovaný uzel.

S použitím profilů můžete mít v systému více než jedno běhové prostředí, aniž byste museli instalovat více kopií binárních souborů produktu IBM Business Process Manager.

Profily lze vytvářet pomocí nástroje Správa profilu nebo obslužného programu příkazového řádku **manageprofiles**.

Poznámka: Na distribuovaných platformách má každý uzel jedinečný název. Na platformě z/OS jsou všechny profily pojmenovány jako “default” nelze profily přejmenovat, upravit, kopírovat ani odstranit.

Adresář profilu

Každý profil v systému má vlastní adresář obsahující všechny jeho soubory. Umístění adresáře profilu určujete při vytváření profilu. Při výchozím nastavení se profil nachází v adresáři profiles v adresáři, kde je nainstalován produkt IBM Business Process Manager. Například: profil Dmgr01 je v adresáři C:\Program Files\IBM\WebSphere\ProcServer\profiles\Dmgr01.

Konzola První kroky

Každý profil v systému má konzolu První kroky. Pomocí tohoto rozhraní se můžete seznámit se samostatným serverem, správcem implementace nebo spravovaným uzlem.

Výchozí profil

Prvním profilem, který vytvoříte v rámci jedné instalace produktu IBM Business Process Manager, je *výchozí profil*. Výchozí profil představuje výchozí cíl pro příkazy vydávané z adresáře bin v adresáři, kde byl nainstalován produkt IBM Business Process Manager. Existuje-li v systému pouze jediný profil, bude každý z příkazů pracovat s tímto profilem. Pokud vytvoříte další profil, můžete jej nastavit jako výchozí.

Poznámka: Výchozím profilem nemusí být nutně profil s názvem “default”.

Rozšiřování profilů

Pokud již máte vytvořen profil správce implementace, vlastní profil nebo profil samostatného serveru pro produkt WebSphere Application Server Network Deployment či produkt WebSphere ESB můžete tento profil *rozšířit*, aby kromě existující funkce podporoval také produkt IBM Business Process Manager. Chcete-li rozšířit profil, nejdříve nainstalujte produkt IBM Business Process Manager. Poté použijte nástroj Správa profilu nebo obslužný program příkazového řádku **manageprofiles**.

Omezení: Nelze rozšířit profil, jenž definuje spravovaný uzel, který je již federován do správce implementace.

Správci implementace

Správce implementace je server, který spravuje operace pro logickou skupinu, nebo buňku ostatních serverů. Správce implementace je centrální umístění pro administraci serverů a klastrů.

Při vytváření prostředí implementace je prvním profilem, který vytvoříte, profil správce implementace. Správce implementace má konzolu První kroky, z níž můžete spustit a zastavit správce implementace a spustit jeho administrativní konzolu. Administrativní konzola správce implementace se používá ke správě serverů a klastrů v buňce. To zahrnuje konfiguraci serverů a klastrů, přidávání serverů do klastrů, spouštění a zastavování serverů a klastrů a implementaci modulů SCA.

Ačkoli je správce implementace typ serveru, nelze implementovat moduly přímo do správce implementace.

Uzly

Uzel je logické seskupení spravovaných serverů.

Uzel obvykle odpovídá logickému nebo fyzickému počítačovému systému s jasně určenou adresou IP hostitele. Uzel nemůže zahrnovat více počítačů. Názvy uzlů jsou obvykle identické s názvem hostitele pro daný počítač.

V rámci topologie síťové implementace mohou být uzly spravované nebo nespravované. Spravovaný uzel obsahuje proces agenta uzlu, který spravuje jeho konfiguraci a servery. Nespravované uzly agenta uzlu nemají.

Spravované uzly:

Spravovaný uzel je uzel, který je federován do správce implementace, obsahuje agenta uzlu a může obsahovat spravované servery. Ve spravovaném uzlu je možné konfigurovat a spouštět spravované servery.

Servery, které jsou nakonfigurované ve spravovaném uzlu tvoří prostředky vašeho prostředí implementace. Tyto servery jsou vytvářeny, konfigurovány, spouštěny, zastavovány, spravovány a odstraňovány pomocí administrativní konzoly správce implementace.

Spravovaný uzel obsahuje agenta uzlu, který spravuje všechny servery v uzlu.

Při federování uzlu se automaticky vytvoří proces agenta uzlu. Tento agent uzlu musí být spuštěný, abyste mohli spravovat konfiguraci profilu. Například při provádění následujících úloh:

- Spouštění a zastavování procesů.
- Synchronizace konfiguračních dat ve správci implementace s kopií v uzlu.

Agent uzlu však nemusí být spuštěný, chcete-li spouštět aplikace nebo konfigurovat prostředky v uzlu.

Spravovaný uzel může obsahovat jeden či více serverů, které spravuje správce implementace. Na serverech ve spravovaném uzlu můžete implementovat řešení, ale spravovaný uzel neobsahuje galerii ukázkových aplikací. Spravovaný uzel je definován vlastním profilem a má konzolu První kroky.

Nespravované uzly:

Nespravovaný uzel nemá agenta uzlu, který by spravoval jeho servery.

V rámci topologie síťové implementace mohou nespravované uzly obsahovat definice serverů, jako např. webových severů, ale nikoliv definice aplikačních serverů. Nespravované uzly nikdy nemohou být sdružené. To znamená, že agenta uzlu nelze nikdy přidat do nespravovaného uzlu. Dalším typem nespravovaného uzlu je samostatný server. Správce implementace tento samostatný server nemůže spravovat, protože jej buňka nezná. Samostatný server může být sdružený. Když sdružený je, dojde k automatickému vytvoření agenta uzlu. Z uzlu se tak stává spravovaný uzel dané buňky.

Agenti uzlů

Agenti uzlů jsou administrativní agenti, kteří trasují administrativní požadavky na servery.

Agent uzlu je server spuštěný v systému každého hostitelského počítače, který se podílí na konfiguraci síťové implementace. Jedná se čistě o administrativního agenta, který se nepodílí na funkcích sloužících účelům aplikací. Agent uzlu je také hostitelem dalších důležitých administrativních funkcí, jako např. služeb pro přenos souborů, synchronizace konfigurace, či monitorování výkonu.

Aspekty pojmenování profilů, uzlů, serverů, hostitelů a buněk

Toto téma pojednává o podmínkách a otázkách, které je třeba zohlednit při pojmenování vašeho profilu, uzlu, serveru, hostitele či buňky (je-li to použitelné). Toto téma platí pro distribuované platformy.

Aspekty pojmenování profilu

Profil může mít libovolný jedinečný název s následujícími omezeními. Při pojmenování profilu nepoužívejte následující znaky:

- mezery,
- speciální znaky, které nejsou povoleny v názvu adresáře ve vašem operačním systému, jako např. *, &, nebo ?
- lomítka (/) a zpětná lomítka (\)

Dvoubajtové znaky jsou povolené.

Windows **Aspekty cesty k adresáři:** Cesta k instalačnímu adresáři musí mít nejvýše 60 znaků. Název adresáře *cesta_k_adresáři_profilů\název_profilu* musí obsahovat nejvýše 80 znaků.

Aspekty pojmenování uzlu, serveru, hostitele a buňky

Vyhrazené názvy: Vyhněte se použití vyhrazených názvů jako hodnot polí. Použití vyhrazených názvů může způsobit nepředvídatelné výsledky. Vyhrazena jsou následující slova:

- Buňky.
- Uzly.
- Servery.

- Klastry.
- Aplikace.
- Implementace.

Popisy polí na stránkách Název uzlu a hostitele a Název uzlu, hostitele a buňky: Tabulka 11 popisuje pole na stránkách Název uzlu a hostitele a Název uzlu, hostitele a buňky v nástroji Správa profilu, včetně názvů polí, výchozích hodnot a omezení. Tyto informace použijte jako vodítko při vytváření profilů.

Tabulka 11. Pokyny pro pojmenování uzlů, serverů, hostitelů a buněk

Název pole	Výchozí hodnota	Omezení	Popis
Profily samostatného serveru			
Název uzlu	Linux UNIX Windows *krátkýNázevHostitele* Node *ČísloUzlu* kde: • *krátkýNázevHostitele* je krátký název hostitele. • *ČísloUzlu* je pořadové číslo počínaje hodnotou 01.	Vyhněte se použití vyhrazených názvů.	Vyberte libovolný požadovaný název. Plánujete-li v systému vytvořit více než jeden server, v zájmu lepšího uspořádání instalace použijte jedinečný název.
Název serveru	Linux UNIX Windows server1	Pro server použijte jedinečný název.	Logický název serveru.
Název hostitele	Linux UNIX Windows Dlouhý tvar názvu serveru názvů domény (DNS).	Název hostitele musí být adresovatelný prostřednictvím vaší sítě. Plánujete-li používat prostor Business Space, použijte úplný název hostitele.	Použijte skutečný název DNS nebo adresu IP vaší pracovní stanice, abyste umožnili komunikaci s ní. Viz další informace o názvu hostitele za touto tabulkou.

Tabulka 11. Pokyny pro pojmenování uzlů, serverů, hostitelů a buněk (pokračování)

Název pole	Výchozí hodnota	Omezení	Popis
Název buňky	Linux UNIX Windows *krátkýNázevHostitele* Node *ČísloUzlu* Cell kde: • *krátkýNázevHostitele* je krátký název hostitele. • *ČísloUzlu* je pořadové číslo počínaje hodnotou 01.	Pro buňku použijte jedinečný název. Název buňky musí být jedinečný vždy, pokud je produkt spuštěn na stejné fyzické pracovní stanici či klastru pracovních stanic, jako např. v prostředí Sysplex. Název buňky navíc musí být jedinečný vždy, když je požadována připojitelnost v síti mezi entitami, buď mezi buňkami, nebo od klienta, který musí komunikovat s každou z buněk. Názvy buněk musí být jedinečné také tehdy, pokud mají být federovány jejich obory názvů. Jinak by mohly nastat problémy, například výjimka javax.naming.Name NotFoundException; v takovém případě je třeba vytvořit buňky s jedinečnými názvy.	Všechny federované uzly se stanou členy buňky správce implementace.
Profily správce implementace			
Název uzlu	Linux UNIX Windows *krátkýNázevHostitele* Cell Manager*ČísloUzlu* kde: • *krátkýNázevHostitele* je krátký název hostitele. • *ČísloUzlu* je pořadové číslo počínaje hodnotou 01.	Pro správce implementace použijte jedinečný název. Vyhněte se použití vyhrazených názvů.	Název se používá pro administraci v rámci buňky správce implementace.
Název hostitele	Linux UNIX Windows Dlouhý tvar názvu serveru názvů domény (DNS).	Název hostitele musí být adresovatelný prostřednictvím vaší sítě. Vyhněte se použití vyhrazených názvů. Plánujete-li používat prostor Business Space, použijte úplný název hostitele.	Použijte skutečný název DNS nebo adresu IP vaší pracovní stanice, abyste umožnili komunikaci s ní. Viz další informace o názvu hostitele za touto tabulkou.

Tabulka 11. Pokyny pro pojmenování uzlů, serverů, hostitelů a buněk (pokračování)

Název pole	Výchozí hodnota	Omezení	Popis
Název buňky	Linux UNIX Windows *krátkýNázevHostitele* Cell *ČísloBuňky* kde: • *krátkýNázevHostitele* je krátký název hostitele. • *ČísloBuňky* je pořadové číslo počínaje hodnotou 01.	Pro buňku správce implementace použijte jedinečný název. Název buňky musí být jedinečný vždy, pokud je produkt spuštěn na stejné fyzické pracovní stanici či klastru pracovních stanic, jako např. v prostředí Sysplex. Název buňky navíc musí být jedinečný vždy, když je požadována připojitelnost v síti mezi entitami, buď mezi buňkami, nebo od klienta, který musí komunikovat s každou z buněk. Názvy buněk musí být jedinečné také tehdy, pokud mají být federovány jejich obory názvů. Jinak by mohly nastat problémy, například výjimka javax.naming.Name NotFoundException; v takovém případě je třeba vytvořit buňky s jedinečnými názvy.	Všechny federované uzly se stanou členy buňky správce implementace, kterou pojmenujete na stránce Název uzlu, hostitele a buňky v nástroji Správa profilu.
Vlastní profily			
Název uzlu	Linux UNIX Windows *krátkýNázevHostitele* Node *ČísloUzlu* kde: • *krátkýNázevHostitele* je krátký název hostitele. • *ČísloUzlu* je pořadové číslo počínaje hodnotou 01.	Vyhněte se použití vyhrazených názvů. V rámci buňky správce implementace použijte jedinečný název.	Název se používá pro administraci v rámci buňky správce implementace, do které je vlastní profil přidáván. V rámci buňky správce implementace použijte jedinečný název.
Název hostitele	Linux UNIX Windows Dlouhý tvar názvu serveru názvů domény (DNS).	Název hostitele musí být adresovatelný prostřednictvím vaší sítě. Plánujete-li používat prostor Business Space, použijte úplný název hostitele.	Použijte skutečný název DNS nebo adresu IP vaší pracovní stanice, abyste umožnili komunikaci s ní. Viz další informace o názvu hostitele za touto tabulkou.

Aspekty názvu hostitele:

Název hostitele je síťový název fyzické pracovní stanice, na které je nainstalován uzel. Název hostitele musí být možné přeložit na síťový uzel na serveru. Pokud na serveru existuje více síťových karet, název hostitele nebo adresu IP musí být možné přeložit na jednu ze síťových karet. Pomocí názvu hostitele se k tomuto uzlu připojují vzdálené uzly, které s ním poté komunikují.

Produkt IBM Business Process Manager je kompatibilní s verzí 4 protokolu IP (IPv4) i verzí 6 tohoto protokolu (IPv6). Kdykoli v administrativní konzole nebo jinde zadáváte adresy IP, můžete tak učinit v obou těchto formátech. Pokud je však na vašem systému implementován protokol IPv6, musíte zadat adresu IP ve formátu IPv6, a naopak, není-li vám protokol IPv6 zatím k dispozici, zadávejte adresy IP ve formátu IPv4. Další informace k IPv6 viz následující popis: IPv6.

Při určování vhodného názvu hostitele pro pracovní stanici vám mohou pomoci následující pokyny:

- Vyberte název hostitele, na který dosáhnou ostatní pracovní stanice v rámci vaší sítě.
- Pro tuto hodnotu nepoužívejte generický identifikátor lokální_hostitel (localhost).
- Nesnažte se nainstalovat produkty IBM Business Process Manager na server s názvem hostitele, který používá znaky z dvoubajtové znakové sady (DBCS). Při použití v názvu hostitele nejsou znaky DBCS podporovány.
- V názvech serverů nepoužívejte znak podtržítko (_). Standardy sítě Internet vyžadují, aby názvy domén splňovaly požadavky popsané v oficiálních standardech protokolů sítě Internet RFC 952 a RFC 1123. Názvy domén smí obsahovat pouze písmena (malá nebo velká) a číslice. Názvy domén dále mohou obsahovat pomlčky (-), avšak pomlčka nesmí být na koncích názvu. Podtržítka (_) nejsou v názvu hostitele podporována. Pokud jste produkt IBM Business Process Manager nainstalovali na server, který v názvu obsahuje podtržítko, do přejmenování můžete k tomuto serveru přistupovat pomocí jeho adresy IP.

Pokud na stejném počítači definujete koexistující uzly s jedinečnými adresami IP, definujte každou adresu IP ve vyhledávací tabulce serveru názvů domény (DNS). Konfigurační soubory serverů neposkytují rozlišování názvů domén v případě více adres IP na pracovní stanici s jedinou síťovou adresou.

Hodnota, kterou určíte pro název hostitele, se použije v konfiguračních dokumentech jako hodnota vlastnosti hostName. Hodnotu názvu hostitele můžete zadat v následujících formátech:

- Úplný řetězec názvu hostitele DNS, například xmachine.manhattan.ibm.com.
- Řetězec výchozího krátkého názvu hostitele DNS, například xmachine.
- Číselná adresa IP, například 127.1.255.3.

Úplný řetězec názvu hostitele DNS má výhodu naprosté jednoznačnosti a flexibility. Máte flexibilitu změny skutečné adresy IP hostitelského systému, aniž byste museli změnit konfiguraci serveru. Taková hodnota názvu hostitele je zvlášť užitečná, plánujete-li často měnit adresu IP při použití protokolu DHCP (Dynamic Host Configuration Protocol) k přiřazování adres IP. Nevýhodou tohoto formátu je závislost na serveru DNS. Není-li server DNS dostupný, je konektivita narušena.

Také krátký název hostitele umožňuje dynamické rozlišování. Formát krátkého názvu navíc může být předefinován v souboru lokálních hostitelů, takže systém může spustit server, i když je odpojen od sítě. Chcete-li spustit server při odpojení, v souboru lokálních hostitelů definujte krátký název jako 127.0.0.1 (lokální zpětná smyčka). Nevýhodou formátu krátkého názvu je závislost na serveru DNS v případě vzdáleného přístupu. Není-li server DNS dostupný, je konektivita narušena.

Číselná adresa IP má tu výhodu, že nevyžaduje rozlišení názvu prostřednictvím serveru DNS. K uzlu pojmenovanému číselnou adresou IP se vzdálený uzel může připojit i při nedostupnosti serveru DNS. Nevýhodou tohoto formátu je, že číselná adresa IP je pevná. Kdykoli změníte adresu IP pracovní stanice, musíte změnit i nastavení vlastnosti hostName v konfiguračních dokumentech. Z tohoto důvodu nepoužívejte číselnou adresu IP, používáte-li protokol DHCP nebo měníte-li pravidelně adresy IP. Další nevýhodou tohoto formátu je, že uzel nelze použít, pokud je hostitel odpojen od sítě.

BPMN 2.0

Definice obchodních procesů v produktu IBM Business Process Manager podporují podtřídu Common Executable třídy shody BPMN 2.0 Process Modeling, která se zabývá modely, jež lze spustit.

BPMN (Business Process Model and Notation) je základním standardem pro procesy v produktech IBM Process Designer a IBM Process Center. Na specifikaci BPMN jsou založeny diagramy definic obchodních procesů (BPD).

Toto téma představuje některé způsoby, jak je standard BPMN 2.0 v produktu IBM Business Process Manager použitý. Podrobné informace o standardu BPMN naleznete na stránce specifikace BPMN na adrese http://www.bpmn.org/.

Produkt IBM Business Process Manager podporuje následující typy úloh dle BPMN 2.0:

- Není (abstraktní úloha ve specifikaci BPMN 2.0).
- Systémová úloha (úloha služby ve specifikaci BPMN 2.0).
- Uživatelská úloha.
- Skript.
- Rozhodovací úloha (úloha obchodního pravidla ve specifikaci BPMN 2.0).

Intermediační událost zpráv IBM BPM poskytuje podobné funkce jako úloha odeslání a úloha příjmu ve specifikaci BPMN.

Notace BPMN 2.0

Počínaje verzí 7.5.1 jsou v komponentě Process Designer ikony úloh BPMN 2.0 v diagramech BPD shromážděny na zjednodušené paletě a zobrazeny v diagramech procesů. Ikony ukazují, zda je daná aktivita systémovou úlohou, uživatelskou úlohou, rozhodovací úlohou, skriptem nebo propojeným procesem. Aktivity v modelech, které byly vytvořeny ve starších verzích, jsou při prohlížení ve verzi 7.5.1 a novější také zobrazeny s příslušnými typy úloh a ikonami úloh podle specifikace BPMN 2.0.

Aktivity a úlohy

Došlo k určitým terminologickým změnám oproti předchozím verzím produktu Process Designer. Řada těchto změn se týká typů aktivit, které byly přejmenovány.

- Aktivity Služba (automatická) jsou nyní systémové úlohy.
- Aktivity Služba (úloha) v nesystémové dráze jsou nyní uživatelské úlohy.
- Aktivity Služba (úloha) v systémové dráze jsou nyní rozhodovací úlohy, pokud odkazují na službu rozhodování.
- Aktivity Služba (úloha) v systémové dráze jsou nyní systémové úlohy, pokud odkazují na jakoukoli jinou službu než službu rozhodování.
- Aktivity Javascript jsou nyní úlohy skriptu.
- Aktivity Vnořený proces jsou nyní propojené procesy.
- Externí aktivity z předchozích verzí produktu Process Designer jsou dostupné jako externí implementace pro uživatelské úlohy nebo systémové úlohy.

Brány

Nedošlo k žádným změnám notace pro brány předchozích verzí. Došlo však ke třem změnám terminologie. Rozhodovací brána je nyní *výlučná brána*, brána jednoduchého rozdělení nebo spojení je nyní *paralelní brána* a brána podmíněného rozdělení nebo spojení je nyní *nevýlučná brána*.

Existuje také nový typ brány, *brána událostí*. Brána událostí představuje bod větvení v procesu, kde jsou alternativní cesty následující za danou branou založeny na událostech, k nimž dojde, a ne na vyhodnocení výrazů s použitím dat procesu (jako u výlučných a nevýlučných bran). Cestu, kterou se bude proces ubírat, určuje specifická událost, obvykle přijetí zprávy.

Nepřerušující události

Specifikace BPMN 2.0 přidala notaci pro nepřerušující události. Událost hranice standardně přeruší aktivitu, k níž je připojena. Při spuštění této události se aktivita zastaví a token postupuje dál odchozím tokem posloupnosti pro danou událost. Pokud je událost nastavena jako nepřerušující, pokračuje připojená aktivita při spuštění události paralelně a vygeneruje se nový token, který se předá dál odchozím tokem posloupnosti pro danou událost. Hranice události se u nepřerušujících událostí změní na čárkovanou čáru.

Intermediační události připojené k aktivitám jsou přerušující intermediační události, pokud své připojené aktivity zavírají, nebo nepřerušující intermediační události, pokud své připojené aktivity nezavírají.

Událost zahájení

Specifikace BPMN umožňuje v modelech procesů vynechat symboly událostí zahájení a událostí ukončení. Komponenta Process Designer vyžaduje, aby modely procesů používaly události zahájení a ukončení.

V komponentě Process Designer jsou k dispozici různé typy událostí zahájení:

procesy

- není
- zprávy
- ad hoc

podprocesy

- není

podprocesy událostí

- chyba
- zprávy
- časovač

Typ události zahájení můžete změnit tak, že upravíte vlastnosti události. V procesu může být celá řada událostí zahájení typu zpráva, ale jen jedna událost zahájení typu není.

Události ukončení

K dispozici jsou čtyři typy událostí ukončení: *zpráva*, *ukončit*, *chyba* a *není*. Typ události ukončení můžete změnit.

Když nadřízený proces volá proces podřízený a podřízený proces provede akci události typu ukončit, sémantika BPMN říká, že se podřízený proces okamžitě zastaví a nadřízený proces pak pokračuje dalšími kroky. Pokud v komponentě Process Designer provede podřízený proces aktivitu události ukončení, zastaví se podřízený i nadřízený proces.

Podprocesy

Specifikace BPMN definuje dva typy podprocesů, vestavěný a znovupoužitelný. V komponentě Process Designer můžete vytvořit oba typy. Vestavěné podprocesy se v produktu Process Designer nazývají prostě *podprocesy* a jsou ve verzi 7.5.1 nové. Znovupoužitelný podproces BPMN se v komponentě Process Designer nazývá *propojený proces*.

Podproces existuje v rámci procesu, který jej obsahuje, a přestavuje způsob, jak seskupit kroky procesu za účelem zjednodušení a zpřehlednění diagramu. Podprocesy sbalí několik kroků do jediné aktivity. Podproces je viditelný pouze pro proces, v němž je definován. Podproces existuje v rozsahu platnosti volajícího procesu a má přístup ke všem proměnným v tomto prostředí. Nedochází k žádnému předávání parametrů do vestavěného procesu a z něj.

Vedle podprocesu a propojeného procesu obsahuje komponenta Process Designer podproces události, což je specializovaný podproces, který se používá pro obsluhu událostí. Není propojen s jinými aktivitami prostřednictvím toku posloupnosti a dochází k němu jen tehdy, je-li spuštěna jeho událost zahájení.

Propojené procesy

Znovupoužitelný podproces dle BPMN se v produktu Process Designer nazývá *propojený proces*. Jde o proces vytvořený mimo aktuální proces, který může být aktuálním procesem volán. Je znovupoužitelný, protože definice jiných procesů mohou tento proces také volat. Propojený proces definuje své vstupní a výstupní parametry a nemá přístup k rozsahu platnosti nebo prostředí volajícího procesu. Propojený proces je podobný vnořenému procesu, který byl k dispozici v předchozích verzích; nedošlo k žádným změnám chování aktivity. Předchozí vnořené procesy se

migrací převedou na propojené procesy. Propojený proces vypadá jako podproces s tlustou hranicí a v okně komponenty Inspector je zvýrazněný.

Cykly

Specifikace BPMN nabízí pojem aktivity, která se může opakovat. Aktivita může být atomická, což znamená, že se opakuje tato aktivita, nebo může jít o podproces, který zahrnuje řadu kroků, jež se opakují. Pokud rozbalíte opakovanou aktivitu, uvidíte obsažené aktivity, které se mají opakovaně spouštět. Podmínka se vyhodnotí vždy na začátku každé iterace cyklu. Nelze provést vyhodnocení na konci jednotlivých iterací cyklu.

Produkt IBM Business Process Manager obsahuje *cyklus s více instancemi*, který se provede konečně krát, přičemž aktivity v něm obsažené se spouští sekvenčně nebo paralelně.

Import jiných procesů než BPMN

Můžete importovat modely, které byly vytvořeny v produktu IBM WebSphere Business Modeler, a používat je v produktu Process Designer. Podrobnosti o importu produktů BPMN 2.0 viz Mapování prvků produktu IBM WebSphere Business Modeler na konstrukce produktu IBM Business Process Manager. Také můžete importovat modely BPMN 2.0, které byly vytvořeny v produktech IBM WebSphere Business Compass, Rational Software Architect nebo v jiných prostředích pro modelování.

Definice obchodního procesu (BPD)

Chcete-li modelovat určitý proces v produktu IBM Process Designer, je třeba vytvořit definici obchodního procesu (BPD).

Definice BPD představuje opakovaně použitelný model procesu definující aspekty, které jsou společné všem běhovým instancím daného modelu procesu. BPD musí obsahovat událost zahájení, událost ukončení, nejméně jednu dráhu a jednu či více aktivit. Podrobnosti o omezení znaků platných pro BPD viz "Konvence pojmenování produktu IBM Process Designer" v souvisejících odkazech.

Do definice BPD (Business Process Definition) je nutné přidat dráhu pro každý systém nebo skupinu uživatelů, kteří se účastní určitého procesu. Plavecká dráha může být účastnická dráha nebo systémová dráha. Pokud však budete chtít, můžete vytvořit BPD, která seskupuje aktivity skupiny a systému do jediné dráhy. Informace o vytvoření BPD viz "Vytvoření definice obchodního procesu (BPD)" v souvisejících odkazech.

Odpovědnost za aktivity v účastnické dráze můžete pověřit libovolnou specifickou osobu nebo skupinu. Každá dráha, kterou vytvoříte, je standardně přiřazena do skupiny účastníků Všichni uživatelé. Pomocí této výchozí skupiny účastníků můžete spustit a otestovat danou definici BPD v komponentě Inspector. Skupina účastníků Všichni uživatelé zahrnuje všechny uživatele, kteří jsou členy skupiny zabezpečení tw_allusers, což je speciální skupina zabezpečení, která automaticky zahrnuje všechny uživatele v systému.

Systémová dráha obsahuje aktivity ošetřené specifickým systémem produktu IBM Process Center. Každá aktivita potřebuje implementaci, která tuto aktivitu definuje a nastavuje vlastnosti úlohy. Během implementace vytvoří vývojář službu nebo napíše skript JavaScript nezbytný k realizaci aktivit v systémové dráze. Informace o službách viz "Základní informace o typech služeb" v souvisejících odkazech.

Pro každou vytvářenou definici BPD je nutné deklarovat proměnné, které umožní zachycení obchodních dat předávaných mezi jednotlivými aktivitami v rámci procesu. Informace o implementaci proměnných viz "Správa a mapování proměnných" v souvisejících odkazech.

Do definice BPD lze také přidávat události. Události v IBM BPM se spouštějí, když nastane určité datum, vyskytne se výjimka nebo je doručena zpráva. Typ události, kterou chcete implementovat, určuje požadovaný typ spouštěče. Podrobné informace o dostupných typech událostí a o jejich spouštěčích viz "Modelování událostí".

Při sestavování definic obchodních procesů v komponentě Process Designer je nutné provést určité konfigurační úlohy, aby běhové instance procesu splňovaly požadavky všech pracovníků dané organizace. Seznam a popis voleb viz "Volby konfigurace".

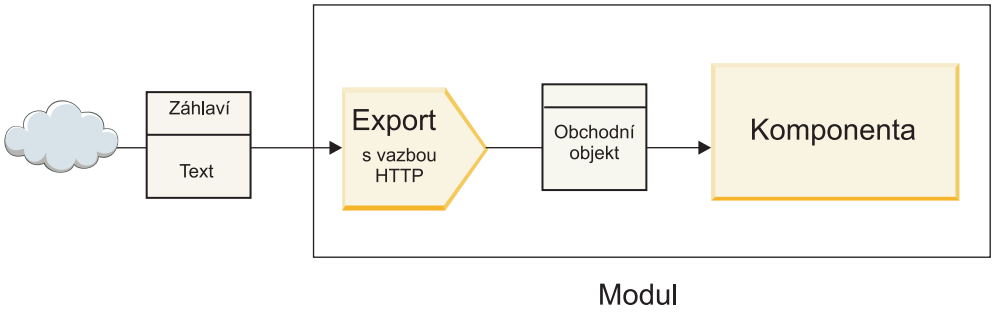
Vazby

Jádrem architektury SOA (Service-Oriented Architecture) je koncepce *služby*, jednotky funkčnosti zajišťované interakcí výpočetních zařízení. *Export* definuje externí rozhraní (nebo přístupový bod) modulu, aby mohly komponenty SCA (Service Component Architecture) v rámci modulu poskytovat služby externím klientům. *Import* definuje rozhraní pro služby mimo modul, aby mohly být tyto služby volány z modulu. *Vazby* specifické pro určitý protokol se používají s importy a exporty k určení prostředků přenosu dat do modulu nebo z modulu.

Exporty

Externí klienti mohou vyvolávat komponenty SCA v integračním modulu pomocí celé řady protokolů (např. HTTP, JMS, MQ a RMI/IIOP) a s daty v nejrůznějších formátech (např. XML, CSV, COBOL a JavaBeans). Exporty jsou komponenty, které přijímají tyto požadavky od externích zdrojů a následně vyvolávají komponenty produktu IBM Business Process Manager pomocí programovacího modelu SCA.

Například na následujícím obrázku obdrží export požadavek z klientské aplikace prostřednictvím protokolu HTTP. Data jsou transformována do obchodního objektu, ve formátu používaném komponentou SCA. Tato komponenta je potom vyvolána s pomocí tohoto datového objektu.

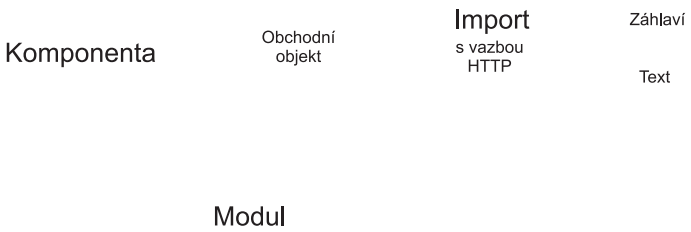


Obrázek 1. Export s vazbou HTTP

Importy

Komponenta SCA může chtít vyvolat externí službu jiného typu než SCA, která očekává data v jiném formátu. Import používá komponenta SCA k vyvolání externí služby s pomocí programovacího modelu SCA. Poté import vyvolá cílovou službu způsobem, který tato služba očekává.

Například na následujícím obrázku je odeslán požadavek z komponenty SCA importem externí službě. Obchodní objekt, což je formát používaný komponentou SCA, je transformován na formát očekávaný danou službou a ta je vyvolána.



Obrázek 2. Import s vazbou HTTP

Seznam vazeb

Pomocí produktu Integration Designer můžete generovat vazbu pro import nebo export a zkonfigurovat vazbu. Dostupné typy vazeb jsou popsány v následujícím seznamu.

- SCA

  Vazba SCA, která je výchozí, umožňuje vaší službě komunikovat se službami v ostatních modulech SCA. Import s vazbou SCA použijete pro přístup ke službě v jiném modulu SCA. Export s vazbou SCA použijete k nabídce služby jiným modulům SCA.

- Webová služba

  Vazba webové služby umožňuje přístup k externí službě pomocí interoperabilních zpráv SOAP a kvalit služby. Vazby webové služby můžete použít také pro zahrnutí příloh do zprávy SOAP.

  Vazba webové služby může používat transportní protokol SOAP/HTTP (SOAP přes HTTP) nebo SOAP/JMS (SOAP přes JMS). Nehledě na přenos (HTTP nebo JMS) použitý k předání zpráv SOAP, ošetřují vazby webové služby interakce požadavků/odezev vždy synchronně.

- HTTP

  Vazba HTTP umožňuje přístup k externí službě pomocí protokolu HTTP, kde jsou používány jiné zprávy než SOAP nebo je nutný přímý přístup HTTP. Tato vazba se používá, když pracujete s webovými službami, které jsou založené na modelu HTTP (tj. službami, které používají známé operace rozhraní HTTP, jako GET, PUT, DELETE apod.).

- Objekty Enterprise JavaBean (EJB)

  Vázání EJB umožňují komponentám SCA interakci se službami, které poskytuje obchodní logika v jazyce Java EE spuštěná na serveru Java EE.

- EIS

  Vazba EIS (podnikového informačního systému), je-li použita s adaptérem prostředku JCA, umožňuje přístup ke službám v podnikovém informačním systému nebo tomuto systému povolí vaše služby.

- Vazby JMS

  Vazby Java Message Service (JMS), generické služby JMS a WebSphere MQ JMS (MQ JMS) jsou používány pro interakci se systémy zpráv, kde je použití asynchronní komunikace prostřednictvím front zpráv důležité pro spolehlivost.

  Export s jednou z vazeb JMS sleduje frontu, čeká na přijetí zprávy a asynchronně odesílá případnou odezvu do fronty odpovědí. Import s jednou z vazeb JMS sestaví a odešle zprávu do fronty JMS, sleduje frontu a čeká na přijetí případné odpovědi.

  – JMS

    Vazba JMS umožňuje přístup k poskytovateli JMS vestavěnému do platformy WebSphere.

  – Generická platforma JMS

    Generická vazba služby JMS umožňuje přístup k systému zasílání zpráv od jiného dodavatele než IBM.

  – MQ JMS

    Vazba MQ JMS umožňuje přístup k podmnožině JMS systému zasílání zpráv WebSphere MQ. Tuto vazbu byste použili, když vaší aplikaci stačí podmnožina funkcí JMS.

- MQ

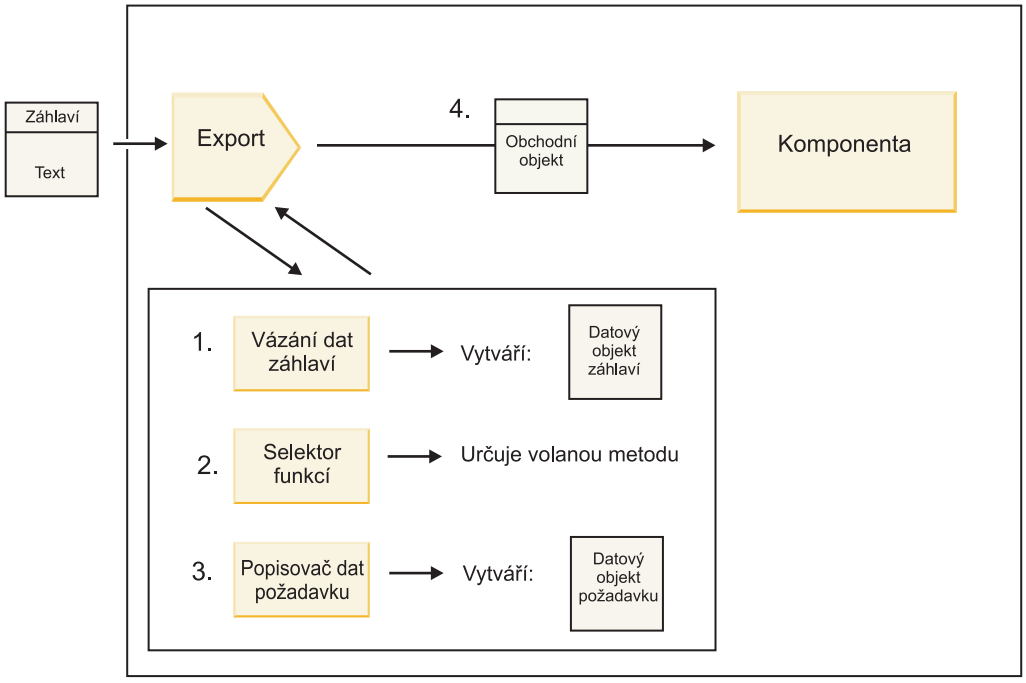
  Vazba WebSphere MQ umožňuje komunikaci s nativními aplikacemi MQ, začleňuje je do rámce architektury SOA (Service-Oriented Architecture) a poskytuje přístup k informacím záhlaví specifickým pro MQ. Tuto vazbu byste použili, když potřebujete nativní funkce MQ.

Přehled vazeb exportu a importu

Export vám umožní zpřístupnit služby v integračním modulu externím klientům a import umožní vašim komponentám SCA v integračním modulu volat externí služby. Vazba přidružená k exportu nebo importu určuje vztah mezi zprávami protokolu a obchodními objekty. Také určuje způsob výběru operací a poruch.

Tok informací exportem

Export obdrží požadavek, který je určen komponentě, s níž je tento export spojen, prostřednictvím specifického přenosu určeného přidruženou vazbou (například HTTP).



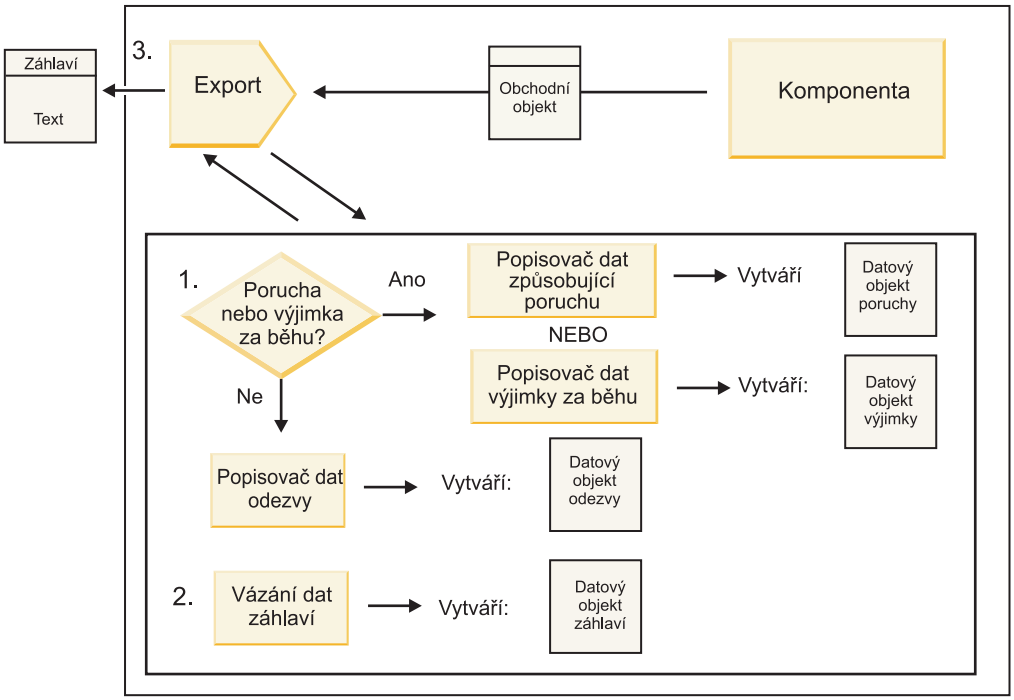
Obrázek 3. Tok požadavku exportem do komponenty

Když export obdrží požadavek, dojde k následující posloupnosti událostí:

1. Pouze v případě vazeb WebSphere MQ provede vázání dat záhlaví transformaci záhlaví protokolu na datový objekt záhlaví.
2. Selektor funkcí určuje název nativní metody ze zprávy protokolu. Název nativní metody je konfigurací exportu mapován na název operace na rozhraní exportu.
3. Popisovač dat požadavku nebo vázání dat na metodě transformuje požadavek na obchodní objekt požadavku.
4. Export vyvolá metodu komponenty s obchodním objektem požadavku.
   - Vazba exportu HTTP, vazba exportu webové služby a vazba exportu EJB vyvolávají komponentu SCA synchronně.
   - Vazby exportu JMS, generické služby JMS, MQ JMS a WebSphere MQ vyvolávají komponentu SCA asynchronně.

Všimněte si, že export může šířit záhlaví a uživatelské vlastnosti, které obdrží prostřednictvím protokolu, pokud je povoleno šíření kontextu. Komponenty, které jsou spojené s exportem, k mohou přistupovat k těmto záhlavím a uživatelským vlastnostem. Další informace viz téma “Šíření” v Informačním centru pro WebSphere Integration Developer.

Pokud se jedná o obousměrnou operaci, vrátí komponenta odezvu.



Obrázek 4. Tok odpovědi zpět exportem

Dochází k této posloupnosti kroků:

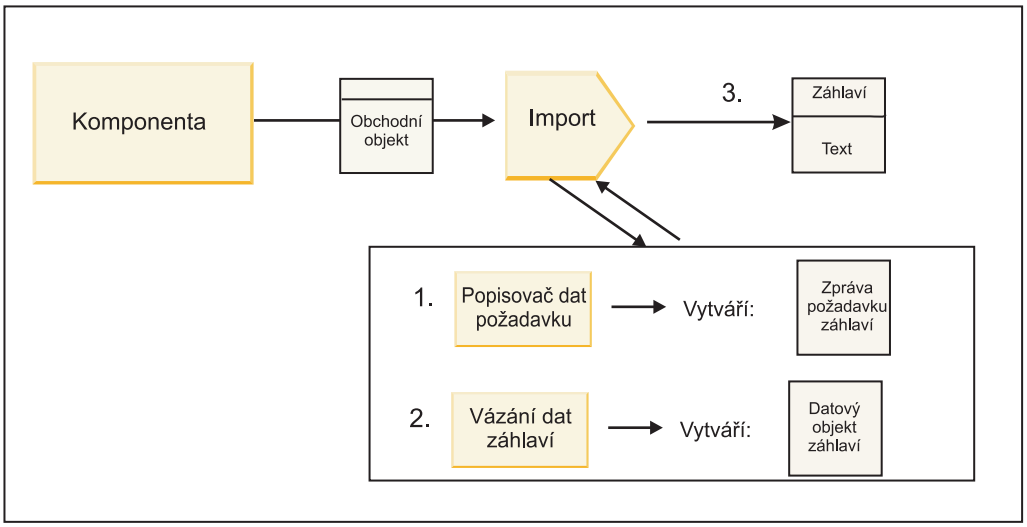
1. Pokud vazba exportu obdrží normální zprávu odpovědi, popisovač dat odpovědi nebo vázání dat na dané metodě transformují tento obchodní objekt na odezvu.

   Pokud je odezvou porucha, transformuje ji popisovač dat způsobující poruchu nebo vázání dat na metodě na odezvu poruchy.

   Pouze pro vazby exportu HTTP, pokud je odezvou výjimka za běhu, je volán popisovač dat výjimky za běhu, je-li nakonfigurován.
2. Pouze pro vazby WebSphere MQ provede vázání dat záhlaví transformaci datových objektů záhlaví na záhlaví protokolu.
3. Export odešle odezvu služby pomocí přenosu.

Tok informací importem

Komponenty odesílají požadavky službám mimo modul pomocí importu. Požadavek je odeslán prostřednictvím specifického přenosu určeného přidruženou vazbou.



Obrázek 5. Tok z komponenty importem do služby

Komponenta vyvolává import pomocí obchodního objektu požadavku.

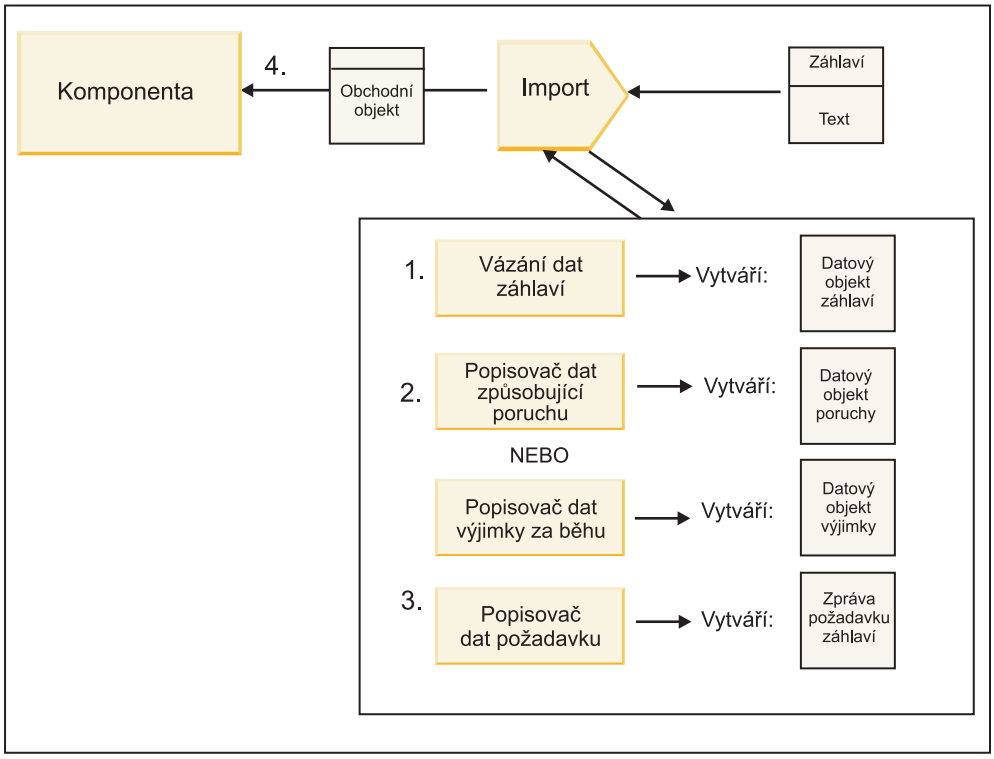
Poznámka:

- Vazbu importu HTTP, vazbu importu webové služby a vazbu importu EJB by měla volající komponenta vyvolávat synchronně.
- Vazby importu JMS, generické služby JMS, MQ JMS a WebSphere MQ mají být vyvolávány asynchronně.

Když komponenta vyvolá import, dojde k následující posloupnosti událostí:

1. Popisovač dat požadavku nebo vázání dat na metodě transformují obchodní objekt požadavku na zprávu požadavku protokolu.
2. Pouze pro vazby WebSphere MQ provede vázání dat záhlaví na metodě nastavení obchodního objektu záhlaví v záhlaví protokolu.
3. Import vyvolává službu pomocí požadavku na službu prostřednictvím přenosu.

Pokud se jedná o obousměrnou operaci, vrátí služba odezvu a dochází k této posloupnosti kroků:



Obrázek 6. Tok odpovědi zpět importem

1. Pouze v případě vazeb WebSphere MQ provede vázání dat záhlaví transformaci záhlaví protokolu na datový objekt záhlaví.
2. Určí se, zda je daná odezva porucha.
   - Pokud je odezvou porucha, zkontroluje ji selektor poruch, aby zjistil, na kterou poruchu WSDL se mapuje. Popisovač dat způsobující poruchu na metodě potom transformuje poruchu na odezvu poruchy.
   - Pokud je odezvou výjimka za běhu, je volán popisovač dat výjimky za běhu, je-li nakonfigurován.
3. Popisovač dat odpovědi nebo vazba na metodě transformuje odezvu na obchodní objekt odpovědi.
4. Import vrací obchodní objekt odpovědi komponentě.

Konfigurace vazby exportu a importu

Jedním z klíčových aspektů vazeb exportu a importu je transformace formátu dat, která označuje, jak jsou data mapována (deserializována) z nativního formátu spojů na obchodní objekt nebo jak jsou mapována (serializována) z obchodního objektu do nativního formátu spojů. V případě vazeb přidružených k exportům můžete určit také selektor funkcí určující, která funkce má být nad daty provedena. V případě vazeb přidružených k exportům nebo importům můžete stanovit, jak pracovat s poruchami, k nimž dojde při zpracování.

Navíc určíte informace týkající se přenosu pro vazby. Například u vazby HTTP určíte adresu URL koncového bodu. Pro vazbu HTTP jsou informace týkající se přenosu popsány v tématech “Generování vazby HTTP pro import” a “Generování vazby HTTP pro export”. V Informačním centru najdete také informace o ostatních vazbách.

Transformace formátu dat v importech a exportech:

Když je v produktu IBM Integration Designer nakonfigurována vazba exportu nebo importu, jednou z vlastností konfigurace, kterou určujete, je datový formát používaný vazbou.

- U vazeb exportu, v jejichž rámci aplikace klienta odesílá požadavky komponentě SCA a přijímá od ní odpovědi, určíte formát nativních dat. V závislosti na tomto formátu systém vybere odpovídající popisovač dat nebo vázání dat pro transformaci nativních dat na obchodní objekt (který používá komponenta SCA) a naopak pro transformaci obchodního objektu na nativní data (což je odezva aplikaci klienta).

- U vazeb importu, v jejichž rámci komponenta SCA odesílá požadavky službě mimo modul a přijímá od ní odpovědi, určíte datový formát nativních dat. V závislosti na tomto formátu systém vybere odpovídající popisovač dat nebo vázání dat pro transformaci obchodního objektu na nativní data a naopak.

Produkt IBM Business Process Manager nabízí sadu předdefinovaných datových formátů a odpovídajících popisovačů dat nebo vázáních dat, která tyto formáty podporují. Můžete si vytvořit i vlastní popisovače dat a zaregistrovat datový formát pro tyto popisovače. Další informace viz téma "Vývoj popisovačů dat" v Informačním centru produktu IBM Integration Designer.

- *Popisovače dat* jsou nezávislé na protokolu a transformují data z jednoho formátu do jiného. V produktu IBM Business Process Manager popisovače dat zpravidla transformují nativní data (např. XML, CSV a COBOL) na obchodní objekt a naopak. Protože jsou nezávislé na protokolu, můžete týž popisovač dat opakovaně použít s celou řadou vazeb exportu a importu. Například můžete použít stejný popisovač dat XML s vazbou exportu nebo importu HTTP či s vazbou exportu nebo importu JMS.
- *Vázání dat* také transformují nativní data na obchodní objekt (a naopak), ale jsou specifické pro konkrétní protokol. Například vázání dat HTTP je možné použít pouze s vazbou exportu nebo importu HTTP. Na rozdíl od popisovačů dat nelze vázání dat HTTP znovu použít s vazbou exportu nebo importu MQ.

  Poznámka: Tři vázání dat HTTP (HTTPStreamDataBindingSOAP, HTTPStreamDataBindingXML a HTTPServiceGatewayDataBinding) jsou od produktu IBM Business Process Manager verze 7.0 zamítnuté. Kdykoli to bude možné, používejte popisovače dat.

Jak bylo uvedeno již dříve, v případě potřeby si můžete vytvořit vlastní popisovače dat. Můžete si vytvořit i vlastní vázání dat, ale doporučuje se vytvářet spíše vlastní popisovače dat, protože ty lze použít pro různé vazby.

Popisovače dat:

Popisovače dat jsou nakonfigurované oproti vazbám exportu a importu za účelem transformace dat z jednoho formátu do druhého nezávisle na protokolu. V rámci produktu je poskytováno několik popisovačů dat, v případě potřeby si však můžete vytvořit vlastní popisovač dat. Popisovač dat můžete přidružit k vazbě exportu nebo importu na jedné ze dvou úrovní: můžete jej přidružit ke všem operacím v rozhraní exportu neb importu, nebo jej můžete přidružit ke specifické operaci pro požadavek nebo odezvu.

Předdefinované popisovače dat

Popisovač dat, který chcete použít, určete v produktu IBM Integration Designer.

Popisovače dat, které máte předdefinovány k použití, jsou uvedeny v následující tabulce, která zároveň popisuje, jak jednotlivé popisovače dat transformují příchozí a odchozí data.

Poznámka: Pokud není uvedeno jinak, je možné tyto popisovače dat používat s vazbami JMS, Generické služby JMS, MQ JMS, WebSphere MQ a HTTP.
Podrobnější informace viz téma "Popisovače dat" v Informačním centru produktu Integration Designer.

Tabulka 12. Předdefinované popisovače dat

Popisovač dat	Nativní data na obchodní objekt	Obchodní objekt na nativní data.
ATOM	Analyzuje kanály ATOM do obchodního objektu kanálu ATOM.	Serializuje obchodní objekt kanálu ATOM do kanálů ATOM.
S oddělovačem	Analyzuje data s oddělovačem do obchodního objektu.	Serializuje obchodní objekt do dat s oddělovačem, včetně CSV.
Pevná šířka	Analyzuje data s pevnou šířkou do obchodního objektu.	Serializuje obchodní objekt na data s pevnou šířkou.

Tabulka 12. Předdefinované popisovače dat (pokračování)

Popisovač dat	Nativní data na obchodní objekt	Obchodní objekt na nativní data.
Zpracováno pomocí WTX	Deleguje transformaci formátu dat produktu WebSphere Transformation Extender (WTX). Název mapy WTX je odvozen popisovačem dat.	Deleguje transformaci formátu dat produktu WebSphere Transformation Extender (WTX). Název mapy WTX je odvozen popisovačem dat.
Zpracováno původcem volání WTX	Deleguje transformaci formátu dat produktu WebSphere Transformation Extender (WTX). Název mapy WTX dodává uživatel.	Deleguje transformaci formátu dat produktu WebSphere Transformation Extender (WTX). Název mapy WTX dodává uživatel.
JAXB	Serializuje objekty Java bean na obchodní objekt s použitím pravidel mapování definovaných specifikací JAXB (Java Architecture for XML Binding).	Deserializuje obchodní objekt na objekty Java bean s použitím pravidel mapování definovaných specifikací JAXB.
JAXWS **Poznámka:** Popisovač dat JAXWS lze použít pouze s vázáním EJB	Používán vázáním EJB k transformaci objektu odpovědi Java nebo objektu výjimky Java na obchodní objekt odpovědi s použitím pravidel mapování definovaných specifikací JAX-WS (Java API for XML Web Services).	Používán vázáním EJB k transformaci obchodního objektu na parametry odchozí metody Java s použitím pravidel mapování definovaných specifikací JAX-WS.
JSON	Analyzuje data JSON do obchodního objektu.	Serializuje obchodní objekt do dat JSON.
Nativní tělo	Analyzuje nativní bajty, text, mapu, proud nebo objekt do jednoho z pěti základních obchodních objektů (text, bajty, mapa, proud nebo objekt).	Transformuje pět základních obchodních objektů na bajt, text, mapu, proud nebo objekt.
SOAP	Analyzuje zprávu SOAP (a záhlaví) do obchodního objektu.	Serializuje obchodní objekt do zprávy SOAP.
Kód jazyka XML	Analyzuje data XML do obchodního objektu.	Serializuje obchodní objekt na data XML.
UTF8XMLDataHandler	Analyzuje data XML s kódováním UTF-8 do obchodního objektu.	Serializuje obchodní objekt při odesílání zprávy na data XML s kódováním UTF-8.

Vytvoření popisovače dat

Podrobné informace o vytvoření popisovače dat naleznete v tématu “Vývoj popisovačů dat” v Informačním centru produktu Integration Designer.

Vázání dat:

Vázání dat jsou nakonfigurována oproti vazbám exportu a importu, aby transformovaly data z jednoho formátu do jiného. Vázání dat jsou specifická pro určitý protokol. V rámci produktu je poskytováno několik vázání dat, v případě potřeby si však můžete vytvořit vlastní vázání dat. Vázání dat můžete přidružit k vazbě exportu nebo importu na jedné ze dvou úrovní - můžete ji přidružit ke všem operacím v rozhraní exportu neb importu, nebo ji můžete přidružit ke specifické operaci pro požadavek nebo odezvu.

Pomocí produktu IBM Integration Designer můžete určit, které vázání dat se má použít, nebo vytvořit své vlastní vázání dat. Diskusi o tvorbě vázání dat naleznete v sekci “Přehled vazeb JMS, MQ JMS a generické služby JMS” v Informačním centru produktu IBM Integration Designer.

Vazby JMS

Následující tabulka uvádí vázání dat, s nimiž lze použít:

- Vazby JMS.
- Generické vazby JMS.
- Vazby WebSphere MQ JMS.

Tabulka obsahuje také popis úloh, která provádějí vázání dat.

Tabulka 13. Předdefinovaná vázání dat pro vazby JMS

Vázání dat	Nativní data na obchodní objekt	Obchodní objekt na nativní data.
Serializovaný objekt Java	Transformuje serializovaný objekt Java na obchodní objekt (který je mapován jako typ vstupu nebo výstupu ve WSDL).	Serializuje obchodní objekt na serializovaný objekt Java ve zprávě objektu JMS.
Zalomené bajty.	Extrahuje bajty z příchozí bajtové zprávy JMS a zabalí je do obchodního objektu JMSBytesBody.	Extrahuje bajty z obchodního objektu JMSBytesBody a zabalí je do odchozí bajtové zprávy JMS.
Zabalená položka mapy	Extrahuje informace o názvu, hodnotě a typu pro všechny položky v příchozí zprávě mapy JMS a vytváří seznam obchodních objektů MapEntry. Tento seznam pak zabalí do obchodního objektu JMSMapBody.	Extrahuje informace o názvu, hodnotě a typu ze seznamu položek MapEntry v obchodním objektu JMSMapBody a vytváří odpovídající položky v odchozí zprávě mapy JMS.
Zabalený objekt	Extrahuje objekt z příchozí zprávy objektu JMS a zabalí jej do obchodního objektu JMSObjectBody.	Extrahuje objekt z obchodního objektu JMSObjectBody a zabalí jej do odchozí zprávy objektu JMS.
Zalomený text	Extrahuje text z příchozí textové zprávy JMS a zabalí jej do obchodního objektu JMSTextBody.	Extrahuje text z obchodního objektu JMSTextBody a zabalí jej do odchozí textové zprávy JMS.

Vazby WebSphere MQ

Následující tabulka obsahuje vázání dat, která lze použít s produktem WebSphere MQ, a popisuje úlohy, které tato vázání dat provádějí.

Tabulka 14. Předdefinovaná vázání dat pro vazby WebSphere MQ

Vázání dat	Nativní data na obchodní objekt	Obchodní objekt na nativní data.
Serializovaný objekt Java	Transformuje serializovaný objekt Java z příchozí zprávy na obchodní objekt (který je mapován jako typ vstupu nebo výstupu ve WSDL).	Transformuje obchodní objekt na serializovaný objekt Java v odchozí zprávě.
Zalomené bajty.	Extrahuje bajty z nestrukturované bajtové zprávy MQ a zabalí je do obchodního objektu JMSBytesBody.	Extrahuje bajty z obchodního objektu JMSBytesBody a zabalí tyto bajty do odchozí nestrukturované bajtové zprávy MQ.
Zalomený text	Extrahuje text z nestrukturované textové zprávy MQ a zabalí jej do obchodního objektu JMSTextBody.	Extrahuje text z obchodního objektu JMSTextBody a zabalí jej do nestrukturované textové zprávy MQ.
Zabalená položka proudu	Extrahuje informace o názvu a typu pro všechny položky v příchozí zprávě proudu JMS a vytváří seznam obchodních objektů StreamEntry. Tento seznam pak zabalí do obchodního objektu JMSStreamBody.	Extrahuje informace o názvu a typu ze seznamu položek StreamEntry v obchodním objektu JMSStreamBody a vytváří odpovídající položky v odchozí zprávě JMSStreamMessage.

Kromě vázání dat uvedených v seznamu Tabulka 14 na stránce 57 používá WebSphere MQ také vázání dat záhlaví. Podrobnosti naleznete v Informačním centru produktu IBM Integration Designer.

Vazby HTTP

Následující tabulka obsahuje vazby dat, které lze použít s HTTP, a popisuje úlohy prováděné vázáními dat.

Tabulka 15. Předdefinovaná vázání dat pro vazby HTTP

Vázání dat	Nativní data na obchodní objekt	Obchodní objekt na nativní data.
Zalomené bajty.	Extrahuje bajty z těla příchozí zprávy HTTP a zabalí je do obchodního objektu HTTPBytes.	Extrahuje bajty z obchodního objektu HTTPBytes a přidá je do těla odchozí zprávy HTTP.
Zalomený text	Extrahuje text z těla příchozí zprávy HTTP a zabalí jej do obchodního objektu HTTPText.	Extrahuje text z obchodního objektu HTTPText a přidá jej do těla odchozí zprávy HTTP.

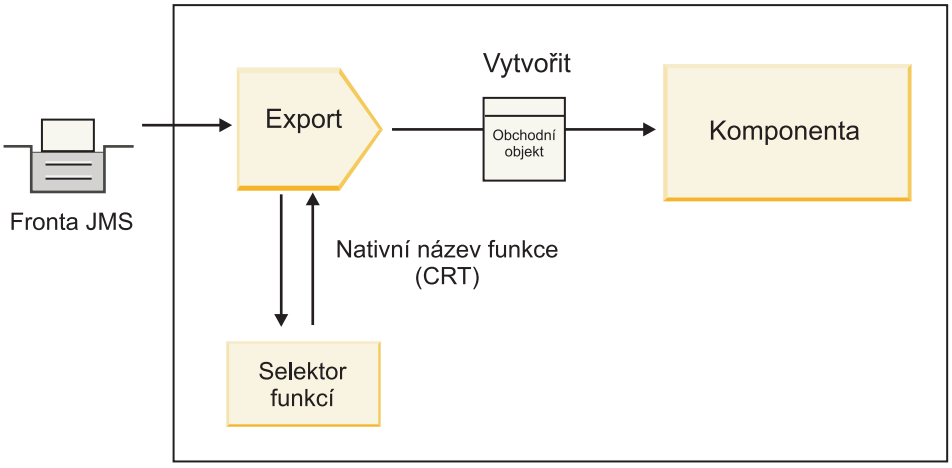
Selektory funkcí ve vazbě exportu:

Selektor funkcí se používá k určení toho, která operace se má provést s daty pro zprávu požadavku. Selektory funkcí jsou nakonfigurovány jako součást vazby exportu.

Zvažte export SCA, který vystavuje rozhraní. Toto rozhraní obsahuje dvě operace: Vytvořit a Aktualizovat. Export má vazbu JMS, která čte z fronty.

Když dorazí zpráva do fronty, jsou exportu předána přidružená data, ale která operace z rozhraní exportu má být vyvolána nad spojenou komponentou? Tuto operaci určuje selektor funkcí a konfigurace vazby exportu..

Selektor funkcí vrací název nativní funkce (název funkce v klientském systému, který zprávu odeslal). Název nativní funkce je pak namapován na název operace nebo funkce v rozhraní přidruženém k exportu. Například na následujícím obrázku vrací selektor funkcí název nativní funkce (CRT) z příchozí zprávy, tento název nativní funkce je namapován na operaci Vytvořit a obchodní objekt je odeslán komponentě SCA s operací Vytvořit.



Obrázek 7. Selektor funkcí

Pokud má rozhraní pouze jednu operaci, není třeba určovat selektor funkcí.

K dispozici je několik předem seskupených selektorů funkcí, které jsou uvedeny v následujících sekcích.

Vazby JMS

Následující tabulka uvádí selektory funkcí, s nimiž lze použít:

- Vazby JMS.
- Generické vazby JMS.
- Vazby WebSphere MQ JMS.

Tabulka 16. Předdefinované selektory funkcí pro vazby JMS

Selektor funkcí	Popis
Selektor funkcí JMS pro jednoduchá vázání dat JMS	Používá vlastnost JMSType zprávy pro výběr názvu operace.
Selektor funkcí vlastnosti záhlaví JMS	Vrací hodnotu vlastnosti řetězce JMS TargetFunctionName ze záhlaví.
Selektor funkcí brány služeb JMS	Určuje, zda je požadavek jednosměrná nebo obousměrná operace prozkoumáním vlastnosti JMSReplyTo nastavené klientem.

Vazby WebSphere MQ

Následující tabulka uvádí selektory funkcí, které lze použít s vazbami WebSphere MQ.

Tabulka 17. Předdefinované selektory funkcí pro vazby WebSphere MQ

Selektor funkcí	Popis
Selektor funkcí MQ handleMessage	Vrací handleMessage jako hodnotu, která je mapována pomocí vazeb metody exportu na název operace na rozhraní.
MQ používá výchozí selektor funkcí JMS	Načte nativní operaci z vlastnosti TargetFunctionName složky záhlaví MQRFH2.
MQ používá formát těla zprávy jako nativní funkci	Vyhledá pole Formát posledního záhlaví a vrací toto pole jako řetězec.
Selektor funkcí typu MQ	Vytvoří ve vaší vazbě exportu metodu načtením adresy URL obsahující vlastnosti Msd, Set, Type a Format, které se nacházejí v záhlaví MQRFH2.
Selektor funkcí brány služeb MQ	Používá vlastnost MsgType v záhlaví MQMD k určení názvu operace.

Vazby HTTP

Následující tabulka uvádí selektory funkcí, které lze použít s vazbami HTTP.

Tabulka 18. Předdefinované selektory funkcí pro vazby HTTP

Selektor funkcí	Popis
Selektor funkcí HTTP na základě záhlaví TargetFunctionName	Používá vlastnost záhlaví HTTP TargetFunctionName z klienta k určení operace, která se má vyvolat za běhu z exportu.
Selektor funkcí HTTP založený na adrese URL a metodě HTTP	Používá relativní cestu z adresy URL připojené k metodě HTTP z klienta k určení nativní operace definované na exportu.
Selektor funkcí brány služeb HTTP založený na adrese URL s názvem operace	Určuje metodu, která se má vyvolat, na základě adresy URL, pokud bylo k adrese URL požadavku připojeno "operationMode = oneWay".

Poznámka: Můžete si vytvořit i vlastní Selektor funkcí pomocí nástroje IBM Integration Designer. Informace o vytvoření selektoru funkcí nabízí Informační centrum pro IBM Integration Designer. Například popis vytvoření selektoru funkcí pro vazby WebSphere MQ naleznete v tématu “Přehled selektorů funkcí pro MQ”.

Zpracování poruch:

Vazby importu a exportu můžete nakonfigurovat tak, aby zpracovávaly poruchy (například obchodní výjimky), k nimž dochází při zpracování určením popisovačů dat způsobujících poruchu. Popisovač dat způsobujících poruchu můžete nastavit na třech úrovních - popisovač dat způsobujících poruchu můžete přidružit k poruše, k operaci nebo pro všechny operace s vazbou.

Popisovač dat způsobující poruchu zpracovává data způsobující poruchu a transformuje je do správného formátu, který má být odeslán vazbou exportu nebo importu.

- V případě vazby exportu popisovač dat způsobující poruchu transformuje obchodní objekt výjimky zaslaný z komponenty na zprávu odpovědi, kterou může použít aplikace klienta.
- V případě vazby importu popisovač dat způsobující poruchu transformuje data způsobující poruchu nebo zprávu odpovědi zaslanou ze služby na obchodní objekt výjimky, který může použít komponenta SCA.

V případě vazeb importu volá tato vazba selektor poruch, který určí, zda je zpráva odpovědi normální odpověď, obchodní porucha nebo výjimka za běhu.

Můžete určit popisovač dat způsobující poruchu pro konkrétní poruchu, pro nějakou operaci a pro všechny operace s vazbou.

- Pokud je popisovač dat způsobující poruchu nastaven na všech třech úrovních, je volán popisovač dat přidružený ke konkrétní poruše.
- Pokud jsou popisovače dat způsobujících poruchu nastaveny na úrovni operace a vazby, je volán popisovač dat přidružený k operaci.

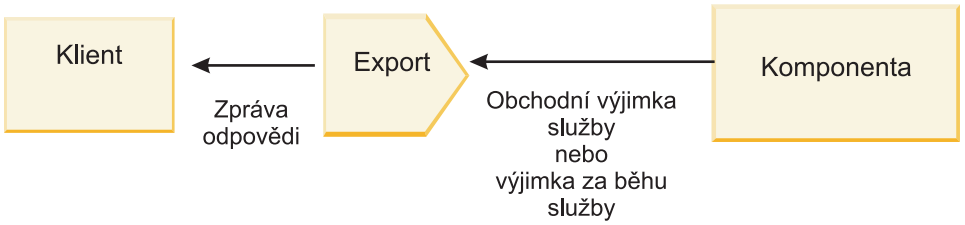
V produktu IBM Integration Designer se k určení zpracování poruch používají dva editory. Editor rozhraní se používá k určení, zda na nějaké operaci dojde k poruše. Po vygenerování vazby pomocí tohoto rozhraní vás nechá tento v zobrazení vlastností nakonfigurovat, jak bude porucha ošetřena. Další informace viz téma “Selektory poruch” v Informačním centru produktu IBM Integration Designer.

Jak jsou ošetřovány poruchy ve vazbách exportu:

Když dojde k poruše při zpracování požadavku od aplikace klienta, může vazba exportu klientovi vrátit informace o poruše. Nakonfigurujete vazbu exportu, aby určovala, jak má být porucha zpracována a vracena klientovi.

Vazbu exportu nakonfigurujete pomocí produktu IBM Integration Designer.

Při zpracování požadavků vyvolá klient export s požadavkem a tento export vyvolá komponentu SCA. Při zpracování požadavku může komponenta SCA buď vrátit obchodní odpověď, nebo může vrátit obchodní výjimku služby nebo výjimku služby za běhu. Když k tomu dojde, transformuje vazba exportu výjimku na chybovou zprávu a odešle ji klientovi, jak ukazuje obrázek níže a popisují následující sekce.



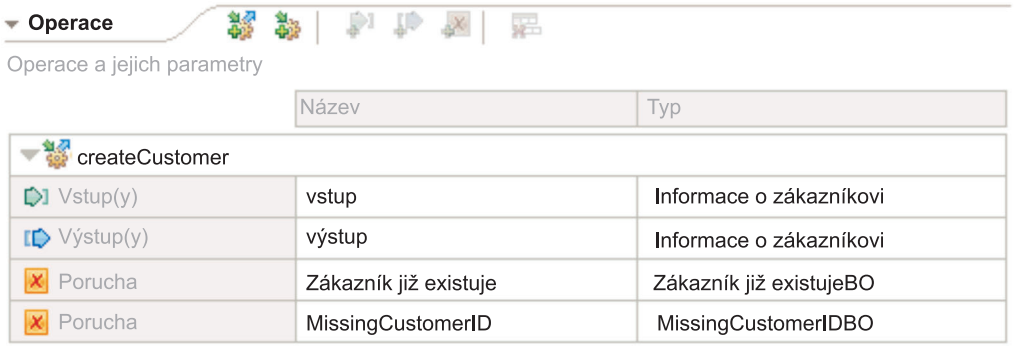
Obrázek 8. Jak jsou informace o poruchách odesílány z komponenty prostřednictvím vazby exportu klientovi

Můžete si vytvořit vlastní popisovač dat nebo vázání dat pro zpracování poruch.

Obchodní poruchy

Obchodní poruchy jsou obchodní chyby nebo výjimky, k nimž dochází při zpracování.

Zvažte použití následujícího rozhraní, kde je definována operace createCustomer. Tato operace má definovány dvě obchodní poruchy: CustomerAlreadyExists a MissingCustomerId.



Obrázek 9. Rozhraní se dvěma poruchami

Pokud v tomto příkladě klient odešle požadavek na vytvoření zákazníka (této komponentě SCA) a daný zákazník již existuje, vrátí komponenta exportu poruchu CustomerAlreadyExists. Export potřebuje tuto obchodní poruchu předat zpět volajícímu klientovi. K tomu použije popisovač dat způsobující poruchu, který je nastaven na vazbě exportu.

Když vazba exportu přijme obchodní poruchu, dojde k tomuto zpracování:

1. Vazba určí, který popisovač dat způsobující poruchu se má vyvolat, aby danou poruchu ošetřil. Pokud obchodní výjimka služby obsahuje název poruchy, je volán popisovač dat, který je nastaven na dané poruše. Pokud obchodní výjimka služby název poruchy neobsahuje, je tento název odvozen porovnáním typů poruch.
2. Vazba zavolá popisovač dat způsobující poruchu s datovým objektem z obchodní výjimky služby.
3. Popisovač dat způsobující poruchu transformuje datový objekt poruchy na zprávu odpovědi a tu vrátí vazbě exportu.
4. Export vrátí zprávu odpovědi klientovi.

Pokud obchodní výjimka služby obsahuje název poruchy, je volán popisovač dat, který je nastaven na dané poruše. Pokud obchodní výjimka služby název poruchy neobsahuje, je tento název odvozen porovnáním typů poruch.

Výjimky za běhu

Výjimka za běhu je výjimka, k níž dojde v aplikaci SCA při zpracování požadavku, která neodpovídá žádné obchodní poruše. Na rozdíl od obchodních poruch nejsou výjimky za běhu definovány v rozhraní.

V určitých scénářích můžete chtít tyto výjimky za běhu předat aplikaci klienta, aby aplikace mohla podniknout odpovídající akci.

Pokud například klient odešle požadavek (této komponentě SCA) na vytvoření zákazníka a při zpracování tohoto požadavku dojde k chybě autorizace, vrátí komponenta výjimku za běhu. Tato výjimka za běhu musí být předána zpět volajícímu klientovi, aby mohl podniknout příslušnou akci ohledně autorizace. Toho je dosaženo prostřednictvím popisovače dat výjimky za běhu nakonfigurovaného pro vazbu exportu.

Poznámka: Popisovač dat výjimek za běhu můžete nakonfigurovat pouze pro vazby HTTP.

Zpracování výjimky za běhu je podobné zpracování obchodní poruchy. Pokud byl nastaven popisovač dat výjimky za běhu, dojde k tomuto zpracování:

1. Vazba exportu zavolá příslušný popisovač dat s výjimkou služby za běhu.
2. Popisovač dat transformuje datový objekt poruchy na zprávu odpovědi a tu vrátí vazbě exportu.
3. Export vrátí zprávu odpovědi klientovi.

Zpracování poruch a zpracování výjimek za běhu jsou volitelná. Pokud nechcete poruchy nebo výjimky za běhu předávat volajícímu klientovi, nekonfigurujte popisovač dat způsobující poruchu či popisovač dat výjimky za běhu.

Jak jsou ošetřovány poruchy ve vazbách importu:

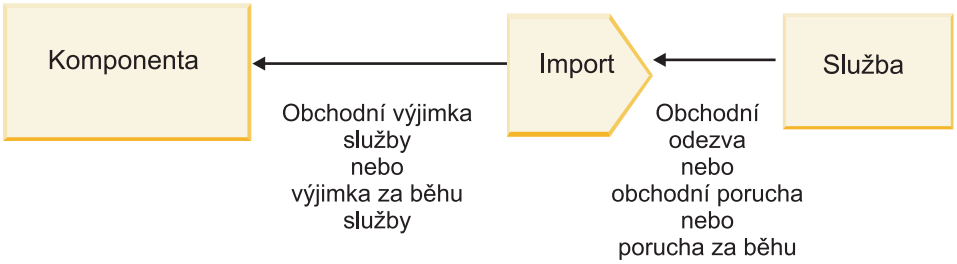
Komponenta používá import k odeslání požadavku službě mimo modul. Když dojde k poruše při zpracování požadavku, vrátí služba poruchu vazbě importu. Vazbu importu můžete nakonfigurovat tak, aby určovala, jak má být porucha zpracována a vrácena komponentě.

Vazbu importu nakonfigurujete pomocí produktu IBM Integration Designer. Můžete určit popisovač dat způsobující poruchu (nebo vázání dat) a také selektor poruch.

Popisovače dat způsobujících poruchu

Služba, která zpracovává požadavek, pošle vazbě importu informace o poruše ve formě výjimky nebo zprávy odpovědi, která obsahuje data způsobující poruchu.

Vazba importu transformuje výjimku služby nebo zprávu odpovědi na obchodní výjimku služby nebo výjimku služby za běhu, jak ukazuje následující obrázek a popisují následující sekce.



Obrázek 10. Jak jsou informace o poruchách odesílány ze služby prostřednictvím importu komponentě

Můžete si vytvořit vlastní popisovač dat nebo vázání dat pro zpracování poruch.

Selektory poruch

Když konfigurujete vazbu importu, můžete určit selektor poruch. Selektor poruch určuje, zda odpověď importu má být skutečnou odpovědí, obchodní výjimkou nebo běhovou poruchou. Také určuje z těla nebo záhlaví odpovědi nativní název poruchy, který konfigurace vazby mapuje na název poruchy v přidruženém rozhraní.

K dispozici jsou dva typy předem seskupených selektorů poruch, které lze používat s importy JMS, MQ JMS, Generické služby JMS, WebSphere MQ a HTTP:

Tabulka 19. Předem seskupené selektory poruch

Typ selektoru poruch	Popis
Podle záhlaví	Určuje, zda je zpráva odpovědi obchodní porucha, výjimka za běhu nebo normální zpráva podle záhlaví v příchozí zprávě odpovědi.

Tabulka 19. Předem seskupené selektory poruch (pokračování)

Typ selektoru poruch	Popis
SOAP	Určuje, zda je zpráva SOAP odpovědi normální zpráva, obchodní porucha nebo výjimka za běhu.

Následujíc příklady selektorů poruch podle záhlaví a selektorů poruch SOAP.

- Selektor poruch podle záhlaví

  Pokud chce aplikace označit, že je příchozí zpráva obchodní porucha, musí být v příchozí zprávě dvě záhlaví pro obchodní poruchy, což vypadá takto:

  ```
  Header name = FaultType, Header value = Business
  Header name = FaultName, Header value = <nativní název poruchy definovaný uživatele,>
  ```

  Pokud chce aplikace označit, že je příchozí zpráva odpovědi výjimka za běhu, musí být v příchozí zprávě jedno záhlaví, což vypadá takto:

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

- Selektor poruch SOAP

  Obchodní poruchu lze poslat jako součást zprávy SOAP s následujícím vlastním záhlavím SOAP. V tomto případě je název poruchy "CustomerAlreadyExists".

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

Selektor poruch je nepovinný. Pokud neurčíte žádný selektor poruch, vazba importu nemůže určit typ odpovědi. Proto ji vazba považuje za obchodní odpověď a volá popisovač dat nebo vázání dat odpovědi.

Můžete si vytvořit vlastní selektor poruch. Postup pro vytvoření vlastního selektoru poruch viz “Vývoj vlastního selektoru poruch” v Informačním centru produktu IBM Integration Designer.

Obchodní poruchy

K obchodní poruše může dojít, když nastane chyba při zpracování požadavku. Pokud například odešlete požadavek na vytvoření zákazníka a tento zákazník již existuje, odešle služba vazbě importu obchodní výjimku.

Když vazba obdrží obchodní výjimku, závisí kroky zpracování na tom, zda byl pro tuto vazbu nastaven selektor poruch.

- Pokud nebyl nastaven žádný selektor poruch, volá vazba popisovač dat nebo vázání dat odpovědi.
- Pokud byl nastaven selektor poruch, dojde k tomuto zpracování:
  1. Vazba importu volá selektor poruch, aby určil, zda je odpověď obchodní porucha, obchodní odpověď nebo porucha za běhu.
  2. Pokud je odpověď obchodní porucha, volá vazba importu selektor poruch, aby poskytl nativní název poruchy.
  3. Vazba importu určí poruchu WSDL odpovídající nativnímu názvu poruchy vrácenému selektorem poruch.
  4. Vazba importu určí popisovač dat způsobující poruchu, který je nakonfigurován pro tuto poruchu WSDL.
  5. Vazba importu volá tento popisovač dat způsobující poruchu s daty o poruše.
  6. Popisovač dat způsobující poruchu transformuje data o poruše na datový objekt a ten vrátí vazbě importu.
  7. Vazba importu zkonstruuje objekt obchodní výjimky služby s datovým objektem a názvem poruchy.
  8. Import vrátí objekt obchodní výjimky služby komponentě.

Výjimky za běhu

K výjimce za běhu může dojít, když nastane problém při komunikaci se službou. Zpracování výjimky za běhu je podobné zpracování obchodní výjimky. Pokud byl nastaven selektor poruch, dojde k tomuto zpracování:

1. Vazba importu volá příslušný popisovač dat výjimky za běhu s daty výjimky.
2. Popisovač dat výjimky za běhu transformuje data výjimky na objekt výjimky služby za běhu a ten vrátí vazbě importu.
3. Import vrátí objekt výjimky služby za běhu komponentě.

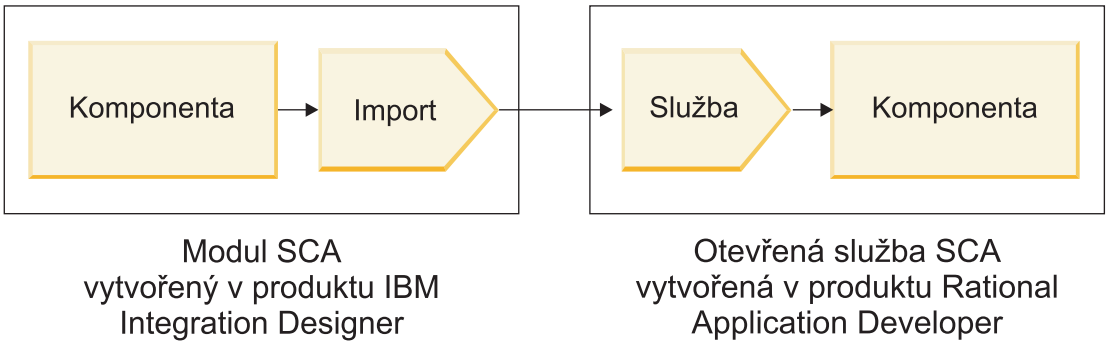
Interoperabilita mezi moduly SCA a službami Open SCA

Balík funkcí IBM WebSphere Application Server V7.0 Feature Pack for Service Component Architecture (SCA) nabízí jednoduchý, avšak účinný programovací model pro tvorbu aplikací na základě specifikací Open SCA. Moduly SCA produktu IBM Business Process Manager používají vazby importu a exportu ke spolupráci se službami Open SCA vyvinutými v prostředí produktu Rational Application Developer s hostitelem serveru WebSphere Application Server Feature Pack for Service Component Architecture.

Aplikace SCA vyvolá aplikaci Open SCA prostřednictvím vazby importu. Aplikace SCA přijme volání od aplikace Open SCA prostřednictvím vazby exportu. Seznam podporovaných vazeb ukazuje “Vyvolání služeb přes spolupracující vazby” na stránce 65.

Vyvolání služeb Open SCA z modulů SCA

Aplikace SCA vyvinuté v produktu IBM Integration Designer mohou vyvolávat aplikace Open SCA vyvinuté v prostředí produktu Rational Application Developer. Tato část obsahuje příklad vyvolání služby Open SCA z modulu SCA pomocí vazby importu SCA.



Obrázek 11. Komponenta v modulu SCA vyvolávající službu Open SCA

K vyvolání služby Open SCA není nutná žádná speciální konfigurace.

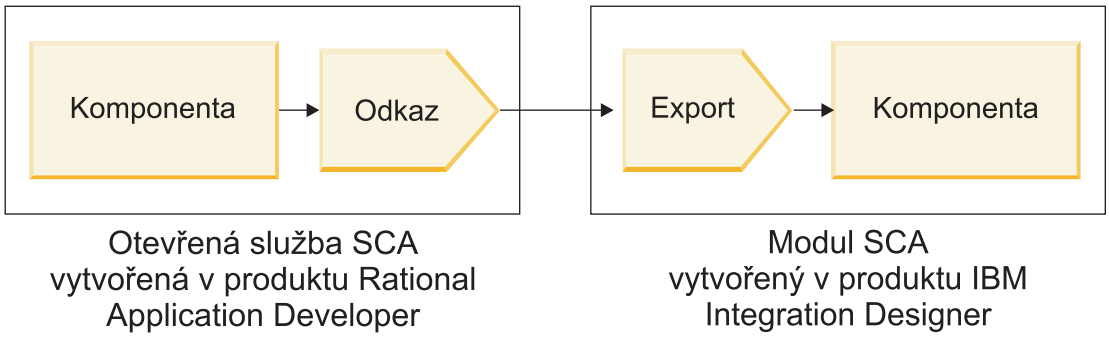
Chcete-li se připojit ke službě Open SCA prostřednictvím vazby importu SCA, musíte zadat název komponenty a název služby Open SCA ve vazbě importu.

1. Chcete-li získat název cílové komponenty a služby ze složeného prvku SCA, postupujte takto:
   a. Zkontrolujte, zda je otevřená karta **Vlastnosti**, klepnutím na volbu **Okno > Zobrazit pohled > Vlastnosti**.
   b. Otevřete editor složených objektů poklepáním na diagram složených objektů, který obsahuje danou komponentu nebo službu. Například diagram složeného objektu pro komponentu nazvanou zákazník (**customer**) je **customer.composite_diagram**.
   c. Klepněte na cílovou komponentu.
   d. Poznamenejte si název cílové komponenty z pole **Název** na kartě **Vlastnosti**.
   e. Klepněte na ikonu služby přidružené k dané komponentě.
   f. Poznamenejte si název služby z pole **Název** na kartě **Vlastnosti**.
2. Chcete-li nakonfigurovat import produktu IBM Business Process Manager pro připojení ke službě Open SCA, postupujte takto:

a. V produktu IBM Integration Designer přejděte na kartu **Vlastnosti** importu SCA, který chcete připojit ke službě Open SCA.
b. Do pole **Název modulu** zadejte název komponenty z kroku 1d na stránce 64.
c. Do pole **Název exportu** zadejte název služby z kroku 1f na stránce 64.
d. Uložte svou práci stisknutím kláves Ctrl+S.

Vyvolání modulů SCA ze služeb Open SCA

Aplikace Open SCA vyvinuté v prostředí produktu Rational Application Developer mohou vyvolávat aplikace SCA vyvinuté v produktu IBM Integration Designer. Tato část obsahuje příklad vyvolání modulu SCA (prostřednictvím vazby exportu SCA) ze služby Open SCA.



Obrázek 12. Služba Open SCA vyvolávající komponentu v modulu SCA

Chcete-li připojit komponentu SCA prostřednictvím vazby odkazu Open SCA, musíte zadat název modulu a název exportu.

1. Chcete-li získat název cílového modulu a exportu, postupujte takto:
   a. V produktu IBM Integration Designer otevřete daný modul poklepáním na tento modul v editoru sestavení.
   b. Klepněte na export.
   c. V poli **Název** na kartě **Vlastnosti** zadejte název exportu.
2. Nakonfigurujte odkaz Open SCA, který chcete připojit k modulu a exportu produktu IBM Business Process Manager:
   a. V produktu Rational Application Developer otevřete editor složených objektů poklepáním na diagram složených objektů, který obsahuje danou komponentu nebo službu.
   b. Klepněte na ikonu odkazu pro odkaz na komponentu a zobrazí se jeho vlastnosti na kartě **Vlastnosti**.
   c. Klepněte na kartu **Vazba** na levé straně stránky.
   d. Klepněte na volbu **Vazby** a potom na tlačítko **Přidat**.
   e. Vyberte vazbu **SCA**.
   f. Do pole **Identifikátor URI** zadejte název modulu produktu IBM Business Process Manager, následovaný lomítkem (“/”), následovaným názvem exportu (který jste určili v kroku 1c).
   g. Klepněte na tlačítko **OK**.
   h. Uložte svou práci stisknutím kláves Ctrl+S.

Vyvolání služeb přes spolupracující vazby

Následující vazby jsou podporovány pro součinnost se službou Open SCA.

- Vazba SCA

  Když v produktu IBM Business Process Manager modul SCA vyvolává službu Open SCA prostřednictvím vazby importu SCA, jsou podporovány tyto styly vyvolání:

– Asynchronní (jednosměrný).
– Synchronní (požadavek/odpověď)

Rozhraní importu SCA a rozhraní služby Open SCA musí využívat interoperabilitu webových služeb (WS-I) kompatibilní s rozhraním WSDL.

Všimněte si, že vazba SCA podporuje šíření kontextu zabezpečení a transakcí.

- Vazba webové služby (JAX-WS) s protokolem SOAP1.1/HTTP nebo SOAP1.2/HTTP

  Rozhraní importu SCA a rozhraní služby Open SCA musí využívat interoperabilitu webových služeb (WS-I) kompatibilní s rozhraním WSDL.

  Navíc jsou podporovány tyto kvality služby:
  – Web Services Atomic Transaction.
  – Web Services Security.
- Vázání EJB

  Rozhraní Java se používá k definování interakce mezi modulem SCA a službou Open SCA, když je použito vázání EJB.

  Poznámka: vázání EJB podporuje šíření kontextu zabezpečení a transakcí.
- Vazby JMS

  Rozhraní importu SCA a rozhraní služby Open SCA musí využívat interoperabilitu webových služeb (WS-I) kompatibilní s rozhraním WSDL.

  Jsou podporováni tito poskytovatelé JMS:
  – WebSphere Platform Messaging (Vazba JMS).
  – WebSphere MQ (Vazba MQ JMS).

Poznámka: Obchodní grafy neumožňují interoperabilitu napříč vazbami SCA, a proto nejsou podporovány v rozhraních používaných pro interoperabilitu s balíkem funkcí serveru WebSphere Application Server Feature Pack for Service Component Architecture.

Typy vazeb

Vazby specifické pro určitý protokol se používají s importy a exporty k určení prostředků přenosu dat do modulu nebo z modulu.

Výběr odpovídajících vazeb:

Při vytváření aplikace musíte vědět, jak vybrat nejvhodnější vazbu pro potřebu vaší aplikace.

Vazby, které jsou k dispozici v produktu IBM Integration Designer, nabízejí celou řadu voleb. V tomto seznamu můžete určit, který typ vazby může být pro potřeby vaší aplikace nejvhodnější.

Pokud se na daný případ vztahují následující faktory, zvažte použití vazby *SCA (Service Component Architecture)*:

- Veškeré služby jsou obsaženy v modulech, neexistují tedy žádné externí služby.
- Chcete rozdělit funkci do různých modulů SCA, které spolu přímo komunikují.
- Moduly jsou pevně přiřazeny.

Pokud se na daný případ vztahují následující faktory, zvažte použití *vazby webové služby*:

- Je vyžadován přístup k externí službě nebo poskytování služby prostřednictvím sítě Internet.
- Služby jsou volně přiřazeny.
- Je preferována synchronní komunikace; tj. požadavek z jedné služby může čekat na odezvu z jiné.
- Protokol externí služby, ke které přistupujete, nebo služby, kterou chcete poskytovat, je SOAP/HTTP nebo SOAP/JMS.

Pokud se na daný případ vztahují následující faktory, zvažte použití *vazby HTTP*:

- Potřebujete internetový přístup k externí službě nebo poskytovat službu po Internetu a pracujete s jinými webovými službami, jako jsou GET, PUT či DELETE.
- Služby jsou volně přiřazeny.
- Je preferována synchronní komunikace; tj. požadavek z jedné služby může čekat na odezvu z jiné.

Pokud se na daný případ vztahují následující faktory, zvažte použití vazby *EJB (Enterprise JavaBeans)*:
- Vazba je určena pro importovanou službu, která je sama objektem EJB, nebo ke které potřebují mít přístup klienti EJB.
- Importovaná služba je volně přiřazena.
- Nejsou požadovány stavové interakce EJB.
- Je preferována synchronní komunikace; tj. požadavek z jedné služby může čekat na odezvu z jiné.

Pokud se na daný případ vztahují následující faktory, zvažte použití vazby *EIS (Enterprise Information Systems)*:
- Ke službě v systému EIS je nutné přistupovat prostřednictvím adaptéru prostředku.
- Je preferován synchronní přenos dat před asynchronním.

Pokud se na daný případ vztahují následující faktory, zvažte použití vazby *JMS (Java Message Service)*:

Důležité: Existuje několik typů vazeb JMS. Pokud očekáváte výměnu zpráv SOAP prostřednictvím služby JMS, zvažte použití vazby webové služby s protokolem SOAP/JMS. Viz “Vazby webové služby” na stránce 68.
- Je nutné přistupovat k systému zpráv.
- Služby jsou volně přiřazeny.
- Je preferován asynchronní přenos dat před synchronním.

Pokud se na daný případ vztahují následující faktory, zvažte použití vazby *Generické JMS (Java Message Service)*:
- Je nutné přistupovat k systému zpráv jiných dodavatelů než IBM.
- Služby jsou volně přiřazeny.
- Spolehlivost je důležitější než výkon, je tedy preferován asynchronní přenos dat před synchronním.

Pokud se na daný případ vztahují následující faktory, zvažte použití vazby *MQ (Message Queue)*:
- Je nutné přistupovat k systému zpráv produktu WebSphere MQ a používat nativní funkce MQ.
- Služby jsou volně přiřazeny.
- Spolehlivost je důležitější než výkon, je tedy preferován asynchronní přenos dat před synchronním.

Pokud se na daný případ vztahují následující faktory, zvažte použití *vazby MQ JMS*:
- Potřebujete přistupovat k systému zpráv produktu WebSphere MQ, ale lze tak činit v kontextu služby JMS. Podmnožina funkcí JMS je tedy pro vaši aplikaci dostatečná.
- Služby jsou volně přiřazeny.
- Spolehlivost je důležitější než výkon, je tedy preferován asynchronní přenos dat před synchronním.

Vazby SCA:

Vazba architektury SCA (Service Component Architecture) umožní službě komunikovat s dalšími službami v jiných modulech. Import s vazbou SCA umožňuje přistupovat ke službě v jiném modulu SCA. Export s vazbou SCA umožňuje nabízet služby jiným modulům.

Pomocí produktu IBM Integration Designer generujete a konfigurujete vazby SCA na importy a exporty v modulech SCA.

Pokud jsou moduly spuštěné na stejném serveru nebo jsou implementované ve stejném klastru, je nejjednodušší a nejrychleji použitelnou vazbou vazba SCA.

Po implementaci modulu, který obsahuje danou vazbu SCA, na serveru můžete v administrativní konzole zobrazit informace o této vazbě, nebo v případě vazby importu změnit její vybrané vlastnosti.

Vazby webové služby:

Vazba webové služby je prostředek pro přenos zpráv z komponenty SCA (Service Component Architecture) do webové služby (a obráceně).

Přehled vazeb webové služby:

Vazba importu webové služby umožňuje volání externí webové služby z komponent SCA (Service Component Architecture). Vazba exportu webové služby umožňuje odkrytí komponent SCA klientům v podobě webových služeb.

Pomocí vazby webové služby přistupujete k externím službám za použití interoperabilních zpráv SOAP a QoS (Qualities of Service).

Pomocí produktu Integration Designer generujete a konfigurujete vazby webové služby na importy a exporty v modulech SCA. K dispozici jsou následující typy vazeb webových služeb:

- SOAP1.2/HTTP a SOAP1.1/HTTP:

  Tyto vazby jsou založené na programovacím rozhraní API jazyka Java pro vytváření webových služeb, označovaném jako JAX-WS (Java API for XML Web Services).

  – Pokud vaše webová služba odpovídá specifikaci SOAP 1.2, použijte vazbu SOAP1.2/HTTP.
  – Pokud vaše webová služba odpovídá specifikaci SOAP 1.1, použijte vazbu SOAP1.1/HTTP.

  Důležité: Když implementujete aplikaci s vazbou webové služby (JAX-WS), nesmí být na cílovém serveru vybrána volba **Spouštět komponenty podle potřeby**. Podrobnosti viz “Kontrola konfigurace serveru” na stránce 75.

  Když vyberete jednu z těchto vazeb, můžete odesílat zprávy SOAP a přílohami.

  Vazby webové služby pracují se standardními zprávami SOAP. Použitím vazeb webové služby JAX-WS můžete nicméně upravit způsob analýzy či zápisu zpráv SOAP. Můžete například zpracovat nestandardní prvky zpráv SOAP nebo použít u zprávy SOAP dodatečné zpracování. Když vazbu konfigurujete, určíte vlastní popisovač dat, který toto zpracování zprávy SOAP provede.

  S vazbou webové služby (JAX-WS) můžete použít sady zásad. Sada zásad je kolekce typů zásad, z nichž každá zajišťuje kvalitu služeb (QoS). Například sada zásad WSAddressing poskytuje způsob jednotného adresování webových služeb a zpráv, který je neutrální vůči transportu. Sadu zásad pro danou vazbu vybíráte prostřednictvím produktu Integration Designer.

  Poznámka: Pokud chcete použít sadu zásad SAML (Security Assertion Markup Language), musíte provést dodatečnou konfiguraci, viz “Importování sad zásad SAML” na stránce 73.

- SOAP1.1/HTTP:

  Tuto vazbu použijte, pokud chcete vytvořit webové služby používající zprávu kódovanou pomocí protokolu SOAP, založenou na rozhraní JAX-RPC (Java API for XML-based RPC).

- SOAP1.1/JMS:

  Tuto vazbu použijte k odeslání či přijetí zpráv SOAP pomocí cíle JMS (Java Message Service).

Bez ohledu na to, jaký přenos (HTTP či JMS) je pro doručení zprávy SOAP použit, vazby webové služby vždy zpracovávají interakce požadavek/odezva synchronně. Podproces provádějící vyvolání poskytovatele služeb je blokován, dokud nedojde k obdržení odezvy od poskytovatele. Další informace o tomto stylu vyvolání viz “Synchronní vyvolání”.

Důležité: Následující kombinace vazeb webových služeb nelze použít pro exporty v rámci jednoho modulu. Pokud potřebujete vystavit komponenty pomocí více než jedné z těchto vazeb exportu, musíte každou z nich mít v odděleném modulu a tyto moduly potom musíte s vašimi komponentami propojit pomocí vazby SCA:

- SOAP 1.1/JMS a SOAP 1.1/HTTP pomocí JAX-RPC.

- SOAP 1.1/HTTP pomocí JAX-RPC a SOAP 1.1/HTTP pomocí JAX-WS.
- SOAP 1.1/HTTP pomocí JAX-RPC a SOAP 1.2/HTTP pomocí JAX-WS.

Po implementaci modulu SCA obsahujícího vazbu webové služby na server si můžete pomocí administrativní konzoly nechat zobrazit informace o dané vazbě, nebo můžete změnit vybrané vlastnosti této vazby.

Poznámka: Webové služby umožňují aplikacím spolupracovat, protože používají standardní popisy služeb a standardní formáty vyměňovaných zpráv. Vazby importu a exportu webové služby mohou například spolupracovat se službami implementovanými pomocí WSE (Web Services Enhancements) verze 3.5 a WCF (Windows Communication Foundation) verze 3.5 pro Microsoft .NET. V případě spolupráce s takovými službami dbejte na následující:

- Soubor WSDL (Web Services Description Language) použitý pro přístup k exportu webové služby musí obsahovat pro každou operaci v daném rozhraní neprázdnou hodnotu akce SOAP.
- Klient webové služby musí při odesílání zpráv do exportu webové služby nastavit buď záhlaví SOAPAction, nebo wsa:Action.

Šíření záhlaví SOAP:

Při obsluze zpráv SOAP může být zapotřebí přistupovat k informacím z určitých záhlaví SOAP přijímaných zpráv. Zajistěte, aby byly zprávy se záhlavími SOAP odesílány se specifickými hodnotami nebo umožněte záhlavím SOAP minout určitý modul.

Při konfiguraci vazby webové služby v produktu Integration Designer můžete označit, že chcete záhlaví SOAP šířit.

- Při obdržení požadavků při exportu nebo odezev při importu lze přistupovat k informacím ze záhlaví SOAP, což umožňuje založit logiku modulu na hodnotách daného záhlaví a umožňuje to také úpravu těchto záhlaví.
- Když jsou požadavky odeslány z exportu nebo odezvy z importu, lze do těchto zpráv zahrnout záhlaví SOAP.

Forma a přítomnost šířených záhlaví SOAP může být ovlivněna sadami zásad nakonfigurovanými pro import či export, viz Tabulka 20 na stránce 70.

Chcete-li nakonfigurovat šíření záhlaví SOAP pro import či export, vyberte (v pohledu Vlastnosti produktu Integration Designer) kartu **Šířit záhlaví protokolu** a vyberte vámi požadované volby.

Záhlaví WS-Addressing

Záhlaví WS-Addressing lze šířit prostřednictvím vazby webové služby (JAX-WS).

Při šíření záhlaví WS-Addressing počítejte s následujícími faktory:

- Pokud povolíte šíření záhlaví WS-Addressing, bude toto záhlaví šířeno do modulu za následujících okolností:
  - Když export přijme požadavky.
  - Když import přijme odezvy.
- Záhlaví WS-Addressing se nešíří do odchozích zpráv z produktu IBM Business Process Manager (tzn. záhlaví se nešíří při odeslání požadavků z importu ani při odeslání odezev z exportu).

Záhlaví WS-Security

Záhlaví WS-Security lze šířit prostřednictvím vazby webové služby (JAX-WS) a také prostřednictvím vazby webové služby (JAX-PRC).

Specifikace zabezpečení WS-Security webových služeb popisuje vylepšení systému zpráv SOAP, které poskytuje kvalitní ochranu prostřednictvím integrity a utajení zpráv, stejně jako prostřednictvím ověřování jednotlivých zpráv. Tyto mechanizmy lze použít k umístění širokého spektra modelů zabezpečení a šifrovacích technologií.

Při šíření záhlaví WS-Security počítejte s následujícími faktory:

- Pokud povolíte šíření záhlaví WS-Security, bude toto záhlaví šířeno napříč modulem za následujících okolností:

- Když export přijme požadavky.
- Když import odešle požadavky.
- Když import přijme odezvy.

- Při výchozím nastavení *nebude* záhlaví šířeno v případě odesílání odezev z exportu. Pokud nicméně nastavíte vlastnost prostředí JVM **WSSECURITY.ECHO.ENABLED** na hodnotu **true**, bude záhlaví šířeno i v případě odesílání odezev z exportu. V tomto případě, pokud nebylo záhlaví WS-Security na cestě požadavku změněno, mohou být záhlaví WS-Security automaticky promítnuty z požadavků do odezev.
- Přesná forma zprávy SOAP odeslané z importu pro požadavek nebo z exportu pro odezvu se nemusí přesně shodovat s původně přijatou zprávou SOAP. Z tohoto důvodu by se mělo předpokládat, že budou všechny digitální podpisy neplatné. Pokud odesílaná zpráva vyžaduje digitální podpis, musí být tento zřízen za použití příslušné sady zásad zabezpečení a záhlaví WS-Security související s digitálním podpisem v přijatých zprávách by měla být z modulu odebrána.

Pokud chcete šířit záhlaví WS-Security, musíte zahrnout schéma zabezpečení WS-Security do modulu aplikace. Procedura zahrnutí schématu viz “Zahrnutí schématu WS-Security do modulu aplikace” na stránce 71.

Způsob šíření záhlaví

Způsob šíření záhlaví závisí na nastavení zásad zabezpečení pro vazbu importu či exportu, viz Tabulka 20:

Tabulka 20. Způsob předávání záhlaví zabezpečení

	Vazba exportu bez zásad zabezpečení	**Vazba exportu se zásadami zabezpečení**
Vazba importu bez zásad zabezpečení	Záhlaví zabezpečení jsou v rámci modulu předávány tak, jak jsou. Nejsou šifrována. Záhlaví jsou odesílána ve stejné formě, v jaké byla obdržena. Digitální podpis se může stát neplatným.	Záhlaví zabezpečení jsou dešifrována a procházejí modulem s verifikací podpisu a ověřením. Dešifrovaná záhlaví jsou odesílána. Digitální podpis se může stát neplatným.
Vazba importu se zásadami zabezpečení	Záhlaví zabezpečení jsou v rámci modulu předávány tak, jak jsou. Nejsou šifrována. Záhlaví by neměla být šířena do importu. Jinak dojde k chybě z důvodu duplicity.	Záhlaví zabezpečení jsou dešifrována a procházejí modulem s verifikací podpisu a ověřením. Záhlaví by neměla být šířena do importu. Jinak dojde k chybě z důvodu duplicity.

Nakonfigurujte příslušnou sadu zásad pro vazby exportu a importu, protože tak izolujete klienta služby od změn konfigurace či požadavků na kvalitu služby poskytovatele služby. Pokud jsou v modulu standardní záhlaví viditelná, lze je potom použít k ovlivnění zpracování v rámci daného modulu (např. protokolování a trasování). Šíření záhlaví SOAP z přijatých zpráv napříč modulem vede ke snížení výhod izolace modulu.

Standardní záhlaví, jako např. záhlaví WS-Security by neměla být šířena při požadavku na import či odezvě na export v případě, že má import či export přidruženou sadu zásad, která by normálně vedla ke generování těchto záhlaví. Jinak se vyskytne chyba z důvodu duplicity záhlaví. Místo toho by měla být záhlaví výslovně odebrána, nebo by měly být vazby importu či exportu nakonfigurovány tak, aby zabraňovaly šíření záhlaví protokolu.

Přístup k záhlavím SOAP

Při obdržení zprávy obsahující záhlaví SOAP z importu či exportu webové služby jsou tato záhlaví umístěna do sekce se záhlavími objektu SMO (Service Message Object). Přístup k informacím záhlaví viz “Přístup k informacím záhlaví SOAP v objektu SMO”.

Zahrnutí schématu WS-Security do modulu aplikace

Následující procedura nastiňuje kroky pro zahrnutí daného schématu do modulu aplikace:

- Pokud má počítač, na kterém je spuštěný produkt Integration Designer, přístup k Internetu, postupujte takto:
  1. V perspektivě Business Integration vyberte volbu **Závislosti** pro svůj projekt.
  2. Rozbalte položku **Předdefinované prostředky** a vyberte buď položku **Soubory schématu WS-Security 1.0** nebo **Soubory schématu WS-Security 1.1**, a importujte tak dané schéma do svého modulu.
  3. Projekt vyčistěte a znovu sestavte.
- Pokud počítač, na kterém je spuštěný produkt Integration Designer nemá přístup k Internetu, můžete schéma stáhnout do druhého počítače, který přístup k Internetu má. Potom může být zkopírováno do počítače, na kterém je spuštěný produkt Integration Designer.
  1. Na počítači, který má přístup k Internetu, stáhněte vzdálené schéma:
     a. Klepněte na volbu **Soubor > Import > Obchodní integrace > WSDL a XSD**.
     b. Vyberte volbu **Vzdálený soubor WSDL** nebo **Soubor XSD**.
     c. Importujte následující schémata:

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

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

        http://www.w3.org/TR/xmldsig-core/xmldsig-core-schema.xsd
  2. Zkopírujte tato schémata do počítače, který nemá přístup k Internetu.
  3. Na počítači, který nemá přístup k Internetu, toto schéma importujte:
     a. Klepněte na volbu **Soubor > Import > Obchodní integrace > WSDL a XSD**.
     b. Vyberte volbu **Lokální soubor WSDL** nebo **Soubor XSD**.
  4. Změňte umístění schématu pro soubor oasis-wss-wssecurity_secext-1.1.xsd:
     a. Schéma otevřete na cestě *umístění_pracovní_blasti*/*název_modulu*/StandardImportFilesGen/oasis-wss-wssecurity-secext-1.1.xsd.
     b. Změňte:

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

        na:

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

     c. Změňte:

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

        na:

        ```
        <xs:import namespace='http://www.w3.org/2001/04/xmlenc#'
        schemaLocation='../w3/tr/_2002/rec_xmlenc_core_20021210/xenc-schema.xsd'/>
        ```
  5. Změňte umístění schématu pro soubor oasis-200401-wss-wssecurity-secext-1.0.xsd:
     a. Schéma otevřete na cestě *umístění_pracovní_blasti*/*název_modulu*/StandardImportFilesGen/oasis-200401-wss-wssecurity-secext-1.0.xsd.
     b. Změňte:

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

        na:

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

Šíření záhlaví přenosu:

Při obsluze zpráv SOAP může být zapotřebí přistupovat k informacím z určitých záhlaví transportu přijímaných zpráv. Zajistěte, aby byly zprávy se záhlavími transportu odesílány se specifickými hodnotami nebo umožněte záhlavím transportu minout určitý modul.

Při konfiguraci vazby webové služby v produktu Integration Designer můžete označit, že chcete záhlaví transportu šířit.

- Při obdržení požadavků při exportu nebo odezev při importu lze přistupovat k informacím ze záhlaví transportu, což umožňuje založit logiku modulu na hodnotách daného záhlaví a umožňuje to také úpravu těchto záhlaví.
- Když jsou odezvy odeslány z exportu nebo požadavky z importu, lze do těchto zpráv zahrnout záhlaví transportu.

Určení šíření záhlaví

Chcete-li nakonfigurovat šíření záhlaví transportu pro import a export, postupujte takto:

1. V pohledu Vlastnosti produktu Integration Designer vyberte položku **Vazba > Šíření**.
2. Nastavte požadovanou volbu šíření záhlaví transportu.

Poznámka: Šíření záhlaví transportu je standardně zakázáno a lze je implementovat pouze do běhového prostředí verze 7.0.0.3 (nebo novějšího). Poznámka: Ve verzi 7.0.0.3 je také šíření záhlaví transportu omezeno pouze na záhlaví transportu HTTP.

Pokud šíření záhlaví transportu povolíte, budou záhlaví z přijatých zpráv šířena napříč modulem a pokud je výslovně neodeberete, budou tato záhlaví použita v následujících vyvoláních v rámci stejného podprocesu.

Poznámka: Záhlaví transportu nelze šířit, pokud používáte vazbu webové služby (JAX-PRC).

Přístup k informacím ze záhlaví

Když je povoleno šíření záhlaví transportu příchozích zpráv, jsou všechna záhlaví transportu (včetně uživatelem definovaných záhlaví) viditelná v objektu SMO. Záhlaví můžete nastavit na jiné hodnoty, a vytvořit tak záhlaví nová. Poznámka: Neprobíhá žádná kontrola či ověření nastavených hodnot a jakákoliv nevhodná či chybná záhlaví mohou způsobit problémy s během webové služby.

Zvažte následující informace týkající se nastavení záhlaví HTTP:

- Žádné změny v záhlavích vyhrazené pro stroj webových služeb nebudou uplatněny v odchozí zprávě. Pro stroj webových služeb jsou například vyhrazena záhlaví verze či metoda HTTP, Content-Type, Content-Length a SOAPAction.
- Pokud je hodnota záhlaví číselná, mělo by být přímo nastaveno toto číslo (spíše než daný řetězec). Použijte například **Max-Forwards = 5** (spíše než **Max-Forwards = Max-Forwards: 5**) a **Age = 300** (spíše než **Age = Age: 300**).
- Pokud je velikost zprávy požadavku menší než 32 KB, stroj webových služeb odebere záhlaví Transfer-Encoding a místo toho nastaví záhlaví Content-Length na pevnou velikost dané zprávy.
- Záhlaví Content-Language resetuje WAS.channel.http na cestě odezvy.
- Neplatné nastavení upgradu vede k chybě 500.
- Následující záhlaví připojí hodnotu vyhrazenou strojem webových služeb k nastavení zákazníka:
  - User-Agent.
  - Cache-Control.
  - Pragma.
  - Accept.
  - Connection.

K informacím ze záhlaví získáte přístup jedním z následujících způsobů:

- Použitím mediačního primitiva pro přístup ke strukturám objektu SMO.
  Informace o použití mediačních primitiv viz odkazy “Související informace”.

- Použití rozhraní poskytovatele služeb (SPI) služby kontextu.

  Následující ukázkový kód přečte záhlaví transportu HTTP ze služby kontextu:

```
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();
     }
}
```

Odstraňování problémů

Pokud při odesílání přezkoumaných záhlaví narazíte na problémy, můžete zprávu TCP/IP zachytit pomocí nástrojů, jako je např. TCP/IP Monitor z produktu Integration Designer. K nástroji TCP/IP Monitor získáte přístup prostřednictvím volby **Spustit/Ladit > TCP/IP Monitor** na stránce Předvolby.

Hodnoty záhlaví si můžete také nechat zobrazit prostřednictvím trasování stroje JAX-WS: **org.apache.axis2.*=all: com.ibm.ws.websvcs.*=all:**

Práce s vazbami webové služby (JAX-WS):

Když ve svých aplikacích používáte vazby webové služby (JAX-WS), můžete dané vazbě přidat kvalitu služby SAML (Security Assertion Markup Language). Nejprve musíte prostřednictvím administrativní konzoly importovat sadu zásad. Prostřednictvím administrativní konzoly se můžete také ujistit, že je server řádně nakonfigurovaný pro použití s vazbou webové služby (JAX-WS).

Importování sad zásad SAML:

Jazyk SAML (Security Assertion Markup Language) je standard OASIS založený na jazyku XML pro výměnu informací o identitě uživatele a atributech zabezpečení. Když konfigurujete vazbu webové služby (JAX-WS) v produktu Integration Designer, můžete určit sadu zásad SAML. Nejprve použijete administrativní konzolu produktu IBM Business Process Manager ke zpřístupnění sad zásad, aby je bylo možné importovat do produktu Integration Designer.

Sady zásad SAML se zpravidla nacházejí v adresáři konfigurace profilu:

kořenový_adresář_profilu/config/templates/PolicySets

Před zahájením této procedury ověřte, že se v adresáři konfigurace profilu nacházejí následující adresáře (které obsahují sady zásad):

- 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

Pokud tyto adresáře v adresáři konfigurace profilu nejsou, zkopírujte je do tohoto adresáře z následujícího umístění:

kořenový_adresář_apl_serveru/profileTemplates/default/documents/config/templates/PolicySets

Sady zásad naimportujete do administrativní konzoly, vyberete ty, které chcete zpřístupnit produktu Integration Designer, a poté uložíte soubor .zip pro každou z těchto sad zásad do umístění, které je přístupné produktu Integration Designer.

1. Naimportujte sady zásad podle následujících kroků:
   a. V administrativní konzole klepněte na volbu **Služby > Sady zásad > Sady zásad aplikace**.
   b. Klepněte na volbu **Importovat > Z výchozího úložiště**.
   c. Vyberte výchozí sady zásad SAML a klepněte na tlačítko **OK**.
2. Exportujte sady zásad, aby je mohl použít produkt Integration Designer:
   a. Na stránce Sady zásad aplikace vyberte sadu zásad SAML, kterou chcete exportovat, a klepněte na volbu **Exportovat**.

      Poznámka: Pokud není stránka Sady zásad aplikace aktuálně zobrazena, klepněte v administrativní konzole na volbu **Služby > Sady zásad > Sady zásad aplikace**.
   b. Na další stránce klepněte na odkaz na soubor .zip pro sadu zásad.
   c. V okně Stažení souboru klepněte na tlačítko **Uložit** a označte umístění, které je přístupné produktu Integration Designer.
   d. Klepněte na tlačítko **Zpět**.
   e. Kroky 2a až 2d proveďte pro všechny sady zásad, které chcete exportovat.

Sady zásad SAML jsou uloženy do souborů .zip a jsou připraveny k importu do produktu Integration Designer.

Naimportujte sady zásad do produktu Integration Designer, jak je popsáno v tématu “Sady zásad”.

Vyvolání webových služeb, které vyžadují základní ověření HTTP:

Základní ověření HTTP ověřuje klienta služby v zabezpečeném koncovém bodu s použitím jména uživatele a hesla. Základní ověření HTTP můžete nastavit při odesílání nebo přijímání požadavků webové služby.

Základní ověření HTTP pro příjem požadavků webové služby nastavíte nakonfigurováním vazby exportu rozhraní Java API pro webové služby XML (JAX-WS), jak je popsáno v tématu Vytvoření a přiřazování rolí zabezpečení exportům webových služeb.

Základní ověření HTTP pro požadavky webové služby odeslané vazbou importu JAX-WS lze povolit jedním ze dvou způsobů:

- Při konfiguraci vazby importu v modulu SCA můžete vybrat dodanou sadu zásad ověření HTTP s názvem BPMHTTPBasicAuthentication (která je poskytována s vazbou importu webové služby (JAX-WS)) nebo kteroukoli jinou sadu zásad, která obsahuje zásadu HTTPTransport.
- Při konstruování modulu SCA můžete s využitím možností mediačního toku dynamicky vytvořit nové záhlaví ověřování HTTP a určit jméno uživatele a heslo v záhlaví.

Poznámka: Sada zásad má přednost před hodnotou určenou v záhlaví. Chcete-li použít hodnotu nastavenou v záhlaví ověření HTTP za běhu, nepřipojujte sadu, která obsahuje zásadu HTTPTransport. Konkrétně nepoužívejte výchozí sadu zásad BPMHTTPBasicAuthentication, a pokud jste definovali sadu zásad, zkontrolujte, zda vylučuje zásadu HTTPTransport.

Další informace o sadách zásad webové služby a vazbách zásad a o jejich použití viz Sady zásad webových služeb Informačního centra aplikačního serveru WAS.

- Chcete-li použít dodávanou sadu zásad, postupujte takto:
  1. Volitelné: V administrativní konzole vytvořte obecnou vazbu zásad klienta nebo upravte stávající vazbu zásad, která obsahuje zásadu HTTPTransport s požadovanými hodnotami ID uživatele a hesla.
  2. V produktu IBM Integration Designer vygenerujte vazbu importu webové služby (JAX-WS) a připojte k ní sadu zásad BPMHTTPBasicAuthentication.
  3. Proveďte *jeden* z následujících kroků:
     - V produktu IBM Integration Designer ve vlastnostech vazby importu webové služby (JAX-WS) zadejte název existující obecné vazby zásad klienta, která obsahuje zásadu HTTPTransport.
     - Po implementaci modulu SCA použijte administrativní konzolu k výběru stávající vazby zásady klienta, nebo k vytvořené nové vazby zásady klienta a poté ji přidružte k vazbě importu.
  4. Volitelné: V administrativní konzole komponenty Process Server upravte vybranou vazbu sady zásad tak, aby určovala požadované ID a heslo.
- Chcete-li určit jméno uživatele a heslo v záhlaví ověření HTTP, proveďte jednu z následujících sad kroků:
  - Pomocí mediačního primitiva Modul nastavení záhlaví HTTP v produktu IBM Integration Designer vytvořte záhlaví ověření HTTP a určete jméno uživatele a heslo.
  - Je-li vyžadována další logika, pak pomocí kódu Java v primitivu Vlastní mediace (viz následující příklad):
    1. Vytvořte záhlaví ověření HTTP.
    2. Určete jméno uživatele a heslo.
    3. Přidejte nové záhlaví ověření HTTP do zásady HTTPControl.
    4. Nastavte aktualizovanou zásadu HTTPControl zpět do kontextové služby.

```
//Získejte HeaderInfoType z contextService
ContextService contextService = (ContextService) ServiceManager.INSTANCE
.locateService("com/ibm/bpm/context/ContextService");
HeaderInfoType headers = contextService.getHeaderInfo();
if(headers == null){
    headers = ContextObjectFactory.eINSTANCE.createHeaderInfoType();
}
//Získejte záhlaví HTTP a řízení HTTP z HeaderInfoType
HTTPHeaderType httpHeaderType = headers.getHTTPHeader();
HTTPControl cp = httpHeaderType.getControl();
HeadersFactory factory = HeadersFactory.eINSTANCE;
if(cp == null){
    cp = factory.createHTTPControl();
}
//Vytvořte nové ověření HTTPAuthentication a nastavte 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);
// Nastavte informace záhlaví zpět do aktuálního kontextu provedení.
contextService.setHeaderInfo(headers);
```

Kontrola konfigurace serveru:

Při implementaci aplikace s vazbou webové služby (JAX-WS) je třeba zkontrolovat, zda server, na který se aplikace implementuje, nemá vybranou volbu **Spouštět komponenty podle potřeby**.

Výběr této volby můžete zkontrolovat provedením následujících kroků z administrativní konzoly:

1. Klepněte na volbu **Servery > Typy serverů > Aplikační servery WebSphere Application Server**.
2. Klepněte na název serveru.

3. Na kartě Konfigurace určete, zda je vybrána volba **Spouštět komponenty podle potřeby**.
4. Proveďte jeden z následujících kroků:
   - Pokud je volba **Spouštět komponenty podle potřeby** vybrána, odeberte její označení a potom klepněte na tlačítko **Použít**.
   - Pokud volba **Spouštět komponenty podle potřeby** vybrána není, klepněte na tlačítko **Storno**.

Přílohy ve zprávách SOAP:

Můžete odesílat a přijímat zprávy SOAP obsahující binární data (např. soubory PDF a obrázky JPEG) jako přílohu. Přílohy mohou být *odkazované* (tzn. reprezentované přímo částí zprávy v rozhraní dané služby) nebo *neodkazované* (u kterých lze zahrnout libovolné množství a typy příloh).

Odkazovaná příloha může mít jednu z následujících forem:

- Přílohy MTOM používají kódování určené mechanizmem MTOM (Message Transmission Optimization Mechanism) (http://www.w3.org/TR/soap12-mtom/) SOAP. Přílohy MTOM jsou povoleny prostřednictvím volby konfigurace ve vazbách importu a exportu a představují doporučený způsob kódování příloh pro nové aplikace.
- Prvek wsi:swaRef-typed ve schématu zprávy

  Přílohy definované pomocí typu wsi:swaRef jsou v souladu s profilem *Attachments Profile Version 1.0* organizace WS-I (Web Services Interoperability Organization)(http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html), který definuje, jakým způsobem prvky zprávy souvisí k částmi formátu MIME.
- Část zprávy nejvyšší úrovně používající typ binárního schématu

  Přílohy ve formě částí zpráv nejvyšší úrovně jsou v souladu se specifikací *Zprávy SOAP s přílohami* (http://www.w3.org/TR/SOAP-attachments).

  Přílohy ve formě zpráv nejvyšší úrovně mohou být také nakonfigurovány tak, aby zajistili soulad dokumentů a zpráv WSDL produkovaných vazbou s profily *Attachments Profile Version 1.0* a *Basic Profile Version 1.1* organizace WS-I (http://www.ws-i.org/Profiles/BasicProfile-1.1.html).

Neodkazovanou přílohu nese zpráva SOAP, aniž by tuto přílohu ve schématu dané zprávy cokoliv představovalo.

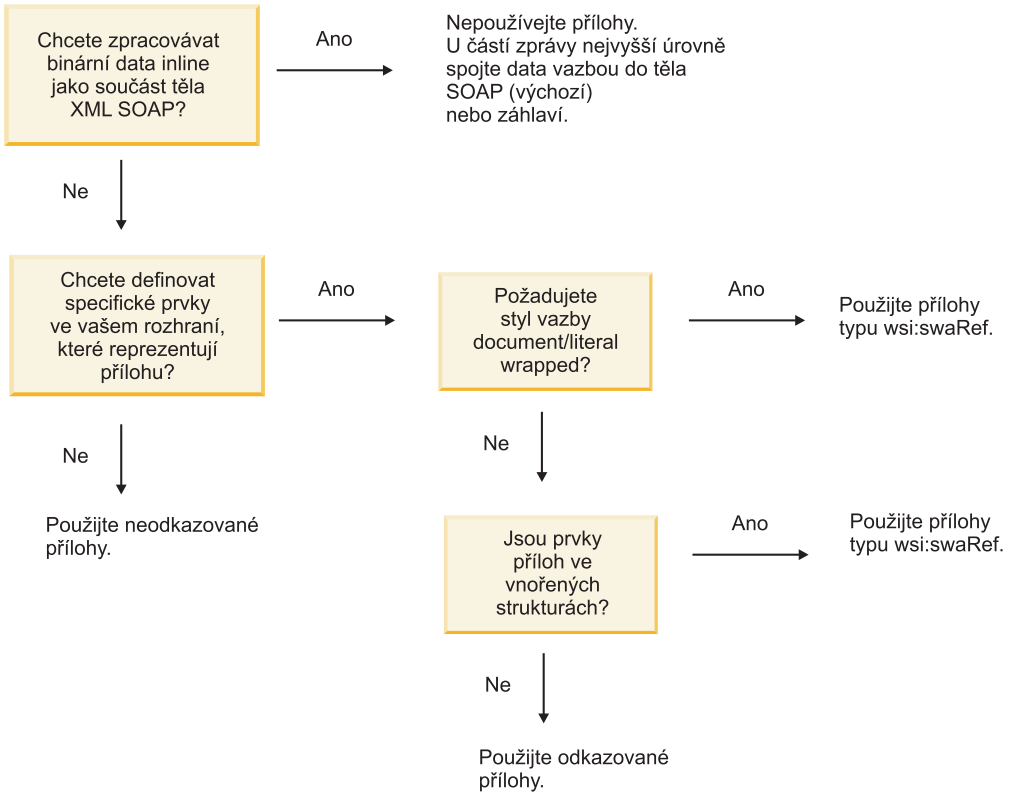
Ve všech případech s výjimkou příloh MTOM musí vazba WSDL SOAP obsahovat vazbu MIME pro přílohy, které se mají použít , a maximální velikost příloh nesmí překročit 20 MB.

Poznámka: Chcete-li odesílat nebo přijímat zprávy SOAP s přílohami, musíte použít jednu z vazeb webové služby založenou na rozhraní JAX-WS (Java API for XML Web Services).

Jak vybrat příslušný styl přílohy:

Když navrhujete nové rozhraní služby, která obsahuje binární data, zvažte, jak jsou tato data ve zprávách SOAP odesílaných a přijímaných touto službou přenášena.

Pokud připojená aplikace webových služeb podporuje mechanizmus MTOM (Message Transmission Optimization Mechanism), měl by se pro přílohy používat. Pokud ne, ukazuje následující diagram, jak vybírat jiné přílohy. Pomocí následující sady otázek určete vhodný styl přílohy:



Přílohy mechanizmu MTOM: části zpráv nejvyšší úrovně:

Můžete odesílat a přijímat zprávy webové služby obsahující v rámci zprávy SOAP přílohy mechanizmu MTOM (Message Transmission Optimization Mechanism). Ve zprávě SOAP formátu MIME s více částmi je tělo zprávy SOAP první částí zprávy a příloha nebo přílohy jsou v navazujících částech.

Když v rámci zprávy SOAP odesíláte nebo přijímáte odkazovanou přílohu, jsou binární data, ze kterých daná příloha sestává (která jsou často rozsáhlá), zadržována odděleně od těla zprávy SOAP, takže nemusí být analyzována jako XML. Výsledkem je efektivnější zpracování, než kdyby byla binární data zadržována v rámci prvku XML.

Následuje ukázka zprávy SOAP využívající mechanizmus MTOM:

```
... other transport headers ... Content-Type: multipart/related;
boundary=MIMEBoundaryurn_uuid_0FE43E4D025F0BF3DC11582467646812;
type="application/xop+xml"; start="
<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:0FE43E4D025F0BF3DC11582467646811@apache.org"/></imageData>
            </input>
         </sendImage>
      </soapenv:Body>
```

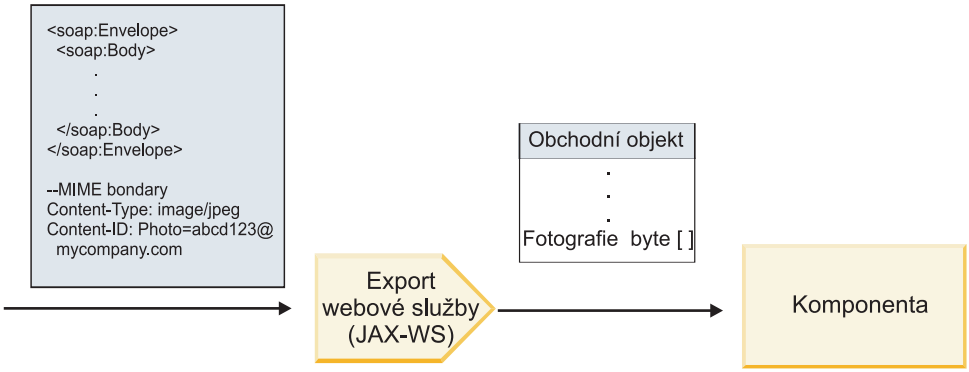
```
   </soapenv:Envelope>
--MIMEBoundaryurn_uuid_0FE43E4D025F0BF3DC11582467646812
content-type: text/plain
content-transfer-encoding: binary
content-id:
         <1.urn:uuid:0FE43E4D025F0BF3DC115824676468l1@apache.org>

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

Poznámka: V této ukázce mechanizmu MTOM má atribut content-type obálky SOAP hodnotu **application/xop+xml** a binární data nahrazuje prvek **xop:Include**, viz níže:

```
<xop:Include xmlns:xop="http://www.w3.org/2004/08/xop/include"
   href="cid:1.urn:uuid:0FE43E4D025F0BF3DC115824676468l1@apache.org"/>
```

Příchozí zpracování odkazovaných příloh

Když klient předává zprávu SOAP s přílohou komponentě SCA (Service Component Architecture), webová služba (JAX-WS) nejprve tuto přílohu odebere. Dále provede analýzu části SOAP dané zprávy a vytvoří obchodní objekt. Nakonec vazba nastaví binární soubor přílohy v tomto obchodním objektu.



Obrázek 13. Jak vazba exportu webové služby (JAX-WS) zpracovává zprávu SOAP obsahující odkazovanou přílohu

Atributy přílohy mechanizmu MTOM

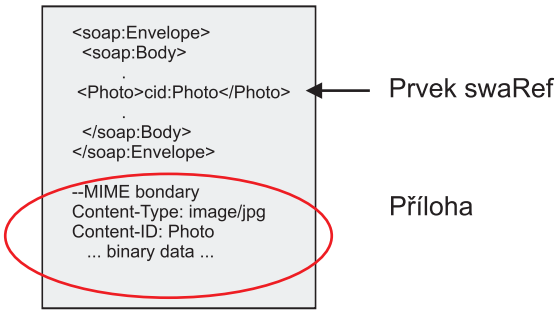
- Mechanizmus MTOM podporuje prvky přílohy s vnořenými strukturami.
- Mechanizmus MTOM je k dispozici pouze pro typ base64Binary.
- Mechanizmus MTOM podporuje prvky přílohy s vnořenými strukturami, což znamená, že je v případě příloh mechanizmu MTOM atribut **bodyPath** roven umístění **xpath** prvku, který danou přílohu mechanizmu MTOM zadržuje. Logika výpočtu prvku **bodyPath** striktně následuje schéma generování umístění **xpath**, jak je uvedeno v příkladech níže:
  - Pro typ bez pole (**maxOccurs** je rovno 1): /sendImage/input/imageData.
  - Pro typ pole (**maxOccurs** > 1): /sendImage/input/imageData[1].
- Nejsou podporovány smíšené typy příloh. Pokud je tedy ve vazbě importu povolen mechanizmus MTOM, bude generována příloha MTOM. Pokud je mechanizmus MTOM zakázaný nebo byla ponechána výchozí hodnota konfigurace mechanizmu MTOM, příchozí zprávy s mechanizmem MTOM nejsou podporovány.

Odkazované přílohy: prvky typu swaRef:

Můžete odesílat a přijímat zprávy SOAP obsahující přílohy reprezentované v rozhraní služby jako prvky typu swaRef.

Prvek typu swaRef je definován v profilu *Attachments Profile* verze 1.0 interoperability WS-I (Web Services Interoperability Organization) (http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html), který definuje, jakým způsobem se prvky zprávy vztahují k částem formátu MIME.

Ve zprávě SOAP obsahuje tělo SOAP prvek typu swaRef, který identifikuje ID obsahu dané přílohy.



Obrázek 14. Zpráva SOAP s prvkem swaRef

Kód jazyka WSDL této zprávy SOAP obsahuje v rámci části zprávy identifikující danou přílohu prvek typu swaRef.

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

Kód jazyka WSDL by měl také obsahovat vazbu formátu MIME označující, že musí být použity zprávy formátu MIME o více částech.

Poznámka: Kód jazyka WSDL *nezahrnuje* vazbu formátu MIME pro specifický prvek zprávy typu swaRef, protože vazby formátu MIME platí pouze pro části zpráv nejvyšší úrovně.

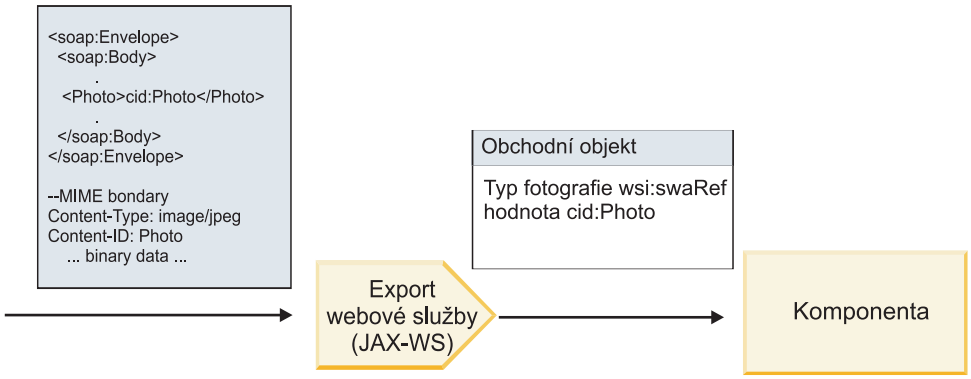
Přílohy reprezentované prvky typu swaRef lze šířit pouze mezi komponentami mediačního toku. Pokud musí mít k příloze přístup jiný typ komponenty, nebo pokud ji musí jiný typ komponenty šířit, přesuňte přílohu pomocí komponenty mediačního toku do umístění, které je pro danou komponentu přístupné.

Příchozí zpracování příloh

Ke konfiguraci přijetí přílohy prostřednictvím vazby exportu se používá produkt Integration Designer. Vytvoříte modul, stejně jako jeho přidružené rozhraní a operace, včetně prvku typu swaRef. Potom vytvoříte vazbu webové služby (JAX-WS).

Poznámka: Další podrobné informace viz téma “Práce s přílohami” v Informačním centru produktu Integration Designer.

Když klient předává zprávu SOAP s přílohou typu swaRef komponentě SCA (Service Component Architecture), vazba exportu webové služby (JAX-WS) nejprve tuto přílohu odebere. Dále provede analýzu části SOAP dané zprávy a vytvoří obchodní objekt. Nakonec vazba nastaví ID obsahu přílohy v tomto obchodním objektu.



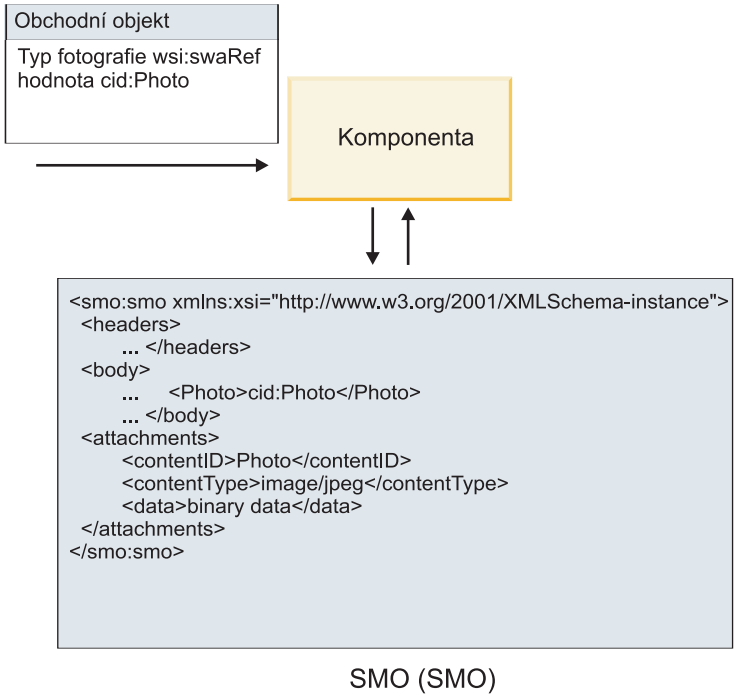
Obrázek 15. Jak vazba exportu webové služby (JAX-WS) zpracovává zprávu SOAP s přílohou typu swaRef

Přístup k metadatům přílohy v komponentě mediačního toku

Jak ukazuje Obrázek 16, když k přílohám typu swaRef přistupují komponenty, zobrazuje se identifikátor obsahu přílohy jako prvek typu swaRef.

Každá příloha zprávy SOAP obsahuje také v objektu SMO odpovídající prvek **attachments**. Když používáte typ swaRef interoperability WS-I, obsahuje prvek **attachments** typ obsahu přílohy a ID obsahu, stejně jako skutečná binární data dané přílohy.

Za účelem získání hodnoty přílohy typu swaRef je proto nutné získat hodnotu prvku typu swaRef a následně vyhledat prvek **attachments** s odpovídající hodnotou atributu **contentID**. Poznámka: Z hodnoty atributu **contentID** je zpravidla odebrána předpona **cid:** hodnoty swaRef.



Obrázek 16. Způsob zobrazování příloh typu swaRef v objektu SMO

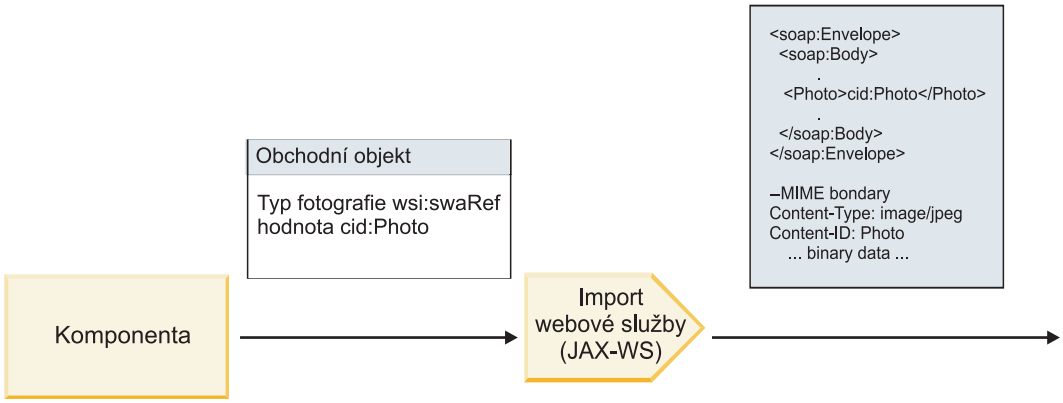
Odchozí zpracování

Pomocí produktu Integration Designer konfigurujete vazbu importu webové služby (JAX-WS) pro účely vyvolání externí webové služby. Vazba importu se konfiguruje pomocí dokumentu WSDL, popisujícího vyvolávanou webovou službu a definujícího, které přílohy budou předány webové službě.

Když vazba importu webové služby (JAX-WS) obdrží zprávu SCA, dojde k odeslání prvků typu swaRef v podobě příloh, pokud je import propojený s komponentou mediačního toku a pokud má prvek typu swaRef odpovídající prvek **attachments**.

V případě odchozího zpracování jsou prvky typu swaRef vždy odesílány se svými hodnotami ID obsahu. Mediační modul však musí vždy zajistit přítomnost příslušného prvku **attachments** s odpovídající hodnotou atributu **contentID**.

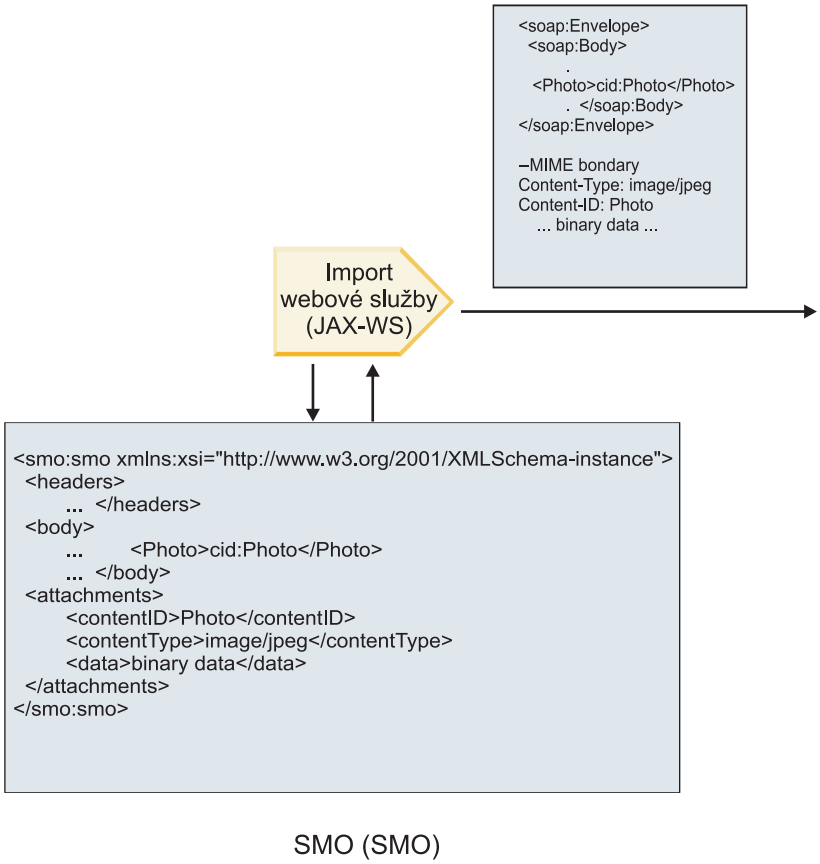
Poznámka: V zájmu zajištění kompatibility s profilem WS-I Attachments Profile by měla hodnota atributu **content ID** odpovídat "kódování části content-id", jak je popsáno v sekci 3.8 profilu interoperability WS-I *Attachments Profile* verze 1.0.



Obrázek 17. Jak vazba importu webové služby (JAX-WS) generuje zprávu SOAP s přílohou typu swaRef

Nastavení metadat přílohy v komponentě mediačního toku

Pokud objekt SMO obsahuje hodnotu prvku typu swaRef a také prvek **attachments**, vazba připraví zprávu SOAP (s přílohou) a odešle ji příjemci.



Obrázek 18. Způsob přístupu k příloze typu swaRef z modulu SMO za účelem vytvoření zprávy SOAP

Prvek **attachments** je v modulu SMO přítomen pouze, pokud je komponenta mediačního toku přímo propojená k importu či exportu. Nepředává se jiným typům komponent. Pokud jsou dané hodnoty potřebné v modulu obsahujícím jiné typy komponent, měly by se tyto hodnoty zkopírovat pomocí komponenty mediačního toku do umístění, ke kterému lze v daném modulu přistupovat. Další komponenta mediačního toku by potom měla být použita k nastavení správných hodnot ještě před odchozím vyvoláním, a to prostřednictvím importu webové služby.

Důležité: Jak je popsáno v tématu “Reprezentace XML objektu SMO,” mediační primitivum Mapování transformuje zprávy pomocí transformace XSLT 1.0. Transformace provádí serializaci XML objektu SMO. Mediační primitivum Mapování umožňuje určit kořen serializace a kořenový prvek dokumentu XML tento kořen odráží.

Když odesíláte zprávy SOAP s přílohami, vámi vybraný kořenový prvek určuje, jak budou přílohy šířeny.

- Pokud vyberete jako kořen mapy XML volbu “/body”, budou všechny přílohy šířeny po této mapě standardně.
- Pokud vyberete jako kořen mapy “/”, můžete šíření příloh řídit.

Odkazované přílohy: části zprávy nejvyšší úrovně:

Můžete odesílat a přijímat zprávy SOAP obsahující binární přílohy deklarované jako části vašeho rozhraní služby.

Ve zprávě SOAP formátu MIME s více částmi je tělo zprávy SOAP první částí zprávy a příloha nebo přílohy jsou v navazujících částech.

Jaké výhody má odesílání či přijímání zpráv SOAP s odkazovanými přílohami? Binární data, ze kterých sestává daná příloha (která jsou často rozsáhlá), jsou zadržována odděleně od těla zprávy SOAP, takže nemusí být analyzována jako XML. Výsledkem je efektivnější zpracování, než kdyby byla binární data zadržována v rámci prvku XML.

Typy zpráv SOAP s odkazovanými přílohami

Počínaje verzí 7.0.0.3 produktu IBM Business Process Manager si můžete volit způsob generování zprávy SOAP:

- **Zprávy kompatibilní s interoperabilitou webových služeb (WS-I)**

  Běhové prostředí může generovat zprávy SOAP, které jsou kompatibilní s profilem WS-I *Attachments Profile verze 1.0* a WS-I *Basic Profile verze 1.1*. V případě zprávy SOAP kompatibilní s těmito profily je pouze jedna část zprávy vázaná na tělo zprávy SOAP. U částí, které jsou vázané jako přílohy, se kódování části content-id (jak je popsáno v profilu WS-I *Attachments Profile verze 1.0*) používá k vytvoření vztahu mezi přílohou a danou částí zprávy.
- **Zprávy, které nejsou kompatibilní s interoperabilitou WS-I**

  Běhové prostředí může generovat zprávy SOAP, které nejsou kompatibilní s profily WS-I, ale které jsou kompatibilní se zprávami generovanými ve verzi 7.0 nebo 7.0.0.2 produktu IBM Business Process Manager. Zprávy SOAP používají prvky nejvyšší úrovně pojmenované po části zprávy s atributem **href**, který zadržuje atribut **content-id** přílohy, ale kódování části content-id (jak je popsáno v profilu WS-I *Attachments Profile verze 1.0*) se nepoužívá.

Výběr kompatibility s interoperabilitou WS-I pro exporty webových služeb

Ke konfiguraci vazby exportu se používá produkt Integration Designer. Vytvoříte modul, stejně jako jeho přidružené rozhraní a operace. Potom vytvoříte vazbu webové služby (JAX-WS). Stránka Odkazované přílohy zobrazí všechny binární části vytvořené operace a vy vyberete, které části budou přílohy. Na stránce Určení kompatibility s WS-I AP 1.0 produktu Integration Designer potom určíte jednu z následujících voleb:

- **Použít zprávu SOAP kompatibilní s WS-I AP 1.0**.

  Pokud vyberete tuto volbu, určíte také, jaká část zprávy má být vázána na tělo zprávy SOAP.

  Poznámka: Tuto volbu lze použít pouze v případě, že je také odpovídající soubor WSDL kompatibilní s interoperabilitou WS-I.

  Soubor WSDL generovaný produktem Integration Designer verze 7.0.0.3 je kompatibilní s interoperabilitou WS-I. Pokud ovšem importujete soubor WSDL, který s interoperabilitou WS-I kompatibilní není, tuto volbu vybrat nemůžete.
- **Použít zprávu SOAP nekompatibilní s WS-I AP 1.0**

  Pokud vyberete tuto volbu, což je výchozí nastavení, je první část zprávy vázaná na tělo zprávy SOAP.

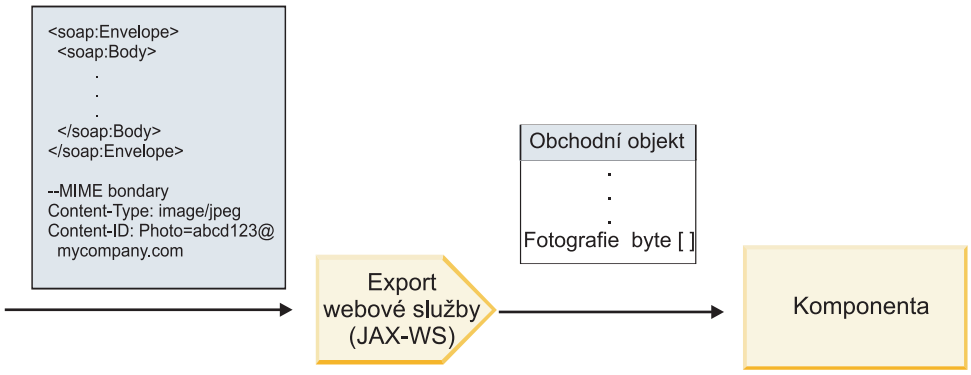
Poznámka: Jako odkazované přílohy lze odesílat či přijímat pouze části zpráv nejvyšší úrovně (tzn. prvky definované v rámci parametru portType jazyka WSDL jako součásti výstupní či vstupní zprávy), které obsahují binární typ (buď base64Binary, nebo hexBinary).
Další podrobné informace viz téma “Práce s přílohami” v Informačním centru produktu Integration Designer.

U zpráv kompatibilních s interkompatibilitou WS-I je identifikátor content-ID generovaný ve zprávě SOAP zřetězením následujících prvků:

- Hodnota atributu **name** prvku **wsdl:part** odkazovaného prvkem **mime:content**.
- Znak **=**.
- Globálně jedinečná hodnota, jako např. identifikátor UUID.
- Znak **@**.
- Platný název domény.

Příchozí zpracování odkazovaných příloh

Když klient předává zprávu SOAP s přílohou komponentě SCA (Service Component Architecture), webová služba (JAX-WS) nejprve tuto přílohu odebere. Dále provede analýzu části SOAP dané zprávy a vytvoří obchodní objekt. Nakonec vazba nastaví binární soubor přílohy v tomto obchodním objektu.

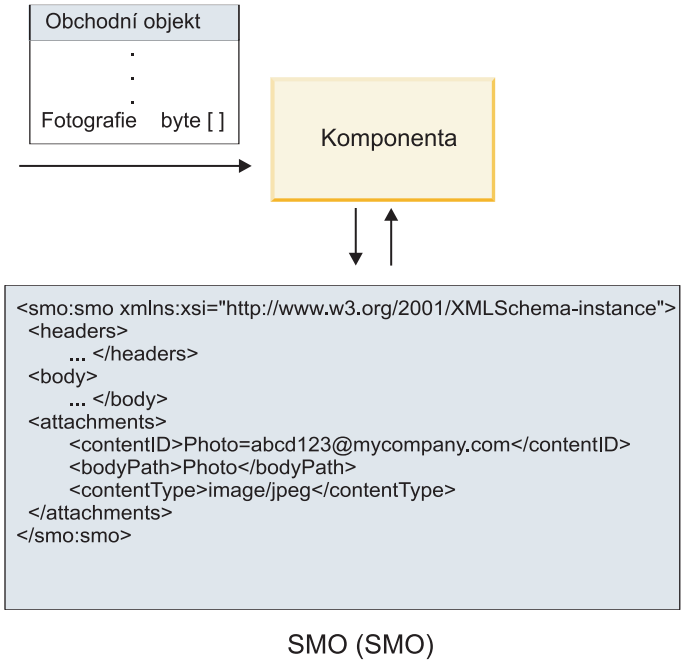


Obrázek 19. Jak vazba exportu webové služby (JAX-WS) zpracovává zprávu SOAP kompatibilní s interoperabilitou WS-I obsahující odkazovanou přílohu

Přístup k metadatům přílohy v komponentě mediačního toku

Jak ukazuje Obrázek 19, když k odkazovaným přílohám přistupují komponenty, zobrazují se data přílohy jako bajtové pole.

Každá odkazované příloha zprávy SOAP obsahuje také v objektu SMO odpovídající prvek **attachments**. Prvek **attachments** obsahuje typ obsahu přílohy a cestu k prvku těla zprávy, ve kterém je příloha zadržována.



Obrázek 20. Způsob zobrazování odkazovaných příloh v objektu SMO

Důležité: V případě transformace zprávy a přesunutí přílohy nedochází k automatické aktualizaci cesty k danému prvku těla zprávy. K aktualizaci prvku **attachments** pomocí nové cesty můžete použít mediační tok (například jako součást transformace nebo pomocí odděleného modulu nastavení prvků zpráv).

Způsob sestavení odchozích zpráv SOAP

Pomocí produktu Integration Designer konfigurujete vazbu importu webové služby (JAX-WS) pro účely vyvolání externí webové služby. Vazba importu se konfiguruje pomocí dokumentu WSDL, popisujícího vyvolávanou webovou

službu a definujícího, které části zpráv mají být předávány jako přílohy. Na stránce Určení kompatibility s WS-I AP 1.0 produktu Integration Designer můžete také označit jednu z následujících voleb:

- **Použít zprávu SOAP kompatibilní s WS-I AP 1.0**.

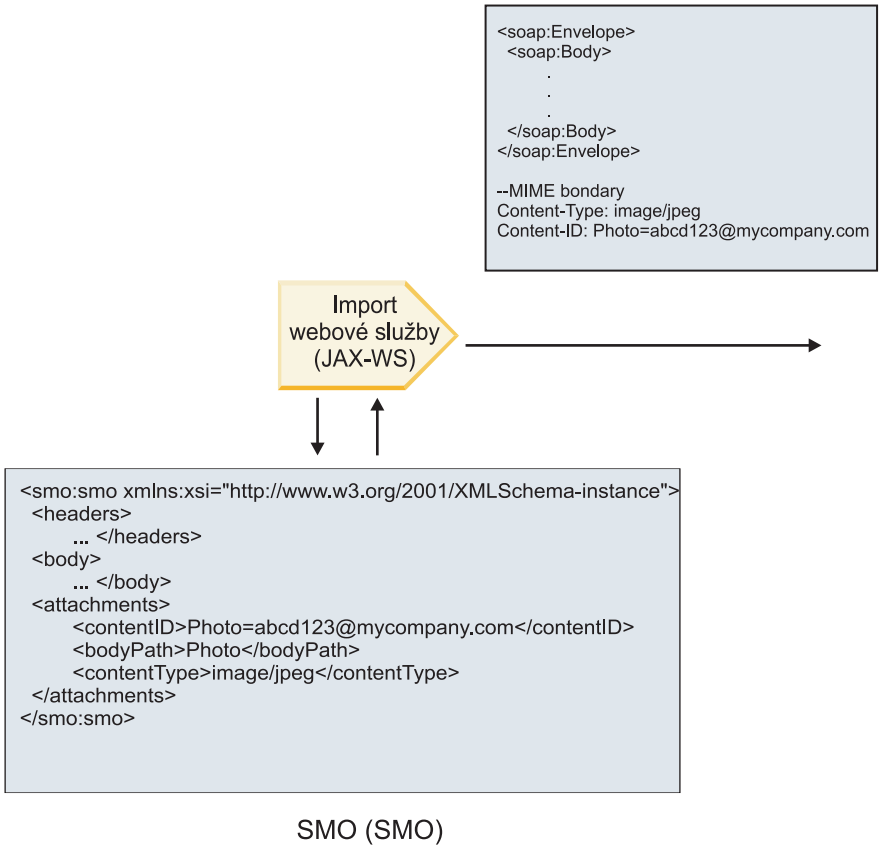
  Pokud vyberete tuto volbu, určíte také, jaká část zprávy má být vázána na tělo zprávy SOAP. Všechny ostatní části budou vázané na přílohy či záhlaví. Zprávy odeslané vazbou nezahrnují prvky těla zprávy SOAP odkazující na tyto přílohy. Tento vztah je vyjádřen prostřednictvím ID obsahu přílohy obsahujícího název dané části zprávy.
- **Použít zprávu SOAP nekompatibilní s WS-I AP 1.0**

  Pokud vyberete tuto volbu, což je výchozí nastavení, je první část zprávy vázaná na tělo zprávy SOAP. Všechny ostatní části jsou vázané na přílohy či záhlaví. Zprávy odeslané vazbou zahrnují minimálně jeden prvek těla zprávy SOAP, který odkazuje na dané přílohy prostřednictvím atributu **href**.

Poznámka: Část představující přílohu, definovaná v jazyce WSDL, musí být jednoduchého typu (buď base64Binary, nebo hexBinary). Pokud určitou část definuje atribut complexType, nelze ji navázat jako přílohu.

Odchozí zpracování odkazovaných příloh

Vazba importu používá informace z objektu SMO k určení způsobu odesílání binárních zpráv nejvyšší úrovně v podobě příloh.



Obrázek 21. Způsob přístupu k odkazované příloze z modulu SMO za účelem vytvoření zprávy SOAP

Prvek **attachments** je v modulu SMO přítomen pouze, pokud je komponenta mediačního toku přímo propojená k importu či exportu. Nepředává se jiným typům komponent. Pokud jsou dané hodnoty potřebné v modulu obsahujícím jiné typy komponent, měly by se tyto hodnoty zkopírovat pomocí komponenty mediačního toku do umístění, ke kterému lze v daném modulu přistupovat. Další komponenta mediačního toku by potom měla být použita k nastavení správných hodnot ještě před odchozím vyvoláním, a to prostřednictvím importu webové služby.

Vazba určuje způsob odeslání zprávy (či zda vůbec byla odeslána) pomocí kombinace následujících podmínek:

- Zda existuje vazba formátu MIME jazyka WSDL pro část binární zprávy nejvyšší úrovně a pokud ano, jak je definovaný typ obsahu.
- Zda v objektu SMO existuje prvek **attachments**, jehož hodnota atributu **bodyPath** odkazuje na binární část nejvyšší úrovně.

Způsob vytváření příloh v případě existence prvku attachment v objektu SMO

Následující tabulka zobrazuje způsob vytvoření a odeslání přílohy v případě, že objekt SMO obsahuje prvek **attachment** s atributem **bodyPath**, který odpovídá části s názvem zprávy:

Tabulka 21. Způsob generování přílohy

Stav vazby formátu MIME jazyka WSDL pro binární část zprávy nejvyšší úrovně	**Způsob vytvoření a odeslání zprávy**
Je přítomna a zároveň platí jedna z následujících možností: • Daná část zprávy nemá definovaný typ obsahu. • Je definováno více typů obsahu. • Je definován zástupný symbol typu obsahu.	Část zprávy je odeslána jako příloha. Parametr **Content-Id** je nastavený na hodnotu prvku attachments, pokud je přítomen, jinak je vygenerován. Parametr **Content-Type** je nastavený na hodnotu prvku attachments, pokud je přítomen, jinak je nastaven na hodnotu **application/octet-stream**.
Je přítomna s jediným obsahem dané části zprávy bez zástupného znaku	Část zprávy je odeslána jako příloha. Parametr **Content-Id** je nastavený na hodnotu prvku attachments, pokud je přítomen, jinak je vygenerován. Parametr **Content-Type** je nastavený na hodnotu prvku attachments, pokud je přítomen, jinak je nastaven na typ definovaný v prvku obsahu MIME jazyka WSDL.
Není přítomna	Část zprávy je odeslána jako příloha. Parametr **Content-Id** je nastavený na hodnotu prvku attachments, pokud je přítomen, jinak je vygenerován. Parametr **Content-Type** je nastavený na hodnotu prvku attachments, pokud je přítomen, jinak je nastaven na hodnotu **application/octet-stream**. **Poznámka:** Odesílání částí zpráv v podobě příloh, pokud nebyly jako přílohy definovány v jazyce WSDL, může vést k porušení kompatibility s profilem WS-I Attachments Profile verze 1.0 a pokud je to možné, měli byste se mu vyhnout.

Způsob vytváření příloh v případě, že v objektu SMO neexistuje prvek attachment

Následující tabulka zobrazuje způsob vytvoření a odeslání přílohy v případě, že objekt SMO neobsahuje prvek **attachment** s atributem **bodyPath**, který odpovídá části s názvem zprávy:

Tabulka 22. Způsob generování přílohy

Stav vazby formátu MIME jazyka WSDL pro binární část zprávy nejvyšší úrovně	**Způsob vytvoření a odeslání zprávy**
Je přítomna a zároveň platí jedna z následujících možností: • Daná část zprávy nemá definovaný typ obsahu. • Je definováno více typů obsahu. • Je definován zástupný symbol typu obsahu.	Část zprávy je odeslána jako příloha. Generuje se atribut **Content-Id**. Atribut **Content-Type** je nastaven na hodnotu **application/octet-stream**.

Tabulka 22. Způsob generování přílohy (pokračování)

Stav vazby formátu MIME jazyka WSDL pro binární část zprávy nejvyšší úrovně	**Způsob vytvoření a odeslání zprávy**
Je přítomna s jediným obsahem dané části zprávy bez zástupného znaku	Část zprávy je odeslána jako příloha. Generuje se atribut **Content-Id**. Atribut **Content-Type** je nastaven na typ definovaný v prvku obsahu MIME jazyka WSDL.
Není přítomna	Část zprávy není odeslána jako příloha.

Důležité: Jak je popsáno v tématu "Reprezentace XML objektu SMO," mediační primitivum Mapování transformuje zprávy pomocí transformace XSLT 1.0. Transformace provádí serializaci XML objektu SMO. Mediační primitivum Mapování umožňuje určit kořen serializace a kořenový prvek dokumentu XML tento kořen odráží.

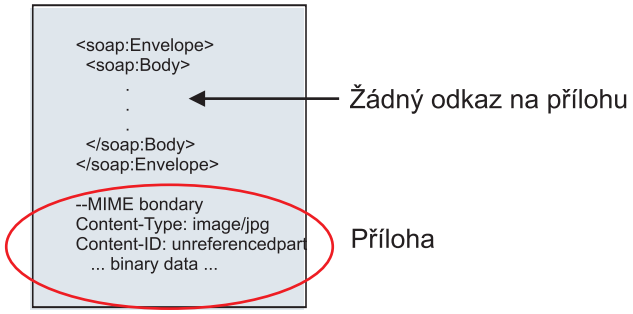
Když odesíláte zprávy SOAP s přílohami, vámi vybraný kořenový prvek určuje, jak budou přílohy šířeny.

- Pokud vyberete jako kořen mapy XML volbu "/body", budou všechny přílohy šířeny po této mapě standardně.
- Pokud vyberete jako kořen mapy "/", můžete šíření příloh řídit.

Neodkazované přílohy:

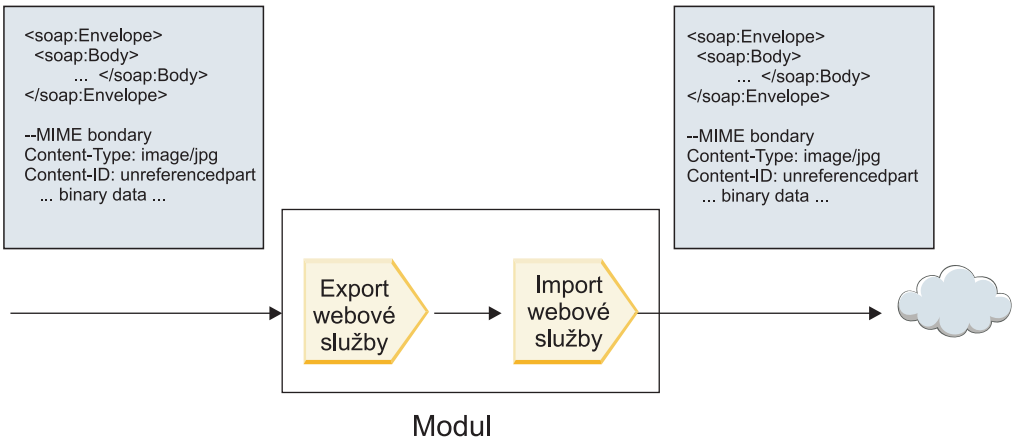
Můžete odesílat a přijímat *neodkazované* přílohy, které nejsou deklarované v rozhraní služby.

Ve zprávě SOAP formátu MIME s více částmi je tělo zprávy SOAP první částí zprávy a přílohy jsou v navazujících částech. Tělo zprávy SOAP neobsahuje žádný odkaz na přílohu.



Obrázek 22. Zpráva SOAP s neodkazovanou přílohou

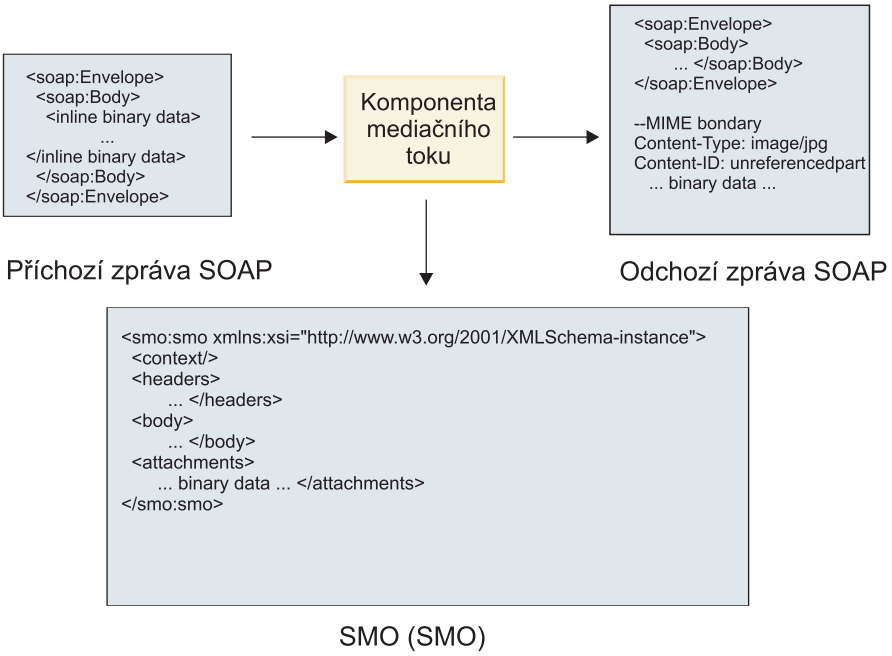
Zprávy SOAP s neodkazovanou přílohou můžete odesílat prostřednictvím exportu webové služby do importu webové služby. Danou přílohu obsahuje výstupní zpráva odeslaná do cílové webové služby.



Obrázek 23. Příloha procházející modulem SCA

Na obrázku Obrázek 23 prochází zpráva SOAP s přílohou beze změn.

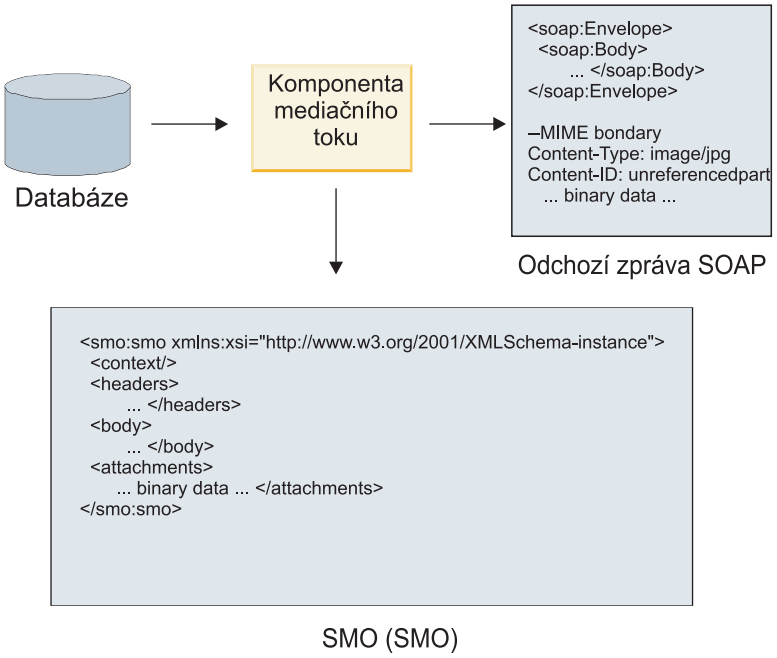
Zprávu SOAP můžete změnit také pomocí komponenty mediačního toku. Komponentu mediačního toku můžete například použít k extrakci dat ze zprávy SOAP (v tomto případě binární data v těle zprávy) a vytvoření zprávy SOAP s přílohami. Data jsou zpracována jako součást prvku přílohy objektu SMO (Service Message Object).



Obrázek 24. Zpráva zpracovaná komponentou mediačního toku

Obráceně může komponenta mediačního toku zase transformovat příchozí zprávu extrakcí a zakódováním přílohy a následně přenést zprávu bez příloh.

Místo extrakce dat z příchozí zprávy SOAP za účelem zformování zprávy SOAP s přílohami můžete získat data přílohy z externího zdroje, např. z databáze.



Obrázek 25. Příloha získaná z databáze a přidaná do zprávy SOAP

Obráceně může zase komponenta mediačního toku extrahovat přílohu z příchozí zprávy SOAP a zprávu zpracovat (např. uložit přílohu do databáze).

Neodkazované přílohy mohou být šířeny pouze napříč komponentami mediačního toku. Pokud musí mít k příloze přístup jiný typ komponenty, nebo pokud ji musí jiný typ komponenty šířit, přesuňte přílohu pomocí komponenty mediačního toku do umístění, které je pro danou komponentu přístupné.

Důležité: Jak je popsáno v tématu “Reprezentace XML objektu SMO,” mediační primitivum Mapování transformuje zprávy pomocí transformace XSLT 1.0. Transformace provádí serializaci XML objektu SMO. Mediační primitivum Mapování umožňuje určit kořen serializace a kořenový prvek dokumentu XML tento kořen odráží.

Když odesíláte zprávy SOAP s přílohami, vámi vybraný kořenový prvek určuje, jak budou přílohy šířeny.

- Pokud vyberete jako kořen mapy XML volbu “/body”, budou všechny přílohy šířeny po této mapě standardně.
- Pokud vyberete jako kořen mapy “/”, můžete šíření příloh řídit.

Použití vazby stylu dokumentu WSDL se zprávami o více částech:

Organizace WS-I (Web Services Interoperability Organization) definovala sadu pravidel týkajících se popisování webových služeb prostřednictvím jazyka WSDL a formování odpovídajících zpráv SOAP v zájmu zajištění interoperability.

Tato pravidla jsou určena v profilu interoperability WS-I nazvaném *Basic Profile verze 1.1* (http://www.ws-i.org/Profiles/BasicProfile-1.1.html). Konkrétně je v části R2712 profilu WS-I Basic Profile 1.1 uvedeno: “Vazba document/literal MUSÍ být serializovaná jako OBÁLKA s prvkem těla soap:Body, jehož podřízený prvek je instancí deklarace globálního prvku odkazovaného příslušnou částí zprávy wsdl:message.”

To znamená, že v případě použití vazby SOAP stylu dokumentu pro operaci se zprávami (vstup, výstup nebo porucha) s více definovanými částmi by měla být pouze jedna z těchto částí vázaná na tělo SOAP, aby byla zaručena kompatibilita s profilem WS-I Basic Profile 1.1.

Dále je v části R2941 profilu WS-I Attachments Profile 1.0 uvedeno: "Vazba wsdl:binding v popisu BY MĚLA vázat každou část wsdl:part zprávy wsdl:message v portu wsdl:portType, na který odkazuje, na jeden z atributů soapbind:body, soapbind:header, soapbind:fault , soapbind:headerfault nebo mime:content”.

To znamená, že v případě použití vazby SOAP stylu dokumentu pro operaci se zprávami (vstup, výstup nebo porucha), které mají definovaných více částí, musí být všechny části, s výjimkou části vázané na tělo SOAP, vázány jako přílohy nebo záhlaví.

Následující metoda se v tomto případě používá při generování popisů jazyka WSDL pro exporty s vazbami webové služby (JAX-WS a JAX-RPC):

- Pokud existuje více než jeden prvek, který není binárního typu, můžete si zvolit, jaká část zprávy bude vázaná na tělo zprávy SOAP. Pokud existuje jediný prvek jiného než binárního typu, je tento vázán na tělo zprávy SOAP automaticky.
- V případě vazby webové služby JAX-WS jsou všechny části zprávy jiné než "hexBinary" nebo "base64Binary" vázány jako odkazované přílohy. Viz téma “Odkazované přílohy: části zprávy nejvyšší úrovně” na stránce 82.
- Všechny ostatní části zprávy jsou vázány jako záhlaví SOAP.

Vazby importu JAX-RPC a JAX-WS ctí vazbu SOAP existujícího dokumentu WSDL se zprávami ve stylu dokumentu s více částmi, a to i v případě, že tento dokument neváže více částí k danému tělu SOAP. Nicméně však nelze pro takové dokumenty WSDL generovat v produktu Rational Application Developer klienty webových služeb.

Poznámka: Vazba JAX-RPC nepodporuje přílohy.

V případě použití zpráv o více částech s operací obsahující vazbu SOAP stylu dokumentu proto doporučujeme postupovat takto:

1. Použijte styl document/literal wrapped. V tomto případě mají zprávy vždy pouze jednu část. Nicméně přílohy musí být v tomto případě neodkazované (viz “Neodkazované přílohy” na stránce 87) nebo typu swaRef (viz “Odkazované přílohy: prvky typu swaRef” na stránce 78).
2. Použijte styl RPC/literal. V tomto případě neexistuje pro vazbu WSDL žádné omezení počtu částí vázaných na tělo SOAP. Výsledná zpráva SOAP obsahuje vždy jediný podřízený prvek představující vyvolávanou operaci, přičemž části dané zprávy jsou podřízené prvky tohoto prvku.
3. Pro vazby webové služby JAX-WS je potřeba mít nanejvýš jednu část zprávy, která není typu "hexBinary" nebo "base64Binary", pokud není přijatelné vázat zbývající nebinární části na záhlaví SOAP.
4. Všechny ostatní případy probíhají podle popsaného scénáře.

Poznámka: V případě použití zpráv SOAP, které nejsou kompatibilní s profilem WS-I *Basic Profile verze 1.1*, existují další omezení.

- První část zprávy by neměla být binární.
- Když přijímáte zprávy SOAP ve stylu dokumentu o více částech s odkazovanými přílohami, vazba webové služby JAX-WS očekává, že bude každá odkazovaná příloha reprezentována podřízeným prvkem těla SOAP s hodnotou atributu href identifikující danou přílohu podle jejího ID obsahu. Vazba webové služby JAX-WS odesílá odkazované přílohy pro takové zprávy stejným způsobem. Toto chování není kompatibilní s profilem WS-I Basic Profile.

  Chcete-li zajistit, aby byly vaše zprávy kompatibilní s profilem Basic Profile, postupujte podle metody 1 nebo 2 z předchozího seznamu, nebo se vyhněte používání odkazovaných příloh pro takové zprávy a místo toho používejte neodkazované přílohy či přílohy typu swaRef.

Vazby HTTP:

Vazba HTTP je navržena tak, aby zajišťovala konektivitu architektury SCA (Service Component Architecture) pro HTTP. V důsledku toho se mohou již existující nebo nově vyvinuté aplikace HTTP podílet na prostředích architektury SOA (Service Oriented Architecture).

Protokol HTTP (Hypertext Transfer Protocol) je široce používaný protokol pro přenos informací na webu. Pro práci s externí aplikací, která využívá protokol HTTP, je nezbytná vazba HTTP. Vazba HTTP transformuje data předaná v

rámci zprávy v nativním formátu obchodnímu objektu v aplikaci SCA. Vazba HTTP může také transformovat data předaná jako obchodní objekt do nativního formátu očekávaného externí aplikací.

Poznámka: Chcete-li umožnit interakci s klienty a službami, které využívají protokol SOAP/HTTP webových služeb, zvažte možnost použití jedné z vazeb webových služeb, které poskytují další funkčnost s ohledem na zpracování kvalit služby standardních pro webové služby.

Některé běžné scénáře použití vazby HTTP jsou popsány v následujícím seznamu:

- Služby s hostitelem SCA mohou vyvolávat aplikace HTTP pomocí importu HTTP.
- Služby s hostitelem SCA se mohou vystavit jako aplikace s povoleným protokolem HTTP, aby je mohli používat klienti HTTP pomocí exportu HTTP.
- Produkty IBM Business Process Manager a Process Server spolu mohou komunikovat přes infrastrukturu HTTP a v důsledku toho mohou uživatelé spravovat svou komunikaci podle firemních standardů.
- Produkty IBM Business Process Manager a Process Server se mohou chovat jako mediátory komunikace HTTP a transformovat a směrovat zprávy, což zlepšuje integraci aplikací pomocí sítě HTTP.
- Produkty IBM Business Process Manager a Process Server lze použít pro přemostění mezi HTTP a ostatními protokoly, např. webovými službami SOAP/HTTP, adaptéry prostředku na bázi JCA (Java Connector Architecture) apod.

Podrobné informace o vytváření vazeb importu a exportu HTTP naleznete v Informačním centru produktu Integration Designer. Viz témata **Vývoj integračních aplikací > Přístup k externím aplikacím s HTTP>**.

Přehled vazeb HTTP:

Vazba HTTP zajišťuje konektivitu k aplikacím s hostitelem HTTP. Zajišťuje mediaci komunikace mezi aplikacemi HTTP a umožňuje volání existujících aplikací na bázi HTTP z modulu.

Vazby importu HTTP

Vazba importu HTTP zajišťuje odchozí konektivitu z aplikací architektury SCA (Service Component Architecture) k serveru nebo aplikacím HTTP.

Import vyvolá adresu URL koncového bodu HTTP. Tuto adresu URL lze určit jedním ze tří způsobů:

- Adresu URL lze nastavit dynamicky v záhlavích HTTP prostřednictvím URL dynamického potlačení.
- Adresu URL lze nastavit dynamicky v prvku adresy cíle SMO.
- Adresu URL lze určit jako konfigurační vlastnost importu.

Toto vyvolání je svým charakterem vždy synchronní.

Ačkoli jsou vyvolání HTTP vždy typu požadavek-odezva, import HTTP podporuje jednosměrné i obousměrné operace a v případě jednosměrné operace odpověď ignoruje.

Vazby exportu HTTP

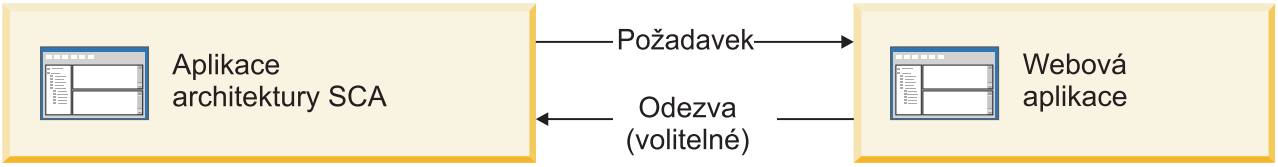
Vazba exportu HTTP zajišťuje příchozí konektivitu z aplikací HTTP do aplikace SCA.

Adresa URL je definována v exportu HTTP. Aplikace HTTP, které chtějí posílat exportu zprávy požadavků, použijí tuto adresu URL k vyvolání exportu.

Export HTTP také podporuje příkazy ping.

Vazby HTTP za běhu

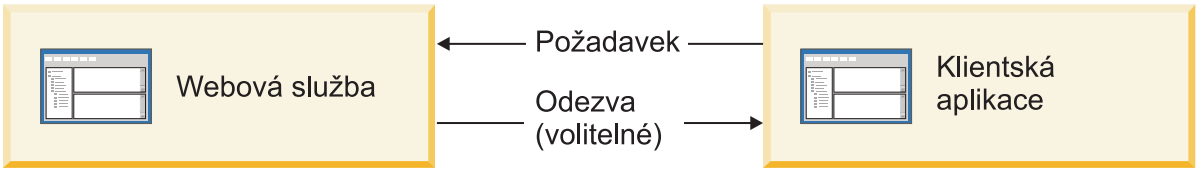
Import s vazbou HTTP za běhu odesílá požadavek s daty nebo bez nich v těle zprávy z aplikace SCA externí webové službě. Požadavek se provádí z aplikace SCA do externí webové služby, jak ukazuje Obrázek 26.



Obrázek 26. Tok požadavku z aplikace SCA do webové aplikace

Volitelně může import s vazbou HTTP přijímat data od webové aplikace jako odpověď na otázku.

S pomocí exportu je aplikací klienta proveden požadavek vůči webové službě, jak ukazuje Obrázek 27.



Obrázek 27. Tok požadavku z webové služby do aplikace klienta

Webová služba je webová aplikace spuštěná na serveru. Export je implementován v této webové aplikaci jako servlet, takže klient odesílá svůj požadavek na adresu URL. Servlet předá požadavek aplikaci SCA za běhu.

Volitelně může export v odpovědi na požadavek poslat aplikaci klienta data.

Záhlaví HTTP:

Vazby importu a exportu HTTP umožňují použití konfigurace záhlaví HTTP a jejich hodnot pro odchozí zprávy. Import HTTP využívá tato záhlaví pro požadavky a export HTTP je využívá pro odpovědi.

Staticky nakonfigurovaná záhlaví a řídicí informace mají přednost před hodnotami dynamicky nastavenými za běhu. Řídicí hodnoty Adresy URL dynamického potlačení, Verze a Metody však potlačí statické hodnoty, které jsou jinak považovány za výchozí.

Vazba podporuje dynamický charakter adresy URL importu HTTP určením hodnoty cílové Adresy URL, Verze a Metody HTTP za běhu. Tyto hodnoty jsou určovány extrahováním hodnoty Odkazu na koncový bod, Adresy URL dynamického potlačení, Verze a Metody.

- Jako odkaz na koncový bod využijte rozhraní API com.ibm.websphere.sca.addressing.EndpointReference nebo nastavte pole /headers/SMOHeader/Target/address v záhlaví objektu SMO.
- Jako Adresu URL, Verzi a Metodu pro dynamické přepsání použijte sekci řídicích parametrů HTTP zprávy architektury SCA (Service Component Architecture). Všimněte si, že Adresa URL pro dynamické přepsání má přednost před cílovým odkazem na koncový bod; odkaz na koncový bod však platí napříč vazbami, takže se jedná o upřednostňovaný přístup, který má být použit, kdekoli je to možné.

Řídicí informace a informace o záhlaví pro odchozí zprávy v rámci vazeb exportu a importu HTTP jsou zpracovány v tomto pořadí:

1. Informace o záhlaví a řídicí informace s výjimkou Adresy URL, Verze a Metody pro dynamické přepsání ze zprávy SCA v HTTP (nejnižší priorita).
2. Změny z administrativní konzoly na úrovni exportu/importu.

3. Změny z administrativní konzoly na úrovni metody exportu nebo importu.
4. Cílová adresa určená prostřednictvím odkazu na koncový bod nebo záhlaví objektu SMO.
5. Adresa URL, Verze a Metoda pro dynamické přepsání ze zprávy SCA.
6. Záhlaví a řídicí informace z popisovače dat nebo vázání dat (nejvyšší priorita).

Export a import HTTP naplní záhlaví a řídicí parametry v příchozím směru daty z příchozí zprávy (HTTPExportRequest a HTTPImportResponse) pouze, pokud je šíření záhlaví protokolu nastaveno na **True**. Na druhou stranu, export a import HTTP přečte a zpracuje odchozí záhlaví a řídicí parametry (HTTPExportResponse a HTTPImportRequest) pouze, pokud je šíření záhlaví protokolu nastaveno na **True**.

Poznámka: Změny, které provede popisovač dat nebo vázání dat v záhlavích nebo řídicích parametrech v odpovědi importu nebo požadavku exportu, nezmění pokyny pro zpracování zprávy uvnitř vazby importu nebo exportu a měly by být používány pouze k šíření upravených hodnot do následných komponent SCA.

Služba kontextu je odpovědná za šíření kontextu (včetně záhlaví protokolů, např. záhlaví HTTP, a uživatelského kontextu, např. ID účtu) po cestě vyvolání SCA. Při vývoji v produktu IBM Integration Designer můžete ovládat šíření kontextu prostřednictvím vlastností importu a exportu. Další podrobnosti viz informace o vazbách importu a exportu v Informačním centru produktu IBM Integration Designer.

Dodávané struktury záhlaví HTTP a podpora

Tabulka 23 rozkládá parametry požadavku/odpovědi pro požadavky a odpovědi importu HTTP a exportu HTTP na položky.

Tabulka 23. Dodávané informace o záhlaví HTTP

Název ovládacího prvku	Požadavek importu HTTP	Odpověď importu HTTP	Požadavek exportu HTTP	Odpověď exportu HTTP
Adresa URL	Ignorováno	Nenastaveno	Načteno ze zprávy požadavku. **Poznámka:** Součástí řídicího parametru adresy URL je i řetězec dotazu.	Ignorováno
Verze (možné hodnoty: 1.0, 1.1; výchozí hodnota je 1.1)	Ignorováno	Nenastaveno	Načteno ze zprávy požadavku	Ignorováno
Metoda	Ignorováno	Nenastaveno	Načteno ze zprávy požadavku	Ignorováno
Adresa URL pro dynamické přepsání	Pokud je nastavena v popisovači dat nebo ve vázání dat, přepíše Adresu importu HTTP. Zapisuje se do zprávy na řádku požadavku. **Poznámka:** Součástí řídicího parametru adresy URL je i řetězec dotazu.	Nenastaveno	Nenastaveno	Ignorováno
Verze pro dynamické přepsání	Je-li nastavena, přepíše Verzi importu HTTP. Zapisuje se do zprávy na řádku požadavku.	Nenastaveno	Nenastaveno	Ignorováno

Tabulka 23. Dodávané informace o záhlaví HTTP (pokračování)

Název ovládacího prvku	Požadavek importu HTTP	Odpověď importu HTTP	Požadavek exportu HTTP	Odpověď exportu HTTP
Metoda pro dynamické přepsání	Je-li nastavena, přepíše Metodu importu HTTP. Zapisuje se do zprávy na řádku požadavku.	Nenastaveno	Nenastaveno	Ignorováno
Typ média (Tento řídicí parametr nese část hodnoty záhlaví HTTP Typ obsahu.)	Je-li uveden, zapisuje se do zprávy jako část záhlaví Typ obsahu. **Poznámka:** Hodnotu tohoto řídicího prvku má poskytovat popisovač dat nebo vázání dat.	Načteno ze zprávy odpovědi, záhlaví Typ obsahu.	Načteno ze zprávy požadavku, záhlaví Typ obsahu.	Je-li uveden, zapisuje se do zprávy jako část záhlaví Typ obsahu. **Poznámka:** Hodnotu tohoto řídicího prvku má poskytovat popisovač dat nebo vázání dat.
Znaková sada (výchozí: UTF-8)	Je-li uvedena, zapisuje se do zprávy jako část záhlaví Typ obsahu. **Poznámka:** Hodnotu tohoto řídicího prvku má poskytovat vázání dat.	Načteno ze zprávy odpovědi, záhlaví Typ obsahu.	Načteno ze zprávy požadavku, záhlaví Typ obsahu.	Podporováno; zapisuje se do zprávy jako část záhlaví Typ obsahu. **Poznámka:** Hodnotu tohoto řídicího prvku má poskytovat vázání dat.
Kódování přenosu (Možné hodnoty: chunked, identity; výchozí je identity)	Je-li uvedeno, zapisuje se do zprávy jako záhlaví a řídí, jak je kódována transformace zprávy.	Načteno ze zprávy odpovědi.	Načteno ze zprávy požadavku.	Je-li uvedeno, zapisuje se do zprávy jako záhlaví a řídí, jak je kódována transformace zprávy.
Kódování přenosu (Možné hodnoty: gzip, x-gzip, deflate, identity; výchozí je identity)	Je-li uvedeno, zapisuje se do zprávy jako záhlaví a řídí, jak je kódován informační obsah.	Načteno ze zprávy odpovědi.	Načteno ze zprávy požadavku.	Je-li uvedeno, zapisuje se do zprávy jako záhlaví a řídí, jak je kódován informační obsah.
Délka obsahu	Ignorováno	Načteno ze zprávy odpovědi.	Načteno ze zprávy požadavku.	Ignorováno
Stavový kód (výchozí: 200)	Nepodporováno	Načteno ze zprávy odpovědi.	Nepodporováno	Je-li uveden, zapisuje se do zprávy na řádku odpovědi.
Fráze příčiny (výchozí: OK)	Nepodporováno	Načteno ze zprávy odpovědi.	Nepodporováno	Řídicí hodnota ignorována. Hodnota na řádku odpovědi zprávy je generována z kódu StatusCode.
Ověření (obsahuje více vlastností)	Je-li uvedeno, používá se ke konstrukci záhlaví Základní ověřování. **Poznámka:** Hodnota tohoto záhlaví bude kódována pouze v protokolu HTTP. V architektuře SCA bude dekódována a předána jako prostý text.	Nelze aplikovat	Načteno ze záhlaví Základní ověřování zprávy požadavku. Přítomnost tohoto záhlaví neznamená, že byl uživatel ověřen. Ověření by mělo být řízeno v konfiguraci servletu. **Poznámka:** Hodnota tohoto záhlaví bude kódována pouze v protokolu HTTP. V architektuře SCA bude dekódována a předána jako prostý text.	Nelze aplikovat

Tabulka 23. Dodávané informace o záhlaví HTTP (pokračování)

Název ovládacího prvku	Požadavek importu HTTP	Odpověď importu HTTP	Požadavek exportu HTTP	Odpověď exportu HTTP
Server proxy (obsahuje více vlastností: Hostitel, Port, Ověření)	Je-li uveden, používá se ke zřízení připojení prostřednictvím serveru proxy.	Nelze aplikovat	Nelze aplikovat	Nelze aplikovat
SSL (obsahuje více vlastností: Úložiště klíčů, Heslo úložiště klíčů, Úložiště údajů o důvěryhodnosti, Heslo úložiště údajů o důvěryhodnosti, Ověření klienta)	Pokud je naplněno a cílová adresa URL je HTTPS, používá se ke zřízení připojení prostřednictvím SSL.	Nelze aplikovat	Nelze aplikovat	Nelze aplikovat

Vázání dat HTTP:

Popisovač dat nebo vázání dat HTTP musí být nakonfigurovány pro všechna různá mapování dat mezi zprávou SCA (Service Component Architecture) a zprávou protokolu HTTP. Popisovače dat poskytují rozhraní nezávislé na vazbě, které umožňuje opakované použití napříč vazbami přenosu a představují doporučený přístup. Vázání dat jsou specifická pro konkrétní vazbu přenosu. Jsou dodávány třídy vázání dat specifické pro HTTP. Také si můžete napsat vlastní popisovače dat nebo datové vazby.

Poznámka: Tři třídy vázání dat HTTP popisované v tomto tématu (HTTPStreamDataBindingSOAP, HTTPStreamDataBindingXML a HTTPServiceGatewayDataBinding) jsou od produktu IBM Business Process Manager verze 7.0 zamítnuté. Místo použití vázání dat popsaných v tomto tématu zvažte použití těchto popisovačů dat:

- Použijte SOAPDataHandler místo HTTPStreamDataBindingSOAP.
- Použijte UTF8XMLDataHandler místo HTTPStreamDataBindingXML.
- Použijte GatewayTextDataHandler místo HTTPServiceGatewayDataBinding.

Jsou poskytnuta vázání dat pro použití s importy HTTP a exporty HTTP: vázání binárních dat, vázání dat XML a vázání dat SOAP. Pro jednosměrné operace není nutné vázání dat odpovědi. Vázání dat představuje název třídy Java, jejíž instance mohou převádět z HTTP na ServiceDataObject i naopak. Selektor funkcí musí být použit na export, který spolu s vazbami metod může určit které vázání dat se použije a která operace je vyvolána. Dodávaná vázání dat jsou tato:

- Vázání binárních dat, která považují tělo za nestrukturovaná binární data. Implementace schématu XSD vázání binárních dat je tato:

```
<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>
```

- Vázání dat XML, která podporují tělo jako data XML. Implementace vázání dat XML je podobná vázání dat JMS XML a nemá žádná omezení schématu rozhraní.
- Vázání dat SOAP, které podporují tělo jako data SOAP. Implementace vázání dat SOAP nemá žádná omezení schématu rozhraní.

Implementace vlastních vázáních dat HTTP

Tento oddíl popisuje, jak implementovat vlastní vázání dat HTTP.

Poznámka: Doporučený přístup je implementovat vlastní popisovač dat, protože ten lze používat opakovaně pro různé přenosové vazby.

Primárním rozhraním pro zpracování vlastních zpráv HTTP je HTTPStreamDataBinding. Toto rozhraní je nevrženo tak, aby umožňovalo zpracování velkých objemů informačního obsahu. Aby však taková implementace fungovala, musí toto vázání dat před zápisem zprávy do proudu vracet řídicí informace a záhlaví.

Vlastní vázání dat musí implementovat níže uvedené metody a jejich pořadí.

Chcete-li upravit vázání dat, napište třídu, která implementuje HTTPStreamDataBinding. Vázání dat by mělo mít čtyři soukromé vlastnosti:

- private DataObject pDataObject
- private HTTPControl pCtrl
- private HTTPHeaders pHeaders
- private yourNativeDataType nativeData

Vazba HTTP vyvolá upravené vázání dat v tomto pořadí:

- Odchozí zpracování (DataObject na Nativní formát):
  1. setDataObject(...)
  2. setHeaders(...)
  3. setControlParameters(...)
  4. setBusinessException(...)
  5. convertToNativeData()
  6. getControlParameters()
  7. getHeaders()
  8. write(...)
- Zpracování příchozích požadavků (Nativní formát na DataObject):
  1. setControlParameters(...)
  2. setHeaders(...)
  3. convertFromNativeData(...)
  4. isBusinessException()
  5. getDataObject()
  6. getControlParameters()
  7. getHeaders()

Potřebujete vyvolat setDataObject(...) v metodě convertFromNativeData(...) pro nastavení hodnoty objektu dataObject, který je převeden z nativních dat na soukromou vlastnost "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;
}
/*
* Přidejte záhlaví http "IsBusinessException" do pHeaders.
* Dva kroky:
* 1. Nejprve odebrat všechna záhlaví s názvem IsBusinessException (bez rozlišení malých a velkých písmen).
*   Tím se zajistí, že bude přítomné pouze jedno záhlaví.
* 2. Přidat nové záhlaví "IsBusinessException"
*/
public void setBusinessException(boolean isBusinessException) {
  //odebrat všechna záhlaví s názvem IsBusinessException (bez rozlišení malých a velkých písmen).
  //Tím se zajistí, že bude přítomné pouze jedno záhlaví.
  //přidat nové záhlaví "IsBusinessException", příklad kódu:
  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;
}
/*
* Získat záhlaví "IsBusinessException" z pHeaders, vrátit jeho logickou hodnotu
*/
public boolean isBusinessException() {
  String headerValue = getHeaderValue(pHeaders,"IsBusinessException");
  boolean result=Boolean.parseBoolean(headerValue);
  return result;
}
public void convertToNativeData() throws DataBindingException {
  DataObject dataObject = getDataObject();
  this.nativeData=realConvertWorkFromSDOToNativeData(dataObject);
}
public void convertFromNativeData(HTTPInputStream arg0){
  //Metoda vyvinutá zákazníkem, která má
  //Číst data z proudu HTTPInputStream
  //Převést je na DataObject
  DataObject dataobject=realConvertWorkFromNativeDataToSDO(arg0);
  setDataObject(dataobject);
}
public void write(HTTPOutputStream output) throws IOException {
  if (nativeData != null)
  output.write(nativeData);
}
```

Vázání EJB:

Vazby importu Enterprise JavaBeans (EJB) umožňují komponentám architektury SCA (Service Component Architecture) vyvolávat služby poskytované obchodní logikou Java EE spuštěnou na serveru Java EE. Vazby exportu EJB umožňují vystavení komponent SCA jako objektů Enterprise JavaBean, takže může obchodní logika Java EE vyvolávat komponenty SCA, které by pro ni jinak byly nedostupné.

Vazby importu EJB:

Vazby importu EJB umožňují modulu SCA volat implementace EJB určením způsobu, jak je přijímající modul vázán k externímu objektu EJB. Import služeb z externí implementace EJB umožňuje uživatelům zapojit obchodní logiku do prostředí IBM Business Process Manager a podílet se na obchodním procesu.

Vazby importu EJB lze vytvářet v produktu Integration Designer. Vazby můžete vygenerovat některou z těchto procedur:

- Vytvoření importu EJB pomocí průvodce externí službou

  Pomocí průvodce externí službou v produktu Integration Designer můžete vytvořit import založený na již existující implementaci. Průvodce externí službou vytváří služby založené na vámi zadaných kritériích. Poté vygeneruje obchodní objekty, rozhraní a soubory importu na základě zjištěných služeb.
- Vytvoření importu EJB pomocí editoru sestavení

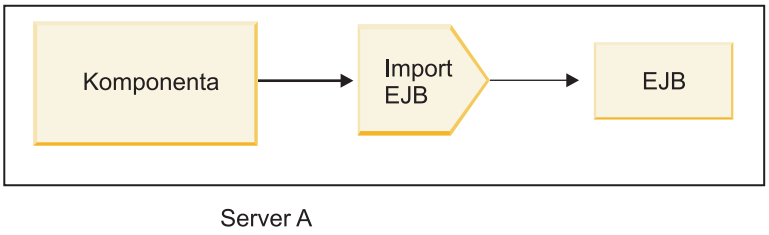
  Import EJB můžete vytvořit v rámci diagramu sestavení v editoru sestavení produktu Integration Designer. Z palety můžete buď použít Import, nebo vytvořit vázání EJB pomocí třídy Java.

Vygenerovaný import má vázání dat, která vytvoří připojení Java-WSDL, místo aby vyžadovala komponentu mostu Java. Komponentu s odkazem WSDL (Web Services Description Language) můžete spojit přímo s importem EJB, který komunikuje se službou založenou na EJB prostřednictvím rozhraní Java.

Import EJB může vstupovat do interakcí s obchodní logikou Java EE buď prostřednictvím programovacího modelu EJB 2.1, nebo prostřednictvím programovacího modelu EJB 3.0.

Vyvolání obchodní logiky Java EE může být lokální (pouze pro EJB 3.0) nebo vzdálené.

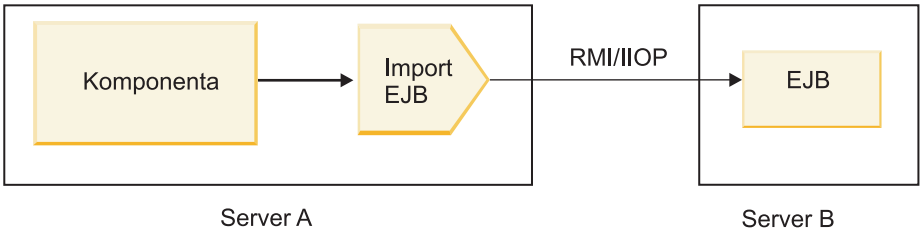
- Lokální vyvolání se používá, když chcete volat obchodní logiku Java EE, která se nachází na stejném serveru jako import.



Obrázek 28. Lokální vyvolání EJB (pouze u EJB 3.0)

- Vzdálené vyvolání se používá, když chcete volat obchodní logiku Java EE, která se nenachází na stejném serveru jako import.

  Například na následujícím obrázku využívá import EJB vyvolání RMI/IIOP (Remote Method Invocation over Internet InterORB Protocol) k vyvolání metody EJB na jiném serveru.



Obrázek 29. Vzdálené vyvolání EJB

Při konfiguraci vázání EJB používá produkt Integration Designer název rozhraní JNDI k určení úrovně programovacího modelu EJB a typu vyvolání (lokální nebo vzdálené).

Vazby importu EJB obsahují tyto hlavní komponenty:

- Popisovač dat JAX-WS.
- Selektor poruch EJB.
- Selektor funkcí importu EJB.

Pokud váš uživatelský scénář není založen na mapování JAX-WS, budete možná potřebovat vlastní popisovač dat, selektor funkcí a selektor poruch, aby prováděly úlohy jinak prováděné komponentami, jež jsou součástí vazeb importu EJB. To zahrnuje i mapování normálně prováděné vlastním algoritmem mapování.

Vazby exportu EJB:

Externí aplikace Java EE mohou vyvolat komponentu SCA prostřednictvím vazby exportu EJB. Použití exportu EJB umožňuje vystavit komponenty SCA tak, aby je mohly externí aplikace Java EE vyvolávat pomocí programovacího modelu EJB.

Poznámka: Export EJB je bezstavový objekt typu bean.

Vázání EJB můžete vytvářet v produktu Integration Designer. Vazby můžete vygenerovat některou z těchto procedur:

- Vytvoření vazeb exportu EJB pomocí průvodce externí službou

  Pomocí průvodce externí službou v produktu Integration Designer můžete vytvořit službu exportu založenou na již existující implementaci. Průvodce externí službou vytváří služby založené na vámi zadaných kritériích. Potom vygeneruje obchodní objekty, rozhraní a soubory exportu založené na zjištěných službách.
- Vytvoření vazeb exportu EJB pomocí editoru sestavení

  Export EJB můžete vytvořit v editoru sestavení produktu Integration Designer.

Důležité: Klient J2SE (Java 2 Platform, Standard Edition) nemůže vyvolat klienta exportu EJB, který je vygenerován v produktu Integration Designer.

Vazbu můžete vygenerovat z již existující komponenty SCA, nebo můžete vygenerovat export s vázáním EJB pro rozhraní Java.

- Při generování exportu pro již existující komponentu SCA, která má existující rozhraní WSDL, je exportu přiřazeno rozhraní Java.
- Při generování exportu pro rozhraní Java můžete pro tento export vybrat buď rozhraní WSDL, nebo Java.

  Poznámka: Rozhraní Java použité k vytvoření exportu EJB má následující omezení s ohledem na objekty (vstupní a výstupní parametry a výjimky) předávané jako parametry v rámci vzdáleného volání:
  - Musí být konkrétního typu (místo rozhraní nebo abstraktního typu).
  - Musí odpovídat specifikaci Enterprise JavaBeans. Musí být serializovatelné a mít výchozí konstruktor bez argumentů a všechny vlastnosti musí být přístupné prostřednictvím metody getter a modulu nastavení.

    Informace o specifikaci Enterprise JavaBeans viz web společnosti Sun Microsystems, Inc. na adrese http://java.sun.com.

  Navíc musí být výjimka kontrolovaná výjimka, zděděná od java.lang.Exception a musí být singulární (tzn. nepodporuje vracení různých typů kontrolovaných výjimek).

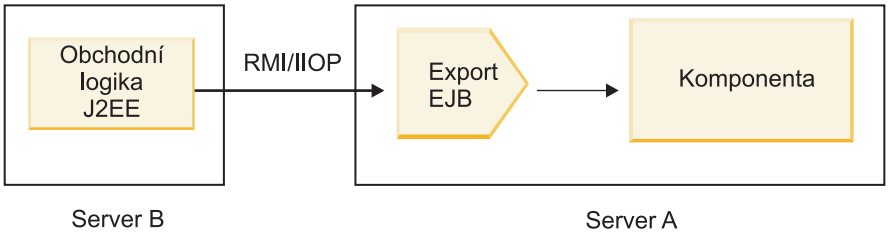
  Všimněte si také, že obchodní rozhraní objektu Java EnterpriseBean je prosté rozhraní Java a nesmí rozšiřovat javax.ejb.EJBObject ani javax.ejb.EJBLocalObject. Metody obchodního rozhraní by neměly vracet výjimku java.rmi.Remote.Exception.

Vazby exportu EJB mohou vstupovat do interakcí s obchodní logikou Java EE s pomocí programovacího modelu EJB 2.1 nebo programovacího modelu EJB 3.0.

Vyvolání může být lokální (pouze pro EJB 3.0) nebo vzdálené.

- Lokální vyvolání se používá, když obchodní logika Java EE volá komponentu SCA, která se nachází na stejném serveru jako export.
- Vzdálené vyvolání se používá, když se obchodní logika Java EE nenachází na stejném serveru jako export.

  Například na následujícím obrázku EJB pomocí RMI/IIOP volá komponentu SCA na jiném serveru.



Obrázek 30. Vzdálené volání z klienta do komponenty SCA prostřednictvím exportu EJB

Při konfiguraci vázání EJB používá produkt Integration Designer název rozhraní JNDI k určení úrovně programovacího modelu EJB a typu vyvolání (lokální nebo vzdálené).

Vazby exportu EJB obsahují tyto hlavní komponenty:

- Popisovač dat JAX-WS.
- Selektor funkcí exportu EJB.

Pokud váš uživatelský scénář není založen na mapování JAX-WS, budete možná potřebovat vlastní popisovač dat a selektor funkcí, aby prováděly úlohy jinak prováděné komponentami, jež jsou součástí vazeb exportu EJB. To zahrnuje i mapování normálně prováděné vlastním algoritmem mapování.

Vlastnosti vázání EJB:

Vazby importu EJB používají své nakonfigurované názvy rozhraní JNDI k určení úrovně programovacího modelu EJB a typu vyvolání (lokální nebo vzdálené). Vazby importu a exportu EJB používají k transformaci dat popisovač dat JAX-WS. Vazba importu EJB využívá selektor funkcí importu EJB a selektor poruch EJB a vazba exportu EJB využívá selektor funkcí exportu EJB.

Názvy rozhraní JNDI a vazby importu EJB:

Při konfiguraci vázání EJB na importu používá produkt Integration Designer název rozhraní JNDI k určení úrovně programovacího modelu EJB a typu vyvolání (lokální nebo vzdálené).

Pokud není určen žádný název rozhraní JNDI, použije se výchozí vazba rozhraní EJB. Výchozí vytvořené názvy závisí na tom, zda vyvoláváte objekty JavaBeans EJB 2.1 nebo objekty JavaBeans EJB 3.1.

Poznámka: Podrobnější informace o konvencích pojmenování viz téma "Přehled vazeb aplikací EJB 3.0" v Informačním centru produktu WebSphere Application Server. .

- JavaBeans EJB 2.1

  Výchozím názvem rozhraní JNDI předvoleného produktem Integration Designer je výchozí vázání EJB 2.1, která má formu **ejb/** plus domovské rozhraní, oddělené lomítky.

  Například pro domovské rozhraní objektů JavaBeans EJB 2.1 pro com.mycompany.myremotebusinesshome je výchozí vazba:

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

  Pro EJB 2.1 je podporováno pouze vzdálené vyvolání EJB.
- JavaBeans EJB 3.0

Výchozím názvem rozhraní JNDI předvoleného produktem Integration Designer pro lokální rozhraní JNDI je úplný název třídy lokálního rozhraní, jemuž předchází **ejblocal:**. Například pro úplné rozhraní lokálního objektu interface com.mycompany.mylocalbusiness je předvolené rozhraní JNDI EJB 3.0:

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

Pro vzdálený objekt interface com.mycompany.myremotebusiness je předvoleným rozhraním JNDI EJB 3.0 úplné rozhraní:

```
com.mycompany.myremotebusiness
```

Výchozí vazby aplikací EJB 3.0 jsou popsány na tomto místě: Přehled vazeb aplikací EJB 3.0.

Produkt Integration Designer použije jako výchozí umístění rozhraní JNDI pro objekty EJB s použitím programovacího modelu verze 3.0 "zkrácený" název.

Poznámka: Pokud se odkaz na implementované rozhraní JNDI cílového objektu EJB liší od výchozího umístění vazby rozhraní JNDI, protože bylo použito nebo nakonfigurováno vlastní mapování, musí být řádně určen název cílového rozhraní JNDI. Název můžete určit v produktu Integration Designer před implementací, nebo pro vazbu importu, můžete název změnit v administrativní konzole (po implementaci) tak, aby odpovídal názvu rozhraní JNDI cílového objektu EJB.

Další informace o tvorbě vázání EJB viz Práce s vazbami EJB v Informačním centru produktu Integration Designer.

Popisovač dat JAX-WS:

Vazba importu EJB (Enterprise JavaBeans) využívá popisovač dat JAX-WS k transformaci obchodních objektů požadavku na parametry objektů Java a k transformaci návratové hodnoty objektu Java na obchodní objekt odpovědi. Vazba exportu EJB využívá popisovač dat JAX-WS k transformaci objektů EJB požadavku na obchodní objekty požadavku a k transformaci obchodního objektu odpovědi na návratovou hodnotu.

Tento popisovač dat mapuje data z rozhraní WSDL specifikovaného v SCA na rozhraní Java cílového objektu EJB (a naopak) s využitím specifikace JAX-WS (Java API for XML Web Services) a specifikace JAXB (Java Architecture for XML Binding).

Poznámka: Aktuální podpora je omezena na specifikace JAX-WS 2.1.1 a JAXB 2.1.3.

Popisovač dat určený na úrovni vázání EJB se používá ke zpracování požadavků, odpovědí, poruch a výjimek za běhu.

Poznámka: V případě poruch je možné pro každou poruchu určit specifický popisovač dat prostřednictvím konfigurační vlastnosti faultBindingType. To přepíše hodnotu určenou na úrovni vázání EJB.

Když má vázání EJB rozhraní WSDL, použije se standardně popisovač dat JAX-WS. Tento popisovač dat nelze použít k transformaci zprávy SOAP představující vyvolání JAX-WS na datový objekt.

Vazba importu EJB používá k transformaci datového objektu na pole objektů Java (Object[]) popisovač dat. Při odchozí komunikaci dochází k tomuto zpracování:

1. Vázání EJB nastaví očekávaný typ, očekávaný prvek a název cílové metody v kontextu BindingContext tak, aby odpovídaly hodnotám určeným ve WSDL.
2. Vázání EJB vyvolá metodu transformace pro datový objekt vyžadující transformaci dat.
3. Popisovač dat vrátí pole Object[] představující parametry dané metody (v pořadí, v němž jsou v rámci metody definovány).
4. Vázání EJB použije pole Object[] k vyvolání metody na cílovém rozhraní EJB.

Vazba také připraví pole Object[] ke zpracování odpovědi od vyvolání EJB.

- Prvním prvkem v poli Object[] je návratová hodnota z vyvolání metody Java.
- Další hodnoty představují vstupní parametry metody.

To je nezbytné pro podporu parametrů typu Vstupní/Výstupní a Výstupní.

Pro parametry typu Výstupní se musí hodnoty vracet v datovém objektu odpovědi.

Popisovač dat zpracovává a transformuje hodnoty nalezení v poli Object[] a poté vrací odpověď datovému objektu.

Popisovač dat podporuje xs:AnyType, xs:AnySimpleType a xs:Any spolu s dalšími datovými typy XSD. Chcete-li povolit podporu pro xs:Any, jako vlastnost objektu JavaBeans v kódu Java použijte anotaci **@XmlAnyElement (lax=true)**, jak ukazuje následující příklad:

```
public class TestType {
 private Object[] object;

 @XmlAnyElement (lax=true)
 public Object[] getObject() {
  return object;
 )

 public void setObject (Object[] object) {
  this.object=object;
 )
)
```

Tím se z objektu vlastnosti v typu TestType stane pole xs:any. Hodnota třídy Java použitá v poli xs:any by měla mít anotaci **@XmlAnyElement**. Pokud je například Adresa třída Java používaná k naplnění pole objektů, měla by mít třída Adresa anotaci **@XmlRootElement**.

Poznámka: Chcete-li upravit mapování z typu XSD na typy Java definované specifikací JAX-WS, změňte anotace JAXB tak, aby odpovídaly vašim obchodním potřebám. Popisovač dat JAX-WS podporuje xs:any, xs:anyType a xs:anySimpleType.

Pro popisovač dat JAX-WS platí tato omezení:

- Popisovač dat nezahrnují podporu pro anotaci **@WebParam** atributu záhlaví.
- Obor názvů pro soubory schémat obchodních objektů (soubory XSD) nezahrnuje výchozí mapování z názvu balíku Java. Anotace **@XMLSchema** v objektu package-info.java také nefunguje. Jediný způsob, jak vytvořit XSD s oborem názvů, je použít anotace **@XmlType** a **@XmlRootElement**. Anotace **@XmlRootElement** definuje cílový obor názvů pro globální prvek v typech JavaBeans.
- Průvodce importem EJB nevytváří soubory XSD pro nesouvisející třídy. Verze 2.0 nepodporuje anotaci **@XmlSeeAlso**, takže pokud se na podřízenou třídu neodkazuje přímo z nadřízené třídy, XSD se nevytvoří. Tento problém lze vyřešit spuštěním obslužného programu SchemaGen pro takové podřízené třídy.

  SchemaGen je obslužný program příkazového řádku (umístěný v adresáři *domovský_adresář_instalace_WPS*/bin) určený pro vytvoření souborů XSD pro daný objekt typu bean. Aby toto řešení fungovalo, musí být tyto soubory XSD ručně zkopírovány do daného modulu.

Selektor poruch EJB:

Selektor poruch EJB určuje, zda je výsledkem vyvolání EJB porucha, výjimka za běhu nebo úspěšná odpověď.

Pokud je zjištěna porucha, vrátí selektor poruch EJB běhovému prostředí vazby nativní název poruchy, takže může popisovač dat JAX-WS převést objekt výjimky na obchodní objekt poruchy.

Při úspěšné (nechybové) odpovědi vazba importu EJB sestaví pole objektů Java (Object[]) pro vracení hodnot.

- Prvním prvkem v poli Object[] je návratová hodnota z vyvolání metody Java.
- Další hodnoty představují vstupní parametry metody.

To je nezbytné pro podporu parametrů typu Vstupní/Výstupní a Výstupní.

Ve scénářích výjimek vazba sestaví pole Object[] a první prvek představuje výjimku vrácenou metodou.

Selektor poruch může vrátit kteroukoli z následujících hodnot:

Tabulka 24. Návratové hodnoty

Typ	Návratová hodnota	Popis
Porucha	**ResponseType.FAULT**	Vrácena, když předané pole Object[] obsahuje objekt výjimky.
Výjimka za běhu	**ResponseType.RUNTIME**	Vrácena, když objekt výjimky neodpovídá žádnému deklarovanému typu výjimek metody.
Normální odpověď	**ResponseType.RESPONSE**	Vracena ve všech ostatních případech.

Pokud selektor poruch vrátí hodnotu **ResponseType.FAULT**, je vrácen nativní název poruchy. Tento nativní název poruchy používá vazba k určení odpovídajícího názvu poruchy WSDL z modelu a k vyvolání správného popisovače dat způsobujících poruchu.

Selektor funkcí EJB:

Vázání EJB využívá k určení metody EJB, která se má volat, selektor funkcí importu (pro zpracování odchozích požadavků) nebo selektor funkcí exportu (pro zpracování příchozích požadavků).

Selektor funkcí importu

Pro zpracování odchozích požadavků selektor funkcí importu odvodí typ metody EJB na základě názvu operace vyvolané komponentou SCA, která je spojena s importem EJB. Selektor funkcí hledá anotaci @WebMethod u třídy Java generované produktem Integration Designer a anotované podle specifikace JAX-WS, aby určil název přidružené cílové operace.

- Pokud je přítomna anotace @WebMethod, použije ji selektor funkcí k určení správného mapování metody Java pro metodu WSDL.
- Pokud anotace @WebMethod chybí, selektor funkcí předpokládá, že je název metody Java stejný jako název vyvolané operace.

Poznámka: Tento selektor funkcí je platný pouze pro rozhraní typu WSDL importu EJB, nikoli pro rozhraní typu Java importu EJB.

Selektor funkcí vrací objekt java.lang.reflect.Method, který představuje metodu rozhraní EJB.

Selektor funkcí používá pole objektů Java (Object[]), který obsahuje odpověď cílové metody. Prvním prvkem pole Object[] je metoda Java s názvem WSDL a druhým prvkem pole Object[] je vstupní obchodní objekt.

Selektor funkcí exportu

Pro zpracování příchozích požadavků selektor funkcí exportu odvodí cílovou metodu, která má být vyvolána, z metody Java.

Selektor funkcí exportu mapuje název operace Java vyvolané klientem EJB na název operace v rozhraní cílové komponenty. Název metody je vracen jako řetězec a je řešen běhovým prostředím SCA v závislosti na typu rozhraní cílové komponenty.

Vazby EIS:

Vazby podnikových informačních systémů (EIS) zajišťují konektivitu mezi komponentami SCA a externím podnikovým informačním systémem. Této komunikace je dosaženo prostřednictvím exportů EIS a importů EIS, které podporují adaptéry prostředku JCA 1.5 a adaptéry WebSphere Adapter.

Vaše komponenty SCA mohou vyžadovat převod dat do nebo z externího podnikového informačního systému. Do vytváření modulu SCA, který vyžaduje takovou konektivitu, zahrnete (kromě své komponenty SCA) import nebo export s vazbou EIS pro komunikaci se specifickým externím podnikovým informačním systémem.

Adaptéry prostředku v produktu IBM Integration Designer se používají v rámci kontextu importu nebo exportu. Pomocí průvodce externí službou vyvinete import či export a při vývoji do něj zahrnete adaptér prostředku. Import EIS, který umožní vaší aplikaci vyvolat službu v systému EIS, nebo export EIS, který aplikaci umožní v systému EIS vyvolat službu vyvinutou v produktu IBM Integration Designer, jsou vytvořeny s konkrétním adaptérem prostředku. Například byste vytvořili import s adaptérem JD Edwards pro vyvolání služby v systému JD Edwards.

Když použijete průvodce externí službou, vytvoří informace o vazbě EIS za vás. Informace o vazbách můžete také přidávat a upravovat pomocí jiného nástroje, editoru sestavení. Další informace viz Přístup k externím službám pomocí adaptérů.

Po implementaci modulu SCA, který obsahuje danou vazbu EIS, na serveru můžete v administrativní konzole zobrazit informace o této vazbě nebo vazbu nakonfigurovat.

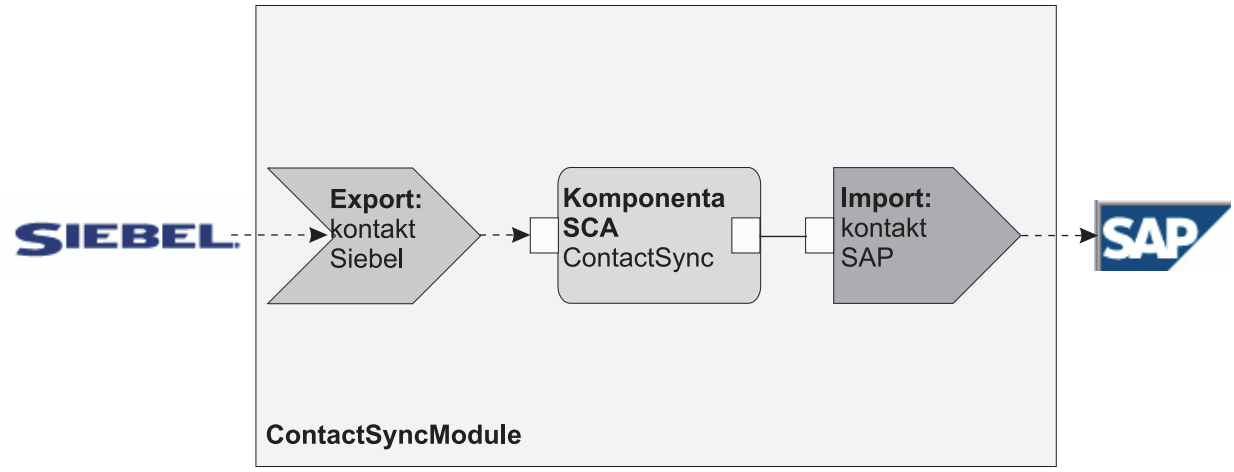
Přehled vazeb EIS:

Vazba EIS (podnikového informačního systému), je-li použita s adaptérem prostředku JCA, umožňuje přístup ke službám v podnikovém informačním systému nebo tomuto systému povolí vaše služby.

Následující příklad ukazuje, jak modul SCA nazvaný ContactSyncModule synchronizuje kontaktní informace mezi systémem Siebel a systémem SAP.

1. Komponenta SCA nazvaná ContactSync naslouchá (prostřednictvím exportu aplikace EIS nazvaného Kontakt Siebel) změnám v kontaktech systému Siebel.
2. Sama komponenta SCA ContactSync využívá aplikaci SAP (prostřednictvím importu aplikace EIS), aby příslušným způsobem aktualizovala kontaktní informace SAP.

Protože se datové struktury používané pro ukládání kontaktů v systémech Siebel a SAP liší, musí komponenta SCA ContactSync zajistit mapování.



Obrázek 31. Tok ze systému Siebel do systému SAP

Export Kontakt Siebel a import Kontakt SAP mají nakonfigurovány příslušné adaptéry prostředku.

Klíčové funkce vázání EIS:

Import EIS je importem architektury SCA, který umožňuje komponentám v modulu SCA využívat aplikace EIS definované mimo modul SCA. Data importu EIS jsou používána k přenosu dat z komponenty SCA na externí EIS. Export EIS se používá k přenosu dat z externího EIS do modulu SCA.

Importy

Rolí importu EIS je přemostit mezeru mezi komponentami SCA a externími systémy EIS. Externí aplikace lze považovat za import EIS. V tom případě import EIS zasílá data na externí EIS a volitelně přijímá data jako odpověď.

Import EIS poskytuje komponenty SCA s jednotným pohledem aplikací mimo modul. To komponentám umožňuje komunikovat s externím EIS, jako je např. SAP, Siebel nebo PeopleSoft, pomocí konzistentního modelu SCA.

Na straně klienta importu existuje rozhraní vystavené aplikací importu EIS s jednou nebo dvěma metodami, z nichž každá přebírá objekty dat jako argumenty a vrací hodnoty. Na straně implementace existuje rozhraní Common Client Interface (CCI), které je implementováno adaptérem prostředku.

Implementace běhového prostředí importu EIS spojuje klientské rozhraní a CCI. Import mapuje vyvolání metody na rozhraní do vyvolání na CCI.

Vazby jsou vytvářeny na třech úrovních: vazba rozhraní, která poté používá vazby obsažené metody následně využívajícími vázání dat.

Vazba rozhraní propojuje rozhraní importu s připojením k systému EIS poskytujícímu aplikaci. To odráží fakt, že sada aplikací, představovaných rozhraním, je poskytnuta specifickou instancí systému EIS a přístup k této instanci zajišťuje připojení. Prvek vazby obsahuje vlastnosti s dostatečným počtem informací k vytvoření připojení (tyto vlastnosti jsou součástí instance javax.resource.spi.ManagedConnectionFactory).

Vazba metody přidružuje metodu se specifickou interakcí k systému EIS. Pro JCA je interakce charakterizována sadou vlastností implementace rozhraní javax.resource.cci.InteractionSpec. Prvek interakce vazby metody obsahuje tyto vlastnosti, společně s názvem třídy, díky čemuž poskytuje dostatek informací k provedení interakce. Vazba metody využívá vázání dat popisující mapování argumentu a výsledku metody rozhraní na reprezentaci EIS.

Scénář běhového prostředí pro import EIS:

1. Metoda na rozhraní importu je vyvolána pomocí programovacího modelu SCA.
2. Požadavek dosahující importu EIS obsahuje název metody a její argumenty.
3. Import nejprve vytvoří implementaci vazby rozhraní. Poté pomocí dat z vazby importu vytvoří nástroj ConnectionFactory a obě položky přidruží. To znamená, že import volá položku setConnectionFactory na vazbě rozhraní.
4. Je vytvořena implementace vazby metody odpovídající vyvolané metodě.
5. Vytvořena a naplněna je instance javax.resource.cci.InteractionSpec. Poté je vázání dat použito k vázání argumentů metod k formátu, kterému porozumí adaptér prostředku.
6. Rozhraní CCI se používá k provedení interakce.
7. Když se vrátí volání, použije se vázání dat k vytvoření výsledku vyvolání a výsledek je vrácen volajícímu.

Exporty

Rolí exportu EIS je přemostit mezeru mezi komponentou SCA a externím systémem EIS. Externí aplikace lze považovat za export EIS. V tomto případě externí aplikace odesílá svá data ve formě periodických oznámení. O exportu EIS lze uvažovat jako o aplikaci předplatného naslouchající externímu požadavku ze strany EIS. Komponenta SCA, která využívá export EIS, jej zobrazuje jako lokální aplikaci.

Export EIS poskytuje komponenty SCA s jednotným pohledem aplikací mimo modul. To komponentám umožňuje komunikovat s EIS, jako je např. SAP, Siebel nebo PeopleSoft, pomocí konzistentního modelu SCA.

Export představuje implementaci listeneru přijímající požadavky z EIS. Listener implementuje rozhraní listeneru specifické pro adaptér prostředku. Export rovněž obsahuje rozhraní implementující komponenty, které je systému EIS vystaveno prostřednictvím exportu.

Implementace běhového prostředí exportu EIS spojuje listener a rozhraní implementující komponenty. Mapy exportu požadavku EIS na vyvolání příslušné operace na komponentě. Vazby jsou vytvářeny na třech úrovních: vázání listeneru, které následně využívá vázání nativní metodou následně využívající vázání dat.

Vázání listeneru vztahuje listener přijímající požadavek ke komponentě vystavené prostřednictvím exportu. Definice exportu obsahuje název komponenty. Běhové prostředí ji vyhledá a předá jí požadavky.

Vázání nativní metody přidružuje nativní metodu nebo typ události přijatý listenerem k operaci implementované komponentou vystavenou pomocí exportu. Mezi metodou vyvolanou listenerem a typem události není žádný vztah. Všechny události přicházejí prostřednictvím jedné nebo více metod listeneru. Vázání nativní metody využívá selektor funkcí definovaný v exportu k extrakci názvu nativní metody z příchozích dat a vázání dat využívá k vazbě formátu dat systému EIS na formát, kterému porozumí komponenta.

Scénář běhového prostředí pro export EIS:

1. Vyvolání spouštěčů požadavku EIS metody na implementaci listeneru.
2. Listener vyhledává a vyvolává export, přičemž mu předává všechny argumenty vyvolání.
3. Export vytváří implementaci vázání listeneru.
4. Export dokládá příkladem selektor funkcí a nastavuje jej na vázání listeneru.
5. Export inicializuje vázání nativní metody a přidává je k vázání listeneru. Pro všechna vázání nativní metody jsou rovněž inicializována vázání dat.
6. Export vyvolává vázání listeneru.
7. Vázání listeneru vyhledává exportované komponenty a využívá selektoru funkcí k načtení názvu nativní metody.
8. Tento název se používá k vyhledání vázání nativní metody, které následně vyvolává cílovou komponentu.

Styl interakce adaptéru umožňuje vázání exportu EIS vyvolat cílovou komponentu asynchronně (výchozí nastavení) nebo synchronně.

Adaptéry prostředku

Import či export vyvinete pomocí průvodce externí službou a během vývoje můžete zahrnout adaptér prostředku. Adaptéry, které jsou k dispozici s nástrojem IBM Integration Designer a jež se používají pro přístup k systémům CICS, IMS, JD Edwards, PeopleSoft, SAP a Siebel, jsou určeny výhradně pro vývoj a testování. To znamená, že je můžete využít pouze k vývoji a testování svých aplikací.

Jakmile aplikaci implementujete, budete ke spuštění aplikace potřebovat licencované adaptéry běhového prostředí. Když však službu sestavujete, můžete k ní vložit adaptér. Váš způsob licencování adaptérů vám může umožnit použití vloženého adaptéru jako licencovaného adaptéru běhového prostředí. Tyto adaptéry vyhovují architektuře Java EE Connector Architecture (JCA 1.5). Otevřený standard JCA je standardem Java EE pro konektivitu systému EIS. Architektura JCA poskytuje spravovaný rámec. To znamená, že nástroj Quality of Service (QoS) je poskytován aplikačním serverem, který pro transakce nabízí správu životního cyklu a zabezpečení. Rovněž je kompatibilní se specifikacemi Enterprise Metadata Discovery s výjimkou nástrojů IBM CICS ECI Resource Adapter a IBM IMS Connector for Java.

Adaptéry WebSphere Business Integration Adapters, starší sada adaptérů, jsou průvodcem rovněž podporovány.

Prostředky Java EE

Modul EIS, modul architektury SCA následující vzor modulu EIS, je možné implementovat na platformu Java EE.

Implementace modulu EIS na platformu Java EE má za následek aplikaci, která je připravena k provedení, zabalena jako souboru EAR a implementována na server. Vytvořeny jsou všechny prostředky a artefakty Java EE. Aplikace je konfigurována a připravena ke spuštění.

Dynamické vlastnosti JCA Interaction Spec a Connection Spec:

Vazba EIS může přijímat vstup pro vlastnosti InteractionSpec a ConnectionSpec určené pomocí dobře definovaného podřízeného datového objektu, který doprovází informační obsah. To umožňuje dynamické interakce typu požadavek-odezva s adaptérem prostředku prostřednictvím vlastnosti InteractionSpec a ověření komponenty prostřednictvím vlastnosti ConnectionSpec.

Vlastnost javax.cci.InteractionSpec nese informace o tom, jak zpracovat požadavek na interakci s adaptérem prostředku. Také může nést informace o tom, jak bylo dosaženo interakce po daném požadavku. Tato obousměrná komunikace prostřednictvím interakcí je někdy označována jako *konverzace*.

Vazba EIS očekává informační obsah, který bude argumentem pro adaptér prostředku, který má obsahovat podřízený datový objekt nazvaný **properties**. Tento datový objekt vlastnosti bude obsahovat dvojice název-hodnota s názvem vlastností Interaction Spec ve specifickém formátu. Pravidla formátování jsou tato:

- Názvy musí začínat předponou **IS**, následovanou názvem vlastnosti. Například specifikace interakce s vlastností JavaBeans nazvanou **InteractionId** by měla název vlastnosti **ISInteractionId**.
- Dvojice název-hodnota představuje název a hodnotu jednoduchého typu vlastnosti Interaction Spec.

V tomto příkladě rozhraní specifikuje, že je vstupem operace datový objekt **Account**. Toto rozhraní vyvolává aplikaci vazby importu EIS se záměrem odeslat a přijmout dynamickou vlastnost InteractionSpec nazvanou **workingSet** s hodnotou **xyz**.

Obchodní graf nebo obchodní objekty na serveru obsahují základní obchodní objekt **properties**, který umožňuje odesílání dat specifických pro určitý protokol s informačním obsahem. Tento obchodní objekt **properties** je vestavěný a nemusí být při konstrukci obchodního objektu určen ve schématu XML. Stačí jej vytvořit a použít. Pokud máte nadefinovány vlastní datové typy na základě schématu XML, potřebujete určit prvek **properties**, který obsahuje očekávané dvojice název-hodnota.

```
BOFactory dataFactory = (BOFactory) \
serviceManager.locateService("com/ibm/websphere/bo/BOFactory");
 //Obálka pro rozhraní zabaleného stylu doc-lit,
 //přeskočit na informační obsah pro non doc-lit
 DataObject docLitWrapper = dataFactory.createByElement /
 ("http://mytest/eis/Account", "AccountWrapper");
```

Vytvořte informační obsah.

```
DataObject account = docLitWrapper.createDataObject(0);
  DataObject accountInfo = account.createDataObject("AccountInfo");
  //Provést nastavení informačního obsahu


  //Vytvořit data vlastností pro dynamickou interakci

  DataObject properties = account.createDataObject("properties");
```

Pro název workingSet nastavte očekávanou hodnotu (**xyz**).

```
properties.setString("ISworkingSet", "xyz");


  //Vyvolat službu s argumentem

  Service accountImport = (Service) \
  serviceManager.locateService("AccountOutbound");
  DataObject result = accountImport.invoke("createAccount", docLitWrapper);

  //Získat vrácenou vlastnost
```

```
DataObject retProperties = result.getDataObject("properties");

String workingset = retProperties.getString("ISworkingSet");
```

Vlastnosti ConnectionSpec můžete použít pro dynamické ověření komponent. Platí stejná pravidla jako výše, jen předpona názvu vlastnosti musí být **CS** (místo **IS**). Vlastnosti ConnectionSpec nejsou obousměrné. Stejný datový objekt **properties** může obsahovat vlastnosti IS i CS.

Chcete-li použít vlastnosti ConnectionSpec, nastavte vlastnost **resAuth** určenou pro vazbu importu na hodnotu **Application**. Ujistěte se také, že adaptér prostředku podporuje autorizaci komponent. Další podrobnosti viz kapitola 8: Specifikace architektury konektoru J2EE.

Externí klienti s vazbami EIS:

Server dokáže odesílat zprávy externím klientům pomocí vazeb EIS nebo od nich zprávy přijímat.

Externí klient, například webový portál nebo podnikový informační systém, potřebuje odeslat zprávu modulu SCA na serveru nebo potřebuje být vyvolán komponentou z tohoto serveru.

Klient vyvolá import EIS jako u všech ostatních aplikací, buď s použitím rozhraní DDI (Dynamic Invocation Interface) nebo s použitím rozhraní Java.

1. Externí klient vytvoří instanci rozhraní ServiceManager a vyhledá import EIS s použitím jeho referenčního názvu. Výsledkem hledání je implementace rozhraní služby.
2. Klient vytvoří vstupní argument, generický datový objekt vytvořený dynamicky s použitím schématu datového objektu. Tento krok se provádí s pomocí implementace rozhraní Service Data Object DataFactory.
3. Externí klient vyvolá podnikový informační systém a získá požadované výsledky.

Případně může klient vyvolat import EIS s pomocí rozhraní Java.

1. Klient vytvoří instanci rozhraní ServiceManager a vyhledá import EIS s použitím jeho referenčního názvu. Výsledkem hledání je rozhraní Java importu EIS.
2. Klient vytvoří vstupní argument a typovaný datový objekt.
3. Klient vyvolá podnikový informační systém a získá požadované výsledky.

Rozhraní exportu EIS definuje rozhraní exportované komponenty SCA, které je k dispozici externím aplikacím podnikového informačního systému. Toto rozhraní lze považovat za rozhraní, které vyvolá externí aplikace (např. SAP nebo PeopleSoft) prostřednictvím implementace běhového prostředí aplikace exportu EIS.

Export použije vazbu EISExportBinding ke svázání exportovaných služeb s externí aplikací EIS. Umožní přihlásit aplikaci obsaženou v modulu SCA k naslouchání požadavkům služby EIS. Vazba exportu EIS určuje mapování mezi definicí příchozích událostí, jak jim rozumí adaptér prostředku (s použitím rozhraní architektury Java EE Connector Architecture) a vyvolání operací SCA.

Vazba EISExportBinding vyžaduje, aby byly externí služby EIS založeny na příchozích kontraktech architektury Java EE Connector Architecture 1.5. Vazba EISExportBinding vyžaduje, aby byly popisovač dat nebo vázání dat určeny buď na úrovni vazby, nebo na úrovni metody.

Vazby JMS:

Poskytovatel JMS (Java Message Service) umožňuje systém zpráv na základě rozhraní API a programovacího modelu Java Messaging Service. . Poskytuje továrny připojení JMS pro vytvoření připojení pro cíle JMS a pro odesílání a příjem zpráv.

Vazby JMS lze použít při interakci s vazbou poskytovatele sběrnice SIB (Service Integration Bus) a jsou kompatibilní s JMS a JCA 1.5.

Vazby exportu a importu JMS umožňují modulu architektury SCA (Service Component Architecture) volat externí systémy JMS a přijímat z nich zprávy.

Vazby importu a exportu JMS zajišťují integraci s aplikacemi JMS s použitím poskytovatele JMS SIB na bázi JCA 1.5, který je součástí serveru WebSphere Application Server. Jiné adaptéry prostředku JMS na bázi JCA 1.5 podporovány nejsou.

Přehled vazeb JMS:

Vazby JMS zajišťují konektivitu mezi prostředím architektury SCA (Service Component Architecture) a systémy JMS.

Vazby JMS

Hlavní komponenty vazeb importu JMS i exportu JMS jsou:

- Adaptér prostředku: umožňuje spravovanou, obousměrnou konektivitu mezi modulem SCA a externím modulem JMS.
- Připojení: zapouzdřují virtuální připojení mezi klientem a aplikací poskytovatele.
- Cíle: používány klientem k určení cíle jím produkovaných zpráv nebo zdroje jím přijímaných zpráv.
- Data ověřování: používána k zabezpečení přístupu k vazbě.

Klíčové vlastnosti vazby JMS

Speciální záhlaví

Vlastnosti speciálního záhlaví se používají v importech a exportech JMS, aby bylo možné cíli říct, jak zpracovat zprávu.

Například mapy TargetFunctionName z nativní metody do operační metody.

Prostředky Java EE

Mnoho prostředků Java EE je vytvářeno, když jsou importy a exporty JMS implementována do prostředí Java EE.

ConnectionFactory
Používaná klienty k vytvoření připojení k poskytovateli JMS.

ActivationSpec
Používané importy pro načtení odpovědi na požadavek; exporty tuto funkci využívají, když konfigurují koncové body zprávy, které představují listenery zpráv v jejich interakci se systémem zasílání zpráv.

Destinations

- Cíl odesílání: při importu je to místo, kam je zasílán požadavek nebo odchozí zpráva; při exportu jde o místo, kam bude zaslána zpráva odpovědi v případě, že není nahrazena polem záhlaví JMSReplyTo v příchozí zprávě.
- Cíl příjmu: kam by měla být umístěna příchozí zpráva; při importu jde o odpověď; při exportu jde o požadavek.
- Cíl zpětného volání: Cíl systému SCA JMS použitý k uložení korelačních informací. Toto místo určení nečtěte, ani do něj nezapisujte.

Úloha instalace vytváří položku ConnectionFactory a tři místa určení. Rovněž vytváří položku ActivationSpec, pomocí které lze povolit listeneru zprávy běhového prostředí naslouchat odpovědím v cíli příjmu. Vlastnosti těchto prostředků jsou určeny v souboru importu a exportu.

Integrace JMS a adaptéry prostředku:

Platforma JMS (Java Message Service) zajišťuje integraci prostřednictvím dostupného adaptéru prostředku na bázi JMS JCA 1.5. Je poskytnuta plná podpora integrace JMS pro adaptér prostředku Service Integration Bus (SIB) JMS.

Pro integraci s externím systémem JMS kompatibilním s JCA 1.5 použijte poskytovatel JMS pro adaptér prostředku JCA 1.5. Externí služby kompatibilní s JCA 1.5 mohou přijímat a odesílat zprávy za účelem integrace s komponentami architektury SCA (Service Component Architecture) pomocí adaptéru prostředku SIB JMS.

Použití adaptérů prostředku JCA 1.5 specifických pro jiné poskytovatele není podporováno.

Vazby importu a exportu JMS:

Pomocí vazeb importu a exportu JMS můžete nastavit interakci modulů SCA se službami poskytovanými externími aplikacemi JMS.

Vazby importu JMS

Připojení k přidruženému poskytovateli JMS cílů JMS je vytvořeno pomocí továrny připojení JMS. Při správě továren připojení JMS pro výchozího poskytovatele systému zpráv použijte administrativní objekty továrny připojení.

Interakce s externími systémy JMS zahrnují použití cílů pro odesílání požadavků a příjem odpovědí.

Jsou podporovány dva typy scénářů využití vazeb importu JMS, v závislosti na typu vyvolávané operace:

- Jednosměrný: Import JMS vloží zprávu do cíle operace odeslání nakonfigurovaného v dané vazbě importu. V poli replyTo záhlaví JMS není nastaveno nic.
- Obousměrný (požadavek-odezva): Import JMS vloží zprávu do cíle operace odeslání a poté trvale uloží odpověď, kterou obdrží od komponenty SCA.

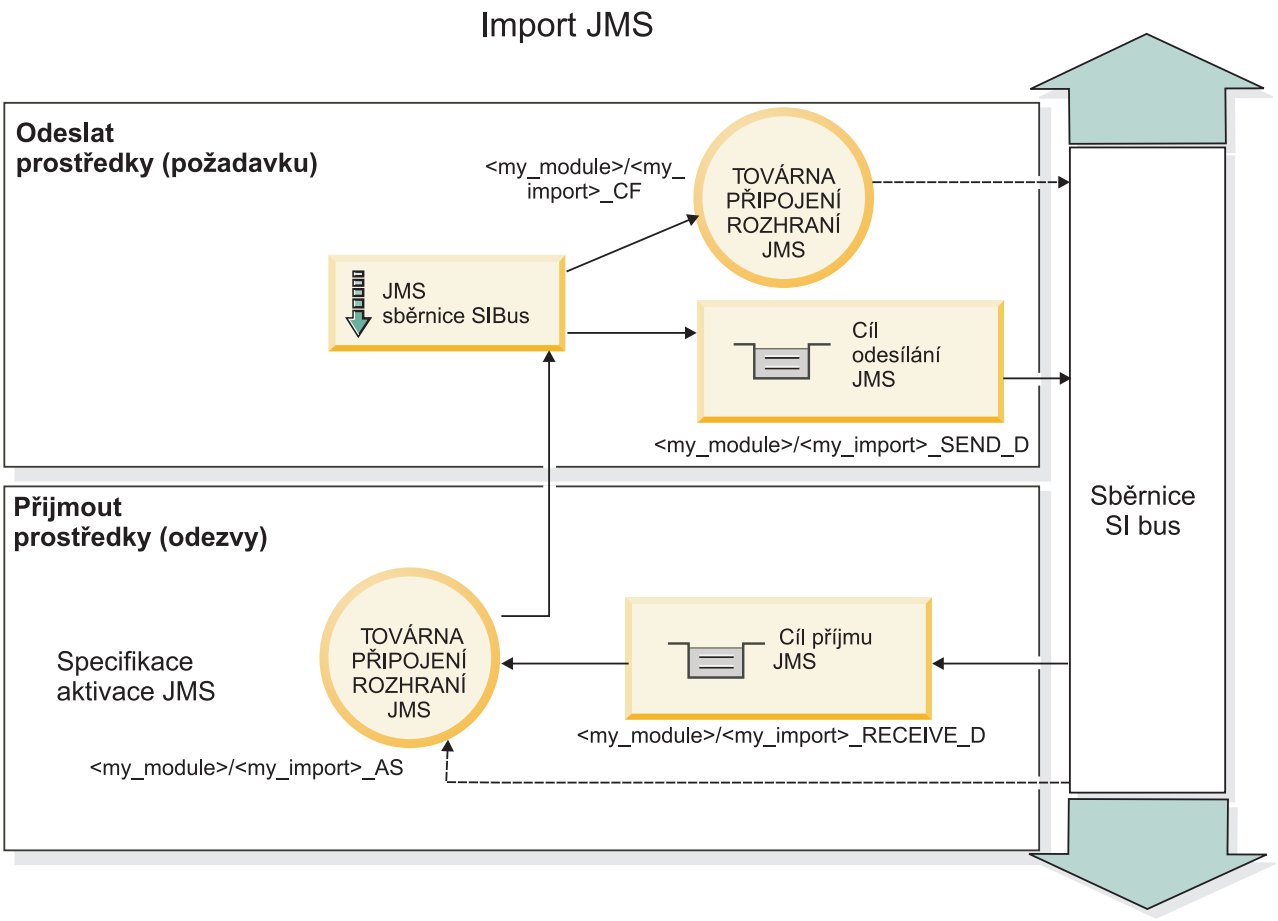
  Vazbu importu lze nakonfigurovat (pomocí pole **Schéma korelace odpovědí** v produktu Integration Designer), aby očekávala ID korelace zprávy odpovědi zkopírované z ID zprávy požadavku (výchozí nastavení), nebo z ID korelace zprávy požadavku. Vazbu importu lze také nakonfigurovat, aby ke korelaci odpovědí s požadavky používala dočasný cíl dynamických odpovědí. Je vytvořen dočasný cíl pro každý požadavek a import tento cíl používá pro příjem odpovědi.

  Cíl operace příjmu (receive) je nastaven ve vlastnosti záhlaví replyTo odchozí zprávy. Je implementován listener zpráv, aby naslouchal cíli operace příjmu (receive), a když je přijata odpověď, listener zpráv předá tuto odpověď zpět komponentě.

Pro jednosměrné i obousměrné scénáře použití lze určit dynamické a statické vlastnosti záhlaví. Statické vlastnosti lze nastavit z vazby metody importu JMS. Některé z těchto vlastností mají zvláštní význam pro běhové prostředí SCA JMS.

Je důležité uvědomit si, že JMS je asynchronní vazba. Pokud volající komponenta vyvolá import JMS synchronně (pro obousměrnou operaci), je volající komponenta zablokována, dokud služba JMS nevrátí odpověď.

Obrázek 32 na stránce 111 ukazuje, jak je import propojen s externí službou.



Obrázek 32. Prostředky vazby importu JMS

Vazby exportu JMS

Vazby exportu JMS zajišťují modulům SCA prostředky pro poskytování služeb externím aplikacím JMS.

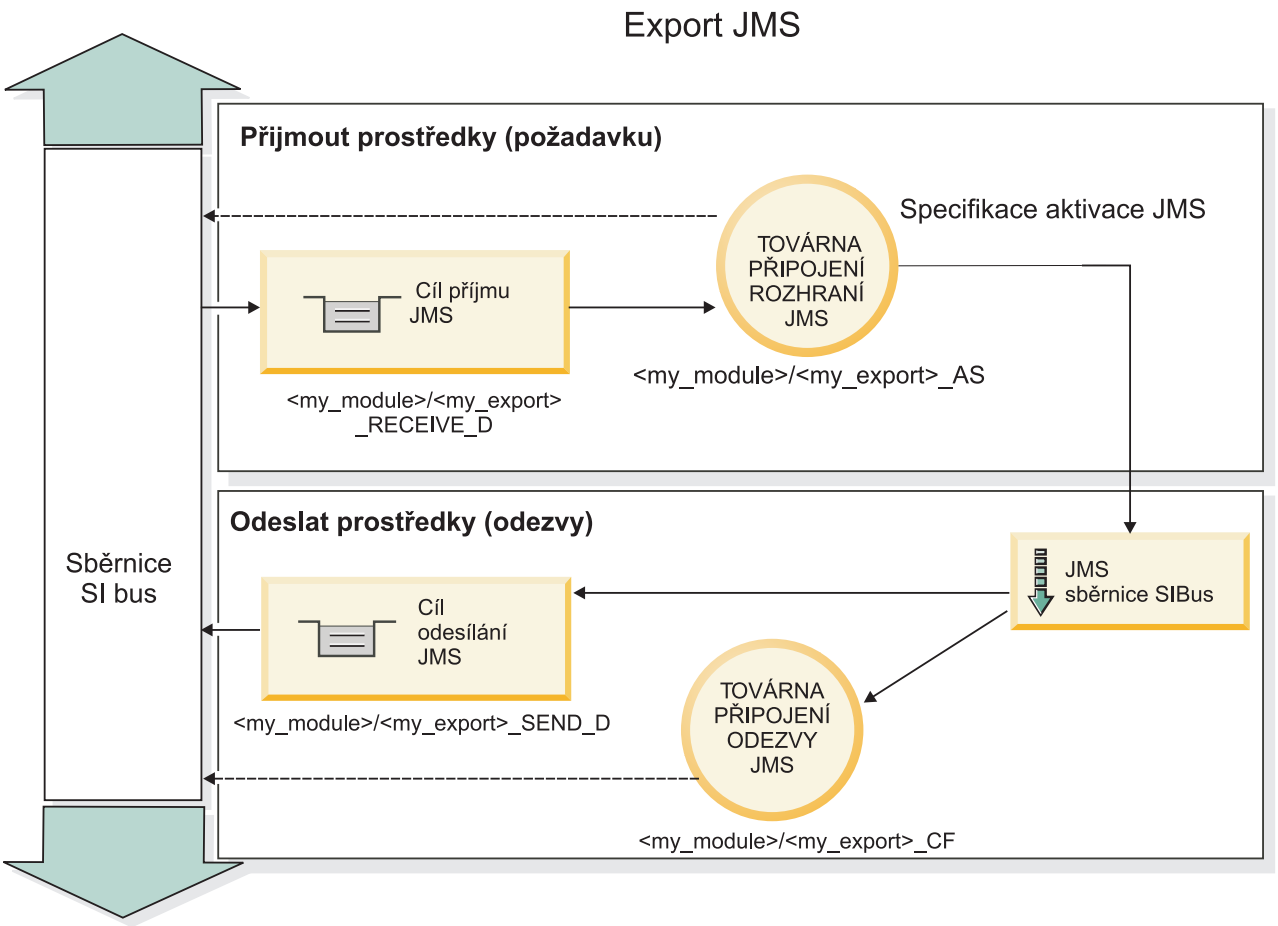
Připojení, které je částí exportu JMS je konfigurovatelnou specifikací aktivace.

Export JMS má cíle operací odeslání a příjmu (send a receive).

- Cíl operace příjmu (receive) je místo, kam se má vložit příchozí zpráva pro cílovou komponentu.
- Cíl operace odeslání (send) je místo, kam bude odeslána odpověď, pokud je příchozí zpráva nepotlačí pomocí vlastnosti záhlaví replyTo.

Listener zpráv je implementován, aby naslouchal požadavkům přicházejícím do cíle operace příjmu (receive) určeného ve vazbě exportu. Cíl určený v poli odeslání (send) se používá k odeslání odpovědi na příchozí požadavek, pokud vyvolaná komponenta poskytuje odpověď. Cíl určený v poli replyTo příchozí zprávy přepíše cíl určený v poli odeslání (send).

Obrázek 33 na stránce 112 ukazuje, jak je externí žadatel propojen s exportem.



Obrázek 33. Prostředky vazby exportu JMS

Záhlaví JMS:

Zpráva JMS obsahuje dva typy záhlaví: systémová záhlaví JMS a různé vlastnosti JMS. K oběma typům záhlaví lze přistupovat buď v mediačním modulu v objektu SMO (Service Message Object), nebo prostřednictvím rozhraní API ContextService.

Systémové záhlaví JMS

Systémové záhlaví JMS je v objektu SMO představováno prvkem JMSHeader, který obsahuje všechna pole, jež se obvykle vyskytují v záhlaví JMS. Ačkoli je lze upravit v mediaci (nebo službě ContextService), některá pole systémových záhlaví JMS nastavená v objektu SMO se nebudou šířit v odchozí zprávě JMS, protože jsou přepsána systémovými nebo statickými hodnotami.

Klíčová pole v systémovém záhlaví JMS, která lze aktualizovat v mediaci (nebo službě ContextService):

- **JMSType** a **JMSCorrelationID** – hodnoty specifických předdefinovaných vlastností záhlaví zprávy.
- **JMSDeliveryMode** – hodnoty způsobu doručení persistent (perzistentní) nebo non-persistent (dočasný); výchozí je persistent).
- **JMSPriority** – hodnota priority (0 až 9; výchozí je JMS_Default_Priority.)

Vlastnosti JMS

Vlastnosti JMS jsou v objektu SMO reprezentovány položkami seznamu Vlastnosti. Vlastnosti lze přidávat, aktualizovat nebo odstraňovat v mediaci nebo prostřednictvím rozhraní API ContextService.

Vlastnosti lze také nastavit staticky ve vazbě JMS. Vlastnosti, které jsou nastaveny staticky, přepisují nastavení (se stejným názvem), která jsou nastavena dynamicky.

Uživatelské vlastnosti šířené z jiných vazeb (například jako vazba HTTP) budou výstupem ve vazbě JMS jako vlastnosti JMS.

Nastavení šíření záhlaví

Šíření systémového záhlaví a vlastností JMS buď z příchozí zprávy JMS do následných komponent, nebo z předchozích komponent do odchozí zprávy JMS lze řídit příznakem Šířit záhlaví protokolů pro danou vazbu.

Když je nastaven příznak Šířit záhlaví protokolů, je informacím o záhlaví dovoleno plynout do zprávy nebo do cílové komponenty, jak popisuje následující seznam:

- Požadavek exportu JMS

  Záhlaví JMS přijaté ve zprávě bude šířeno do cílové komponenty prostřednictvím kontextové služby. Vlastnosti JMS přijaté ve zprávě budou šířeny do cílové komponenty prostřednictvím kontextové služby.
- Odpověď exportu JMS

  Všechna pole záhlaví JMS nastavená v kontextové službě budou použita v odchozí zprávě, pokud nejsou přepsána statickými vlastnostmi nastavenými pro vazbu exportu JMS. Všechny vlastnosti nastavené v kontextové službě budou použity v odchozí zprávě, pokud nejsou přepsány statickými vlastnostmi nastavenými pro vazbu exportu JMS.
- Požadavek importu JMS

  Všechna pole záhlaví JMS nastavená v kontextové službě budou použita v odchozí zprávě, pokud nejsou přepsána statickými vlastnostmi nastavenými pro vazbu importu JMS. Všechny vlastnosti nastavené v kontextové službě budou použity v odchozí zprávě, pokud nejsou přepsány statickými vlastnostmi nastavenými pro vazbu importu JMS.
- Odpověď importu JMS

  Záhlaví JMS přijaté ve zprávě bude šířeno do cílové komponenty prostřednictvím kontextové služby. Vlastnosti JMS přijaté ve zprávě budou šířeny do cílové komponenty prostřednictvím kontextové služby.

Schéma korelace dočasného dynamického cíle odpovědi JMS:

Schéma korelace dočasného dynamického cíle odpovědi způsobí vytvoření jedinečné dynamické fronty nebo tématu pro každý odeslaný požadavek.

Statický cíl odpovědi určený v importu se používá k odvození charakteru dočasné dynamické fronty nebo tématu cíle. To se nastavuje v poli **ReplyTo** požadavku a import JMS naslouchá odpovědím v tomto cíli. Když je přijata odpověď, je znovu zařazena do fronty pro statický cíl odpovědi pro asynchronní zpracování. Pole **CorrelationID** odpovědi se nepoužívá a není nutné je nastavovat.

Transakční problematika

Když se používá dočasný dynamický cíl, musí být odpověď využita ve stejném podprocesu jako odeslaná odpověď. Požadavek musí být odeslán mimo globální transakci a musí být potvrzen dřív, než bude přijat back-endovou službou a bude vrácena odpověď.

Perzistence

Dočasné dynamické fronty jsou krátkodobé entity a nezaručují stejnou úroveň perzistence, jaká se přidružuje ke statické frontě nebo tématu. Dočasná dynamická fronta nebo téma nepřežije restart serveru a totéž platí pro zprávy. Poté, co je zpráva nově zařazena do fronty pro statický cíl odpovědi, si zachová perzistenci, která je definovaná ve zprávě.

Časový limit

Import čeká na přijetí odpovědi v dočasném dynamickém cíli odpovědi po stanovenou dobu. Tento časový interval bude převzat z časového kvalifikátoru Vypršení platnosti odpovědi SCA, je-li nastaven, jinak je výchozí nastavení této doby 60 sekund. Pokud je překročena doba čekání, vrátí import výjimku ServiceTimeoutRuntimeException.

Externí klienti:

Server dokáže odesílat zprávy externím klientům pomocí vazeb JMS nebo od nich zprávy přijímat.

Externí klient (například webový portál nebo podnikový informační systém) může odeslat zprávu modulu SCA na serveru nebo může být vyvolán komponentou z tohoto serveru.

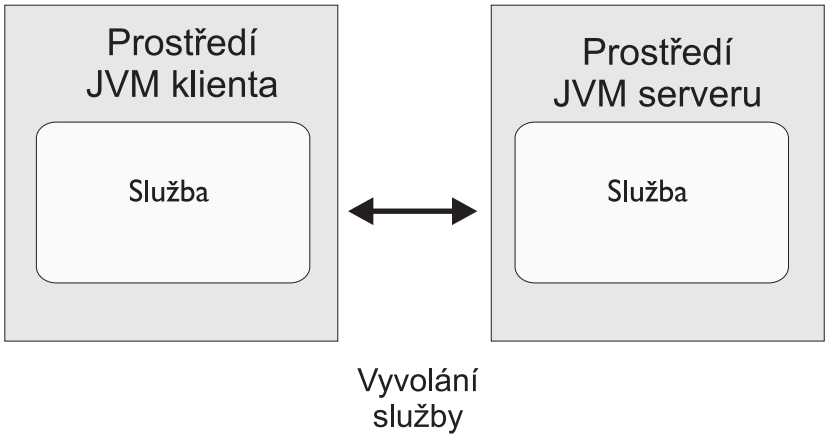
Komponenty exportu JMS implementují listenery zpráv, aby naslouchaly požadavkům přicházejícím do cíle příjmu určeného ve vazbě exportu. Cíl určený v poli odeslání se používá k odeslání odpovědi na příchozí požadavek, pokud vyvolaná aplikace nějakou odpověď poskytne. Proto je externí klient schopen vyvolávat aplikace s pomocí vazby exportu.

Importy JMS zajišťují interakci s externími klienty prostřednictvím zasílání zpráv do front JMS a příjmu zpráv z front JMS.

Práce s externími klienty:

Externí klient (tj. klient mimo server) může vyžadovat interakci s aplikací instalovanou na serveru.

Zvažte velmi jednoduchý scénář, ve kterém externí klient požaduje interakci s aplikací na serveru. Na obrázku je znázorněn typický jednoduchý scénář.



Obrázek 34. Jednoduchý scénář příkladu použití: Externí klient umožňuje interakci se serverovou aplikací

Aplikace SCA zahrnuje export s vazbou JMS; tím je aplikace zpřístupněna pro externí klienty.

Když máte externího klienta v prostředí JVM (Java Virtual Machine) odděleném od serveru, je třeba pro připojení a interakci s exportem JMS provést několik kroků. Klient získá objekt InitialContext se správnými hodnotami a poté vyhledá prostředky pomocí rozhraní JNDI. Poté klient prostřednictvím klienta specifikace JMS 1.1 přistupuje k cílům a odesílá a přijímá zprávy v cílech.

Výchozí názvy rozhraní JNDI prostředků automaticky vytvořených běhovým prostředím jsou uvedeny v tématu o konfiguraci v tomto oddílu. Máte-li však předem vytvořené prostředky, použijte tyto názvy rozhraní JNDI.

1. Nakonfigurujte místa určení JMS a továrnu připojení k odeslání zprávy.

2. Přesvědčte se, že kontext rozhraní JNDI, port pro adaptér prostředku SIB i port samozavedení systému zpráv jsou správné.

   Server používá některé výchozí porty, ale pokud je na daném systému instalováno více serverů, jsou při instalaci vytvořeny alternativní porty, aby se předešlo konfliktům s dalšími instancemi serveru. Pomocí administrativní konzoly můžete určit, které porty server používá. Přejděte na **Servery > Aplikační servery > *název_vašeho_serveru* > Konfigurace** a klepněte na volbu **Porty** pod volbou **Komunikace**. Poté můžete upravit používaný port.
3. Klient obdrží počáteční kontext se správnými hodnotami a poté vyhledá prostředky prostřednictvím rozhraní JNDI.
4. Klient pomocí specifikací JMS 1.1 přistupuje k cílům a odesílá a přijímá zprávy v cílech.

Odstraňování problémů s vazbami služby JMS:

Problémy s vazbami služby JMS lze diagnostikovat a opravit.

Výjimky implementace

Jako odpověď na různé chybové stavy může implementace importu a exportu JMS vracet jeden ze dvou typů výjimek:

- Obchodní výjimka služby: Tato výjimka je vrácena, pokud došlo k poruše určené na obchodním rozhraní služby (typ portu WSDL).
- Výjimka za běhu služby: Generuje se ve všech ostatních případech. Ve většině případů bude výjimka příčina obsahovat původní výjimku (JMSException).

  Například import očekává pouze jednu zprávu odezvy na každou zprávu požadavku. Pokud dorazí více než jedna odezva nebo pokud dorazí pozdní odezva (odezva, pro kterou došlo k vypršení odezvy SCA), je vyvolána výjimka za běhu služby. Transakce je odvolána a zpráva odezvy je vrácena zpět z fronty nebo zpracována správcem událostí se selháním.

Primární podmínky selhání

Primární podmínky selhání vazeb JMS jsou určeny transakční sémantikou, konfigurací poskytovatele JMS nebo odkazem na existující chování v jiných komponentách. Primární podmínky selhání zahrnují:

- Selhání při připojení k poskytovateli JMS nebo k místu určení.

  Selhání při připojení k poskytovateli JMS při přijímání zpráv bude mít za následek selhání spuštění listeneru zpráv. Tato podmínka bude zaznamenána do protokolu produktu WebSphere Application Server. Trvalé zprávy zůstanou v místě určení, dokud nebudou úspěšně načteny (nebo dokud nevyprší jejich platnost).

  Selhání při připojení k poskytovateli JMS při odesílání odchozích zpráv způsobí odvolání transakce, která řídí odeslání.
- Selhání při analýze příchozí zprávy nebo při vytvoření odchozí zprávy.

  Selhání ve vázání dat nebo popisovači dat způsobí odvolání transakce, která tuto práci řídí.
- Selhání při odesílání odchozí zprávy.

  Selhání při odesílání zprávy způsobí odvolání příslušné transakce.
- Vícenásobné nebo neočekávané zprávy o pozdní odezvě.

  Import očekává pouze jednu zprávu odezvy na každou zprávu požadavku. Kvalifikátor vypršení odezvy SCA u požadavku také určuje platné časové období, během kterého může být odezva přijata. Když odezva dorazí nebo je překročen čas vypršení platnosti, záznam korelace je odstraněn. Pokud dorazí zprávy odezvy neočekávaně nebo pozdě, dojde k výjimce za běhu služby.
- Výjimka vypršení časového limitu za běhu služby způsobená pozdní odezvou při použití schématu korelace cíle dočasné dynamické odezvy.

  Importu JMS vyprší časový limit po uplynutí časového úseku, který je určen kvalifikátorem vypršení odezvy SCA, nebo není-li tento kvalifikátor nastaven, použije se standardně časový úsek 60 sekund.

Zprávy SCA založené na JMS se nezobrazují ve správci událostí se selháním

Pokud dojde k selhání zpráv SCA vytvořených prostřednictvím interakce JMS, měly by se tyto zprávy objevit ve správci událostí se selháním. Pokud se tyto zprávy ve správci událostí se selháním neobjeví, zkontrolujte, zda má základní místo určení SIB místa určení JMS hodnotu maximálního počtu nezdařených doručení větší než **1**. Nastavení této hodnoty na **2** a více umožní interakci se správcem událostí se selháním během vyvolání SCA pro vazby JMS.

Zpracování výjimek:

Způsob, jakým je vazba nakonfigurována, určuje, jak jsou zpracovány výjimky, které jsou způsobeny popisovači dat nebo vázáními dat. Chování systému při vrácení takové výjimky je dále určeno také charakterem mediačního toku.

Když vaše vazba volá popisovač dat nebo vázání dat, může dojít k nejrůznějším problémům. Například může popisovač dat obdržet zprávu s poškozeným informačním obsahem nebo se může pokusit přečíst zprávu v nesprávném formátu.

Způsob, jakým vaše vazba ošetří takovou výjimku, je určen tím, jak implementujete daný popisovač dat nebo vázání dat. Doporučené chování je navrhnout vázání dat tak, aby vracelo výjimku **DataBindingException**.

Když je vrácena jakákoli výjimka za běhu, včetně výjimky **DataBindingException**:

- Pokud je mediační tok nakonfigurován jako transakční, je zpráva JMS standardně uložena do komponenty Failed Event Manager pro ruční přehrání nebo odstranění.

  Poznámka: Režim zotavení pro vazbu je možné změnit tak, aby byla zpráva namísto uložení do komponenty Failed Event Manager odvolána.
- Pokud mediační tok není transakční, je výjimka zaprotokolována a zpráva je ztracena.

Podobná situace je i u popisovače dat. Protože je popisovač dat vyvolán vázáním dat, je jakákoli výjimka popisovače dat zabalena do výjimky vázání dat. Proto je výjimka **DataHandlerException** hlášena jako **DataBindingException**.

Generické vazby JMS:

Generická vazba služby JMS zajišťuje konektivitu pro poskytovatele jiných dodavatelů odpovídající standardu JMS 1.1. Činnost Generické vazby služby JMS je podobná činnosti vazeb JMS.

Služba poskytovaná prostřednictvím JMS umožňuje modulu SCA (Service Component Architecture) provádět volání a přijímat zprávy z externích systémů. Takový systém může být externím systémem JMS.

Generická vazba služby JMS zajišťuje integraci s poskytovateli JMS nekompatibilními s JCA 1.5, kteří podporují JMS 1.1 a implementují volitelný mechanizmus JMS Application Server Facility. Generická vazba služby JMS podporuje poskytovatele JMS (včetně Oracle AQ, TIBCO, SonicMQ, WebMethods a BEA WebLogic), kteří nepodporují JCA 1.5, ale podporují ASF (Application Server Facility) podle specifikace JMS 1.1. Vestavěný poskytovatel WebSphere JMS (SIBJMS), což je poskytovatel JMS JCA 1.5, není touto vazbou podporován. Při použití tohoto poskytovatele použijte “Vazby JMS” na stránce 108.

Tuto Generickou vazbu použijte při integraci se systémem JMS nekompatibilním s JCA 1.5 v rámci prostředí SCA. Cílové externí aplikace pak mohou přijímat zprávy a odesílat zprávy za účelem integrace s komponentou SCA.

Přehled generických vazeb služby JMS:

Generické vazby JMS jsou vazby JMS nekompatibilní s JCA, které zajišťují konektivitu mezi prostředím SCA (Service Component Architecture) a systémy JMS kompatibilními s JMS 1.1 a implementujícími volitelný mechanizmus JMS Application Server Facility.

Generické vazby JMS

Hlavní aspekty generických vazeb importu a exportu služby JMS jsou:

- Port modulu listener: umožňuje poskytovatelům JMS nekompatibilním s JCA přijímat zprávy a odesílat je objektu typu message-driven bean (MDB).
- Připojení: zapouzdřují virtuální připojení mezi klientem a aplikací poskytovatele.
- Cíle: používány klientem k určení cíle jím produkovaných zpráv nebo zdroje jím přijímaných zpráv.
- Data ověřování: používána k zabezpečení přístupu k vazbě.

Generické vazby importu služby JMS

Generické vazby importu JMS umožňují komponentám v rámci vašeho modulu SCA komunikovat se službami poskytovanými externími poskytovateli JMS nekompatibilními s JCA 1.5.

Část připojení importu JMS je továrna připojení. Továrna připojení, objekt používaný klientem k vytvoření připojení k poskytovateli, zapouzdřuje sadu konfiguračních parametrů připojení definovaných administrátorem. Každá továrna připojení je instancí rozhraní ConnectionFactory, QueueConnectionFactory nebo TopicConnectionFactory.

Interakce s externími systémy JMS zahrnují použití cílů pro odesílání požadavků a příjem odpovědí.

Jsou podporovány dva typy scénářů využití generických vazeb importu JMS, v závislosti na typu vyvolávané operace:

- Jednosměrný: Generický import JMS vloží zprávu do cíle operace odeslání nakonfigurovaného v dané vazbě importu. Do pole replyTo v záhlaví JMS se nic nepošle.
- Obousměrný (požadavek-odezva): Generický import JMS vloží zprávu do cíle operace odeslání a poté trvale uloží odpověď, kterou obdrží od komponenty SCA.

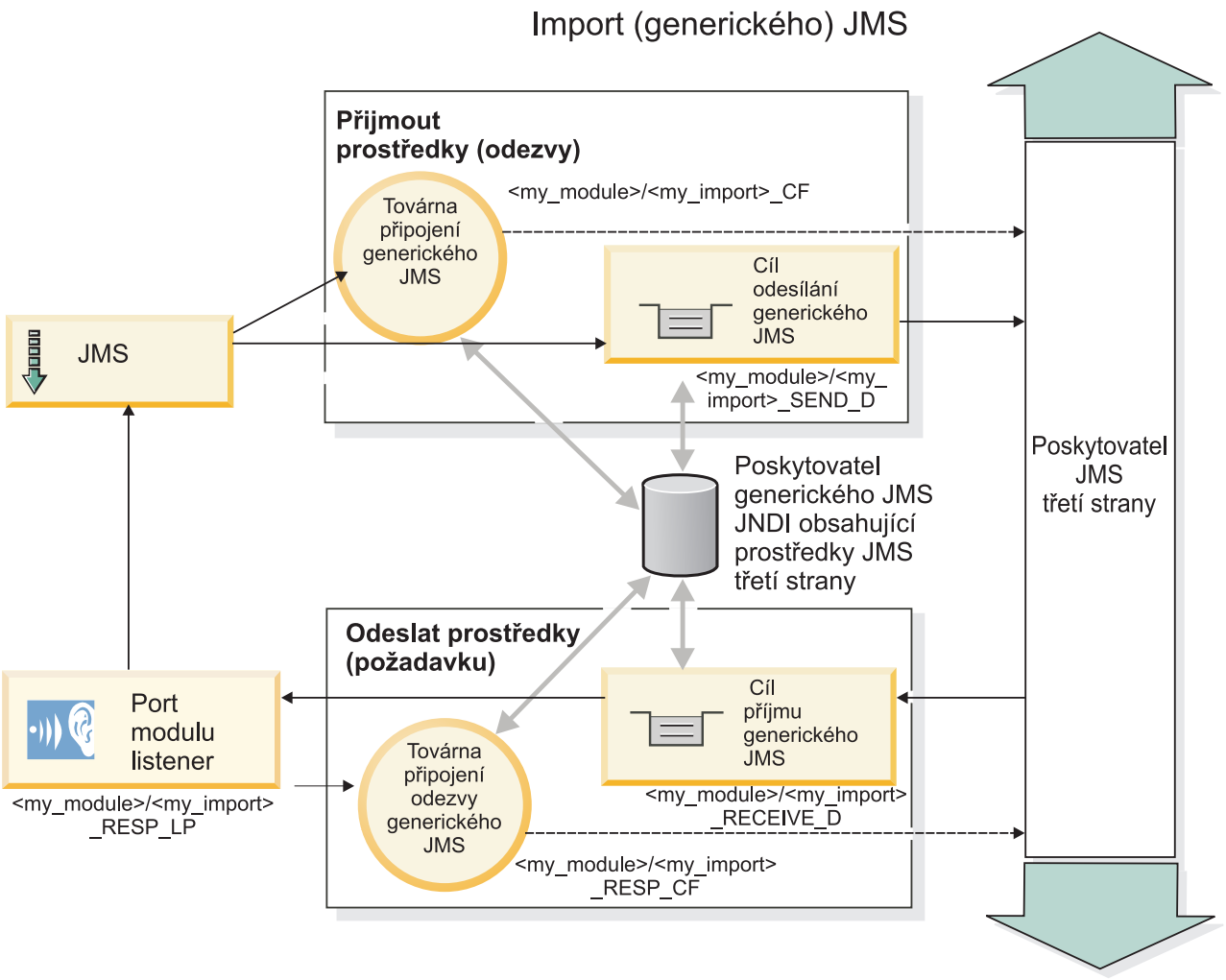
  Cíl operace příjmu (receive) je nastaven ve vlastnosti záhlaví replyTo odchozí zprávy. Je implementován objekt typu message-driven bean (MDB), aby naslouchal cíli operace příjmu (receive), a když je přijata odpověď, MDB předá tuto odpověď zpět komponentě.

  Vazbu importu lze nakonfigurovat (pomocí pole **Schéma korelace odpovědí** v produktu Integration Designer), aby očekávala ID korelace zprávy odpovědi zkopírované z ID zprávy požadavku (výchozí nastavení), nebo z ID korelace zprávy požadavku.

Pro jednosměrné i obousměrné scénáře použití lze určit dynamické a statické vlastnosti záhlaví. Statické vlastnosti lze nastavit z generické vazby metody importu JMS. Některé z těchto vlastností mají zvláštní význam pro běhové prostředí SCA JMS.

Je důležité uvědomit si, že Generická platforma JMS je asynchronní vazba. Pokud volající komponenta vyvolá Generický import JMS synchronně (pro obousměrnou operaci), je volající komponenta zablokována, dokud služba JMS nevrátí odpověď.

Obrázek 35 na stránce 118 ukazuje, jak je import propojen s externí službou.



Obrázek 35. Prostředky generické vazby importu JMS

Generické vazby exportu JMS

Generické vazby exportu JMS zajišťují modulům SCA prostředky pro poskytování služeb externím aplikacím JMS.

Část připojení exportu JMS tvoří továrna ConnectionFactory a port ListenerPort.

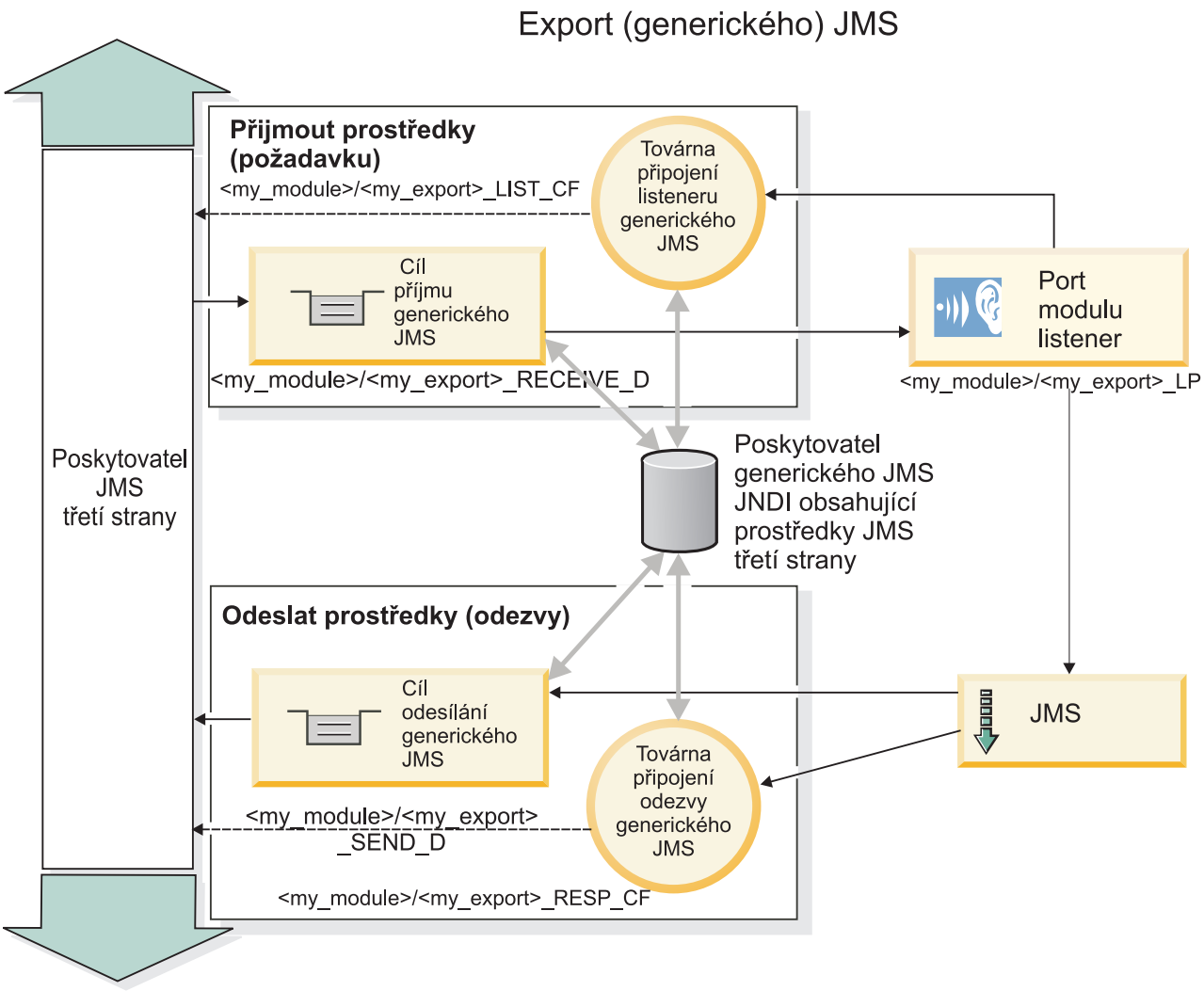
Generický export JMS má cíle operací odeslání a příjmu (send a receive).

- Cíl operace příjmu (receive) je místo, kam se má vložit příchozí zpráva pro cílovou komponentu.
- Cíl operace odeslání (send) je místo, kam bude odeslána odpověď, pokud je příchozí zpráva nepotlačí pomocí vlastnosti záhlaví replyTo.

Je implementován MDB, aby naslouchal požadavkům přicházejícím do cíle operace příjmu (receive) určeného ve vazbě exportu.

- Cíl určený v poli odeslání (send) se používá k odeslání odpovědi na příchozí požadavek, pokud vyvolaná komponenta nějakou odpověď poskytne.
- Cíl určený v poli replyTo příchozí zprávy přepíše cíl určený v poli odeslání (send).
- Pro scénáře požadavku/odpovědi může být nakonfigurována vazba importu (pomocí pole **Schéma korelace odpovědi** v produktu Integration Designer), aby očekávala, že odpověď zkopíruje ID zprávy požadavku do pole ID korelace odpovědi (výchozí nastavení), nebo může odpověď zkopírovat ID korelace požadavku do pole ID korelace zprávy odpovědi.

Obrázek 36 ukazuje, jak je externí žadatel propojen s exportem.



Obrázek 36. Prostředky generické vazby exportu JMS

Klíčové faktory vazby Generických vazeb služby JMS:

Faktory Generické vazby importu a export služby JMS jsou konzistentní s vestavěnými vazbami importu služby JMS MQ a JMS produktu WebSphere. Mezi klíčové faktory patří definice záhlaví a přístup k existujícím prostředkům Java EE. Avšak z důvodu jejich generického charakteru neexistují žádné možnosti připojení specifické pro poskytovatele služby a tato vazba má omezenou schopnost generovat prostředky při implementaci a instalaci.

Generické importy

Jako aplikace importu služby MQ JMS je Generická implementace služby JMS asynchronní a podporuje tři vyvolání: jednosměrné, obousměrné (rovněž známé jako požadavek-odezva) a zpětné volání.

Když je implementován import služby JMS, je implementován objekt typu message-driven bean (MDB) poskytovaný běhovým prostředím. Objekt MDB naslouchá odpovědím na zprávu požadavku. Objekt MDB je přidružen k (naslouchá) cíli zaslanému s požadavkem v poli záhlaví replyTo zprávy služby JMS.

Generické exporty

Generická vazba služby JMS se liší od vazeb exportu EIS v tom, jak pracují s návratem výsledku. Generický export služby JMS explicitně zasílá odezvu na cíl replyTo určený v příchozí zprávě. Pokud není určen, je použit cíl odeslání.

Když je implementován Generický export služby JMS, je implementován objekt typu message-driven bean (jiný objekt typu MDB než byl ten, jenž byl použit pro Generické importy systému JMS). Naslouchá příchozím požadavkům na cíli příjmu a poté odbavuje požadavky, aby byly zpracovány běhovým prostředím SCA.

Speciální záhlaví

Vlastnosti speciálního záhlaví se používají v Generických v importech a exportech JMS, aby bylo možné cíli vazby říct, jak zpracovat zprávu.

Vlastnost TargetFunctionName používá výchozí selektor funkcí k identifikaci názvu vlastnosti v rozhraní exportu, které je vyvoláváno.

Poznámka: Vazbu importu lze konfigurovat tak, aby nastavilo záhlaví TargetFunctionName na názvy jednotlivých operací.

Prostředky Java EE

Mnoho prostředků Java EE je vytvářeno, když je vazba JMS implementována do prostředí Java EE.

- Port modulu listener v cíli příjmu (odezvy, pouze obousměrný) pro importy a v cíli příjmu (požadavku) pro exporty.
- Generická továrna připojení JMS pro odchozí připojení (import) a příchozí připojení (export).
- Generické cíle JMS pro cíle odesílání (import) a přijímání (export, pouze obousměrný).
- Generická továrna připojení JMS pro připojení odezvy (pouze obousměrné a volitelné, jinak je pro importy používáno odchozí připojení a pro exporty je používáno příchozí připojení).
- Generický cíl JMS pro cíle příjmu (import) a odesílání (export, pouze obousměrné).
- Výchozí cíl JMS zpětného volání poskytovatele systému zpráv použitý k přístupu k cíli fronty zpětného volání SIB (pouze obousměrné).
- Výchozí továrna připojení JMS zpětného volání poskytovatele systému zpráv (pouze obousměrné).
- Cíl fronty zpětného volání SIB k uložení informací o zprávě požadavku pro použití během zpracování odezvy (pouze obousměrné)

Instalační úloha vytváří Továrnu připojení, tři cíle a Specifikace aktivace z informací v souborech importu a exportu.

Generická záhlaví JMS:

Generická záhlaví JMS jsou objekty SDO (Service Data Objects), které obsahují všechny vlastnosti vlastností Generických zpráv JMS. Tyto vlastnosti mohou pocházet z příchozí zprávy nebo se může jednat o vlastnosti, které budou použity na odchozí zprávu.

Zpráva JMS obsahuje dva typy záhlaví: systémová záhlaví JMS a různé vlastnosti JMS. K oběma typům záhlaví lze přistupovat buď v mediačním modulu v objektu SMO (Service Message Object), nebo prostřednictvím rozhraní API ContextService.

Následující vlastnosti jsou staticky nastaveny na vazbě methodBinding:

- JMSType.
- JMSCorrelationID.
- JMSDeliveryMode.
- JMSPriority.

Generická vazba JMS podporuje také dynamické úpravy záhlaví a vlastností JMS stejným způsobem jako u vazeb JMS a MQ JMS.

Někteří Generičtí poskytovatelé JMS omezují vlastnosti, které může aplikace nastavit, a jejich kombinace. Další informace viz dokumentace příslušných produktů jiných dodavatelů. K vazbě methodBinding, však byla přidána doplňková vlastnost ignoreInvalidOutboundJMSProperties, která umožňuje šíření libovolných výjimek.

Generická záhlaví JMS a vlastnosti zpráv se používají, jen když je zapnutý přepínač vazby SCDL architektury SCA. Když je tento přepínač zapnutý, jsou šířeny informace o kontextu. Standardně je tento přepínač zapnutý. Chcete-li zabránit šíření informací o kontextu, změňte hodnotu na **false**.

Když je povoleno šíření kontextu, smí informace ze záhlaví plynout do zprávy nebo do cílové komponenty. Chcete-li zapnout či vypnout šíření kontextu, zadejte pro atribut contextPropagationEnabled vazeb importu a exportu hodnotu **true** nebo **false**. Například:

```
<esbBinding xsi:type="eis:JMSImportBinding" contextProgagationEnabled="true">
```

Výchozí hodnota je **true**.

Odstraňování problémů s generickými vazbami služby JMS:

Problémy s generickými vazbami služby JMS lze diagnostikovat a opravit.

Výjimky implementace

Jako odpověď na různé chybové stavy může generická implementace importu a exportu JMS vracet jeden ze dvou typů výjimek:

- Obchodní výjimka služby: Tato výjimka je vrácena, pokud došlo k poruše určené na obchodním rozhraní služby (typ portu WSDL).
- Výjimka za běhu služby: Generuje se ve všech ostatních případech. Ve většině případů bude výjimka příčina obsahovat původní výjimku (JMSException).

Odstraňování problémů s vypršením generické zprávy JMS

Zpráva požadavku poskytovatele JMS podléhá vypršení platnosti.

Vypršení platnosti požadavku odkazuje na vypršení platnosti zprávy požadavku poskytovatele JMS při dosažení času JMSExpiration na zprávě požadavku. Stejně jako u jiných vazeb JMS zpracovává generická vazba JMS vypršení platnosti požadavku tím, že na zprávě zpětného volání podané importem nastaví stejné vypršení platnosti jako pro odchozí požadavek. Oznámení k vypršení platnosti zprávy zpětného volání bude indikovat, že došlo k vypršení platnosti zprávy požadavku a klient by měl být upozorněn prostřednictvím obchodní výjimky.

Při přesunutí cíle zpětného volání k poskytovateli jiného dodavatele však není tento typ vypršení platnosti požadavku podporován.

Vypršení platnosti odezvy odkazuje na vypršení platnosti zprávy odezvy poskytovatele JMS při dosažení času JMSExpiration na zprávě odezvy.

Vypršení platnosti odezvy není pro generickou vazbu JMS podporováno, protože není definováno přesné chování poskytovatele JMS jiného dodavatele při vypršení platnosti. Můžete však zkontrolovat, zda nedošlo k vypršení platnosti odezvy při jejím přijetí.

Pro odchozí zprávy požadavku se hodnota JMSExpiration vypočítá z času čekání a z hodnot requestExpiration v asyncHeader, jsou-li nastaveny.

Odstraňování problémů s chybami generické továrny připojení JMS

Když definujete určité typy továren připojení v generickém poskytovateli JMS, můžete při pokusu o spuštění aplikace obdržet chybovou zprávu. Abyste předešli tomuto problému, můžete upravit externí továrnu připojení.

Při spuštění aplikace můžete obdržet následující chybovou zprávu:

```
MDB Listener Port JMSConnectionFactory type does not match
JMSDestination type (Typ JMSConnectionFactory portu modulu listener MDB neodpovídá typu JMSDestination)
```

K tomuto problému může dojít, když definujete externí továrny připojení. Konkrétně může k výjimce dojít, když vytvoříte továrnu připojení tématu JMS 1.0.2 místo (unifikované) továrny připojení JMS 1.1 (tj. továrny, která může podporovat jak komunikaci Odesilatel-příjemce, tak komunikaci publikování-odběru).

K vyřešení tohoto problému postupujte takto:

1. Přejděte na generického poskytovatele JMS, kterého používáte.
2. Nahraďte továrnu připojení tématu JMS 1.0.2, kterou jste definovali, (unifikovanou) továrnou připojení JMS 1.1.

Když spustíte aplikaci s nově definovanou továrnou připojení JMS 1.1, již byste neměli obdržet chybovou zprávu.

Generické zprávy SCA založené na JMS se nezobrazují ve správci událostí se selháním

Pokud dojde k selhání zpráv SCA vytvořených prostřednictvím generické interakce JMS, měly by se tyto zprávy objevit ve správci událostí se selháním. Pokud se tyto zprávy ve správci událostí se selháním neobjeví, zkontrolujte, zda je hodnota vlastnosti maximálního počtu pokusů základního portu modulu listener rovna nebo větší než 1. Nastavení této hodnoty na 1 a více umožní interakci se správcem událostí se selháním během vyvolání SCA pro generické vazby JMS.

Zpracování výjimek:

Způsob, jakým je vazba nakonfigurována, určuje, jak jsou zpracovány výjimky, které jsou způsobeny popisovači dat nebo vázáními dat. Chování systému při vrácení takové výjimky je dále určeno také charakterem mediačního toku.

Když vaše vazba volá popisovač dat nebo vázání dat, může dojít k nejrůznějším problémům. Například může popisovač dat obdržet zprávu s poškozeným informačním obsahem nebo se může pokusit přečíst zprávu v nesprávném formátu.

Způsob, jakým vaše vazba ošetří takovou výjimku, je určen tím, jak implementujete daný popisovač dat nebo vázání dat. Doporučené chování je navrhnout vázání dat tak, aby vracelo výjimku **DataBindingException**.

Podobná situace je i u popisovače dat. Protože je popisovač dat vyvolán vázáním dat, je jakákoli výjimka popisovače dat zabalena do výjimky vázání dat. Proto je výjimka **DataHandlerException** hlášena jako **DataBindingException**.

Když je vrácena jakákoli výjimka za běhu, včetně výjimky **DataBindingException**:

- Pokud je mediační tok nakonfigurován jako transakční, je zpráva JMS standardně uložena do komponenty Failed Event Manager pro ruční přehrání nebo odstranění.

  Poznámka: Režim zotavení pro vazbu je možné změnit tak, aby byla zpráva namísto uložení do komponenty Failed Event Manager odvolána.
- Pokud mediační tok není transakční, je výjimka zaprotokolována a zpráva je ztracena.

Podobná situace je i u popisovače dat. Protože je popisovač dat volán vázáním dat, je výjimka popisovače dat vytvořena uvnitř výjimky vázání dat. Proto je výjimka **DataHandlerException** hlášena jako **DataBindingException**.

Vazby WebSphere MQ JMS:

Vazba JMS WebSphere MQ zajišťuje integraci s externími aplikacemi, které používají poskytovatele na bázi WebSphere MQ JMS.

Pomocí vazeb exportu a importu produktu WebSphere MQ JMS můžete zajistit integraci s externími systémy JMS nebo MQ JMS přímo ze svého prostředí serveru. Tím odpadá nutnost používat funkce Propojení MQ nebo Propojení klienta sběrnice SIBus.

Když vstoupí komponenta do interakce se službou založenou na WebSphere MQ JMS prostřednictvím importu, vazba importu WebSphere MQ JMS využije cíl, kam budou odeslána data, a cíl, kde je možné přijmout odpověď. Převod dat do zprávy JMS i z ní se provádí prostřednictvím komponenty Edge popisovače dat nebo vazby dat JMS.

Když modul SCA poskytuje službu klientům WebSphere MQ JMS, používá vazba exportu WebSphere MQ JMS cíl, kde je možné přijmout požadavek a odeslat odpověď. Převod dat do zprávy JMS i z ní se provádí prostřednictvím popisovače dat nebo vázáním dat JMS.

Selektor funkcí zajišťuje mapování na operaci na cílové komponentě, která má být vyvolána.

Přehled vazeb WebSphere MQ JMS:

Vazba WebSphere MQ JMS zajišťuje integraci s externími aplikacemi, které používají poskytovatele WebSphere MQ JMS.

Administrativní úlohy WebSphere MQ

Od administrátora systému WebSphere MQ se očekává, že vytvoří základní produkt WebSphere MQ Queue Manager, který budou používat vazby WebSphere MQ JMS, před spuštěním aplikace obsahující tyto vazby.

Vazby importu produktu WebSphere MQ JMS

Import produktu WebSphere MQ JMS umožňuje komponentám ve vašem modulu SCA komunikovat se službami poskytovanými poskytovateli na bázi WebSphere MQ JMS. Musíte používat podporovanou verzi produktu WebSphere MQ. Podrobné požadavky na hardware a software naleznete na stránkách podpory IBM.

Jsou podporovány dva typy scénářů využití vazeb importu produktu WebSphere MQ JMS, v závislosti na typu vyvolávané operace:

- Jednosměrný: Import WebSphere MQ JMS vloží zprávu do cíle operace odeslání nakonfigurovaného v dané vazbě importu. Do pole replyTo v záhlaví JMS se nic nepošle.
- Obousměrný (požadavek-odezva): Import WebSphere MQ JMS vloží zprávu do cíle operace odeslání.

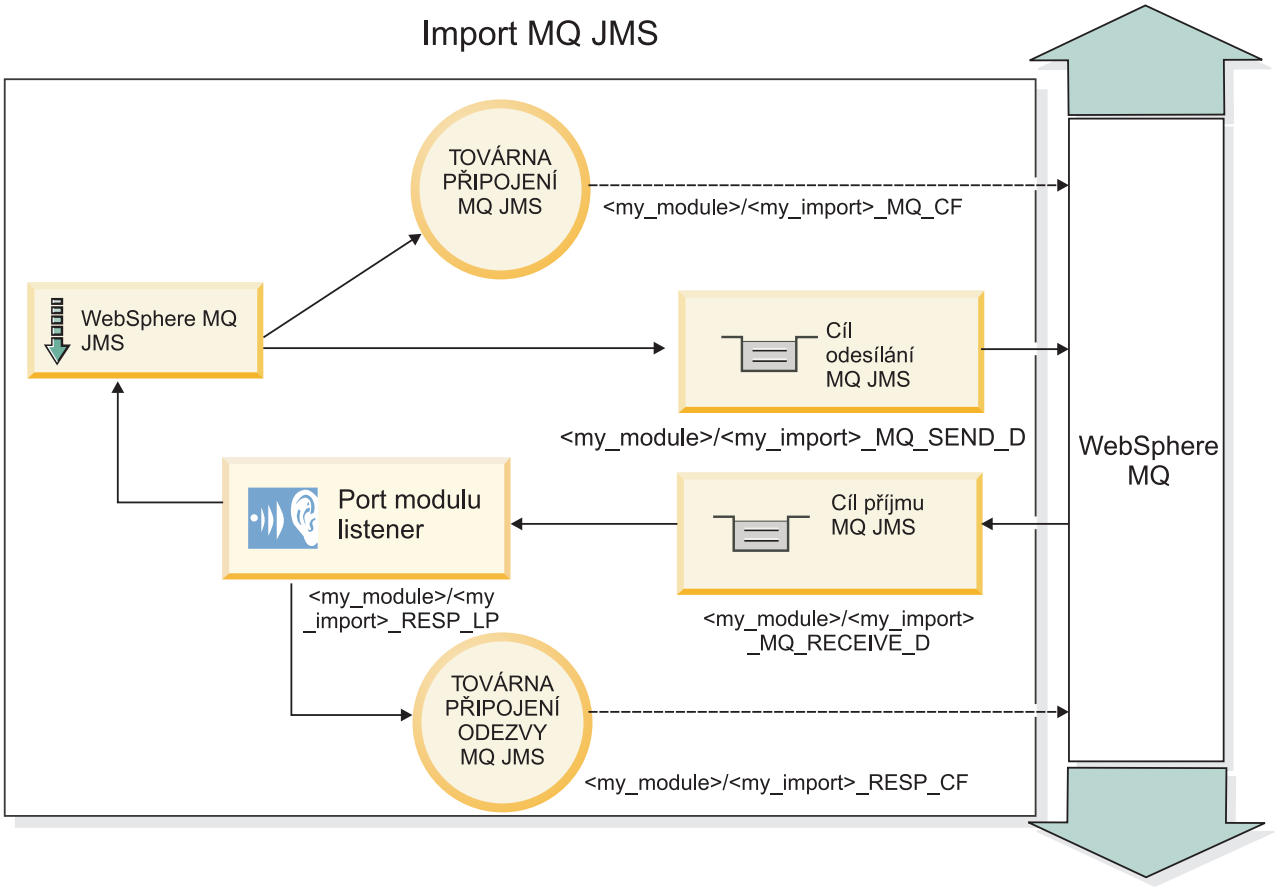
  Cíl operace příjmu (receive) je nastaven v poli záhlaví replyTo. Je implementován objekt typu message-driven bean (MDB), aby naslouchal cíli operace příjmu (receive), a když je přijata odpověď, MDB předá tuto odpověď zpět komponentě.

  Vazbu importu lze nakonfigurovat (pomocí pole **Schéma korelace odpovědí** v produktu Integration Designer), aby očekávala ID korelace zprávy odpovědi zkopírované z ID zprávy požadavku (výchozí nastavení), nebo z ID korelace zprávy požadavku.

Pro jednosměrné i obousměrné scénáře použití lze určit dynamické a statické vlastnosti záhlaví. Statické vlastnosti lze nastavit z vazby metody importu JMS. Některé z těchto vlastností mají zvláštní význam pro běhové prostředí SCA JMS.

Je důležité uvědomit si, že WebSphere MQ JMS je asynchronní vazba. Pokud volající komponenta vyvolá import produktu WebSphere MQ JMS synchronně (pro obousměrnou operaci), je volající komponenta zablokována, dokud služba JMS nevrátí odpověď.

Obrázek 37 na stránce 124 ukazuje, jak je import propojen s externí službou.



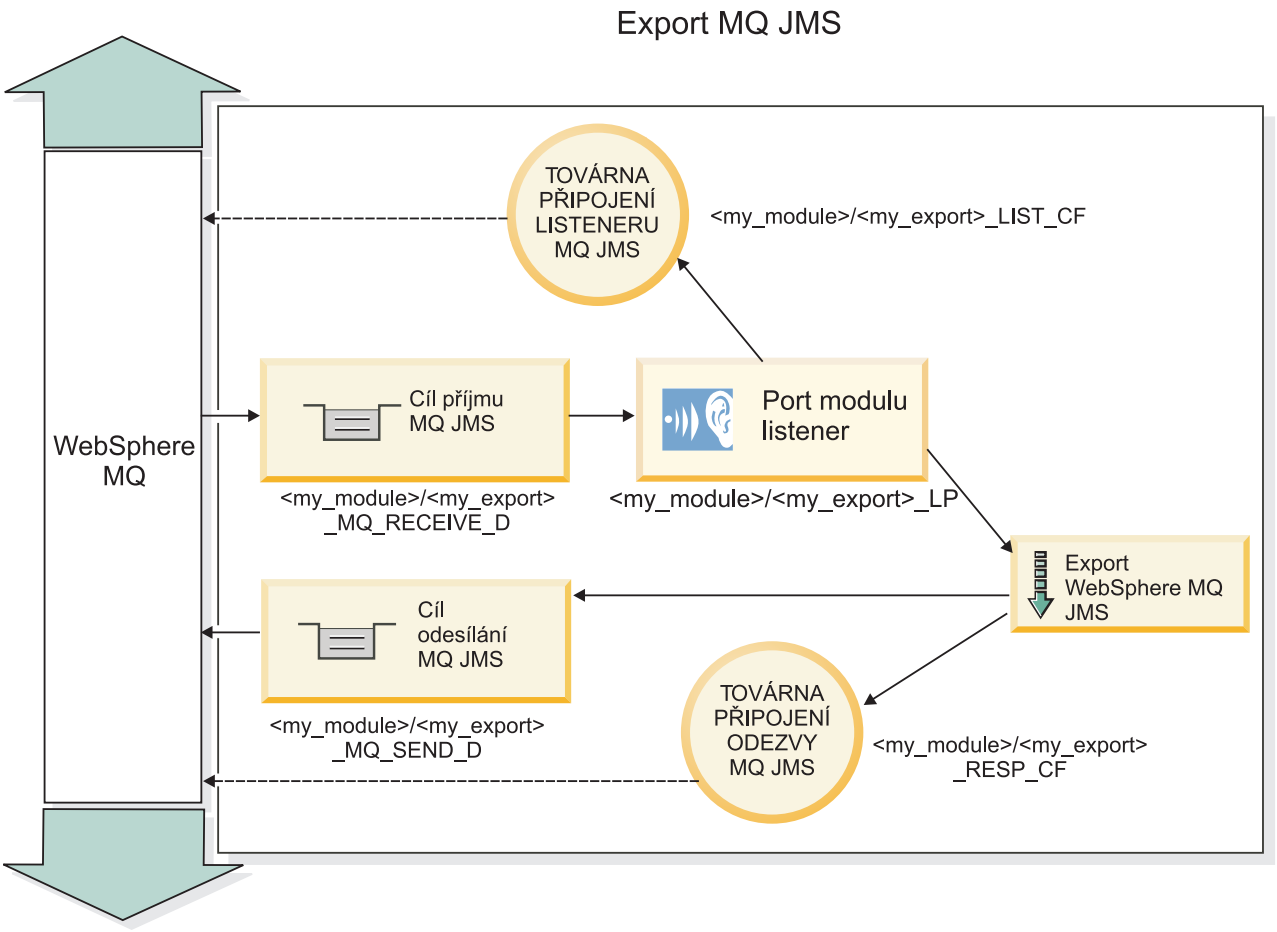
Obrázek 37. Prostředky vazby importu produktu WebSphere MQ JMS

Vazby exportu WebSphere MQ JMS

Vazba exportu produktu WebSphere MQ JMS zajišťuje modulům SCA prostředky pro poskytování služeb externím aplikacím JMS na poskytovateli JMS na bázi WebSphere MQ.

Je implementován MDB, aby naslouchal požadavkům přicházejícím do cíle operace příjmu (receive) určeného ve vazbě exportu. Cíl určený v poli odeslání (send) se používá k odeslání odpovědi na příchozí požadavek, pokud vyvolaná komponenta poskytuje odpověď. Cíl určený v poli replyTo zprávy odpovědi přepíše cíl určený v poli odeslání (send).

Obrázek 38 na stránce 125 ukazuje, jak je externí žadatel propojen s exportem.



Obrázek 38. Prostředky vazeb exportu produktu WebSphere MQ JMS

Poznámka: Obrázek 37 na stránce 124 a Obrázek 38 ukazují, jak je aplikace z předchozí verze produktu IBM Business Process Manager propojena s externí službou. U aplikací vyvinutých pro produkt IBM Business Process Manager verze 7.0 se místo Portu modulu listener a Továrny připojení používá Specifikace aktivace.

Klíčové vlastnosti vazeb WebSphere MQ JMS:

Klíčové vlastnosti vazeb WebSphere MQ JMS zahrnují záhlaví, artefakty Java EE a vytvoření prostředky Java EE.

Záhlaví

Záhlaví platformy JMS obsahuje mnoho předdefinovaných polí obsahujících hodnoty používané klientem i poskytovatelem k identifikaci a směrování zpráv. Vlastnosti vazby můžete použít ke konfiguraci těchto záhlaví s pevnými hodnotami nebo je možné záhlaví určit dynamicky za běhu.

JMSCorrelationID
Odkazuje na související zprávu. Běžně je toto pole nastaveno na řetězec identifikátoru zprávy, na níž je odpovídáno.

TargetFunctionName
Toto záhlaví používá jeden z dodaných selektorů funkce k identifikaci operace, která je vyvolávána. Nastavení vlastnosti záhlaví TargetFunctionName ve zprávách zaslaných na export JMS umožňuje použití tohoto selektoru funkce. Vlastnost je možné nastavit přímo v aplikacích klienta JMS nebo při připojení importu s vazbou JMS k takovému exportu. V tom případě by měla být vazba importu JMS konfigurována tak, aby nastavila záhlaví TargetFunctionName pro každou operaci v rozhraní na název operace.

Schémata korelace

Vazby WebSphere MQ JMS poskytují různá schémata korelace, která jsou použita k určení toho, jak korelovat zprávy požadavku se zprávami odpovědi.

RequestMsgIDToCorrelID
Položka JMSMessageID je zkopírována do pole JMSCorrelationID. Toto je výchozí nastavení.

RequestCorrelIDToCorrelID
Položka JMSCorrelationID je kopírována do pole JMSCorrelationID.

Prostředky Java EE

Mnoho prostředků Java EE je vytvářeno, když je import MQ JMS implementován do prostředí Java EE.

Parametry

MQ Connection Factory
Používaná klienty k vytvoření připojení k poskytovateli MQ JMS.

Response Connection Factory
Používá ji běhové prostředí MQ JMS architektury SCA, když je místo určení odeslání na jiném produktu Queue Manager než cíl příjmu.

Activation specification
Specifikace aktivace MQ JMS je přidružena k jednomu nebo více objektům typu message-driven bean a poskytuje konfiguraci, která je pro ně nezbytná pro příjem zpráv.

Destinations

- Cíl odesílání:
  - Importy: Kam je zasílána odchozí zpráva nebo požadavek.
  - Exporty: Kam bude zaslána zpráva odpovědi v případě, že není nahrazena polem záhlaví JMSReplyTo příchozí zprávy.
- Cíl příjmu:
  - Importy: Kam by měla být umístěna příchozí zpráva či odpověď.
  - Exporty: Kam by měla být umístěna příchozí zpráva či zpráva požadavku.

Záhlaví JMS:

Zpráva JMS obsahuje dva typy záhlaví: systémová záhlaví JMS a různé vlastnosti JMS. K oběma typům záhlaví lze přistupovat buď v mediačním modulu v objektu SMO (Service Message Object), nebo prostřednictvím rozhraní API ContextService.

Systémové záhlaví JMS

Systémové záhlaví JMS je v objektu SMO představováno prvkem JMSHeader, který obsahuje všechna pole, jež se obvykle vyskytují v záhlaví JMS. Ačkoli je lze upravit v mediaci (nebo službě ContextService), některá pole systémových záhlaví JMS nastavená v objektu SMO se nebudou šířit v odchozí zprávě JMS, protože jsou přepsána systémovými nebo statickými hodnotami.

Klíčová pole v systémovém záhlaví JMS, která lze aktualizovat v mediaci (nebo službě ContextService):

- **JMSType** a **JMSCorrelationID** – hodnoty specifických předdefinovaných vlastností záhlaví zprávy.
- **JMSDeliveryMode** – hodnoty způsobu doručení persistent (perzistentní) nebo non-persistent (dočasný); výchozí je persistent).
- **JMSPriority** – hodnota priority (0 až 9; výchozí je JMS_Default_Priority.)

Vlastnosti JMS

Vlastnosti JMS jsou v objektu SMO reprezentovány položkami seznamu Vlastnosti. Vlastnosti lze přidávat, aktualizovat nebo odstraňovat v mediaci nebo prostřednictvím rozhraní API ContextService.

Vlastnosti lze také nastavit staticky ve vazbě JMS. Vlastnosti, které jsou nastaveny staticky, přepisují nastavení (se stejným názvem), která jsou nastavena dynamicky.

Uživatelské vlastnosti šířené z jiných vazeb (například jako vazba HTTP) budou výstupem ve vazbě JMS jako vlastnosti JMS.

Nastavení šíření záhlaví

Šíření systémového záhlaví a vlastností JMS buď z příchozí zprávy JMS do následných komponent, nebo z předchozích komponent do odchozí zprávy JMS lze řídit příznakem Šířit záhlaví protokolů pro danou vazbu.

Když je nastaven příznak Šířit záhlaví protokolů, je informacím o záhlaví dovoleno plynout do zprávy nebo do cílové komponenty, jak popisuje následující seznam:

- Požadavek exportu JMS

  Záhlaví JMS přijaté ve zprávě bude šířeno do cílové komponenty prostřednictvím kontextové služby. Vlastnosti JMS přijaté ve zprávě budou šířeny do cílové komponenty prostřednictvím kontextové služby.
- Odpověď exportu JMS

  Všechna pole záhlaví JMS nastavená v kontextové službě budou použita v odchozí zprávě, pokud nejsou přepsána statickými vlastnostmi nastavenými pro vazbu exportu JMS. Všechny vlastnosti nastavené v kontextové službě budou použity v odchozí zprávě, pokud nejsou přepsány statickými vlastnostmi nastavenými pro vazbu exportu JMS.
- Požadavek importu JMS

  Všechna pole záhlaví JMS nastavená v kontextové službě budou použita v odchozí zprávě, pokud nejsou přepsána statickými vlastnostmi nastavenými pro vazbu importu JMS. Všechny vlastnosti nastavené v kontextové službě budou použity v odchozí zprávě, pokud nejsou přepsány statickými vlastnostmi nastavenými pro vazbu importu JMS.
- Odpověď importu JMS

  Záhlaví JMS přijaté ve zprávě bude šířeno do cílové komponenty prostřednictvím kontextové služby. Vlastnosti JMS přijaté ve zprávě budou šířeny do cílové komponenty prostřednictvím kontextové služby.

Externí klienti:

Server může odesílat zprávy externím klientům nebo od nich zprávy přijímat pomocí vazeb WebSphere MQ JMS.

Externí klient (například webový portál nebo podnikový informační systém) může odeslat zprávu komponentě SCA v aplikaci prostřednictvím exportu nebo může být vyvolán komponentou SCA v aplikaci prostřednictvím importu.

Vazba exportu WebSphere MQ JMS implementuje objekty typu message-driven bean (MDB) za účelem naslouchání požadavkům přicházejícím do cíle operace příjmu (receive) určeného ve vazbě exportu. Cíl určený v poli odeslání (send) se používá k odeslání odpovědi na příchozí požadavek, pokud vyvolaná aplikace nějakou odpověď poskytne. Proto je externí klient schopen vyvolávat aplikace prostřednictvím vazby exportu.

Importy WebSphere MQ JMS se vážou k externím klientům a mohou jim doručit zprávu. Tato zpráva může, ale nemusí od externího klienta požadovat odpověď.

Další informace o způsobu interakcí s externími klienty pomocí produktu WebSphere MQ naleznete v Informačním centru tohoto produktu WebSphere MQ..

Odstraňování problémů vazeb produktu WebSphere MQ JMS:

Problémy s vazbami produktu WebSphere MQ JMS lze diagnostikovat a opravit.

Výjimky implementace

Jako odpověď na různé chybové stavy může implementace importu a exportu MQ JMS vracet jeden ze dvou typů výjimek:

- Obchodní výjimka služby: Tato výjimka je vrácena, pokud došlo k poruše určené na obchodním rozhraní služby (typ portu WSDL).
- Výjimka za běhu služby: Generuje se ve všech ostatních případech. Ve většině případů bude výjimka příčina obsahovat původní výjimku (JMSException).

  Například import očekává pouze jednu zprávu odezvy na každou zprávu požadavku. Pokud dorazí více než jedna odezva nebo pokud dorazí pozdní odezva (odezva, pro kterou došlo k vypršení odezvy SCA), je vyvolána výjimka za běhu služby. Transakce je odvolána a zpráva odezvy je vrácena zpět z fronty nebo zpracována správcem událostí se selháním.

Zprávy SCA založené na produktu WebSphere MQ JMS se nezobrazují ve správci událostí se selháním

Pokud dojde k selhání zpráv SCA vytvořených prostřednictvím interakce produktu WebSphere MQ JMS, měly by se tyto zprávy objevit ve správci událostí se selháním. Pokud se tyto zprávy ve správci událostí se selháním neobjeví, zkontrolujte, zda je hodnota vlastnosti maximálního počtu pokusů základního portu modulu listener rovna nebo větší než **1**. Nastavení této hodnoty na **1** a více umožní interakci se správcem událostí se selháním během vyvolání SCA pro vazby MQ JMS.

Scénáře nesprávného použití: Porovnání s vazbami produktu WebSphere MQ

Vazba produktu WebSphere MQ JMS je určena ke spolupráci s aplikacemi JMS implementovanými na produktu WebSphere MQ, který vystavuje zprávy podle modelu zpráv JMS. Import a export WebSphere MQ je však v podstatě určen ke spolupráci s nativními aplikacemi produktu WebSphere MQ a vystavuje úplný obsah těla zprávyWebSphere MQ mediacím.

Následující scénáře by měly být sestaveny pomocí vazby WebSphere MQ JMS, nikoli vazby WebSphere MQ:

- Vyvolání objektu JMS typu message-driven bean (MDB) z modulu SCA, kde je objekt typu message-driven bean implementován na poskytovateli produktu WebSphere MQ JMS. Použijte import produktu WebSphere MQ JMS.
- Povolení volání modulu SCA ze servletu komponenty Java EE nebo EJB pomocí JMS. Použijte export produktu WebSphere MQ JMS.
- Mediace obsahu JMS MapMessage při přenosu přes produkt WebSphere MQ. Použijte export a import produktu WebSphere MQ JMS ve spojení s příslušným popisovačem dat nebo vázáním dat.

Existují situace, ve kterých lze očekávat spolupráci vazeb produktů WebSphere MQ a WebSphere MQ JMS. Zejména při přemostění mezi aplikacemi WebSphere MQ patřícími a nepatřícími k prostředí Java EE použijte export WebSphere MQ a import WebSphere MQ JMS (nebo naopak) ve spojení s příslušnými vázáními dat nebo mediačními moduly (nebo obojím).

Zpracování výjimek:

Způsob, jakým je vazba nakonfigurována, určuje, jak jsou zpracovány výjimky, které jsou způsobeny popisovači dat nebo vázáními dat. Chování systému při vrácení takové výjimky je dále určeno také charakterem mediačního toku.

Když vaše vazba volá popisovač dat nebo vázání dat, může dojít k nejrůznějším problémům. Například může popisovač dat obdržet zprávu s poškozeným informačním obsahem nebo se může pokusit přečíst zprávu v nesprávném formátu.

Způsob, jakým vaše vazba ošetří takovou výjimku, je určen tím, jak implementujete daný popisovač dat nebo vázání dat. Doporučené chování je navrhnout vázání dat tak, aby vracelo výjimku **DataBindingException**.

Podobná situace je i u popisovače dat. Protože je popisovač dat vyvolán vázáním dat, je jakákoli výjimka popisovače dat zabalena do výjimky vázání dat. Proto je výjimka **DataHandlerException** hlášena jako **DataBindingException**.

Když je vrácena jakákoli výjimka za běhu, včetně výjimky **DataBindingException**:

- Pokud je mediační tok nakonfigurován jako transakční, je zpráva JMS standardně uložena do komponenty Failed Event Manager pro ruční přehrání nebo odstranění.

  Poznámka: Režim zotavení pro vazbu je možné změnit tak, aby byla zpráva namísto uložení do komponenty Failed Event Manager odvolána.
- Pokud mediační tok není transakční, je výjimka zaprotokolována a zpráva je ztracena.

Podobná situace je i u popisovače dat. Protože je popisovač dat volán vázáním dat, je výjimka popisovače dat vytvořena uvnitř výjimky vázání dat. Proto je výjimka **DataHandlerException** hlášena jako **DataBindingException**.

Vazby WebSphere MQ:

Vazba WebSphere MQ zajišťuje konektivitu architektury SCA (Service Component Architecture) s aplikacemi WebSphere MQ.

Pomocí vazeb exportu a importu WebSphere MQ můžete zajistit integraci se systémem založeným na produktu WebSphere MQ přímo z prostředí vašeho serveru. Tím odpadá nutnost používat funkce Propojení MQ nebo Propojení klienta sběrnice SIBus.

Když vstoupí komponenta do interakce se službou WebSphere MQ prostřednictvím importu, vazba importu WebSphere MQ využije frontu, kam budou odeslána data, a frontu, kde je možné přijmout odpověď.

Když modul SCA poskytuje službu klientům WebSphere MQ, používá vazba exportu WebSphere MQ frontu, kde je možné přijmout požadavek a odeslat odpověď. Selektor funkcí zajišťuje mapování na operaci na cílové komponentě, která má být vyvolána.

Převod dat informačního obsahu do zprávy MQ i z ní se provádí prostřednictvím popisovače dat nebo vázáním dat MQ. Převod dat záhlaví do zprávy MQ i z ní se provádí prostřednictvím vázáním dat záhlaví MQ.

Informace o podporovaných verzích WebSphere MQ viz webová stránka se systémovými požadavky.

Přehled vazeb WebSphere MQ:

Vazba WebSphere MQ zajišťuje integraci a nativními aplikacemi založenými na MQ.

Administrativní úlohy WebSphere MQ

Od administrátora systému WebSphere MQ se očekává, že vytvoří základní produkt WebSphere MQ Queue Manager, který budou používat vazby WebSphere MQ, před spuštěním aplikace obsahující tyto vazby.

Administrativní úlohy WebSphere

Musíte nastavit vlastnost **Cesta k nativní knihovně** adaptéru prostředku MQ ve Websphere na verzi WebSphere MQ podporovanou serverem a restartovat server. To zajistí, že budou využívány knihovny podporované verze WebSphere MQ. Podrobné požadavky na hardware a software naleznete na stránkách podpory IBM.

Vazby importu WebSphere MQ

Vazba importu WebSphere MQ umožňuje komponentám ve vašem modulu SCA komunikovat se službami poskytovanými externími aplikacemi založenými na WebSphere MQ. Musíte používat podporovanou verzi produktu WebSphere MQ. Podrobné požadavky na hardware a software naleznete na stránkách podpory IBM.

Interakce s externími systémy WebSphere MQ zahrnuje použití front pro odesílání požadavků a příjem odpovědí.

Jsou podporovány dva typy scénářů využití vazeb importu WebSphere MQ, v závislosti na typu vyvolávané operace:

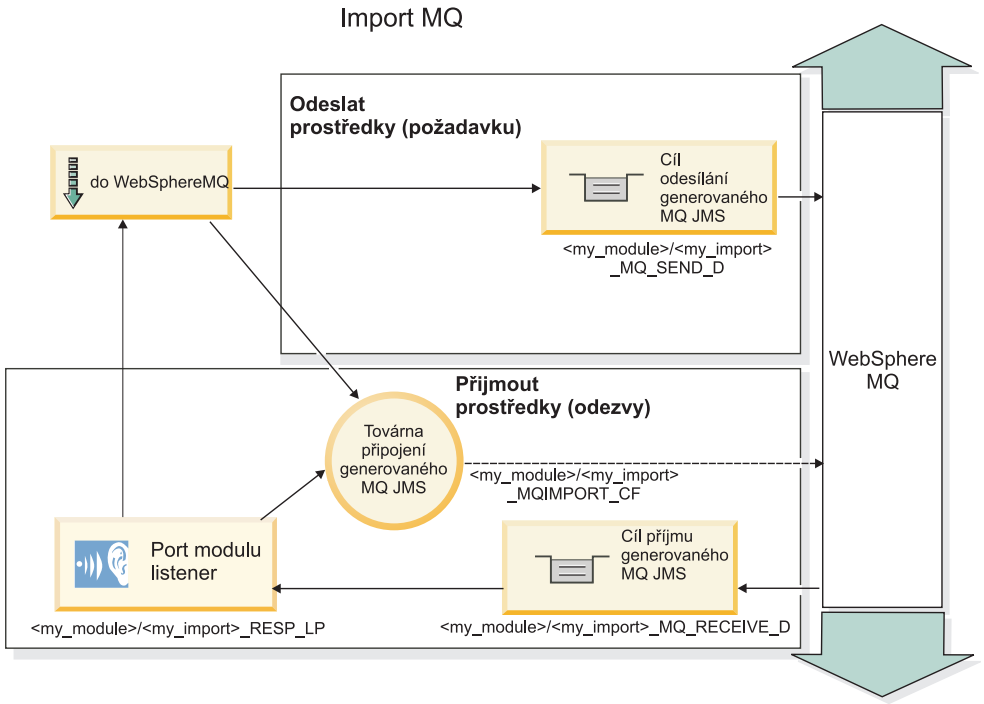
- Jednosměrný: Import WebSphere MQ vloží zprávu do fronty nakonfigurované v poli **Cílová fronta odeslání** vazby importu. Do pole replyTo v záhlaví MQMD se nic nepošle.
- Obousměrný (požadavek-odezva): Import WebSphere MQ vloží zprávu do fronty nakonfigurované v poli **Cílová fronta odeslání**.

  Fronta operace příjmu (receive) je nastavena v poli záhlaví replyTo MQMD. Je implementován objekt typu message-driven bean (MDB), aby naslouchal frontě operace příjmu (receive), a když je přijata odpověď, MDB předá tuto odpověď zpět komponentě.

  Vazbu importu lze nakonfigurovat (pomocí pole **Schéma korelace odpovědí**), aby očekávala ID korelace zprávy odpovědi zkopírované z ID zprávy požadavku (výchozí nastavení), nebo z ID korelace zprávy požadavku.

Je důležité uvědomit si, že WebSphere MQ je asynchronní vazba. Pokud volající komponenta vyvolá import WebSphere MQ synchronně (pro obousměrnou operaci), je volající komponenta zablokována, dokud služba WebSphere MQ nevrátí odpověď.

Obrázek 39 ukazuje, jak je import propojen s externí službou.



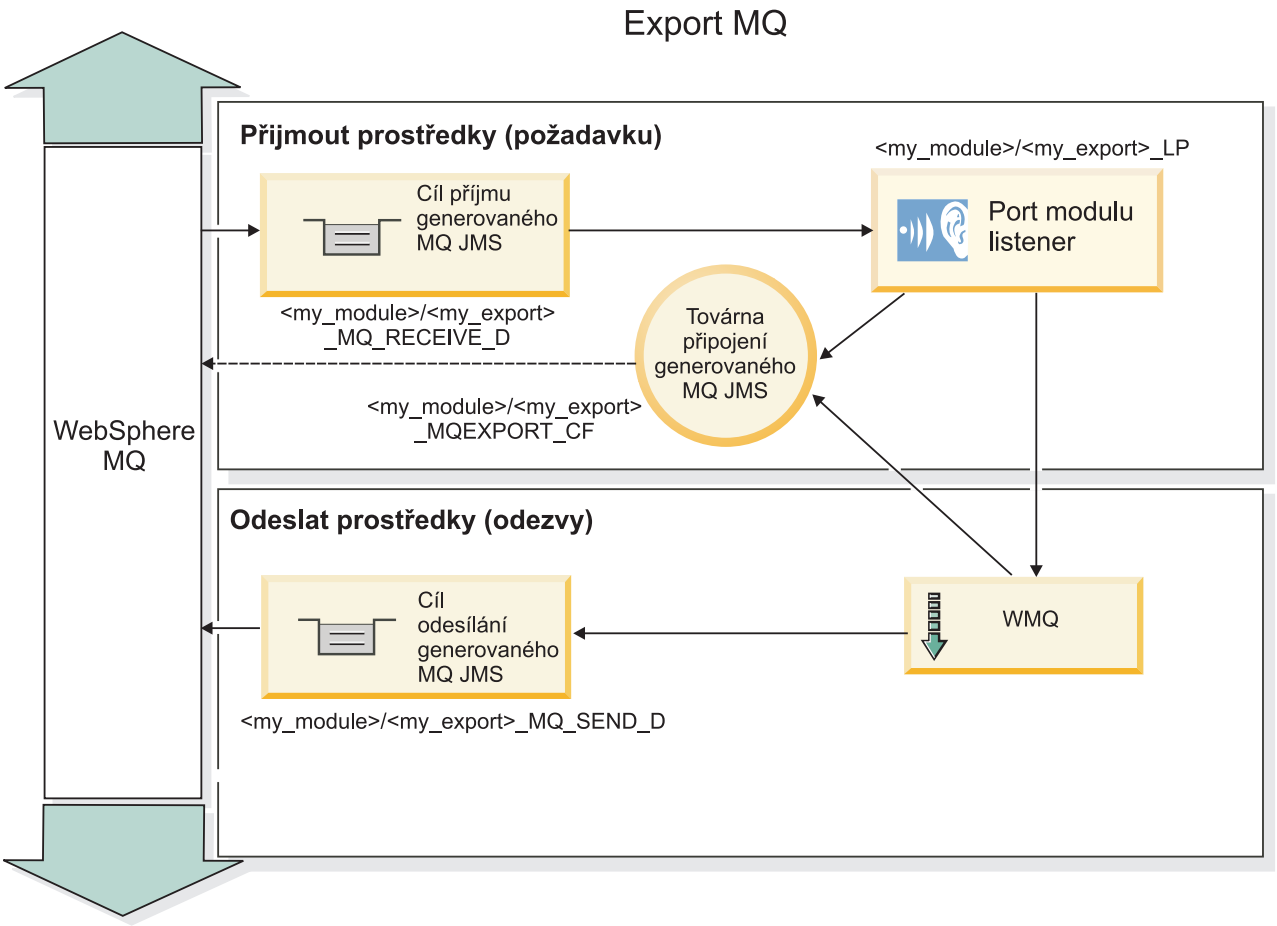
Obrázek 39. Prostředky vazby importu WebSphere MQ

Vazby exportu WebSphere MQ

Vazba exportu WebSphere MQ zajišťuje modulům SCA prostředky pro poskytování služeb externím aplikacím založeným na WebSphere MQ.

Je implementován MDB, aby naslouchal požadavkům přicházejícím do **cílové fronty příjmu** určené ve vazbě exportu. Fronta určená v poli **Cílová fronta odeslání** se používá k odeslání odpovědi na příchozí požadavek, pokud vyvolaná Komponenta nějakou odpověď poskytuje. Fronta určená v poli replyTo zprávy odpovědi přepíše frontu určenou v poli **Cílová fronta odeslání**.

Obrázek 40 ukazuje, jak je externí žadatel propojen s exportem.



Obrázek 40. Prostředky vazeb exportu WebSphere MQ

Poznámka: Obrázek 39 na stránce 130 a Obrázek 40 ukazují, jak je aplikace z předchozí verze produktu IBM Business Process Manager propojena s externí službou. U aplikací vyvinutých pro produkt IBM Business Process Manager verze 7.x se místo Portu modulu listener a Továrny připojení používá Specifikace aktivace.

Klíčové vlastnosti vazby WebSphere MQ:

Klíčové vlastnosti vazby WebSphere MQ zahrnují záhlaví, artefakty Java EE a vytvořené prostředky Java EE.

Schémata korelace

Aplikace požadavku/odezvy WebSphere MQ může využít jednu z mnoha technik ke korelaci zpráv odpovědi s požadavky, které jsou sestaveny kolem polí MQMD MessageID a CorrelID. Ve většině případů umožní žadatel správci front vybrat položku MessageID a očekává, že odpovídající aplikace tuto položku zkopíruje do položky CorrelID odpovědi. Většinou žadatel i odpovídající aplikace implicitně ví, která technika korelace se používá. Občas odpovídající aplikace přidá různé příznaky do pole Sestava požadavku a ty popisují, jak s těmito poli pracovat.

Vazby exportu zpráv WebSphere MQ lze konfigurovat pomocí následujících voleb:

Volby položky MsgId odezvy:

New MsgID
Umožní správci front vybrat jedinečné MsgId pro odezvu (výchozí).

Copy from Request MsgID
Zkopíruje pole MsgId z pole MsgId v požadavku.

Copy from SCA message
Nastavuje, že MsgId bude stejné, jako to v záhlaví WebSphere MQ ve zprávě požadavku SCA, nebo nechá správce front definovat nové Id, pokud hodnota neexistuje.

As Report Options
Prozkoumává pole Sestava MQMD v požadavku a hledá radu, jak pracovat s MsgId. Podporovány jsou volby MQRO_NEW_MSG_ID a MQRO_PASS_MSG_ID a chovají se jako položky Nové MsgId, respektive Copy from Request MsgI.

Volby položky CorrelId odezvy:

Copy from Request MsgID
Zkopíruje pole CorrelId z pole MsgId v požadavku (výchozí).

Copy from Request CorrelID
Zkopíruje pole CorrelId z pole CorrelId v požadavku.

Copy from SCA message
Nastavuje, že CorrelId bude stejné, jako to v záhlaví WebSphere MQ ve zprávě požadavku SCA, nebo jej nechává prázdné v případě, že hodnota neexistuje.

As Report Options
Prozkoumává pole Sestava MQMD v požadavku a hledá radu, jak pracovat s CorrelId. Podporovány jsou volby MQRO_COPY_MSG_ID_TO_CORREL_ID a MQRO_PASS_CORREL_ID a chovají se jako položky Copy from Request MsgID, respektive Copy from Request CorrelID.

Vazby importu zpráv WebSphere MQ lze konfigurovat pomocí následujících voleb:

Volby položky MsgId požadavku:

New MsgID
Umožní správci front vybrat jedinečné MsgId pro žádost (výchozí).

Copy from SCA message
Nastavuje, že MsgId bude stejné, jako to v záhlaví WebSphere MQ ve zprávě požadavku SCA, nebo nechá správce front definovat nové Id, pokud hodnota neexistuje.

Volby korelace odezvy:

Response has CorrelID copied from MsgId
Očekává, že zpráva odpovědi bude mít nastaveno pole CorrelId, na základě položky MsgId požadavku (výchozí).

Response has MsgID copied from MsgId
Očekává, že zpráva odpovědi bude mít nastaveno pole MsgId, na základě položky MsgId požadavku.

Response has CorrelID copied from CorrelId
Očekává, že zpráva odpovědi bude mít nastaveno pole CorrelId, na základě položky CorrelId požadavku.

Prostředky Java EE

Mnoho prostředků Java EE je vytvářeno, když je vazba WebSphere MQ implementována do prostředí Java EE.

Parametry

MQ Connection Factory
Používaná klienty k vytvoření připojení k poskytovateli WebSphere MQ.

Response Connection Factory

Používá ji běhové prostředí MQ architektury SCA, když je místo určení odeslání na jiném produktu Queue Manager než cíl příjmu.

Activation specification

Specifikace aktivace MQ JMS je přidružena k jednomu nebo více objektům typu message-driven bean a poskytuje konfiguraci, která je pro ně nezbytná pro příjem zpráv.

Destinations

- Cíl odesílání: kam je zasílán požadavek nebo odchozí zpráva (import); kam bude zaslána zpráva odpovědi (export) v případě, že není nahrazena polem záhlaví ReplyTo MQMD v příchozí zprávě.
- Cíl příjmu: Kam má být umístěna příchozí zpráva či odezva/požadavek.

Záhlaví WebSphere MQ:

Záhlaví WebSphere MQ zahrnují určité konvence pro převod zpráv SCA (Service Component Architecture).

Zprávy WebSphere MQ sestávají ze záhlaví systému (MQMD), nula nebo více jiných záhlaví MQ (systémových nebo přizpůsobených) a těla zprávy. Pokud zpráva obsahuje více záhlaví zprávy, je důležité jejich pořadí.

Každé záhlaví obsahuje informace popisující strukturu následujícího záhlaví. Záhlaví systému MQMD popisuje první záhlaví.

Způsob analýzy záhlaví MQ

Vázání dat záhlaví MQ se používá k analýza záhlaví MQ. Následující záhlaví jsou podporována automaticky:

- MQRFH.
- MQRFH2.
- MQCIH.
- MQIIH.

Záhlaví začínající na **MQH** jsou zpracována různě. Specifická pole záhlaví nejsou analyzována a zůstávají ve formě neanalyzovaných bajtů.

V případě záhlaví MQ můžete zapsat vlastní vázání dat záhlaví MQ, která budou tato záhlaví analyzovat.

Způsob přístupu k záhlavím MQ

K záhlavím MQ lze v produktu přistoupit dvěma způsoby:

- Prostřednictvím objektu SMO (Service Message Object) v rámci mediace.
- Prostřednictvím rozhraní API ContextService.

Záhlaví MQ jsou interně reprezentována prvkem MQHeader objektu SMO. Prvek MQHeader je kontejner dat záhlaví, který rozšiřuje prvek MQControl, ale obsahuje prvek s hodnotou libovolného typu. Obsahuje záhlaví MQMD, MQControl (řídící informace těla zprávy MQ) a seznam dalších záhlaví MQ.

- Záhlaví MQMD představuje obsah popisu zprávy WebSphere MQ, kromě informací určujících strukturu a kódování těla.
- Záhlaví MQControl obsahuje informace určující strukturu a kódování těla zprávy.
- Záhlaví MQHeaders obsahuje seznam objektů MQHeader.

Řetězec záhlaví MQ je nesvázaný, takže v rámci objektu SMO nese každé záhlaví MQ své vlastní řídicí informace (CCSID, kódování a formát). Záhlaví lze snadno přidat či odebrat bez nutnosti pozměnění jiných dat záhlaví.

Nastavení polí v záhlaví MQMD

Záhlaví MQMD můžete aktualizovat pomocí rozhraní API kontextu nebo prostřednictvím objektu SMO (Service Message Object) v rámci mediace. Následující pole jsou automaticky šířena do odchozí zprávy MQ:

- Encoding.
- CodedCharacterSet.
- Format.
- Report.
- Expiry.
- Feedback.
- Priority.
- Persistence.
- CorrelId.
- MsgFlags.

Pokud chcete do odchozí zprávy MQ šířit následující vlastnosti, nakonfigurujte vazbu MQ na import či export:

MsgID
Volbu **ID zprávy požadavku** nastavte na hodnotu zkopírovat ze zprávy SCA.

MsgType
Zrušte zaškrtnutí políčka u volby **Nastavit typ zprávy pro operaci požadavek-odezva na MQMT_DATAGRAM nebo MQMT_REQUEST**.

ReplyToQ
Zrušte zaškrtnutí políčka vedle volby **Potlačit odpověď na frontu zprávy požadavku**.

ReplyToQMgr
Zrušte zaškrtnutí políčka vedle volby **Potlačit odpověď na frontu zprávy požadavku**.

Počínaje verzí 7.0 lze kontextová pole přepsat pomocí přizpůsobené vlastnosti z definice cíle rozhraní JNDI. Nastavením přizpůsobené vlastnosti MDCTX na hodnotu SET_IDENTITY_CONTEXT v cíli odeslání zajistíte šíření následujících polí do odchozí zprávy MQ:

- UserIdentifier.
- AppIdentityData.

Nastavením přizpůsobené vlastnosti MDCTX na hodnotu SET_ALL_CONTEXT v cíli odeslání zajistíte šíření následujících polí do odchozí zprávy MQ:

- UserIdentifier.
- AppIdentityData.
- PutApplType.
- PutApplName.
- ApplOriginData.

Některá pole nejsou do odchozí zprávy MQ šířena. Následující pole jsou přepsána během odesílání zprávy:

- BackoutCount.
- AccountingToken.
- PutDate.
- PutTime.
- Posunutí
- OriginalLength.

Statické přidání MQCIH do vazby WebSphere MQ:

Produkt IBM Business Process Manager podporuje statické přidání informace záhlaví MQCIH bez použití mediačního modulu.

Ke zprávě lze přidat informace záhlaví MQCIH několika různými způsoby (například pomocí mediačního primitiva Modul nastavení záhlaví). Mohlo by být užitečné přidávat tyto informace záhlaví staticky, bez použití dalšího mediačního modulu. Statické informace záhlaví, včetně názvu programu CICS, ID transakce a dalších podrobností záhlaví datového formátu, lze nadefinovat a přidat jako součást vazby WebSphere MQ.

WebSphere MQ, MQ CICS Bridge a CICS je pro statické přidávání informací záhlaví MQCIH nutné nakonfigurovat.

Ke konfiguraci importu WebSphere MQ se statickými hodnotami nezbytnými pro informace záhlaví MQCIH můžete použít produkt Integration Designer.

Při zpracování příchozí zprávy importem WebSphere MQ se provádí kontrola, zda je ve zprávě již přítomna informace záhlaví MQCIH. Pokud je informace záhlaví MQCIH k dispozici, statické hodnoty definované v importu WebSphere MQ se použijí k potlačení odpovídajících dynamických hodnot ve zprávě. Pokud MQCIH k dispozici není, tyto informace záhlaví se ve zprávě vytvoří a poté se přidají statické hodnoty definované v importu WebSphere MQ.

Statické hodnoty definované v importu WebSphere MQ jsou specifické pro danou metodu. Pro jiné metody v rámci téhož importu WebSphere MQ můžete zadat odlišné statické hodnoty MQCIH.

Pokud MQCIH specifické informace záhlaví postrádá, tento mechanizmus se k poskytování výchozích hodnot nepoužívá, protože statická hodnota definovaná v importu WebSphere MQ potlačí odpovídající hodnotu obsaženou v příchozí zprávě.

Externí klienti:

Produkt IBM Business Process Manager může odesílat zprávy externím klientům nebo od nich zprávy přijímat pomocí vazeb WebSphere MQ.

Externí klient (například webový portál nebo podnikový informační systém) může odeslat zprávu komponentě SCA v aplikaci prostřednictvím exportu nebo může být vyvolán komponentou SCA v aplikaci prostřednictvím importu.

Vazba exportu WebSphere MQ implementuje objekty typu message-driven bean (MDB) za účelem naslouchání požadavkům přicházejícím do cíle operace příjmu (receive) určeného ve vazbě exportu. Cíl určený v poli odeslání (send) se používá k odeslání odpovědi na příchozí požadavek, pokud vyvolaná aplikace nějakou odpověď poskytne. Proto je externí klient schopen vyvolávat aplikace prostřednictvím vazby exportu.

Importy WebSphere MQ se vážou k externím klientům a mohou jim doručit zprávu. Tato zpráva může, ale nemusí od externího klienta požadovat odpověď.

Další informace o způsobu interakcí s externími klienty pomocí produktu WebSphere MQ naleznete v Informačním centru tohoto produktu WebSphere MQ..

Odstraňování problémů vazeb WebSphere MQ:

Poruchy a podmínky selhání, ke kterým dojde u vazeb WebSphere MQ lze diagnostikovat a opravit.

Primární podmínky selhání

Primární podmínky selhání vazeb WebSphere MQ jsou určeny transakční sémantikou, konfigurací produktu WebSphere MQ, nebo odkazem na existující chování v jiných komponentách. Primární podmínky selhání zahrnují:

- Selhání při připojení ke správci front WebSphere MQ nebo ke frontě.

Selhání při připojení k produktu WebSphere MQ při přijímání zpráv bude mít za následek selhání spuštění portu modulu listener MDB. Tato podmínka bude zaznamenána do protokolu produktu WebSphere Application Server. Trvalé zprávy zůstanou ve frontě produktu WebSphere MQ, dokud nebudou úspěšně načteny (nebo dokud nevyprší jejich platnost v produktu WebSphere MQ).

Selhání při připojení k produktu WebSphere MQ při odesílání odchozích zpráv způsobí odvolání transakce, která řídí odeslání.

- Selhání při analýze příchozí zprávy nebo při vytvoření odchozí zprávy.

  Selhání ve vázání dat způsobí odvolání transakce, která tuto práci řídí.
- Selhání při odesílání odchozí zprávy.

  Selhání při odesílání zprávy způsobí odvolání příslušné transakce.
- Vícenásobné nebo neočekávané zprávy odezvy.

  Import očekává pouze jednu zprávu odezvy na každou zprávu požadavku. Pokud dorazí více než jedna odezva nebo pokud dorazí pozdní odezva (odezva, pro kterou došlo k vypršení odezvy SCA), je vyvolána výjimka za běhu služby. Transakce je odvolána a zpráva odezvy je vrácena zpět z fronty nebo zpracována správcem událostí se selháním.

Scénáře nesprávného použití: Porovnání s vazbami WebSphere MQ JMS

Import a export WebSphere MQ je v podstatě určen ke spolupráci s nativními aplikacemi produktu WebSphere MQ a vystavuje úplný obsah těla zprávyWebSphere MQ mediacím. Vazba WebSphere MQ JMS je však určena ke spolupráci s aplikacemi JMS implementovanými na produktu WebSphere MQ, který vystavuje zprávy podle modelu zpráv JMS.

Následující scénáře by měly být sestaveny pomocí vazby WebSphere MQ JMS, nikoli vazby WebSphere MQ:

- Vyvolání objektu JMS typu message-driven bean (MDB) z modulu SCA, kde je objekt typu message-driven bean implementován na poskytovateli produktu WebSphere MQ JMS. Použijte import produktu WebSphere MQ JMS.
- Povolení volání modulu SCA ze servletu komponenty Java EE nebo EJB pomocí JMS. Použijte export produktu WebSphere MQ JMS.
- Mediace obsahu JMS MapMessage při přenosu přes produkt WebSphere MQ. Použijte export a import WebSphere MQ JMS ve spojení s příslušným vázáním dat.

Existují situace, ve kterých lze očekávat spolupráci vazeb produktů WebSphere MQ a WebSphere MQ JMS. Zejména při přemostění mezi aplikacemi WebSphere MQ patřícími a nepatřícími k prostředí Java EE použijte export WebSphere MQ a import WebSphere MQ JMS (nebo naopak) ve spojení s příslušnými vázáními dat nebo mediačními moduly (nebo obojím).

Nedoručené zprávy

Pokud produkt WebSphere MQ nemůže doručit zprávu na určené místo určení (například z důvodu chyb konfigurace), odešle namísto toho zprávu do nominované fronty nedoručených zpráv.

Přitom přidá na začátek těla zprávy záhlaví nedoručené zprávy. Toto záhlaví obsahuje příčiny selhání, původní místo určení a další informace.

Zprávy SCA založené na MQ se nezobrazují ve správci událostí se selháním

Pokud dojde k selhání zpráv SCA vytvořených prostřednictvím interakce WebSphere MQ, měly by se tyto zprávy objevit ve správci událostí se selháním. Pokud se tyto zprávy ve správci událostí se selháním neobjeví, zkontrolujte, zda je hodnota vlastnosti maximálního počtu pokusů základního místa určení WebSphere MQ větší než 1. Nastavení této hodnoty na 2 a více umožní interakci se správcem událostí se selháním během vyvolání SCA pro vazby WebSphere MQ.

Události MQ se selháním se přehrávají do chybného správce front

Když se má použít předdefinovaná továrna připojení pro odchozí připojení, vlastnosti připojení musí odpovídat vlastnostem definovaným ve specifikaci aktivace použité pro příchozí připojení.

Předdefinovaná továrna připojení se používá k vytvoření připojení při přehrávání události se selháním, a proto musí být nakonfigurována k použití stejného správce front,ze kterého byla zpráva původně přijata.

Zpracování výjimek:

Způsob, jakým je vazba nakonfigurována, určuje, jak jsou zpracovány výjimky, které jsou způsobeny popisovači dat nebo vázáními dat. Chování systému při vrácení takové výjimky je dále určeno také charakterem mediačního toku.

Když vaše vazba volá popisovač dat nebo vázání dat, může dojít k nejrůznějším problémům. Například může popisovač dat obdržet zprávu s poškozeným informačním obsahem nebo se může pokusit přečíst zprávu v nesprávném formátu.

Způsob, jakým vaše vazba ošetří takovou výjimku, je určen tím, jak implementujete daný popisovač dat nebo vázání dat. Doporučené chování je navrhnout vázání dat tak, aby vracelo výjimku **DataBindingException**.

Podobná situace je i u popisovače dat. Protože je popisovač dat vyvolán vázáním dat, je jakákoli výjimka popisovače dat zabalena do výjimky vázání dat. Proto je výjimka **DataHandlerException** hlášena jako **DataBindingException**.

Když je vrácena jakákoli výjimka za běhu, včetně výjimky **DataBindingException**:

- Pokud je mediační tok nakonfigurován jako transakční, je zpráva JMS standardně uložena do komponenty Failed Event Manager pro ruční přehrání nebo odstranění.

  Poznámka: Režim zotavení pro vazbu je možné změnit tak, aby byla zpráva namísto uložení do komponenty Failed Event Manager odvolána.
- Pokud mediační tok není transakční, je výjimka zaprotokolována a zpráva je ztracena.

Podobná situace je i u popisovače dat. Protože je popisovač dat volán vázáním dat, je výjimka popisovače dat vytvořena uvnitř výjimky vázání dat. Proto je výjimka **DataHandlerException** hlášena jako **DataBindingException**.

Omezení vazeb:

Použití vazeb má jistá omezení, která jsou vypsána zde.

Omezení vazby MQ:

Použití vazby MQ má jistá omezení, která jsou vypsána zde.

Žádná distribuce zpráv publikování-odběr

Metoda publikování-odběr pro distribuci zpráv není v současnosti vazbou MQ podporována, ačkoli samotné WMQ publikování-odběr podporuje. Vazba MQ JMS však tuto metodu distribuce podporuje.

Sdílené fronty příjmu

Různé vazby exportu a importu WebSphere MQ očekávají, že jsou veškeré zprávy v jejich nakonfigurované frontě příjmu určeny pro daný export nebo import. Vazby importu a exportu by měly být nakonfigurovány se zohledněním těchto aspektů:

- Každý import MQ musí mít jinou frontu příjmu, protože import MQ předpokládá, že jsou všechny zprávy ve frontě příjmu odpovědi na požadavky, které odeslal. Pokud je fronta příjmu sdílená více importy, mohl by odpovědi přijímat nesprávný import a nebyly by korelovány s původními zprávami požadavků.

- Každý export MQ by měl mít jinou frontu příjmu, protože jinak nelze předpovědět, který export dostane určitou zprávu požadavku.
- Importy a exporty MQ mohou ukazovat na stejnou frontu odesílání.

Omezení vazeb JMS, MQ JMS a generické vazba služby JMS:

Vazby JMS a MQ JMS mají určitá omezení.

Implikace generování výchozích vazeb

O omezeních použití vazeb JMS, MQ JMS a generické vazby JMS pojednávají tyto části:
- Implikace generování výchozích vazeb.
- Schéma korelace odpovědi.
- Podpora obousměrného zápisu.

Pokud se při generování vazby nerozhodnete zadat hodnoty sami, bude několik polí vygenerováno za vás. Bude za vás vytvořen například název továrny připojení. Pokud víte, že svou aplikaci umístíte na server a budete k ní přistupovat vzdáleně prostřednictvím klienta, měli byste při vytvoření vazby místo přijetí výchozích hodnot zadat názvy rozhraní JNDI, protože tyto hodnoty budete pravděpodobně chtít kontrolovat pomocí administrativní konzoly za běhu.

Pokud jste však přijali výchozí nastavení a později zjistili, že k vaší aplikaci nelze přistupovat ze vzdáleného klienta, můžete hodnotu továrny připojení nastavit explicitně v administrativní konzole. V nastavení továrny připojení vyhledejte pole Koncové body poskytovatele a přidejte hodnotu, jako např. <název_hostitele_serveru>:7276 (pokud používáte výchozí číslo portu).

Schéma korelace odpovědi

Pokud používáte schéma korelace odpovědi ID korelace - ID korelace používané ke korelaci zpráv v operaci požadavek-odezva, musíte ve zprávě používat dynamické ID korelace.

Chcete-li vytvořit dynamické ID korelace v mediačním modulu pomocí editoru mediačních toků, přidejte před importem s vazbou JMS mediační primitivum Mapování. Otevřete editor mapování. V cílové zprávě budou dostupná známá záhlaví architektury SCA (Service Component Architecture). Přetáhněte pole obsahující jedinečné ID ve zdrojové zprávě na ID korelace v záhlaví JMS v cílové zprávě.

Podpora obousměrného zápisu

V běhovém prostředí se v názvech rozhraní JNDI (Java Naming and Directory Interface) podporují pouze znaky ASCII.

Sdílené fronty příjmu

Různé vazby exportu a importu očekávají, že jsou veškeré zprávy v jejich nakonfigurované frontě příjmu určeny pro daný export nebo import. Vazby importu a exportu by měly být nakonfigurovány se zohledněním těchto aspektů:
- Každá vazba importu musí mít jinou frontu příjmu, protože vazba importu předpokládá, že jsou všechny zprávy ve frontě příjmu odpovědi na požadavky, které odeslala. Pokud je fronta příjmu sdílená více importy, mohl by odpovědi přijímat nesprávný import a nebyly by korelovány s původními zprávami požadavků.
- Každý export by měl mít jinou frontu příjmu, protože jinak nelze předpovědět, který export dostane kterou zprávu požadavku.
- Importy a exporty mohou ukazovat na stejnou frontu odesílání.

Obchodní objekty

Průmyslové odvětví vývoje počítačového softwaru vyvinulo několik programovacích modelů a rámců, v rámci kterých poskytují *obchodní objekty* přirozenou reprezentaci obchodních dat pro účely zpracování aplikacemi.

Obecně vzato, tyto obchodní objekty:

- Jsou definované pomocí odvětvových standardů.
- Transparentně mapují data na databázové tabulky či podnikové informační systémy.
- Podporují protokoly vzdáleného vyvolání.
- Poskytují základnu modelu datového programování pro účely programování aplikací.

Z perspektivy nástrojů poskytuje produkt Integration Designer vývojářům jeden takový společný model obchodních objektů pro účely reprezentace různých druhů obchodních entit z různých domén. Při vývoji umožňuje tento model vývojářům definovat obchodní objekty jako definice schématu XML.

Za běhu jsou obchodní data definovaná definicemi schématu XML reprezentována jako Obchodní objekty Java. V rámci tohoto modelu jsou obchodní objekty volně založené na raných konceptech specifikace SDO (Service Data Object) a poskytují kompletní sadu aplikačních rozhraní programovacích modelů potřebných k manipulaci s obchodními daty.

Definování obchodních objektů

Obchodní objekty definujete pomocí editoru obchodních objektů z produktu Integration Designer. Editor obchodních objektů ukládá obchodní objekty jako definice schémat XML.

Použití schématu XML k definování obchodních objektů poskytuje několik výhod:

- Schémata XML poskytují standardizovaný model definice dat a základu pro součinnost různorodých systémů a aplikací. Schémata XML se používají ve spojení s jazykem WSDL (Web Services Description Language), a poskytují tak standardizované smlouvy rozhraní mezi komponentami, aplikacemi a systémy.
- Schémata XML definují bohatý model definice dat pro účely představování obchodních dat. Tento model obsahuje mimo jiné funkce pro komplexní typy, jednoduché typy, typy definované uživatelem, dědičnost typů či kardinalitu.
- Obchodní objekty mohou být definovány obchodními rozhraními a daty definovanými v jazyce WSDL (Web Services Description Language), stejně jako schématem XML od organizací odvětvového standardu nebo od jiných organizací či systémů. Produkt Integration Designer dokáže tyto obchodní objekty importovat přímo.

Produkt Integration Designer také poskytuje podporu vyhledávání obchodních dat v databázích a podnikových informačních systémech a následné generování standardizované definice obchodního objektu schématu XML na základě těchto obchodních dat. Tímto způsobem generované obchodní objekty jsou často označovány jako *obchodní objekty specifické pro aplikace*, protože napodobují strukturu obchodních dat definovaných v podnikovém informačním systému.

Když proces manipuluje daty z mnoha různých informačních systémů, může být hodnotným krokem transformovat různorodé reprezentace obchodních dat (např. ZákaznikEIS1 A ZákaznikEIS2 nebo PořadíEIS1 a Pořadí EIS2) do jediné kanonické reprezentace (např. Zákazník a Pořadí). Kanonická reprezentace je často označována jako *generický obchodní objekt*.

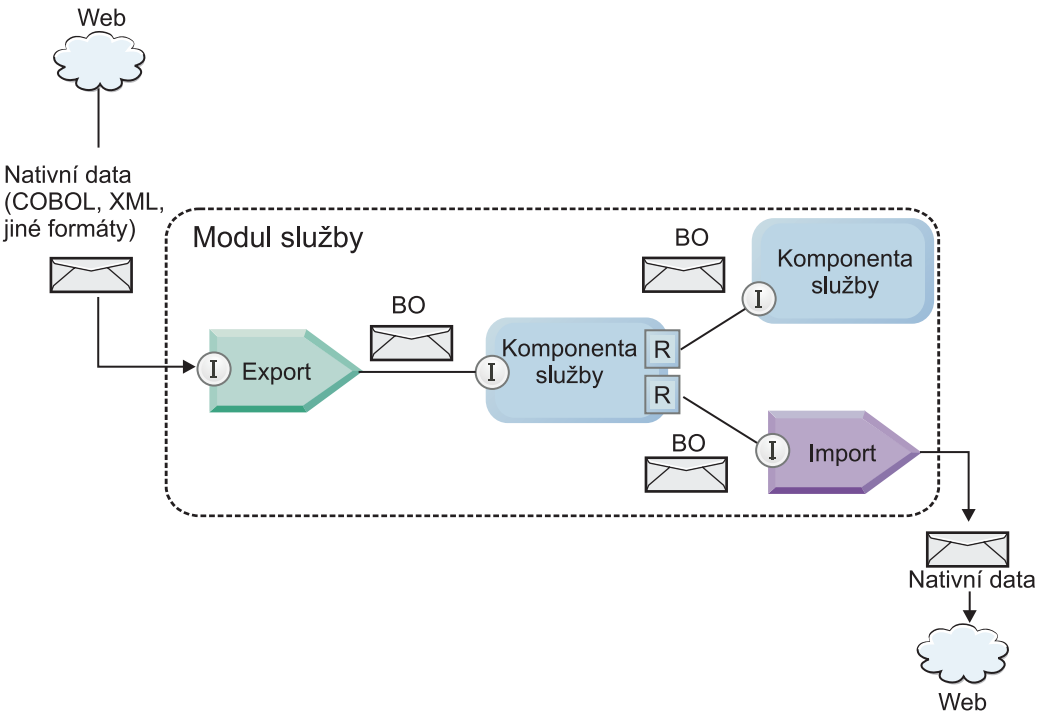
Definice obchodních objektů, obzvláště potom generických obchodních objektů, často používá více než jedna aplikace. V zájmu podpory tohoto opětovného používání umožňuje produkt Integration Designer vytvoření obchodních objektů v knihovnách, které lze přidružit k více modulům aplikací.

Jazyk WSDL (Web Services Description Language) definuje smlouvy ke službám poskytovaným a přijímaným modulem aplikace SCA (Service Component Architecture), stejně jako smlouvy používané k vytvoření komponent v rámci modulu aplikace. V rámci smlouvy může jazyk WSDL reprezentovat jak operace, tak i obchodní objekty (definované schématem XML za účelem reprezentace obchodních dat).

Práce s obchodními objekty

Architektura SCA (Service Component Architecture) poskytuje rámec pro definování modulu aplikace, služeb, které tento modul poskytuje, služeb, které přijímá, stejně jako kompozici komponent poskytujících jeho obchodní logiku. Obchodní objekty hrají v případě aplikací důležitou roli. Definují obchodní data používaná k popisu smluv dané služby a komponenty, a také obchodní data, se kterými tyto komponenty manipulují.

Následující diagram znázorňuje modul aplikace SCA a vykresluje řadu míst, na kterých vývojář pracuje s obchodními objekty.



Obrázek 41. Obchodní objekty reprezentují data tekoucí mezi službami v rámci určité aplikace

Poznámka: Toto téma popisuje, jak moduly aplikace SCA používají obchodní objekty. Pokud používáte rozhraní Java, mohou moduly aplikace SCA zpracovat také objekty jazyka Java.

Programovací model obchodních objektů

Programovací model obchodního objektu sestává ze sady rozhraní Java, která představují:

- Definici obchodního objektu a data instance.
- Sadu služeb podporujících operace s obchodními objekty.

Definice typů obchodních objektů reprezentují rozhraní commonj.sdo.Type a commonj.sdo.Property. Programovací model obchodních objektů poskytuje sadu pravidel mapování informací o komplexním typu schématu XML na rozhraní typu a mapování každého prvku z definice komplexních typů na rozhraní vlastností.

Instance obchodních objektů jsou reprezentována rozhraním commonj.sdo.DataObject. Programovací model obchodních objektů nepoužívá typy, což znamená, že lze stejné rozhraní commonj.sdo.DataObject použít k reprezentaci různých definic obchodních objektů, jako např. Zákazník či Pořadí. Definici, jejíž vlastnosti lze nastavit a získat z každého obchodního objektu, určují informace o typu definované ve schématu XML přidruženému ke každému obchodnímu objektu.

Chování programovacího modelu obchodního objektu je založeno na specifikaci Service Data Object 2.1. Další informace viz objekt SDO 2.1 for Java, výukové programy a dokumenty javadoc na webu: http://osoa.org/display/Main/Service+Data+Objects+Specifications.

Služby obchodních objektů podporují různé operace životního cyklu (jako např. vytvoření, rovnost, syntaktickou analýzu a serializaci) s obchodními objekty.

Specifika modelu programování obchodních objektů viz téma Programování pomocí služeb obchodních objektů a Generovaná dokumentace rozhraní API a SPI týkající se obchodních objektů.

Vazby, vázání dat a popisovače dat

Jak dokládá názorná ukázka, viz Obrázek 41 na stránce 140, jsou obchodní data používaná k vyvolání služeb poskytovaných moduly aplikace SCA transformována v obchodní objekty, aby tak mohly komponenty SCA obchodními daty manipulovat. Podobně jsou obchodní objekty, kterými manipulují komponenty SCA, převedeny do datového formátu, který vyžadují externí služby.

V některých případech, jako např. u vazeb webové služby, vazba používaná k exportu a importu služeb automaticky transformuje data do příslušného formátu. V jiných případech, např. u vazeb služeb JMS, mohou vývojáři poskytnout vázání dat nebo popisovač dat, který převede neintegrované formáty v obchodní objekty reprezentované rozhraním objektu DataObject.

Další informace o vývoji vázání dat a popisovačů dat viz “Popisovače dat” na stránce 55 a “Vázání dat” na stránce 56.

Komponenty

Komponenty SCA definují své smlouvy služeb spotřeby a zajišťování kombinací jazyka WSDL (Web Services Description Language) a schématu XML. Obchodní data, která architektura SCA předává mezi komponentami, jsou prostřednictvím rozhraní objektu DataObject reprezentovány jako obchodní objekty. Architektura SCA ověřuje, zda jsou tyto typy obchodních objektů kompatibilní se smlouvou rozhraní, kterou definovala vyvolávaná komponenta.

Abstrakce programovacího modelu za účelem manipulace obchodními objekty se u jednotlivých komponent liší. Komponenta POJO a vlastní primitivum komponenty mediačního toku poskytují přímou manipulaci obchodními objekty tím, že prostřednictvím programovacích rozhraní a služeb obchodních objektů umožňují přímé programování v jazyce Java. Většina komponent poskytuje abstrakce vyšší úrovně pro účely manipulace obchodními objekty, ale také úseky kódu jazyka Java pro účely definování vlastního chování v rámci rozhraní a služeb obchodních objektů.

Obchodní objekty lze transformovat buď pomocí kombinace komponent Mediace toku rozhraní a Mapa obchodních objektů, nebo pomocí kombinace komponenty mediačního toku a primitiva Mapa XML. Tyto schopnosti transformace obchodních objektů jsou užitečné při převodu obchodních objektů specifických pro aplikace do generických obchodních objektů a z nich.

Speciální obchodní objekty

Objekty SMO a obchodní grafy jsou dva specializované typy obchodních objektů používané pro specifické účely aplikací.

Objekt SMO

Objekt SMO (Service Message Object) je specializovaný obchodní objekt, který používají komponenty mediačního toku k reprezentaci kolekce dat přidružených k vyvolání služby.

Objekt SMO má pevnou strukturu nejvyšší úrovně sestávající ze záhlaví, kontextu, těla a příloh (pokud existují).

- Záhlaví nesou informace týkající se vyvolání služby prostřednictvím určitého protokolu nebo vazby. Příkladem jsou záhlaví zpráv SOAP a platformy JMS.
- Při zpracování prostřednictvím komponenty mediačního toku nesou kontextová data další logické informace přidružené k danému vyvolání. Tyto informace zpravidla nejsou součástí dat aplikace, která odesílají či přijímají klienti.
- Tělo objektu SMO nese obchodní data informačního obsahu zastupující zprávu základní aplikace nebo data vyvolání ve formě standardního obchodního objektu.

Objekt SMO může také nést data příloh pro účely vyvolání webových služeb pomocí zprávy SOAP s přílohami.

Mediační tok provádí úlohy, jako je směrování požadavku či transformace dat, a objekt SMO poskytuje kombinovaný pohled na obsahy záhlaví a informační obsahy v rámci jediné unifikované struktury.

Obchodní graf

Obchodní graf je speciální obchodní objekt používaný za účelem poskytování podpory synchronizace dat v rámci scénářů integrace.

Zvažte příklad, kdy dva podnikové informační systémy obsahují reprezentaci specifické objednávky. Pokud se v jednom ze systémů daná objednávka změní, lze do druhého systému odeslat zprávu za účelem synchronizace dat této objednávky. Obchodní grafy podporují takové pojetí, kdy je do druhého systému odeslána pouze změněná část objednávky společně s anotací v podobě souhrnných informací o změně, které definují její typ.

V tomto příkladu by obchodní graf Objednávka druhému systému sdělil, že došlo k odstranění jedné z položek a že byla aktualizována vlastnost předpokládaný datum expedice.

Obchodní grafy lze snadno přidávat do existujících obchodních objektů v produktu Integration Designer. Nejčastěji se s nimi setkáte ve scénářích, ve kterých jsou používány adaptéry WebSphere a také v případě podpory migrace aplikací serveru WebSphere InterChange Server.

Režim syntaktické analýzy obchodního objektu

Produkt Integration Designer poskytuje vlastnost modulů a knihoven, kterou lze použít k nakonfigurování režimu syntaktické analýzy XML pro obchodní objekty na hodnotu Urychleně nebo Pomalu.

- Pokud je tato volba nastavena na hodnotu *Urychleně*, dojde k vytvoření obchodního objektu prostřednictvím urychlené analýzy proudů bajtů XML.
- Pokud je tato volba nastavena na hodnotu *Pomalu*, je obchodní objekt vytvořen normálně, ale skutečná analýza proudu bajtů XML je odložena a dojde pouze k částečné analýze při přístupu k vlastnostem daného obchodního objektu.

V obou režimech syntaktické analýzy XML jsou za účelem vytvoření obchodního objektu data jiná než XML vždy urychleně analyzována.

Aspekty volby režimu syntaktické analýzy obchodního objektu:

Režim syntaktické analýzy obchodního objektu určuje, jakým způsobem je prováděna syntaktická analýza XML dat v době běhu. Režim syntaktické analýzy obchodního objektu je definován v modulu nebo knihovně při jejich vytvoření. Režim syntaktické analýzy modulu nebo knihovny můžete změnit, avšak měli byste si uvědomit možné důsledky.

Režim syntaktické analýzy obchodního objektu je nastaven na úrovni modulu a knihovny. Moduly, které byly vytvořeny v produktu IBM Integration Designer nižší verze než verze 7, poběží v režimu časné syntaktické analýzy bez nutnosti jakýchkoli změn. Standardně bude modulům a knihovnám vytvořeným v produktu IBM Integration Designer verze 7 a novějších poskytnut nejvhodnější režim syntaktické analýzy v závislosti na počtu faktorů, například režim syntaktické analýzy stávajících projektů ve vašem pracovním prostoru nebo režim syntaktické analýzy závislých projektů nebo jiných projektů ve stejném řešení atd. Režim syntaktické analýzy obchodního objektu pro modul nebo knihovnu můžete změnit tak, aby vyhovoval vaší implementaci, avšak měli byste vzít do úvahy následující aspekty.

Aspekty

- Režim zpožděné syntaktické analýzy obchodního objektu rychleji zpracovává XML data; avšak mezi režimem časné a zpožděné syntaktické analýzy existují rozdíly v kompatibilitě, které musíte vzít do úvahy před změnou konfigurace modulu nebo knihovny. Tyto rozdíly ovlivní chování modulů v době běhu. Informace o tom, který režim syntaktické analýzy je optimální pro vaši aplikaci, naleznete v tématu "Výhody použití režimu zpožděné syntaktické analýzy oproti použití režimu časné syntaktické analýzy".
- Modul lze konfigurovat pro provoz pouze v jednom z režimů syntaktické analýzy. Knihovny mohou být konfigurovány tak, aby podporovaly jen některý z režimů syntaktické analýzy nebo oba režimy. Knihovna nakonfigurovaná pro podporu obou režimů syntaktické analýzy může být odkazována modulem využívajícím režim

časné syntaktické analýzy i modulem, který využívá režim zpožděné syntaktické analýzy. Režim syntaktické analýzy pro danou knihovnu v době běhu je určen moduly, které na tuto knihovnu odkazují. V době běhu modul deklaruje svůj režim syntaktické analýzy a tento režim je využíván modulem a všemi knihovnami, které modul používá.

- Moduly a knihovny, které jsou nakonfigurovány pro různé režimy syntaktické analýzy, jsou kompatibilní v následujících případech:
  - Moduly a knihovny nakonfigurované s režimem zpožděné syntaktické analýzy jsou kompatibilní s knihovnami, které využívají buďto režim zpožděné syntaktické analýzy, nebo režim zpožděné i režim časné syntaktické analýzy.
  - Moduly a knihovny nakonfigurované s režimem časné syntaktické analýzy jsou kompatibilní s knihovnami, které využívají buďto režim časné syntaktické analýzy, nebo režim zpožděné i režim časné syntaktické analýzy.
  - Knihovny nakonfigurované s režimy časné i zpožděné syntaktické analýzy jsou kompatibilní pouze s knihovnami, které používají režim časné i režim zpožděné syntaktické analýzy.
- Používejte stejný režim syntaktické analýzy pro moduly ve vzájemné interakci, které komunikují pomocí vazby SCA. Jestliže moduly komunikují pomocí různých režimů syntaktické analýzy, mohou se vyskytnout problémy s výkonností.

Související pojmy:

"Výhody používání režimu pomalé versus urychlené syntaktické analýzy"
Pro některé aplikace je výhodný režim pomalé syntaktické analýzy XML, zatímco jiné dosáhnou lepšího výkonu v režimu časné syntaktické analýzy. Doporučujeme vaši aplikaci otestovat typovou úlohou v obou režimech syntaktické analýzy, abyste mohli určit, který z daných režimů nejlépe vyhovuje specifické charakteristice vaší aplikace.

Výhody používání režimu pomalé versus urychlené syntaktické analýzy:

Pro některé aplikace je výhodný režim pomalé syntaktické analýzy XML, zatímco jiné dosáhnou lepšího výkonu v režimu časné syntaktické analýzy. Doporučujeme vaši aplikaci otestovat typovou úlohou v obou režimech syntaktické analýzy, abyste mohli určit, který z daných režimů nejlépe vyhovuje specifické charakteristice vaší aplikace.

Aplikace provádějící analýzu rozsáhlých datových proudů XML budou pravděpodobně moci těžit ze zlepšení výkonu při použití režimu pomalé syntaktické analýzy XML. Výhody vyššího výkonu se zvyšují úměrně se zvyšováním velikosti proudu bajtů XML a snižováním množství dat z daného proudu dat, ke kterým aplikace přistupuje.

Poznámka: Režim pomalé syntaktické analýzy obchodních objektů podporuje server WebSphere Process Server verze 7.0.0.3 a novější. Také jej podporuje server IBM Process Server. Moduly a mediační moduly obsahující komponenty mediačního toku podporovány nejsou.

Následující aplikace pravděpodobně podají lepší výkon v režimu časné syntaktické analýzy:

- Aplikace analyzující datové proudy jiné než XML.
- Aplikace, které používají zprávy vytvořené pomocí služby BOFactory.
- Aplikace analyzující velmi malé zprávy XML.

Související odkazy:

"Aspekty volby režimu syntaktické analýzy obchodního objektu" na stránce 142
Režim syntaktické analýzy obchodního objektu určuje, jakým způsobem je prováděna syntaktická analýza XML dat v době běhu. Režim syntaktické analýzy obchodního objektu je definován v modulu nebo knihovně při jejich vytvoření. Režim syntaktické analýzy modulu nebo knihovny můžete změnit, avšak měli byste si uvědomit možné důsledky.

Aspekty migrace a vývoje aplikací:

Pokud konfigurujete aplikaci původně vyvinutou pro využití režimu urychlené syntaktické analýzy tak, aby nyní používala režim pomalé syntaktické analýzy, nebo pokud plánujete přepnout aplikaci mezi režimem pomalé a urychlené analýzy, mějte na paměti rozdíly mezi těmito režimy a aspekty přepínání režimů.

Ošetřování chyb

Pokud je analyzovaný proud bajtů XML chybně vytvořený, dojde k výjimkám analýzy.

- V režimu urychlené syntaktické analýzy XML k těmto výjimkám dojde, jakmile je obchodní objekt analyzován z příchozího proudu XML.
- Pokud je nakonfigurován režim pomalé syntaktické analýzy XML, vyskytují se výjimky analýzy latentně při přístupu k vlastnostem obchodních objektů, kdy je analyzována chybně vytvořená část kódu XML.

Chcete-li se s chybně vytvořeným kódem XML vypořádat, vyberte jednu z následujících voleb:

- Implementujte sběrnici ESB na hranách, a ověřte tak příchozí XML.
- Autorizujte logiku pomalé detekce chyb v bodě, kde dochází k přístupu k vlastnostem obchodního objektu.

Zásobník a zprávy výjimek

Vzhledem k tomu, že má pomalý a urychlený režim syntaktické analýzy XML každý jiné základní implementace, obsahují sice trasování zásobníku vrácené programovacími rozhraními a službami obchodních objektů stejný název třídy výjimek, ale nemusí obsahovat stejnou zprávu výjimky nebo zabalenou sadu tříd výjimek specifických pro implementaci.

Formát serializace XML

Režim pomalé syntaktické analýzy XML poskytuje optimalizaci výkonu, která se při serializaci pokouší zkopírovat nepozměněné XML z příchozího proudu bajtů do odchozího proudu bajtů. Výsledkem je zvýšení výkonu, ale formát serializace odchozího proudu bajtů XML se může lišit, pokud byl celý obchodní objekt aktualizován v režimu pomalé syntaktické analýzy XML, nebo pokud tento objekt běžel v režimu urychlené syntaktické analýzy XML.

Ačkoli formát serializace XML nemůže být přesně syntakticky ekvivalentní, je sémantická hodnota poskytnutá obchodním objektem ekvivalentní bez ohledu na zvolený režim analýzy. XML lze tedy bezpečně předat mezi aplikacemi spuštěnými v různých režimech syntaktické analýzy se sémantickou ekvivalencí.

Validátor instance obchodního objektu

Validátor instancí režimu pomalé syntaktické analýzy obchodního objektu poskytuje vyšší přesnost ověřování obchodních objektů, zejména ověřování fazet hodnot vlastností. Díky těmto vylepšením zachytává validátor instancí režimu pomalé syntaktické analýzy další problémy, které urychlený režim analýzy nezachytí a poskytuje podrobnější chybové zprávy.

Mapy XML verze 602

Mediační tok původně vyvinutý před verzí 6.1 produktu WebSphere Integration Developer může obsahovat primitiva Mapování obsahující mapu či šablonu stylů, kterou nelze přímo v režimu pomalé syntaktické analýzy XML provést. Když je aplikace převáděna migrací za účelem jejího použití v režimu pomalé syntaktické analýzy XML, může soubory mapování přidružené k primitivům Mapování automaticky aktualizovat průvodce migrací, aby mohly být spuštěny v novém režimu. Pokud ovšem primitivum Mapování odkazuje přímo na ručně upravenou šablonu stylů, daná šablona převedena migrací není a nemůže tudíž být spuštěna v režimu pomalé syntaktické analýzy XML.

Soukromá nepublikovaná rozhraní API

Pokud určitá aplikace využívá nepublikovaná, soukromá, pro implementaci specifická programovací rozhraní obchodních objektů, pravděpodobně v případě přepnutí režimu analýzy selže její kompilace. V režimu urychlené syntaktické analýzy jsou tato soukromá rozhraní zpravidla třídy implementace obchodních objektů definovaná rámcem EMF (Eclipse Modeling Framework).

Ve všech případech doporučujeme z aplikace odebrat soukromá rozhraní API.

Rozhraní API rámce EMF objektu SMO

Mediační komponenta produktu IBM Integration Designer poskytuje schopnost manipulovat obsahem zpráv pomocí tříd a rozhraní Java poskytnutých v balíku com.ibm.websphere.sibx.smobo. V případě režimu pomalé syntaktické analýzy XML lze stále použít rozhraní Java z balíku com.ibm.websphere.sibx.smobo, ale metody odkazující přímo na třídy a rozhraní rámce EMF (Eclipse Modeling Framework) nebo metody zděděné z rozhraní EMF pravděpodobně selžou.

V režimu pomalé syntaktické analýzy XML nelze objekt ServiceMessageObject a jeho obsah přetypovat na objekty rámce EMF.

Služba BOMode

Služba BOMode se používá k určení, zda je aktuálně prováděný režim syntaktické analýzy XML pomalý nebo urychlený.

Migrace

Všechny aplikace předcházející verzi 7.0.0.0 běží v režimu urychlené syntaktické analýzy XML. Když jsou migrovány za běhu prostřednictvím nástrojů migrace za běhu řízení BPM, jsou nadále spuštěny v režimu urychlené syntaktické analýzy XML.

Pokud chcete umožnit konfiguraci aplikace verze předcházející verzi 7.0.0.0 pro použití režimu pomalé syntaktické analýzy XML, musíte nejprve pomocí produktu Integration Designer migrovat její artefakty. Po migraci potom aplikaci nakonfigurujte pro použití pomalé syntaktické analýzy XML.

V tématu Migrace zdrojových artefaktů najdete informace o migraci artefaktů pomocí produktu Integration Designer a v tématu Konfigurace režimu syntaktické analýzy obchodního objektu pro moduly a knihovny najdete informace o nastavení režimu syntaktické analýzy.

Vztahy

Vztah je přidružení mezi dvěma nebo více datovými entitami, zpravidla obchodními objekty. V produktu IBM Business Process Manager Advanced lze vztahy použít k transformaci dat, která jsou ekvivalentní napříč obchodními objekty, a jiných dat, která jsou však reprezentována odlišně. Lze je také použít ke kreslení přidružení napříč různými objekty v různých aplikacích. Vztahy mohou být sdíleny mezi aplikacemi, řešeními a dokonce i mezi produkty.

Infrastrukturu a operace pro správu vztahů v produktu poskytuje IBM Business Process Manager Advanced služba vztahů. Protože tato služba umožňuje zacházet s obchodními objekty bez ohledu na to, kde se nacházejí, může poskytnout komplexní pohled na všechny aplikace v podniku a sloužit jako stavební kámen řešení řízení BPM. Jelikož vztahy jsou rozšiřitelné a spravovatelné, mohou být použity v komplexních řešeních integrace.

Co jsou vztahy?

Vztah je přidružení mezi obchodními objekty. Jednotlivé obchodní objekty ve vztahu se nazývají *účastníci* vztahu. Každý z účastníků vztahu je odlišen od ostatních účastníků funkcí neboli *rolí*, kterou v daném vztahu plní. Vztah obsahuje seznam rolí.

Jednotlivé role a jejich vzájemné vztahy popisuje *definice* vztahu. Definice rovněž popisuje celkový "tvar" vztahu. Například jedna role může mít pouze jediného účastníka, zatímco jiná role může mít libovolný potřebný počet účastníků. Můžete například definovat vztah *automobil-vlastník*, v němž může jeden vlastník vlastnit více automobilů. Například jedna instance může mít v jednotlivých rolích tyto účastníky:

- Automobil (Ferrari).
- Vlastník (John).

Definice vztahu představuje šablonu pro *instanci* vztahu. Instance je běhová konkretizace vztahu. V uvedeném příkladu vztahu *automobil-vlastník* může instance popisovat kterékoli z následujících přidružení:

- John vlastní automobil Ferrari.
- Sara vlastní automobil Mazda.
- Bob vlastní automobil Ferrari.

Použití vztahů odstraňuje potřebu sestavení vlastní perzistence sledování vztahů v rámci vaší obchodní logiky. U některých scénářů za vás dělá veškerou práci služba vztahů. Viz příklad popsaný v oddílu Vztahy identity.

Scénáře

Toto je typický příklad situace, ve které může řešení integrace používat vztahy. Velká společnost kupuje více společností neboli obchodních jednotek. Každá z těchto obchodních jednotek používá jiný software ke sledování pracovníků a laptopů. Společnost potřebuje způsob, jak sledovat zaměstnance a jejich laptopy. Potřebuje řešení, které jí umožní:

- Zobrazit všechny zaměstnance v různých obchodních jednotkách, jako kdyby se nacházeli v jediné databázi.
- Získat jediný pohled na všechny jejich laptopy.
- Povolit zaměstnancům přihlášení do systému a koupi laptopu.
- Pojmout různé systémy podnikových aplikací v různých obchodních jednotkách.

Za tímto účelem potřebuje společnost zajistit, aby například John Smith a John A. Smith v různých aplikacích byli považováni za téhož zaměstnance. Potřebuje způsob, jak konsolidovat jedinou entitu napříč prostory několika aplikací.

Složitější scénáře vztahů zahrnují sestavení procesů BPEL, které kreslí vztahy napříč různými objekty v několika aplikacích. U složitých scénářů vztahů se obchodní objekty nacházejí v integračním řešení, nikoli tedy v aplikacích. Platformu pro trvalou správu vztahů poskytuje služba vztahů. Před službou vztahů byste měli sestavit vlastní službu perzistence objektů. Dva příklady složitých scénářů vztahů:

- Máte obchodní objekt **car** s číslem VIN v aplikaci SAP a chcete sledovat skutečnost, že tento automobil vlastní někdo jiný. Nicméně vztah vlastnictví je s někým v aplikaci PeopleSoft. V takovém vzorci vztahů máte dvě řešení a potřebujete mezi nimi vybudovat most.
- Velká maloobchodní společnost chce monitorovat vratky zboží výměnou za hotovost nebo kredit. Zapojeny jsou dvě aplikace: systém správy objednávek (OMS) pro nákupy a systém správy vratek (RMS) pro vratky. Obchodní objekty se nacházejí ve více než jedné aplikaci, a proto potřebujete způsob, jak znázornit vztahy, které mezi nimi existují.

obecné vzory využívání

Nejběžnějším vzorcem vztahů jsou *ekvivalence*. Ty jsou založeny na křížovém odkazování či korelaci. Tomuto vzorku odpovídají dva typy vztahů: *bez identity* a *identita*.

- **Vztahy bez identity** navazují přidružení mezi obchodními objekty či jinými daty na bázi jedna ku n nebo n ku n. Pro každou instanci vztahu může existovat jedna či více instancí každého účastníka. Jedním typem vztahu bez identity je statický vyhledávací vztah. Příkladem může být vztah, ve kterém **CA** v aplikaci SAP souvisí s objektem **California** v aplikaci Siebel.
- **Vztahy identity** navazují přidružení mezi obchodními objekty či jinými daty na bázi jedna ku jedné. Pro každou instanci vztahu může existovat pouze jedna instance každého účastníka. Vztahy identity zachycují křížové odkazy mezi obchodními objekty, které jsou sice sémanticky ekvivalentní, ale v různých aplikacích jsou identifikovány různě. Každý účastník vztahu je přidružen k obchodnímu objektu, jenž má určitou hodnotu (nebo kombinaci hodnot), která jej jedinečně identifikuje. Vztahy identity většinou transformují klíčové atributy obchodních objektů, jako jsou číselné identifikátory a kódy produktů.

  Máte-li například v aplikacích SAP, PeopleSoft a Siebel obchodní objekty **car** a chcete-li vytvořit řešení, které je synchronizuje, zpravidla by bylo třeba zavést ručně vytvořenou logiku synchronizace vztahů v šesti mapách:

  SAP -> generický

generický -> SAP

PeopleSoft-> generický

generický-> PeopleSoft

Siebel-> generický

generický-> Siebel

Pokud však ve svém řešení použijete vztahy, poskytne služba vztahů předem sestavené implementace vzorců, které vám zachovají všechny tyto instance vztahů.

Nástroje pro práci se vztahy

Nástrojem pro modelování a návrh vztahů a rolí obchodní integrace je *editor vztahů* v produktu Integration Designer. Podrobné informace o pozadí a úlohách souvisejících s vytvářením vztahů a použitím editoru vztahů najdete v tématu Vytvoření vztahů.

Službou infrastruktury v produktu IBM Business Process Manager, která udržuje vztahy a role v systému a poskytuje operace pro správu vztahů a rolí, je *služba vztahů*.

Rozhraním pro správu vztahů je *Správce vztahů*. Lze k němu přistupovat prostřednictvím stránek Správce vztahů v administrativní konzole.

Vztahy lze programově vyvolávat prostřednictvím rozhraní API služby vztahů.

Služba vztahů

Služba vztahů ukládá data vztahů v tabulkách vztahů, kde napříč aplikacemi a řešeními sleduje hodnoty specifické pro různé aplikace. Služba vztahů poskytuje operace pro správu vztahů a rolí.

Princip činnosti vztahů

Vztahy a role jsou definovány pomocí grafického rozhraní nástroje editoru vztahů v produktu Integration Designer. Služba vztahů ukládá korelační data do tabulek v databázi vztahů ve výchozím zdroji dat, který určíte při konfiguraci služby vztahů. Oddělená tabulka (někdy se nazývá tabulka účastníků) ukládá informace o jednotlivých účastnících vztahu. Na základě těchto tabulek vztahů služba vztahů sleduje související hodnoty specifické pro různé aplikace a šíří aktualizované informace napříč všemi řešeními.

Vztahy, což jsou obchodní artefakty, jsou implementovány v rámci projektu nebo ve sdílené knihovně. Při první implementaci služba vztahů provede naplnění daty.

Pokud mapy či jiné komponenty produktu IBM Business Process Manager požadují za běhu instanci vztahu, jsou v závislosti na scénáři aktualizovány nebo načteny instance vztahu.

Manipulace s daty instance role či vztahu je možná třemi způsoby:

- Vyvoláním rozhraní API služby vztahů úryvkem kódu Java komponenty produktu IBM Business Process Manager.
- Transformací vztahu ve službě mapování obchodních objektů produktu IBM Business Process Manager.
- Nástrojem Správce vztahů.

Podrobné informace o pozadí a úlohách souvisejících s vytvářením vztahů, identifikací typů vztahů a použitím editoru vztahů viz tématu Vytvoření vztahů.

Správce vztahů

Správce vztahů představuje rozhraní pro správu vztahů. Lze k němu přistupovat prostřednictvím stránek Správce vztahů v administrativní konzole.

Správce vztahů poskytuje grafické uživatelské rozhraní pro vytváření a manipulaci se vztahy a daty rolí za běhu. Entity vztahů můžete spravovat na všech úrovních: instance vztahu, instance role a úrovně dat atributu a dat vlastností. Správce vztahů umožňuje:

- Zobrazit seznam vztahů v systému a podrobné informace o jednotlivých vztazích.
- Spravovat instance vztahů:
  - Dotazovat se na data vztahů za účelem zobrazení podmnožin dat instancí.
  - Dotazovat se na data vztahů za účelem zobrazení podmnožin dat instancí pomocí pohledů databáze.
  - Zobrazit seznam instancí vztahů, které odpovídají dotazu na vztah, včetně podrobných informací o instanci.
  - Upravit hodnoty vlastností pro instanci vztahu.
  - Vytvořit a odstranit instance vztahu.
- Spravovat role a instance role:
  - Zobrazit podrobnosti o roli nebo instanci role.
  - Upravit vlastnosti instance role.
  - Vytvořit a odstranit instance role pro určitý vztah.
  - Odvolat data instance vztahu do časového bodu, o němž víte, že v něm byla data spolehlivá.
- Importovat data z existujícího statického vztahu do systému nebo exportovat data z existujícího statického vztahu do souboru RI nebo CSV.
- Odebrat data a schéma vztahů z úložiště po odinstalaci aplikace, která je používá.

Vztahy v prostředích síťové implementace

V prostředích síťové implementace (ND) lze vztahy používat bez jakékoli další konfigurace.

V prostředích síťové implementace (ND) jsou vztahy nainstalovány v klastru aplikací. Poté jsou vztahy viditelné v rámci klastru, přičemž k datům instancí uloženým v databázi vztahů mají přístup všechny servery v tomto klastru. Schopnost spuštění služby vztahů v prostředí ND zajišťuje rozšiřitelnost a vysokou dostupnost.

Správce vztahů umožňuje spravovat vztahy napříč různými klastry prostřednictvím centralizovaného rozhraní pro správu. Správce vztahů připojíte k serveru v klastru pomocí výběru jeho objektu MBean vztahu.

Rozhraní API služby vztahů

Prostřednictvím rozhraní API služby vztahů lze programově vyvolávat vztahy, a to v rámci map obchodních objektů i mimo ně.

K dispozici jsou tři typy rozhraní API:

- Rozhraní API pro manipulaci s instancí vztahu (včetně přímého vytvoření, aktualizace a odstranění dat instance).
- Rozhraní API pro podporu vzorců vztahů (včetně correlate() a correlateforeignKeyLookup).
- Vzorce pro vyhledání vztahu (rozhraní API pro vyhledání).

Sběrnice Enterprise Service Bus v produktu IBM Business Process Manager

Produkt IBM Business Process Manager podporuje integraci služeb aplikací, včetně stejných schopností jako WebSphere Enterprise Service Bus.

Spojování služeb prostřednictvím sběrnice Enterprise Service Bus

Pomocí sběrnice Enterprise Service Bus (ESB) můžete maximalizovat flexibilitu architektury SOA. Účastníci interakce služeb nejsou připojeni k sobě navzájem, ale ke sběrnici ESB.

Když se klient služby připojí ke sběrnici ESB, tato přebírá zodpovědnost za doručování (prostřednictvím zpráv) jeho požadavků na poskytovatele služeb, který nabízí požadovanou funkci a kvalitu služeb. Sběrnice ESB usnadňuje interakce mezi žadateli a poskytovateli a adresuje neshody v protokolech, vzorcích interakcí a schopnostech služeb.

Sběrnice ESB může také povolit nebo rozšířit monitorování a správu. Sběrnice ESB poskytuje funkce virtualizace a správy, které implementují a rozšiřují schopnosti jádra architektury SOA.

Sběrnice ESB abstrahuje tyto funkce:

Umístění a identita
Účastníci nemusí znát umístění ani identitu jiných účastníků. Žadatelé například nemusí vědět, že by určitý požadavek mohl obsloužit kterýkoli z několika poskytovatelů. Poskytovatele služeb lze přidávat nebo odebírat bez přerušení.

Protokol interakce
Účastníci nemusejí sdílet stejný komunikační protokol či styl interakce. Například požadavek vyjádřený jako protokol SOAP přes protokol HTTP může obsloužit poskytovatel, který rozumí pouze protokolu SOAP přes službu JMS (Java Message Service).

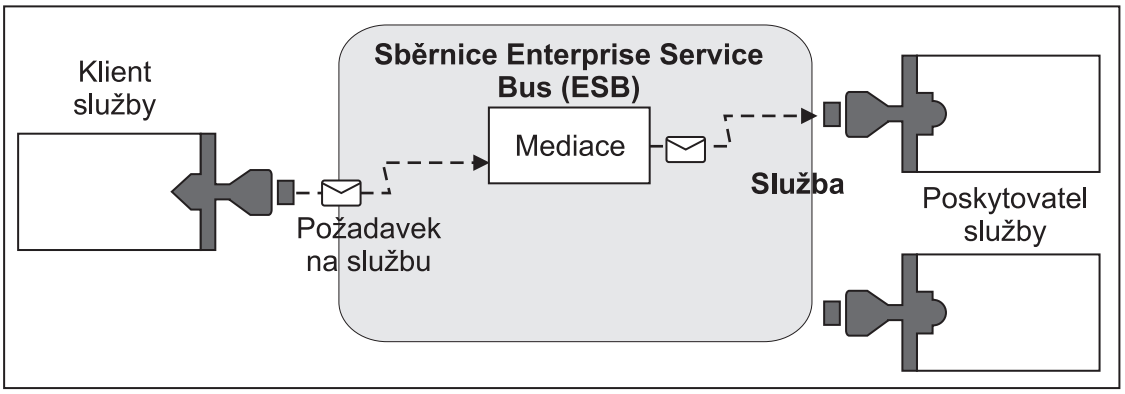
Rozhraní
Žadatelé a poskytovatelé se nemusí shodnout na společném rozhraní. Sběrnice ESB srovnává rozdíly tím, že transformuje zprávy požadavků a odpovědí do formy, kterou poskytovatel očekává.

Kvalita služeb (interakcí)
Účastníci nebo také administrátoři systému deklarují své požadavky na kvalitu služeb, včetně autorizace požadavků, šifrování a dešifrování obsahů zpráv, automatické auditování interakcí služeb, a také způsob, a jakým by jejich požadavky měly být trasovány (např. optimalizace pro rychlost či náklady).

Vložení sběrnice ESB mezi účastníky vám umožní modulovat interakce těchto účastníků pomocí logické konstrukce nazvané *mediace*. Mediace fungují u zpráv mezi žadateli a poskytovateli průběžně. Mediační toky lze například použít k nalezení služeb se specifickými charakteristikami, které hledá žadatel a k interpretaci rozdílů rozhraní mezi žadateli a poskytovateli. Pro účely komplexních interakcí lze vytvořit postupný řetězec mediací.

Pomocí mediací provádí sběrnice Enterprise Service Bus následující akce mezi žadatelem a službou:

- *Trasování* zpráv mezi službami. Sběrnice Enterprise Service Bus nabízí společnou komunikační infrastrukturu, kterou lze použít k propojení služeb, a tím také obchodních funkcí, které tyto služby reprezentují, a to bez potřeby toho, aby programátoři zapisovali a udržovali komplexní logiku konektivity.
- *Převod* přenosových protokolů mezi žadatelem a službou. Sběrnice Enterprise Service Bus poskytuje konzistentní, standardizovaný způsob integrace obchodních funkcí využívajících různé standardy IT. To umožňuje integraci obchodních funkcí, které by normálně komunikovat nemohly. Např. připojení aplikací do zásobníků v rámci různých oddělení, nebo umožnění podílení se na interakcích služeb aplikacím z různých společností.
- *Transformování* formátu zpráv mezi žadatelem a službou. Sběrnice Enterprise Service Bus umožňuje obchodním funkcím vyměňovat informace v různých formátech, přičemž sběrnice zajišťuje, že informace dodané obchodní funkci jsou ve formátu, který daná aplikace požaduje.
- *Obsluha* obchodních událostí z různorodých zdrojů. Sběrnice Enterprise Service Bus podporuje pro účely zpracování požadavků na službu vedle výměny zpráv také interakce založené na událostech.



Obrázek 42. Sběrnice Enterprise Service Bus. Sběrnice Enterprise Service Bus trasuje zprávy mezi aplikacemi, které jsou žadateli nebo poskytovateli služeb. Sběrnice převádí přenosové protokoly a transformuje formát zpráv mezi žadateli a poskytovateli. Na tomto obrázku používá každá aplikace jiný protokol (reprezentovaný jiným geometrickým tvarem konektoru) a jiný formát zpráv.

Při použití sběrnice Enterprise Service Bus se můžete zaměřit na jádro svého podnikání a nemusíte věnovat pozornost svým počítačovým systémům. Služby můžete měnit či přidávat v případě potřeby - například jako odezvu na změny v obchodních požadavcích, chcete-li přidat další kapacitu služeb nebo nové schopnosti. Požadované změny můžete provést předkonfigurováním sběrnice, což bude mít minimální, nebo vůbec žádný, dopad na existující služby a aplikace, které tuto sběrnici používají.

Infrastruktura systému zpráv sběrnice Enterprise Service Bus

Produkt IBM Business Process Manager zahrnuje schopnosti sběrnice Enterprise Service Bus. Produkt IBM Business Process Manager podporuje integraci technologií řízených událostmi a orientovaných na služby či na zprávy za účelem poskytnutí infrastruktury systému zpráv založené na standardech v integrované sběrnici Enterprise Service Bus.

Schopnosti Enterprise Services, které můžete používat v rámci podnikových aplikací, poskytují nejen transportní vrstvu, ale také podporu mediace usnadňující interakce služeb. Sběrnice Enterprise Service Bus je postavena na otevřených standardech a architektuře SOA (service-oriented architecture). Je založena na robustní infrastruktuře Java EE a přidružených službách platformy poskytované serverem IBM WebSphere Application Server Network Deployment.

Produkt IBM Business Process Manager je založen na stejné technologii dostupné se sběrnicí IBM WebSphere Enterprise Service Bus. Tato schopnost je součástí základní funkčnosti produktu IBM Business Process Manager a k využití těchto schopností není vyžadována žádná další licence na WebSphere Enterprise Service Bus.

Nicméně, kolem svého podniku můžete implementovat další samostatné licence sběrnice WebSphere Enterprise Service Bus, a tím rozšířit dosah konektivity u řešení integrace procesů založených na produktu IBM Business Process Manager. Například je možné nainstalovat sběrnici WebSphere Enterprise Service Bus blíže k aplikaci SAP za účelem hostování adaptéru IBM WebSphere Adapter for SAP a transformace zpráv SAP, předtím než budou tyto informace odeslány po síti do obchodního procesu, jehož choreografem je produkt IBM Business Process Manager.

Hostitelé systému zpráv nebo cíle fronty:

Hostitel systému zpráv nebo cíle fronty poskytuje funkci zasílání zpráv v rámci serveru. Server se stává cílovým hostitelem systému zpráv, když jej nakonfigurujete jako cíl systému zpráv.

Stroj systému zpráv je spuštěn v rámci serveru. Stroj systému zpráv poskytuje funkce zasílání zpráv a bod připojení, který aplikace používají k připojení ke sběrnici. Asynchronní komunikace architektury SCA (Service Component Architecture), importy a exporty služby JMS i asynchronní interní zpracování používají v rámci stroje systému zpráv fronty zpráv.

Při implementaci modulů aplikace připojuje prostředí implementace zdroj zprávy k cíli zprávy prostřednictvím sběrnice. Když budete znát zdroj a cíl zprávy, můžete snadněji určit potřebný typ prostředí implementace.

Aplikace mohou ukládat trvalá data v datovém úložišti, což je sada tabulek databáze či schématu, nebo v úložišti souborů. Pro účely interakce s touto databází používá stroj systému zpráv instanci zdroje dat JDBC.

Cíl systému zpráv nakonfigurujte při definování prostředí implementace prostřednictvím volby **Server** v administrativní konzole nebo server určete jako cílového hostitele během instalace softwaru.

Datová úložiště:

Každý stroj systému zpráv může používat datové úložiště, což je sada tabulek databáze či schématu, ve kterých jsou uložena trvalá data.

Všechny tabulky datového úložiště jsou zadržovány ve stejném schématu databáze. Každé datové úložiště můžete vytvořit v oddělené databázi. Můžete také vytvořit více datových úložišť v jedné databázi, ale každé z nich musí používat jiné schéma.

Stroj systému zpráv používá instanci zdroje dat JDBC k zajištění interakce s databází obsahující dané datové úložiště pro daný stroj systému zpráv.

Poskytovatelé JDBC:

Pomocí poskytovatelů JDBC lze umožnit interakci aplikací s relačními databázemi.

Aplikace využívají poskytovatele JDBC k zajištění interakce s relačními databázemi. Poskytovatel JDBC dodává specifickou implementační třídu ovladačů JDBC pro přístup ke specifickému typu databáze. Přidružením zdroje dat k poskytovateli JDBC můžete vytvořit fond připojení k této databázi. Poskytovatel JDBC spolu s objekty zdroje dat jsou funkčně shodné s továrnou připojení architektury Java EE Connector Architecture (JCA), která poskytuje konektivitu k nerelační databázi.

Viz příklady nastavení typického samostatného prostředí a nastavení typického prostředí implementace v předchozím tématu.

Podrobnější informace o poskytovatelích JDBC viz téma “Poskytovatelé JDBC” v Informačním centru aplikačního serveru WebSphere Application Server.

Sběrnice SIBus produktu IBM Business Process Manager:

Sběrnice SIBus je spravovaný mechanizmus komunikace podporující integraci služeb prostřednictvím synchronního a asynchronního zasílání zpráv. Sběrnice sestává z propojujících se strojů systému zpráv, které spravují prostředky této sběrnice. Jedná se o jednu z technologií serveru WebSphere Application Server, na kterých je založený produkt IBM Business Process Manager.

Některé používané sběrnice vytváří automaticky systém, implementované aplikace architektury SCA (Service Component Architecture), či další komponenty. Sběrnice můžete také vytvořit za účelem podpory logiky integrace služeb nebo jiných aplikací. Například kvůli podpoře aplikací, které fungují jako klienti a poskytovatelé služeb v rámci produktu IBM Business Process Manager, nebo kvůli odkazování na produkt WebSphere MQ.

Cíl sběrnice je logická adresa, ke které se mohou aplikace připojovat jako producenti, odběratelé nebo obojí. Cíl fronty je cíl sběrnice používaný pro účely systému zpráv typu point-to-point.

Každá sběrnice může mít minimálně jeden člen sběrnice, přičemž každý takový člen je buď server, nebo klastr.

Topologie sběrnice je fyzické uspořádání aplikačních serverů, strojů systému zpráv a správců front produktu WebSphere MQ, stejně jako šablona připojení sběrnice a odkazy mezi těmito připojeními, ze kterých sestává vaše sběrnice Enterprise Service Bus.

Některé sběrnice SIBus jsou vytvořeny za účelem podpory produktu IBM Business Process Manager automaticky. Když vytváříte prostředí implementace nebo konfigurujete server či klastr za účelem podpory aplikací SCA, jsou vytvořeny až šest sběrnic. Každá z těchto sběrnic má pět aliasů ověřování, které je třeba nakonfigurovat.

Systémová sběrnice SCA:

Systémová sběrnice architektury SCA je sběrnice SIBus používaná jako hostitel cílů fronty modulů architektury SCA (Service Component Architecture). Běhové prostředí architektury SCA, které podporuje mediační moduly, používá cíle systémové sběrnice jako infrastrukturu podporující asynchronní interakce mezi komponentami a moduly.

Aplikační sběrnice se vytváří automaticky při vytvoření prostředí implementace nebo při konfiguraci serveru či klastru za účelem podpory aplikací SCA. Systémová sběrnice poskytuje rozsah, v rámci kterého jsou konfigurovány prostředky, jako např. cíle fronty, pro účely mediačních modulů a koncových bodů interakcí. Sběrnice umožňuje směrování zpráv mezi koncovými body. Pro danou sběrnici můžete určit kvalitu služby, včetně její priority a spolehlivosti.

Název sběrnice je SCA.SYSTEM.busID.Bus. Alias ověřování používaný k zabezpečení této sběrnice je SCA_Auth_Alias.

Aplikační sběrnice SCA:

Cíle aplikační sběrnice podporují asynchronní komunikaci adaptérů WebSphere Business Integration Adapters a dalších komponent architektury System Component Architecture.

Aplikační sběrnice se vytváří automaticky při vytvoření prostředí implementace nebo při konfiguraci serveru či klastru za účelem podpory aplikací SCA. Aplikační sběrnice se podobá sběrnicím SIBus, které lze vytvořit za účelem podpory logiky integrace služeb nebo jiných aplikací.

Název sběrnice je SCA.APPLICATION.busID.Bus. Alias ověřování používaný k zabezpečení této sběrnice je SCA_Auth_Alias.

Sběrnice infrastruktury Common Event Infrastructure:

Sběrnice infrastruktury Common Event Infrastructure se používá k asynchronnímu přenosu událostí Common Base Event do nakonfigurovaného serveru infrastruktury Common Event Infrastructure.

Název této sběrnice je CommonEventInfrastructure_Bus. Alias ověřování používaný k zabezpečení této sběrnice je CommonEventInfrastructureJMSAuthAlias.

Sběrnice komponenty Business Process Choreographer:

Pro účely interního přenosu zpráv použijte název a ověření sběrnice komponenty Business Process Choreographer.

Sběrnice komponenty Business Process Choreographer se používá k internímu přenosu zpráv a také pro účely rozhraní API platformy JMS (Java Messaging Service) komponenty Business Flow Manager.

Název této sběrnice je BPC.cellMane.Bus. Alias ověřování je BPC_Auth_Alias

Sběrnice komponenty Performance Data Warehouse:

Sběrnice komponenty Performance Data Warehouse se používá k internímu přenosu zpráv infrastrukturou a ke komunikaci s klienty produktu IBM Business Process Manager.

Sběrnice komponenty Performance Data Warehouse se vytváří automaticky při vytvoření prostředí implementace.

Název sběrnice je PERFDW.busID.Bus. Alias ověřování používaný k zabezpečený této sběrnice je PERFDWME_Auth_Alias.

Sběrnice serveru Process Server:

Sběrnice serveru Process Server se používá k internímu přenosu zpráv infrastrukturou a ke komunikaci s klienty produktu IBM Business Process Manager.

Sběrnice serveru Process Server se vytváří automaticky při vytvoření prostředí implementace.

Název sběrnice je PROCSVR.busID.Bus. Alias ověřování používaný k zabezpečení této sběrnice je PROCSVRME_Auth_Alias.

Aplikace služeb a moduly služeb

Modul služby je modul architektury SCA(Service Component Architecture), který poskytuje služby v rámci běhového prostředí. Když implementujete modul služby do produktu IBM Business Process Manager, sestavujete přidruženou aplikaci služeb, která je zabalená v podobě souboru EAR.

Moduly služby jsou základní jednotky implementace, které mohou obsahovat komponenty, knihovny a moduly fázování používané přidruženými aplikacemi služeb. Moduly služby obsahují exporty a volitelně také importy, které definují vztahy mezi moduly a klienty i poskytovateli služeb. Produkt WebSphere Process Server podporuje moduly obchodních služeb a mediačních modulů. Moduly o mediační moduly jsou typy modulů SCA. Mediační modul umožňuje komunikace mezi aplikacemi, neboť převádí vyvolání služby do formátu, kterému cíl rozumí, předá požadavek na cíl a vrátí výsledek původci. Modul pro obchodní službu implementuje logiku obchodního procesu. Modul však může také obsahovat stejnou mediační logiku, která se balí do mediačního modulu.

Implementace aplikace služeb

Proces implementace souboru EAR obsahujícího aplikaci služeb je shodný s procesem implementace libovolného souboru EAR. Hodnoty parametrů mediace lze změnit v době implementace. Po implementaci souboru EAR obsahujícího modul SCA si můžete nechat zobrazit podrobnosti o dané aplikaci služeb a o modulu, který je k ní přidružený. Můžete vidět, jak je modul služby připojený ke klientům služby (prostřednictvím exportů) a poskytovatelům služby (prostřednictvím importů).

Zobrazení podrobností o modulu SCA

To, jaké podrobnosti o modulu služby si můžete nechat zobrazit, závisí na daném modulu SCA. Podrobnosti mohou obsahovat následující atributy.

- Název modulu SCA.
- Popis modulu SCA.
- Název přidružené aplikace.
- Informace o verzi modulu SCA, pokud je tento opatřen verzí.
- Importy modulu SCA:
  - Rozhraní importu jsou abstraktní definice popisující způsob přístupu modulu SCA ke službě.
  - Vazby exportu jsou konkrétní definice určující fyzický mechanizmus, který modul SCA používá pro přístup ke službě. Například prostřednictvím vazby SOAP/HTTP.
- Exporty modulu SCA:
  - Rozhraní exportu jsou abstraktní definice popisující způsob přístupu klienta služby k modulu SCA.
  - Vazby exportu jsou konkrétní definice určující fyzický mechanizmus, který modul klient služby používá pro přístup k modulu SCA a nepřímo také ke službě.
- Vlastnosti modulu SCA.

Importy a vazby importu:

Importy definují interakce mezi moduly SCA a poskytovateli služeb. Moduly SCA používají importy za účelem povolení přístupu komponent k externím službám (službám vyskytujícím se mimo modul SCA) za použití lokální reprezentace. Vazby importu definují specifický způsob přístupu k externí službě.

Pokud moduly SCA nepotřebují přistupovat k externím službám, nemusí mít importy. Mediační moduly mají obvykle minimálně jeden import používaný k předávání zpráv či požadavků určeným cílům.

Rozhraní a vazby

Import modulu SCA musí mít minimálně jedno rozhraní a má jedinou vazbu.

- Rozhraní exportu jsou abstraktní definice, které definují sadu operací za použití jazyka WSDL (Web Services Description Language), jazyka XML pro popis webových služeb. Modul SCA může mít řadu rozhraní importu.
- Vazby importu jsou konkrétní definice určující fyzický mechanizmus, který moduly architektury SCA používají pro přístup k externí službě.

Podporované vazby importu

Produkt IBM Business Process Manager podporuje následující vazby importu:

- Vazby SCA spojují moduly SCA s jinými moduly SCA. Vazby SCA bývají také označovány jako výchozí vazby.
- Vazby webové služby povolují komponentám vyvolávat webové služby. Podporované jsou protokoly SOAP1.1/HTTP, SOAP1.2/HTTP a SOAP1.1/JMS.

  Můžete použít vazbu SOAP1.1/HTTP nebo SOAP1.2/HTTP založenou na rozhraní Java API for XML Web Services (JAX-WS), která umožňuje interakce se službami používajícími vazby dokumentu nebo RPC/literal a která používá k úpravě vyvolání obslužné rutiny JAX-WS. Oddělená vazba SOAP1.1/HTTP je poskytována za účelem povolení interakcí se službami používajícími vazbu RPC/encoded nebo také když existuje požadavek na úpravu vyvolání za použití obslužných rutin JAX-RPC.
- Vazby HTTP povolují přístup k aplikacím za použití protokolu HTTP.
- Vazby importu objektu EJB (Enterprise JavaBeans) umožňují komponentám SCA vyvolávat služby poskytované obchodní logikou prostředí Java EE na serveru Java EE.
- Vazby podnikových informačních systémů (EIS) zajišťují konektivitu mezi komponentami SCA a externím podnikovým informačním systémem. Tuto komunikaci umožňuje použití adaptérů prostředku.
- Vazby platformy JMS (Java Message Service) verze 1.1 povolují s výchozím poskytovatelem systému zpráv serveru WebSphere Application Server. Služba JMS dokáže využívat různé typy transportu, včetně protokolu TCP/IP a HTTP či HTTPS. Automaticky je podporována třída zpráv JMS a jejích pět podtypů (text, bajty, objekt, proud a mapa).
- Generické vazby služby JMS povolují interoperabilitu s poskytovateli služby JMS od jiných dodavatelů, integrovanými do serveru WebSphere Application Server pomocí zařízení ASF (Application Server Facility) služby JMS.
- Vazby služby produktu WebSphere MQ JMS povolují interoperabilitu s poskytovateli služby JMS založenými na produktu WebSphere MQ. Automaticky je podporována třída zpráv JMS a jejích pět podtypů (text, bajty, objekt, proud a mapa). Pokud chcete použít produkt WebSphere MQ jako poskytovatele služby JMS, použijte vazby WebSphere MQ JMS.
- Vazby WebSphere MQ povolují interoperabilitu s produktem WebSphere MQ. Vazby produktu WebSphere MQ lze použít pouze pomocí vzdálených správců front, a to za použití připojení klienta produktu WebSphere. Pomocí lokálních správců front je použít nelze. Vazby produktu WebSphere MQ použijte pouze v případě, že chcete komunikovat s nativními aplikacemi produktu WebSphere MQ.

Dynamické vyvolání služeb

Služby lze vyvolat prostřednictvím jakékoli podporované vazby importu. Služba se normálně nachází v koncovém bodu určeném v importu. Tento koncový bod se nazývá statický koncový bod. Potlačením statického koncového bodu je možné vyvolat jinou vazbu. Dynamické potlačení statických koncových bodů vám umožňuje vyvolat službu v jiném

koncovém bodu, a to prostřednictvím libovolné podporované vazby importu. Dynamické vyvolání služeb vám také umožňuje vyvolat službu v případě, kdy podporované vazba importu nemá statický koncový bod.

Import s přidruženou vazbou se používá k určení protokolu a jeho konfigurace pro účely dynamického vyvolání. Import používaný pro účely dynamického vyvolání lze s volající komponentou spojit nebo jej lze dynamicky vybrat za běhu.

Pro účely vyvolání webových služeb a architektury SCA je možné provést dynamické vyvolání také bez importu a protokol a konfigurace budou odečteny u adresy URL koncového bodu. Cílový typ invokace se identifikuje na základě adresy URL koncového bodu. Pokud se použije import, musí být adresa URL kompatibilní s protokolem dané vazby importu.

- Adresa URL architektury SCA označuje vyvolání jiného modulu SCA.
- Adresa URL protokolu HTTP nebo služby JMS standardně označuje vyvolání webové služby. V případě těchto adres URL je možné poskytnout další hodnotu typu vazby, která označuje, že tato adresa URL představuje vyvolání prostřednictvím vazby HTTP nebo JMS.
- Adresa URL protokolu HTTP webové služby standardně používá protokol SOAP 1.1 a lze určit hodnotu typu vazby, která bude označovat použití protokolu SOAP 1.2.

Exporty a vazby exportu:

Exporty definují interakce mezi moduly SCA a klienty služby. Moduly SCA nabízejí služby ostatním prostřednictvím exportů. Vazby exportu definují specifický způsob přístupu klientů služby k modulu SCA.

Rozhraní a vazby

Export modulu SCA vyžaduje nejméně jedno rozhraní.

- Rozhraní exportu jsou abstraktní definice, které definují sadu operací za použití jazyka WSDL (Web Services Description Language), jazyka XML pro popis webových služeb. Modul SCA může mít řadu rozhraní exportu.
- Vazby exportu jsou konkrétní definice určující fyzický mechanizmus, který klienti služby používají pro přístup ke službě. Obvykle je pro export modulu SCA určena jedna vazba. Export, který nemá určenou žádnou vazbu, běhové prostředí interpretuje jako export s vazbou SCA.

Podporované vazby exportu

Produkt IBM Business Process Manager podporuje následující vazby exportu:

- Vazby SCA spojují moduly SCA s jinými moduly SCA. Vazby SCA bývají také označovány jako výchozí vazby.
- Vazby webových služeb povolují vyvolání exportů jako webových služeb. Podporované jsou protokoly SOAP1.1/HTTP, SOAP1.2/HTTP a SOAP1.1/JMS.

  Můžete použít vazbu SOAP1.1/HTTP nebo SOAP1.2/HTTP založenou na rozhraní Java API for XML Web Services (JAX-WS), která umožňuje interakce se službami používajícími vazby dokumentu nebo RPC/literal a která používá k úpravě vyvolání obslužné rutiny JAX-WS. Oddělená vazba SOAP1.1/HTTP je poskytována za účelem povolení interakcí se službami používajícími vazbu RPC/encoded nebo také když existuje požadavek na úpravu vyvolání za použití obslužných rutin JAX-RPC.
- Vazby HTTP povolují přístup k exportům za použití protokolu HTTP.
- Vazby exportu EJB (Enterprise JavaBeans) umožňují vystavit komponenty SCA jako objekty EJB. Obchodní logika prostředí Java EE tak může vyvolat komponenty, které jsou jinak pro tyto vazby nedostupné.
- Vazby podnikových informačních systémů (EIS) zajišťují konektivitu mezi komponentami SCA a externím podnikovým informačním systémem. Tuto komunikaci umožňuje použití adaptérů prostředku.
- Vazby platformy JMS (Java Message Service) verze 1.1 povolují s výchozím poskytovatelem systému zpráv serveru WebSphere Application Server. Služba JMS dokáže využívat různé typy transportu, včetně protokolu TCP/IP a HTTP či HTTPS. Automaticky je podporována třída zpráv JMS a jejích pět podtypů (text, bajty, objekt, proud a mapa).

- Generické vazby služby JMS povolují interoperabilitu s poskytovateli služby JMS od jiných dodavatelů, integrovanými do serveru WebSphere Application Server pomocí zařízení ASF (Application Server Facility) služby JMS.
- Vazby služby produktu WebSphere MQ JMS povolují interoperabilitu s poskytovateli služby JMS založenými na produktu WebSphere MQ. Automaticky je podporována třída zpráv JMS a jejích pět podtypů (text, bajty, objekt, proud a mapa). Pokud chcete použít produkt WebSphere MQ jako poskytovatele služby JMS, použijte vazby WebSphere MQ JMS.
- Vazby WebSphere MQ povolují interoperabilitu s produktem WebSphere MQ. Ke správci front MQ nebo vzdálenému počítači se připojujete pomocí vzdáleného (či klientského) připojení. Místní připojení (nebo vazby) je přímé připojení k produktu WebSphere MQ. To lze použít pouze k připojení ke správci front MQ v rámci stejného počítače. Produkt WebSphere MQ povolí oba typy připojení, ale vazby MQ podporují pouze "vzdálené" (nebo "klientské") připojení.

Mediační moduly:

Mediační moduly jsou moduly architektury SCA (Service Component Architecture), které mohou měnit formát, obsah nebo cíl požadavků na službu.

Mediační moduly pracují se zprávami, které jsou v přípravě mezi klienty služeb a poskytovateli služeb. Můžete směrovat zprávy k jiným poskytovatelům služeb a rovněž upravit obsah či strukturu zprávy. Mediační moduly mohou poskytovat funkce, jako např. protokolování zpráv a zpracování chyb, které jsou přizpůsobeny vašim požadavkům.

Z administrativní konzoly můžete měnit určité aspekty mediačních modulů bez nutnosti jejich nové implementace.

Komponenty mediačních modulů

Mediační moduly obsahují následující položky:

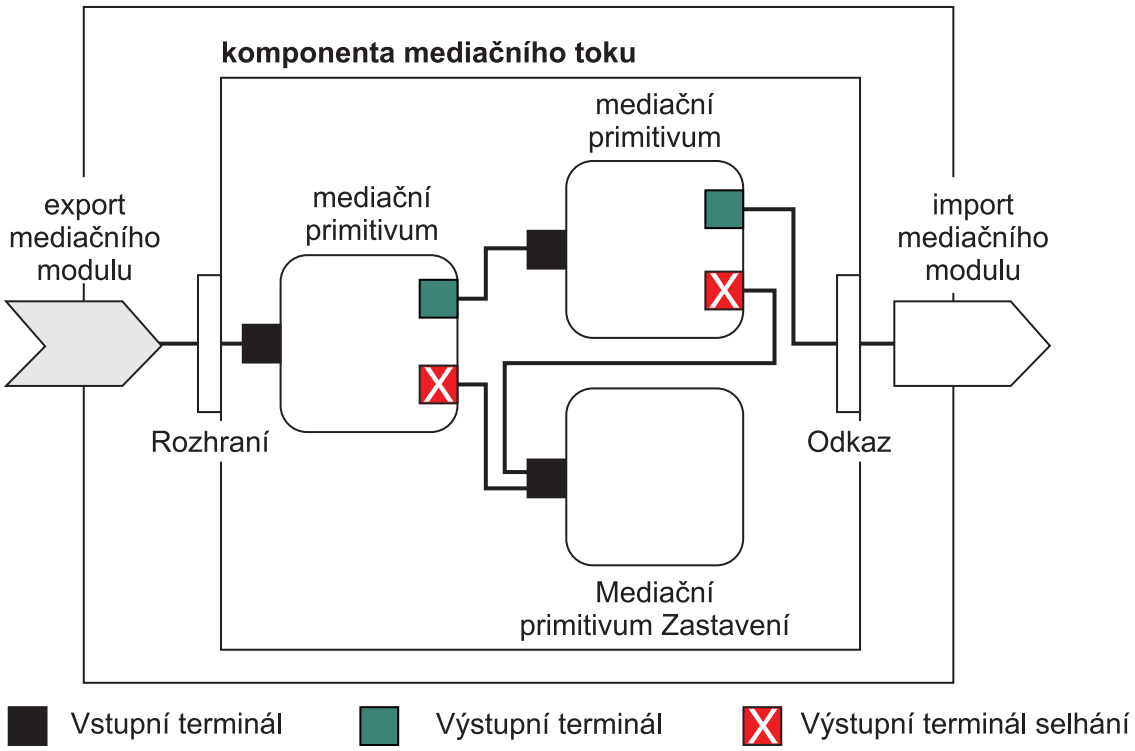
- Importy, které definují interakce mezi moduly SCA a poskytovateli služeb. Importy umožňují modulům SCA volat externí služby, jako kdyby byly lokální. Můžete si nechat zobrazit importy mediačního modulu a upravit jeho vazbu.
- Exporty, které definují interakce mezi moduly SCA a klienty služby. Umožňují modulu SCA nabídnout službu a definovat externí rozhraní (přístupový bod) modulu SCA. Můžete si nechat zobrazit exporty mediačního modulu.
- Komponenty SCA, které jsou stavebními bloky modulů SCA, jako jsou např. mediační moduly. Moduly a komponenty SCA lze pomocí produktu Integration Designer upravovat graficky. Po implementaci mediačního modulu můžete prostřednictvím administrativní konzoly upravovat určité jeho aspekty, a to bez nutnosti jeho nové implementace.

  Mediační moduly obvykle obsahují specifické typy komponent SCA nazvané *komponenta mediačního toku*. Komponenty mediačního toku definují mediační toky.

  Komponenta mediačního toku může obsahovat jedno, žádné nebo několik mediačních primitiv. Produkt IBM Business Process Manager podporuje dodanou sadu mediačních primitiv, která poskytuje funkce trasování a transformace zpráv. Pro další flexibilitu mediačních primitiv můžete pomocí primitiva Vlastní mediace volat vlastní logiku.

  Účelem mediačního modulu, který neobsahuje komponentu mediačního toku, je transformovat požadavky na službu z jednoho protokolu do jiného. Požadavek na službu může být například proveden pomocí SOAP/JMS, ale před odesláním jej může být potřeba transformovat na SOAP/HTTP.

  Poznámka: Mediační moduly si můžete nechat zobrazit a také na nich provádět určité změny prostřednictvím produktu IBM Business Process Manager. V produktu IBM Business Process Manager nicméně nemůžete zobrazovat ani měnit komponenty SCA určitého modulu. Komponenty SCA upravujte pomocí produktu Integration Designer.



Obrázek 43. Zjednodušený příklad mediačního modulu. Mediační modul obsahuje jednu komponentu mediačního toku, která obsahuje mediační primitiva.

- Vlastnosti

  Mediační primitiva mají vlastnosti, z nichž některé lze zobrazit v administrativní konzole jako další vlastnosti modulu SCA.

  Aby byly v administrativní konzole produktu IBM Business Process Manager vlastnosti mediačního primitiva viditelné, musí vývojář Integration Developer zvýšit jejich úroveň. Určité vlastnosti se hodí k tomu, aby byly konfigurovány administrativně a produkt Integration Designer takové vlastnosti popisuje jako podporovatelné vlastnosti, protože je lze povýšit z cyklu integrace do cyklu administrace. Jiné vlastnosti zase pro administrativní konfiguraci vhodné nejsou, protože jejich změna může ovlivnit mediační tok takovým způsobem, že by bylo nutné nově implementovat mediační modul. Produkt Integration Designer uvádí seznam vlastností, které můžete zařadit pod význačné vlastnosti mediačního primitiva.

  Hodnotu povýšených vlastností můžete pomocí administrativní konzoly produktu IBM Business Process Manager měnit bez nutnosti nové implementace mediačního modulu či restartování serveru či modulu.

  Obecně vzato mediační toky použijí změny vlastností okamžitě. Pokud nicméně dojde ke změně vlastnosti v buňce správce implementace, projeví se tyto změny u každého uzlu při jeho synchronizaci. Také mediační toky, které jsou v přípravě, používají dále své předchozí hodnoty.

  Poznámka: Prostřednictvím administrativní konzoly můžete měnit pouze hodnoty vlastností, nikoli skupiny vlastností, jejich názvy či typy. Chcete-li změnit skupiny vlastností, jejich názvy nebo typy, musíte použít produkt Integration Designer.
- Mediační modul nebo závislá knihovna mohou také definovat dílčí toky. Dílčí tok zapouzdřuje sadu mediačních primitiv spojených do znovupoužitelné části logiky integrace. Do mediačního toku lze přidat primitivum za účelem vyvolání dílčího toku.

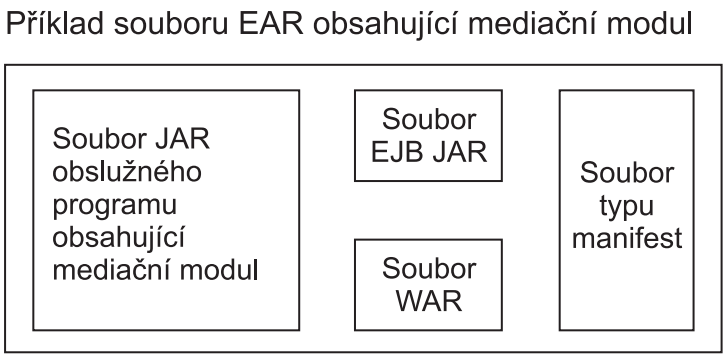
Implementace mediačních modulů

Mediační moduly se vytvářejí prostřednictvím produktu Integration Designer a obecně se implementují do produktu IBM Business Process Manager, do souboru EAR.

V době implementace lze změnit hodnotu povýšených vlastností.

Z produktu Integration Designer můžete mediační modul exportovat, a nechat jej tak zabalit do archivu JAR a ten zase do souboru EAR. Tento soubor EAR můžete potom implementovat tak, že prostřednictvím administrativní konzoly nainstalujete novou aplikaci.

Mediační moduly lze brát jako jednu entitu. Moduly SCA nicméně definuje řada souborů XML uložených v souboru JAR.



Obrázek 44. Zjednodušený příklad souboru EAR obsahujícího mediační modul. Soubor EAR obsahuje soubory JAR. Soubor JAR s obslužným programem obsahuje mediační modul.

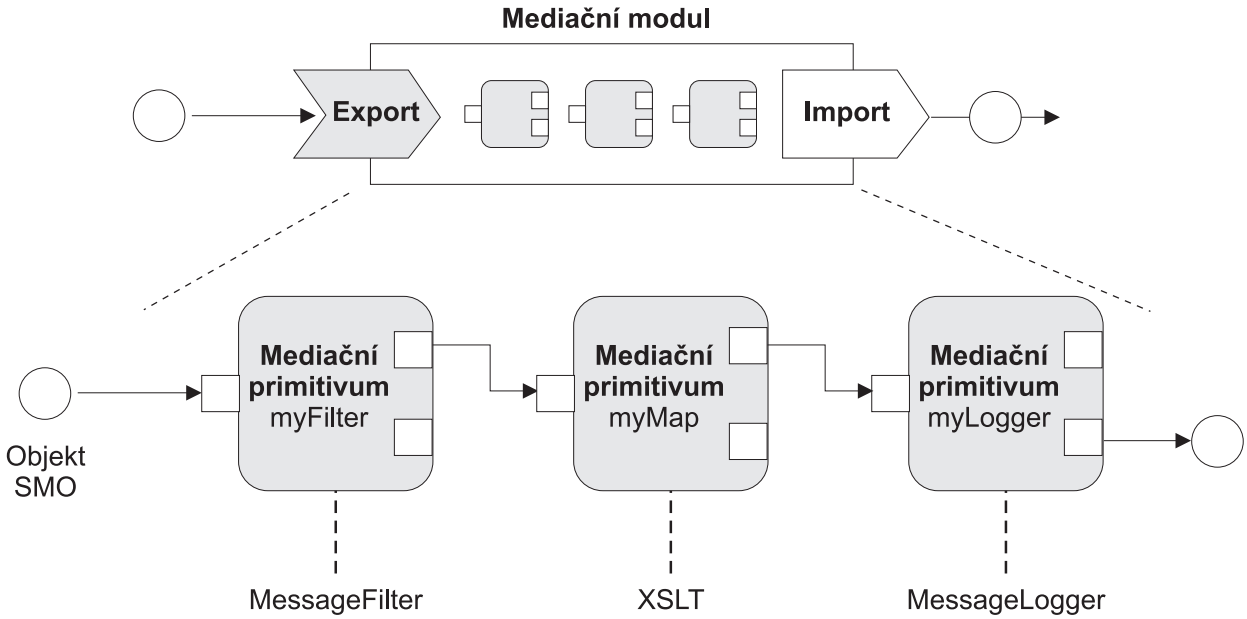
Mediační primitiva:

Komponenty mediačního toku operují v rámci toků zpráv mezi komponentami služby. Schopnosti mediačních komponent implementují *mediační primitiva*, která implementují standardní typy implementace služeb.

Komponenta mediačního toku obsahuje minimálně jeden tok. Například jeden pro požadavek a jeden pro odpověď.

Produkt IBM Business Process Manager podporuje dodanou sadu mediačních primitiv, které implementují standardní schopnosti mediace mediačních modulů nebo modulů implementovaných do produktu IBM Business Process Manager. Potřebujete-li speciální schopnosti mediace, můžete vyvinou svá primitiva Vlastní mediace.

Mediační primitivum definuje “vnitřní” operaci, která zpracovává zprávy reprezentované objekty SMO. Mediační primitivum může také definovat “vnější” operace, které odesílají zprávy do jiné komponenty nebo modulu.



Obrázek 45. Mediační modul obsahující tři mediační primitiva

Nakonfigurovat mediační primitiva a nastavit jejich vlastnosti můžete pomocí produktu Integration Designer. Některé z těchto vlastností lze zviditelnit pro administrátora běhového prostředí tím, že zvýšíte jejich úroveň. Jakákoli vlastnost mediačního primitiva, jejíž úroveň lze zvýšit, může být také dynamickou vlastností. Dynamickou vlastnost lze za běhu potlačit pomocí souboru zásad.

Produkt Integration Designer také umožňuje grafické modelování a sestavení komponent mediačního toku z mediačních primitiv a sestavení mediačních modulů nebo modulů z komponent mediačního toku. Administrativní konzola označuje mediační moduly a moduly jako moduly SCA.

Produkt Integration Designer umožňuje také definovat dílčí toky v rámci modulů nebo jejich závislých knihoven. Dílčí tok může obsahovat jakékoli mediační primitivum, s výjimkou mediačního primitiva Vyhodnocení zásady. Dílčí tok se vyvolává z požadavku nebo toku odezvy nebo z jiného dílčího toku za pomoci mediační primitiva Dílčí tok. Vlastnosti povýšené z mediačních primitiv dílčího proudu jsou vystavené jako vlastnosti mediačního primitiva Dílčí tok. Úroveň těchto vlastností lze dále zvyšovat, až dokud nedosáhnou úrovně modulu, což je bod, kdy je již může upravovat administrátor běhového prostředí.

Podporovaná mediační primitiva

Produkt IBM Business Process Manager podporuje následující mediační primitiva:

Mapa obchodních objektů
Transformuje zprávy.

- Definuje transformace zpráv pomocí mapy obchodních objektů, kterou lze používat opakovaně.
- Prostřednictvím editoru map obchodních objektů umožňuje grafické definování transformací zpráv.
- Může změnit obsah zprávy.
- Může transformovat vstupní typ zprávy na jiný výstupní typ zprávy.

Vlastní mediace
Umožňuje vám implementaci vlastní logiky mediace do kódu jazyka Java. Primitivum Vlastní mediace kombinuje flexibilitu mediačního primitiva definovaného uživatelem a jednoduchost předdefinovaného mediačního primitiva. Následujícími způsoby můžete vytvořit komplexní šablony transformací a trasování:

- Vytvořením kódu jazyka Java.

- Vytvořením svých vlastních vlastností.
- Přidáním nových terminálů.

Službu můžete vyvolat prostřednictvím primitiva Vlastní mediace, ale mediační primitivum Vyvolání služby je navrženo tak, aby volalo služby a poskytovalo další funkčnosti, jako např. opakovaný pokus.

Popisovač dat

Umožňuje vám transformovat část zprávy. Používá se k převodu prvku zprávy z fyzického formátu na logickou strukturu nebo k převodu logické struktury do fyzického formátu. Toto primitivum se primárně používá k převodu fyzického formátu, jako např. textového řetězce z objektu Textová zpráva JMS, do logické struktury obchodních objektů a zpátky. Tato mediace se běžně používá na:

- Transformování sekce vstupní zprávy z definované struktury na strukturu jinou. Příkladem může být situace, kdy objekt SMO obsahuje hodnotu řetězce oddělenou čárkami, kterou chcete analyzovat pro potřeby specifického obchodního objektu.
- Pozměnění typu zprávy. Příkladem by mohla být situace, kdy byl export služby JMS konfigurován pro použití základního typu vázání dat služby JMS a v rámci mediačního modulu se vývojář Integration Developer rozhodne, že by měl být obsah předán do specifické struktury obchodních objektů.

Vyhledávání v databázi

Upravuje zprávy za použití informací z databáze dodané uživatelem.

- Musíte nastavit databázi, zdroj dat a všechna nastavení ověření serveru, která má mediační primitivum Vyhledávání v databázi použít. S tímto úkolem vám pomůže administrativní konzola.
- Mediační primitivum Vyhledávání v databázi může číst pouze z jedné tabulky.
- Určený sloupec klíče musí obsahovat jedinečnou hodnotu.
- Data ve sloupci hodnot musí být buď typ jednoduchého schématu XML, nebo typ schématu XML, který rozšiřuje jednoduchý typ schématu XML.

Vyhledávání koncového bodu

Umožňuje dynamické směrování požadavků díky vyhledávání koncových bodů služeb v úložišti.

- Informace o koncovém bodu služby se získává z produktu WSSR (WebSphere Service Registry and Repository). Registr produktu WSRR může být buď místní, nebo vzdálený.
- Změny v registru se provádí prostřednictvím administrativní konzoly produktu WSRR.
- Produkt IBM Business Process Manager musí vědět, jaký registr má používat. Z toho důvodu musíte pomocí administrativní konzoly produktu IBM Business Process Manager vytvořit definice přístupu produktu WSRR.

Emitor událostí

Rozšiřuje monitorování tím, že vám umožní odesílat události z komponenty mediačního toku.

- Zrušením zaškrtnutí příslušného políčka můžete akci mediace pozastavit.
- Události emitoru událostí si můžete nechat zobrazit pomocí prohlížeče událostí CBE (Common Base Event)v produktu IBM Business Process Manager.
- Události byste měli odesílat pouze v případě významné události v rámci mediačního toku nebo z důvodu zajištění výkonu.
- Můžete nadefinovat části zprávy, které daná událost obsahuje.
- Události jsou odesílány ve formě událostí Common Base Event na server Common Event Infrastructure.
- Aby bylo možné informace z emitoru událostí využít plně, musí odběratelé událostí chápat strukturu událostí Common Base Event. Událost Common Base Event má celkové schéma, které ale nemodeluje data specifická pro aplikace, ta jsou obsažena v rozšířených datových prvcích. Pro účely modelování rozšířených datových prvků generují nástroje produktu Integration Designer pro každé nakonfigurované mediační primitivum Emitor událostí soubor definice katalogu událostí infrastruktury Common Event Infrastructure. Definiční soubory katalogu událostí jsou artefakty exportu poskytované jako asistence. Produkt Integration Designer ani běhové prostředí produktu IBM Business Process Manager je nepoužívají. Aplikace spotřebovávající události emitoru událostí byste měli vytvářet s ohledem na soubory definic katalogu událostí.

- V produktu IBM Business Process Manager můžete určit jiné monitorování. Můžete například monitorovat události emitované z importů či exportů.

Selhání

Zastaví konkrétní cestu toku a vygeneruje výjimku.

Vstupní větvení

Pomáhá agregovat (sloučit) zprávy.

- Lze použít pouze ve spojení s mediačním primitivem Výstupní větvení.
- Společné použití mediačních primitiv Vstupní větvení a Výstupní větvení umožňuje agregovat data do jedné výstupní zprávy.
- Mediační primitivum Vstupní větvení přijímá zprávy až do dosažení bodu rozhodování, kdy se výstupem stává jedna zpráva.
- Data by měla být zadržována pomocí sdíleného kontextu.

Výstupní větvení

Pomáhá dělit a agregovat (slučovat) zprávy.

- Společné použití mediačních primitiv Vstupní větvení a Výstupní větvení umožňuje agregovat data do jedné výstupní zprávy.
- V režimu iterace umožňuje mediační primitivum Výstupní větvení iterovat na základě jediné vstupní zprávy obsahující opakující se prvek. Pro každý výskyt tohoto opakujícího se prvku je odeslána zpráva.
- Data by měla být zadržována pomocí sdíleného kontextu.

Modul nastavení záhlaví HTTP

Poskytuje mechanizmus správy záhlaví zpráv HTTP.

- Může vytvořit, nastavit, zkopírovat nebo odstranit záhlaví zpráv HTTP.
- Může nastavit více akcí za účelem změny více záhlaví HTTP.

Mapování

Transformuje zprávy.

- Umožňuje provádět transformace jazyka XSL (Extensible Stylesheet Language) nebo transformace mapy obchodních objektů.
- Zprávy transformujete pomocí transformace XSLT 1.0 nebo XSLT 2.0, případně pomocí transformace mapy obchodních objektů. Transformace XSL operují na serializaci XML zprávy, kdežto transformace mapy obchodních objektů operují na objektech SDO (Service Data Object).

Modul nastavení prvku zprávy

Poskytuje jednoduchý mechanizmus nastavení obsahu zpráv.

- Může změnit, přidat nebo odstranit prvky zprávy.
- Nemění typ zprávy.
- Data ve sloupci hodnot musí být buď typ jednoduchého schématu XML, nebo typ schématu XML, který rozšiřuje jednoduchý typ schématu XML.

Filtr zpráv

Směruje zprávy po různých cestách na základě jejich obsahu.

- Zrušením zaškrtnutí příslušného políčka můžete akci mediace pozastavit.

Modul protokolování zpráv

Protokoluje zprávy v relační databázi nebo prostřednictvím vašeho vlastního modulu protokolování. Tyto zprávy se ukládají jako XML, a jejich data lze tedy následně zpracovat v aplikacích, které s XML pracují.

- Zrušením zaškrtnutí příslušného políčka můžete akci mediace pozastavit.
- Schéma databáze Rational (strukturu tabulek) definuje IBM.
- Standardně používá mediační primitivum Modul protokolování zpráv obecnou databázi. Běhové prostředí mapuje zdroj dat v protokolu **jdbc/mediation/messageLog** k obecné databázi.

- Chcete-li upravit chování vlastního modulu protokolování, můžete nastavit implementační třídy obslužné rutiny. Dále můžete chování vlastního modulu protokolování upravit poskytnutím implementačních tříd formátovače.

Modul nastavení záhlaví MQ

Poskytuje mechanizmus správy záhlaví zpráv MQ.

- Může vytvořit, nastavit, zkopírovat nebo odstranit záhlaví zpráv MQ.
- Může nastavit více akcí za účelem změny více záhlaví MQ.

Vyhodnocení zásady

Umožňuje dynamickou konfiguraci požadavků díky vyhledávání koncových bodů služeb a přidružených souborů zásad v úložišti.

- Soubor zásad můžete použít k dynamickému potlačení povýšených nastavení jiných mediačních primitiv.
- Informace o koncovém bodu služby a informace o zásadách se získávají z produktu WSSR (WebSphere Service Registry and Repository). Registr produktu WSRR může být buď místní, nebo vzdálený.
- Změny v registru se provádí prostřednictvím administrativní konzoly produktu WSRR.
- Produkt IBM Business Process Manager musí vědět, jaký registr má používat. Z toho důvodu musíte pomocí administrativní konzoly produktu IBM Business Process Manager vytvořit definice přístupu produktu WSRR.

Vyvolání služby

Volá službu z mediačního toku, takže není třeba čekat na konec mediačního toku a používat mechanizmus volání.

- Pokud služba vrátí poruchu, můžete opakovat stejnou službu nebo volat službu jinou.
- Mediační primitivum Vyvolání služby je výkonné mediační primitivum, které lze pro jednoduchá volání služby použít samotné, nebo jej lze pro účely komplexních mediací použít v kombinaci s dalšími mediačními primitivy.

Nastavení typu zprávy

Při vývoji integrace umožňuje považovat pole zpráv slabého typu za pole zpráv silného typu. Pole je slabého typu, když může obsahovat více než jeden typ dat. Pole je silného typu, pokud je znám jeho typ a interní struktura.

- Za běhu umožňuje mediační primitivum Nastavení typu zprávy kontrolovat, zda obsah zprávy odpovídá očekávanému typu dat.

Modul nastavení záhlaví SOAP

Poskytuje mechanizmus správy záhlaví zpráv SOAP.

- Může vytvořit, nastavit, zkopírovat nebo odstranit záhlaví zpráv SOAP.
- Může nastavit více akcí za účelem změny více záhlaví SOAP.

Zastavení

Zastaví konkrétní cestu toku bez generování výjimky.

Filtr typů

Umožňuje vám směrovat zprávy do různých cest toku na základě typu zprávy.

Načtení z produktu WebSphere eXtreme Scale

Můžete načíst informace z prostředí mezipaměti serveru eXtreme Scale.

- Můžete vyhledat hodnoty v mezipaměti a uložit je jako prvky do zprávy pomocí klíče.
- Kombinací mediačních primitiv Uložení do produktu eXtreme Scale a Načtení z produktu eXtreme Scale lze uložit do mezipaměti odezvu z back-endového systému. Budoucí požadavky již nebudou vyžadovat přístup k back-endovému systému.
- Je třeba vytvořit definice produktu eXtreme Scale pomocí administrativní konzoly produktu WebSphere ESB, abyste mohli určit, který server eXtreme Scale se má použít.

Uložení do produktu WebSphere eXtreme Scale

Můžete uložit informace do prostředí mezipaměti serveru eXtreme Scale.

- Informace lze uložit do mezipaměti produktu eXtreme Scale pomocí klíče a objektu.
- Můžete využít kombinaci mediačních primitiv Uložení do produktu eXtreme Scale a Načtení z produktu eXtreme Scale; pomocí mediačního primitiva Uložení uložíte data do mezipaměti a pomocí mediačního primitiva Načtení dříve uložená data načtete.
- Je třeba vytvořit definice produktu eXtreme Scale pomocí administrativní konzoly produktu WebSphere ESB, abyste mohli určit, který server eXtreme Scale se má použít.

Dynamické směrování:

Zprávy můžete směrovat různými způsoby za použití koncových bodů definovaných v době integrace nebo dynamicky za běhu určených koncových bodů.

Dynamické směrování pokrývá dva případy směrování zpráv:

- Směrování zpráv, kdy je tok dynamický, ale všechny možné koncové body jsou předdefinované v modulu architektury SCA (Service Component Architecture).
- Směrování zpráv, kdy je dynamický nejen tok, ale také výběr koncového bodu. Koncové body služeb se vybírají z externího zdroje za běhu.

Dynamický výběr koncového bodu

Běhové prostředí má schopnost směrovat zprávy požadavků a odezev na adresu koncového bodu identifikovanou prvkem záhlaví zprávy. Tento prvek záhlaví zprávy lze aktualizovat pomocí mediačních primitiv v mediačním toku. Adresu koncového bodu lze aktualizovat pomocí informací z registru, databáze nebo pomocí informací ze samotné zprávy. Směrování zpráv odpovědí přichází v úvahu pouze, když je odezva odesílána prostřednictvím exportu rozhraní JAX-WS webové služby.

Aby běhové prostředí mohlo dynamické směrování na základě požadavku nebo odezvy implementovat, musí být pro modul SCA nastavena vlastnost Použít dynamický koncový bod, pokud je nastavený v záhlaví zprávy. Vývojáři integrace mohou vlastnost Použít dynamický koncový bod, pokud je nastavený v záhlaví zprávy nastavit nebo mohou zvýšit její úroveň (zviditelnit ji za běhu), aby ji mohl nastavit administrátor běhového prostředí. Vlastnosti modulu si lze nechat zobrazit v okně Vlastnosti modulu. Okno se zobrazí po klepnutí na položku **Aplikace > Moduly SCA > Vlastnosti modulu**. Vývojář Integration Developer přidělí každé povýšené vlastnosti název aliasu, což budou názvy zobrazované v administrativní konzole.

Registr

Informace o koncových bodech služby můžete uložit v produktu WSRR (IBM WebSphere Service Registry and Repository) a následně můžete vytvořit moduly SCA, které koncové body z registru WSRR načtou.

Když vyvíjíte moduly SCA, umožňujete mediačnímu toku dotazovat se registru WSRR na koncový bod služby či na sadu koncových bodů služeb prostřednictvím mediačního primitiva Vyhledávání koncového bodu. Pokud modul SCA obdrží sadu koncových bodů, musí preferovaný koncový bod vybrat pomocí dalšího mediačního primitiva.

Řízení zásad mediace požadavků na službu:

Zásady mediace můžete použít k řízení mediačních toků mezi klienty služeb a poskytovateli služeb.

Mediační toky lze řídit pomocí zásad mediace uložených v produktu IBM WebSphere Service Registry and Repository (WSRR). Implementace správy zásad služeb v produktu WSRR je založena na rámci zásad WS-Policy (Web Services Policy Framework).

Abyste mohli řídit požadavky na službu pomocí zásad mediace, musíte mít v registru WSRR vhodné moduly SCA (Service Component Architecture) a dokumenty zásad služeb.

Postup připojení zásady mediace k požadavku na službu

Při vyvíjení modulu SCA, který musí využívat zásadu mediace, je třeba do mediačního toku zahrnout mediační primitivum Vyhodnocení zásady. Za běhu toto mediační primitivum Vyhodnocení zásady získá z registru informace o zásadě mediace. Kvůli podpoře řízení zásad mediace u požadavků na služby proto modul SCA musí obsahovat komponentu mediačního toku.

V registru můžete připojit jednu či více zásad mediace k modulu SCA nebo k cílové službě, kterou modul SCA používá. Připojené zásady mediace lze použít (tj. jsou v dosahu) pro všechny zprávy služby zpracovávaných daným modulem SCA. Zásady mediace mohou obsahovat připojení zásady, která definují podmínky. Podmínky zásady mediace umožňují použít v různých kontextech různé zásady mediace. Kromě toho mohou zásady mediace obsahovat klasifikace, které lze použít k určení stavu řízení.

WebSphere Service Registry and Repository:

Produkt WSRR (WebSphere Service Registry and Repository) vám umožňuje uložit a spravovat informace o koncových bodech služeb a zásadách mediace, stejně jako k těmto informacím přistupovat. Pomocí produktu WSRR mohou být vaše aplikace služeb dynamičtější a přizpůsobitelnější vůči měnícím se obchodním podmínkám.

Úvod

Mediační toky mohou produkt WSRR používat jako mechanizmus dynamického vyhledávání, a získávat tak informace o koncových bodech služeb a zásadách mediace.

Za účelem získání přístupu k produktu WSRR se prostřednictvím administrativní konzoly vytváří dokumenty definic WSRR. Můžete také použít příkazy administrace produktu WSRR prostřednictvím skriptovacího klienta wsadmin. Mechanizmem používaným pro připojení k instanci registru a získání koncového bodu služby či zásady mediace jsou definice produktu WSRR a vlastnosti jejich připojení.

Koncové body služeb

Úložiště WSRR můžete používat k uchovávání informací o službách, které již používáte, plánujete používat nebo o kterých chcete být informováni. Tyto služby se mohou nacházet ve vašich systémech nebo i v jiných systémech. Určitá aplikace může například použít produkt WSRR k vyhledání nejvhodnější služby, která by splňovala potřeby jejího fungování a výkonu.

Když vyvíjíte modul SCA, který musí přistupovat ke koncovým bodům služeb z produktu WSRR, musíte do daného mediačního toku zahrnout mediační primitivum Vyhledávání koncového bodu. Za běhu získává mediační primitivum Vyhledávání koncového bodu koncové body služeb z registru.

Zásady mediace

Produkt WSRR můžete také použít k uložení informací o zásadách mediace. Zásady mediace vám mohou pomoci s řízením požadavků na služby tím, že potlačí vlastnosti modulu. Pokud produkt WSRR obsahuje zásady mediace, které jsou přiložené k objektu reprezentujícímu buď vás modul SCA, nebo vaši cílovou službu, mohou zásady mediace potlačit vlastnosti modulu. Pokud chcete v rámci různých kontextů aplikovat různé zásady mediace, můžete vytvořit podmínky zásad mediace.

Poznámka: Zásady mediace se zabývají řízením mediačních toků, nikoliv zabezpečením.

Při vyvíjení modulu SCA, který musí využívat zásadu mediace, je třeba do mediačního toku zahrnout mediační primitivum Vyhodnocení zásady. Za běhu toto mediační primitivum Vyhodnocení zásady získá z registru informace o zásadě mediace.

WebSphere eXtreme Scale:

Produkt WebSphere eXtreme Scale (eXtreme Scale) umožňuje poskytovat systém mezipaměti, který lze integrovat s aplikací produktu IBM Business Process Manager. Použití produktu eXtreme Scale s produktem IBM Business Process Manager může zlepšit spolehlivost a dobu odezvy služeb a zajistit další funkčnost integrace.

Produkt eXtreme Scale funguje jako elastická, přizpůsobitelná datová mřížka v paměti. Tato datová mřížka dynamicky ukládá do mezipaměti, segmentuje, replikuje a spravuje data aplikací a obchodní logiku v rámci více serverů. Pomocí produktu eXtreme Scale lze rovněž získat kvality služby, jako např. transakční integritu, vysokou dostupnost a předvídatelnou dobu odezvy.

Pro přístup k funkci ukládání do mezipaměti produktu eXtreme Scale lze použít mediační toky obsahující mediační primitiva produktu WebSphere eXtreme Scale. Při vyvíjení modulu SCA, který musí ukládat informace do mezipaměti produktu eXtreme Scale, je třeba do mediačního toku zahrnout mediační primitivum Uložení do produktu WebSphere eXtreme Scale. Chcete-li načítat informace z mezipaměti produktu eXtreme Scale, je třeba zahrnout mediační primitivum Načtení z produktu WebSphere eXtreme Scale. Kombinací těchto dvou mediačních primitiv v mediačním toku lze uložit do mezipaměti odezvu z back-endového systému, odkud ji mohou načítat budoucí požadavky.

Chcete-li konfigurovat přístup k produktu eXtreme Scale, musíte pomocí administrativní konzoly vytvořit definici produktu WebSphere eXtreme Scale. Volitelně můžete použít příkazy administrace produktu WebSphere eXtreme Scale ve skriptovacím klientu wsadmin. Definice produktu eXtreme Scale je mechanizmus, s jehož pomocí se mediační primitiva Načtení z produktu WebSphere eXtreme Scale a Uložení do produktu WebSphere eXtreme Scale připojují k serveru eXtreme Scale.

Klient Message Service

Klienti Message Service jsou k dispozici pro jazyk C/C++ a .NET za účelem povolení připojení aplikací nezaložených na platformě Java ke sběrnici Enterprise Service Bus.

Klienti Message Service Clients pro C/C++ a .NET poskytují rozhraní API nazvané XMS, které má stejnou sadu rozhraní jako rozhraní API služby JMS (Java Message Service). Klient Message Service Client for C/C++ obsahuje dvě implementace XMS, jednu pro použití aplikacemi v jazyce C a druhou pro použití aplikacemi v jazyce C++. Klient Message Service Client pro .NET obsahuje plně spravovanou implementaci XMS, kterou může používat libovolný jazyk vyhovující rámci .NET.

Klienty Message Service Client for .NET lze získat na adrese http://www-01.ibm.com/support/docview.wss?rs=0&q1=IA9H&uid=swg24011756&loc=en_US&cs=utf-8&cc=us&lang=en

Klienty Message Service Clients for C/C++ lze získat http://www-01.ibm.com/support/docview.wss?rs=0&q1=ia94&uid=swg24007092&loc=en_US&cs=utf-8&cc=us&lang=en.

Můžete také nainstalovat a používat podporu klienta prostředí Java EE z produktu WebSphere Application Server Network Deployment, včetně klienta webových služeb, klienta objektu EJB a klienta služby JMS.

Kapitola 2. Další informace o klíčových konceptech

Tento oddíl použijte jako výchozí bod pro zkoumání technologií, které se používají v produktu IBM Business Process Manager a které tento produkt využívají.

Scénáře tvorby obsahu

Pomocí scénářů lze porozumět komponentám a produktům z řady pro řízení BPM a pracovat s nimi.

Správa verzí

Životní cyklus komponenty Process Application začíná jejím vytvořením a pokračuje cyklem její aktualizace, implementace, souběžné implementace, odstranění implementace a archivace. *Správa verzí* je mechanizmus správy životního cyklu komponenty Process Application, který provede jedinečnou identifikaci jednotlivých verzí této komponenty.

Způsob fungování správy verzí v produktu IBM Business Process Manager závisí na tom, co implementujete - komponentu Process Application implementovanou z úložiště produktu IBM Process Center nebo podnikovou aplikaci implementovanou přímo z produktu IBM Integration Designer.

Komponenty Process Application a Toolkit, které implementujete do běhového prostředí z komponenty Process Center jsou standardně opatřeny verzí. V případě podnikových aplikací můžete moduly a knihovny opatřit verzí v produktu IBM Integration Designer.

Navíc můžete vytvořit verze lidské úlohy či stavového automatu, aby tak mohlo v běhovém prostředí současně existovat více verzí dané úlohy či stavového automatu.

Správa verzí komponent Process Application

Díky správě verzí dokáže běhové prostředí identifikovat snímky v životním cyklu komponenty Process Application a může souběžně spouštět více snímků ve stejný čas.

Na komponentu Process Application se můžete dívat jako na kontejner. Všechny snímky, implementace a správa verzí se řeší na úrovni kontejneru, nikoli na úrovni jednotlivých artefaktů v tomto kontejneru. Snímky se spravují prostřednictvím konzoly Process Center Console.

Změny se ukládají dynamicky do úložiště komponenty Process Center na vrchol, který je aktuální pracovní verzí komponenty Process Application. Komponenta Process Application zůstává na této úrovni vrcholu tak dlouho, dokud se nerozhodnete vytvořit snímek (sn1). Snímek komponenty Process Application lze implementovat na server komponenty Process Center nebo na Process Server pro účely testování, fázování nebo produkce.

Pokud provedete změny a chcete implementovat novou verzi, musíte vytvořit nový snímek (sn2). Při implementaci snímku sn2 můžete snímek sn1 buď odebrat, nebo jej ponechat spuštěný na serveru.

Kontext verze

Kontextem verze jsou metadata identifikující určitou verzi. Identifikátor přiřazujete vy sami, nicméně IBM doporučuje používat tříciferný způsob označování verzí ve formátu <hlavní>.<vedlejší>.<služba>. Podrobnější popis tohoto schématu číslování verzí najdete v tématech o konvencích pojmenování.

Produkt IBM Business Process Manager přiřazuje každé komponentě Process Application globální obor názvů. Globálním oborem názvů je buď vrchol komponenty Process Application, nebo konkrétní snímek komponenty Process Application. Název verze používaný serverem nesmí obsahovat více než sedm znaků, a tak přiřazený název bude

akronymem s použitím znaků z názvu snímku, který jste přiřadili. Akronymy snímků se budou shodovat s příslušnými názvy snímků - za předpokladu, že názvy snímků splňují doporučený styl IBM VRM a nejsou delší než sedm znaků. Například název snímku 1.0.0 bude mít akronym 1.0.0, resp. název snímku 10.3.0 bude mít akronym 10.3.0. U akronymu snímku je zaručena jedinečnost v kontextu komponenty Process Application v rámci rozsahu serveru komponenty Process Center. Z tohoto důvodu nemůžete akronym snímku měnit.

Správa verzí komponent Process Application a Toolkit produktu Process Designer

Chcete-li spravovat verze komponent Process Application a Toolkit v úložišti komponenty Process Center, můžete ukládat snímky a přiřazovat jim názvy. Poté můžete například navzájem porovnat určitý snímek s jiným a zjistit, jaké jsou mezi nimi rozdíly. Opraví-li například vývojář určitý problém se službou, pořídí v tomto bodě snímek nadřízené komponenty Process Application nebo Toolkit a poté přijde jiný vývojář, provede několik dalších změn ve stejné službě a pořídí další snímek. Správce projektu může oba snímky porovnat a zjistit, které změny kdy kdo provedl. Zjistí-li správce projektu, že dodatečně provedené změny nevedly k požadovaným výsledkům, může se vrátit zpět ke snímku původní opravy.

Snímek komponenty Process Application zpravidla pořizujete pokaždé, když jste připraveni nebo téměř připraveni implementovat do produkčního prostředí nebo testovat integraci. Snímek komponenty Process Application musíte pořídit v případě, že chcete implementovat na samostatný Process Server. V případě komponent Toolkit je situace poněkud odlišná; snímek komponenty Toolkit pořizujete ve chvíli, kdy je daná komponenta Toolkit připravena k použití komponentami Process Application. Pokud později chcete komponentu Toolkit aktualizovat, musíte v příslušném okamžiku pořídit nový snímek "vrcholu", přičemž vlastníci komponent Process Application a Toolkit se rozhodnou, zda chtějí na nový snímek přejít. Jako vrchol označujeme speciální a zároveň jediný snímek, ve kterém lze měnit obsah, ale který můžete spouštět pouze na serveru komponenty Process Center. Na serveru Process Server nemůžete vrchol implementovat.

Komponenty Process Application ve více klastrech

Jednu verzi komponenty Process Application můžete implementovat na více klastrech v rámci jedné buňky. Chcete-li rozlišovat mezi více implementacemi stejné verze komponenty Process Application, vytvořte pro každou implementaci snímek a použijte v názvu snímku jedinečný identifikátor v rámci buňky (například v1.0_cell1_1 a v1.0_cell1_2). Pokud bychom chtěli být zcela přesní, každý snímek je novou verzí komponenty Process Application (čistě z pohledu správy životního cyklu), ale obsah a funkce jsou stejné.

Při implementaci komponenty Process Application na klastr se provádí automatická synchronizace uzlů.

Správa verzí modulů a knihoven

Pokud se modul či knihovna nachází v komponentě Process Application nebo Toolkit, přebírá životní cyklus těchto komponent (verze, snímky, stopy atd.). Názvy modulů a knihoven musí být v rámci rozsahu komponenty Process Application či Toolkit jedinečné.

Toto téma popisuje správu verzí modulů a knihoven používaných s komponentami Process Application. Poznámka: Pokud nicméně implementujete moduly na komponentu Process Server přímo z produktu IBM Integration Designer, můžete i nadále postupovat tak, že přiřadíte modulům čísla verzí během implementace, jak je popsáno v tématu “Vytvoření modulů a knihoven s verzemi”.

Modul či knihovna přidružená k produktu IBM Process Center musí mít své závislé knihovny ve stejné komponentě Process Application nebo v závislé komponentě Toolkit.

V následující tabulce jsou vypsány možné výběry v editoru závislostí z produktu IBM Integration Designer při přidružení knihovny ke komponentě Process Application či Toolkit:

Tabulka 25. Závislosti pro knihovny Modulu, komponent Process Application či Toolkit a Globální knihovnu

Rozsah knihovny	Popis	Může záviset na . . .
Modul	Na serveru existuje kopie této knihovny pro každý modul, který ji používá.	Knihovna s oborem modulu může záviset na všech typech knihoven.
komponenty Process Application či Toolkit	Tato knihovna je sdílena mezi všemi moduly v rámci oboru komponenty Process Application či Toolkit. Toto nastavení se projeví, pokud implementace probíhá prostřednictvím produktu IBM Process Center. Pokud implementace probíhá mimo produkt IBM Process Center, dojde ke zkopírování knihovny do každého modulu. **Poznámka:** Knihovny vytvořené v produktu IBM Integration Designer verze 8 mají standardně nastavenu úroveň sdílení na volbu **komponenta Process Application či Toolkit**.	Knihovna tohoto typu může záviset pouze na globálních knihovnách.
Globální	Tato knihovna je sdílena mezi všemi spuštěnými moduly.	Globální knihovna může být závislá pouze na jiných globálních knihovnách. **Poznámka:** Abyste mohli implementovat globální knihovnu, musíte nakonfigurovat sdílenou knihovnu WebSphere. Další informace viz "Závislosti modulů a knihoven".

Moduly a knihovny přidružené ke komponentám Process Application nebo Toolkit

Modulům a knihovnám přidruženým ke komponentám Process Application nebo Toolkit nemusíte opatřit verzi.

Moduly a knihovny, které jsou přidruženy ke komponentě Process Application nebo Toolkit, nemusí být opatřeny verzí. Ve skutečnosti nemůžete v editoru závislostí vytvořit verzi modulu nebo knihovny přidružené ke komponentě Process Application nebo Toolkit. Moduly a knihovny přidružené ke komponentě Process Application nebo Toolkit používají snímky a funkci v produktu Process Center, aby dosáhly stejného výsledku jako verze.

Knihovny přidružené s komponentou Process Application nebo Toolkit nebudou mít požadované číslo verze v části editoru závislostí Knihovny, protože verze není potřeba.

Konvence pojmenování

Konvence pojmenování se používá pro rozrůznění možných verzí komponenty Process Application při jejím průchodu svým životním cyklem, který zahrnuje aktualizace, implementace, souběžné implementace, zrušení implementací či archivace.

V tomto oddílu naleznete konvence používané k jedinečné identifikaci verzí komponenty Process Application.

Kontext verze je kombinace akronymů, sloužící jako jedinečný popis komponenty Process Application či Toolkit. Každý typ akronymu má svou konvenci pojmenování. Akronym je omezený maximální délkou sedmi znaků ze znakové sady [A-Z0-9_], s výjimkou akronymu snímku, který může obsahovat také tečku.

- Akronym komponenty Process Application je vytvořen při vytvoření této komponenty. Jeho délka nesmí přesáhnout sedm znaků.
- Akronym snímku je vytvořen automaticky při vytvoření snímku. Jeho délka nesmí přesáhnout sedm znaků.

  Pokud název akronymu splňuje kritéria platného akronymu snímku, název snímku a akronym jsou shodné.

  Poznámka: S použitím funkce trasování s ohledem na verzi komponenty mediačního toku pojmenujte svůj snímek tak, aby odpovídal schématu <*verze*>.<*vydání*>.<*úprava*> (např. **1.0.0**). Vzhledem k tomu, že je délka akronymu

snímku omezena na sedm znaků, jsou číselné hodnoty omezeny na maximálně pět číslic (pět číslic a dvě tečky). Při zvyšování hodnot polí s číslicemi proto postupujte obezřetně, protože cokoliv, co přesáhne prvních sedm znaků, bude oříznuto.

Například název snímku **11.22.33** povede k akronymu snímku **11.22.3**.

- Akronym stopy se automaticky generuje z prvního znaku každého slova názvu dané stopy. Například nově vytvořená stopa s názvem **Moje Nová Stopa** by vedla k hodnotě akronymu **MNS**.

  Výchozí název stopy a akronym jsou **Hlavní**. Implementace na server produktu IBM Process Center obsahuje akronym stopy v kontextu správy verzí, pokud akronym stopy není **Hlavní**.

Definici obchodního procesu z komponenty Process Application zpravidla definuje akronym názvu komponenty Process Application, akronym snímku a název definice obchodního procesu. Kdykoliv to bude možné, vybírejte pro své definice obchodních procesů jedinečné názvy. V případě výskytu duplicitních názvů můžete narazit na následující problémy:

- Definice obchodních procesů může být možné vystavit jako webové služby pouze prostřednictvím určité mediace.
- Může se stát, že nebudete moci vyvolat definici obchodního procesu vytvořenou v produktu IBM Process Designer z procesu BPEL vytvořeného v produktu IBM Integration Designer.

Kontext verze je proměnlivý v závislosti na způsobu implementace komponenty Process Application.

Konvence pojmenování pro implementace serveru Process Center

Na server IBM Process Center můžete implementovat snímek komponenty Process Application, stejně jako snímek komponenty Toolkit. Navíc můžete také implementovat vrchol komponenty Process Application nebo vrchol komponenty Toolkit. (*Vrchol* je aktuální pracovní verze vaší komponenty Process Application či Toolkit.) Kontext verze se různí podle typu implementace.

V případě komponent Process Application se vrchol či specifický snímek používá jako jedinečný identifikátor verze.

Komponenty Toolkit lze implementovat s jednou či více komponentami Process Application, ale životní cyklus každé komponenty Toolkit je svázán s životním cyklem dané komponenty Process Application. Každá komponenta Process Application obsahuje vlastní kopii závislé komponenty Toolkit či komponent Toolkit implementovaných na serveru. Implementovaná komponenta Toolkit není sdílena mezi komponentami Process Application.

Pokud je stopa přidružená ke komponentě Process Application pojmenovaná jinak než výchozím názvem **Hlavní**, je akronym trasy také součástí kontextu dané verze.

Snímky komponenty Process Application

V případě implementací snímků komponenty Process Application je kontext verze kombinací následujících položek:

- Akronym názvu komponenty Process Application.
- Akronym stopy komponenty Process Application (v případě použití jiné stopy než **Hlavní**).
- Akronym snímku komponenty Process Application.

Samostatné komponenty Toolkit

V případě implementací snímků komponent Toolkit je kontext verze kombinací následujících položek:

- Akronym názvu komponenty Toolkit.
- Akronym stopy komponenty Toolkit (v případě použití jiné stopy než **Hlavní**).
- Akronym snímku komponenty Toolkit.

Tipy

Během iterativního testování v produktu Process Designer se používají vrcholy komponenty Process Application. Lze je implementovat pouze na servery Process Center.

V případě implementací vrcholu komponenty Process Application je kontext verze kombinací následujících položek:

- Akronym názvu komponenty Process Application.
- Akronym stopy komponenty Process Application (v případě použití jiné stopy než **Hlavní**).
- "Vrchol".

Vrcholy komponenty Toolkit se používají také během iterativního testování v produktu Process Designer. Neimplementují se na produkční server.

V případě implementací vrcholu komponenty Toolkit je kontext verze kombinací následujících položek:

- Akronym názvu komponenty Toolkit.
- Akronym stopy komponenty Toolkit (v případě použití jiné stopy než **Hlavní**).
- "Vrchol".

Příklady

Prostředky by měly být opatřeny jedinečným názvem a externě identifikovány pomocí kontextu verze.

- V následující tabulce jsou zobrazeny příklady názvů, které jsou identifikovány jedinečně. V tomto příkladu používá vrchol komponenty Process Application výchozí název stopy (**Hlavní**):

Tabulka 26. Vrchol komponenty Process Application s výchozím názvem stopy

Typ názvu	**Příklad**
Název komponenty Process Application	**Process Application 1**
Akronym názvu komponenty Process Application	**PA1**
Stopa komponenty Process Application	**hlavní**
Akronym stopy komponenty Process Application	“” (když je stopa **Hlavní**)
Snímek komponenty Process Application	
Akronym snímku komponenty Process Application.	**Rada**

Všechny moduly SCA přidružené k tomuto vrcholu komponenty Process Application zahrnují daný kontext verze, jak je uvedeno v následující tabulce:

Tabulka 27. Moduly SCA a soubory EAR s ohledem na verzi

Název modulu SCA	**Název s ohledem na verzi**	**Název souboru EAR/aplikace s ohledem na verzi**
M1	PA1-Rada-M1	PA1-Rada-M1.ear
M2	PA1-Rada-M2	PA1-Rada-M2.ear

- V následující tabulce je zobrazen příklad vrcholu komponenty Process Application, která používá jiný než výchozí název stopy:

Tabulka 28. Vrchol komponenty Process Application s jiným než výchozím názvem stopy

Typ názvu	**Příklad**
Název komponenty Process Application	**Process Application 1**
Akronym názvu komponenty Process Application	**PA1**
Stopa komponenty Process Application	**Stopa1**
Akronym stopy komponenty Process Application	**T1**
Snímek komponenty Process Application	
Akronym snímku komponenty Process Application.	**Rada**

Všechny moduly SCA přidružené k tomuto vrcholu komponenty Process Application zahrnují daný kontext verze, jak je uvedeno v následující tabulce:

Tabulka 29. Moduly SCA a soubory EAR s ohledem na verzi

Název modulu SCA	Název s ohledem na verzi	Název souboru EAR/aplikace s ohledem na verzi
M1	PA1-T1-Rada-M1	PA1-T1-Rada-M1.ear
M2	PA1-T1-Rada-M2	PA1-T1-Rada-M2.ear

Konvence pojmenování pro implementace komponenty Process Server

Na komponentu Process Server můžete implementovat snímek komponenty Process Application. Akronym snímku komponenty Process Application se používá za účelem jedinečné identifikace verze.

V případě implementací snímků komponenty Process Application je kontext verze kombinací následujících položek:

- Akronym názvu komponenty Process Application.
- Akronym snímku komponenty Process Application.

Prostředky by měly být opatřeny jedinečným názvem a externě identifikovány pomocí kontextu verze. V následující tabulce jsou zobrazeny příklady názvů, které jsou identifikovány jedinečně:

Tabulka 30. Příklad názvů a akronymů

Typ názvu	Příklad
Název komponenty Process Application	**Process Application 1**
Akronym názvu komponenty Process Application	**PA1**
Snímek komponenty Process Application	**1.0.0**
Akronym snímku komponenty Process Application.	**1.0.0**

V případě prostředku, jako např. modulu či knihovny, je kontext verze součástí jeho identifikace.

V následující tabulce je zobrazen příklad dvou modulů a také zahrnutí kontextu verze do přidružených souborů EAR:

Tabulka 31. Moduly SCA a soubory EAR s ohledem na verzi

Název modulu SCA	Název s ohledem na verzi	Název souboru EAR/aplikace s ohledem na verzi
M1	PA1-1.0.0-M1	PA1-1.0.0-M1.ear
M2	PA1-1.0.0-M2	PA1-1.0.0-M2.ear

V následující tabulce je zobrazen příklad dvou knihoven s oborem komponenty Process Application a také zahrnutí kontextu verze do přidružených souborů JAR:

Tabulka 32. Knihovny s oborem komponenty Process Application a soubory JAR s ohledem na verzi

Název knihovny s oborem komponenty Process Application architektury SCA	Název s ohledem na verzi	Název souboru JAR s ohledem na verzi
Lib1	PA1-1.0.0-Lib1	PA1-1.0.0-Lib1.jar
Lib2	PA1-1.0.0-Lib2	PA1-1.0.0-Lib2.jar

Vazba s ohledem na verzi

Komponenty Process Application mohou obsahovat moduly SCA zahrnující vazby exportu a importu. V případě souběžné implementace aplikací musí být vazba pro každou verzi dané aplikace jedinečná. Některé vazby jsou během

implementace automaticky aktualizovány, aby tak byla mezi verzemi zajištěna jedinečnost. Jindy je v zájmu zajištění jedinečnosti vazby nutné po implementaci tuto vazbu aktualizovat.

Rozsah vazby *s ohledem na verzi* je vymezený pro konkrétní verzi komponenty Process Application, což zaručuje její jedinečnost mezi těmito komponentami. Následující oddíl popisuje vazby, které jsou automaticky aktualizovány tak, aby braly ohled na verzi, stejně jako akce, které je třeba provést za běhu v případě, že vazba nefunguje s ohledem na verzi. Informace o bodech, které je třeba zvážit při vytváření modulů, viz "Aspekty vytváření vazeb".

SCA

Pokud jsou vazby exportu a importu daného modulu definovány ve stejném oboru komponenty Process Application, dojde během implementace k automatickému přejmenování cíle vazby SCA na takový, který bude brát ohled na verzi.

Pokud dané vazby ve stejném oboru komponenty Process Application definovány nejsou, dojde k zaprotokolování informační zprávy. Po implementaci musíte upravit vazbu importu za účelem změny cílové adresy koncového bodu. Cílovou adresu koncového bodu můžete změnit prostřednictvím administrativní konzoly.

Webová služba (JAX-WS nebo JAX-RPC)

Cílová adresa koncového bodu vazby webové služby je během implementace automaticky přejmenována tak, aby brala ohled na verzi, pokud jsou všechny následující podmínky pravdivé:

- V případě adresy jste postupovali v souladu s výchozí konvencí pojmenování:
  http://*ip*:*port*/*NázevModulu*Web/sca/*Název Exportu*
- Adresa koncového bodu je SOAP/HTTP.
- Vazby exportu a importu daného modulu byly definovány v rámci stejného oboru komponenty Process Application.

Pokud tyto podmínky pravdivé nejsou, dojde k zaprotokolování informační zprávy. Následující akce potom závisí na způsobu implementace vaší komponenty Process Application:

- Pokud souběžně implementujete svoji komponentu Process Application, musíte ručně přejmenovat adresu URL koncového bodu SOAP/HTTP nebo cílovou frontuSOAP/JMS tak, aby byla mezi verzemi dané komponenty Process Application jedinečná. Po implementaci můžete cílovou adresu koncového bodu změnit prostřednictvím administrativní konzoly.
- Pokud implementujete pouze jedinou verzi komponenty Process Application, můžete tuto zprávu ignorovat

V případě souběžné implementace snímku vazby webové služby SOAP/JMS se typ akce, kterou budete provádět, odvíjí od způsobu implementace vaší komponenty Process Application:

- Pokud se import a cílový export nachází ve stejné komponentě Process Application, postupujte před publikováním komponenty Process Application do komponenty Process Center a vytvořením snímků takto:
  1. Změňte adresu URL koncového bodu exportu. Ujistěte se, že jsou cíl a továrna připojení jedinečné.
  2. Změňte adresu URL koncového bodu importu tak, aby se shodovala s adresou, kterou jste v předchozím kroku určili pro export.
- Pokud se import a cílový export nachází v různých komponentách Process Application, postupujte takto:
  1. Změňte adresu URL koncového bodu exportu. Ujistěte se, že jsou cíl a továrna připojení jedinečné.
  2. Publikujte komponentu Process Application do komponenty Process Center.
  3. Vytvořte snímek.
  4. Implementujte komponentu Process Application na komponentu Process Server.
  5. Pomocí administrativní konzoly WebSphere Administrative Console změňte adresu URL koncového bodu odpovídajícího importu tak, aby se shodovala s adresou určenou pro export.

HTTP

Adresa URL koncového bodu vazby HTTP je během implementace automaticky přejmenována tak, aby brala ohled na verzi, pokud jsou všechny následující podmínky pravdivé:

- V případě adresy jste postupovali v souladu s výchozí konvencí pojmenování:
  http(s)://*ip*:*port*/*NázevModulu*Web/*kontextováCestavExportu*
- Vazby exportu a importu daného modulu byly definovány v rámci stejného oboru komponenty Process Application.

Pokud tyto podmínky pravdivé nejsou, dojde k zaprotokolování informační zprávy. Následující akce potom závisí na způsobu implementace vaší komponenty Process Application:

- Pokud souběžně implementujete svoji komponentu Process Application, musíte ručně přejmenovat adresu URL koncového bodu tak, aby byla mezi verzemi dané komponenty Process Application jedinečná. Po implementaci můžete cílovou adresu koncového bodu změnit prostřednictvím administrativní konzoly.
- Pokud implementujete pouze jedinou verzi komponenty Process Application, můžete tuto zprávu ignorovat.

Platforma JMS a generická platforma JMS

Systémem generovaná služba JMS a generické vazby JMS automaticky berou ohled na verzi.

Poznámka: V případě služby JMS a generických vazeb služby JMS definovaných uživatelem nedochází během implementace k automatickému přejmenování, aby tak mohly vazby brát ohled na verzi. Pokud je vazba definovaná uživatelem, musíte následující atributy přejmenovat tak, aby byly mezi verzemi dané komponenty Process Application jedinečné:

- Konfigurace koncového bodu.
- Cílová fronta příjmu.
- Název portu modulu listener (pokud je definovaný).

Pokud změníte koncový bod cílového modulu, nastavte odpovídající cíl odeslání.

MQ/JMS a MQ

Během implementace nedochází k automatickému přejmenování, aby tak mohly vazby typu MQ/JMS či MQ brát ohled na verzi.

Následující atributy musíte přejmenovat tak, aby byly mezi verzemi dané komponenty Process Application jedinečné:

- Konfigurace koncového bodu.
- Cílová fronta příjmu.

Pokud změníte koncový bod cílového modulu, nastavte odpovídající cíl odeslání.

EJB

Během implementace nedochází k automatickému přejmenování, aby tak mohly vazby typu EJB brát ohled na verzi.

Atribut Názvy rozhraní JNDI musíte přejmenovat tak, aby byl mezi verzemi dané komponenty Process Application jedinečný.

Poznámka: Je třeba aktualizovat také klientské aplikace, aby používali nové názvy rozhraní JNDI.

EIS

Adaptér prostředku je během implementace automaticky přejmenován tak, aby bral ohled na verzi, pokud ovšem nebyl změněn výchozí název prostředku (***NázevModulu*App:Adapter Description**).

Pokud výchozí název prostředku změněn byl, musí být názvy adaptérů prostředku mezi verzemi dané komponenty Process Application jedinečné.

Pokud názvy adaptérů prostředku jedinečné nejsou, dojde během implementace k zaprotokolování informační zprávy, která vás na tuto skutečnost upozorní. Pomocí administrativní konzoly můžete adaptéry prostředku přejmenovat po dokončení implementace ručně.

Dynamické vyvolání s ohledem na verzi

Můžete definovat komponenty mediačního toku, které budou směrovat zprávy na koncové body, které jsou určovány dynamicky za běhu. Při vytváření mediačního modulu konfigurujete vyhledání koncového bodu tak, aby používalo trasování s ohledem na verzi.

Pokud pro snímek používáte styl IBM_VRM (<*verze*>.<*vydání*>.<*úprava*), můžete exportovat soubor EAR komponenty Process Application do produktu WSRR (WebSphere Service Registry and Repository). Když konfigurujete mediační modul, konfigurujete vyhledání koncového bodu tak, aby používalo směrování s ohledem na verzi. Například v poli **Zásada shody** vyberete volbu **Vrátit koncový bod odpovídající nejnovější kompatibilní verzi služeb založených na modulu SCA** a pro položku **Typ vazby** vyberete volbu **SCA**.

Budoucí verze komponenty Process Application budou implementovány na server a publikovány do produktu WSRR. Vyhledávání koncového bodu mediačního modulu dynamicky vyvolá nejnovější kompatibilní verzi koncového bodu dané služby.

Poznámka: Můžete také nastavit cíl v záhlaví SMOHeader a příslušnou hodnotu může nést zpráva požadavku.

Implementace komponent Process Application s moduly a projekty jazyka Java

Komponenty Process Application mohou obsahovat vlastní moduly Java EE a projekty Java. V případě souběžné implementace aplikací musí být vlastní modul Java EE pro každou verzi dané aplikace jedinečný.

Poznámka: Vlastní moduly Java EE a projekty Java jsou implementovány na server v případě, že jsou implementovány s modulem SCA s deklarovanou závislostí. Pokud při deklaraci dané závislosti nevyberete volbu **Implementovat s modulem** (což je výchozí nastavení), musíte modul či projekt implementovat ručně.

Implementace komponent Process Application s obchodními pravidly a selektory

Pokud implementujete více verzí komponenty Process Application, které obsahují komponentu obchodního pravidla nebo selektoru, všímejte si způsobu, jakým tyto verze používají přidružená metadata.

Dynamická metadata komponenty obchodního pravidla či selektoru se definují za běhu na základě názvu, cílového oboru názvů a typu dané komponenty. Pokud dojde k implementaci dvou či více verzí komponenty Process Application obsahujících obchodní pravidlo či selektor do stejného běhového prostředí, budou tyto sdílet stejná metadata logiky pravidla (obchodní pravidlo) nebo směrování (selektor).

Chcete-li umožnit, aby každá verze komponenty obchodního pravidla či selektoru v rámci komponenty Process Application používala vlastní dynamická metadata (logiky pravidla či směrování), refaktorujte cílový obor názvů tak, aby byl pro každou verzi komponenty Process Application jedinečný.

Architektura implementace

Architektura implementace produktu IBM Business Process Manager sestává ze softwarových procesů, nazývaných servery, topologických jednotek, na které je odkazováno jako na uzly a buňky, a z úložiště konfigurací používaného k ukládání informací o konfiguraci.

Buňky

V rámci produktu IBM Business Process Manager jsou *buňky* logická seskupení jednoho či více uzlů rozdělené sítě.

Buňka je konceptem konfigurace. Je to způsob, kterým administrátoři vzájemně logicky přidružují uzly. Na základě specifických kritérií, která dávají smysl v rámci organizačních prostředí, vytvoří administrátoři uzly, ze kterých daná buňka sestává.

Konfigurační data administrace jsou uložena v souborech XML. Buňka uchovává soubory hlavní konfigurace každého serveru v každém svém uzlu. Každý uzel a server má také vlastní lokální konfigurační soubory. Změny v lokálním uzlu nebo v konfiguračním souboru serveru jsou dočasné, pokud tento server patří k dané buňce. Aktuálně platné lokální změny potlačují konfiguraci buňky. Změny provedené v konfiguračních souborech hlavního serveru a hlavního uzlu na úrovni buňky nahrazují jakékoli dočasné změny provedené v uzlu při jeho synchronizaci s konfiguračními dokumenty buňky. K synchronizaci dochází při určitých událostech, jako např. při spuštění serveru.

Servery

Jádro funkčnosti produktu IBM Business Process Manager poskytují servery. Servery Process Server rozšiřují či zvyšují schopnost aplikačního serveru zpracovávat moduly architektury SCA (Service Component Architecture). Jiné servery (správci implementace a agenti uzlu) se používají ke správě serverů Process Server.

Server Process server může být buď *samostatný server*, nebo *spravovaný server*. Spravovaný server může volitelně být členem *klastru*. Kolekce spravovaných serverů, serverových klastrů a jiného middleware se nazývá *prostředí implementace*. V rámci prostředí implementace je každý spravovaný server nakonfigurovaný pro specifickou funkci (např. cílový hostitel, hostitel modulu aplikace, či server Common Event Infrastructure). Samostatný server je nakonfigurovaný tak, aby poskytoval všechny povinné funkce.

Servery poskytují běhové prostředí pro moduly SCA, pro prostředky, které tyto moduly používají (zdroje dat, specifikace aktivace a cíle služby JMS), a také pro prostředky dodané IBM (cíle zpráv, kontejnery komponenty Business Process Choreographer a servery Common Event Infrastructure).

Agent uzlu je administrativní agent reprezentující uzel pro váš systém a spravující servery tohoto uzlu. Agenti uzlu sledují servery hostitelského systému a trasují administrativní požadavky na servery. K vytvoření agenta uzlu dochází při sdružení uzlu v rámci správce implementace.

Správce implementace je administrativní agent poskytující centralizovaný pohled na správu více serverů a klastrů.

Samostatný server je definován samostatným profilem. Správce implementace je definován profilem správce implementace. Spravované servery se vytvářejí v rámci *spravovaného uzlu*, který je definován vlastním profilem.

Samostatné servery

Samostatný server poskytuje prostředí pro implementaci modulů SCA v jediném procesu serveru. Tento proces serveru zahrnuje mimo jiné administrativní konzolu, cíl implementace, podporu systému zpráv, správce Business Process Rules Manager a server CEI (Common Event Infrastructure).

Samostatný server lze snadno nastavit a má konzolu První kroky, ze které můžete spustit s zastavit server a otevřít galerii ukázek a administrativní konzolu. Pokud nainstalujete ukázky produktu IBM Business Process Manager a poté otevřete galerii ukázek, provede se implementace ukázkového řešení na samostatný server. Prostředky použité v této ukázce můžete zkoumat v administrativní konzole.

Na samostatný server můžete implementovat i svá řešení, ale samostatný server nemůže zajistit kapacitu, rozšiřitelnost ani odolnost, které jsou vyžadovány po produkčním prostředí. Pro produkční prostředí je lepší použít prostředí síťové implementace.

Je možné začít samostatným serverem a později jej zahrnout do prostředí síťové implementace, a to federováním do buňky správce implementace, *pokud do této buňky nebyly federovány žádné jiné uzly*. Do jedné buňky nelze federovat

více samostatných serverů. Chcete-li federovat samostatný server, použijte administrativní konzolu správce implementace nebo příkaz **addNode**. Při federování pomocí příkazu **addNode** nesmí být samostatný server spuštěný.

Samostatný server je definován profilem samostatného serveru.

Klastry

Klastry jsou skupiny společně spravovaných serverů podílejících se na správě pracovní zátěže.

Klastr může obsahovat uzly nebo individuální aplikační servery. Uzel je zpravidla fyzický počítačový systém s jasně určenou adresou IP hostitele, na kterém je spuštěn minimálně jeden aplikační server. Klastry lze seskupit v rámci konfigurace buňky, čímž dojde k logickému vzájemnému přidružení mnoha serverů a klastrů, stejně jako aplikací s různými konfiguracemi. Výběr závisí čistě na příslušném administrátorovi a odvíjí se od toho, co je smysluplné v rámci jeho organizačního prostředí.

Klastry zodpovídají za vyrovnávání pracovní zátěže mezi servery. Servery, které jsou součástí klastru, se nazývají členy klastru. Když instalujete aplikaci na klastr, dojde k její automatické instalaci na každý člen klastru.

Vzhledem k tomu, že každý člen klastru obsahuje stejné aplikace, můžete úlohy klienta rozdělit na základě kapacity různých počítačů, a to tím, že každému serveru přiřadíte váhu.

Přiřazením váhy serverům klastru dosáhnete zlepšení výkonu a lepšího překonávání selhání. Úlohy se přiřazují serverům, které mají kapacitu na provedení operací úloh. Pokud je určitý server pro provedení dané úlohy nedostupný, přiřadí se tato úloha jinému členu klastru. Tato schopnost nového přiřazení má zjevné výhody oproti spuštění jediného aplikačního serveru, který se může v případě příliš velkého počtu požadavků přetížit.

Profily

Profil definuje jedinečné běhové prostředí se samostatnými příkazovými soubory, konfiguračními soubory a soubory protokolů. Profily definují v systémech produktu IBM Business Process Manager tři různé typy prostředí: samostatný server, správce implementace a spravovaný uzel.

S použitím profilů můžete mít v systému více než jedno běhové prostředí, aniž byste museli instalovat více kopií binárních souborů produktu IBM Business Process Manager.

Profily lze vytvářet pomocí nástroje Správa profilu nebo obslužného programu příkazového řádku **manageprofiles**.

Poznámka: Na distribuovaných platformách má každý uzel jedinečný název. Na platformě z/OS jsou všechny profily pojmenovány jako “default” nelze profily přejmenovat, upravit, kopírovat ani odstranit.

Adresář profilu

Každý profil v systému má vlastní adresář obsahující všechny jeho soubory. Umístění adresáře profilu určujete při vytváření profilu. Při výchozím nastavení se profil nachází v adresáři profiles v adresáři, kde je nainstalován produkt IBM Business Process Manager. Například: profil Dmgr01 je v adresáři C:\Program Files\IBM\WebSphere\ProcServer\profiles\Dmgr01.

Konzola První kroky

Každý profil v systému má konzolu První kroky. Pomocí tohoto rozhraní se můžete seznámit se samostatným serverem, správcem implementace nebo spravovaným uzlem.

Výchozí profil

Prvním profilem, který vytvoříte v rámci jedné instalace produktu IBM Business Process Manager, je *výchozí profil*. Výchozí profil představuje výchozí cíl pro příkazy vydávané z adresáře bin v adresáři, kde byl nainstalován produkt IBM Business Process Manager. Existuje-li v systému pouze jediný profil, bude každý z příkazů pracovat s tímto profilem. Pokud vytvoříte další profil, můžete jej nastavit jako výchozí.

Poznámka: Výchozím profilem nemusí být nutně profil s názvem “default”.

Rozšiřování profilů

Pokud již máte vytvořen profil správce implementace, vlastní profil nebo profil samostatného serveru pro produkt WebSphere Application Server Network Deployment či produkt WebSphere ESB můžete tento profil *rozšířit*, aby kromě existující funkce podporoval také produkt IBM Business Process Manager. Chcete-li rozšířit profil, nejdříve nainstalujte produkt IBM Business Process Manager. Poté použijte nástroj Správa profilu nebo obslužný program příkazového řádku **manageprofiles**.

Omezení: Nelze rozšířit profil, jenž definuje spravovaný uzel, který je již federován do správce implementace.

Správci implementace

Správce implementace je server, který spravuje operace pro logickou skupinu, nebo buňku ostatních serverů. Správce implementace je centrální umístění pro administraci serverů a klastrů.

Při vytváření prostředí implementace je prvním profilem, který vytvoříte, profil správce implementace. Správce implementace má konzolu První kroky, z níž můžete spustit a zastavit správce implementace a spustit jeho administrativní konzolu. Administrativní konzola správce implementace se používá ke správě serverů a klastrů v buňce. To zahrnuje konfiguraci serverů a klastrů, přidávání serverů do klastrů, spouštění a zastavování serverů a klastrů a implementaci modulů SCA.

Ačkoli je správce implementace typ serveru, nelze implementovat moduly přímo do správce implementace.

Uzly

Uzel je logické seskupení spravovaných serverů.

Uzel obvykle odpovídá logickému nebo fyzickému počítačovému systému s jasně určenou adresou IP hostitele. Uzel nemůže zahrnovat více počítačů. Názvy uzlů jsou obvykle identické s názvem hostitele pro daný počítač.

V rámci topologie síťové implementace mohou být uzly spravované nebo nespravované. Spravovaný uzel obsahuje proces agenta uzlu, který spravuje jeho konfiguraci a servery. Nespravované uzly agenta uzlu nemají.

Spravované uzly

Spravovaný uzel je uzel, který je federován do správce implementace, obsahuje agenta uzlu a může obsahovat spravované servery. Ve spravovaném uzlu je možné konfigurovat a spouštět spravované servery.

Servery, které jsou nakonfigurované ve spravovaném uzlu tvoří prostředky vašeho prostředí implementace. Tyto servery jsou vytvářeny, konfigurovány, spouštěny, zastavovány, spravovány a odstraňovány pomocí administrativní konzoly správce implementace.

Spravovaný uzel obsahuje agenta uzlu, který spravuje všechny servery v uzlu.

Při federování uzlu se automaticky vytvoří proces agenta uzlu. Tento agent uzlu musí být spuštěný, abyste mohli spravovat konfiguraci profilu. Například při provádění následujících úloh:

- Spouštění a zastavování procesů.
- Synchronizace konfiguračních dat ve správci implementace s kopií v uzlu.

Agent uzlu však nemusí být spuštěný, chcete-li spouštět aplikace nebo konfigurovat prostředky v uzlu.

Spravovaný uzel může obsahovat jeden či více serverů, které spravuje správce implementace. Na serverech ve spravovaném uzlu můžete implementovat řešení, ale spravovaný uzel neobsahuje galerii ukázkových aplikací. Spravovaný uzel je definován vlastním profilem a má konzolu První kroky.

Nespravované uzly
Nespravovaný uzel nemá agenta uzlu, který by spravoval jeho servery.

V rámci topologie síťové implementace mohou nespravované uzly obsahovat definice serverů, jako např. webových severů, ale nikoliv definice aplikačních serverů. Nespravované uzly nikdy nemohou být sdružené. To znamená, že agenta uzlu nelze nikdy přidat do nespravovaného uzlu. Dalším typem nespravovaného uzlu je samostatný server. Správce implementace tento samostatný server nemůže spravovat, protože jej buňka nezná. Samostatný server může být sdružený. Když sdružený je, dojde k automatickému vytvoření agenta uzlu. Z uzlu se tak stává spravovaný uzel dané buňky.

Agenti uzlů
Agenti uzlů jsou administrativní agenti, kteří trasují administrativní požadavky na servery.

Agent uzlu je server spuštěný v systému každého hostitelského počítače, který se podílí na konfiguraci síťové implementace. Jedná se čistě o administrativního agenta, který se nepodílí na funkcích sloužících účelům aplikací. Agent uzlu je také hostitelem dalších důležitých administrativních funkcí, jako např. služeb pro přenos souborů, synchronizace konfigurace, či monitorování výkonu.

Aspekty pojmenování profilů, uzlů, serverů, hostitelů a buněk
Toto téma pojednává o podmínkách a otázkách, které je třeba zohlednit při pojmenování vašeho profilu, uzlu, serveru, hostitele či buňky (je-li to použitelné). Toto téma platí pro distribuované platformy.

Aspekty pojmenování profilu

Profil může mít libovolný jedinečný název s následujícími omezeními. Při pojmenování profilu nepoužívejte následující znaky:

- mezery,
- speciální znaky, které nejsou povoleny v názvu adresáře ve vašem operačním systému, jako např. *, &, nebo ?
- lomítka (/) a zpětná lomítka (\)

Dvoubajtové znaky jsou povolené.

Windows **Aspekty cesty k adresáři:** Cesta k instalačnímu adresáři musí mít nejvýše 60 znaků. Název adresáře *cesta_k_adresáři_profilů\název_profilu* musí obsahovat nejvýše 80 znaků.

Aspekty pojmenování uzlu, serveru, hostitele a buňky

Vyhrazené názvy: Vyhněte se použití vyhrazených názvů jako hodnot polí. Použití vyhrazených názvů může způsobit nepředvídatelné výsledky. Vyhrazena jsou následující slova:

- Buňky.
- Uzly.
- Servery.
- Klastry.
- Aplikace.
- Implementace.

Popisy polí na stránkách Název uzlu a hostitele a Název uzlu, hostitele a buňky: Tabulka 11 na stránce 42 popisuje pole na stránkách Název uzlu a hostitele a Název uzlu, hostitele a buňky v nástroji Správa profilu, včetně názvů polí, výchozích hodnot a omezení. Tyto informace použijte jako vodítko při vytváření profilů.

Tabulka 33. Pokyny pro pojmenování uzlů, serverů, hostitelů a buněk

Název pole	Výchozí hodnota	Omezení	Popis
Profily samostatného serveru			
Název uzlu	Linux UNIX Windows *krátkýNázevHostitele* Node *ČísloUzlu* kde: • *krátkýNázevHostitele* je krátký název hostitele. • *ČísloUzlu* je pořadové číslo počínaje hodnotou 01.	Vyhněte se použití vyhrazených názvů.	Vyberte libovolný požadovaný název. Plánujete-li v systému vytvořit více než jeden server, v zájmu lepšího uspořádání instalace použijte jedinečný název.
Název serveru	Linux UNIX Windows server1	Pro server použijte jedinečný název.	Logický název serveru.
Název hostitele	Linux UNIX Windows Dlouhý tvar názvu serveru názvů domény (DNS).	Název hostitele musí být adresovatelný prostřednictvím vaší sítě. Plánujete-li používat prostor Business Space, použijte úplný název hostitele.	Použijte skutečný název DNS nebo adresu IP vaší pracovní stanice, abyste umožnili komunikaci s ní. Viz další informace o názvu hostitele za touto tabulkou.
Název buňky	Linux UNIX Windows *krátkýNázevHostitele* Node *ČísloUzlu* Cell kde: • *krátkýNázevHostitele* je krátký název hostitele. • *ČísloUzlu* je pořadové číslo počínaje hodnotou 01.	Pro buňku použijte jedinečný název. Název buňky musí být jedinečný vždy, pokud je produkt spuštěn na stejné fyzické pracovní stanici či klastru pracovních stanic, jako např. v prostředí Sysplex. Název buňky navíc musí být jedinečný vždy, když je požadována připojitelnost v síti mezi entitami, buď mezi buňkami, nebo od klienta, který musí komunikovat s každou z buněk. Názvy buněk musí být jedinečné také tehdy, pokud mají být federovány jejich obory názvů. Jinak by mohly nastat problémy, například výjimka javax.naming.Name NotFoundException; v takovém případě je třeba vytvořit buňky s jedinečnými názvy.	Všechny federované uzly se stanou členy buňky správce implementace.
Profily správce implementace			

Tabulka 33. Pokyny pro pojmenování uzlů, serverů, hostitelů a buněk (pokračování)

Název pole	Výchozí hodnota	Omezení	Popis
Název uzlu	Linux UNIX Windows *krátkýNázevHostitele* Cell Manager*ČísloUzlu* kde: • *krátkýNázevHostitele* je krátký název hostitele. • *ČísloUzlu* je pořadové číslo počínaje hodnotou 01.	Pro správce implementace použijte jedinečný název. Vyhněte se použití vyhrazených názvů.	Název se používá pro administraci v rámci buňky správce implementace.
Název hostitele	Linux UNIX Windows Dlouhý tvar názvu serveru názvů domény (DNS).	Název hostitele musí být adresovatelný prostřednictvím vaší sítě. Vyhněte se použití vyhrazených názvů. Plánujete-li používat prostor Business Space, použijte úplný název hostitele.	Použijte skutečný název DNS nebo adresu IP vaší pracovní stanice, abyste umožnili komunikaci s ní. Viz další informace o názvu hostitele za touto tabulkou.
Název buňky	Linux UNIX Windows *krátkýNázevHostitele* Cell *ČísloBuňky* kde: • *krátkýNázevHostitele* je krátký název hostitele. • *ČísloBuňky* je pořadové číslo počínaje hodnotou 01.	Pro buňku správce implementace použijte jedinečný název. Název buňky musí být jedinečný vždy, pokud je produkt spuštěn na stejné fyzické pracovní stanici či klastru pracovních stanic, jako např. v prostředí Sysplex. Název buňky navíc musí být jedinečný vždy, když je požadována připojitelnost v síti mezi entitami, buď mezi buňkami, nebo od klienta, který musí komunikovat s každou z buněk. Názvy buněk musí být jedinečné také tehdy, pokud mají být federovány jejich obory názvů. Jinak by mohly nastat problémy, například výjimka javax.naming.Name NotFoundException; v takovém případě je třeba vytvořit buňky s jedinečnými názvy.	Všechny federované uzly se stanou členy buňky správce implementace, kterou pojmenujete na stránce Název uzlu, hostitele a buňky v nástroji Správa profilu.
Vlastní profily			

Tabulka 33. Pokyny pro pojmenování uzlů, serverů, hostitelů a buněk (pokračování)

Název pole	Výchozí hodnota	Omezení	Popis
Název uzlu	Linux UNIX Windows *krátkýNázevHostitele* Node *ČísloUzlu* kde: • *krátkýNázevHostitele* je krátký název hostitele. • *ČísloUzlu* je pořadové číslo počínaje hodnotou 01.	Vyhněte se použití vyhrazených názvů. V rámci buňky správce implementace použijte jedinečný název.	Název se používá pro administraci v rámci buňky správce implementace, do které je vlastní profil přidáván. V rámci buňky správce implementace použijte jedinečný název.
Název hostitele	Linux UNIX Windows Dlouhý tvar názvu serveru názvů domény (DNS).	Název hostitele musí být adresovatelný prostřednictvím vaší sítě. Plánujete-li používat prostor Business Space, použijte úplný název hostitele.	Použijte skutečný název DNS nebo adresu IP vaší pracovní stanice, abyste umožnili komunikaci s ní. Viz další informace o názvu hostitele za touto tabulkou.

Aspekty názvu hostitele:

Název hostitele je síťový název fyzické pracovní stanice, na které je nainstalován uzel. Název hostitele musí být možné přeložit na síťový uzel na serveru. Pokud na serveru existuje více síťových karet, název hostitele nebo adresu IP musí být možné přeložit na jednu ze síťových karet. Pomocí názvu hostitele se k tomuto uzlu připojují vzdálené uzly, které s ním poté komunikují.

Produkt IBM Business Process Manager je kompatibilní s verzí 4 protokolu IP (IPv4) i verzí 6 tohoto protokolu (IPv6). Kdykoli v administrativní konzole nebo jinde zadáváte adresy IP, můžete tak učinit v obou těchto formátech. Pokud je však na vašem systému implementován protokol IPv6, musíte zadat adresu IP ve formátu IPv6, a naopak, není-li vám protokol IPv6 zatím k dispozici, zadávejte adresy IP ve formátu IPv4. Další informace k IPv6 viz následující popis: IPv6.

Při určování vhodného názvu hostitele pro pracovní stanici vám mohou pomoci následující pokyny:

- Vyberte název hostitele, na který dosáhnou ostatní pracovní stanice v rámci vaší sítě.
- Pro tuto hodnotu nepoužívejte generický identifikátor lokální_hostitel (localhost).
- Nesnažte se nainstalovat produkty IBM Business Process Manager na server s názvem hostitele, který používá znaky z dvoubajtové znakové sady (DBCS). Při použití v názvu hostitele nejsou znaky DBCS podporovány.
- V názvech serverů nepoužívejte znak podtržítko (_). Standardy sítě Internet vyžadují, aby názvy domén splňovaly požadavky popsané v oficiálních standardech protokolů sítě Internet RFC 952 a RFC 1123. Názvy domén smí obsahovat pouze písmena (malá nebo velká) a číslice. Názvy domén dále mohou obsahovat pomlčky (-), avšak pomlčka nesmí být na koncích názvu. Podtržítka (_) nejsou v názvu hostitele podporována. Pokud jste produkt IBM Business Process Manager nainstalovali na server, který v názvu obsahuje podtržítko, do přejmenování můžete k tomuto serveru přistupovat pomocí jeho adresy IP.

Pokud na stejném počítači definujete koexistující uzly s jedinečnými adresami IP, definujte každou adresu IP ve vyhledávací tabulce serveru názvů domény (DNS). Konfigurační soubory serverů neposkytují rozlišování názvů domén v případě více adres IP na pracovní stanici s jedinou síťovou adresou.

Hodnota, kterou určíte pro název hostitele, se použije v konfiguračních dokumentech jako hodnota vlastnosti hostName. Hodnotu názvu hostitele můžete zadat v následujících formátech:

- Úplný řetězec názvu hostitele DNS, například xmachine.manhattan.ibm.com.
- Řetězec výchozího krátkého názvu hostitele DNS, například xmachine.
- Číselná adresa IP, například 127.1.255.3.

Úplný řetězec názvu hostitele DNS má výhodu naprosté jednoznačnosti a flexibility. Máte flexibilitu změny skutečné adresy IP hostitelského systému, aniž byste museli změnit konfiguraci serveru. Taková hodnota názvu hostitele je zvlášť užitečná, plánujete-li často měnit adresu IP při použití protokolu DHCP (Dynamic Host Configuration Protocol) k přiřazování adres IP. Nevýhodou tohoto formátu je závislost na serveru DNS. Není-li server DNS dostupný, je konektivita narušena.

Také krátký název hostitele umožňuje dynamické rozlišování. Formát krátkého názvu navíc může být předefinován v souboru lokálních hostitelů, takže systém může spustit server, i když je odpojen od sítě. Chcete-li spustit server při odpojení, v souboru lokálních hostitelů definujte krátký název jako 127.0.0.1 (lokální zpětná smyčka). Nevýhodou formátu krátkého názvu je závislost na serveru DNS v případě vzdáleného přístupu. Není-li server DNS dostupný, je konektivita narušena.

Číselná adresa IP má tu výhodu, že nevyžaduje rozlišení názvu prostřednictvím serveru DNS. K uzlu pojmenovanému číselnou adresou IP se vzdálený uzel může připojit i při nedostupnosti serveru DNS. Nevýhodou tohoto formátu je, že číselná adresa IP je pevná. Kdykoli změníte adresu IP pracovní stanice, musíte změnit i nastavení vlastnosti hostName v konfiguračních dokumentech. Z tohoto důvodu nepoužívejte číselnou adresu IP, používáte-li protokol DHCP nebo měníte-li pravidelně adresy IP. Další nevýhodou tohoto formátu je, že uzel nelze použít, pokud je hostitel odpojen od sítě.

BPMN 2.0

Definice obchodních procesů v produktu IBM Business Process Manager podporují podtřídu Common Executable třídy shody BPMN 2.0 Process Modeling, která se zabývá modely, jež lze spustit.

BPMN (Business Process Model and Notation) je základním standardem pro procesy v produktech IBM Process Designer a IBM Process Center. Na specifikaci BPMN jsou založeny diagramy definic obchodních procesů (BPD). Toto téma představuje některé způsoby, jak je standard BPMN 2.0 v produktu IBM Business Process Manager použitý. Podrobné informace o standardu BPMN naleznete na stránce specifikace BPMN na adrese http://www.bpmn.org/.

Produkt IBM Business Process Manager podporuje následující typy úloh dle BPMN 2.0:

- Není (abstraktní úloha ve specifikaci BPMN 2.0).
- Systémová úloha (úloha služby ve specifikaci BPMN 2.0).
- Uživatelská úloha.
- Skript.
- Rozhodovací úloha (úloha obchodního pravidla ve specifikaci BPMN 2.0).

Intermediační událost zpráv IBM BPM poskytuje podobné funkce jako úloha odeslání a úloha příjmu ve specifikaci BPMN.

Notace BPMN 2.0

Počínaje verzí 7.5.1 jsou v komponentě Process Designer ikony úloh BPMN 2.0 v diagramech BPD shromážděny na zjednodušené paletě a zobrazeny v diagramech procesů. Ikony ukazují, zda je daná aktivita systémovou úlohou, uživatelskou úlohou, rozhodovací úlohou, skriptem nebo propojeným procesem. Aktivity v modelech, které byly vytvořeny ve starších verzích, jsou při prohlížení ve verzi 7.5.1 a novější také zobrazeny s příslušnými typy úloh a ikonami úloh podle specifikace BPMN 2.0.

Aktivity a úlohy

Došlo k určitým terminologickým změnám oproti předchozím verzím produktu Process Designer. Řada těchto změn se týká typů aktivit, které byly přejmenovány.

- Aktivity Služba (automatická) jsou nyní systémové úlohy.
- Aktivity Služba (úloha) v nesystémové dráze jsou nyní uživatelské úlohy.
- Aktivity Služba (úloha) v systémové dráze jsou nyní rozhodovací úlohy, pokud odkazují na službu rozhodování.
- Aktivity Služba (úloha) v systémové dráze jsou nyní systémové úlohy, pokud odkazují na jakoukoli jinou službu než službu rozhodování.
- Aktivity Javascript jsou nyní úlohy skriptu.
- Aktivity Vnořený proces jsou nyní propojené procesy.
- Externí aktivity z předchozích verzí produktu Process Designer jsou dostupné jako externí implementace pro uživatelské úlohy nebo systémové úlohy.

Brány

Nedošlo k žádným změnám notace pro brány předchozích verzí. Došlo však ke třem změnám terminologie. Rozhodovací brána je nyní *výlučná brána*, brána jednoduchého rozdělení nebo spojení je nyní *paralelní brána* a brána podmíněného rozdělení nebo spojení je nyní *nevýlučná brána*.

Existuje také nový typ brány, *brána událostí*. Brána událostí představuje bod větvení v procesu, kde jsou alternativní cesty následující za danou branou založeny na událostech, k nimž dojde, a ne na vyhodnocení výrazů s použitím dat procesu (jako u výlučných a nevýlučných bran). Cestu, kterou se bude proces ubírat, určuje specifická událost, obvykle přijetí zprávy.

Nepřerušující události

Specifikace BPMN 2.0 přidala notaci pro nepřerušující události. Událost hranice standardně přeruší aktivitu, k níž je připojena. Při spuštění této události se aktivita zastaví a token postupuje dál odchozím tokem posloupnosti pro danou událost. Pokud je událost nastavena jako nepřerušující, pokračuje připojená aktivita při spuštění události paralelně a vygeneruje se nový token, který se předá dál odchozím tokem posloupnosti pro danou událost. Hranice události se u nepřerušujících událostí změní na čárkovanou čáru.

Intermediační události připojené k aktivitám jsou přerušující intermediační události, pokud své připojené aktivity zavírají, nebo nepřerušující intermediační události, pokud své připojené aktivity nezavírají.

Událost zahájení

Specifikace BPMN umožňuje v modelech procesů vynechat symboly událostí zahájení a událostí ukončení. Komponenta Process Designer vyžaduje, aby modely procesů používaly události zahájení a ukončení.

V komponentě Process Designer jsou k dispozici různé typy událostí zahájení:

procesy

- není
- zprávy
- ad hoc

podprocesy

- není

podprocesy událostí

- chyba
- zprávy

- časovač

Typ události zahájení můžete změnit tak, že upravíte vlastnosti události. V procesu může být celá řada událostí zahájení typu zpráva, ale jen jedna událost zahájení typu není.

Události ukončení

K dispozici jsou čtyři typy událostí ukončení: *zpráva*, *ukončit*, *chyba* a *není*. Typ události ukončení můžete změnit.

Když nadřízený proces volá proces podřízený a podřízený proces provede akci události typu ukončit, sémantika BPMN říká, že se podřízený proces okamžitě zastaví a nadřízený proces pak pokračuje dalšími kroky. Pokud v komponentě Process Designer provede podřízený proces aktivitu události ukončení, zastaví se podřízený i nadřízený proces.

Podprocesy

Specifikace BPMN definuje dva typy podprocesů, vestavěný a znovupoužitelný. V komponentě Process Designer můžete vytvořit oba typy. Vestavěné podprocesy se v produktu Process Designer nazývají prostě *podprocesy* a jsou ve verzi 7.5.1 nové. Znovupoužitelný podproces BPMN se v komponentě Process Designer nazývá *propojený proces*.

Podproces existuje v rámci procesu, který jej obsahuje, a přestavuje způsob, jak seskupit kroky procesu za účelem zjednodušení a zpřehlednění diagramu. Podprocesy sbalí několik kroků do jediné aktivity. Podproces je viditelný pouze pro proces, v němž je definován. Podproces existuje v rozsahu platnosti volajícího procesu a má přístup ke všem proměnným v tomto prostředí. Nedochází k žádnému předávání parametrů do vestavěného procesu a z něj.

Vedle podprocesu a propojeného procesu obsahuje komponenta Process Designer podproces události, což je specializovaný podproces, který se používá pro obsluhu událostí. Není propojen s jinými aktivitami prostřednictvím toku posloupnosti a dochází k němu jen tehdy, je-li spuštěna jeho událost zahájení.

Propojené procesy

Znovupoužitelný podproces dle BPMN se v produktu Process Designer nazývá *propojený proces*. Jde o proces vytvořený mimo aktuální proces, který může být aktuálním procesem volán. Je znovupoužitelný, protože definice jiných procesů mohou tento proces také volat. Propojený proces definuje své vstupní a výstupní parametry a nemá přístup k rozsahu platnosti nebo prostředí volajícího procesu. Propojený proces je podobný vnořenému procesu, který byl k dispozici v předchozích verzích; nedošlo k žádným změnám chování aktivity. Předchozí vnořené procesy se migrací převedou na propojené procesy. Propojený proces vypadá jako podproces s tlustou hranicí a v okně komponenty Inspector je zvýrazněný.

Cykly

Specifikace BPMN nabízí pojem aktivity, která se může opakovat. Aktivita může být atomická, což znamená, že se opakuje tato aktivita, nebo může jít o podproces, který zahrnuje řadu kroků, jež se opakují. Pokud rozbalíte opakovanou aktivitu, uvidíte obsažené aktivity, které se mají opakovaně spouštět. Podmínka se vyhodnotí vždy na začátku každé iterace cyklu. Nelze provést vyhodnocení na konci jednotlivých iterací cyklu.

Produkt IBM Business Process Manager obsahuje *cyklus s více instancemi*, který se provede konečně krát, přičemž aktivity v něm obsažené se spouští sekvenčně nebo paralelně.

Import jiných procesů než BPMN

Můžete importovat modely, které byly vytvořeny v produktu IBM WebSphere Business Modeler, a používat je v produktu Process Designer. Podrobnosti o importu produktů BPMN 2.0 viz Mapování prvků produktu IBM WebSphere Business Modeler na konstrukce produktu IBM Business Process Manager. Také můžete importovat modely BPMN 2.0, které byly vytvořeny v produktech IBM WebSphere Business Compass, Rational Software Architect nebo v jiných prostředích pro modelování.

Definice obchodního procesu (BPD)

Chcete-li modelovat určitý proces v produktu IBM Process Designer, je třeba vytvořit definici obchodního procesu (BPD).

Definice BPD představuje opakovaně použitelný model procesu definující aspekty, které jsou společné všem běhovým instancím daného modelu procesu. BPD musí obsahovat událost zahájení, událost ukončení, nejméně jednu dráhu a jednu či více aktivit. Podrobnosti o omezení znaků platných pro BPD viz "Konvence pojmenování produktu IBM Process Designer" v souvisejících odkazech.

Do definice BPD (Business Process Definition) je nutné přidat dráhu pro každý systém nebo skupinu uživatelů, kteří se účastní určitého procesu. Plavecká dráha může být účastnická dráha nebo systémová dráha. Pokud však budete chtít, můžete vytvořit BPD, která seskupuje aktivity skupiny a systému do jediné dráhy. Informace o vytvoření BPD viz "Vytvoření definice obchodního procesu (BPD)" v souvisejících odkazech.

Odpovědnost za aktivity v účastnické dráze můžete pověřit libovolnou specifickou osobu nebo skupinu. Každá dráha, kterou vytvoříte, je standardně přiřazena do skupiny účastníků Všichni uživatelé. Pomocí této výchozí skupiny účastníků můžete spustit a otestovat danou definici BPD v komponentě Inspector. Skupina účastníků Všichni uživatelé zahrnuje všechny uživatele, kteří jsou členy skupiny zabezpečení tw_allusers, což je speciální skupina zabezpečení, která automaticky zahrnuje všechny uživatele v systému.

Systémová dráha obsahuje aktivity ošetřené specifickým systémem produktu IBM Process Center. Každá aktivita potřebuje implementaci, která tuto aktivitu definuje a nastavuje vlastnosti úlohy. Během implementace vytvoří vývojář službu nebo napíše skript JavaScript nezbytný k realizaci aktivit v systémové dráze. Informace o službách viz "Základní informace o typech služeb" v souvisejících odkazech.

Pro každou vytvářenou definici BPD je nutné deklarovat proměnné, které umožní zachycení obchodních dat předávaných mezi jednotlivými aktivitami v rámci procesu. Informace o implementaci proměnných viz "Správa a mapování proměnných" v souvisejících odkazech.

Do definice BPD lze také přidávat události. Události v IBM BPM se spouštějí, když nastane určité datum, vyskytne se výjimka nebo je doručena zpráva. Typ události, kterou chcete implementovat, určuje požadovaný typ spouštěče. Podrobné informace o dostupných typech událostí a o jejich spouštěčích viz "Modelování událostí".

Při sestavování definic obchodních procesů v komponentě Process Designer je nutné provést určité konfigurační úlohy, aby běhové instance procesu splňovaly požadavky všech pracovníků dané organizace. Seznam a popis voleb viz "Volby konfigurace".

Vazby

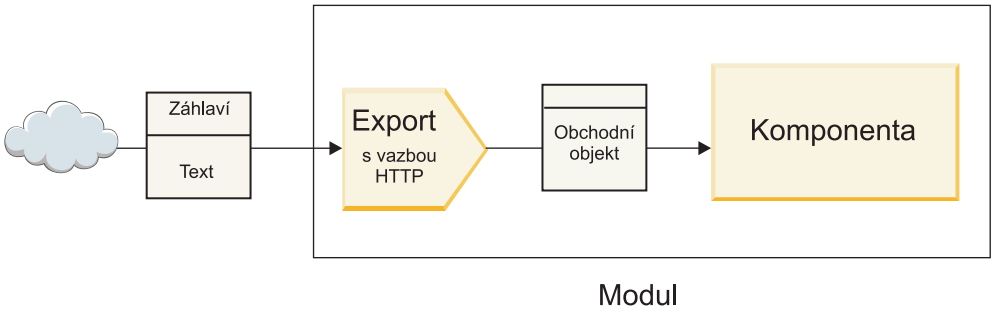
Jádrem architektury SOA (Service-Oriented Architecture) je koncepce *služby*, jednotky funkčnosti zajišťované interakcí výpočetních zařízení. *Export* definuje externí rozhraní (nebo přístupový bod) modulu, aby mohly komponenty SCA (Service Component Architecture) v rámci modulu poskytovat služby externím klientům. *Import* definuje rozhraní pro služby mimo modul, aby mohly být tyto služby volány z modulu. *Vazby* specifické pro určitý protokol se používají s importy a exporty k určení prostředků přenosu dat do modulu nebo z modulu.

Exporty

Externí klienti mohou vyvolávat komponenty SCA v integračním modulu pomocí celé řady protokolů (např. HTTP, JMS, MQ a RMI/IIOP) a s daty v nejrůznějších formátech (např. XML, CSV, COBOL a JavaBeans). Exporty jsou komponenty, které přijímají tyto požadavky od externích zdrojů a následně vyvolávají komponenty produktu IBM Business Process Manager pomocí programovacího modelu SCA.

Například na následujícím obrázku obdrží export požadavek z klientské aplikace prostřednictvím protokolu HTTP. Data jsou transformována do obchodního objektu, ve formátu používaném komponentou SCA. Tato komponenta je

potom vyvolána s pomocí tohoto datového objektu.

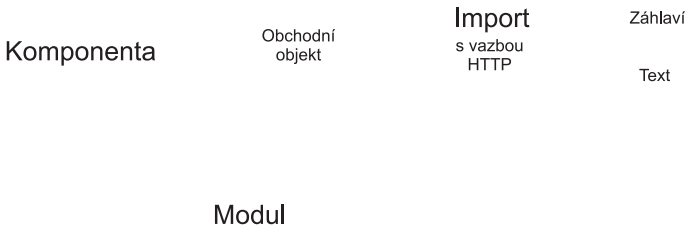


Obrázek 46. Export s vazbou HTTP

Importy

Komponenta SCA může chtít vyvolat externí službu jiného typu než SCA, která očekává data v jiném formátu. Import používá komponenta SCA k vyvolání externí služby s pomocí programovacího modelu SCA. Poté import vyvolá cílovou službu způsobem, který tato služba očekává.

Například na následujícím obrázku je odeslán požadavek z komponenty SCA importem externí službě. Obchodní objekt, což je formát používaný komponentou SCA, je transformován na formát očekávaný danou službou a ta je vyvolána.



Obrázek 47. Import s vazbou HTTP

Seznam vazeb

Pomocí produktu Integration Designer můžete generovat vazbu pro import nebo export a zkonfigurovat vazbu. Dostupné typy vazeb jsou popsány v následujícím seznamu.

- SCA

  Vazba SCA, která je výchozí, umožňuje vaší službě komunikovat se službami v ostatních modulech SCA. Import s vazbou SCA použijete pro přístup ke službě v jiném modulu SCA. Export s vazbou SCA použijete k nabídce služby jiným modulům SCA.
- Webová služba

  Vazba webové služby umožňuje přístup k externí službě pomocí interoperabilních zpráv SOAP a kvalit služby. Vazby webové služby můžete použít také pro zahrnutí příloh do zprávy SOAP.

  Vazba webové služby může používat transportní protokol SOAP/HTTP (SOAP přes HTTP) nebo SOAP/JMS (SOAP přes JMS). Nehledě na přenos (HTTP nebo JMS) použitý k předání zpráv SOAP, ošetřují vazby webové služby interakce požadavků/odezev vždy synchronně.
- HTTP

  Vazba HTTP umožňuje přístup k externí službě pomocí protokolu HTTP, kde jsou používány jiné zprávy než SOAP nebo je nutný přímý přístup HTTP. Tato vazba se používá, když pracujete s webovými službami, které jsou založené na modelu HTTP (tj. službami, které používají známé operace rozhraní HTTP, jako GET, PUT, DELETE apod.).
- Objekty Enterprise JavaBean (EJB)

Vázání EJB umožňují komponentám SCA interakci se službami, které poskytuje obchodní logika v jazyce Java EE spuštěná na serveru Java EE.

- EIS

  Vazba EIS (podnikového informačního systému), je-li použita s adaptérem prostředku JCA, umožňuje přístup ke službám v podnikovém informačním systému nebo tomuto systému povolí vaše služby.

- Vazby JMS

  Vazby Java Message Service (JMS), generické služby JMS a WebSphere MQ JMS (MQ JMS) jsou používány pro interakci se systémy zpráv, kde je použití asynchronní komunikace prostřednictvím front zpráv důležité pro spolehlivost.

  Export s jednou z vazeb JMS sleduje frontu, čeká na přijetí zprávy a asynchronně odesílá případnou odezvu do fronty odpovědí. Import s jednou z vazeb JMS sestaví a odešle zprávu do fronty JMS, sleduje frontu a čeká na přijetí případné odpovědi.

  – JMS

    Vazba JMS umožňuje přístup k poskytovateli JMS vestavěnému do platformy WebSphere.

  – Generická platforma JMS

    Generická vazba služby JMS umožňuje přístup k systému zasílání zpráv od jiného dodavatele než IBM.

  – MQ JMS

    Vazba MQ JMS umožňuje přístup k podmnožině JMS systému zasílání zpráv WebSphere MQ. Tuto vazbu byste použili, když vaší aplikaci stačí podmnožina funkcí JMS.

- MQ

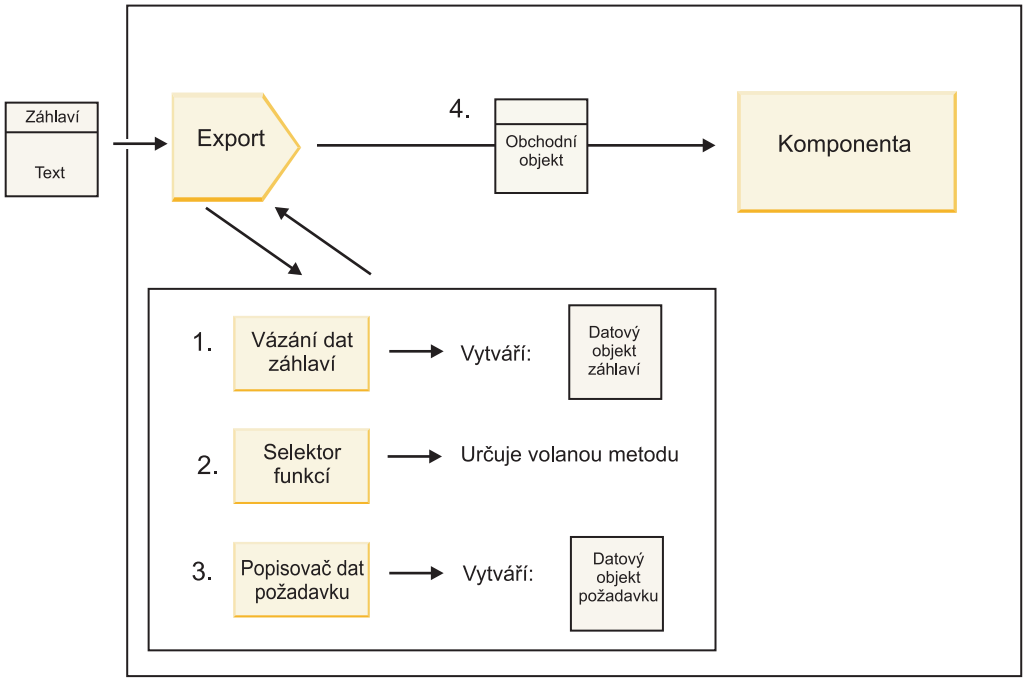
  Vazba WebSphere MQ umožňuje komunikaci s nativními aplikacemi MQ, začleňuje je do rámce architektury SOA (Service-Oriented Architecture) a poskytuje přístup k informacím záhlaví specifickým pro MQ. Tuto vazbu byste použili, když potřebujete nativní funkce MQ.

Přehled vazeb exportu a importu

Export vám umožní zpřístupnit služby v integračním modulu externím klientům a import umožní vašim komponentám SCA v integračním modulu volat externí služby. Vazba přidružená k exportu nebo importu určuje vztah mezi zprávami protokolu a obchodními objekty. Také určuje způsob výběru operací a poruch.

Tok informací exportem

Export obdrží požadavek, který je určen komponentě, s níž je tento export spojen, prostřednictvím specifického přenosu určeného přidruženou vazbou (například HTTP).



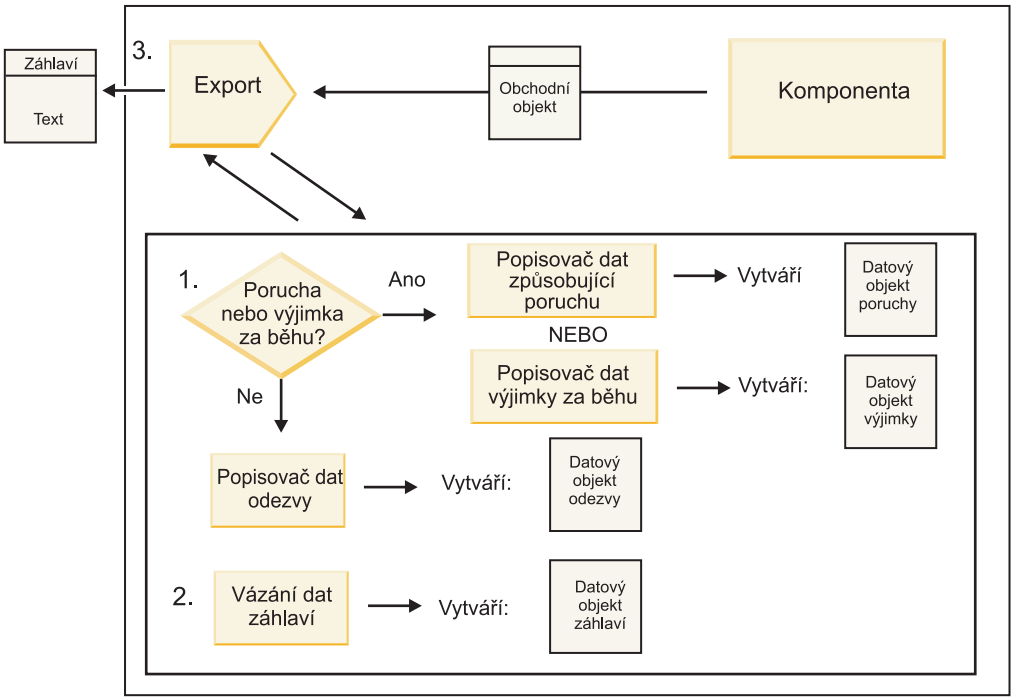
Obrázek 48. Tok požadavku exportem do komponenty

Když export obdrží požadavek, dojde k následující posloupnosti událostí:

1. Pouze v případě vazeb WebSphere MQ provede vázání dat záhlaví transformaci záhlaví protokolu na datový objekt záhlaví.
2. Selektor funkcí určuje název nativní metody ze zprávy protokolu. Název nativní metody je konfigurací exportu mapován na název operace na rozhraní exportu.
3. Popisovač dat požadavku nebo vázání dat na metodě transformuje požadavek na obchodní objekt požadavku.
4. Export vyvolá metodu komponenty s obchodním objektem požadavku.
   - Vazba exportu HTTP, vazba exportu webové služby a vazba exportu EJB vyvolávají komponentu SCA synchronně.
   - Vazby exportu JMS, generické služby JMS, MQ JMS a WebSphere MQ vyvolávají komponentu SCA asynchronně.

Všimněte si, že export může šířit záhlaví a uživatelské vlastnosti, které obdrží prostřednictvím protokolu, pokud je povoleno šíření kontextu. Komponenty, které jsou spojené s exportem, k mohou přistupovat k těmto záhlavím a uživatelským vlastnostem. Další informace viz téma “Šíření” v Informačním centru pro WebSphere Integration Developer.

Pokud se jedná o obousměrnou operaci, vrátí komponenta odezvu.



Obrázek 49. Tok odpovědi zpět exportem

Dochází k této posloupnosti kroků:

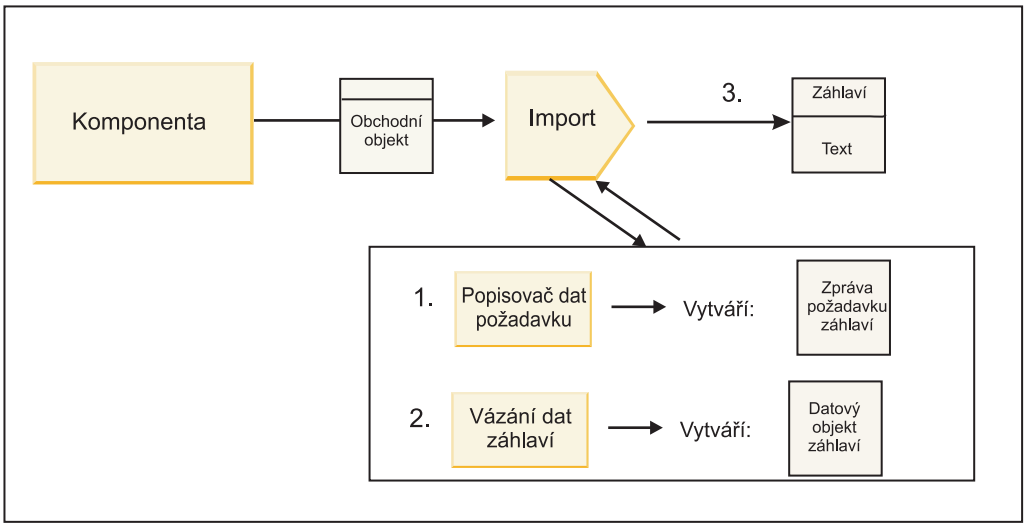
1. Pokud vazba exportu obdrží normální zprávu odpovědi, popisovač dat odpovědi nebo vázání dat na dané metodě transformují tento obchodní objekt na odezvu.

   Pokud je odezvou porucha, transformuje ji popisovač dat způsobující poruchu nebo vázání dat na metodě na odezvu poruchy.

   Pouze pro vazby exportu HTTP, pokud je odezvou výjimka za běhu, je volán popisovač dat výjimky za běhu, je-li nakonfigurován.
2. Pouze pro vazby WebSphere MQ provede vázání dat záhlaví transformaci datových objektů záhlaví na záhlaví protokolu.
3. Export odešle odezvu služby pomocí přenosu.

Tok informací importem

Komponenty odesílají požadavky službám mimo modul pomocí importu. Požadavek je odeslán prostřednictvím specifického přenosu určeného přidruženou vazbou.



Obrázek 50. Tok z komponenty importem do služby

Komponenta vyvolává import pomocí obchodního objektu požadavku.

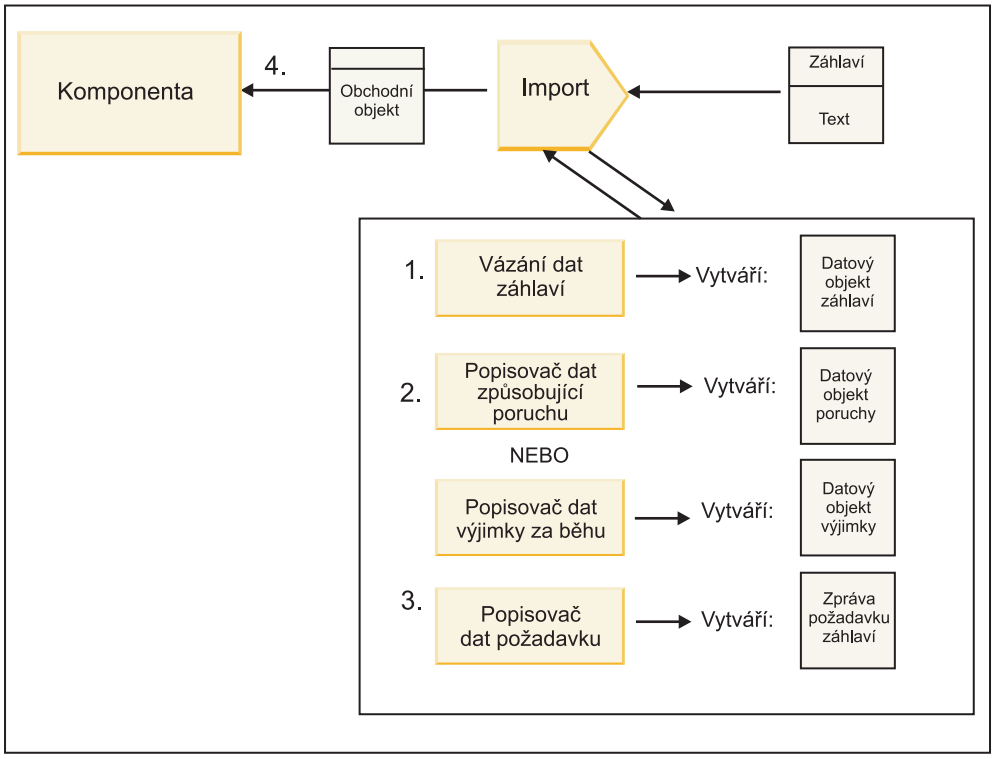
Poznámka:

- Vazbu importu HTTP, vazbu importu webové služby a vazbu importu EJB by měla volající komponenta vyvolávat synchronně.
- Vazby importu JMS, generické služby JMS, MQ JMS a WebSphere MQ mají být vyvolávány asynchronně.

Když komponenta vyvolá import, dojde k následující posloupnosti událostí:

1. Popisovač dat požadavku nebo vázání dat na metodě transformují obchodní objekt požadavku na zprávu požadavku protokolu.
2. Pouze pro vazby WebSphere MQ provede vázání dat záhlaví na metodě nastavení obchodního objektu záhlaví v záhlaví protokolu.
3. Import vyvolává službu pomocí požadavku na službu prostřednictvím přenosu.

Pokud se jedná o obousměrnou operaci, vrátí služba odezvu a dochází k této posloupnosti kroků:



Obrázek 51. Tok odpovědi zpět importem

1. Pouze v případě vazeb WebSphere MQ provede vázání dat záhlaví transformaci záhlaví protokolu na datový objekt záhlaví.
2. Určí se, zda je daná odezva porucha.
   - Pokud je odezvou porucha, zkontroluje ji selektor poruch, aby zjistil, na kterou poruchu WSDL se mapuje. Popisovač dat způsobující poruchu na metodě potom transformuje poruchu na odezvu poruchy.
   - Pokud je odezvou výjimka za běhu, je volán popisovač dat výjimky za běhu, je-li nakonfigurován.
3. Popisovač dat odpovědi nebo vazba na metodě transformuje odezvu na obchodní objekt odpovědi.
4. Import vrací obchodní objekt odpovědi komponentě.

Konfigurace vazby exportu a importu

Jedním z klíčových aspektů vazeb exportu a importu je transformace formátu dat, která označuje, jak jsou data mapována (deserializována) z nativního formátu spojů na obchodní objekt nebo jak jsou mapována (serializována) z obchodního objektu do nativního formátu spojů. V případě vazeb přidružených k exportům můžete určit také selektor funkcí určující, která funkce má být nad daty provedena. V případě vazeb přidružených k exportům nebo importům můžete stanovit, jak pracovat s poruchami, k nimž dojde při zpracování.

Navíc určíte informace týkající se přenosu pro vazby. Například u vazby HTTP určíte adresu URL koncového bodu. Pro vazbu HTTP jsou informace týkající se přenosu popsány v tématech “Generování vazby HTTP pro import” a “Generování vazby HTTP pro export”. V Informačním centru najdete také informace o ostatních vazbách.

Transformace formátu dat v importech a exportech

Když je v produktu IBM Integration Designer nakonfigurována vazba exportu nebo importu, jednou z vlastností konfigurace, kterou určujete, je datový formát používaný vazbou.

- U vazeb exportu, v jejichž rámci aplikace klienta odesílá požadavky komponentě SCA a přijímá od ní odpovědi, určíte formát nativních dat. V závislosti na tomto formátu systém vybere odpovídající popisovač dat nebo vázání dat pro transformaci nativních dat na obchodní objekt (který používá komponenta SCA) a naopak pro transformaci obchodního objektu na nativní data (což je odezva aplikaci klienta).

- U vazeb importu, v jejichž rámci komponenta SCA odesílá požadavky službě mimo modul a přijímá od ní odpovědi, určíte datový formát nativních dat. V závislosti na tomto formátu systém vybere odpovídající popisovač dat nebo vázání dat pro transformaci obchodního objektu na nativní data a naopak.

Produkt IBM Business Process Manager nabízí sadu předdefinovaných datových formátů a odpovídajících popisovačů dat nebo vázáních dat, která tyto formáty podporují. Můžete si vytvořit i vlastní popisovače dat a zaregistrovat datový formát pro tyto popisovače. Další informace viz téma "Vývoj popisovačů dat" v Informačním centru produktu IBM Integration Designer.

- *Popisovače dat* jsou nezávislé na protokolu a transformují data z jednoho formátu do jiného. V produktu IBM Business Process Manager popisovače dat zpravidla transformují nativní data (např. XML, CSV a COBOL) na obchodní objekt a naopak. Protože jsou nezávislé na protokolu, můžete týž popisovač dat opakovaně použít s celou řadou vazeb exportu a importu. Například můžete použít stejný popisovač dat XML s vazbou exportu nebo importu HTTP či s vazbou exportu nebo importu JMS.
- *Vázání dat* také transformují nativní data na obchodní objekt (a naopak), ale jsou specifické pro konkrétní protokol. Například vázání dat HTTP je možné použít pouze s vazbou exportu nebo importu HTTP. Na rozdíl od popisovačů dat nelze vázání dat HTTP znovu použít s vazbou exportu nebo importu MQ.

  Poznámka: Tři vázání dat HTTP (HTTPStreamDataBindingSOAP, HTTPStreamDataBindingXML a HTTPServiceGatewayDataBinding) jsou od produktu IBM Business Process Manager verze 7.0 zamítnuté. Kdykoli to bude možné, používejte popisovače dat.

Jak bylo uvedeno již dříve, v případě potřeby si můžete vytvořit vlastní popisovače dat. Můžete si vytvořit i vlastní vázání dat, ale doporučuje se vytvářet spíše vlastní popisovače dat, protože ty lze použít pro různé vazby.

Popisovače dat:

Popisovače dat jsou nakonfigurované oproti vazbám exportu a importu za účelem transformace dat z jednoho formátu do druhého nezávisle na protokolu. V rámci produktu je poskytováno několik popisovačů dat, v případě potřeby si však můžete vytvořit vlastní popisovač dat. Popisovač dat můžete přidružit k vazbě exportu nebo importu na jedné ze dvou úrovní: můžete jej přidružit ke všem operacím v rozhraní exportu neb importu, nebo jej můžete přidružit ke specifické operaci pro požadavek nebo odezvu.

Předdefinované popisovače dat

Popisovač dat, který chcete použít, určete v produktu IBM Integration Designer.

Popisovače dat, které máte předdefinovány k použití, jsou uvedeny v následující tabulce, která zároveň popisuje, jak jednotlivé popisovače dat transformují příchozí a odchozí data.

Poznámka: Pokud není uvedeno jinak, je možné tyto popisovače dat používat s vazbami JMS, Generické služby JMS, MQ JMS, WebSphere MQ a HTTP.
Podrobnější informace viz téma "Popisovače dat" v Informačním centru produktu Integration Designer.

Tabulka 34. Předdefinované popisovače dat

Popisovač dat	Nativní data na obchodní objekt	Obchodní objekt na nativní data.
ATOM	Analyzuje kanály ATOM do obchodního objektu kanálu ATOM.	Serializuje obchodní objekt kanálu ATOM do kanálů ATOM.
S oddělovačem	Analyzuje data s oddělovačem do obchodního objektu.	Serializuje obchodní objekt do dat s oddělovačem, včetně CSV.
Pevná šířka	Analyzuje data s pevnou šířkou do obchodního objektu.	Serializuje obchodní objekt na data s pevnou šířkou.

Tabulka 34. Předdefinované popisovače dat (pokračování)

Popisovač dat	Nativní data na obchodní objekt	Obchodní objekt na nativní data.
Zpracováno pomocí WTX	Deleguje transformaci formátu dat produktu WebSphere Transformation Extender (WTX). Název mapy WTX je odvozen popisovačem dat.	Deleguje transformaci formátu dat produktu WebSphere Transformation Extender (WTX). Název mapy WTX je odvozen popisovačem dat.
Zpracováno původcem volání WTX	Deleguje transformaci formátu dat produktu WebSphere Transformation Extender (WTX). Název mapy WTX dodává uživatel.	Deleguje transformaci formátu dat produktu WebSphere Transformation Extender (WTX). Název mapy WTX dodává uživatel.
JAXB	Serializuje objekty Java bean na obchodní objekt s použitím pravidel mapování definovaných specifikací JAXB (Java Architecture for XML Binding).	Deserializuje obchodní objekt na objekty Java bean s použitím pravidel mapování definovaných specifikací JAXB.
JAXWS **Poznámka:** Popisovač dat JAXWS lze použít pouze s vázáním EJB	Používán vázáním EJB k transformaci objektu odpovědi Java nebo objektu výjimky Java na obchodní objekt odpovědi s použitím pravidel mapování definovaných specifikací JAX-WS (Java API for XML Web Services).	Používán vázáním EJB k transformaci obchodního objektu na parametry odchozí metody Java s použitím pravidel mapování definovaných specifikací JAX-WS.
JSON	Analyzuje data JSON do obchodního objektu.	Serializuje obchodní objekt do dat JSON.
Nativní tělo	Analyzuje nativní bajty, text, mapu, proud nebo objekt do jednoho z pěti základních obchodních objektů (text, bajty, mapa, proud nebo objekt).	Transformuje pět základních obchodních objektů na bajt, text, mapu, proud nebo objekt.
SOAP	Analyzuje zprávu SOAP (a záhlaví) do obchodního objektu.	Serializuje obchodní objekt do zprávy SOAP.
Kód jazyka XML	Analyzuje data XML do obchodního objektu.	Serializuje obchodní objekt na data XML.
UTF8XMLDataHandler	Analyzuje data XML s kódováním UTF-8 do obchodního objektu.	Serializuje obchodní objekt při odesílání zprávy na data XML s kódováním UTF-8.

Vytvoření popisovače dat

Podrobné informace o vytvoření popisovače dat naleznete v tématu “Vývoj popisovačů dat” v Informačním centru produktu Integration Designer.

Vázání dat:

Vázání dat jsou nakonfigurována oproti vazbám exportu a importu, aby transformovaly data z jednoho formátu do jiného. Vázání dat jsou specifická pro určitý protokol. V rámci produktu je poskytováno několik vázání dat, v případě potřeby si však můžete vytvořit vlastní vázání dat. Vázání dat můžete přidružit k vazbě exportu nebo importu na jedné ze dvou úrovní - můžete ji přidružit ke všem operacím v rozhraní exportu neb importu, nebo ji můžete přidružit ke specifické operaci pro požadavek nebo odezvu.

Pomocí produktu IBM Integration Designer můžete určit, které vázání dat se má použít, nebo vytvořit své vlastní vázání dat. Diskusi o tvorbě vázání dat naleznete v sekci “Přehled vazeb JMS, MQ JMS a generické služby JMS” v Informačním centru produktu IBM Integration Designer.

Vazby JMS

Následující tabulka uvádí vázání dat, s nimiž lze použít:

- Vazby JMS.
- Generické vazby JMS.
- Vazby WebSphere MQ JMS.

Tabulka obsahuje také popis úloh, která provádějí vázání dat.

Tabulka 35. Předdefinovaná vázání dat pro vazby JMS

Vázání dat	Nativní data na obchodní objekt	Obchodní objekt na nativní data.
Serializovaný objekt Java	Transformuje serializovaný objekt Java na obchodní objekt (který je mapován jako typ vstupu nebo výstupu ve WSDL).	Serializuje obchodní objekt na serializovaný objekt Java ve zprávě objektu JMS.
Zalomené bajty.	Extrahuje bajty z příchozí bajtové zprávy JMS a zabalí je do obchodního objektu JMSBytesBody.	Extrahuje bajty z obchodního objektu JMSBytesBody a zabalí je do odchozí bajtové zprávy JMS.
Zabalená položka mapy	Extrahuje informace o názvu, hodnotě a typu pro všechny položky v příchozí zprávě mapy JMS a vytváří seznam obchodních objektů MapEntry. Tento seznam pak zabalí do obchodního objektu JMSMapBody.	Extrahuje informace o názvu, hodnotě a typu ze seznamu položek MapEntry v obchodním objektu JMSMapBody a vytváří odpovídající položky v odchozí zprávě mapy JMS.
Zabalený objekt	Extrahuje objekt z příchozí zprávy objektu JMS a zabalí jej do obchodního objektu JMSObjectBody.	Extrahuje objekt z obchodního objektu JMSObjectBody a zabalí jej do odchozí zprávy objektu JMS.
Zalomený text	Extrahuje text z příchozí textové zprávy JMS a zabalí jej do obchodního objektu JMSTextBody.	Extrahuje text z obchodního objektu JMSTextBody a zabalí jej do odchozí textové zprávy JMS.

Vazby WebSphere MQ

Následující tabulka obsahuje vázání dat, která lze použít s produktem WebSphere MQ, a popisuje úlohy, které tato vázání dat provádějí.

Tabulka 36. Předdefinovaná vázání dat pro vazby WebSphere MQ

Vázání dat	Nativní data na obchodní objekt	Obchodní objekt na nativní data.
Serializovaný objekt Java	Transformuje serializovaný objekt Java z příchozí zprávy na obchodní objekt (který je mapován jako typ vstupu nebo výstupu ve WSDL).	Transformuje obchodní objekt na serializovaný objekt Java v odchozí zprávě.
Zalomené bajty.	Extrahuje bajty z nestrukturované bajtové zprávy MQ a zabalí je do obchodního objektu JMSBytesBody.	Extrahuje bajty z obchodního objektu JMSBytesBody a zabalí tyto bajty do odchozí nestrukturované bajtové zprávy MQ.
Zalomený text	Extrahuje text z nestrukturované textové zprávy MQ a zabalí jej do obchodního objektu JMSTextBody.	Extrahuje text z obchodního objektu JMSTextBody a zabalí jej do nestrukturované textové zprávy MQ.
Zabalená položka proudu	Extrahuje informace o názvu a typu pro všechny položky v příchozí zprávě proudu JMS a vytváří seznam obchodních objektů StreamEntry. Tento seznam pak zabalí do obchodního objektu JMSStreamBody.	Extrahuje informace o názvu a typu ze seznamu položek StreamEntry v obchodním objektu JMSStreamBody a vytváří odpovídající položky v odchozí zprávě JMSStreamMessage.

Kromě vázání dat uvedených v seznamu Tabulka 14 na stránce 57 používá WebSphere MQ také vázání dat záhlaví. Podrobnosti naleznete v Informačním centru produktu IBM Integration Designer.

Vazby HTTP

Následující tabulka obsahuje vazby dat, které lze použít s HTTP, a popisuje úlohy prováděné vázáními dat.

Tabulka 37. Předdefinovaná vázání dat pro vazby HTTP

Vázání dat	Nativní data na obchodní objekt	Obchodní objekt na nativní data.
Zalomené bajty.	Extrahuje bajty z těla příchozí zprávy HTTP a zabalí je do obchodního objektu HTTPBytes.	Extrahuje bajty z obchodního objektu HTTPBytes a přidá je do těla odchozí zprávy HTTP.
Zalomený text	Extrahuje text z těla příchozí zprávy HTTP a zabalí jej do obchodního objektu HTTPText.	Extrahuje text z obchodního objektu HTTPText a přidá jej do těla odchozí zprávy HTTP.

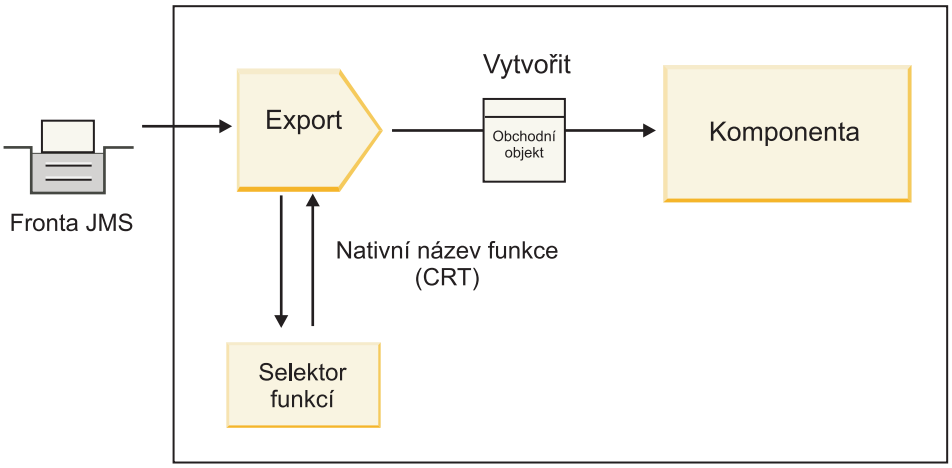
Selektory funkcí ve vazbě exportu

Selektor funkcí se používá k určení toho, která operace se má provést s daty pro zprávu požadavku. Selektory funkcí jsou nakonfigurovány jako součást vazby exportu.

Zvažte export SCA, který vystavuje rozhraní. Toto rozhraní obsahuje dvě operace: Vytvořit a Aktualizovat. Export má vazbu JMS, která čte z fronty.

Když dorazí zpráva do fronty, jsou exportu předána přidružená data, ale která operace z rozhraní exportu má být vyvolána nad spojenou komponentou? Tuto operaci určuje selektor funkcí a konfigurace vazby exportu..

Selektor funkcí vrací název nativní funkce (název funkce v klientském systému, který zprávu odeslal). Název nativní funkce je pak namapován na název operace nebo funkce v rozhraní přidruženém k exportu. Například na následujícím obrázku vrací selektor funkcí název nativní funkce (CRT) z příchozí zprávy, tento název nativní funkce je namapován na operaci Vytvořit a obchodní objekt je odeslán komponentě SCA s operací Vytvořit.



Obrázek 52. Selektor funkcí

Pokud má rozhraní pouze jednu operaci, není třeba určovat selektor funkcí.

K dispozici je několik předem seskupených selektorů funkcí, které jsou uvedeny v následujících sekcích.

Vazby JMS

Následující tabulka uvádí selektory funkcí, s nimiž lze použít:

- Vazby JMS.
- Generické vazby JMS.
- Vazby WebSphere MQ JMS.

Tabulka 38. Předdefinované selektory funkcí pro vazby JMS

Selektor funkcí	Popis
Selektor funkcí JMS pro jednoduchá vázání dat JMS	Používá vlastnost JMSType zprávy pro výběr názvu operace.
Selektor funkcí vlastnosti záhlaví JMS	Vrací hodnotu vlastnosti řetězce JMS TargetFunctionName ze záhlaví.
Selektor funkcí brány služeb JMS	Určuje, zda je požadavek jednosměrná nebo obousměrná operace prozkoumáním vlastnosti JMSReplyTo nastavené klientem.

Vazby WebSphere MQ

Následující tabulka uvádí selektory funkcí, které lze použít s vazbami WebSphere MQ.

Tabulka 39. Předdefinované selektory funkcí pro vazby WebSphere MQ

Selektor funkcí	Popis
Selektor funkcí MQ handleMessage	Vrací handleMessage jako hodnotu, která je mapována pomocí vazeb metody exportu na název operace na rozhraní.
MQ používá výchozí selektor funkcí JMS	Načte nativní operaci z vlastnosti TargetFunctionName složky záhlaví MQRFH2.
MQ používá formát těla zprávy jako nativní funkci	Vyhledá pole Formát posledního záhlaví a vrací toto pole jako řetězec.
Selektor funkcí typu MQ	Vytvoří ve vaší vazbě exportu metodu načtením adresy URL obsahující vlastnosti Msd, Set, Type a Format, které se nacházejí v záhlaví MQRFH2.
Selektor funkcí brány služeb MQ	Používá vlastnost MsgType v záhlaví MQMD k určení názvu operace.

Vazby HTTP

Následující tabulka uvádí selektory funkcí, které lze použít s vazbami HTTP.

Tabulka 40. Předdefinované selektory funkcí pro vazby HTTP

Selektor funkcí	Popis
Selektor funkcí HTTP na základě záhlaví TargetFunctionName	Používá vlastnost záhlaví HTTP TargetFunctionName z klienta k určení operace, která se má vyvolat za běhu z exportu.
Selektor funkcí HTTP založený na adrese URL a metodě HTTP	Používá relativní cestu z adresy URL připojené k metodě HTTP z klienta k určení nativní operace definované na exportu.
Selektor funkcí brány služeb HTTP založený na adrese URL s názvem operace	Určuje metodu, která se má vyvolat, na základě adresy URL, pokud bylo k adrese URL požadavku připojeno "operationMode = oneWay".

Poznámka: Můžete si vytvořit i vlastní Selektor funkcí pomocí nástroje IBM Integration Designer. Informace o vytvoření selektoru funkcí nabízí Informační centrum pro IBM Integration Designer. Například popis vytvoření selektoru funkcí pro vazby WebSphere MQ naleznete v tématu “Přehled selektorů funkcí pro MQ”.

Zpracování poruch

Vazby importu a exportu můžete nakonfigurovat tak, aby zpracovávaly poruchy (například obchodní výjimky), k nimž dochází při zpracování určením popisovačů dat způsobujících poruchu. Popisovač dat způsobujících poruchu můžete nastavit na třech úrovních - popisovač dat způsobujících poruchu můžete přidružit k poruše, k operaci nebo pro všechny operace s vazbou.

Popisovač dat způsobující poruchu zpracovává data způsobující poruchu a transformuje je do správného formátu, který má být odeslán vazbou exportu nebo importu.

- V případě vazby exportu popisovač dat způsobující poruchu transformuje obchodní objekt výjimky zaslaný z komponenty na zprávu odpovědi, kterou může použít aplikace klienta.
- V případě vazby importu popisovač dat způsobující poruchu transformuje data způsobující poruchu nebo zprávu odpovědi zaslanou ze služby na obchodní objekt výjimky, který může použít komponenta SCA.

V případě vazeb importu volá tato vazba selektor poruch, který určí, zda je zpráva odpovědi normální odpověď, obchodní porucha nebo výjimka za běhu.

Můžete určit popisovač dat způsobující poruchu pro konkrétní poruchu, pro nějakou operaci a pro všechny operace s vazbou.

- Pokud je popisovač dat způsobující poruchu nastaven na všech třech úrovních, je volán popisovač dat přidružený ke konkrétní poruše.
- Pokud jsou popisovače dat způsobujících poruchu nastaveny na úrovni operace a vazby, je volán popisovač dat přidružený k operaci.

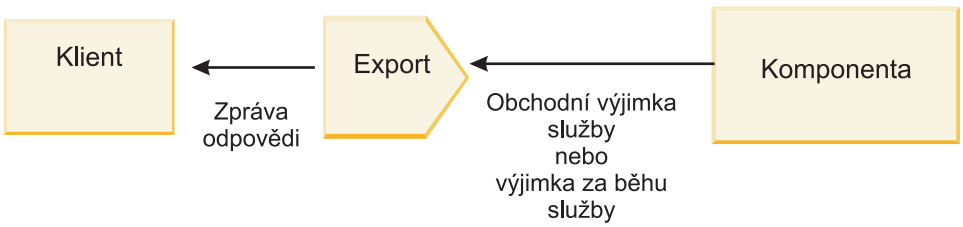
V produktu IBM Integration Designer se k určení zpracování poruch používají dva editory. Editor rozhraní se používá k určení, zda na nějaké operaci dojde k poruše. Po vygenerování vazby pomocí tohoto rozhraní vás nechá tento v zobrazení vlastností nakonfigurovat, jak bude porucha ošetřena. Další informace viz téma “Selektory poruch” v Informačním centru produktu IBM Integration Designer.

Jak jsou ošetřovány poruchy ve vazbách exportu:

Když dojde k poruše při zpracování požadavku od aplikace klienta, může vazba exportu klientovi vrátit informace o poruše. Nakonfigurujete vazbu exportu, aby určovala, jak má být porucha zpracována a vracena klientovi.

Vazbu exportu nakonfigurujete pomocí produktu IBM Integration Designer.

Při zpracování požadavků vyvolá klient export s požadavkem a tento export vyvolá komponentu SCA. Při zpracování požadavku může komponenta SCA buď vrátit obchodní odpověď, nebo může vrátit obchodní výjimku služby nebo výjimku služby za běhu. Když k tomu dojde, transformuje vazba exportu výjimku na chybovou zprávu a odešle ji klientovi, jak ukazuje obrázek níže a popisují následující sekce.



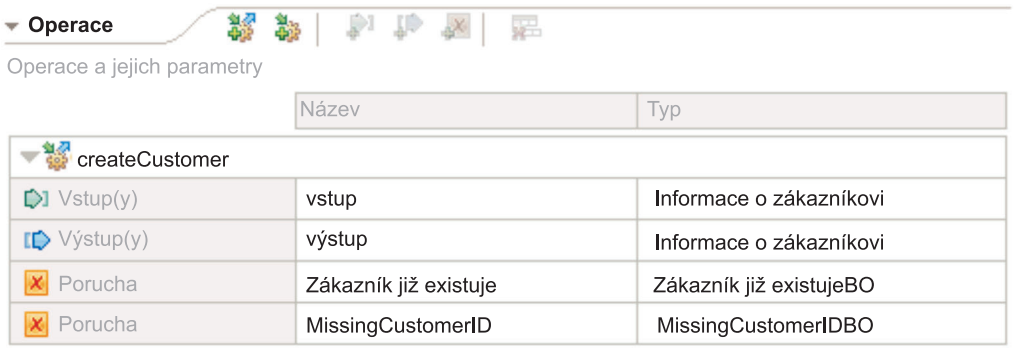
Obrázek 53. Jak jsou informace o poruchách odesílány z komponenty prostřednictvím vazby exportu klientovi

Můžete si vytvořit vlastní popisovač dat nebo vázání dat pro zpracování poruch.

Obchodní poruchy

Obchodní poruchy jsou obchodní chyby nebo výjimky, k nimž dochází při zpracování.

Zvažte použití následujícího rozhraní, kde je definována operace createCustomer. Tato operace má definovány dvě obchodní poruchy: CustomerAlreadyExists a MissingCustomerId.



Obrázek 54. Rozhraní se dvěma poruchami

Pokud v tomto příkladě klient odešle požadavek na vytvoření zákazníka (této komponentě SCA) a daný zákazník již existuje, vrátí komponenta exportu poruchu CustomerAlreadyExists. Export potřebuje tuto obchodní poruchu předat zpět volajícímu klientovi. K tomu použije popisovač dat způsobující poruchu, který je nastaven na vazbě exportu.

Když vazba exportu přijme obchodní poruchu, dojde k tomuto zpracování:

1. Vazba určí, který popisovač dat způsobující poruchu se má vyvolat, aby danou poruchu ošetřil. Pokud obchodní výjimka služby obsahuje název poruchy, je volán popisovač dat, který je nastaven na dané poruše. Pokud obchodní výjimka služby název poruchy neobsahuje, je tento název odvozen porovnáním typů poruch.
2. Vazba zavolá popisovač dat způsobující poruchu s datovým objektem z obchodní výjimky služby.
3. Popisovač dat způsobující poruchu transformuje datový objekt poruchy na zprávu odpovědi a tu vrátí vazbě exportu.
4. Export vrátí zprávu odpovědi klientovi.

Pokud obchodní výjimka služby obsahuje název poruchy, je volán popisovač dat, který je nastaven na dané poruše. Pokud obchodní výjimka služby název poruchy neobsahuje, je tento název odvozen porovnáním typů poruch.

Výjimky za běhu

Výjimka za běhu je výjimka, k níž dojde v aplikaci SCA při zpracování požadavku, která neodpovídá žádné obchodní poruše. Na rozdíl od obchodních poruch nejsou výjimky za běhu definovány v rozhraní.

V určitých scénářích můžete chtít tyto výjimky za běhu předat aplikaci klienta, aby aplikace mohla podniknout odpovídající akci.

Pokud například klient odešle požadavek (této komponentě SCA) na vytvoření zákazníka a při zpracování tohoto požadavku dojde k chybě autorizace, vrátí komponenta výjimku za běhu. Tato výjimka za běhu musí být předána zpět volajícímu klientovi, aby mohl podniknout příslušnou akci ohledně autorizace. Toho je dosaženo prostřednictvím popisovače dat výjimky za běhu nakonfigurovaného pro vazbu exportu.

Poznámka: Popisovač dat výjimek za běhu můžete nakonfigurovat pouze pro vazby HTTP.

Zpracování výjimky za běhu je podobné zpracování obchodní poruchy. Pokud byl nastaven popisovač dat výjimky za běhu, dojde k tomuto zpracování:

1. Vazba exportu zavolá příslušný popisovač dat s výjimkou služby za běhu.

2. Popisovač dat transformuje datový objekt poruchy na zprávu odpovědi a tu vrátí vazbě exportu.
3. Export vrátí zprávu odpovědi klientovi.

Zpracování poruch a zpracování výjimek za běhu jsou volitelná. Pokud nechcete poruchy nebo výjimky za běhu předávat volajícímu klientovi, nekonfigurujte popisovač dat způsobující poruchu či popisovač dat výjimky za běhu.

Jak jsou ošetřovány poruchy ve vazbách importu:

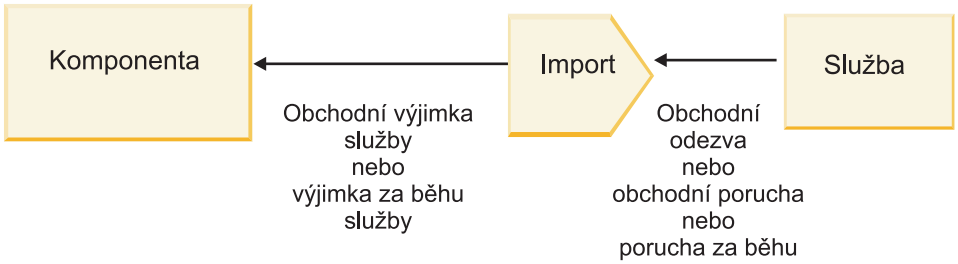
Komponenta používá import k odeslání požadavku službě mimo modul. Když dojde k poruše při zpracování požadavku, vrátí služba poruchu vazbě importu. Vazbu importu můžete nakonfigurovat tak, aby určovala, jak má být porucha zpracována a vrácena komponentě.

Vazbu importu nakonfigurujete pomocí produktu IBM Integration Designer. Můžete určit popisovač dat způsobující poruchu (nebo vázání dat) a také selektor poruch.

Popisovače dat způsobujících poruchu

Služba, která zpracovává požadavek, pošle vazbě importu informace o poruše ve formě výjimky nebo zprávy odpovědi, která obsahuje data způsobující poruchu.

Vazba importu transformuje výjimku služby nebo zprávu odpovědi na obchodní výjimku služby nebo výjimku služby za běhu, jak ukazuje následující obrázek a popisují následující sekce.



Obrázek 55. Jak jsou informace o poruchách odesílány ze služby prostřednictvím importu komponentě

Můžete si vytvořit vlastní popisovač dat nebo vázání dat pro zpracování poruch.

Selektory poruch

Když konfigurujete vazbu importu, můžete určit selektor poruch. Selektor poruch určuje, zda odpověď importu má být skutečnou odpovědí, obchodní výjimkou nebo běhovou poruchou. Také určuje z těla nebo záhlaví odpovědi nativní název poruchy, který konfigurace vazby mapuje na název poruchy v přidruženém rozhraní.

K dispozici jsou dva typy předem seskupených selektorů poruch, které lze používat s importy JMS, MQ JMS, Generické služby JMS, WebSphere MQ a HTTP:

Tabulka 41. Předem seskupené selektory poruch

Typ selektoru poruch	Popis
Podle záhlaví	Určuje, zda je zpráva odpovědi obchodní porucha, výjimka za běhu nebo normální zpráva podle záhlaví v příchozí zprávě odpovědi.
SOAP	Určuje, zda je zpráva SOAP odpovědi normální zpráva, obchodní porucha nebo výjimka za běhu.

Následujíc příklady selektorů poruch podle záhlaví a selektorů poruch SOAP.

- Selektor poruch podle záhlaví

  Pokud chce aplikace označit, že je příchozí zpráva obchodní porucha, musí být v příchozí zprávě dvě záhlaví pro obchodní poruchy, což vypadá takto:

  ```
  Header name = FaultType, Header value = Business
  Header name = FaultName, Header value = <nativní název poruchy definovaný uživatele,>
  ```

  Pokud chce aplikace označit, že je příchozí zpráva odpovědi výjimka za běhu, musí být v příchozí zprávě jedno záhlaví, což vypadá takto:

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

- Selektor poruch SOAP

  Obchodní poruchu lze poslat jako součást zprávy SOAP s následujícím vlastním záhlavím SOAP. V tomto případě je název poruchy "CustomerAlreadyExists".

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

Selektor poruch je nepovinný. Pokud neurčíte žádný selektor poruch, vazba importu nemůže určit typ odpovědi. Proto ji vazba považuje za obchodní odpověď a volá popisovač dat nebo vázání dat odpovědi.

Můžete si vytvořit vlastní selektor poruch. Postup pro vytvoření vlastního selektoru poruch viz “Vývoj vlastního selektoru poruch” v Informačním centru produktu IBM Integration Designer.

Obchodní poruchy

K obchodní poruše může dojít, když nastane chyba při zpracování požadavku. Pokud například odešlete požadavek na vytvoření zákazníka a tento zákazník již existuje, odešle služba vazbě importu obchodní výjimku.

Když vazba obdrží obchodní výjimku, závisí kroky zpracování na tom, zda byl pro tuto vazbu nastaven selektor poruch.

- Pokud nebyl nastaven žádný selektor poruch, volá vazba popisovač dat nebo vázání dat odpovědi.
- Pokud byl nastaven selektor poruch, dojde k tomuto zpracování:
  1. Vazba importu volá selektor poruch, aby určil, zda je odpověď obchodní porucha, obchodní odpověď nebo porucha za běhu.
  2. Pokud je odpověď obchodní porucha, volá vazba importu selektor poruch, aby poskytl nativní název poruchy.
  3. Vazba importu určí poruchu WSDL odpovídající nativnímu názvu poruchy vrácenému selektorem poruch.
  4. Vazba importu určí popisovač dat způsobující poruchu, který je nakonfigurován pro tuto poruchu WSDL.
  5. Vazba importu volá tento popisovač dat způsobující poruchu s daty o poruše.
  6. Popisovač dat způsobující poruchu transformuje data o poruše na datový objekt a ten vrátí vazbě importu.
  7. Vazba importu zkonstruuje objekt obchodní výjimky služby s datovým objektem a názvem poruchy.
  8. Import vrátí objekt obchodní výjimky služby komponentě.

Výjimky za běhu

K výjimce za běhu může dojít, když nastane problém při komunikaci se službou. Zpracování výjimky za běhu je podobné zpracování obchodní výjimky. Pokud byl nastaven selektor poruch, dojde k tomuto zpracování:

1. Vazba importu volá příslušný popisovač dat výjimky za běhu s daty výjimky.
2. Popisovač dat výjimky za běhu transformuje data výjimky na objekt výjimky služby za běhu a ten vrátí vazbě importu.
3. Import vrátí objekt výjimky služby za běhu komponentě.

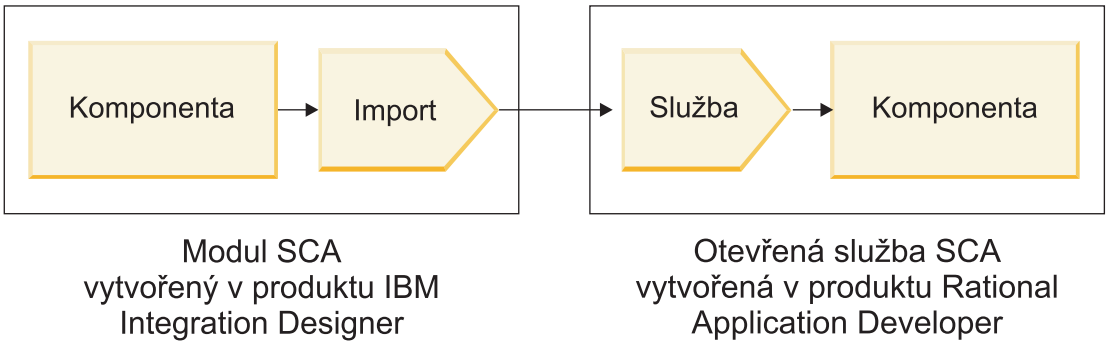
Interoperabilita mezi moduly SCA a službami Open SCA

Balík funkcí IBM WebSphere Application Server V7.0 Feature Pack for Service Component Architecture (SCA) nabízí jednoduchý, avšak účinný programovací model pro tvorbu aplikací na základě specifikací Open SCA. Moduly SCA produktu IBM Business Process Manager používají vazby importu a exportu ke spolupráci se službami Open SCA vyvinutými v prostředí produktu Rational Application Developer s hostitelem serveru WebSphere Application Server Feature Pack for Service Component Architecture.

Aplikace SCA vyvolá aplikaci Open SCA prostřednictvím vazby importu. Aplikace SCA přijme volání od aplikace Open SCA prostřednictvím vazby exportu. Seznam podporovaných vazeb ukazuje “Vyvolání služeb přes spolupracující vazby” na stránce 65.

Vyvolání služeb Open SCA z modulů SCA

Aplikace SCA vyvinuté v produktu IBM Integration Designer mohou vyvolávat aplikace Open SCA vyvinuté v prostředí produktu Rational Application Developer. Tato část obsahuje příklad vyvolání služby Open SCA z modulu SCA pomocí vazby importu SCA.



Obrázek 56. Komponenta v modulu SCA vyvolávající službu Open SCA

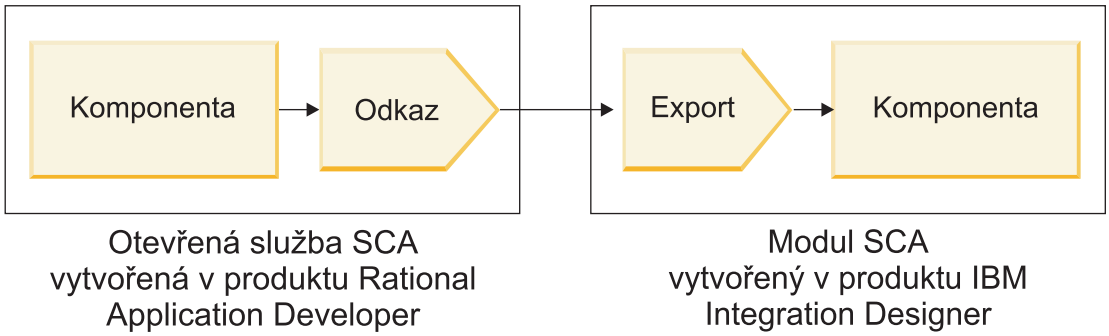
K vyvolání služby Open SCA není nutná žádná speciální konfigurace.

Chcete-li se připojit ke službě Open SCA prostřednictvím vazby importu SCA, musíte zadat název komponenty a název služby Open SCA ve vazbě importu.

1. Chcete-li získat název cílové komponenty a služby ze složeného prvku SCA, postupujte takto:
   a. Zkontrolujte, zda je otevřená karta **Vlastnosti**, klepnutím na volbu **Okno > Zobrazit pohled > Vlastnosti**.
   b. Otevřete editor složených objektů poklepáním na diagram složených objektů, který obsahuje danou komponentu nebo službu. Například diagram složeného objektu pro komponentu nazvanou zákazník (**customer**) je **customer.composite_diagram**.
   c. Klepněte na cílovou komponentu.
   d. Poznamenejte si název cílové komponenty z pole **Název** na kartě **Vlastnosti**.
   e. Klepněte na ikonu služby přidružené k dané komponentě.
   f. Poznamenejte si název služby z pole **Název** na kartě **Vlastnosti**.
2. Chcete-li nakonfigurovat import produktu IBM Business Process Manager pro připojení ke službě Open SCA, postupujte takto:
   a. V produktu IBM Integration Designer přejděte na kartu **Vlastnosti** importu SCA, který chcete připojit ke službě Open SCA.
   b. Do pole **Název modulu** zadejte název komponenty z kroku 1d na stránce 64.
   c. Do pole **Název exportu** zadejte název služby z kroku 1f na stránce 64.
   d. Uložte svou práci stisknutím kláves Ctrl+S.

Vyvolání modulů SCA ze služeb Open SCA

Aplikace Open SCA vyvinuté v prostředí produktu Rational Application Developer mohou vyvolávat aplikace SCA vyvinuté v produktu IBM Integration Designer. Tato část obsahuje příklad vyvolání modulu SCA (prostřednictvím vazby exportu SCA) ze služby Open SCA.



Obrázek 57. Služba Open SCA vyvolávající komponentu v modulu SCA

Chcete-li připojit komponentu SCA prostřednictvím vazby odkazu Open SCA, musíte zadat název modulu a název exportu.

1. Chcete-li získat název cílového modulu a exportu, postupujte takto:
   a. V produktu IBM Integration Designer otevřete daný modul poklepáním na tento modul v editoru sestavení.
   b. Klepněte na export.
   c. V poli **Název** na kartě **Vlastnosti** zadejte název exportu.
2. Nakonfigurujte odkaz Open SCA, který chcete připojit k modulu a exportu produktu IBM Business Process Manager:
   a. V produktu Rational Application Developer otevřete editor složených objektů poklepáním na diagram složených objektů, který obsahuje danou komponentu nebo službu.
   b. Klepněte na ikonu odkazu pro odkaz na komponentu a zobrazí se jeho vlastnosti na kartě **Vlastnosti**.
   c. Klepněte na kartu **Vazba** na levé straně stránky.
   d. Klepněte na volbu **Vazby** a potom na tlačítko **Přidat**.
   e. Vyberte vazbu **SCA**.
   f. Do pole **Identifikátor URI** zadejte název modulu produktu IBM Business Process Manager, následovaný lomítkem (“/”), následovaným názvem exportu (který jste určili v kroku 1c na stránce 65).
   g. Klepněte na tlačítko **OK**.
   h. Uložte svou práci stisknutím kláves Ctrl+S.

Vyvolání služeb přes spolupracující vazby

Následující vazby jsou podporovány pro součinnost se službou Open SCA.

- Vazba SCA

  Když v produktu IBM Business Process Manager modul SCA vyvolává službu Open SCA prostřednictvím vazby importu SCA, jsou podporovány tyto styly vyvolání:

  – Asynchronní (jednosměrný).
  – Synchronní (požadavek/odpověď)

  Rozhraní importu SCA a rozhraní služby Open SCA musí využívat interoperabilitu webových služeb (WS-I) kompatibilní s rozhraním WSDL.

  Všimněte si, že vazba SCA podporuje šíření kontextu zabezpečení a transakcí.

- Vazba webové služby (JAX-WS) s protokolem SOAP1.1/HTTP nebo SOAP1.2/HTTP

  Rozhraní importu SCA a rozhraní služby Open SCA musí využívat interoperabilitu webových služeb (WS-I) kompatibilní s rozhraním WSDL.

  Navíc jsou podporovány tyto kvality služby:

  – Web Services Atomic Transaction.
  – Web Services Security.
- Vázání EJB

  Rozhraní Java se používá k definování interakce mezi modulem SCA a službou Open SCA, když je použito vázání EJB.

  Poznámka: vázání EJB podporuje šíření kontextu zabezpečení a transakcí.
- Vazby JMS

  Rozhraní importu SCA a rozhraní služby Open SCA musí využívat interoperabilitu webových služeb (WS-I) kompatibilní s rozhraním WSDL.

  Jsou podporováni tito poskytovatelé JMS:

  – WebSphere Platform Messaging (Vazba JMS).
  – WebSphere MQ (Vazba MQ JMS).

Poznámka: Obchodní grafy neumožňují interoperabilitu napříč vazbami SCA, a proto nejsou podporovány v rozhraních používaných pro interoperabilitu s balíkem funkcí serveru WebSphere Application Server Feature Pack for Service Component Architecture.

Typy vazeb

Vazby specifické pro určitý protokol se používají s importy a exporty k určení prostředků přenosu dat do modulu nebo z modulu.

Výběr odpovídajících vazeb

Při vytváření aplikace musíte vědět, jak vybrat nejvhodnější vazbu pro potřebu vaší aplikace.

Vazby, které jsou k dispozici v produktu IBM Integration Designer, nabízejí celou řadu voleb. V tomto seznamu můžete určit, který typ vazby může být pro potřeby vaší aplikace nejvhodnější.

Pokud se na daný případ vztahují následující faktory, zvažte použití vazby *SCA (Service Component Architecture)*:

- Veškeré služby jsou obsaženy v modulech, neexistují tedy žádné externí služby.
- Chcete rozdělit funkci do různých modulů SCA, které spolu přímo komunikují.
- Moduly jsou pevně přiřazeny.

Pokud se na daný případ vztahují následující faktory, zvažte použití *vazby webové služby*:

- Je vyžadován přístup k externí službě nebo poskytování služby prostřednictvím sítě Internet.
- Služby jsou volně přiřazeny.
- Je preferována synchronní komunikace; tj. požadavek z jedné služby může čekat na odezvu z jiné.
- Protokol externí služby, ke které přistupujete, nebo služby, kterou chcete poskytovat, je SOAP/HTTP nebo SOAP/JMS.

Pokud se na daný případ vztahují následující faktory, zvažte použití *vazby HTTP*:

- Potřebujete internetový přístup k externí službě nebo poskytovat službu po Internetu a pracujete s jinými webovými službami, jako jsou GET, PUT či DELETE.
- Služby jsou volně přiřazeny.
- Je preferována synchronní komunikace; tj. požadavek z jedné služby může čekat na odezvu z jiné.

Pokud se na daný případ vztahují následující faktory, zvažte použití vazby *EJB (Enterprise JavaBeans)*:

- Vazba je určena pro importovanou službu, která je sama objektem EJB, nebo ke které potřebují mít přístup klienti EJB.
- Importovaná služba je volně přiřazena.
- Nejsou požadovány stavové interakce EJB.
- Je preferována synchronní komunikace; tj. požadavek z jedné služby může čekat na odezvu z jiné.

Pokud se na daný případ vztahují následující faktory, zvažte použití vazby *EIS (Enterprise Information Systems)*:
- Ke službě v systému EIS je nutné přistupovat prostřednictvím adaptéru prostředku.
- Je preferován synchronní přenos dat před asynchronním.

Pokud se na daný případ vztahují následující faktory, zvažte použití vazby *JMS (Java Message Service)*:

Důležité: Existuje několik typů vazeb JMS. Pokud očekáváte výměnu zpráv SOAP prostřednictvím služby JMS, zvažte použití vazby webové služby s protokolem SOAP/JMS. Viz "Vazby webové služby" na stránce 68.
- Je nutné přistupovat k systému zpráv.
- Služby jsou volně přiřazeny.
- Je preferován asynchronní přenos dat před synchronním.

Pokud se na daný případ vztahují následující faktory, zvažte použití vazby *Generické JMS (Java Message Service)*:
- Je nutné přistupovat k systému zpráv jiných dodavatelů než IBM.
- Služby jsou volně přiřazeny.
- Spolehlivost je důležitější než výkon, je tedy preferován asynchronní přenos dat před synchronním.

Pokud se na daný případ vztahují následující faktory, zvažte použití vazby *MQ (Message Queue)*:
- Je nutné přistupovat k systému zpráv produktu WebSphere MQ a používat nativní funkce MQ.
- Služby jsou volně přiřazeny.
- Spolehlivost je důležitější než výkon, je tedy preferován asynchronní přenos dat před synchronním.

Pokud se na daný případ vztahují následující faktory, zvažte použití *vazby MQ JMS*:
- Potřebujete přistupovat k systému zpráv produktu WebSphere MQ, ale lze tak činit v kontextu služby JMS. Podmnožina funkcí JMS je tedy pro vaši aplikaci dostatečná.
- Služby jsou volně přiřazeny.
- Spolehlivost je důležitější než výkon, je tedy preferován asynchronní přenos dat před synchronním.

Vazby SCA

Vazba architektury SCA (Service Component Architecture) umožní službě komunikovat s dalšími službami v jiných modulech. Import s vazbou SCA umožňuje přistupovat ke službě v jiném modulu SCA. Export s vazbou SCA umožňuje nabízet služby jiným modulům.

Pomocí produktu IBM Integration Designer generujete a konfigurujete vazby SCA na importy a exporty v modulech SCA.

Pokud jsou moduly spuštěné na stejném serveru nebo jsou implementované ve stejném klastru, je nejjednodušší a nejrychleji použitelnou vazbou vazba SCA.

Po implementaci modulu, který obsahuje danou vazbu SCA, na serveru můžete v administrativní konzole zobrazit informace o této vazbě, nebo v případě vazby importu změnit její vybrané vlastnosti.

Vazby webové služby

Vazba webové služby je prostředek pro přenos zpráv z komponenty SCA (Service Component Architecture) do webové služby (a obráceně).

Přehled vazeb webové služby:

Vazba importu webové služby umožňuje volání externí webové služby z komponent SCA (Service Component Architecture). Vazba exportu webové služby umožňuje odkrytí komponent SCA klientům v podobě webových služeb.

Pomocí vazby webové služby přistupujete k externím službám za použití interoperabilních zpráv SOAP a QoS (Qualities of Service).

Pomocí produktu Integration Designer generujete a konfigurujete vazby webové služby na importy a exporty v modulech SCA. K dispozici jsou následující typy vazeb webových služeb:

- SOAP1.2/HTTP a SOAP1.1/HTTP:

  Tyto vazby jsou založené na programovacím rozhraní API jazyka Java pro vytváření webových služeb, označovaném jako JAX-WS (Java API for XML Web Services).

  – Pokud vaše webová služba odpovídá specifikaci SOAP 1.2, použijte vazbu SOAP1.2/HTTP.
  – Pokud vaše webová služba odpovídá specifikaci SOAP 1.1, použijte vazbu SOAP1.1/HTTP.

  Důležité: Když implementujete aplikaci s vazbou webové služby (JAX-WS), nesmí být na cílovém serveru vybrána volba **Spouštět komponenty podle potřeby**. Podrobnosti viz “Kontrola konfigurace serveru” na stránce 75.

  Když vyberete jednu z těchto vazeb, můžete odesílat zprávy SOAP a přílohami.

  Vazby webové služby pracují se standardními zprávami SOAP. Použitím vazeb webové služby JAX-WS můžete nicméně upravit způsob analýzy či zápisu zpráv SOAP. Můžete například zpracovat nestandardní prvky zpráv SOAP nebo použít u zprávy SOAP dodatečné zpracování. Když vazbu konfigurujete, určíte vlastní popisovač dat, který toto zpracování zprávy SOAP provede.

  S vazbou webové služby (JAX-WS) můžete použít sady zásad. Sada zásad je kolekce typů zásad, z nichž každá zajišťuje kvalitu služeb (QoS). Například sada zásad WSAddressing poskytuje způsob jednotného adresování webových služeb a zpráv, který je neutrální vůči transportu. Sadu zásad pro danou vazbu vybíráte prostřednictvím produktu Integration Designer.

  Poznámka: Pokud chcete použít sadu zásad SAML (Security Assertion Markup Language), musíte provést dodatečnou konfiguraci, viz “Importování sad zásad SAML” na stránce 73.

- SOAP1.1/HTTP:

  Tuto vazbu použijte, pokud chcete vytvořit webové služby používající zprávu kódovanou pomocí protokolu SOAP, založenou na rozhraní JAX-RPC (Java API for XML-based RPC).

- SOAP1.1/JMS:

  Tuto vazbu použijte k odeslání či přijetí zpráv SOAP pomocí cíle JMS (Java Message Service).

Bez ohledu na to, jaký přenos (HTTP či JMS) je pro doručení zprávy SOAP použit, vazby webové služby vždy zpracovávají interakce požadavek/odezva synchronně. Podproces provádějící vyvolání poskytovatele služeb je blokován, dokud nedojde k obdržení odezvy od poskytovatele. Další informace o tomto stylu vyvolání viz “Synchronní vyvolání”.

Důležité: Následující kombinace vazeb webových služeb nelze použít pro exporty v rámci jednoho modulu. Pokud potřebujete vystavit komponenty pomocí více než jedné z těchto vazeb exportu, musíte každou z nich mít v odděleném modulu a tyto moduly potom musíte s vašimi komponentami propojit pomocí vazby SCA:

- SOAP 1.1/JMS a SOAP 1.1/HTTP pomocí JAX-RPC.
- SOAP 1.1/HTTP pomocí JAX-RPC a SOAP 1.1/HTTP pomocí JAX-WS.
- SOAP 1.1/HTTP pomocí JAX-RPC a SOAP 1.2/HTTP pomocí JAX-WS.

Po implementaci modulu SCA obsahujícího vazbu webové služby na server si můžete pomocí administrativní konzoly nechat zobrazit informace o dané vazbě, nebo můžete změnit vybrané vlastnosti této vazby.

Poznámka: Webové služby umožňují aplikacím spolupracovat, protože používají standardní popisy služeb a standardní formáty vyměňovaných zpráv. Vazby importu a exportu webové služby mohou například spolupracovat se službami implementovanými pomocí WSE (Web Services Enhancements) verze 3.5 a WCF (Windows Communication Foundation) verze 3.5 pro Microsoft .NET. V případě spolupráce s takovými službami dbejte na následující:

- Soubor WSDL (Web Services Description Language) použitý pro přístup k exportu webové služby musí obsahovat pro každou operaci v daném rozhraní neprázdnou hodnotu akce SOAP.
- Klient webové služby musí při odesílání zpráv do exportu webové služby nastavit buď záhlaví SOAPAction, nebo wsa:Action.

Šíření záhlaví SOAP:

Při obsluze zpráv SOAP může být zapotřebí přistupovat k informacím z určitých záhlaví SOAP přijímaných zpráv. Zajistěte, aby byly zprávy se záhlavími SOAP odesílány se specifickými hodnotami nebo umožněte záhlavím SOAP minout určitý modul.

Při konfiguraci vazby webové služby v produktu Integration Designer můžete označit, že chcete záhlaví SOAP šířit.

- Při obdržení požadavků při exportu nebo odezev při importu lze přistupovat k informacím ze záhlaví SOAP, což umožňuje založit logiku modulu na hodnotách daného záhlaví a umožňuje to také úpravu těchto záhlaví.
- Když jsou požadavky odeslány z exportu nebo odezvy z importu, lze do těchto zpráv zahrnout záhlaví SOAP.

Forma a přítomnost šířených záhlaví SOAP může být ovlivněna sadami zásad nakonfigurovanými pro import či export, viz Tabulka 20 na stránce 70.

Chcete-li nakonfigurovat šíření záhlaví SOAP pro import či export, vyberte (v pohledu Vlastnosti produktu Integration Designer) kartu **Šířit záhlaví protokolu** a vyberte vámi požadované volby.

Záhlaví WS-Addressing

Záhlaví WS-Addressing lze šířit prostřednictvím vazby webové služby (JAX-WS).

Při šíření záhlaví WS-Addressing počítejte s následujícími faktory:

- Pokud povolíte šíření záhlaví WS-Addressing, bude toto záhlaví šířeno do modulu za následujících okolností:
  - Když export přijme požadavky.
  - Když import přijme odezvy.
- Záhlaví WS-Addressing se nešíří do odchozích zpráv z produktu IBM Business Process Manager (tzn. záhlaví se nešíří při odeslání požadavků z importu ani při odeslání odezev z exportu).

Záhlaví WS-Security

Záhlaví WS-Security lze šířit prostřednictvím vazby webové služby (JAX-WS) a také prostřednictvím vazby webové služby (JAX-PRC).

Specifikace zabezpečení WS-Security webových služeb popisuje vylepšení systému zpráv SOAP, které poskytuje kvalitní ochranu prostřednictvím integrity a utajení zpráv, stejně jako prostřednictvím ověřování jednotlivých zpráv. Tyto mechanizmy lze použít k umístění širokého spektra modelů zabezpečení a šifrovacích technologií.

Při šíření záhlaví WS-Security počítejte s následujícími faktory:

- Pokud povolíte šíření záhlaví WS-Security, bude toto záhlaví šířeno napříč modulem za následujících okolností:
  - Když export přijme požadavky.
  - Když import odešle požadavky.
  - Když import přijme odezvy.
- Při výchozím nastavení *nebude* záhlaví šířeno v případě odesílání odezev z exportu. Pokud nicméně nastavíte vlastnost prostředí JVM **WSSECURITY.ECHO.ENABLED** na hodnotu **true**, bude záhlaví šířeno i v případě

odesílání odezev z exportu. V tomto případě, pokud nebylo záhlaví WS-Security na cestě požadavku změněno, mohou být záhlaví WS-Security automaticky promítnuty z požadavků do odezev.

- Přesná forma zprávy SOAP odeslané z importu pro požadavek nebo z exportu pro odezvu se nemusí přesně shodovat s původně přijatou zprávou SOAP. Z tohoto důvodu by se mělo předpokládat, že budou všechny digitální podpisy neplatné. Pokud odesílaná zpráva vyžaduje digitální podpis, musí být tento zřízen za použití příslušné sady zásad zabezpečení a záhlaví WS-Security související s digitálním podpisem v přijatých zprávách by měla být z modulu odebrána.

Pokud chcete šířit záhlaví WS-Security, musíte zahrnout schéma zabezpečení WS-Security do modulu aplikace. Procedura zahrnutí schématu viz “Zahrnutí schématu WS-Security do modulu aplikace” na stránce 71.

Způsob šíření záhlaví

Způsob šíření záhlaví závisí na nastavení zásad zabezpečení pro vazbu importu či exportu, viz Tabulka 20 na stránce 70:

Tabulka 42. Způsob předávání záhlaví zabezpečení

	Vazba exportu bez zásad zabezpečení	**Vazba exportu se zásadami zabezpečení**
Vazba importu bez zásad zabezpečení	Záhlaví zabezpečení jsou v rámci modulu předávány tak, jak jsou. Nejsou šifrována. Záhlaví jsou odesílána ve stejné formě, v jaké byla obdržena. Digitální podpis se může stát neplatným.	Záhlaví zabezpečení jsou dešifrována a procházejí modulem s verifikací podpisu a ověřením. Dešifrovaná záhlaví jsou odesílána. Digitální podpis se může stát neplatným.
Vazba importu se zásadami zabezpečení	Záhlaví zabezpečení jsou v rámci modulu předávány tak, jak jsou. Nejsou šifrována. Záhlaví by neměla být šířena do importu. Jinak dojde k chybě z důvodu duplicity.	Záhlaví zabezpečení jsou dešifrována a procházejí modulem s verifikací podpisu a ověřením. Záhlaví by neměla být šířena do importu. Jinak dojde k chybě z důvodu duplicity.

Nakonfigurujte příslušnou sadu zásad pro vazby exportu a importu, protože tak izolujete klienta služby od změn konfigurace či požadavků na kvalitu služby poskytovatele služby. Pokud jsou v modulu standardní záhlaví viditelná, lze je potom použít k ovlivnění zpracování v rámci daného modulu (např. protokolování a trasování). Šíření záhlaví SOAP z přijatých zpráv napříč modulem vede ke snížení výhod izolace modulu.

Standardní záhlaví, jako např. záhlaví WS-Security by neměla být šířena při požadavku na import či odezvě na export v případě, že má import či export přidruženou sadu zásad, která by normálně vedla ke generování těchto záhlaví. Jinak se vyskytne chyba z důvodu duplicity záhlaví. Místo toho by měla být záhlaví výslovně odebrána, nebo by měly být vazby importu či exportu nakonfigurovány tak, aby zabraňovaly šíření záhlaví protokolu.

Přístup k záhlavím SOAP

Při obdržení zprávy obsahující záhlaví SOAP z importu či exportu webové služby jsou tato záhlaví umístěna do sekce se záhlavími objektu SMO (Service Message Object). Přístup k informacím záhlaví viz “Přístup k informacím záhlaví SOAP v objektu SMO”.

Zahrnutí schématu WS-Security do modulu aplikace

Následující procedura nastiňuje kroky pro zahrnutí daného schématu do modulu aplikace:

- Pokud má počítač, na kterém je spuštěný produkt Integration Designer, přístup k Internetu, postupujte takto:
  1. V perspektivě Business Integration vyberte volbu **Závislosti** pro svůj projekt.
  2. Rozbalte položku **Předdefinované prostředky** a vyberte buď položku **Soubory schématu WS-Security 1.0** nebo **Soubory schématu WS-Security 1.1**, a importujte tak dané schéma do svého modulu.

3. Projekt vyčistěte a znovu sestavte.

- Pokud počítač, na kterém je spuštěný produkt Integration Designer nemá přístup k Internetu, můžete schéma stáhnout do druhého počítače, který přístup k Internetu má. Potom může být zkopírováno do počítače, na kterém je spuštěný produkt Integration Designer.
  1. Na počítači, který má přístup k Internetu, stáhněte vzdálené schéma:
     a. Klepněte na volbu **Soubor > Import > Obchodní integrace > WSDL a XSD**.
     b. Vyberte volbu **Vzdálený soubor WSDL** nebo **Soubor XSD**.
     c. Importujte následující schémata:

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

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

        http://www.w3.org/TR/xmldsig-core/xmldsig-core-schema.xsd
  2. Zkopírujte tato schémata do počítače, který nemá přístup k Internetu.
  3. Na počítači, který nemá přístup k Internetu, toto schéma importujte:
     a. Klepněte na volbu **Soubor > Import > Obchodní integrace > WSDL a XSD**.
     b. Vyberte volbu **Lokální soubor WSDL** nebo **Soubor XSD**.
  4. Změňte umístění schématu pro soubor oasis-wss-wssecurity_secext-1.1.xsd:
     a. Schéma otevřete na cestě *umístění_pracovní_blasti*/*název_modulu*/StandardImportFilesGen/oasis-wss-wssecurity-secext-1.1.xsd.
     b. Změňte:

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

        na:

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

     c. Změňte:

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

        na:

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

  5. Změňte umístění schématu pro soubor oasis-200401-wss-wssecurity-secext-1.0.xsd:
     a. Schéma otevřete na cestě *umístění_pracovní_blasti*/*název_modulu*/StandardImportFilesGen/oasis-200401-wss-wssecurity-secext-1.0.xsd.
     b. Změňte:

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

        na:

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

  6. Projekt vyčistěte a znovu sestavte.

Šíření záhlaví přenosu:

Při obsluze zpráv SOAP může být zapotřebí přistupovat k informacím z určitých záhlaví transportu přijímaných zpráv. Zajistěte, aby byly zprávy se záhlavími transportu odesílány se specifickými hodnotami nebo umožněte záhlavím transportu minout určitý modul.

Při konfiguraci vazby webové služby v produktu Integration Designer můžete označit, že chcete záhlaví transportu šířit.

- Při obdržení požadavků při exportu nebo odezev při importu lze přistupovat k informacím ze záhlaví transportu, což umožňuje založit logiku modulu na hodnotách daného záhlaví a umožňuje to také úpravu těchto záhlaví.

- Když jsou odezvy odeslány z exportu nebo požadavky z importu, lze do těchto zpráv zahrnout záhlaví transportu.

Určení šíření záhlaví

Chcete-li nakonfigurovat šíření záhlaví transportu pro import a export, postupujte takto:

1. V pohledu Vlastnosti produktu Integration Designer vyberte položku **Vazba > Šíření**.
2. Nastavte požadovanou volbu šíření záhlaví transportu.

Poznámka: Šíření záhlaví transportu je standardně zakázáno a lze je implementovat pouze do běhového prostředí verze 7.0.0.3 (nebo novějšího). Poznámka: Ve verzi 7.0.0.3 je také šíření záhlaví transportu omezeno pouze na záhlaví transportu HTTP.

Pokud šíření záhlaví transportu povolíte, budou záhlaví z přijatých zpráv šířena napříč modulem a pokud je výslovně neodeberete, budou tato záhlaví použita v následujících vyvoláních v rámci stejného podprocesu.

Poznámka: Záhlaví transportu nelze šířit, pokud používáte vazbu webové služby (JAX-PRC).

Přístup k informacím ze záhlaví

Když je povoleno šíření záhlaví transportu příchozích zpráv, jsou všechna záhlaví transportu (včetně uživatelem definovaných záhlaví) viditelná v objektu SMO. Záhlaví můžete nastavit na jiné hodnoty, a vytvořit tak záhlaví nová. Poznámka: Neprobíhá žádná kontrola či ověření nastavených hodnot a jakákoliv nevhodná či chybná záhlaví mohou způsobit problémy s během webové služby.

Zvažte následující informace týkající se nastavení záhlaví HTTP:

- Žádné změny v záhlavích vyhrazené pro stroj webových služeb nebudou uplatněny v odchozí zprávě. Pro stroj webových služeb jsou například vyhrazena záhlaví verze či metoda HTTP, Content-Type, Content-Length a SOAPAction.
- Pokud je hodnota záhlaví číselná, mělo by být přímo nastaveno toto číslo (spíše než daný řetězec). Použijte například **Max-Forwards = 5** (spíše než **Max-Forwards = Max-Forwards: 5**) a **Age = 300** (spíše než **Age = Age: 300**).
- Pokud je velikost zprávy požadavku menší než 32 KB, stroj webových služeb odebere záhlaví Transfer-Encoding a místo toho nastaví záhlaví Content-Length na pevnou velikost dané zprávy.
- Záhlaví Content-Language resetuje WAS.channel.http na cestě odezvy.
- Neplatné nastavení upgradu vede k chybě 500.
- Následující záhlaví připojí hodnotu vyhrazenou strojem webových služeb k nastavení zákazníka:
  - User-Agent.
  - Cache-Control.
  - Pragma.
  - Accept.
  - Connection.

K informacím ze záhlaví získáte přístup jedním z následujících způsobů:

- Použitím mediačního primitiva pro přístup ke strukturám objektu SMO.

  Informace o použití mediačních primitiv viz odkazy “Související informace”.
- Použití rozhraní poskytovatele služeb (SPI) služby kontextu.

  Následující ukázkový kód přečte záhlaví transportu HTTP ze služby kontextu:

```
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();
       }
  }
```

Odstraňování problémů

Pokud při odesílání přezkoumaných záhlaví narazíte na problémy, můžete zprávu TCP/IP zachytit pomocí nástrojů, jako je např. TCP/IP Monitor z produktu Integration Designer. K nástroji TCP/IP Monitor získáte přístup prostřednictvím volby **Spustit/Ladit > TCP/IP Monitor** na stránce Předvolby.

Hodnoty záhlaví si můžete také nechat zobrazit prostřednictvím trasování stroje JAX-WS: **org.apache.axis2.*=all: com.ibm.ws.websvcs.*=all:**

Práce s vazbami webové služby (JAX-WS):

Když ve svých aplikacích používáte vazby webové služby (JAX-WS), můžete dané vazbě přidat kvalitu služby SAML (Security Assertion Markup Language). Nejprve musíte prostřednictvím administrativní konzoly importovat sadu zásad. Prostřednictvím administrativní konzoly se můžete také ujistit, že je server řádně nakonfigurovaný pro použití s vazbou webové služby (JAX-WS).

Importování sad zásad SAML:

Jazyk SAML (Security Assertion Markup Language) je standard OASIS založený na jazyku XML pro výměnu informací o identitě uživatele a atributech zabezpečení. Když konfigurujete vazbu webové služby (JAX-WS) v produktu Integration Designer, můžete určit sadu zásad SAML. Nejprve použijete administrativní konzolu produktu IBM Business Process Manager ke zpřístupnění sad zásad, aby je bylo možné importovat do produktu Integration Designer.

Sady zásad SAML se zpravidla nacházejí v adresáři konfigurace profilu:

kořenový_adresář_profilu/config/templates/PolicySets

Před zahájením této procedury ověřte, že se v adresáři konfigurace profilu nacházejí následující adresáře (které obsahují sady zásad):

- 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

Pokud tyto adresáře v adresáři konfigurace profilu nejsou, zkopírujte je do tohoto adresáře z následujícího umístění:

kořenový_adresář_apl_serveru/profileTemplates/default/documents/config/templates/PolicySets

Sady zásad naimportujete do administrativní konzoly, vyberete ty, které chcete zpřístupnit produktu Integration Designer, a poté uložíte soubor .zip pro každou z těchto sad zásad do umístění, které je přístupné produktu Integration Designer.

1. Naimportujte sady zásad podle následujících kroků:
   a. V administrativní konzole klepněte na volbu **Služby > Sady zásad > Sady zásad aplikace**.
   b. Klepněte na volbu **Importovat > Z výchozího úložiště**.
   c. Vyberte výchozí sady zásad SAML a klepněte na tlačítko **OK**.
2. Exportujte sady zásad, aby je mohl použít produkt Integration Designer:
   a. Na stránce Sady zásad aplikace vyberte sadu zásad SAML, kterou chcete exportovat, a klepněte na volbu **Exportovat**.

      Poznámka: Pokud není stránka Sady zásad aplikace aktuálně zobrazena, klepněte v administrativní konzole na volbu **Služby > Sady zásad > Sady zásad aplikace**.
   b. Na další stránce klepněte na odkaz na soubor .zip pro sadu zásad.
   c. V okně Stažení souboru klepněte na tlačítko **Uložit** a označte umístění, které je přístupné produktu Integration Designer.
   d. Klepněte na tlačítko **Zpět**.
   e. Kroky 2a na stránce 74 až 2d na stránce 74 proveďte pro všechny sady zásad, které chcete exportovat.

Sady zásad SAML jsou uloženy do souborů .zip a jsou připraveny k importu do produktu Integration Designer.

Naimportujte sady zásad do produktu Integration Designer, jak je popsáno v tématu “Sady zásad”.

Vyvolání webových služeb, které vyžadují základní ověření HTTP:

Základní ověření HTTP ověřuje klienta služby v zabezpečeném koncovém bodu s použitím jména uživatele a hesla. Základní ověření HTTP můžete nastavit při odesílání nebo přijímání požadavků webové služby.

Základní ověření HTTP pro příjem požadavků webové služby nastavíte nakonfigurováním vazby exportu rozhraní Java API pro webové služby XML (JAX-WS), jak je popsáno v tématu Vytvoření a přiřazování rolí zabezpečení exportům webových služeb.

Základní ověření HTTP pro požadavky webové služby odeslané vazbou importu JAX-WS lze povolit jedním ze dvou způsobů:

- Při konfiguraci vazby importu v modulu SCA můžete vybrat dodanou sadu zásad ověření HTTP s názvem BPMHTTPBasicAuthentication (která je poskytována s vazbou importu webové služby (JAX-WS)) nebo kteroukoli jinou sadu zásad, která obsahuje zásadu HTTPTransport.
- Při konstruování modulu SCA můžete s využitím možností mediačního toku dynamicky vytvořit nové záhlaví ověřování HTTP a určit jméno uživatele a heslo v záhlaví.

Poznámka: Sada zásad má přednost před hodnotou určenou v záhlaví. Chcete-li použít hodnotu nastavenou v záhlaví ověření HTTP za běhu, nepřipojujte sadu, která obsahuje zásadu HTTPTransport. Konkrétně nepoužívejte výchozí sadu zásad BPMHTTPBasicAuthentication, a pokud jste definovali sadu zásad, zkontrolujte, zda vylučuje zásadu HTTPTransport.

Další informace o sadách zásad webové služby a vazbách zásad a o jejich použití viz Sady zásad webových služeb Informačního centra aplikačního serveru WAS.

- Chcete-li použít dodávanou sadu zásad, postupujte takto:
  1. Volitelné: V administrativní konzole vytvořte obecnou vazbu zásad klienta nebo upravte stávající vazbu zásad, která obsahuje zásadu HTTPTransport s požadovanými hodnotami ID uživatele a hesla.
  2. V produktu IBM Integration Designer vygenerujte vazbu importu webové služby (JAX-WS) a připojte k ní sadu zásad BPMHTTPBasicAuthentication.

3. Proveďte *jeden* z následujících kroků:
   – V produktu IBM Integration Designer ve vlastnostech vazby importu webové služby (JAX-WS) zadejte název existující obecné vazby zásad klienta, která obsahuje zásadu HTTPTransport.
   – Po implementaci modulu SCA použijte administrativní konzolu k výběru stávající vazby zásady klienta, nebo k vytvořené nové vazby zásady klienta a poté ji přidružte k vazbě importu.
4. Volitelné: V administrativní konzole komponenty Process Server upravte vybranou vazbu sady zásad tak, aby určovala požadované ID a heslo.

- Chcete-li určit jméno uživatele a heslo v záhlaví ověření HTTP, proveďte jednu z následujících sad kroků:
  – Pomocí mediačního primitiva Modul nastavení záhlaví HTTP v produktu IBM Integration Designer vytvořte záhlaví ověření HTTP a určete jméno uživatele a heslo.
  – Je-li vyžadována další logika, pak pomocí kódu Java v primitivu Vlastní mediace (viz následující příklad):
    1. Vytvořte záhlaví ověření HTTP.
    2. Určete jméno uživatele a heslo.
    3. Přidejte nové záhlaví ověření HTTP do zásady HTTPControl.
    4. Nastavte aktualizovanou zásadu HTTPControl zpět do kontextové služby.

```
//Získejte HeaderInfoType z contextService
ContextService contextService = (ContextService) ServiceManager.INSTANCE
.locateService("com/ibm/bpm/context/ContextService");
HeaderInfoType headers = contextService.getHeaderInfo();
if(headers == null){
    headers = ContextObjectFactory.eINSTANCE.createHeaderInfoType();
}
//Získejte záhlaví HTTP a řízení HTTP z HeaderInfoType
HTTPHeaderType httpHeaderType = headers.getHTTPHeader();
HTTPControl cp = httpHeaderType.getControl();
HeadersFactory factory = HeadersFactory.eINSTANCE;
if(cp == null){
    cp = factory.createHTTPControl();
}
//Vytvořte nové ověření HTTPAuthentication a nastavte 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);
// Nastavte informace záhlaví zpět do aktuálního kontextu provedení.
contextService.setHeaderInfo(headers);
```

Kontrola konfigurace serveru:

Při implementaci aplikace s vazbou webové služby (JAX-WS) je třeba zkontrolovat, zda server, na který se aplikace implementuje, nemá vybranou volbu **Spouštět komponenty podle potřeby**.

Výběr této volby můžete zkontrolovat provedením následujících kroků z administrativní konzoly:

1. Klepněte na volbu **Servery > Typy serverů > Aplikační servery WebSphere Application Server**.
2. Klepněte na název serveru.
3. Na kartě Konfigurace určete, zda je vybrána volba **Spouštět komponenty podle potřeby**.
4. Proveďte jeden z následujících kroků:
   - Pokud je volba **Spouštět komponenty podle potřeby** vybrána, odeberte její označení a potom klepněte na tlačítko **Použít**.
   - Pokud volba **Spouštět komponenty podle potřeby** vybrána není, klepněte na tlačítko **Storno**.

Přílohy ve zprávách SOAP:

Můžete odesílat a přijímat zprávy SOAP obsahující binární data (např. soubory PDF a obrázky JPEG) jako přílohu. Přílohy mohou být *odkazované* (tzn. reprezentované přímo částí zprávy v rozhraní dané služby) nebo *neodkazované* (u kterých lze zahrnout libovolné množství a typy příloh).

Odkazovaná příloha může mít jednu z následujících forem:

- Přílohy MTOM používají kódování určené mechanizmem MTOM (Message Transmission Optimization Mechanism) (http://www.w3.org/TR/soap12-mtom/) SOAP. Přílohy MTOM jsou povoleny prostřednictvím volby konfigurace ve vazbách importu a exportu a představují doporučený způsob kódování příloh pro nové aplikace.
- Prvek wsi:swaRef-typed ve schématu zprávy

  Přílohy definované pomocí typu wsi:swaRef jsou v souladu s profilem *Attachments Profile Version 1.0* organizace WS-I (Web Services Interoperability Organization)(http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html), který definuje, jakým způsobem prvky zprávy souvisí k částmi formátu MIME.
- Část zprávy nejvyšší úrovně používající typ binárního schématu

  Přílohy ve formě částí zpráv nejvyšší úrovně jsou v souladu se specifikací *Zprávy SOAP s přílohami* (http://www.w3.org/TR/SOAP-attachments).

  Přílohy ve formě zpráv nejvyšší úrovně mohou být také nakonfigurovány tak, aby zajistili soulad dokumentů a zpráv WSDL produkovaných vazbou s profily *Attachments Profile Version 1.0* a *Basic Profile Version 1.1* organizace WS-I (http://www.ws-i.org/Profiles/BasicProfile-1.1.html).

Neodkazovanou přílohu nese zpráva SOAP, aniž by tuto přílohu ve schématu dané zprávy cokoliv představovalo.

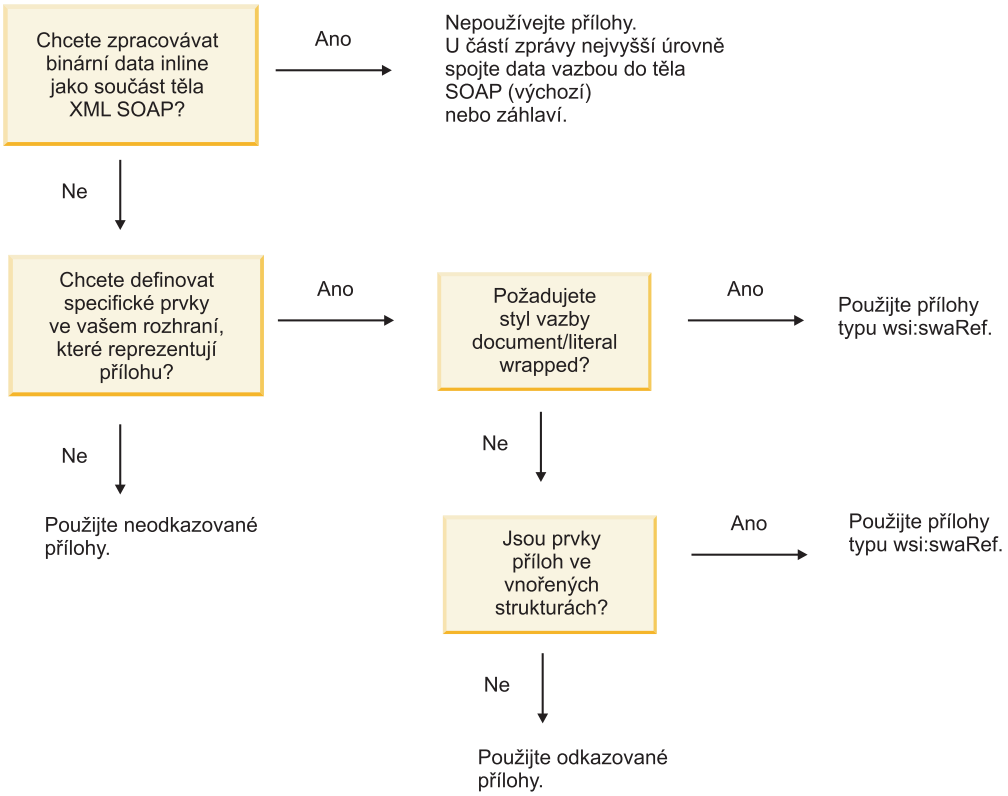
Ve všech případech s výjimkou příloh MTOM musí vazba WSDL SOAP obsahovat vazbu MIME pro přílohy, které se mají použít , a maximální velikost příloh nesmí překročit 20 MB.

Poznámka: Chcete-li odesílat nebo přijímat zprávy SOAP s přílohami, musíte použít jednu z vazeb webové služby založenou na rozhraní JAX-WS (Java API for XML Web Services).

Jak vybrat příslušný styl přílohy:

Když navrhujete nové rozhraní služby, která obsahuje binární data, zvažte, jak jsou tato data ve zprávách SOAP odesílaných a přijímaných touto službou přenášena.

Pokud připojená aplikace webových služeb podporuje mechanizmus MTOM (Message Transmission Optimization Mechanism), měl by se pro přílohy používat. Pokud ne, ukazuje následující diagram, jak vybírat jiné přílohy. Pomocí následující sady otázek určete vhodný styl přílohy:



Přílohy mechanizmu MTOM: části zpráv nejvyšší úrovně:

Můžete odesílat a přijímat zprávy webové služby obsahující v rámci zprávy SOAP přílohy mechanizmu MTOM (Message Transmission Optimization Mechanism). Ve zprávě SOAP formátu MIME s více částmi je tělo zprávy SOAP první částí zprávy a příloha nebo přílohy jsou v navazujících částech.

Když v rámci zprávy SOAP odesíláte nebo přijímáte odkazovanou přílohu, jsou binární data, ze kterých daná příloha sestává (která jsou často rozsáhlá), zadržována odděleně od těla zprávy SOAP, takže nemusí být analyzována jako XML. Výsledkem je efektivnější zpracování, než kdyby byla binární data zadržována v rámci prvku XML.

Následuje ukázka zprávy SOAP využívající mechanizmus MTOM:

```
... other transport headers ... Content-Type: multipart/related;
boundary=MIMEBoundaryurn_uuid_0FE43E4D025F0BF3DC11582467646812;
type="application/xop+xml"; start="
<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:0FE43E4D025F0BF3DC11582467646811@apache.org"/></imageData>
            </input>
         </sendImage>
      </soapenv:Body>
```

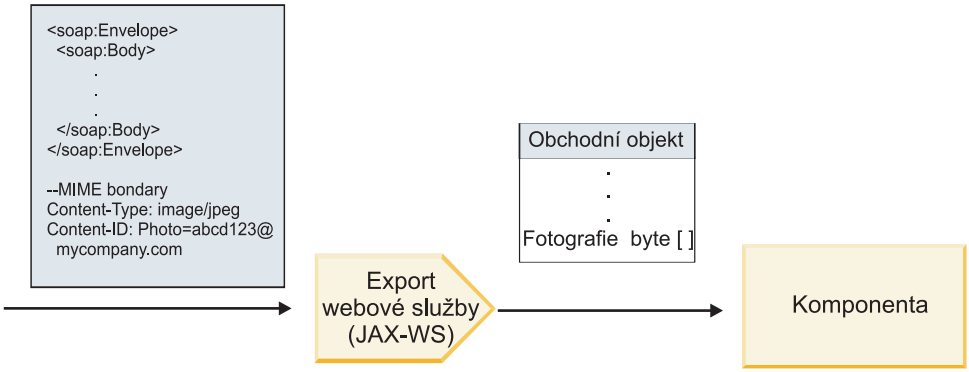
```
   </soapenv:Envelope>
--MIMEBoundaryurn_uuid_0FE43E4D025F0BF3DC11582467646812
content-type: text/plain
content-transfer-encoding: binary
content-id:
         <1.urn:uuid:0FE43E4D025F0BF3DC115824676468l1@apache.org>

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

Poznámka: V této ukázce mechanizmu MTOM má atribut content-type obálky SOAP hodnotu **application/xop+xml** a binární data nahrazuje prvek **xop:Include**, viz níže:

```
<xop:Include xmlns:xop="http://www.w3.org/2004/08/xop/include"
   href="cid:1.urn:uuid:0FE43E4D025F0BF3DC115824676468l1@apache.org"/>
```

Příchozí zpracování odkazovaných příloh

Když klient předává zprávu SOAP s přílohou komponentě SCA (Service Component Architecture), webová služba (JAX-WS) nejprve tuto přílohu odebere. Dále provede analýzu části SOAP dané zprávy a vytvoří obchodní objekt. Nakonec vazba nastaví binární soubor přílohy v tomto obchodním objektu.



Obrázek 58. Jak vazba exportu webové služby (JAX-WS) zpracovává zprávu SOAP obsahující odkazovanou přílohu

Atributy přílohy mechanizmu MTOM

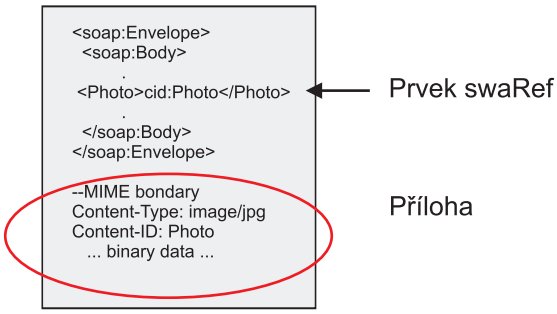
- Mechanizmus MTOM podporuje prvky přílohy s vnořenými strukturami.
- Mechanizmus MTOM je k dispozici pouze pro typ base64Binary.
- Mechanizmus MTOM podporuje prvky přílohy s vnořenými strukturami, což znamená, že je v případě příloh mechanizmu MTOM atribut **bodyPath** roven umístění **xpath** prvku, který danou přílohu mechanizmu MTOM zadržuje. Logika výpočtu prvku **bodyPath** striktně následuje schéma generování umístění **xpath**, jak je uvedeno v příkladech níže:
  - Pro typ bez pole (**maxOccurs** je rovno 1): /sendImage/input/imageData.
  - Pro typ pole (**maxOccurs** > 1): /sendImage/input/imageData[1].
- Nejsou podporovány smíšené typy příloh. Pokud je tedy ve vazbě importu povolen mechanizmus MTOM, bude generována příloha MTOM. Pokud je mechanizmus MTOM zakázaný nebo byla ponechána výchozí hodnota konfigurace mechanizmu MTOM, příchozí zprávy s mechanizmem MTOM nejsou podporovány.

Odkazované přílohy: prvky typu swaRef:

Můžete odesílat a přijímat zprávy SOAP obsahující přílohy reprezentované v rozhraní služby jako prvky typu swaRef.

Prvek typu swaRef je definován v profilu *Attachments Profile* verze 1.0 interoperability WS-I (Web Services Interoperability Organization) (http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html), který definuje, jakým způsobem se prvky zprávy vztahují k částem formátu MIME.

Ve zprávě SOAP obsahuje tělo SOAP prvek typu swaRef, který identifikuje ID obsahu dané přílohy.



Obrázek 59. Zpráva SOAP s prvkem swaRef

Kód jazyka WSDL této zprávy SOAP obsahuje v rámci části zprávy identifikující danou přílohu prvek typu swaRef.

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

Kód jazyka WSDL by měl také obsahovat vazbu formátu MIME označující, že musí být použity zprávy formátu MIME o více částech.

Poznámka: Kód jazyka WSDL *nezahrnuje* vazbu formátu MIME pro specifický prvek zprávy typu swaRef, protože vazby formátu MIME platí pouze pro části zpráv nejvyšší úrovně.

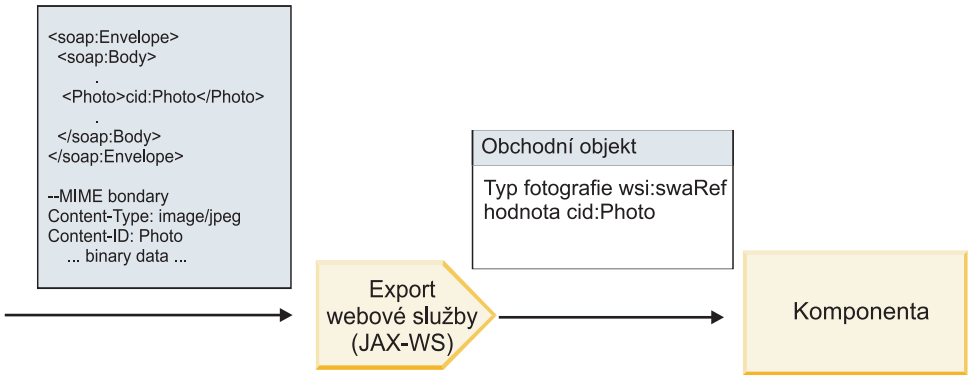
Přílohy reprezentované prvky typu swaRef lze šířit pouze mezi komponentami mediačního toku. Pokud musí mít k příloze přístup jiný typ komponenty, nebo pokud ji musí jiný typ komponenty šířit, přesuňte přílohu pomocí komponenty mediačního toku do umístění, které je pro danou komponentu přístupné.

Příchozí zpracování příloh

Ke konfiguraci přijetí přílohy prostřednictvím vazby exportu se používá produkt Integration Designer. Vytvoříte modul, stejně jako jeho přidružené rozhraní a operace, včetně prvku typu swaRef. Potom vytvoříte vazbu webové služby (JAX-WS).

Poznámka: Další podrobné informace viz téma “Práce s přílohami” v Informačním centru produktu Integration Designer.

Když klient předává zprávu SOAP s přílohou typu swaRef komponentě SCA (Service Component Architecture), vazba exportu webové služby (JAX-WS) nejprve tuto přílohu odebere. Dále provede analýzu části SOAP dané zprávy a vytvoří obchodní objekt. Nakonec vazba nastaví ID obsahu přílohy v tomto obchodním objektu.



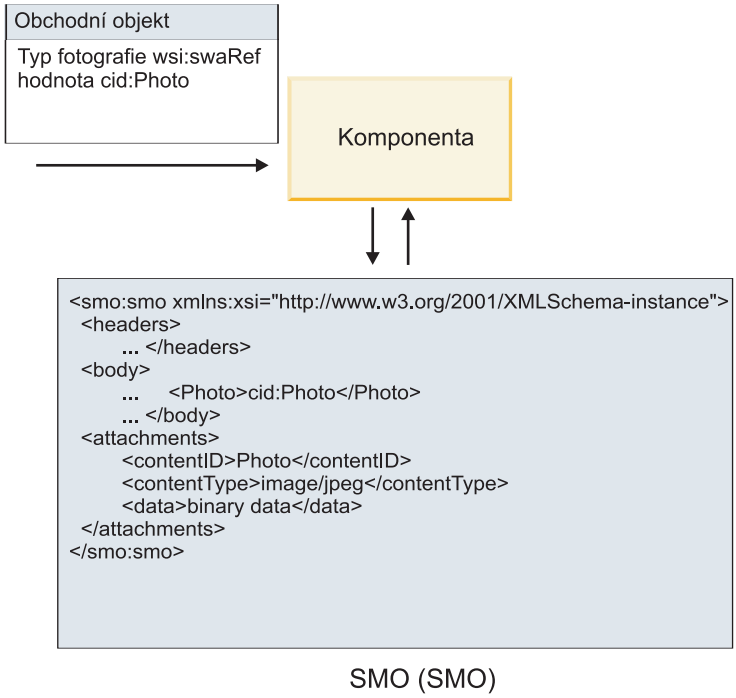
Obrázek 60. Jak vazba exportu webové služby (JAX-WS) zpracovává zprávu SOAP s přílohou typu swaRef

Přístup k metadatům přílohy v komponentě mediačního toku

Jak ukazuje Obrázek 16 na stránce 80, když k přílohám typu swaRef přistupují komponenty, zobrazuje se identifikátor obsahu přílohy jako prvek typu swaRef.

Každá příloha zprávy SOAP obsahuje také v objektu SMO odpovídající prvek **attachments**. Když používáte typ swaRef interoperability WS-I, obsahuje prvek **attachments** typ obsahu přílohy a ID obsahu, stejně jako skutečná binární data dané přílohy.

Za účelem získání hodnoty přílohy typu swaRef je proto nutné získat hodnotu prvku typu swaRef a následně vyhledat prvek **attachments** s odpovídající hodnotou atributu **contentID**. Poznámka: Z hodnoty atributu **contentID** je zpravidla odebrána předpona **cid:** hodnoty swaRef.



Obrázek 61. Způsob zobrazování příloh typu swaRef v objektu SMO

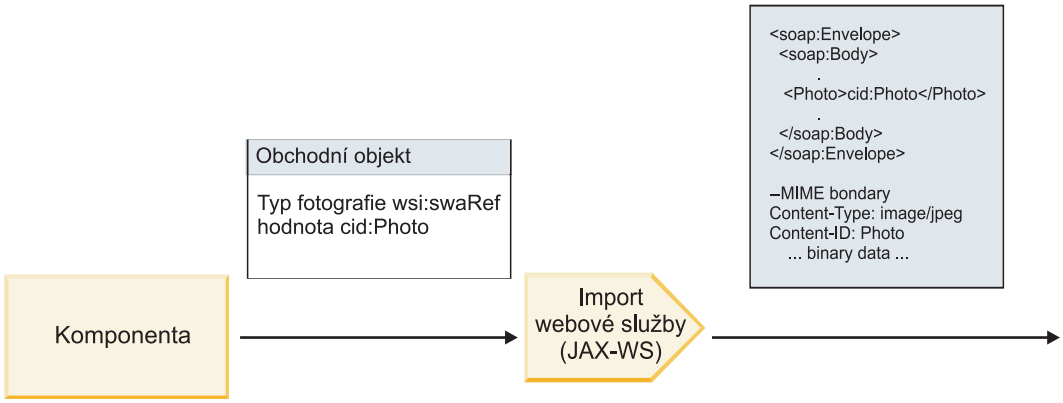
Odchozí zpracování

Pomocí produktu Integration Designer konfigurujete vazbu importu webové služby (JAX-WS) pro účely vyvolání externí webové služby. Vazba importu se konfiguruje pomocí dokumentu WSDL, popisujícího vyvolávanou webovou službu a definujícího, které přílohy budou předány webové službě.

Když vazba importu webové služby (JAX-WS) obdrží zprávu SCA, dojde k odeslání prvků typu swaRef v podobě příloh, pokud je import propojený s komponentou mediačního toku a pokud má prvek typu swaRef odpovídající prvek **attachments**.

V případě odchozího zpracování jsou prvky typu swaRef vždy odesílány se svými hodnotami ID obsahu. Mediační modul však musí vždy zajistit přítomnost příslušného prvku **attachments** s odpovídající hodnotou atributu **contentID**.

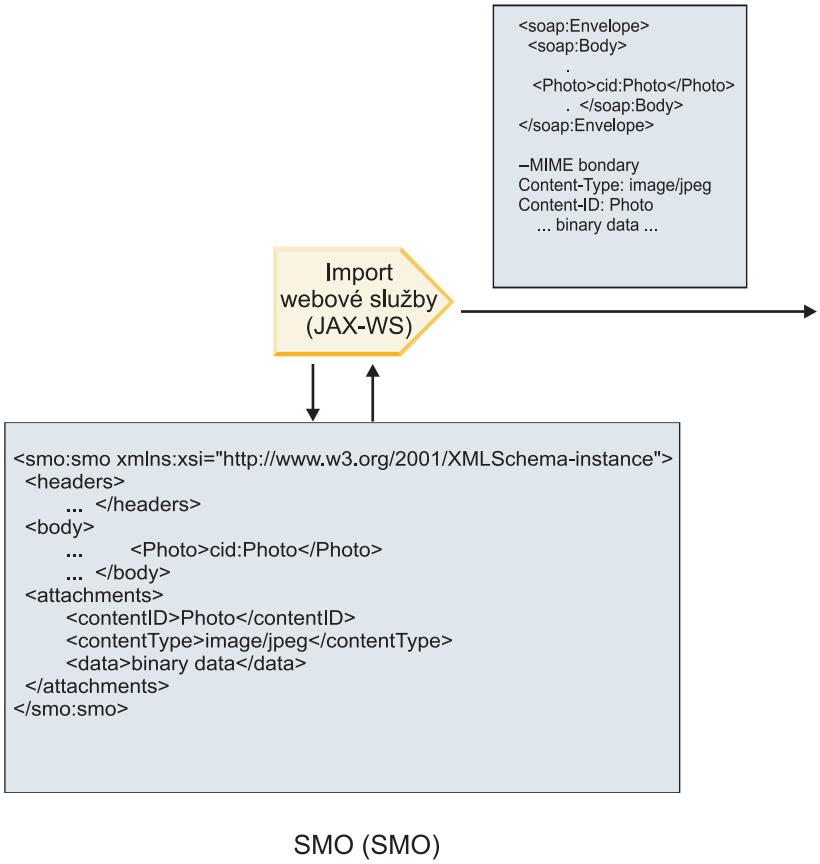
Poznámka: V zájmu zajištění kompatibility s profilem WS-I Attachments Profile by měla hodnota atributu **content ID** odpovídat "kódování části content-id", jak je popsáno v sekci 3.8 profilu interoperability WS-I *Attachments Profile* verze 1.0.



Obrázek 62. Jak vazba importu webové služby (JAX-WS) generuje zprávu SOAP s přílohou typu swaRef

Nastavení metadat přílohy v komponentě mediačního toku

Pokud objekt SMO obsahuje hodnotu prvku typu swaRef a také prvek **attachments**, vazba připraví zprávu SOAP (s přílohou) a odešle ji příjemci.



Obrázek 63. Způsob přístupu k příloze typu swaRef z modulu SMO za účelem vytvoření zprávy SOAP

Prvek **attachments** je v modulu SMO přítomen pouze, pokud je komponenta mediačního toku přímo propojená k importu či exportu. Nepředává se jiným typům komponent. Pokud jsou dané hodnoty potřebné v modulu obsahujícím jiné typy komponent, měly by se tyto hodnoty zkopírovat pomocí komponenty mediačního toku do umístění, ke kterému lze v daném modulu přistupovat. Další komponenta mediačního toku by potom měla být použita k nastavení správných hodnot ještě před odchozím vyvoláním, a to prostřednictvím importu webové služby.

Důležité: Jak je popsáno v tématu “Reprezentace XML objektu SMO,” mediační primitivum Mapování transformuje zprávy pomocí transformace XSLT 1.0. Transformace provádí serializaci XML objektu SMO. Mediační primitivum Mapování umožňuje určit kořen serializace a kořenový prvek dokumentu XML tento kořen odráží.

Když odesíláte zprávy SOAP s přílohami, vámi vybraný kořenový prvek určuje, jak budou přílohy šířeny.

- Pokud vyberete jako kořen mapy XML volbu “/body”, budou všechny přílohy šířeny po této mapě standardně.
- Pokud vyberete jako kořen mapy “/”, můžete šíření příloh řídit.

Odkazované přílohy: části zprávy nejvyšší úrovně:

Můžete odesílat a přijímat zprávy SOAP obsahující binární přílohy deklarované jako části vašeho rozhraní služby.

Ve zprávě SOAP formátu MIME s více částmi je tělo zprávy SOAP první částí zprávy a příloha nebo přílohy jsou v navazujících částech.

Jaké výhody má odesílání či přijímání zpráv SOAP s odkazovanými přílohami? Binární data, ze kterých sestává daná příloha (která jsou často rozsáhlá), jsou zadržována odděleně od těla zprávy SOAP, takže nemusí být analyzována jako XML. Výsledkem je efektivnější zpracování, než kdyby byla binární data zadržována v rámci prvku XML.

Typy zpráv SOAP s odkazovanými přílohami

Počínaje verzí 7.0.0.3 produktu IBM Business Process Manager si můžete volit způsob generování zprávy SOAP:

- **Zprávy kompatibilní s interoperabilitou webových služeb (WS-I)**

  Běhové prostředí může generovat zprávy SOAP, které jsou kompatibilní s profilem WS-I *Attachments Profile verze 1.0* a WS-I *Basic Profile verze 1.1*. V případě zprávy SOAP kompatibilní s těmito profily je pouze jedna část zprávy vázaná na tělo zprávy SOAP. U částí, které jsou vázané jako přílohy, se kódování části content-id (jak je popsáno v profilu WS-I *Attachments Profile verze 1.0*) používá k vytvoření vztahu mezi přílohou a danou částí zprávy.
- **Zprávy, které nejsou kompatibilní s interoperabilitou WS-I**

  Běhové prostředí může generovat zprávy SOAP, které nejsou kompatibilní s profily WS-I, ale které jsou kompatibilní se zprávami generovanými ve verzi 7.0 nebo 7.0.0.2 produktu IBM Business Process Manager. Zprávy SOAP používají prvky nejvyšší úrovně pojmenované po části zprávy s atributem **href**, který zadržuje atribut **content-id** přílohy, ale kódování části content-id (jak je popsáno v profilu WS-I *Attachments Profile verze 1.0*) se nepoužívá.

Výběr kompatibility s interoperabilitou WS-I pro exporty webových služeb

Ke konfiguraci vazby exportu se používá produkt Integration Designer. Vytvoříte modul, stejně jako jeho přidružené rozhraní a operace. Potom vytvoříte vazbu webové služby (JAX-WS). Stránka Odkazované přílohy zobrazí všechny binární části vytvořené operace a vy vyberete, které části budou přílohy. Na stránce Určení kompatibility s WS-I AP 1.0 produktu Integration Designer potom určíte jednu z následujících voleb:

- **Použít zprávu SOAP kompatibilní s WS-I AP 1.0**.

  Pokud vyberete tuto volbu, určíte také, jaká část zprávy má být vázána na tělo zprávy SOAP.

  Poznámka: Tuto volbu lze použít pouze v případě, že je také odpovídající soubor WSDL kompatibilní s interoperabilitou WS-I.

  Soubor WSDL generovaný produktem Integration Designer verze 7.0.0.3 je kompatibilní s interoperabilitou WS-I. Pokud ovšem importujete soubor WSDL, který s interoperabilitou WS-I kompatibilní není, tuto volbu vybrat nemůžete.
- **Použít zprávu SOAP nekompatibilní s WS-I AP 1.0**

  Pokud vyberete tuto volbu, což je výchozí nastavení, je první část zprávy vázaná na tělo zprávy SOAP.

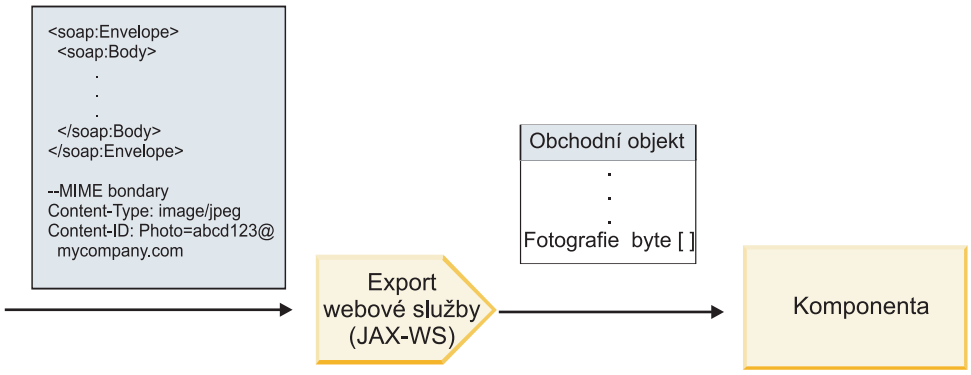
Poznámka: Jako odkazované přílohy lze odesílat či přijímat pouze části zpráv nejvyšší úrovně (tzn. prvky definované v rámci parametru portType jazyka WSDL jako součásti výstupní či vstupní zprávy), které obsahují binární typ (buď base64Binary, nebo hexBinary).
Další podrobné informace viz téma “Práce s přílohami” v Informačním centru produktu Integration Designer.

U zpráv kompatibilních s interkompatibilitou WS-I je identifikátor content-ID generovaný ve zprávě SOAP zřetězením následujících prvků:

- Hodnota atributu **name** prvku **wsdl:part** odkazovaného prvkem **mime:content**.
- Znak **=**.
- Globálně jedinečná hodnota, jako např. identifikátor UUID.
- Znak **@**.
- Platný název domény.

Příchozí zpracování odkazovaných příloh

Když klient předává zprávu SOAP s přílohou komponentě SCA (Service Component Architecture), webová služba (JAX-WS) nejprve tuto přílohu odebere. Dále provede analýzu části SOAP dané zprávy a vytvoří obchodní objekt. Nakonec vazba nastaví binární soubor přílohy v tomto obchodním objektu.

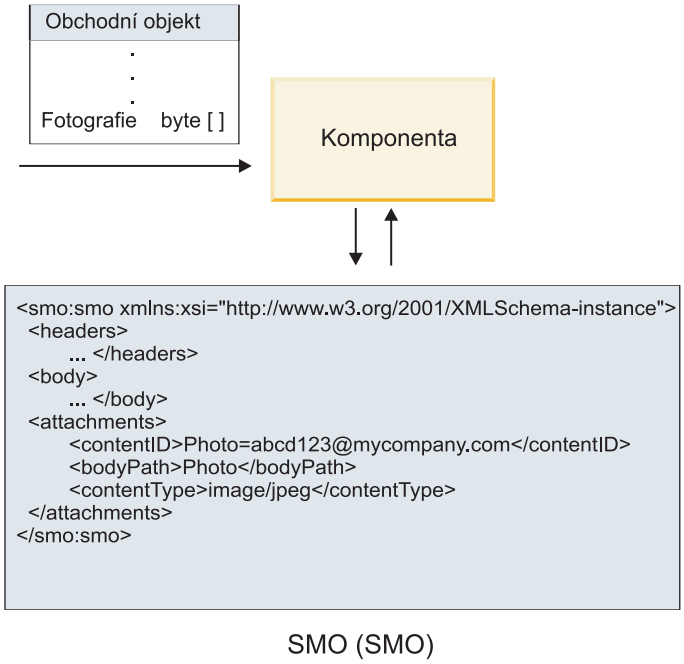


Obrázek 64. Jak vazba exportu webové služby (JAX-WS) zpracovává zprávu SOAP kompatibilní s interoperabilitou WS-I obsahující odkazovanou přílohu

Přístup k metadatům přílohy v komponentě mediačního toku

Jak ukazuje Obrázek 19 na stránce 84, když k odkazovaným přílohám přistupují komponenty, zobrazují se data přílohy jako bajtové pole.

Každá odkazované příloha zprávy SOAP obsahuje také v objektu SMO odpovídající prvek **attachments**. Prvek **attachments** obsahuje typ obsahu přílohy a cestu k prvku těla zprávy, ve kterém je příloha zadržována.



Obrázek 65. Způsob zobrazování odkazovaných příloh v objektu SMO

Důležité: V případě transformace zprávy a přesunutí přílohy nedochází k automatické aktualizaci cesty k danému prvku těla zprávy. K aktualizaci prvku **attachments** pomocí nové cesty můžete použít mediační tok (například jako součást transformace nebo pomocí odděleného modulu nastavení prvků zpráv).

Způsob sestavení odchozích zpráv SOAP

Pomocí produktu Integration Designer konfigurujete vazbu importu webové služby (JAX-WS) pro účely vyvolání externí webové služby. Vazba importu se konfiguruje pomocí dokumentu WSDL, popisujícího vyvolávanou webovou

službu a definujícího, které části zpráv mají být předávány jako přílohy. Na stránce Určení kompatibility s WS-I AP 1.0 produktu Integration Designer můžete také označit jednu z následujících voleb:

- **Použít zprávu SOAP kompatibilní s WS-I AP 1.0**.

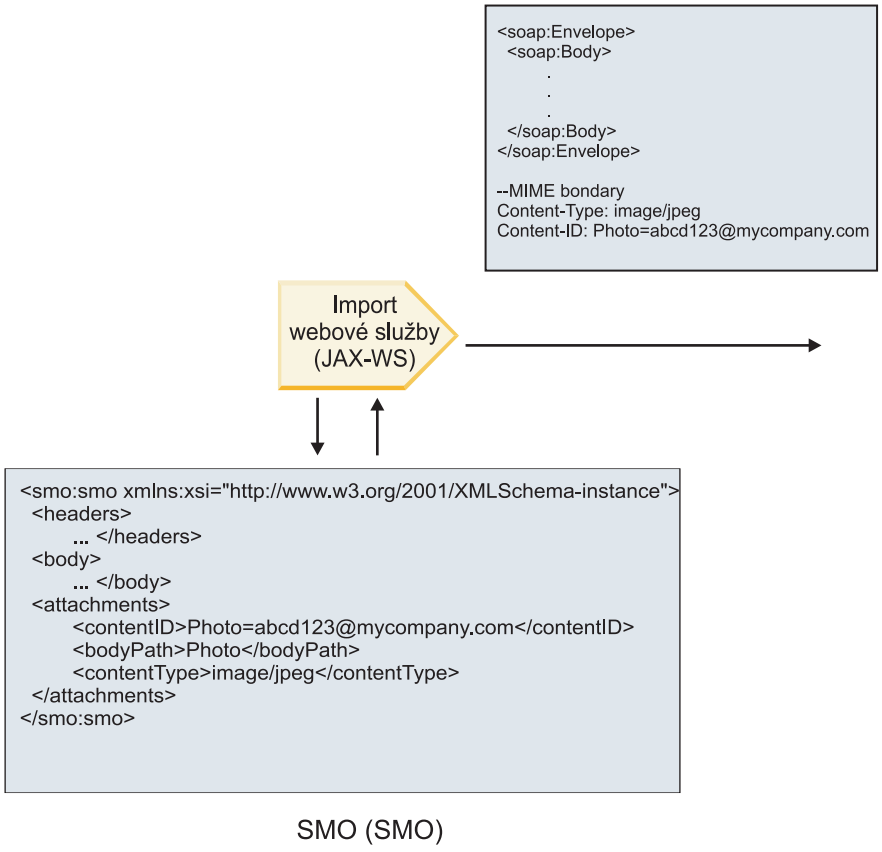
  Pokud vyberete tuto volbu, určíte také, jaká část zprávy má být vázána na tělo zprávy SOAP. Všechny ostatní části budou vázané na přílohy či záhlaví. Zprávy odeslané vazbou nezahrnují prvky těla zprávy SOAP odkazující na tyto přílohy. Tento vztah je vyjádřen prostřednictvím ID obsahu přílohy obsahujícího název dané části zprávy.
- **Použít zprávu SOAP nekompatibilní s WS-I AP 1.0**

  Pokud vyberete tuto volbu, což je výchozí nastavení, je první část zprávy vázaná na tělo zprávy SOAP. Všechny ostatní části jsou vázané na přílohy či záhlaví. Zprávy odeslané vazbou zahrnují minimálně jeden prvek těla zprávy SOAP, který odkazuje na dané přílohy prostřednictvím atributu **href**.

Poznámka: Část představující přílohu, definovaná v jazyce WSDL, musí být jednoduchého typu (buď base64Binary, nebo hexBinary). Pokud určitou část definuje atribut complexType, nelze ji navázat jako přílohu.

Odchozí zpracování odkazovaných příloh

Vazba importu používá informace z objektu SMO k určení způsobu odesílání binárních zpráv nejvyšší úrovně v podobě příloh.



Obrázek 66. Způsob přístupu k odkazované příloze z modulu SMO za účelem vytvoření zprávy SOAP

Prvek **attachments** je v modulu SMO přítomen pouze, pokud je komponenta mediačního toku přímo propojená k importu či exportu. Nepředává se jiným typům komponent. Pokud jsou dané hodnoty potřebné v modulu obsahujícím jiné typy komponent, měly by se tyto hodnoty zkopírovat pomocí komponenty mediačního toku do umístění, ke kterému lze v daném modulu přistupovat. Další komponenta mediačního toku by potom měla být použita k nastavení správných hodnot ještě před odchozím vyvoláním, a to prostřednictvím importu webové služby.

Vazba určuje způsob odeslání zprávy (či zda vůbec byla odeslána) pomocí kombinace následujících podmínek:

- Zda existuje vazba formátu MIME jazyka WSDL pro část binární zprávy nejvyšší úrovně a pokud ano, jak je definovaný typ obsahu.
- Zda v objektu SMO existuje prvek **attachments**, jehož hodnota atributu **bodyPath** odkazuje na binární část nejvyšší úrovně.

Způsob vytváření příloh v případě existence prvku attachment v objektu SMO

Následující tabulka zobrazuje způsob vytvoření a odeslání přílohy v případě, že objekt SMO obsahuje prvek **attachment** s atributem **bodyPath**, který odpovídá části s názvem zprávy:

Tabulka 43. Způsob generování přílohy

Stav vazby formátu MIME jazyka WSDL pro binární část zprávy nejvyšší úrovně	**Způsob vytvoření a odeslání zprávy**
Je přítomna a zároveň platí jedna z následujících možností: • Daná část zprávy nemá definovaný typ obsahu. • Je definováno více typů obsahu. • Je definován zástupný symbol typu obsahu.	Část zprávy je odeslána jako příloha. Parametr **Content-Id** je nastavený na hodnotu prvku attachments, pokud je přítomen, jinak je vygenerován. Parametr **Content-Type** je nastavený na hodnotu prvku attachments, pokud je přítomen, jinak je nastaven na hodnotu **application/octet-stream**.
Je přítomna s jediným obsahem dané části zprávy bez zástupného znaku	Část zprávy je odeslána jako příloha. Parametr **Content-Id** je nastavený na hodnotu prvku attachments, pokud je přítomen, jinak je vygenerován. Parametr **Content-Type** je nastavený na hodnotu prvku attachments, pokud je přítomen, jinak je nastaven na typ definovaný v prvku obsahu MIME jazyka WSDL.
Není přítomna	Část zprávy je odeslána jako příloha. Parametr **Content-Id** je nastavený na hodnotu prvku attachments, pokud je přítomen, jinak je vygenerován. Parametr **Content-Type** je nastavený na hodnotu prvku attachments, pokud je přítomen, jinak je nastaven na hodnotu **application/octet-stream**. **Poznámka:** Odesílání částí zpráv v podobě příloh, pokud nebyly jako přílohy definovány v jazyce WSDL, může vést k porušení kompatibility s profilem WS-I Attachments Profile verze 1.0 a pokud je to možné, měli byste se mu vyhnout.

Způsob vytváření příloh v případě, že v objektu SMO neexistuje prvek attachment

Následující tabulka zobrazuje způsob vytvoření a odeslání přílohy v případě, že objekt SMO neobsahuje prvek **attachment** s atributem **bodyPath**, který odpovídá části s názvem zprávy:

Tabulka 44. Způsob generování přílohy

Stav vazby formátu MIME jazyka WSDL pro binární část zprávy nejvyšší úrovně	**Způsob vytvoření a odeslání zprávy**
Je přítomna a zároveň platí jedna z následujících možností: • Daná část zprávy nemá definovaný typ obsahu. • Je definováno více typů obsahu. • Je definován zástupný symbol typu obsahu.	Část zprávy je odeslána jako příloha. Generuje se atribut **Content-Id**. Atribut **Content-Type** je nastaven na hodnotu **application/octet-stream**.

Tabulka 44. Způsob generování přílohy (pokračování)

Stav vazby formátu MIME jazyka WSDL pro binární část zprávy nejvyšší úrovně	**Způsob vytvoření a odeslání zprávy**
Je přítomna s jediným obsahem dané části zprávy bez zástupného znaku	Část zprávy je odeslána jako příloha. Generuje se atribut **Content-Id**. Atribut **Content-Type** je nastaven na typ definovaný v prvku obsahu MIME jazyka WSDL.
Není přítomna	Část zprávy není odeslána jako příloha.

Důležité: Jak je popsáno v tématu "Reprezentace XML objektu SMO," mediační primitivum Mapování transformuje zprávy pomocí transformace XSLT 1.0. Transformace provádí serializaci XML objektu SMO. Mediační primitivum Mapování umožňuje určit kořen serializace a kořenový prvek dokumentu XML tento kořen odráží.

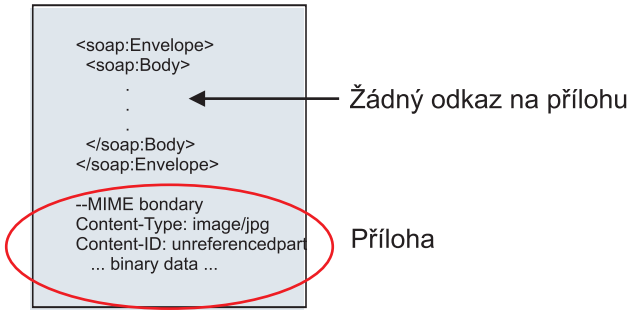
Když odesíláte zprávy SOAP s přílohami, vámi vybraný kořenový prvek určuje, jak budou přílohy šířeny.

- Pokud vyberete jako kořen mapy XML volbu "/body", budou všechny přílohy šířeny po této mapě standardně.
- Pokud vyberete jako kořen mapy "/", můžete šíření příloh řídit.

Neodkazované přílohy:

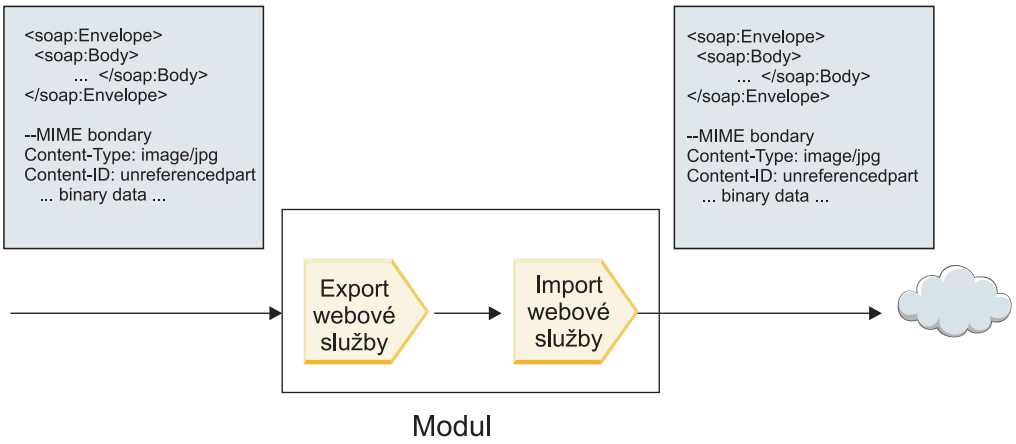
Můžete odesílat a přijímat *neodkazované* přílohy, které nejsou deklarované v rozhraní služby.

Ve zprávě SOAP formátu MIME s více částmi je tělo zprávy SOAP první částí zprávy a přílohy jsou v navazujících částech. Tělo zprávy SOAP neobsahuje žádný odkaz na přílohu.



Obrázek 67. Zpráva SOAP s neodkazovanou přílohou

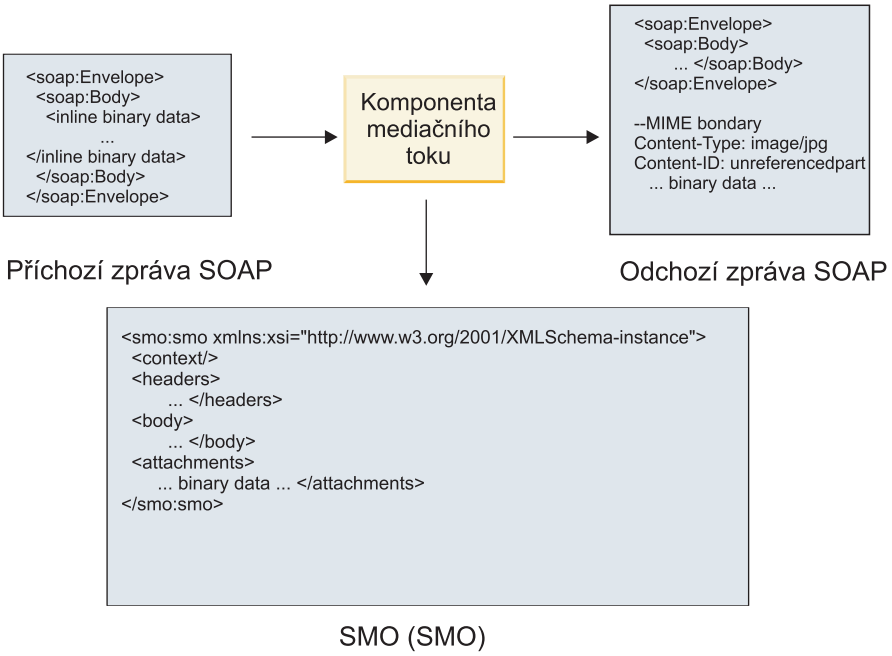
Zprávy SOAP s neodkazovanou přílohou můžete odesílat prostřednictvím exportu webové služby do importu webové služby. Danou přílohu obsahuje výstupní zpráva odeslaná do cílové webové služby.



Obrázek 68. Příloha procházející modulem SCA

Na obrázku Obrázek 23 na stránce 88 prochází zpráva SOAP s přílohou beze změn.

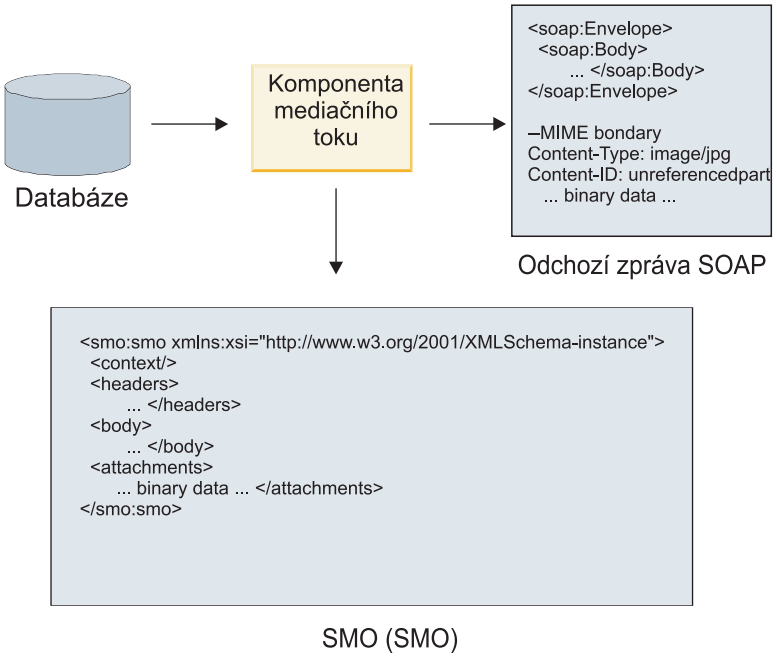
Zprávu SOAP můžete změnit také pomocí komponenty mediačního toku. Komponentu mediačního toku můžete například použít k extrakci dat ze zprávy SOAP (v tomto případě binární data v těle zprávy) a vytvoření zprávy SOAP s přílohami. Data jsou zpracována jako součást prvku přílohy objektu SMO (Service Message Object).



Obrázek 69. Zpráva zpracovaná komponentou mediačního toku

Obráceně může komponenta mediačního toku zase transformovat příchozí zprávu extrakcí a zakódováním přílohy a následně přenést zprávu bez příloh.

Místo extrakce dat z příchozí zprávy SOAP za účelem zformování zprávy SOAP s přílohami můžete získat data přílohy z externího zdroje, např. z databáze.



Obrázek 70. Příloha získaná z databáze a přidaná do zprávy SOAP

Obráceně může zase komponenta mediačního toku extrahovat přílohu z příchozí zprávy SOAP a zprávu zpracovat (např. uložit přílohu do databáze).

Neodkazované přílohy mohou být šířeny pouze napříč komponentami mediačního toku. Pokud musí mít k příloze přístup jiný typ komponenty, nebo pokud ji musí jiný typ komponenty šířit, přesuňte přílohu pomocí komponenty mediačního toku do umístění, které je pro danou komponentu přístupné.

Důležité: Jak je popsáno v tématu “Reprezentace XML objektu SMO,” mediační primitivum Mapování transformuje zprávy pomocí transformace XSLT 1.0. Transformace provádí serializaci XML objektu SMO. Mediační primitivum Mapování umožňuje určit kořen serializace a kořenový prvek dokumentu XML tento kořen odráží.

Když odesíláte zprávy SOAP s přílohami, vámi vybraný kořenový prvek určuje, jak budou přílohy šířeny.

- Pokud vyberete jako kořen mapy XML volbu “/body”, budou všechny přílohy šířeny po této mapě standardně.
- Pokud vyberete jako kořen mapy “/”, můžete šíření příloh řídit.

Použití vazby stylu dokumentu WSDL se zprávami o více částech:

Organizace WS-I (Web Services Interoperability Organization) definovala sadu pravidel týkajících se popisování webových služeb prostřednictvím jazyka WSDL a formování odpovídajících zpráv SOAP v zájmu zajištění interoperability.

Tato pravidla jsou určena v profilu interoperability WS-I nazvaném *Basic Profile verze 1.1* (http://www.ws-i.org/Profiles/BasicProfile-1.1.html). Konkrétně je v části R2712 profilu WS-I Basic Profile 1.1 uvedeno: “Vazba document/literal MUSÍ být serializovaná jako OBÁLKA s prvkem těla soap:Body, jehož podřízený prvek je instancí deklarace globálního prvku odkazovaného příslušnou částí zprávy wsdl:message.”

To znamená, že v případě použití vazby SOAP stylu dokumentu pro operaci se zprávami (vstup, výstup nebo porucha) s více definovanými částmi by měla být pouze jedna z těchto částí vázaná na tělo SOAP, aby byla zaručena kompatibilita s profilem WS-I Basic Profile 1.1.

Dále je v části R2941 profilu WS-I Attachments Profile 1.0 uvedeno: "Vazba wsdl:binding v popisu BY MĚLA vázat každou část wsdl:part zprávy wsdl:message v portu wsdl:portType, na který odkazuje, na jeden z atributů soapbind:body, soapbind:header, soapbind:fault , soapbind:headerfault nebo mime:content”.

To znamená, že v případě použití vazby SOAP stylu dokumentu pro operaci se zprávami (vstup, výstup nebo porucha), které mají definovaných více částí, musí být všechny části, s výjimkou části vázané na tělo SOAP, vázány jako přílohy nebo záhlaví.

Následující metoda se v tomto případě používá při generování popisů jazyka WSDL pro exporty s vazbami webové služby (JAX-WS a JAX-RPC):

- Pokud existuje více než jeden prvek, který není binárního typu, můžete si zvolit, jaká část zprávy bude vázaná na tělo zprávy SOAP. Pokud existuje jediný prvek jiného než binárního typu, je tento vázán na tělo zprávy SOAP automaticky.
- V případě vazby webové služby JAX-WS jsou všechny části zprávy jiné než "hexBinary" nebo "base64Binary" vázány jako odkazované přílohy. Viz téma “Odkazované přílohy: části zprávy nejvyšší úrovně” na stránce 82.
- Všechny ostatní části zprávy jsou vázány jako záhlaví SOAP.

Vazby importu JAX-RPC a JAX-WS ctí vazbu SOAP existujícího dokumentu WSDL se zprávami ve stylu dokumentu s více částmi, a to i v případě, že tento dokument neváže více částí k danému tělu SOAP. Nicméně však nelze pro takové dokumenty WSDL generovat v produktu Rational Application Developer klienty webových služeb.

Poznámka: Vazba JAX-RPC nepodporuje přílohy.

V případě použití zpráv o více částech s operací obsahující vazbu SOAP stylu dokumentu proto doporučujeme postupovat takto:

1. Použijte styl document/literal wrapped. V tomto případě mají zprávy vždy pouze jednu část. Nicméně přílohy musí být v tomto případě neodkazované (viz “Neodkazované přílohy” na stránce 87) nebo typu swaRef (viz “Odkazované přílohy: prvky typu swaRef” na stránce 78).
2. Použijte styl RPC/literal. V tomto případě neexistuje pro vazbu WSDL žádné omezení počtu částí vázaných na tělo SOAP. Výsledná zpráva SOAP obsahuje vždy jediný podřízený prvek představující vyvolávanou operaci, přičemž části dané zprávy jsou podřízené prvky tohoto prvku.
3. Pro vazby webové služby JAX-WS je potřeba mít nanejvýš jednu část zprávy, která není typu "hexBinary" nebo "base64Binary", pokud není přijatelné vázat zbývající nebinární části na záhlaví SOAP.
4. Všechny ostatní případy probíhají podle popsaného scénáře.

Poznámka: V případě použití zpráv SOAP, které nejsou kompatibilní s profilem WS-I *Basic Profile verze 1.1*, existují další omezení.

- První část zprávy by neměla být binární.
- Když přijímáte zprávy SOAP ve stylu dokumentu o více částech s odkazovanými přílohami, vazba webové služby JAX-WS očekává, že bude každá odkazovaná příloha reprezentována podřízeným prvkem těla SOAP s hodnotou atributu href identifikující danou přílohu podle jejího ID obsahu. Vazba webové služby JAX-WS odesílá odkazované přílohy pro takové zprávy stejným způsobem. Toto chování není kompatibilní s profilem WS-I Basic Profile.

  Chcete-li zajistit, aby byly vaše zprávy kompatibilní s profilem Basic Profile, postupujte podle metody 1 na stránce 90 nebo 2 na stránce 90 z předchozího seznamu, nebo se vyhněte používání odkazovaných příloh pro takové zprávy a místo toho používejte neodkazované přílohy či přílohy typu swaRef.

Vazby HTTP

Vazba HTTP je navržena tak, aby zajišťovala konektivitu architektury SCA (Service Component Architecture) pro HTTP. V důsledku toho se mohou již existující nebo nově vyvinuté aplikace HTTP podílet na prostředích architektury SOA (Service Oriented Architecture).

Protokol HTTP (Hypertext Transfer Protocol) je široce používaný protokol pro přenos informací na webu. Pro práci s externí aplikací, která využívá protokol HTTP, je nezbytná vazba HTTP. Vazba HTTP transformuje data předaná v

rámci zprávy v nativním formátu obchodnímu objektu v aplikaci SCA. Vazba HTTP může také transformovat data předaná jako obchodní objekt do nativního formátu očekávaného externí aplikací.

Poznámka: Chcete-li umožnit interakci s klienty a službami, které využívají protokol SOAP/HTTP webových služeb, zvažte možnost použití jedné z vazeb webových služeb, které poskytují další funkčnost s ohledem na zpracování kvalit služby standardních pro webové služby.

Některé běžné scénáře použití vazby HTTP jsou popsány v následujícím seznamu:

- Služby s hostitelem SCA mohou vyvolávat aplikace HTTP pomocí importu HTTP.
- Služby s hostitelem SCA se mohou vystavit jako aplikace s povoleným protokolem HTTP, aby je mohli používat klienti HTTP pomocí exportu HTTP.
- Produkty IBM Business Process Manager a Process Server spolu mohou komunikovat přes infrastrukturu HTTP a v důsledku toho mohou uživatelé spravovat svou komunikaci podle firemních standardů.
- Produkty IBM Business Process Manager a Process Server se mohou chovat jako mediátory komunikace HTTP a transformovat a směrovat zprávy, což zlepšuje integraci aplikací pomocí sítě HTTP.
- Produkty IBM Business Process Manager a Process Server lze použít pro přemostění mezi HTTP a ostatními protokoly, např. webovými službami SOAP/HTTP, adaptéry prostředku na bázi JCA (Java Connector Architecture) apod.

Podrobné informace o vytváření vazeb importu a exportu HTTP naleznete v Informačním centru produktu Integration Designer. Viz témata **Vývoj integračních aplikací > Přístup k externím aplikacím s HTTP>**.

Přehled vazeb HTTP:

Vazba HTTP zajišťuje konektivitu k aplikacím s hostitelem HTTP. Zajišťuje mediaci komunikace mezi aplikacemi HTTP a umožňuje volání existujících aplikací na bázi HTTP z modulu.

Vazby importu HTTP

Vazba importu HTTP zajišťuje odchozí konektivitu z aplikací architektury SCA (Service Component Architecture) k serveru nebo aplikacím HTTP.

Import vyvolá adresu URL koncového bodu HTTP. Tuto adresu URL lze určit jedním ze tří způsobů:

- Adresu URL lze nastavit dynamicky v záhlavích HTTP prostřednictvím URL dynamického potlačení.
- Adresu URL lze nastavit dynamicky v prvku adresy cíle SMO.
- Adresu URL lze určit jako konfigurační vlastnost importu.

Toto vyvolání je svým charakterem vždy synchronní.

Ačkoli jsou vyvolání HTTP vždy typu požadavek-odezva, import HTTP podporuje jednosměrné i obousměrné operace a v případě jednosměrné operace odpověď ignoruje.

Vazby exportu HTTP

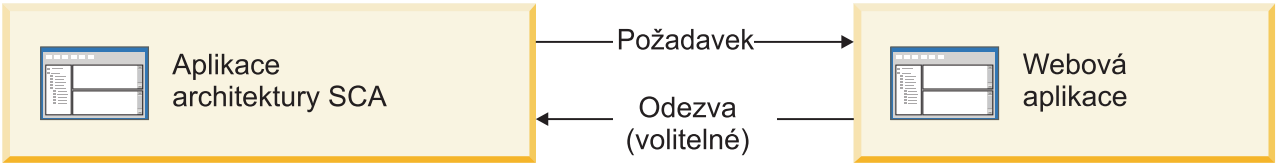
Vazba exportu HTTP zajišťuje příchozí konektivitu z aplikací HTTP do aplikace SCA.

Adresa URL je definována v exportu HTTP. Aplikace HTTP, které chtějí posílat exportu zprávy požadavků, použijí tuto adresu URL k vyvolání exportu.

Export HTTP také podporuje příkazy ping.

Vazby HTTP za běhu

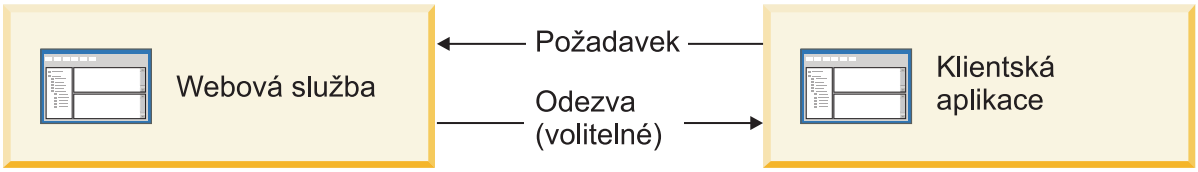
Import s vazbou HTTP za běhu odesílá požadavek s daty nebo bez nich v těle zprávy z aplikace SCA externí webové službě. Požadavek se provádí z aplikace SCA do externí webové služby, jak ukazuje Obrázek 26 na stránce 92.



Obrázek 71. Tok požadavku z aplikace SCA do webové aplikace

Volitelně může import s vazbou HTTP přijímat data od webové aplikace jako odpověď na otázku.

S pomocí exportu je aplikací klienta proveden požadavek vůči webové službě, jak ukazuje Obrázek 27 na stránce 92.



Obrázek 72. Tok požadavku z webové služby do aplikace klienta

Webová služba je webová aplikace spuštěná na serveru. Export je implementován v této webové aplikaci jako servlet, takže klient odesílá svůj požadavek na adresu URL. Servlet předá požadavek aplikaci SCA za běhu.

Volitelně může export v odpovědi na požadavek poslat aplikaci klienta data.

Záhlaví HTTP:

Vazby importu a exportu HTTP umožňují použití konfigurace záhlaví HTTP a jejich hodnot pro odchozí zprávy. Import HTTP využívá tato záhlaví pro požadavky a export HTTP je využívá pro odpovědi.

Staticky nakonfigurovaná záhlaví a řídicí informace mají přednost před hodnotami dynamicky nastavenými za běhu. Řídicí hodnoty Adresy URL dynamického potlačení, Verze a Metody však potlačí statické hodnoty, které jsou jinak považovány za výchozí.

Vazba podporuje dynamický charakter adresy URL importu HTTP určením hodnoty cílové Adresy URL, Verze a Metody HTTP za běhu. Tyto hodnoty jsou určovány extrahováním hodnoty Odkazu na koncový bod, Adresy URL dynamického potlačení, Verze a Metody.

- Jako odkaz na koncový bod využijte rozhraní API com.ibm.websphere.sca.addressing.EndpointReference nebo nastavte pole /headers/SMOHeader/Target/address v záhlaví objektu SMO.
- Jako Adresu URL, Verzi a Metodu pro dynamické přepsání použijte sekci řídicích parametrů HTTP zprávy architektury SCA (Service Component Architecture). Všimněte si, že Adresa URL pro dynamické přepsání má přednost před cílovým odkazem na koncový bod; odkaz na koncový bod však platí napříč vazbami, takže se jedná o upřednostňovaný přístup, který má být použit, kdekoli je to možné.

Řídicí informace a informace o záhlaví pro odchozí zprávy v rámci vazeb exportu a importu HTTP jsou zpracovány v tomto pořadí:

1. Informace o záhlaví a řídicí informace s výjimkou Adresy URL, Verze a Metody pro dynamické přepsání ze zprávy SCA v HTTP (nejnižší priorita).
2. Změny z administrativní konzoly na úrovni exportu/importu.

3. Změny z administrativní konzoly na úrovni metody exportu nebo importu.
4. Cílová adresa určená prostřednictvím odkazu na koncový bod nebo záhlaví objektu SMO.
5. Adresa URL, Verze a Metoda pro dynamické přepsání ze zprávy SCA.
6. Záhlaví a řídicí informace z popisovače dat nebo vázání dat (nejvyšší priorita).

Export a import HTTP naplní záhlaví a řídicí parametry v příchozím směru daty z příchozí zprávy (HTTPExportRequest a HTTPImportResponse) pouze, pokud je šíření záhlaví protokolu nastaveno na **True**. Na druhou stranu, export a import HTTP přečte a zpracuje odchozí záhlaví a řídicí parametry (HTTPExportResponse a HTTPImportRequest) pouze, pokud je šíření záhlaví protokolu nastaveno na **True**.

Poznámka: Změny, které provede popisovač dat nebo vázání dat v záhlavích nebo řídicích parametrech v odpovědi importu nebo požadavku exportu, nezmění pokyny pro zpracování zprávy uvnitř vazby importu nebo exportu a měly by být používány pouze k šíření upravených hodnot do následných komponent SCA.

Služba kontextu je odpovědná za šíření kontextu (včetně záhlaví protokolů, např. záhlaví HTTP, a uživatelského kontextu, např. ID účtu) po cestě vyvolání SCA. Při vývoji v produktu IBM Integration Designer můžete ovládat šíření kontextu prostřednictvím vlastností importu a exportu. Další podrobnosti viz informace o vazbách importu a exportu v Informačním centru produktu IBM Integration Designer.

Dodávané struktury záhlaví HTTP a podpora

Tabulka 23 na stránce 93 rozkládá parametry požadavku/odpovědi pro požadavky a odpovědi importu HTTP a exportu HTTP na položky.

Tabulka 45. Dodávané informace o záhlaví HTTP

Název ovládacího prvku	Požadavek importu HTTP	Odpověď importu HTTP	Požadavek exportu HTTP	Odpověď exportu HTTP
Adresa URL	Ignorováno	Nenastaveno	Načteno ze zprávy požadavku. **Poznámka:** Součástí řídicího parametru adresy URL je i řetězec dotazu.	Ignorováno
Verze (možné hodnoty: 1.0, 1.1; výchozí hodnota je 1.1)	Ignorováno	Nenastaveno	Načteno ze zprávy požadavku	Ignorováno
Metoda	Ignorováno	Nenastaveno	Načteno ze zprávy požadavku	Ignorováno
Adresa URL pro dynamické přepsání	Pokud je nastavena v popisovači dat nebo ve vázání dat, přepíše Adresu importu HTTP. Zapisuje se do zprávy na řádku požadavku. **Poznámka:** Součástí řídicího parametru adresy URL je i řetězec dotazu.	Nenastaveno	Nenastaveno	Ignorováno
Verze pro dynamické přepsání	Je-li nastavena, přepíše Verzi importu HTTP. Zapisuje se do zprávy na řádku požadavku.	Nenastaveno	Nenastaveno	Ignorováno

Tabulka 45. Dodávané informace o záhlaví HTTP (pokračování)

Název ovládacího prvku	Požadavek importu HTTP	Odpověď importu HTTP	Požadavek exportu HTTP	Odpověď exportu HTTP
Metoda pro dynamické přepsání	Je-li nastavena, přepíše Metodu importu HTTP. Zapisuje se do zprávy na řádku požadavku.	Nenastaveno	Nenastaveno	Ignorováno
Typ média (Tento řídicí parametr nese část hodnoty záhlaví HTTP Typ obsahu.)	Je-li uveden, zapisuje se do zprávy jako část záhlaví Typ obsahu. **Poznámka:** Hodnotu tohoto řídicího prvku má poskytovat popisovač dat nebo vázání dat.	Načteno ze zprávy odpovědi, záhlaví Typ obsahu.	Načteno ze zprávy požadavku, záhlaví Typ obsahu.	Je-li uveden, zapisuje se do zprávy jako část záhlaví Typ obsahu. **Poznámka:** Hodnotu tohoto řídicího prvku má poskytovat popisovač dat nebo vázání dat.
Znaková sada (výchozí: UTF-8)	Je-li uvedena, zapisuje se do zprávy jako část záhlaví Typ obsahu. **Poznámka:** Hodnotu tohoto řídicího prvku má poskytovat vázání dat.	Načteno ze zprávy odpovědi, záhlaví Typ obsahu.	Načteno ze zprávy požadavku, záhlaví Typ obsahu.	Podporováno; zapisuje se do zprávy jako část záhlaví Typ obsahu. **Poznámka:** Hodnotu tohoto řídicího prvku má poskytovat vázání dat.
Kódování přenosu (Možné hodnoty: chunked, identity; výchozí je identity)	Je-li uvedeno, zapisuje se do zprávy jako záhlaví a řídí, jak je kódována transformace zprávy.	Načteno ze zprávy odpovědi.	Načteno ze zprávy požadavku.	Je-li uvedeno, zapisuje se do zprávy jako záhlaví a řídí, jak je kódována transformace zprávy.
Kódování přenosu (Možné hodnoty: gzip, x-gzip, deflate, identity; výchozí je identity)	Je-li uvedeno, zapisuje se do zprávy jako záhlaví a řídí, jak je kódován informační obsah.	Načteno ze zprávy odpovědi.	Načteno ze zprávy požadavku.	Je-li uvedeno, zapisuje se do zprávy jako záhlaví a řídí, jak je kódován informační obsah.
Délka obsahu	Ignorováno	Načteno ze zprávy odpovědi.	Načteno ze zprávy požadavku.	Ignorováno
Stavový kód (výchozí: 200)	Nepodporováno	Načteno ze zprávy odpovědi.	Nepodporováno	Je-li uveden, zapisuje se do zprávy na řádku odpovědi.
Fráze příčiny (výchozí: OK)	Nepodporováno	Načteno ze zprávy odpovědi.	Nepodporováno	Řídicí hodnota ignorována. Hodnota na řádku odpovědi zprávy je generována z kódu StatusCode.
Ověření (obsahuje více vlastností)	Je-li uvedeno, používá se ke konstrukci záhlaví Základní ověřování. **Poznámka:** Hodnota tohoto záhlaví bude kódována pouze v protokolu HTTP. V architektuře SCA bude dekódována a předána jako prostý text.	Nelze aplikovat	Načteno ze záhlaví Základní ověřování zprávy požadavku. Přítomnost tohoto záhlaví neznamená, že byl uživatel ověřen. Ověření by mělo být řízeno v konfiguraci servletu. **Poznámka:** Hodnota tohoto záhlaví bude kódována pouze v protokolu HTTP. V architektuře SCA bude dekódována a předána jako prostý text.	Nelze aplikovat

Tabulka 45. Dodávané informace o záhlaví HTTP (pokračování)

Název ovládacího prvku	Požadavek importu HTTP	Odpověď importu HTTP	Požadavek exportu HTTP	Odpověď exportu HTTP
Server proxy (obsahuje více vlastností: Hostitel, Port, Ověření)	Je-li uveden, používá se ke zřízení připojení prostřednictvím serveru proxy.	Nelze aplikovat	Nelze aplikovat	Nelze aplikovat
SSL (obsahuje více vlastností: Úložiště klíčů, Heslo úložiště klíčů, Úložiště údajů o důvěryhodnosti, Heslo úložiště údajů o důvěryhodnosti, Ověření klienta)	Pokud je naplněno a cílová adresa URL je HTTPS, používá se ke zřízení připojení prostřednictvím SSL.	Nelze aplikovat	Nelze aplikovat	Nelze aplikovat

Vázání dat HTTP:

Popisovač dat nebo vázání dat HTTP musí být nakonfigurovány pro všechna různá mapování dat mezi zprávou SCA (Service Component Architecture) a zprávou protokolu HTTP. Popisovače dat poskytují rozhraní nezávislé na vazbě, které umožňuje opakované použití napříč vazbami přenosu a představují doporučený přístup. Vázání dat jsou specifická pro konkrétní vazbu přenosu. Jsou dodávány třídy vázání dat specifické pro HTTP. Také si můžete napsat vlastní popisovače dat nebo datové vazby.

Poznámka: Tři třídy vázání dat HTTP popisované v tomto tématu (HTTPStreamDataBindingSOAP, HTTPStreamDataBindingXML a HTTPServiceGatewayDataBinding) jsou od produktu IBM Business Process Manager verze 7.0 zamítnuté. Místo použití vázání dat popsaných v tomto tématu zvažte použití těchto popisovačů dat:

- Použijte SOAPDataHandler místo HTTPStreamDataBindingSOAP.
- Použijte UTF8XMLDataHandler místo HTTPStreamDataBindingXML.
- Použijte GatewayTextDataHandler místo HTTPServiceGatewayDataBinding.

Jsou poskytnuta vázání dat pro použití s importy HTTP a exporty HTTP: vázání binárních dat, vázání dat XML a vázání dat SOAP. Pro jednosměrné operace není nutné vázání dat odpovědi. Vázání dat představuje název třídy Java, jejíž instance mohou převádět z HTTP na ServiceDataObject i naopak. Selektor funkcí musí být použit na export, který spolu s vazbami metod může určit které vázání dat se použije a která operace je vyvolána. Dodávaná vázání dat jsou tato:

- Vázání binárních dat, která považují tělo za nestrukturovaná binární data. Implementace schématu XSD vázání binárních dat je tato:

```
<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>
```

- Vázání dat XML, která podporují tělo jako data XML. Implementace vázání dat XML je podobná vázání dat JMS XML a nemá žádná omezení schématu rozhraní.
- Vázání dat SOAP, které podporují tělo jako data SOAP. Implementace vázání dat SOAP nemá žádná omezení schématu rozhraní.

Implementace vlastních vázáních dat HTTP

Tento oddíl popisuje, jak implementovat vlastní vázání dat HTTP.

Poznámka: Doporučený přístup je implementovat vlastní popisovač dat, protože ten lze používat opakovaně pro různé přenosové vazby.

Primárním rozhraním pro zpracování vlastních zpráv HTTP je HTTPStreamDataBinding. Toto rozhraní je nevrženo tak, aby umožňovalo zpracování velkých objemů informačního obsahu. Aby však taková implementace fungovala, musí toto vázání dat před zápisem zprávy do proudu vracet řídicí informace a záhlaví.

Vlastní vázání dat musí implementovat níže uvedené metody a jejich pořadí.

Chcete-li upravit vázání dat, napište třídu, která implementuje HTTPStreamDataBinding. Vázání dat by mělo mít čtyři soukromé vlastnosti:

- private DataObject pDataObject
- private HTTPControl pCtrl
- private HTTPHeaders pHeaders
- private yourNativeDataType nativeData

Vazba HTTP vyvolá upravené vázání dat v tomto pořadí:

- Odchozí zpracování (DataObject na Nativní formát):
  1. setDataObject(...)
  2. setHeaders(...)
  3. setControlParameters(...)
  4. setBusinessException(...)
  5. convertToNativeData()
  6. getControlParameters()
  7. getHeaders()
  8. write(...)
- Zpracování příchozích požadavků (Nativní formát na DataObject):
  1. setControlParameters(...)
  2. setHeaders(...)
  3. convertFromNativeData(...)
  4. isBusinessException()
  5. getDataObject()
  6. getControlParameters()
  7. getHeaders()

Potřebujete vyvolat setDataObject(...) v metodě convertFromNativeData(...) pro nastavení hodnoty objektu dataObject, který je převeden z nativních dat na soukromou vlastnost "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;
}
/*
* Přidejte záhlaví http "IsBusinessException" do pHeaders.
* Dva kroky:
* 1. Nejprve odebrat všechna záhlaví s názvem IsBusinessException (bez rozlišení malých a velkých písmen).
*   Tím se zajistí, že bude přítomné pouze jedno záhlaví.
* 2. Přidat nové záhlaví "IsBusinessException"
*/
public void setBusinessException(boolean isBusinessException) {
  //odebrat všechna záhlaví s názvem IsBusinessException (bez rozlišení malých a velkých písmen).
  //Tím se zajistí, že bude přítomné pouze jedno záhlaví.
  //přidat nové záhlaví "IsBusinessException", příklad kódu:
  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;
}
/*
* Získat záhlaví "IsBusinessException" z pHeaders, vrátit jeho logickou hodnotu
*/
public boolean isBusinessException() {
  String headerValue = getHeaderValue(pHeaders,"IsBusinessException");
  boolean result=Boolean.parseBoolean(headerValue);
  return result;
}
public void convertToNativeData() throws DataBindingException {
  DataObject dataObject = getDataObject();
  this.nativeData=realConvertWorkFromSDOToNativeData(dataObject);
}
public void convertFromNativeData(HTTPInputStream arg0){
  //Metoda vyvinutá zákazníkem, která má
  //Číst data z proudu HTTPInputStream
  //Převést je na DataObject
  DataObject dataobject=realConvertWorkFromNativeDataToSDO(arg0);
  setDataObject(dataobject);
}
public void write(HTTPOutputStream output) throws IOException {
  if (nativeData != null)
  output.write(nativeData);
}
```

Vázání EJB

Vazby importu Enterprise JavaBeans (EJB) umožňují komponentám architektury SCA (Service Component Architecture) vyvolávat služby poskytované obchodní logikou Java EE spuštěnou na serveru Java EE. Vazby exportu EJB umožňují vystavení komponent SCA jako objektů Enterprise JavaBean, takže může obchodní logika Java EE vyvolávat komponenty SCA, které by pro ni jinak byly nedostupné.

Vazby importu EJB:

Vazby importu EJB umožňují modulu SCA volat implementace EJB určením způsobu, jak je přijímající modul vázán k externímu objektu EJB. Import služeb z externí implementace EJB umožňuje uživatelům zapojit obchodní logiku do prostředí IBM Business Process Manager a podílet se na obchodním procesu.

Vazby importu EJB lze vytvářet v produktu Integration Designer. Vazby můžete vygenerovat některou z těchto procedur:

- Vytvoření importu EJB pomocí průvodce externí službou

  Pomocí průvodce externí službou v produktu Integration Designer můžete vytvořit import založený na již existující implementaci. Průvodce externí službou vytváří služby založené na vámi zadaných kritériích. Poté vygeneruje obchodní objekty, rozhraní a soubory importu na základě zjištěných služeb.
- Vytvoření importu EJB pomocí editoru sestavení

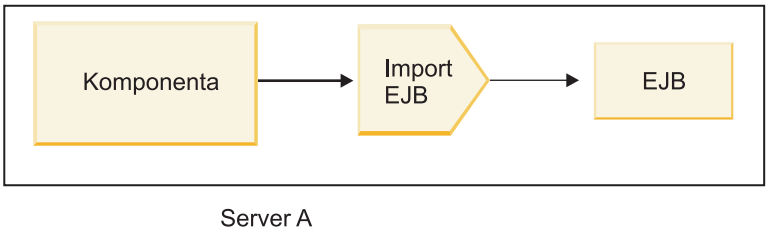
  Import EJB můžete vytvořit v rámci diagramu sestavení v editoru sestavení produktu Integration Designer. Z palety můžete buď použít Import, nebo vytvořit vázání EJB pomocí třídy Java.

Vygenerovaný import má vázání dat, která vytvoří připojení Java-WSDL, místo aby vyžadovala komponentu mostu Java. Komponentu s odkazem WSDL (Web Services Description Language) můžete spojit přímo s importem EJB, který komunikuje se službou založenou na EJB prostřednictvím rozhraní Java.

Import EJB může vstupovat do interakcí s obchodní logikou Java EE buď prostřednictvím programovacího modelu EJB 2.1, nebo prostřednictvím programovacího modelu EJB 3.0.

Vyvolání obchodní logiky Java EE může být lokální (pouze pro EJB 3.0) nebo vzdálené.

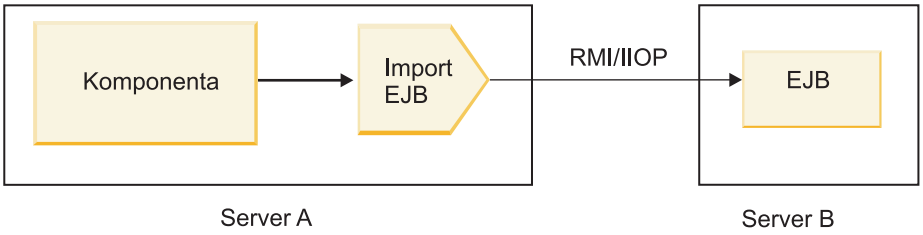
- Lokální vyvolání se používá, když chcete volat obchodní logiku Java EE, která se nachází na stejném serveru jako import.



Obrázek 73. Lokální vyvolání EJB (pouze u EJB 3.0)

- Vzdálené vyvolání se používá, když chcete volat obchodní logiku Java EE, která se nenachází na stejném serveru jako import.

  Například na následujícím obrázku využívá import EJB vyvolání RMI/IIOP (Remote Method Invocation over Internet InterORB Protocol) k vyvolání metody EJB na jiném serveru.



Obrázek 74. Vzdálené vyvolání EJB

Při konfiguraci vázání EJB používá produkt Integration Designer název rozhraní JNDI k určení úrovně programovacího modelu EJB a typu vyvolání (lokální nebo vzdálené).

Vazby importu EJB obsahují tyto hlavní komponenty:

- Popisovač dat JAX-WS.
- Selektor poruch EJB.
- Selektor funkcí importu EJB.

Pokud váš uživatelský scénář není založen na mapování JAX-WS, budete možná potřebovat vlastní popisovač dat, selektor funkcí a selektor poruch, aby prováděly úlohy jinak prováděné komponentami, jež jsou součástí vazeb importu EJB. To zahrnuje i mapování normálně prováděné vlastním algoritmem mapování.

Vazby exportu EJB:

Externí aplikace Java EE mohou vyvolat komponentu SCA prostřednictvím vazby exportu EJB. Použití exportu EJB umožňuje vystavit komponenty SCA tak, aby je mohly externí aplikace Java EE vyvolávat pomocí programovacího modelu EJB.

Poznámka: Export EJB je bezstavový objekt typu bean.

Vázání EJB můžete vytvářet v produktu Integration Designer. Vazby můžete vygenerovat některou z těchto procedur:

- Vytvoření vazeb exportu EJB pomocí průvodce externí službou

  Pomocí průvodce externí službou v produktu Integration Designer můžete vytvořit službu exportu založenou na již existující implementaci. Průvodce externí službou vytváří služby založené na vámi zadaných kritériích. Potom vygeneruje obchodní objekty, rozhraní a soubory exportu založené na zjištěných službách.
- Vytvoření vazeb exportu EJB pomocí editoru sestavení

  Export EJB můžete vytvořit v editoru sestavení produktu Integration Designer.

Důležité: Klient J2SE (Java 2 Platform, Standard Edition) nemůže vyvolat klienta exportu EJB, který je vygenerován v produktu Integration Designer.

Vazbu můžete vygenerovat z již existující komponenty SCA, nebo můžete vygenerovat export s vázáním EJB pro rozhraní Java.

- Při generování exportu pro již existující komponentu SCA, která má existující rozhraní WSDL, je exportu přiřazeno rozhraní Java.
- Při generování exportu pro rozhraní Java můžete pro tento export vybrat buď rozhraní WSDL, nebo Java.

  Poznámka: Rozhraní Java použité k vytvoření exportu EJB má následující omezení s ohledem na objekty (vstupní a výstupní parametry a výjimky) předávané jako parametry v rámci vzdáleného volání:
  - Musí být konkrétního typu (místo rozhraní nebo abstraktního typu).
  - Musí odpovídat specifikaci Enterprise JavaBeans. Musí být serializovatelné a mít výchozí konstruktor bez argumentů a všechny vlastnosti musí být přístupné prostřednictvím metody getter a modulu nastavení.

    Informace o specifikaci Enterprise JavaBeans viz web společnosti Sun Microsystems, Inc. na adrese http://java.sun.com.

  Navíc musí být výjimka kontrolovaná výjimka, zděděná od java.lang.Exception a musí být singulární (tzn. nepodporuje vracení různých typů kontrolovaných výjimek).

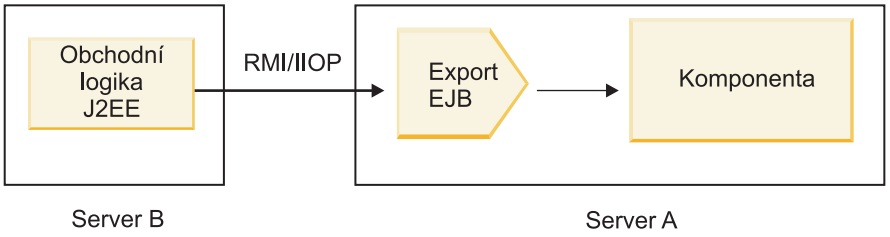
  Všimněte si také, že obchodní rozhraní objektu Java EnterpriseBean je prosté rozhraní Java a nesmí rozšiřovat javax.ejb.EJBObject ani javax.ejb.EJBLocalObject. Metody obchodního rozhraní by neměly vracet výjimku java.rmi.Remote.Exception.

Vazby exportu EJB mohou vstupovat do interakcí s obchodní logikou Java EE s pomocí programovacího modelu EJB 2.1 nebo programovacího modelu EJB 3.0.

Vyvolání může být lokální (pouze pro EJB 3.0) nebo vzdálené.

- Lokální vyvolání se používá, když obchodní logika Java EE volá komponentu SCA, která se nachází na stejném serveru jako export.
- Vzdálené vyvolání se používá, když se obchodní logika Java EE nenachází na stejném serveru jako export.

  Například na následujícím obrázku EJB pomocí RMI/IIOP volá komponentu SCA na jiném serveru.



Obrázek 75. Vzdálené volání z klienta do komponenty SCA prostřednictvím exportu EJB

Při konfiguraci vázání EJB používá produkt Integration Designer název rozhraní JNDI k určení úrovně programovacího modelu EJB a typu vyvolání (lokální nebo vzdálené).

Vazby exportu EJB obsahují tyto hlavní komponenty:

- Popisovač dat JAX-WS.
- Selektor funkcí exportu EJB.

Pokud váš uživatelský scénář není založen na mapování JAX-WS, budete možná potřebovat vlastní popisovač dat a selektor funkcí, aby prováděly úlohy jinak prováděné komponentami, jež jsou součástí vazeb exportu EJB. To zahrnuje i mapování normálně prováděné vlastním algoritmem mapování.

Vlastnosti vázání EJB:

Vazby importu EJB používají své nakonfigurované názvy rozhraní JNDI k určení úrovně programovacího modelu EJB a typu vyvolání (lokální nebo vzdálené). Vazby importu a exportu EJB používají k transformaci dat popisovač dat JAX-WS. Vazba importu EJB využívá selektor funkcí importu EJB a selektor poruch EJB a vazba exportu EJB využívá selektor funkcí exportu EJB.

Názvy rozhraní JNDI a vazby importu EJB:

Při konfiguraci vázání EJB na importu používá produkt Integration Designer název rozhraní JNDI k určení úrovně programovacího modelu EJB a typu vyvolání (lokální nebo vzdálené).

Pokud není určen žádný název rozhraní JNDI, použije se výchozí vazba rozhraní EJB. Výchozí vytvořené názvy závisí na tom, zda vyvoláváte objekty JavaBeans EJB 2.1 nebo objekty JavaBeans EJB 3.1.

Poznámka: Podrobnější informace o konvencích pojmenování viz téma "Přehled vazeb aplikací EJB 3.0" v Informačním centru produktu WebSphere Application Server. .

- JavaBeans EJB 2.1

  Výchozím názvem rozhraní JNDI předvoleného produktem Integration Designer je výchozí vázání EJB 2.1, která má formu **ejb/** plus domovské rozhraní, oddělené lomítky.

  Například pro domovské rozhraní objektů JavaBeans EJB 2.1 pro com.mycompany.myremotebusinesshome je výchozí vazba:

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

  Pro EJB 2.1 je podporováno pouze vzdálené vyvolání EJB.
- JavaBeans EJB 3.0

Výchozím názvem rozhraní JNDI předvoleného produktem Integration Designer pro lokální rozhraní JNDI je úplný název třídy lokálního rozhraní, jemuž předchází **ejblocal:**. Například pro úplné rozhraní lokálního objektu interface com.mycompany.mylocalbusiness je předvolené rozhraní JNDI EJB 3.0:

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

Pro vzdálený objekt interface com.mycompany.myremotebusiness je předvoleným rozhraním JNDI EJB 3.0 úplné rozhraní:

```
com.mycompany.myremotebusiness
```

Výchozí vazby aplikací EJB 3.0 jsou popsány na tomto místě: Přehled vazeb aplikací EJB 3.0.

Produkt Integration Designer použije jako výchozí umístění rozhraní JNDI pro objekty EJB s použitím programovacího modelu verze 3.0 "zkrácený" název.

Poznámka: Pokud se odkaz na implementované rozhraní JNDI cílového objektu EJB liší od výchozího umístění vazby rozhraní JNDI, protože bylo použito nebo nakonfigurováno vlastní mapování, musí být řádně určen název cílového rozhraní JNDI. Název můžete určit v produktu Integration Designer před implementací, nebo pro vazbu importu, můžete název změnit v administrativní konzole (po implementaci) tak, aby odpovídal názvu rozhraní JNDI cílového objektu EJB.

Další informace o tvorbě vázání EJB viz Práce s vazbami EJB v Informačním centru produktu Integration Designer.

Popisovač dat JAX-WS:

Vazba importu EJB (Enterprise JavaBeans) využívá popisovač dat JAX-WS k transformaci obchodních objektů požadavku na parametry objektů Java a k transformaci návratové hodnoty objektu Java na obchodní objekt odpovědi. Vazba exportu EJB využívá popisovač dat JAX-WS k transformaci objektů EJB požadavku na obchodní objekty požadavku a k transformaci obchodního objektu odpovědi na návratovou hodnotu.

Tento popisovač dat mapuje data z rozhraní WSDL specifikovaného v SCA na rozhraní Java cílového objektu EJB (a naopak) s využitím specifikace JAX-WS (Java API for XML Web Services) a specifikace JAXB (Java Architecture for XML Binding).

Poznámka: Aktuální podpora je omezena na specifikace JAX-WS 2.1.1 a JAXB 2.1.3.

Popisovač dat určený na úrovni vázání EJB se používá ke zpracování požadavků, odpovědí, poruch a výjimek za běhu.

Poznámka: V případě poruch je možné pro každou poruchu určit specifický popisovač dat prostřednictvím konfigurační vlastnosti faultBindingType. To přepíše hodnotu určenou na úrovni vázání EJB.

Když má vázání EJB rozhraní WSDL, použije se standardně popisovač dat JAX-WS. Tento popisovač dat nelze použít k transformaci zprávy SOAP představující vyvolání JAX-WS na datový objekt.

Vazba importu EJB používá k transformaci datového objektu na pole objektů Java (Object[]) popisovač dat. Při odchozí komunikaci dochází k tomuto zpracování:

1. Vázání EJB nastaví očekávaný typ, očekávaný prvek a název cílové metody v kontextu BindingContext tak, aby odpovídaly hodnotám určeným ve WSDL.
2. Vázání EJB vyvolá metodu transformace pro datový objekt vyžadující transformaci dat.
3. Popisovač dat vrátí pole Object[] představující parametry dané metody (v pořadí, v němž jsou v rámci metody definovány).
4. Vázání EJB použije pole Object[] k vyvolání metody na cílovém rozhraní EJB.

Vazba také připraví pole Object[] ke zpracování odpovědi od vyvolání EJB.

- Prvním prvkem v poli Object[] je návratová hodnota z vyvolání metody Java.
- Další hodnoty představují vstupní parametry metody.

To je nezbytné pro podporu parametrů typu Vstupní/Výstupní a Výstupní.

Pro parametry typu Výstupní se musí hodnoty vracet v datovém objektu odpovědi.

Popisovač dat zpracovává a transformuje hodnoty nalezení v poli Object[] a poté vrací odpověď datovému objektu.

Popisovač dat podporuje xs:AnyType, xs:AnySimpleType a xs:Any spolu s dalšími datovými typy XSD. Chcete-li povolit podporu pro xs:Any, jako vlastnost objektu JavaBeans v kódu Java použijte anotaci **@XmlAnyElement (lax=true)**, jak ukazuje následující příklad:

```
public class TestType {
 private Object[] object;

 @XmlAnyElement (lax=true)
 public Object[] getObject() {
  return object;
 )

 public void setObject (Object[] object) {
  this.object=object;
 )
)
```

Tím se z objektu vlastnosti v typu TestType stane pole xs:any. Hodnota třídy Java použitá v poli xs:any by měla mít anotaci **@XmlAnyElement**. Pokud je například Adresa třída Java používaná k naplnění pole objektů, měla by mít třída Adresa anotaci **@XmlRootElement**.

Poznámka: Chcete-li upravit mapování z typu XSD na typy Java definované specifikací JAX-WS, změňte anotace JAXB tak, aby odpovídaly vašim obchodním potřebám. Popisovač dat JAX-WS podporuje xs:any, xs:anyType a xs:anySimpleType.

Pro popisovač dat JAX-WS platí tato omezení:

- Popisovač dat nezahrnují podporu pro anotaci **@WebParam** atributu záhlaví.
- Obor názvů pro soubory schémat obchodních objektů (soubory XSD) nezahrnuje výchozí mapování z názvu balíku Java. Anotace **@XMLSchema** v objektu package-info.java také nefunguje. Jediný způsob, jak vytvořit XSD s oborem názvů, je použít anotace **@XmlType** a **@XmlRootElement**. Anotace **@XmlRootElement** definuje cílový obor názvů pro globální prvek v typech JavaBeans.
- Průvodce importem EJB nevytváří soubory XSD pro nesouvisející třídy. Verze 2.0 nepodporuje anotaci **@XmlSeeAlso**, takže pokud se na podřízenou třídu neodkazuje přímo z nadřízené třídy, XSD se nevytvoří. Tento problém lze vyřešit spuštěním obslužného programu SchemaGen pro takové podřízené třídy.

  SchemaGen je obslužný program příkazového řádku (umístěný v adresáři *domovský_adresář_instalace_WPS*/bin) určený pro vytvoření souborů XSD pro daný objekt typu bean. Aby toto řešení fungovalo, musí být tyto soubory XSD ručně zkopírovány do daného modulu.

Selektor poruch EJB:

Selektor poruch EJB určuje, zda je výsledkem vyvolání EJB porucha, výjimka za běhu nebo úspěšná odpověď.

Pokud je zjištěna porucha, vrátí selektor poruch EJB běhovému prostředí vazby nativní název poruchy, takže může popisovač dat JAX-WS převést objekt výjimky na obchodní objekt poruchy.

Při úspěšné (nechybové) odpovědi vazba importu EJB sestaví pole objektů Java (Object[]) pro vracení hodnot.

- Prvním prvkem v poli Object[] je návratová hodnota z vyvolání metody Java.
- Další hodnoty představují vstupní parametry metody.

To je nezbytné pro podporu parametrů typu Vstupní/Výstupní a Výstupní.

Ve scénářích výjimek vazba sestaví pole Object[] a první prvek představuje výjimku vrácenou metodou.

Selektor poruch může vrátit kteroukoli z následujících hodnot:

Tabulka 46. Návratové hodnoty

Typ	Návratová hodnota	Popis
Porucha	**ResponseType.FAULT**	Vrácena, když předané pole Object[] obsahuje objekt výjimky.
Výjimka za běhu	**ResponseType.RUNTIME**	Vrácena, když objekt výjimky neodpovídá žádnému deklarovanému typu výjimek metody.
Normální odpověď	**ResponseType.RESPONSE**	Vracena ve všech ostatních případech.

Pokud selektor poruch vrátí hodnotu **ResponseType.FAULT**, je vrácen nativní název poruchy. Tento nativní název poruchy používá vazba k určení odpovídajícího názvu poruchy WSDL z modelu a k vyvolání správného popisovače dat způsobujících poruchu.

Selektor funkcí EJB:

Vázání EJB využívá k určení metody EJB, která se má volat, selektor funkcí importu (pro zpracování odchozích požadavků) nebo selektor funkcí exportu (pro zpracování příchozích požadavků).

Selektor funkcí importu

Pro zpracování odchozích požadavků selektor funkcí importu odvodí typ metody EJB na základě názvu operace vyvolané komponentou SCA, která je spojena s importem EJB. Selektor funkcí hledá anotaci @WebMethod u třídy Java generované produktem Integration Designer a anotované podle specifikace JAX-WS, aby určil název přidružené cílové operace.

- Pokud je přítomna anotace @WebMethod, použije ji selektor funkcí k určení správného mapování metody Java pro metodu WSDL.
- Pokud anotace @WebMethod chybí, selektor funkcí předpokládá, že je název metody Java stejný jako název vyvolané operace.

Poznámka: Tento selektor funkcí je platný pouze pro rozhraní typu WSDL importu EJB, nikoli pro rozhraní typu Java importu EJB.

Selektor funkcí vrací objekt java.lang.reflect.Method, který představuje metodu rozhraní EJB.

Selektor funkcí používá pole objektů Java (Object[]), který obsahuje odpověď cílové metody. Prvním prvkem pole Object[] je metoda Java s názvem WSDL a druhým prvkem pole Object[] je vstupní obchodní objekt.

Selektor funkcí exportu

Pro zpracování příchozích požadavků selektor funkcí exportu odvodí cílovou metodu, která má být vyvolána, z metody Java.

Selektor funkcí exportu mapuje název operace Java vyvolané klientem EJB na název operace v rozhraní cílové komponenty. Název metody je vracen jako řetězec a je řešen běhovým prostředím SCA v závislosti na typu rozhraní cílové komponenty.

Vazby EIS

Vazby podnikových informačních systémů (EIS) zajišťují konektivitu mezi komponentami SCA a externím podnikovým informačním systémem. Této komunikace je dosaženo prostřednictvím exportů EIS a importů EIS, které podporují adaptéry prostředku JCA 1.5 a adaptéry WebSphere Adapter.

Vaše komponenty SCA mohou vyžadovat převod dat do nebo z externího podnikového informačního systému. Do vytváření modulu SCA, který vyžaduje takovou konektivitu, zahrnete (kromě své komponenty SCA) import nebo export s vazbou EIS pro komunikaci se specifickým externím podnikovým informačním systémem.

Adaptéry prostředku v produktu IBM Integration Designer se používají v rámci kontextu importu nebo exportu. Pomocí průvodce externí službou vyvinete import či export a při vývoji do něj zahrnete adaptér prostředku. Import EIS, který umožní vaší aplikaci vyvolat službu v systému EIS, nebo export EIS, který aplikaci umožní v systému EIS vyvolat službu vyvinutou v produktu IBM Integration Designer, jsou vytvořeny s konkrétním adaptérem prostředku. Například byste vytvořili import s adaptérem JD Edwards pro vyvolání služby v systému JD Edwards.

Když použijete průvodce externí službou, vytvoří informace o vazbě EIS za vás. Informace o vazbách můžete také přidávat a upravovat pomocí jiného nástroje, editoru sestavení. Další informace viz Přístup k externím službám pomocí adaptérů.

Po implementaci modulu SCA, který obsahuje danou vazbu EIS, na serveru můžete v administrativní konzole zobrazit informace o této vazbě nebo vazbu nakonfigurovat.

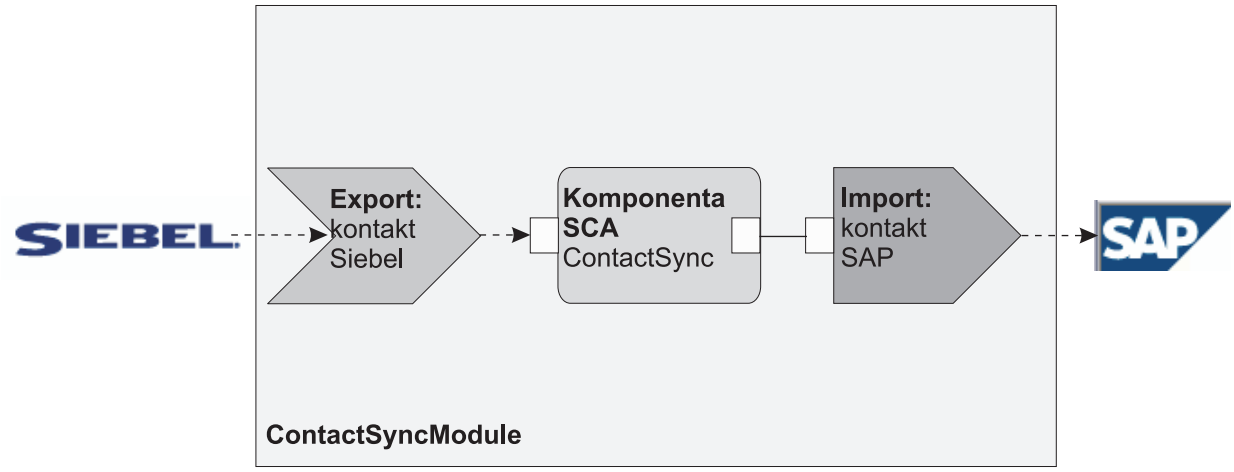
Přehled vazeb EIS:

Vazba EIS (podnikového informačního systému), je-li použita s adaptérem prostředku JCA, umožňuje přístup ke službám v podnikovém informačním systému nebo tomuto systému povolí vaše služby.

Následující příklad ukazuje, jak modul SCA nazvaný ContactSyncModule synchronizuje kontaktní informace mezi systémem Siebel a systémem SAP.

1. Komponenta SCA nazvaná ContactSync naslouchá (prostřednictvím exportu aplikace EIS nazvaného Kontakt Siebel) změnám v kontaktech systému Siebel.
2. Sama komponenta SCA ContactSync využívá aplikaci SAP (prostřednictvím importu aplikace EIS), aby příslušným způsobem aktualizovala kontaktní informace SAP.

Protože se datové struktury používané pro ukládání kontaktů v systémech Siebel a SAP liší, musí komponenta SCA ContactSync zajistit mapování.



Obrázek 76. Tok ze systému Siebel do systému SAP

Export Kontakt Siebel a import Kontakt SAP mají nakonfigurovány příslušné adaptéry prostředku.

Klíčové funkce vázání EIS:

Import EIS je importem architektury SCA, který umožňuje komponentám v modulu SCA využívat aplikace EIS definované mimo modul SCA. Data importu EIS jsou používána k přenosu dat z komponenty SCA na externí EIS. Export EIS se používá k přenosu dat z externího EIS do modulu SCA.

Importy

Rolí importu EIS je přemostit mezeru mezi komponentami SCA a externími systémy EIS. Externí aplikace lze považovat za import EIS. V tom případě import EIS zasílá data na externí EIS a volitelně přijímá data jako odpověď.

Import EIS poskytuje komponenty SCA s jednotným pohledem aplikací mimo modul. To komponentám umožňuje komunikovat s externím EIS, jako je např. SAP, Siebel nebo PeopleSoft, pomocí konzistentního modelu SCA.

Na straně klienta importu existuje rozhraní vystavené aplikací importu EIS s jednou nebo dvěma metodami, z nichž každá přebírá objekty dat jako argumenty a vrací hodnoty. Na straně implementace existuje rozhraní Common Client Interface (CCI), které je implementováno adaptérem prostředku.

Implementace běhového prostředí importu EIS spojuje klientské rozhraní a CCI. Import mapuje vyvolání metody na rozhraní do vyvolání na CCI.

Vazby jsou vytvářeny na třech úrovních: vazba rozhraní, která poté používá vazby obsažené metody následně využívajícími vázání dat.

Vazba rozhraní propojuje rozhraní importu s připojením k systému EIS poskytujícímu aplikaci. To odráží fakt, že sada aplikací, představovaných rozhraním, je poskytnuta specifickou instancí systému EIS a přístup k této instanci zajišťuje připojení. Prvek vazby obsahuje vlastnosti s dostatečným počtem informací k vytvoření připojení (tyto vlastnosti jsou součástí instance javax.resource.spi.ManagedConnectionFactory).

Vazba metody přidružuje metodu se specifickou interakcí k systému EIS. Pro JCA je interakce charakterizována sadou vlastností implementace rozhraní javax.resource.cci.InteractionSpec. Prvek interakce vazby metody obsahuje tyto vlastnosti, společně s názvem třídy, díky čemuž poskytuje dostatek informací k provedení interakce. Vazba metody využívá vázání dat popisující mapování argumentu a výsledku metody rozhraní na reprezentaci EIS.

Scénář běhového prostředí pro import EIS:

1. Metoda na rozhraní importu je vyvolána pomocí programovacího modelu SCA.
2. Požadavek dosahující importu EIS obsahuje název metody a její argumenty.
3. Import nejprve vytvoří implementaci vazby rozhraní. Poté pomocí dat z vazby importu vytvoří nástroj ConnectionFactory a obě položky přidruží. To znamená, že import volá položku setConnectionFactory na vazbě rozhraní.
4. Je vytvořena implementace vazby metody odpovídající vyvolané metodě.
5. Vytvořena a naplněna je instance javax.resource.cci.InteractionSpec. Poté je vázání dat použito k vázání argumentů metod k formátu, kterému porozumí adaptér prostředku.
6. Rozhraní CCI se používá k provedení interakce.
7. Když se vrátí volání, použije se vázání dat k vytvoření výsledku vyvolání a výsledek je vrácen volajícímu.

Exporty

Rolí exportu EIS je přemostit mezeru mezi komponentou SCA a externím systémem EIS. Externí aplikace lze považovat za export EIS. V tomto případě externí aplikace odesílá svá data ve formě periodických oznámení. O exportu EIS lze uvažovat jako o aplikaci předplatného naslouchající externímu požadavku ze strany EIS. Komponenta SCA, která využívá export EIS, jej zobrazuje jako lokální aplikaci.

Export EIS poskytuje komponenty SCA s jednotným pohledem aplikací mimo modul. To komponentám umožňuje komunikovat s EIS, jako je např. SAP, Siebel nebo PeopleSoft, pomocí konzistentního modelu SCA.

Export představuje implementaci listeneru přijímající požadavky z EIS. Listener implementuje rozhraní listeneru specifické pro adaptér prostředku. Export rovněž obsahuje rozhraní implementující komponenty, které je systému EIS vystaveno prostřednictvím exportu.

Implementace běhového prostředí exportu EIS spojuje listener a rozhraní implementující komponenty. Mapy exportu požadavku EIS na vyvolání příslušné operace na komponentě. Vazby jsou vytvářeny na třech úrovních: vázání listeneru, které následně využívá vázání nativní metodou následně využívající vázání dat.

Vázání listeneru vztahuje listener přijímající požadavek ke komponentě vystavené prostřednictvím exportu. Definice exportu obsahuje název komponenty. Běhové prostředí ji vyhledá a předá jí požadavky.

Vázání nativní metody přidružuje nativní metodu nebo typ události přijatý listenerem k operaci implementované komponentou vystavenou pomocí exportu. Mezi metodou vyvolanou listenerem a typem události není žádný vztah. Všechny události přicházejí prostřednictvím jedné nebo více metod listeneru. Vázání nativní metody využívá selektor funkcí definovaný v exportu k extrakci názvu nativní metody z příchozích dat a vázání dat využívá k vazbě formátu dat systému EIS na formát, kterému porozumí komponenta.

Scénář běhového prostředí pro export EIS:

1. Vyvolání spouštěčů požadavku EIS metody na implementaci listeneru.
2. Listener vyhledává a vyvolává export, přičemž mu předává všechny argumenty vyvolání.
3. Export vytváří implementaci vázání listeneru.
4. Export dokládá příkladem selektor funkcí a nastavuje jej na vázání listeneru.
5. Export inicializuje vázání nativní metody a přidává je k vázání listeneru. Pro všechna vázání nativní metody jsou rovněž inicializována vázání dat.
6. Export vyvolává vázání listeneru.
7. Vázání listeneru vyhledává exportované komponenty a využívá selektoru funkcí k načtení názvu nativní metody.
8. Tento název se používá k vyhledání vázání nativní metody, které následně vyvolává cílovou komponentu.

Styl interakce adaptéru umožňuje vázání exportu EIS vyvolat cílovou komponentu asynchronně (výchozí nastavení) nebo synchronně.

Adaptéry prostředku

Import či export vyvinete pomocí průvodce externí službou a během vývoje můžete zahrnout adaptér prostředku. Adaptéry, které jsou k dispozici s nástrojem IBM Integration Designer a jež se používají pro přístup k systémům CICS, IMS, JD Edwards, PeopleSoft, SAP a Siebel, jsou určeny výhradně pro vývoj a testování. To znamená, že je můžete využít pouze k vývoji a testování svých aplikací.

Jakmile aplikaci implementujete, budete ke spuštění aplikace potřebovat licencované adaptéry běhového prostředí. Když však službu sestavujete, můžete k ní vložit adaptér. Váš způsob licencování adaptérů vám může umožnit použití vloženého adaptéru jako licencovaného adaptéru běhového prostředí. Tyto adaptéry vyhovují architektuře Java EE Connector Architecture (JCA 1.5). Otevřený standard JCA je standardem Java EE pro konektivitu systému EIS. Architektura JCA poskytuje spravovaný rámec. To znamená, že nástroj Quality of Service (QoS) je poskytován aplikačním serverem, který pro transakce nabízí správu životního cyklu a zabezpečení. Rovněž je kompatibilní se specifikacemi Enterprise Metadata Discovery s výjimkou nástrojů IBM CICS ECI Resource Adapter a IBM IMS Connector for Java.

Adaptéry WebSphere Business Integration Adapters, starší sada adaptérů, jsou průvodcem rovněž podporovány.

Prostředky Java EE

Modul EIS, modul architektury SCA následující vzor modulu EIS, je možné implementovat na platformu Java EE.

Implementace modulu EIS na platformu Java EE má za následek aplikaci, která je připravena k provedení, zabalena jako souboru EAR a implementována na server. Vytvořeny jsou všechny prostředky a artefakty Java EE. Aplikace je konfigurována a připravena ke spuštění.

Dynamické vlastnosti JCA Interaction Spec a Connection Spec:

Vazba EIS může přijímat vstup pro vlastnosti InteractionSpec a ConnectionSpec určené pomocí dobře definovaného podřízeného datového objektu, který doprovází informační obsah. To umožňuje dynamické interakce typu požadavek-odezva s adaptérem prostředku prostřednictvím vlastnosti InteractionSpec a ověření komponenty prostřednictvím vlastnosti ConnectionSpec.

Vlastnost javax.cci.InteractionSpec nese informace o tom, jak zpracovat požadavek na interakci s adaptérem prostředku. Také může nést informace o tom, jak bylo dosaženo interakce po daném požadavku. Tato obousměrná komunikace prostřednictvím interakcí je někdy označována jako *konverzace*.

Vazba EIS očekává informační obsah, který bude argumentem pro adaptér prostředku, který má obsahovat podřízený datový objekt nazvaný **properties**. Tento datový objekt vlastnosti bude obsahovat dvojice název-hodnota s názvem vlastností Interaction Spec ve specifickém formátu. Pravidla formátování jsou tato:

- Názvy musí začínat předponou **IS**, následovanou názvem vlastnosti. Například specifikace interakce s vlastností JavaBeans nazvanou **InteractionId** by měla název vlastnosti **ISInteractionId**.
- Dvojice název-hodnota představuje název a hodnotu jednoduchého typu vlastnosti Interaction Spec.

V tomto příkladě rozhraní specifikuje, že je vstupem operace datový objekt **Account**. Toto rozhraní vyvolává aplikaci vazby importu EIS se záměrem odeslat a přijmout dynamickou vlastnost InteractionSpec nazvanou **workingSet** s hodnotou **xyz**.

Obchodní graf nebo obchodní objekty na serveru obsahují základní obchodní objekt **properties**, který umožňuje odesílání dat specifických pro určitý protokol s informačním obsahem. Tento obchodní objekt **properties** je vestavěný a nemusí být při konstrukci obchodního objektu určen ve schématu XML. Stačí jej vytvořit a použít. Pokud máte nadefinovány vlastní datové typy na základě schématu XML, potřebujete určit prvek **properties**, který obsahuje očekávané dvojice název-hodnota.

```
BOFactory dataFactory = (BOFactory) \
serviceManager.locateService("com/ibm/websphere/bo/BOFactory");
 //Obálka pro rozhraní zabaleného stylu doc-lit,
 //přeskočit na informační obsah pro non doc-lit
 DataObject docLitWrapper = dataFactory.createByElement /
 ("http://mytest/eis/Account", "AccountWrapper");
```

Vytvořte informační obsah.

```
DataObject account = docLitWrapper.createDataObject(0);
  DataObject accountInfo = account.createDataObject("AccountInfo");
  //Provést nastavení informačního obsahu


  //Vytvořit data vlastností pro dynamickou interakci

  DataObject properties = account.createDataObject("properties");
```

Pro název workingSet nastavte očekávanou hodnotu (**xyz**).

```
properties.setString("ISworkingSet", "xyz");


  //Vyvolat službu s argumentem

  Service accountImport = (Service) \
  serviceManager.locateService("AccountOutbound");
  DataObject result = accountImport.invoke("createAccount", docLitWrapper);

  //Získat vrácenou vlastnost
```

```
DataObject retProperties = result.getDataObject("properties");

String workingset = retProperties.getString("ISworkingSet");
```

Vlastnosti ConnectionSpec můžete použít pro dynamické ověření komponent. Platí stejná pravidla jako výše, jen předpona názvu vlastnosti musí být **CS** (místo **IS**). Vlastnosti ConnectionSpec nejsou obousměrné. Stejný datový objekt **properties** může obsahovat vlastnosti IS i CS.

Chcete-li použít vlastnosti ConnectionSpec, nastavte vlastnost **resAuth** určenou pro vazbu importu na hodnotu **Application**. Ujistěte se také, že adaptér prostředku podporuje autorizaci komponent. Další podrobnosti viz kapitola 8: Specifikace architektury konektoru J2EE.

Externí klienti s vazbami EIS:

Server dokáže odesílat zprávy externím klientům pomocí vazeb EIS nebo od nich zprávy přijímat.

Externí klient, například webový portál nebo podnikový informační systém, potřebuje odeslat zprávu modulu SCA na serveru nebo potřebuje být vyvolán komponentou z tohoto serveru.

Klient vyvolá import EIS jako u všech ostatních aplikací, buď s použitím rozhraní DDI (Dynamic Invocation Interface) nebo s použitím rozhraní Java.

1. Externí klient vytvoří instanci rozhraní ServiceManager a vyhledá import EIS s použitím jeho referenčního názvu. Výsledkem hledání je implementace rozhraní služby.
2. Klient vytvoří vstupní argument, generický datový objekt vytvořený dynamicky s použitím schématu datového objektu. Tento krok se provádí s pomocí implementace rozhraní Service Data Object DataFactory.
3. Externí klient vyvolá podnikový informační systém a získá požadované výsledky.

Případně může klient vyvolat import EIS s pomocí rozhraní Java.

1. Klient vytvoří instanci rozhraní ServiceManager a vyhledá import EIS s použitím jeho referenčního názvu. Výsledkem hledání je rozhraní Java importu EIS.
2. Klient vytvoří vstupní argument a typovaný datový objekt.
3. Klient vyvolá podnikový informační systém a získá požadované výsledky.

Rozhraní exportu EIS definuje rozhraní exportované komponenty SCA, které je k dispozici externím aplikacím podnikového informačního systému. Toto rozhraní lze považovat za rozhraní, které vyvolá externí aplikace (např. SAP nebo PeopleSoft) prostřednictvím implementace běhového prostředí aplikace exportu EIS.

Export použije vazbu EISExportBinding ke svázání exportovaných služeb s externí aplikací EIS. Umožní přihlásit aplikaci obsaženou v modulu SCA k naslouchání požadavkům služby EIS. Vazba exportu EIS určuje mapování mezi definicí příchozích událostí, jak jim rozumí adaptér prostředku (s použitím rozhraní architektury Java EE Connector Architecture) a vyvolání operací SCA.

Vazba EISExportBinding vyžaduje, aby byly externí služby EIS založeny na příchozích kontraktech architektury Java EE Connector Architecture 1.5. Vazba EISExportBinding vyžaduje, aby byly popisovač dat nebo vázání dat určeny buď na úrovni vazby, nebo na úrovni metody.

Vazby JMS

Poskytovatel JMS (Java Message Service) umožňuje systém zpráv na základě rozhraní API a programovacího modelu Java Messaging Service. . Poskytuje továrny připojení JMS pro vytvoření připojení pro cíle JMS a pro odesílání a příjem zpráv.

Vazby JMS lze použít při interakci s vazbou poskytovatele sběrnice SIB (Service Integration Bus) a jsou kompatibilní s JMS a JCA 1.5.

Vazby exportu a importu JMS umožňují modulu architektury SCA (Service Component Architecture) volat externí systémy JMS a přijímat z nich zprávy.

Vazby importu a exportu JMS zajišťují integraci s aplikacemi JMS s použitím poskytovatele JMS SIB na bázi JCA 1.5, který je součástí serveru WebSphere Application Server. Jiné adaptéry prostředku JMS na bázi JCA 1.5 podporovány nejsou.

Přehled vazeb JMS:

Vazby JMS zajišťují konektivitu mezi prostředím architektury SCA (Service Component Architecture) a systémy JMS.

Vazby JMS

Hlavní komponenty vazeb importu JMS i exportu JMS jsou:

- Adaptér prostředku: umožňuje spravovanou, obousměrnou konektivitu mezi modulem SCA a externím modulem JMS.
- Připojení: zapouzdřují virtuální připojení mezi klientem a aplikací poskytovatele.
- Cíle: používány klientem k určení cíle jím produkovaných zpráv nebo zdroje jím přijímaných zpráv.
- Data ověřování: používána k zabezpečení přístupu k vazbě.

Klíčové vlastnosti vazby JMS

Speciální záhlaví

Vlastnosti speciálního záhlaví se používají v importech a exportech JMS, aby bylo možné cíli říct, jak zpracovat zprávu.

Například mapy TargetFunctionName z nativní metody do operační metody.

Prostředky Java EE

Mnoho prostředků Java EE je vytvářeno, když jsou importy a exporty JMS implementována do prostředí Java EE.

ConnectionFactory
Používaná klienty k vytvoření připojení k poskytovateli JMS.

ActivationSpec
Používané importy pro načtení odpovědi na požadavek; exporty tuto funkci využívají, když konfigurují koncové body zprávy, které představují listenery zpráv v jejich interakci se systémem zasílání zpráv.

Destinations

- Cíl odesílání: při importu je to místo, kam je zasílán požadavek nebo odchozí zpráva; při exportu jde o místo, kam bude zaslána zpráva odpovědi v případě, že není nahrazena polem záhlaví JMSReplyTo v příchozí zprávě.
- Cíl příjmu: kam by měla být umístěna příchozí zpráva; při importu jde o odpověď; při exportu jde o požadavek.
- Cíl zpětného volání: Cíl systému SCA JMS použitý k uložení korelačních informací. Toto místo určení nečtěte, ani do něj nezapisujte.

Úloha instalace vytváří položku ConnectionFactory a tři místa určení. Rovněž vytváří položku ActivationSpec, pomocí které lze povolit listeneru zprávy běhového prostředí naslouchat odpovědím v cíli příjmu. Vlastnosti těchto prostředků jsou určeny v souboru importu a exportu.

Integrace JMS a adaptéry prostředku:

Platforma JMS (Java Message Service) zajišťuje integraci prostřednictvím dostupného adaptéru prostředku na bázi JMS JCA 1.5. Je poskytnuta plná podpora integrace JMS pro adaptér prostředku Service Integration Bus (SIB) JMS.

Pro integraci s externím systémem JMS kompatibilním s JCA 1.5 použijte poskytovatel JMS pro adaptér prostředku JCA 1.5. Externí služby kompatibilní s JCA 1.5 mohou přijímat a odesílat zprávy za účelem integrace s komponentami architektury SCA (Service Component Architecture) pomocí adaptéru prostředku SIB JMS.

Použití adaptérů prostředku JCA 1.5 specifických pro jiné poskytovatele není podporováno.

Vazby importu a exportu JMS:

Pomocí vazeb importu a exportu JMS můžete nastavit interakci modulů SCA se službami poskytovanými externími aplikacemi JMS.

Vazby importu JMS

Připojení k přidruženému poskytovateli JMS cílů JMS je vytvořeno pomocí továrny připojení JMS. Při správě továren připojení JMS pro výchozího poskytovatele systému zpráv použijte administrativní objekty továrny připojení.

Interakce s externími systémy JMS zahrnují použití cílů pro odesílání požadavků a příjem odpovědí.

Jsou podporovány dva typy scénářů využití vazeb importu JMS, v závislosti na typu vyvolávané operace:

- Jednosměrný: Import JMS vloží zprávu do cíle operace odeslání nakonfigurovaného v dané vazbě importu. V poli replyTo záhlaví JMS není nastaveno nic.
- Obousměrný (požadavek-odezva): Import JMS vloží zprávu do cíle operace odeslání a poté trvale uloží odpověď, kterou obdrží od komponenty SCA.

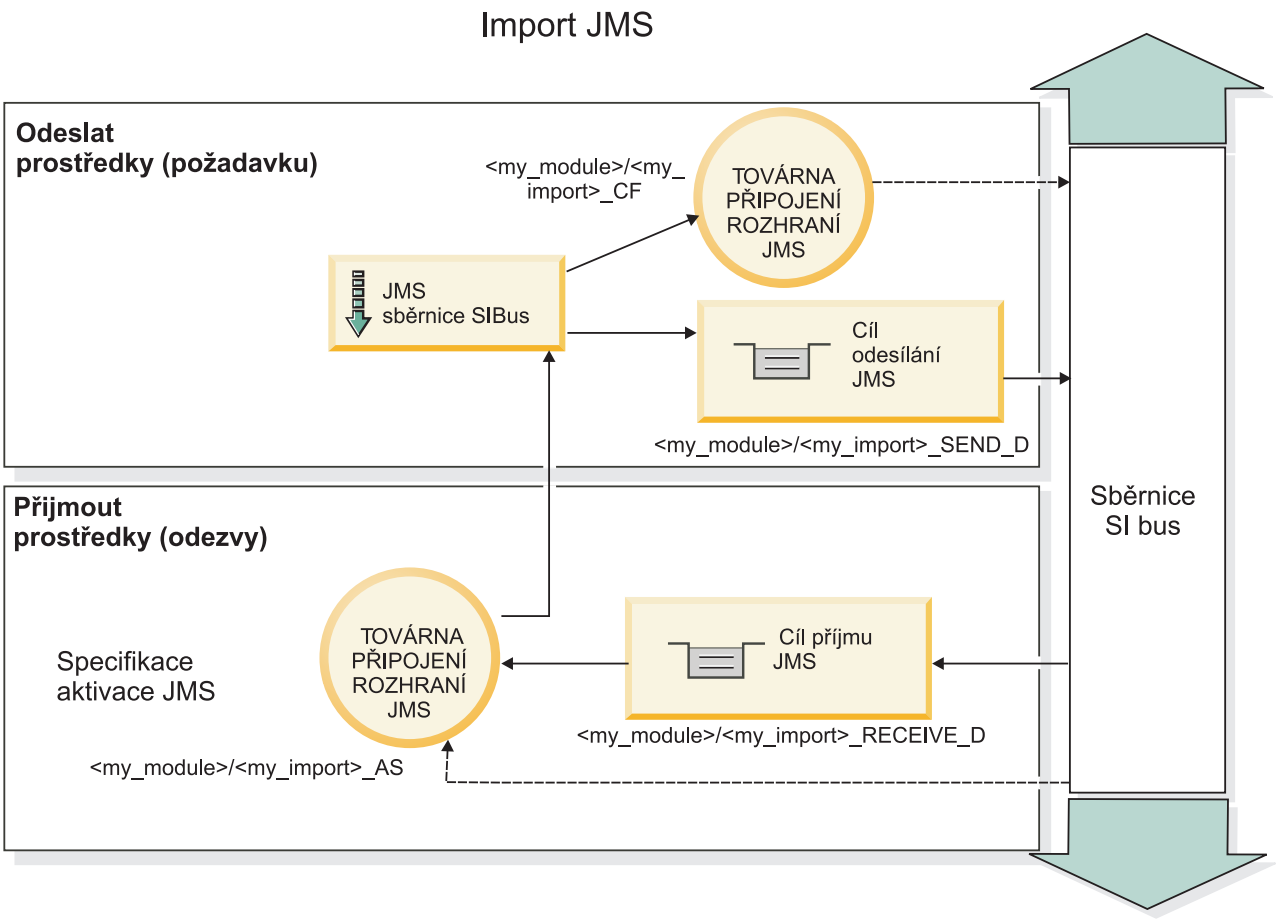
  Vazbu importu lze nakonfigurovat (pomocí pole **Schéma korelace odpovědí** v produktu Integration Designer), aby očekávala ID korelace zprávy odpovědi zkopírované z ID zprávy požadavku (výchozí nastavení), nebo z ID korelace zprávy požadavku. Vazbu importu lze také nakonfigurovat, aby ke korelaci odpovědí s požadavky používala dočasný cíl dynamických odpovědí. Je vytvořen dočasný cíl pro každý požadavek a import tento cíl používá pro příjem odpovědi.

  Cíl operace příjmu (receive) je nastaven ve vlastnosti záhlaví replyTo odchozí zprávy. Je implementován listener zpráv, aby naslouchal cíli operace příjmu (receive), a když je přijata odpověď, listener zpráv předá tuto odpověď zpět komponentě.

Pro jednosměrné i obousměrné scénáře použití lze určit dynamické a statické vlastnosti záhlaví. Statické vlastnosti lze nastavit z vazby metody importu JMS. Některé z těchto vlastností mají zvláštní význam pro běhové prostředí SCA JMS.

Je důležité uvědomit si, že JMS je asynchronní vazba. Pokud volající komponenta vyvolá import JMS synchronně (pro obousměrnou operaci), je volající komponenta zablokována, dokud služba JMS nevrátí odpověď.

Obrázek 32 na stránce 111 ukazuje, jak je import propojen s externí službou.



Obrázek 77. Prostředky vazby importu JMS

Vazby exportu JMS

Vazby exportu JMS zajišťují modulům SCA prostředky pro poskytování služeb externím aplikacím JMS.

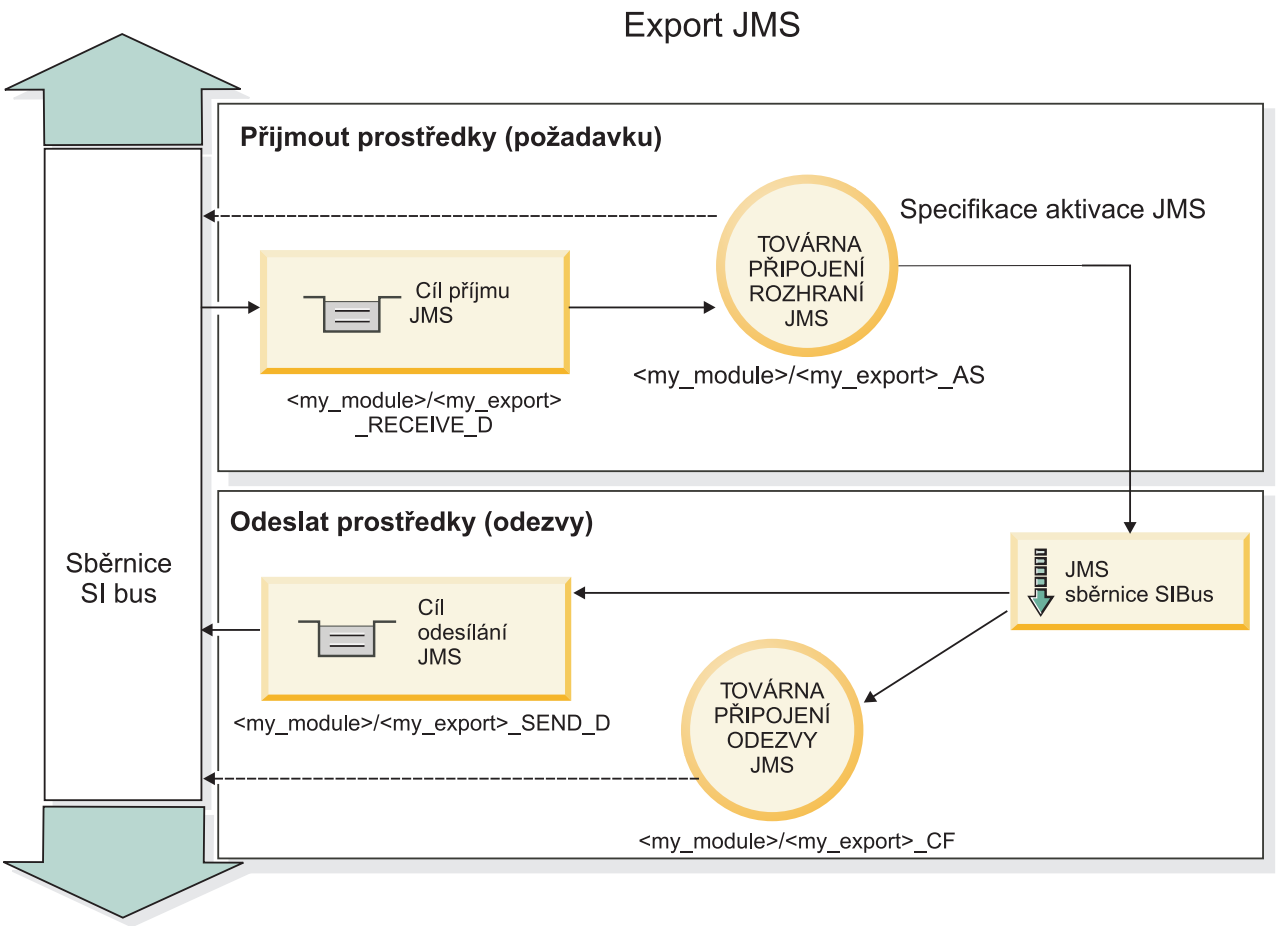
Připojení, které je částí exportu JMS je konfigurovatelnou specifikací aktivace.

Export JMS má cíle operací odeslání a příjmu (send a receive).

- Cíl operace příjmu (receive) je místo, kam se má vložit příchozí zpráva pro cílovou komponentu.
- Cíl operace odeslání (send) je místo, kam bude odeslána odpověď, pokud je příchozí zpráva nepotlačí pomocí vlastnosti záhlaví replyTo.

Listener zpráv je implementován, aby naslouchal požadavkům přicházejícím do cíle operace příjmu (receive) určeného ve vazbě exportu. Cíl určený v poli odeslání (send) se používá k odeslání odpovědi na příchozí požadavek, pokud vyvolaná komponenta poskytuje odpověď. Cíl určený v poli replyTo příchozí zprávy přepíše cíl určený v poli odeslání (send).

Obrázek 33 na stránce 112 ukazuje, jak je externí žadatel propojen s exportem.



Obrázek 78. Prostředky vazby exportu JMS

Záhlaví JMS:

Zpráva JMS obsahuje dva typy záhlaví: systémová záhlaví JMS a různé vlastnosti JMS. K oběma typům záhlaví lze přistupovat buď v mediačním modulu v objektu SMO (Service Message Object), nebo prostřednictvím rozhraní API ContextService.

Systémové záhlaví JMS

Systémové záhlaví JMS je v objektu SMO představováno prvkem JMSHeader, který obsahuje všechna pole, jež se obvykle vyskytují v záhlaví JMS. Ačkoli je lze upravit v mediaci (nebo službě ContextService), některá pole systémových záhlaví JMS nastavená v objektu SMO se nebudou šířit v odchozí zprávě JMS, protože jsou přepsána systémovými nebo statickými hodnotami.

Klíčová pole v systémovém záhlaví JMS, která lze aktualizovat v mediaci (nebo službě ContextService):

- **JMSType** a **JMSCorrelationID** – hodnoty specifických předdefinovaných vlastností záhlaví zprávy.
- **JMSDeliveryMode** – hodnoty způsobu doručení persistent (perzistentní) nebo non-persistent (dočasný); výchozí je persistent).
- **JMSPriority** – hodnota priority (0 až 9; výchozí je JMS_Default_Priority.)

Vlastnosti JMS

Vlastnosti JMS jsou v objektu SMO reprezentovány položkami seznamu Vlastnosti. Vlastnosti lze přidávat, aktualizovat nebo odstraňovat v mediaci nebo prostřednictvím rozhraní API ContextService.

Vlastnosti lze také nastavit staticky ve vazbě JMS. Vlastnosti, které jsou nastaveny staticky, přepisují nastavení (se stejným názvem), která jsou nastavena dynamicky.

Uživatelské vlastnosti šířené z jiných vazeb (například jako vazba HTTP) budou výstupem ve vazbě JMS jako vlastnosti JMS.

Nastavení šíření záhlaví

Šíření systémového záhlaví a vlastností JMS buď z příchozí zprávy JMS do následných komponent, nebo z předchozích komponent do odchozí zprávy JMS lze řídit příznakem Šířit záhlaví protokolů pro danou vazbu.

Když je nastaven příznak Šířit záhlaví protokolů, je informacím o záhlaví dovoleno plynout do zprávy nebo do cílové komponenty, jak popisuje následující seznam:

- Požadavek exportu JMS

  Záhlaví JMS přijaté ve zprávě bude šířeno do cílové komponenty prostřednictvím kontextové služby. Vlastnosti JMS přijaté ve zprávě budou šířeny do cílové komponenty prostřednictvím kontextové služby.
- Odpověď exportu JMS

  Všechna pole záhlaví JMS nastavená v kontextové službě budou použita v odchozí zprávě, pokud nejsou přepsána statickými vlastnostmi nastavenými pro vazbu exportu JMS. Všechny vlastnosti nastavené v kontextové službě budou použity v odchozí zprávě, pokud nejsou přepsány statickými vlastnostmi nastavenými pro vazbu exportu JMS.
- Požadavek importu JMS

  Všechna pole záhlaví JMS nastavená v kontextové službě budou použita v odchozí zprávě, pokud nejsou přepsána statickými vlastnostmi nastavenými pro vazbu importu JMS. Všechny vlastnosti nastavené v kontextové službě budou použity v odchozí zprávě, pokud nejsou přepsány statickými vlastnostmi nastavenými pro vazbu importu JMS.
- Odpověď importu JMS

  Záhlaví JMS přijaté ve zprávě bude šířeno do cílové komponenty prostřednictvím kontextové služby. Vlastnosti JMS přijaté ve zprávě budou šířeny do cílové komponenty prostřednictvím kontextové služby.

Schéma korelace dočasného dynamického cíle odpovědi JMS:

Schéma korelace dočasného dynamického cíle odpovědi způsobí vytvoření jedinečné dynamické fronty nebo tématu pro každý odeslaný požadavek.

Statický cíl odpovědi určený v importu se používá k odvození charakteru dočasné dynamické fronty nebo tématu cíle. To se nastavuje v poli **ReplyTo** požadavku a import JMS naslouchá odpovědím v tomto cíli. Když je přijata odpověď, je znovu zařazena do fronty pro statický cíl odpovědi pro asynchronní zpracování. Pole **CorrelationID** odpovědi se nepoužívá a není nutné je nastavovat.

Transakční problematika

Když se používá dočasný dynamický cíl, musí být odpověď využita ve stejném podprocesu jako odeslaná odpověď. Požadavek musí být odeslán mimo globální transakci a musí být potvrzen dřív, než bude přijat back-endovou službou a bude vrácena odpověď.

Perzistence

Dočasné dynamické fronty jsou krátkodobé entity a nezaručují stejnou úroveň perzistence, jaká se přidružuje ke statické frontě nebo tématu. Dočasná dynamická fronta nebo téma nepřežije restart serveru a totéž platí pro zprávy. Poté, co je zpráva nově zařazena do fronty pro statický cíl odpovědi, si zachová perzistenci, která je definovaná ve zprávě.

Časový limit

Import čeká na přijetí odpovědi v dočasném dynamickém cíli odpovědi po stanovenou dobu. Tento časový interval bude převzat z časového kvalifikátoru Vypršení platnosti odpovědi SCA, je-li nastaven, jinak je výchozí nastavení této doby 60 sekund. Pokud je překročena doba čekání, vrátí import výjimku ServiceTimeoutRuntimeException.

Externí klienti:

Server dokáže odesílat zprávy externím klientům pomocí vazeb JMS nebo od nich zprávy přijímat.

Externí klient (například webový portál nebo podnikový informační systém) může odeslat zprávu modulu SCA na serveru nebo může být vyvolán komponentou z tohoto serveru.

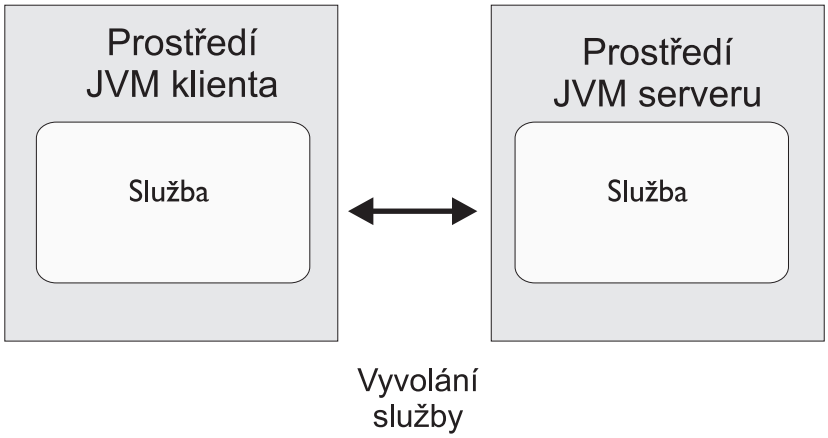
Komponenty exportu JMS implementují listenery zpráv, aby naslouchaly požadavkům přicházejícím do cíle příjmu určeného ve vazbě exportu. Cíl určený v poli odeslání se používá k odeslání odpovědi na příchozí požadavek, pokud vyvolaná aplikace nějakou odpověď poskytne. Proto je externí klient schopen vyvolávat aplikace s pomocí vazby exportu.

Importy JMS zajišťují interakci s externími klienty prostřednictvím zasílání zpráv do front JMS a příjmu zpráv z front JMS.

Práce s externími klienty:

Externí klient (tj. klient mimo server) může vyžadovat interakci s aplikací instalovanou na serveru.

Zvažte velmi jednoduchý scénář, ve kterém externí klient požaduje interakci s aplikací na serveru. Na obrázku je znázorněn typický jednoduchý scénář.



Obrázek 79. Jednoduchý scénář příkladu použití: Externí klient umožňuje interakci se serverovou aplikací

Aplikace SCA zahrnuje export s vazbou JMS; tím je aplikace zpřístupněna pro externí klienty.

Když máte externího klienta v prostředí JVM (Java Virtual Machine) odděleném od serveru, je třeba pro připojení a interakci s exportem JMS provést několik kroků. Klient získá objekt InitialContext se správnými hodnotami a poté vyhledá prostředky pomocí rozhraní JNDI. Poté klient prostřednictvím klienta specifikace JMS 1.1 přistupuje k cílům a odesílá a přijímá zprávy v cílech.

Výchozí názvy rozhraní JNDI prostředků automaticky vytvořených běhovým prostředím jsou uvedeny v tématu o konfiguraci v tomto oddílu. Máte-li však předem vytvořené prostředky, použijte tyto názvy rozhraní JNDI.

1. Nakonfigurujte místa určení JMS a továrnu připojení k odeslání zprávy.

2. Přesvědčte se, že kontext rozhraní JNDI, port pro adaptér prostředku SIB i port samozavedení systému zpráv jsou správné.

   Server používá některé výchozí porty, ale pokud je na daném systému instalováno více serverů, jsou při instalaci vytvořeny alternativní porty, aby se předešlo konfliktům s dalšími instancemi serveru. Pomocí administrativní konzoly můžete určit, které porty server používá. Přejděte na **Servery > Aplikační servery > *název_vašeho_serveru* > Konfigurace** a klepněte na volbu **Porty** pod volbou **Komunikace**. Poté můžete upravit používaný port.
3. Klient obdrží počáteční kontext se správnými hodnotami a poté vyhledá prostředky prostřednictvím rozhraní JNDI.
4. Klient pomocí specifikací JMS 1.1 přistupuje k cílům a odesílá a přijímá zprávy v cílech.

Odstraňování problémů s vazbami služby JMS:

Problémy s vazbami služby JMS lze diagnostikovat a opravit.

Výjimky implementace

Jako odpověď na různé chybové stavy může implementace importu a exportu JMS vracet jeden ze dvou typů výjimek:

- Obchodní výjimka služby: Tato výjimka je vrácena, pokud došlo k poruše určené na obchodním rozhraní služby (typ portu WSDL).
- Výjimka za běhu služby: Generuje se ve všech ostatních případech. Ve většině případů bude výjimka příčina obsahovat původní výjimku (JMSException).

  Například import očekává pouze jednu zprávu odezvy na každou zprávu požadavku. Pokud dorazí více než jedna odezva nebo pokud dorazí pozdní odezva (odezva, pro kterou došlo k vypršení odezvy SCA), je vyvolána výjimka za běhu služby. Transakce je odvolána a zpráva odezvy je vrácena zpět z fronty nebo zpracována správcem událostí se selháním.

Primární podmínky selhání

Primární podmínky selhání vazeb JMS jsou určeny transakční sémantikou, konfigurací poskytovatele JMS nebo odkazem na existující chování v jiných komponentách. Primární podmínky selhání zahrnují:

- Selhání při připojení k poskytovateli JMS nebo k místu určení.

  Selhání při připojení k poskytovateli JMS při přijímání zpráv bude mít za následek selhání spuštění listeneru zpráv. Tato podmínka bude zaznamenána do protokolu produktu WebSphere Application Server. Trvalé zprávy zůstanou v místě určení, dokud nebudou úspěšně načteny (nebo dokud nevyprší jejich platnost).

  Selhání při připojení k poskytovateli JMS při odesílání odchozích zpráv způsobí odvolání transakce, která řídí odeslání.
- Selhání při analýze příchozí zprávy nebo při vytvoření odchozí zprávy.

  Selhání ve vázání dat nebo popisovači dat způsobí odvolání transakce, která tuto práci řídí.
- Selhání při odesílání odchozí zprávy.

  Selhání při odesílání zprávy způsobí odvolání příslušné transakce.
- Vícenásobné nebo neočekávané zprávy o pozdní odezvě.

  Import očekává pouze jednu zprávu odezvy na každou zprávu požadavku. Kvalifikátor vypršení odezvy SCA u požadavku také určuje platné časové období, během kterého může být odezva přijata. Když odezva dorazí nebo je překročen čas vypršení platnosti, záznam korelace je odstraněn. Pokud dorazí zprávy odezvy neočekávaně nebo pozdě, dojde k výjimce za běhu služby.
- Výjimka vypršení časového limitu za běhu služby způsobená pozdní odezvou při použití schématu korelace cíle dočasné dynamické odezvy.

  Importu JMS vyprší časový limit po uplynutí časového úseku, který je určen kvalifikátorem vypršení odezvy SCA, nebo není-li tento kvalifikátor nastaven, použije se standardně časový úsek 60 sekund.

Zprávy SCA založené na JMS se nezobrazují ve správci událostí se selháním

Pokud dojde k selhání zpráv SCA vytvořených prostřednictvím interakce JMS, měly by se tyto zprávy objevit ve správci událostí se selháním. Pokud se tyto zprávy ve správci událostí se selháním neobjeví, zkontrolujte, zda má základní místo určení SIB místa určení JMS hodnotu maximálního počtu nezdařených doručení větší než **1**. Nastavení této hodnoty na **2** a více umožní interakci se správcem událostí se selháním během vyvolání SCA pro vazby JMS.

Zpracování výjimek:

Způsob, jakým je vazba nakonfigurována, určuje, jak jsou zpracovány výjimky, které jsou způsobeny popisovači dat nebo vázáními dat. Chování systému při vrácení takové výjimky je dále určeno také charakterem mediačního toku.

Když vaše vazba volá popisovač dat nebo vázání dat, může dojít k nejrůznějším problémům. Například může popisovač dat obdržet zprávu s poškozeným informačním obsahem nebo se může pokusit přečíst zprávu v nesprávném formátu.

Způsob, jakým vaše vazba ošetří takovou výjimku, je určen tím, jak implementujete daný popisovač dat nebo vázání dat. Doporučené chování je navrhnout vázání dat tak, aby vracelo výjimku **DataBindingException**.

Když je vrácena jakákoli výjimka za běhu, včetně výjimky **DataBindingException**:

- Pokud je mediační tok nakonfigurován jako transakční, je zpráva JMS standardně uložena do komponenty Failed Event Manager pro ruční přehrání nebo odstranění.

  Poznámka: Režim zotavení pro vazbu je možné změnit tak, aby byla zpráva namísto uložení do komponenty Failed Event Manager odvolána.
- Pokud mediační tok není transakční, je výjimka zaprotokolována a zpráva je ztracena.

Podobná situace je i u popisovače dat. Protože je popisovač dat vyvolán vázáním dat, je jakákoli výjimka popisovače dat zabalena do výjimky vázání dat. Proto je výjimka **DataHandlerException** hlášena jako **DataBindingException**.

Generické vazby JMS

Generická vazba služby JMS zajišťuje konektivitu pro poskytovatele jiných dodavatelů odpovídající standardu JMS 1.1. Činnost Generické vazby služby JMS je podobná činnosti vazeb JMS.

Služba poskytovaná prostřednictvím JMS umožňuje modulu SCA (Service Component Architecture) provádět volání a přijímat zprávy z externích systémů. Takový systém může být externím systémem JMS.

Generická vazba služby JMS zajišťuje integraci s poskytovateli JMS nekompatibilními s JCA 1.5, kteří podporují JMS 1.1 a implementují volitelný mechanizmus JMS Application Server Facility. Generická vazba služby JMS podporuje poskytovatele JMS (včetně Oracle AQ, TIBCO, SonicMQ, WebMethods a BEA WebLogic), kteří nepodporují JCA 1.5, ale podporují ASF (Application Server Facility) podle specifikace JMS 1.1. Vestavěný poskytovatel WebSphere JMS (SIBJMS), což je poskytovatel JMS JCA 1.5, není touto vazbou podporován. Při použití tohoto poskytovatele použijte “Vazby JMS” na stránce 108.

Tuto Generickou vazbu použijte při integraci se systémem JMS nekompatibilním s JCA 1.5 v rámci prostředí SCA. Cílové externí aplikace pak mohou přijímat zprávy a odesílat zprávy za účelem integrace s komponentou SCA.

Přehled generických vazeb služby JMS:

Generické vazby JMS jsou vazby JMS nekompatibilní s JCA, které zajišťují konektivitu mezi prostředím SCA (Service Component Architecture) a systémy JMS kompatibilními s JMS 1.1 a implementujícími volitelný mechanizmus JMS Application Server Facility.

Generické vazby JMS

Hlavní aspekty generických vazeb importu a exportu služby JMS jsou:

- Port modulu listener: umožňuje poskytovatelům JMS nekompatibilním s JCA přijímat zprávy a odesílat je objektu typu message-driven bean (MDB).
- Připojení: zapouzdřují virtuální připojení mezi klientem a aplikací poskytovatele.
- Cíle: používány klientem k určení cíle jím produkovaných zpráv nebo zdroje jím přijímaných zpráv.
- Data ověřování: používána k zabezpečení přístupu k vazbě.

Generické vazby importu služby JMS

Generické vazby importu JMS umožňují komponentám v rámci vašeho modulu SCA komunikovat se službami poskytovanými externími poskytovateli JMS nekompatibilními s JCA 1.5.

Část připojení importu JMS je továrna připojení. Továrna připojení, objekt používaný klientem k vytvoření připojení k poskytovateli, zapouzdřuje sadu konfiguračních parametrů připojení definovaných administrátorem. Každá továrna připojení je instancí rozhraní ConnectionFactory, QueueConnectionFactory nebo TopicConnectionFactory.

Interakce s externími systémy JMS zahrnují použití cílů pro odesílání požadavků a příjem odpovědí.

Jsou podporovány dva typy scénářů využití generických vazeb importu JMS, v závislosti na typu vyvolávané operace:
- Jednosměrný: Generický import JMS vloží zprávu do cíle operace odeslání nakonfigurovaného v dané vazbě importu. Do pole replyTo v záhlaví JMS se nic nepošle.
- Obousměrný (požadavek-odezva): Generický import JMS vloží zprávu do cíle operace odeslání a poté trvale uloží odpověď, kterou obdrží od komponenty SCA.

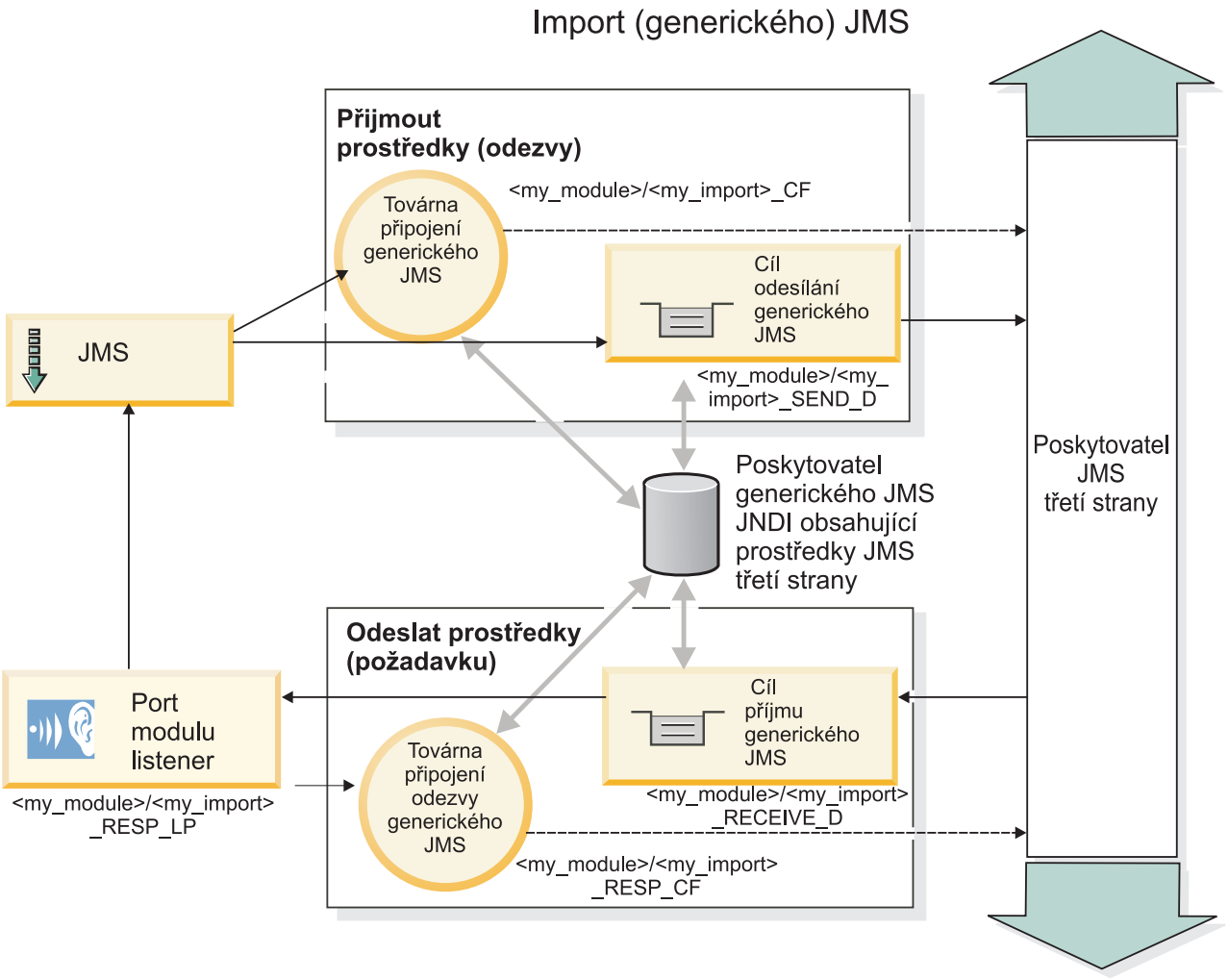
  Cíl operace příjmu (receive) je nastaven ve vlastnosti záhlaví replyTo odchozí zprávy. Je implementován objekt typu message-driven bean (MDB), aby naslouchal cíli operace příjmu (receive), a když je přijata odpověď, MDB předá tuto odpověď zpět komponentě.

  Vazbu importu lze nakonfigurovat (pomocí pole **Schéma korelace odpovědí** v produktu Integration Designer), aby očekávala ID korelace zprávy odpovědi zkopírované z ID zprávy požadavku (výchozí nastavení), nebo z ID korelace zprávy požadavku.

Pro jednosměrné i obousměrné scénáře použití lze určit dynamické a statické vlastnosti záhlaví. Statické vlastnosti lze nastavit z generické vazby metody importu JMS. Některé z těchto vlastností mají zvláštní význam pro běhové prostředí SCA JMS.

Je důležité uvědomit si, že Generická platforma JMS je asynchronní vazba. Pokud volající komponenta vyvolá Generický import JMS synchronně (pro obousměrnou operaci), je volající komponenta zablokována, dokud služba JMS nevrátí odpověď.

Obrázek 35 na stránce 118 ukazuje, jak je import propojen s externí službou.



Obrázek 80. Prostředky generické vazby importu JMS

Generické vazby exportu JMS

Generické vazby exportu JMS zajišťují modulům SCA prostředky pro poskytování služeb externím aplikacím JMS.

Část připojení exportu JMS tvoří továrna ConnectionFactory a port ListenerPort.

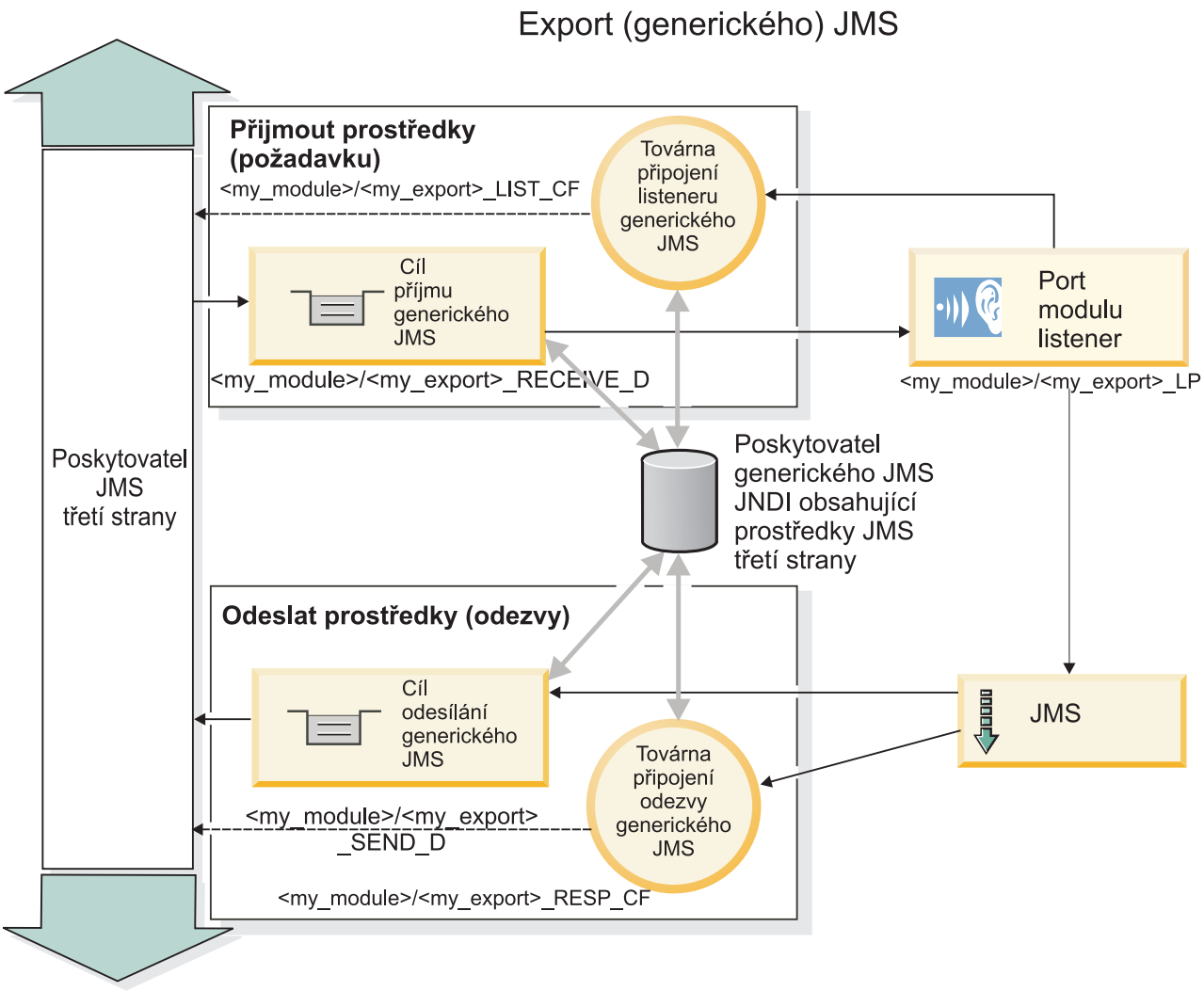
Generický export JMS má cíle operací odeslání a příjmu (send a receive).

- Cíl operace příjmu (receive) je místo, kam se má vložit příchozí zpráva pro cílovou komponentu.
- Cíl operace odeslání (send) je místo, kam bude odeslána odpověď, pokud je příchozí zpráva nepotlačí pomocí vlastnosti záhlaví replyTo.

Je implementován MDB, aby naslouchal požadavkům přicházejícím do cíle operace příjmu (receive) určeného ve vazbě exportu.

- Cíl určený v poli odeslání (send) se používá k odeslání odpovědi na příchozí požadavek, pokud vyvolaná komponenta nějakou odpověď poskytne.
- Cíl určený v poli replyTo příchozí zprávy přepíše cíl určený v poli odeslání (send).
- Pro scénáře požadavku/odpovědi může být nakonfigurována vazba importu (pomocí pole **Schéma korelace odpovědi** v produktu Integration Designer), aby očekávala, že odpověď zkopíruje ID zprávy požadavku do pole ID korelace odpovědi (výchozí nastavení), nebo může odpověď zkopírovat ID korelace požadavku do pole ID korelace zprávy odpovědi.

Obrázek 36 na stránce 119 ukazuje, jak je externí žadatel propojen s exportem.



Obrázek 81. Prostředky generické vazby exportu JMS

Klíčové faktory vazby Generických vazeb služby JMS:

Faktory Generické vazby importu a export služby JMS jsou konzistentní s vestavěnými vazbami importu služby JMS MQ a JMS produktu WebSphere. Mezi klíčové faktory patří definice záhlaví a přístup k existujícím prostředkům Java EE. Avšak z důvodu jejich generického charakteru neexistují žádné možnosti připojení specifické pro poskytovatele služby a tato vazba má omezenou schopnost generovat prostředky při implementaci a instalaci.

Generické importy

Jako aplikace importu služby MQ JMS je Generická implementace služby JMS asynchronní a podporuje tři vyvolání: jednosměrné, obousměrné (rovněž známé jako požadavek-odezva) a zpětné volání.

Když je implementován import služby JMS, je implementován objekt typu message-driven bean (MDB) poskytovaný běhovým prostředím. Objekt MDB naslouchá odpovědím na zprávu požadavku. Objekt MDB je přidružen k (naslouchá) cíli zaslanému s požadavkem v poli záhlaví replyTo zprávy služby JMS.

Generické exporty

Generická vazba služby JMS se liší od vazeb exportu EIS v tom, jak pracují s návratem výsledku. Generický export služby JMS explicitně zasílá odezvu na cíl replyTo určený v příchozí zprávě. Pokud není určen, je použit cíl odeslání.

Když je implementován Generický export služby JMS, je implementován objekt typu message-driven bean (jiný objekt typu MDB než byl ten, jenž byl použit pro Generické importy systému JMS). Naslouchá příchozím požadavkům na cíli příjmu a poté odbavuje požadavky, aby byly zpracovány běhovým prostředím SCA.

Speciální záhlaví

Vlastnosti speciálního záhlaví se používají v Generických v importech a exportech JMS, aby bylo možné cíli vazby říct, jak zpracovat zprávu.

Vlastnost TargetFunctionName používá výchozí selektor funkcí k identifikaci názvu vlastnosti v rozhraní exportu, které je vyvoláváno.

Poznámka: Vazbu importu lze konfigurovat tak, aby nastavilo záhlaví TargetFunctionName na názvy jednotlivých operací.

Prostředky Java EE

Mnoho prostředků Java EE je vytvářeno, když je vazba JMS implementována do prostředí Java EE.

- Port modulu listener v cíli příjmu (odezvy, pouze obousměrný) pro importy a v cíli příjmu (požadavku) pro exporty.
- Generická továrna připojení JMS pro odchozí připojení (import) a příchozí připojení (export).
- Generické cíle JMS pro cíle odesílání (import) a přijímání (export, pouze obousměrný).
- Generická továrna připojení JMS pro připojení odezvy (pouze obousměrné a volitelné, jinak je pro importy používáno odchozí připojení a pro exporty je používáno příchozí připojení).
- Generický cíl JMS pro cíle příjmu (import) a odesílání (export, pouze obousměrné).
- Výchozí cíl JMS zpětného volání poskytovatele systému zpráv použitý k přístupu k cíli fronty zpětného volání SIB (pouze obousměrné).
- Výchozí továrna připojení JMS zpětného volání poskytovatele systému zpráv (pouze obousměrné).
- Cíl fronty zpětného volání SIB k uložení informací o zprávě požadavku pro použití během zpracování odezvy (pouze obousměrné)

Instalační úloha vytváří Továrnu připojení, tři cíle a Specifikace aktivace z informací v souborech importu a exportu.

Generická záhlaví JMS:

Generická záhlaví JMS jsou objekty SDO (Service Data Objects), které obsahují všechny vlastnosti vlastností Generických zpráv JMS. Tyto vlastnosti mohou pocházet z příchozí zprávy nebo se může jednat o vlastnosti, které budou použity na odchozí zprávu.

Zpráva JMS obsahuje dva typy záhlaví: systémová záhlaví JMS a různé vlastnosti JMS. K oběma typům záhlaví lze přistupovat buď v mediačním modulu v objektu SMO (Service Message Object), nebo prostřednictvím rozhraní API ContextService.

Následující vlastnosti jsou staticky nastaveny na vazbě methodBinding:

- JMSType.
- JMSCorrelationID.
- JMSDeliveryMode.
- JMSPriority.

Generická vazba JMS podporuje také dynamické úpravy záhlaví a vlastností JMS stejným způsobem jako u vazeb JMS a MQ JMS.

Někteří Generičtí poskytovatelé JMS omezují vlastnosti, které může aplikace nastavit, a jejich kombinace. Další informace viz dokumentace příslušných produktů jiných dodavatelů. K vazbě methodBinding, však byla přidána doplňková vlastnost ignoreInvalidOutboundJMSProperties, která umožňuje šíření libovolných výjimek.

Generická záhlaví JMS a vlastnosti zpráv se používají, jen když je zapnutý přepínač vazby SCDL architektury SCA. Když je tento přepínač zapnutý, jsou šířeny informace o kontextu. Standardně je tento přepínač zapnutý. Chcete-li zabránit šíření informací o kontextu, změňte hodnotu na **false**.

Když je povoleno šíření kontextu, smí informace ze záhlaví plynout do zprávy nebo do cílové komponenty. Chcete-li zapnout či vypnout šíření kontextu, zadejte pro atribut contextPropagationEnabled vazeb importu a exportu hodnotu **true** nebo **false**. Například:

```
<esbBinding xsi:type="eis:JMSImportBinding" contextProgagationEnabled="true">
```

Výchozí hodnota je **true**.

Odstraňování problémů s generickými vazbami služby JMS:

Problémy s generickými vazbami služby JMS lze diagnostikovat a opravit.

Výjimky implementace

Jako odpověď na různé chybové stavy může generická implementace importu a exportu JMS vracet jeden ze dvou typů výjimek:

- Obchodní výjimka služby: Tato výjimka je vrácena, pokud došlo k poruše určené na obchodním rozhraní služby (typ portu WSDL).
- Výjimka za běhu služby: Generuje se ve všech ostatních případech. Ve většině případů bude výjimka příčina obsahovat původní výjimku (JMSException).

Odstraňování problémů s vypršením generické zprávy JMS

Zpráva požadavku poskytovatele JMS podléhá vypršení platnosti.

Vypršení platnosti požadavku odkazuje na vypršení platnosti zprávy požadavku poskytovatele JMS při dosažení času JMSExpiration na zprávě požadavku. Stejně jako u jiných vazeb JMS zpracovává generická vazba JMS vypršení platnosti požadavku tím, že na zprávě zpětného volání podané importem nastaví stejné vypršení platnosti jako pro odchozí požadavek. Oznámení k vypršení platnosti zprávy zpětného volání bude indikovat, že došlo k vypršení platnosti zprávy požadavku a klient by měl být upozorněn prostřednictvím obchodní výjimky.

Při přesunutí cíle zpětného volání k poskytovateli jiného dodavatele však není tento typ vypršení platnosti požadavku podporován.

Vypršení platnosti odezvy odkazuje na vypršení platnosti zprávy odezvy poskytovatele JMS při dosažení času JMSExpiration na zprávě odezvy.

Vypršení platnosti odezvy není pro generickou vazbu JMS podporováno, protože není definováno přesné chování poskytovatele JMS jiného dodavatele při vypršení platnosti. Můžete však zkontrolovat, zda nedošlo k vypršení platnosti odezvy při jejím přijetí.

Pro odchozí zprávy požadavku se hodnota JMSExpiration vypočítá z času čekání a z hodnot requestExpiration v asyncHeader, jsou-li nastaveny.

Odstraňování problémů s chybami generické továrny připojení JMS

Když definujete určité typy továren připojení v generickém poskytovateli JMS, můžete při pokusu o spuštění aplikace obdržet chybovou zprávu. Abyste předešli tomuto problému, můžete upravit externí továrnu připojení.

Při spuštění aplikace můžete obdržet následující chybovou zprávu:

```
MDB Listener Port JMSConnectionFactory type does not match
JMSDestination type (Typ JMSConnectionFactory portu modulu listener MDB neodpovídá typu JMSDestination)
```

K tomuto problému může dojít, když definujete externí továrny připojení. Konkrétně může k výjimce dojít, když vytvoříte továrnu připojení tématu JMS 1.0.2 místo (unifikované) továrny připojení JMS 1.1 (tj. továrny, která může podporovat jak komunikaci Odesilatel-příjemce, tak komunikaci publikování-odběru).

K vyřešení tohoto problému postupujte takto:

1. Přejděte na generického poskytovatele JMS, kterého používáte.
2. Nahraďte továrnu připojení tématu JMS 1.0.2, kterou jste definovali, (unifikovanou) továrnou připojení JMS 1.1.

Když spustíte aplikaci s nově definovanou továrnou připojení JMS 1.1, již byste neměli obdržet chybovou zprávu.

Generické zprávy SCA založené na JMS se nezobrazují ve správci událostí se selháním

Pokud dojde k selhání zpráv SCA vytvořených prostřednictvím generické interakce JMS, měly by se tyto zprávy objevit ve správci událostí se selháním. Pokud se tyto zprávy ve správci událostí se selháním neobjeví, zkontrolujte, zda je hodnota vlastnosti maximálního počtu pokusů základního portu modulu listener rovna nebo větší než 1. Nastavení této hodnoty na 1 a více umožní interakci se správcem událostí se selháním během vyvolání SCA pro generické vazby JMS.

Zpracování výjimek:

Způsob, jakým je vazba nakonfigurována, určuje, jak jsou zpracovány výjimky, které jsou způsobeny popisovači dat nebo vázáními dat. Chování systému při vrácení takové výjimky je dále určeno také charakterem mediačního toku.

Když vaše vazba volá popisovač dat nebo vázání dat, může dojít k nejrůznějším problémům. Například může popisovač dat obdržet zprávu s poškozeným informačním obsahem nebo se může pokusit přečíst zprávu v nesprávném formátu.

Způsob, jakým vaše vazba ošetří takovou výjimku, je určen tím, jak implementujete daný popisovač dat nebo vázání dat. Doporučené chování je navrhnout vázání dat tak, aby vracelo výjimku **DataBindingException**.

Podobná situace je i u popisovače dat. Protože je popisovač dat vyvolán vázáním dat, je jakákoli výjimka popisovače dat zabalena do výjimky vázání dat. Proto je výjimka **DataHandlerException** hlášena jako **DataBindingException**.

Když je vrácena jakákoli výjimka za běhu, včetně výjimky **DataBindingException**:

- Pokud je mediační tok nakonfigurován jako transakční, je zpráva JMS standardně uložena do komponenty Failed Event Manager pro ruční přehrání nebo odstranění.

  Poznámka: Režim zotavení pro vazbu je možné změnit tak, aby byla zpráva namísto uložení do komponenty Failed Event Manager odvolána.
- Pokud mediační tok není transakční, je výjimka zaprotokolována a zpráva je ztracena.

Podobná situace je i u popisovače dat. Protože je popisovač dat volán vázáním dat, je výjimka popisovače dat vytvořena uvnitř výjimky vázání dat. Proto je výjimka **DataHandlerException** hlášena jako **DataBindingException**.

Vazby WebSphere MQ JMS

Vazba JMS WebSphere MQ zajišťuje integraci s externími aplikacemi, které používají poskytovatele na bázi WebSphere MQ JMS.

Pomocí vazeb exportu a importu produktu WebSphere MQ JMS můžete zajistit integraci s externími systémy JMS nebo MQ JMS přímo ze svého prostředí serveru. Tím odpadá nutnost používat funkce Propojení MQ nebo Propojení klienta sběrnice SIBus.

Když vstoupí komponenta do interakce se službou založenou na WebSphere MQ JMS prostřednictvím importu, vazba importu WebSphere MQ JMS využije cíl, kam budou odeslána data, a cíl, kde je možné přijmout odpověď. Převod dat do zprávy JMS i z ní se provádí prostřednictvím komponenty Edge popisovače dat nebo vazby dat JMS.

Když modul SCA poskytuje službu klientům WebSphere MQ JMS, používá vazba exportu WebSphere MQ JMS cíl, kde je možné přijmout požadavek a odeslat odpověď. Převod dat do zprávy JMS i z ní se provádí prostřednictvím popisovače dat nebo vázáním dat JMS.

Selektor funkcí zajišťuje mapování na operaci na cílové komponentě, která má být vyvolána.

Přehled vazeb WebSphere MQ JMS:

Vazba WebSphere MQ JMS zajišťuje integraci s externími aplikacemi, které používají poskytovatele WebSphere MQ JMS.

Administrativní úlohy WebSphere MQ

Od administrátora systému WebSphere MQ se očekává, že vytvoří základní produkt WebSphere MQ Queue Manager, který budou používat vazby WebSphere MQ JMS, před spuštěním aplikace obsahující tyto vazby.

Vazby importu produktu WebSphere MQ JMS

Import produktu WebSphere MQ JMS umožňuje komponentám ve vašem modulu SCA komunikovat se službami poskytovanými poskytovateli na bázi WebSphere MQ JMS. Musíte používat podporovanou verzi produktu WebSphere MQ. Podrobné požadavky na hardware a software naleznete na stránkách podpory IBM.

Jsou podporovány dva typy scénářů využití vazeb importu produktu WebSphere MQ JMS, v závislosti na typu vyvolávané operace:

- Jednosměrný: Import WebSphere MQ JMS vloží zprávu do cíle operace odeslání nakonfigurovaného v dané vazbě importu. Do pole replyTo v záhlaví JMS se nic nepošle.
- Obousměrný (požadavek-odezva): Import WebSphere MQ JMS vloží zprávu do cíle operace odeslání.

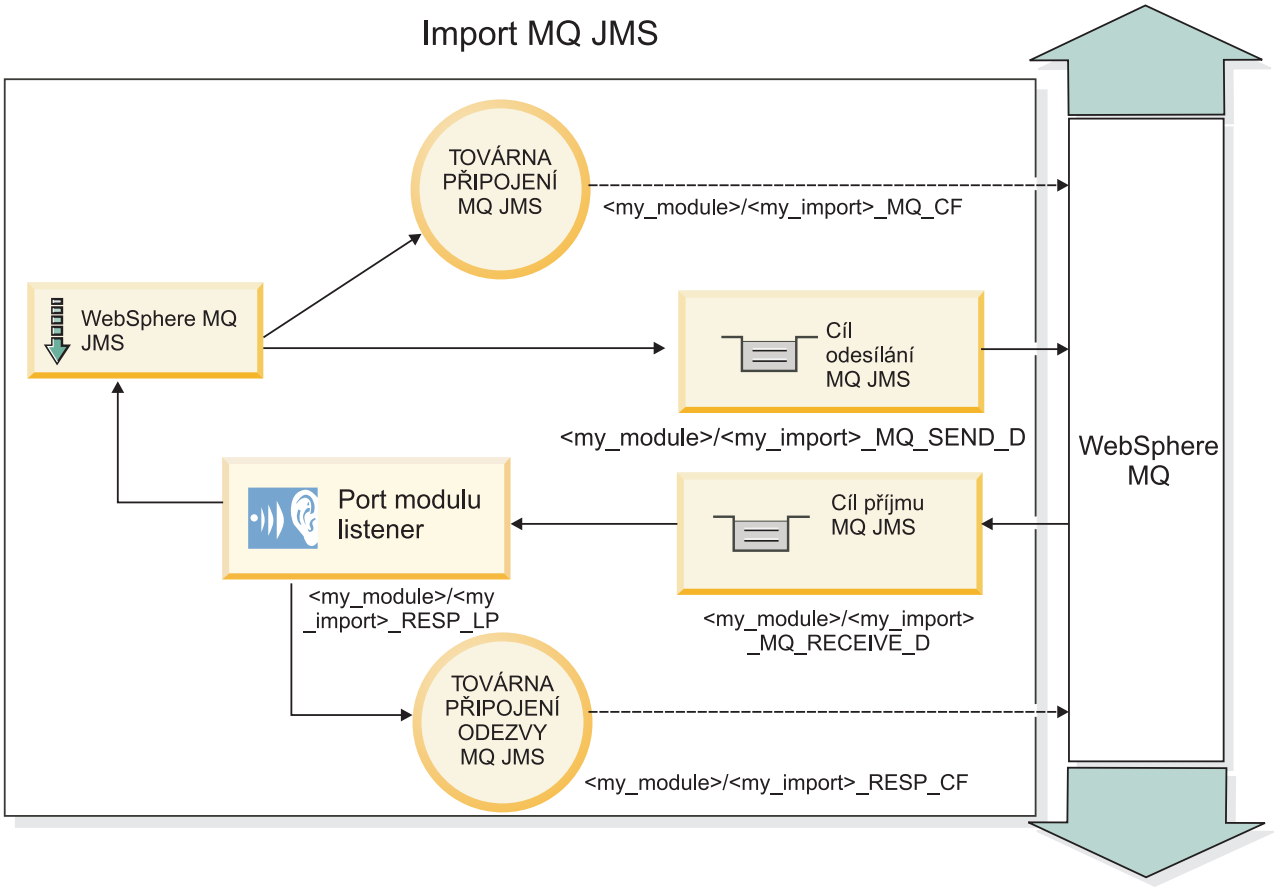
  Cíl operace příjmu (receive) je nastaven v poli záhlaví replyTo. Je implementován objekt typu message-driven bean (MDB), aby naslouchal cíli operace příjmu (receive), a když je přijata odpověď, MDB předá tuto odpověď zpět komponentě.

  Vazbu importu lze nakonfigurovat (pomocí pole **Schéma korelace odpovědí** v produktu Integration Designer), aby očekávala ID korelace zprávy odpovědi zkopírované z ID zprávy požadavku (výchozí nastavení), nebo z ID korelace zprávy požadavku.

Pro jednosměrné i obousměrné scénáře použití lze určit dynamické a statické vlastnosti záhlaví. Statické vlastnosti lze nastavit z vazby metody importu JMS. Některé z těchto vlastností mají zvláštní význam pro běhové prostředí SCA JMS.

Je důležité uvědomit si, že WebSphere MQ JMS je asynchronní vazba. Pokud volající komponenta vyvolá import produktu WebSphere MQ JMS synchronně (pro obousměrnou operaci), je volající komponenta zablokována, dokud služba JMS nevrátí odpověď.

Obrázek 37 na stránce 124 ukazuje, jak je import propojen s externí službou.



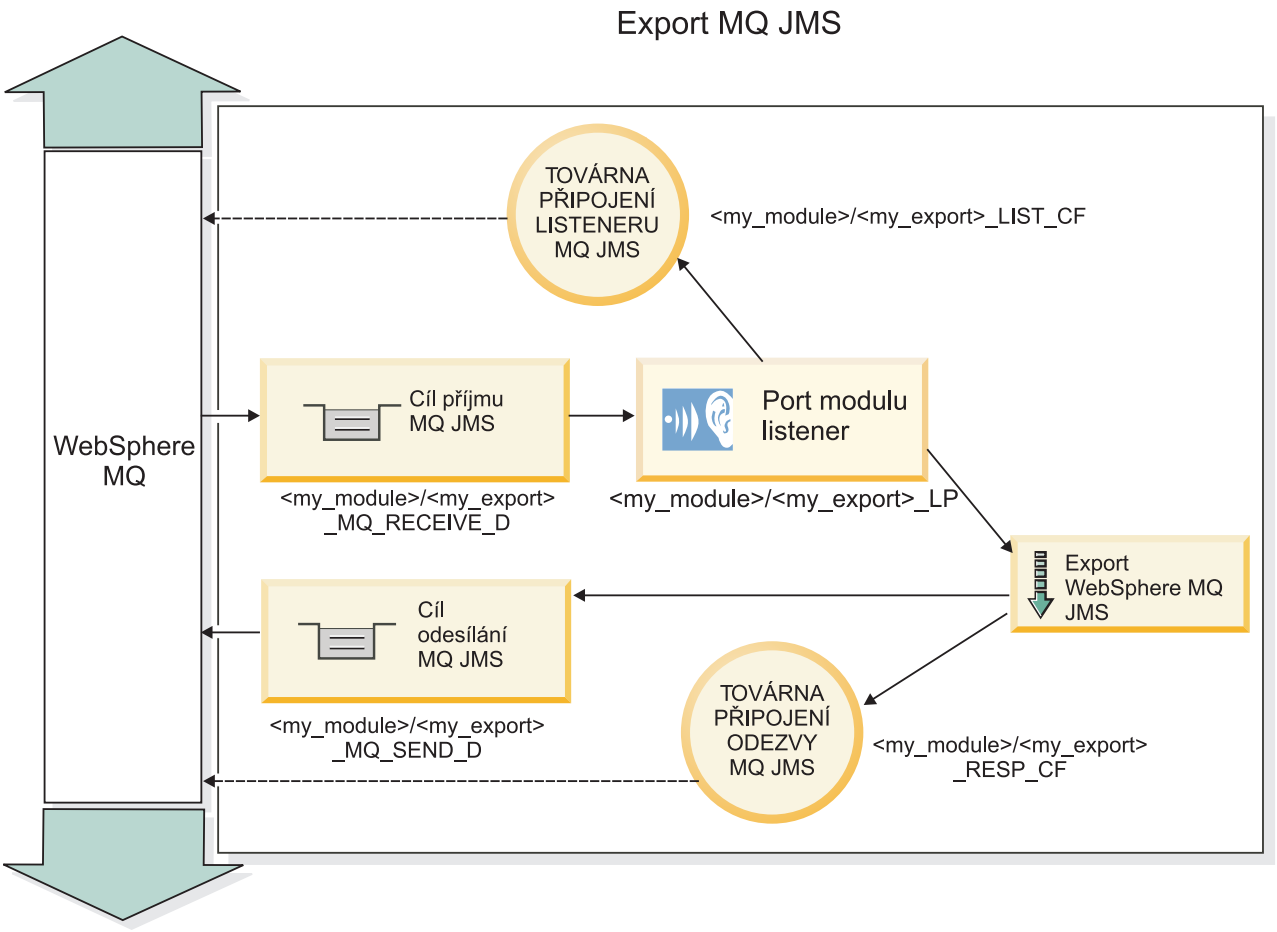
Obrázek 82. Prostředky vazby importu produktu WebSphere MQ JMS

Vazby exportu WebSphere MQ JMS

Vazba exportu produktu WebSphere MQ JMS zajišťuje modulům SCA prostředky pro poskytování služeb externím aplikacím JMS na poskytovateli JMS na bázi WebSphere MQ.

Je implementován MDB, aby naslouchal požadavkům přicházejícím do cíle operace příjmu (receive) určeného ve vazbě exportu. Cíl určený v poli odeslání (send) se používá k odeslání odpovědi na příchozí požadavek, pokud vyvolaná komponenta poskytuje odpověď. Cíl určený v poli replyTo zprávy odpovědi přepíše cíl určený v poli odeslání (send).

Obrázek 38 na stránce 125 ukazuje, jak je externí žadatel propojen s exportem.



Obrázek 83. Prostředky vazeb exportu produktu WebSphere MQ JMS

Poznámka: Obrázek 37 na stránce 124 a Obrázek 38 na stránce 125 ukazují, jak je aplikace z předchozí verze produktu IBM Business Process Manager propojena s externí službou. U aplikací vyvinutých pro produkt IBM Business Process Manager verze 7.0 se místo Portu modulu listener a Továrny připojení používá Specifikace aktivace.

Klíčové vlastnosti vazeb WebSphere MQ JMS:

Klíčové vlastnosti vazeb WebSphere MQ JMS zahrnují záhlaví, artefakty Java EE a vytvoření prostředky Java EE.

Záhlaví

Záhlaví platformy JMS obsahuje mnoho předdefinovaných polí obsahujících hodnoty používané klientem i poskytovatelem k identifikaci a směrování zpráv. Vlastnosti vazby můžete použít ke konfiguraci těchto záhlaví s pevnými hodnotami nebo je možné záhlaví určit dynamicky za běhu.

JMSCorrelationID
Odkazuje na související zprávu. Běžně je toto pole nastaveno na řetězec identifikátoru zprávy, na níž je odpovídáno.

TargetFunctionName
Toto záhlaví používá jeden z dodaných selektorů funkce k identifikaci operace, která je vyvolávána. Nastavení vlastnosti záhlaví TargetFunctionName ve zprávách zaslaných na export JMS umožňuje použití tohoto selektoru funkce. Vlastnost je možné nastavit přímo v aplikacích klienta JMS nebo při připojení importu s vazbou JMS k takovému exportu. V tom případě by měla být vazba importu JMS konfigurována tak, aby nastavila záhlaví TargetFunctionName pro každou operaci v rozhraní na název operace.

Schémata korelace

Vazby WebSphere MQ JMS poskytují různá schémata korelace, která jsou použita k určení toho, jak korelovat zprávy požadavku se zprávami odpovědi.

RequestMsgIDToCorrelID
Položka JMSMessageID je zkopírována do pole JMSCorrelationID. Toto je výchozí nastavení.

RequestCorrelIDToCorrelID
Položka JMSCorrelationID je kopírována do pole JMSCorrelationID.

Prostředky Java EE

Mnoho prostředků Java EE je vytvářeno, když je import MQ JMS implementován do prostředí Java EE.

Parametry

MQ Connection Factory
Používaná klienty k vytvoření připojení k poskytovateli MQ JMS.

Response Connection Factory
Používá ji běhové prostředí MQ JMS architektury SCA, když je místo určení odeslání na jiném produktu Queue Manager než cíl příjmu.

Activation specification
Specifikace aktivace MQ JMS je přidružena k jednomu nebo více objektům typu message-driven bean a poskytuje konfiguraci, která je pro ně nezbytná pro příjem zpráv.

Destinations

- Cíl odesílání:
  - Importy: Kam je zasílána odchozí zpráva nebo požadavek.
  - Exporty: Kam bude zaslána zpráva odpovědi v případě, že není nahrazena polem záhlaví JMSReplyTo příchozí zprávy.
- Cíl příjmu:
  - Importy: Kam by měla být umístěna příchozí zpráva či odpověď.
  - Exporty: Kam by měla být umístěna příchozí zpráva či zpráva požadavku.

Záhlaví JMS:

Zpráva JMS obsahuje dva typy záhlaví: systémová záhlaví JMS a různé vlastnosti JMS. K oběma typům záhlaví lze přistupovat buď v mediačním modulu v objektu SMO (Service Message Object), nebo prostřednictvím rozhraní API ContextService.

Systémové záhlaví JMS

Systémové záhlaví JMS je v objektu SMO představováno prvkem JMSHeader, který obsahuje všechna pole, jež se obvykle vyskytují v záhlaví JMS. Ačkoli je lze upravit v mediaci (nebo službě ContextService), některá pole systémových záhlaví JMS nastavená v objektu SMO se nebudou šířit v odchozí zprávě JMS, protože jsou přepsána systémovými nebo statickými hodnotami.

Klíčová pole v systémovém záhlaví JMS, která lze aktualizovat v mediaci (nebo službě ContextService):

- **JMSType** a **JMSCorrelationID** – hodnoty specifických předdefinovaných vlastností záhlaví zprávy.
- **JMSDeliveryMode** – hodnoty způsobu doručení persistent (perzistentní) nebo non-persistent (dočasný); výchozí je persistent).
- **JMSPriority** – hodnota priority (0 až 9; výchozí je JMS_Default_Priority.)

Vlastnosti JMS

Vlastnosti JMS jsou v objektu SMO reprezentovány položkami seznamu Vlastnosti. Vlastnosti lze přidávat, aktualizovat nebo odstraňovat v mediaci nebo prostřednictvím rozhraní API ContextService.

Vlastnosti lze také nastavit staticky ve vazbě JMS. Vlastnosti, které jsou nastaveny staticky, přepisují nastavení (se stejným názvem), která jsou nastavena dynamicky.

Uživatelské vlastnosti šířené z jiných vazeb (například jako vazba HTTP) budou výstupem ve vazbě JMS jako vlastnosti JMS.

Nastavení šíření záhlaví

Šíření systémového záhlaví a vlastností JMS buď z příchozí zprávy JMS do následných komponent, nebo z předchozích komponent do odchozí zprávy JMS lze řídit příznakem Šířit záhlaví protokolů pro danou vazbu.

Když je nastaven příznak Šířit záhlaví protokolů, je informacím o záhlaví dovoleno plynout do zprávy nebo do cílové komponenty, jak popisuje následující seznam:

- Požadavek exportu JMS

  Záhlaví JMS přijaté ve zprávě bude šířeno do cílové komponenty prostřednictvím kontextové služby. Vlastnosti JMS přijaté ve zprávě budou šířeny do cílové komponenty prostřednictvím kontextové služby.
- Odpověď exportu JMS

  Všechna pole záhlaví JMS nastavená v kontextové službě budou použita v odchozí zprávě, pokud nejsou přepsána statickými vlastnostmi nastavenými pro vazbu exportu JMS. Všechny vlastnosti nastavené v kontextové službě budou použity v odchozí zprávě, pokud nejsou přepsány statickými vlastnostmi nastavenými pro vazbu exportu JMS.
- Požadavek importu JMS

  Všechna pole záhlaví JMS nastavená v kontextové službě budou použita v odchozí zprávě, pokud nejsou přepsána statickými vlastnostmi nastavenými pro vazbu importu JMS. Všechny vlastnosti nastavené v kontextové službě budou použity v odchozí zprávě, pokud nejsou přepsány statickými vlastnostmi nastavenými pro vazbu importu JMS.
- Odpověď importu JMS

  Záhlaví JMS přijaté ve zprávě bude šířeno do cílové komponenty prostřednictvím kontextové služby. Vlastnosti JMS přijaté ve zprávě budou šířeny do cílové komponenty prostřednictvím kontextové služby.

Externí klienti:

Server může odesílat zprávy externím klientům nebo od nich zprávy přijímat pomocí vazeb WebSphere MQ JMS.

Externí klient (například webový portál nebo podnikový informační systém) může odeslat zprávu komponentě SCA v aplikaci prostřednictvím exportu nebo může být vyvolán komponentou SCA v aplikaci prostřednictvím importu.

Vazba exportu WebSphere MQ JMS implementuje objekty typu message-driven bean (MDB) za účelem naslouchání požadavkům přicházejícím do cíle operace příjmu (receive) určeného ve vazbě exportu. Cíl určený v poli odeslání (send) se používá k odeslání odpovědi na příchozí požadavek, pokud vyvolaná aplikace nějakou odpověď poskytne. Proto je externí klient schopen vyvolávat aplikace prostřednictvím vazby exportu.

Importy WebSphere MQ JMS se vážou k externím klientům a mohou jim doručit zprávu. Tato zpráva může, ale nemusí od externího klienta požadovat odpověď.

Další informace o způsobu interakcí s externími klienty pomocí produktu WebSphere MQ naleznete v Informačním centru tohoto produktu WebSphere MQ..

Odstraňování problémů vazeb produktu WebSphere MQ JMS:

Problémy s vazbami produktu WebSphere MQ JMS lze diagnostikovat a opravit.

Výjimky implementace

Jako odpověď na různé chybové stavy může implementace importu a exportu MQ JMS vracet jeden ze dvou typů výjimek:

- Obchodní výjimka služby: Tato výjimka je vrácena, pokud došlo k poruše určené na obchodním rozhraní služby (typ portu WSDL).
- Výjimka za běhu služby: Generuje se ve všech ostatních případech. Ve většině případů bude výjimka příčina obsahovat původní výjimku (JMSException).

  Například import očekává pouze jednu zprávu odezvy na každou zprávu požadavku. Pokud dorazí více než jedna odezva nebo pokud dorazí pozdní odezva (odezva, pro kterou došlo k vypršení odezvy SCA), je vyvolána výjimka za běhu služby. Transakce je odvolána a zpráva odezvy je vrácena zpět z fronty nebo zpracována správcem událostí se selháním.

Zprávy SCA založené na produktu WebSphere MQ JMS se nezobrazují ve správci událostí se selháním

Pokud dojde k selhání zpráv SCA vytvořených prostřednictvím interakce produktu WebSphere MQ JMS, měly by se tyto zprávy objevit ve správci událostí se selháním. Pokud se tyto zprávy ve správci událostí se selháním neobjeví, zkontrolujte, zda je hodnota vlastnosti maximálního počtu pokusů základního portu modulu listener rovna nebo větší než **1**. Nastavení této hodnoty na **1** a více umožní interakci se správcem událostí se selháním během vyvolání SCA pro vazby MQ JMS.

Scénáře nesprávného použití: Porovnání s vazbami produktu WebSphere MQ

Vazba produktu WebSphere MQ JMS je určena ke spolupráci s aplikacemi JMS implementovanými na produktu WebSphere MQ, který vystavuje zprávy podle modelu zpráv JMS. Import a export WebSphere MQ je však v podstatě určen ke spolupráci s nativními aplikacemi produktu WebSphere MQ a vystavuje úplný obsah těla zprávyWebSphere MQ mediacím.

Následující scénáře by měly být sestaveny pomocí vazby WebSphere MQ JMS, nikoli vazby WebSphere MQ:

- Vyvolání objektu JMS typu message-driven bean (MDB) z modulu SCA, kde je objekt typu message-driven bean implementován na poskytovateli produktu WebSphere MQ JMS. Použijte import produktu WebSphere MQ JMS.
- Povolení volání modulu SCA ze servletu komponenty Java EE nebo EJB pomocí JMS. Použijte export produktu WebSphere MQ JMS.
- Mediace obsahu JMS MapMessage při přenosu přes produkt WebSphere MQ. Použijte export a import produktu WebSphere MQ JMS ve spojení s příslušným popisovačem dat nebo vázáním dat.

Existují situace, ve kterých lze očekávat spolupráci vazeb produktů WebSphere MQ a WebSphere MQ JMS. Zejména při přemostění mezi aplikacemi WebSphere MQ patřícími a nepatřícími k prostředí Java EE použijte export WebSphere MQ a import WebSphere MQ JMS (nebo naopak) ve spojení s příslušnými vázáními dat nebo mediačními moduly (nebo obojím).

Zpracování výjimek:

Způsob, jakým je vazba nakonfigurována, určuje, jak jsou zpracovány výjimky, které jsou způsobeny popisovači dat nebo vázáními dat. Chování systému při vrácení takové výjimky je dále určeno také charakterem mediačního toku.

Když vaše vazba volá popisovač dat nebo vázání dat, může dojít k nejrůznějším problémům. Například může popisovač dat obdržet zprávu s poškozeným informačním obsahem nebo se může pokusit přečíst zprávu v nesprávném formátu.

Způsob, jakým vaše vazba ošetří takovou výjimku, je určen tím, jak implementujete daný popisovač dat nebo vázání dat. Doporučené chování je navrhnout vázání dat tak, aby vracelo výjimku **DataBindingException**.

Podobná situace je i u popisovače dat. Protože je popisovač dat vyvolán vázáním dat, je jakákoli výjimka popisovače dat zabalena do výjimky vázání dat. Proto je výjimka **DataHandlerException** hlášena jako **DataBindingException**.

Když je vrácena jakákoli výjimka za běhu, včetně výjimky **DataBindingException**:

- Pokud je mediační tok nakonfigurován jako transakční, je zpráva JMS standardně uložena do komponenty Failed Event Manager pro ruční přehrání nebo odstranění.

  Poznámka: Režim zotavení pro vazbu je možné změnit tak, aby byla zpráva namísto uložení do komponenty Failed Event Manager odvolána.
- Pokud mediační tok není transakční, je výjimka zaprotokolována a zpráva je ztracena.

Podobná situace je i u popisovače dat. Protože je popisovač dat volán vázáním dat, je výjimka popisovače dat vytvořena uvnitř výjimky vázání dat. Proto je výjimka **DataHandlerException** hlášena jako **DataBindingException**.

Vazby WebSphere MQ

Vazba WebSphere MQ zajišťuje konektivitu architektury SCA (Service Component Architecture) s aplikacemi WebSphere MQ.

Pomocí vazeb exportu a importu WebSphere MQ můžete zajistit integraci se systémem založeným na produktu WebSphere MQ přímo z prostředí vašeho serveru. Tím odpadá nutnost používat funkce Propojení MQ nebo Propojení klienta sběrnice SIBus.

Když vstoupí komponenta do interakce se službou WebSphere MQ prostřednictvím importu, vazba importu WebSphere MQ využije frontu, kam budou odeslána data, a frontu, kde je možné přijmout odpověď.

Když modul SCA poskytuje službu klientům WebSphere MQ, používá vazba exportu WebSphere MQ frontu, kde je možné přijmout požadavek a odeslat odpověď. Selektor funkcí zajišťuje mapování na operaci na cílové komponentě, která má být vyvolána.

Převod dat informačního obsahu do zprávy MQ i z ní se provádí prostřednictvím popisovače dat nebo vázáním dat MQ. Převod dat záhlaví do zprávy MQ i z ní se provádí prostřednictvím vázáním dat záhlaví MQ.

Informace o podporovaných verzích WebSphere MQ viz webová stránka se systémovými požadavky.

Přehled vazeb WebSphere MQ:

Vazba WebSphere MQ zajišťuje integraci a nativními aplikacemi založenými na MQ.

Administrativní úlohy WebSphere MQ

Od administrátora systému WebSphere MQ se očekává, že vytvoří základní produkt WebSphere MQ Queue Manager, který budou používat vazby WebSphere MQ, před spuštěním aplikace obsahující tyto vazby.

Administrativní úlohy WebSphere

Musíte nastavit vlastnost **Cesta k nativní knihovně** adaptéru prostředku MQ ve Websphere na verzi WebSphere MQ podporovanou serverem a restartovat server. To zajistí, že budou využívány knihovny podporované verze WebSphere MQ. Podrobné požadavky na hardware a software naleznete na stránkách podpory IBM.

Vazby importu WebSphere MQ

Vazba importu WebSphere MQ umožňuje komponentám ve vašem modulu SCA komunikovat se službami poskytovanými externími aplikacemi založenými na WebSphere MQ. Musíte používat podporovanou verzi produktu WebSphere MQ. Podrobné požadavky na hardware a software naleznete na stránkách podpory IBM.

Interakce s externími systémy WebSphere MQ zahrnuje použití front pro odesílání požadavků a příjem odpovědí.

Jsou podporovány dva typy scénářů využití vazeb importu WebSphere MQ, v závislosti na typu vyvolávané operace:

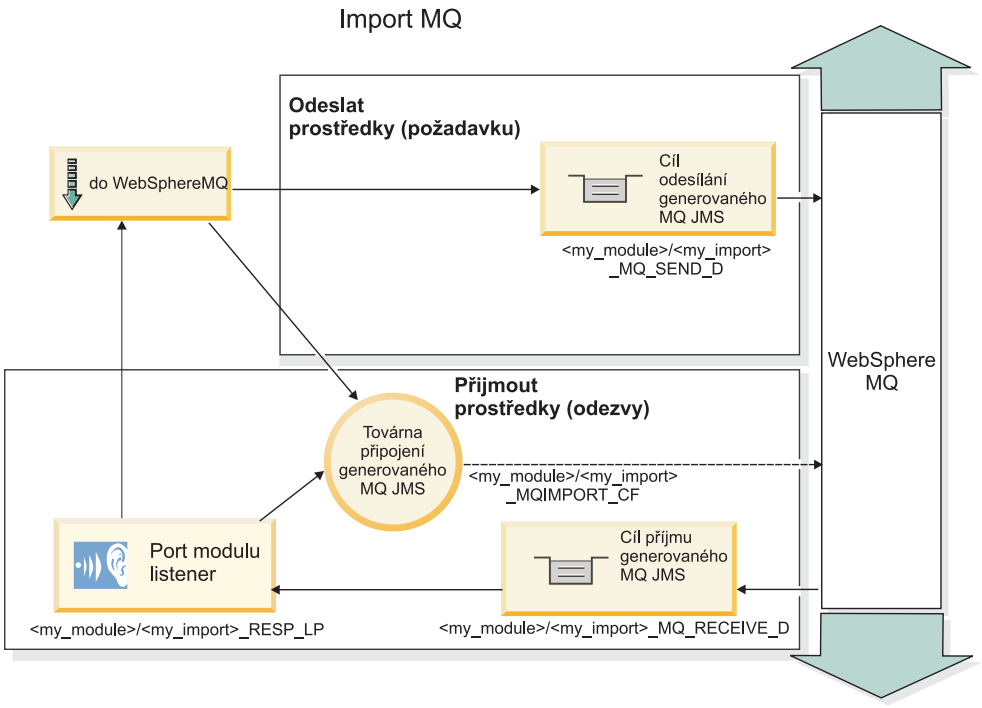
- Jednosměrný: Import WebSphere MQ vloží zprávu do fronty nakonfigurované v poli **Cílová fronta odeslání** vazby importu. Do pole replyTo v záhlaví MQMD se nic nepošle.
- Obousměrný (požadavek-odezva): Import WebSphere MQ vloží zprávu do fronty nakonfigurované v poli **Cílová fronta odeslání**.

  Fronta operace příjmu (receive) je nastavena v poli záhlaví replyTo MQMD. Je implementován objekt typu message-driven bean (MDB), aby naslouchal frontě operace příjmu (receive), a když je přijata odpověď, MDB předá tuto odpověď zpět komponentě.

  Vazbu importu lze nakonfigurovat (pomocí pole **Schéma korelace odpovědí**), aby očekávala ID korelace zprávy odpovědi zkopírované z ID zprávy požadavku (výchozí nastavení), nebo z ID korelace zprávy požadavku.

Je důležité uvědomit si, že WebSphere MQ je asynchronní vazba. Pokud volající komponenta vyvolá import WebSphere MQ synchronně (pro obousměrnou operaci), je volající komponenta zablokována, dokud služba WebSphere MQ nevrátí odpověď.

Obrázek 39 na stránce 130 ukazuje, jak je import propojen s externí službou.



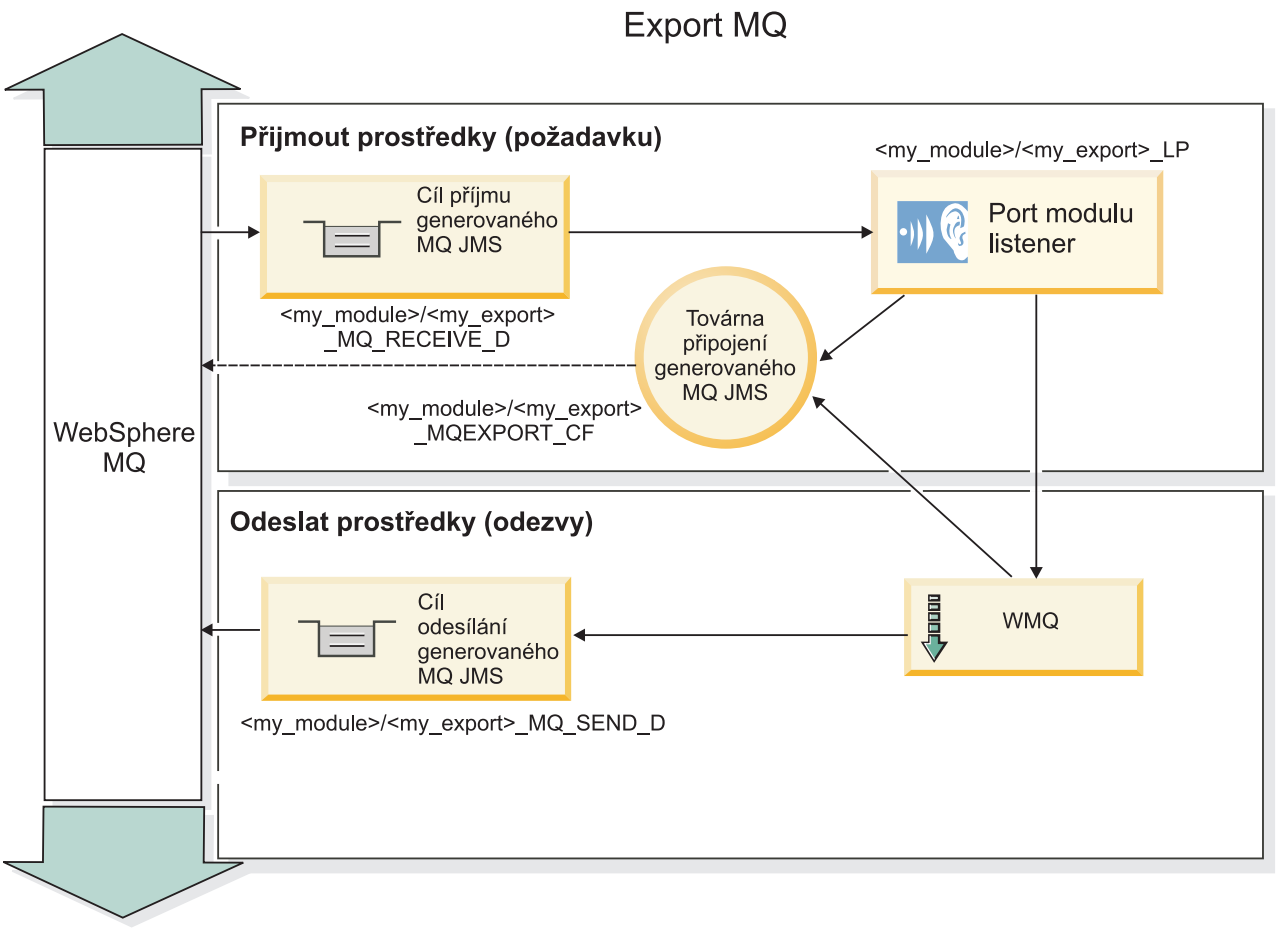
Obrázek 84. Prostředky vazby importu WebSphere MQ

Vazby exportu WebSphere MQ

Vazba exportu WebSphere MQ zajišťuje modulům SCA prostředky pro poskytování služeb externím aplikacím založeným na WebSphere MQ.

Je implementován MDB, aby naslouchal požadavkům přicházejícím do **cílové fronty příjmu** určené ve vazbě exportu. Fronta určená v poli **Cílová fronta odeslání** se používá k odeslání odpovědi na příchozí požadavek, pokud vyvolaná Komponenta nějakou odpověď poskytuje. Fronta určená v poli replyTo zprávy odpovědi přepíše frontu určenou v poli **Cílová fronta odeslání**.

Obrázek 40 na stránce 131 ukazuje, jak je externí žadatel propojen s exportem.



Obrázek 85. Prostředky vazeb exportu WebSphere MQ

Poznámka: Obrázek 39 na stránce 130 a Obrázek 40 na stránce 131 ukazují, jak je aplikace z předchozí verze produktu IBM Business Process Manager propojena s externí službou. U aplikací vyvinutých pro produkt IBM Business Process Manager verze 7.x se místo Portu modulu listener a Továrny připojení používá Specifikace aktivace.

Klíčové vlastnosti vazby WebSphere MQ:

Klíčové vlastnosti vazby WebSphere MQ zahrnují záhlaví, artefakty Java EE a vytvořené prostředky Java EE.

Schémata korelace

Aplikace požadavku/odezvy WebSphere MQ může využít jednu z mnoha technik ke korelaci zpráv odpovědi s požadavky, které jsou sestaveny kolem polí MQMD MessageID a CorrelID. Ve většině případů umožní žadatel správci front vybrat položku MessageID a očekává, že odpovídající aplikace tuto položku zkopíruje do položky CorrelID odpovědi. Většinou žadatel i odpovídající aplikace implicitně ví, která technika korelace se používá. Občas odpovídající aplikace přidá různé příznaky do pole Sestava požadavku a ty popisují, jak s těmito poli pracovat.

Vazby exportu zpráv WebSphere MQ lze konfigurovat pomocí následujících voleb:

Volby položky MsgId odezvy:

New MsgID
Umožní správci front vybrat jedinečné MsgId pro odezvu (výchozí).

Copy from Request MsgID
Zkopíruje pole MsgId z pole MsgId v požadavku.

Copy from SCA message
Nastavuje, že MsgId bude stejné, jako to v záhlaví WebSphere MQ ve zprávě požadavku SCA, nebo nechá správce front definovat nové Id, pokud hodnota neexistuje.

As Report Options
Prozkoumává pole Sestava MQMD v požadavku a hledá radu, jak pracovat s MsgId. Podporovány jsou volby MQRO_NEW_MSG_ID a MQRO_PASS_MSG_ID a chovají se jako položky Nové MsgId, respektive Copy from Request MsgI.

Volby položky CorrelId odezvy:

Copy from Request MsgID
Zkopíruje pole CorrelId z pole MsgId v požadavku (výchozí).

Copy from Request CorrelID
Zkopíruje pole CorrelId z pole CorrelId v požadavku.

Copy from SCA message
Nastavuje, že CorrelId bude stejné, jako to v záhlaví WebSphere MQ ve zprávě požadavku SCA, nebo jej nechává prázdné v případě, že hodnota neexistuje.

As Report Options
Prozkoumává pole Sestava MQMD v požadavku a hledá radu, jak pracovat s CorrelId. Podporovány jsou volby MQRO_COPY_MSG_ID_TO_CORREL_ID a MQRO_PASS_CORREL_ID a chovají se jako položky Copy from Request MsgID, respektive Copy from Request CorrelID.

Vazby importu zpráv WebSphere MQ lze konfigurovat pomocí následujících voleb:

Volby položky MsgId požadavku:

New MsgID
Umožní správci front vybrat jedinečné MsgId pro žádost (výchozí).

Copy from SCA message
Nastavuje, že MsgId bude stejné, jako to v záhlaví WebSphere MQ ve zprávě požadavku SCA, nebo nechá správce front definovat nové Id, pokud hodnota neexistuje.

Volby korelace odezvy:

Response has CorrelID copied from MsgId
Očekává, že zpráva odpovědi bude mít nastaveno pole CorrelId, na základě položky MsgId požadavku (výchozí).

Response has MsgID copied from MsgId
Očekává, že zpráva odpovědi bude mít nastaveno pole MsgId, na základě položky MsgId požadavku.

Response has CorrelID copied from CorrelId
Očekává, že zpráva odpovědi bude mít nastaveno pole CorrelId, na základě položky CorrelId požadavku.

Prostředky Java EE

Mnoho prostředků Java EE je vytvářeno, když je vazba WebSphere MQ implementována do prostředí Java EE.

Parametry

MQ Connection Factory
Používaná klienty k vytvoření připojení k poskytovateli WebSphere MQ.

Response Connection Factory
Používá ji běhové prostředí MQ architektury SCA, když je místo určení odeslání na jiném produktu Queue Manager než cíl příjmu.

Activation specification

Specifikace aktivace MQ JMS je přidružena k jednomu nebo více objektům typu message-driven bean a poskytuje konfiguraci, která je pro ně nezbytná pro příjem zpráv.

Destinations

- Cíl odesílání: kam je zasílán požadavek nebo odchozí zpráva (import); kam bude zaslána zpráva odpovědi (export) v případě, že není nahrazena polem záhlaví ReplyTo MQMD v příchozí zprávě.
- Cíl příjmu: Kam má být umístěna příchozí zpráva či odezva/požadavek.

Záhlaví WebSphere MQ:

Záhlaví WebSphere MQ zahrnují určité konvence pro převod zpráv SCA (Service Component Architecture).

Zprávy WebSphere MQ sestávají ze záhlaví systému (MQMD), nula nebo více jiných záhlaví MQ (systémových nebo přizpůsobených) a těla zprávy. Pokud zpráva obsahuje více záhlaví zprávy, je důležité jejich pořadí.

Každé záhlaví obsahuje informace popisující strukturu následujícího záhlaví. Záhlaví systému MQMD popisuje první záhlaví.

Způsob analýzy záhlaví MQ

Vázání dat záhlaví MQ se používá k analýza záhlaví MQ. Následující záhlaví jsou podporována automaticky:

- MQRFH.
- MQRFH2.
- MQCIH.
- MQIIH.

Záhlaví začínající na **MQH** jsou zpracována různě. Specifická pole záhlaví nejsou analyzována a zůstávají ve formě neanalyzovaných bajtů.

V případě záhlaví MQ můžete zapsat vlastní vázání dat záhlaví MQ, která budou tato záhlaví analyzovat.

Způsob přístupu k záhlavím MQ

K záhlavím MQ lze v produktu přistoupit dvěma způsoby:

- Prostřednictvím objektu SMO (Service Message Object) v rámci mediace.
- Prostřednictvím rozhraní API ContextService.

Záhlaví MQ jsou interně reprezentována prvkem MQHeader objektu SMO. Prvek MQHeader je kontejner dat záhlaví, který rozšiřuje prvek MQControl, ale obsahuje prvek s hodnotou libovolného typu. Obsahuje záhlaví MQMD, MQControl (řídící informace těla zprávy MQ) a seznam dalších záhlaví MQ.

- Záhlaví MQMD představuje obsah popisu zprávy WebSphere MQ, kromě informací určujících strukturu a kódování těla.
- Záhlaví MQControl obsahuje informace určující strukturu a kódování těla zprávy.
- Záhlaví MQHeaders obsahuje seznam objektů MQHeader.

Řetězec záhlaví MQ je nesvázaný, takže v rámci objektu SMO nese každé záhlaví MQ své vlastní řídicí informace (CCSID, kódování a formát). Záhlaví lze snadno přidat či odebrat bez nutnosti pozměnění jiných dat záhlaví.

Nastavení polí v záhlaví MQMD

Záhlaví MQMD můžete aktualizovat pomocí rozhraní API kontextu nebo prostřednictvím objektu SMO (Service Message Object) v rámci mediace. Následující pole jsou automaticky šířena do odchozí zprávy MQ:

- Encoding.

- CodedCharacterSet.
- Format.
- Report.
- Expiry.
- Feedback.
- Priority.
- Persistence.
- CorrelId.
- MsgFlags.

Pokud chcete do odchozí zprávy MQ šířit následující vlastnosti, nakonfigurujte vazbu MQ na import či export:

MsgID
Volbu **ID zprávy požadavku** nastavte na hodnotu zkopírovat ze zprávy SCA.

MsgType
Zrušte zaškrtnutí políčka u volby **Nastavit typ zprávy pro operaci požadavek-odezva na MQMT_DATAGRAM nebo MQMT_REQUEST**.

ReplyToQ
Zrušte zaškrtnutí políčka vedle volby **Potlačit odpověď na frontu zprávy požadavku**.

ReplyToQMgr
Zrušte zaškrtnutí políčka vedle volby **Potlačit odpověď na frontu zprávy požadavku**.

Počínaje verzí 7.0 lze kontextová pole přepsat pomocí přizpůsobené vlastnosti z definice cíle rozhraní JNDI. Nastavením přizpůsobené vlastnosti MDCTX na hodnotu SET_IDENTITY_CONTEXT v cíli odeslání zajistíte šíření následujících polí do odchozí zprávy MQ:

- UserIdentifier.
- AppIdentityData.

Nastavením přizpůsobené vlastnosti MDCTX na hodnotu SET_ALL_CONTEXT v cíli odeslání zajistíte šíření následujících polí do odchozí zprávy MQ:

- UserIdentifier.
- AppIdentityData.
- PutApplType.
- PutApplName.
- ApplOriginData.

Některá pole nejsou do odchozí zprávy MQ šířena. Následující pole jsou přepsána během odesílání zprávy:

- BackoutCount.
- AccountingToken.
- PutDate.
- PutTime.
- Posunutí
- OriginalLength.

Statické přidání MQCIH do vazby WebSphere MQ:

Produkt IBM Business Process Manager podporuje statické přidání informace záhlaví MQCIH bez použití mediačního modulu.

Ke zprávě lze přidat informace záhlaví MQCIH několika různými způsoby (například pomocí mediačního primitiva Modul nastavení záhlaví). Mohlo by být užitečné přidávat tyto informace záhlaví staticky, bez použití dalšího mediačního modulu. Statické informace záhlaví, včetně názvu programu CICS, ID transakce a dalších podrobností záhlaví datového formátu, lze nadefinovat a přidat jako součást vazby WebSphere MQ.

WebSphere MQ, MQ CICS Bridge a CICS je pro statické přidávání informací záhlaví MQCIH nutné nakonfigurovat.

Ke konfiguraci importu WebSphere MQ se statickými hodnotami nezbytnými pro informace záhlaví MQCIH můžete použít produkt Integration Designer.

Při zpracování příchozí zprávy importem WebSphere MQ se provádí kontrola, zda je ve zprávě již přítomna informace záhlaví MQCIH. Pokud je informace záhlaví MQCIH k dispozici, statické hodnoty definované v importu WebSphere MQ se použijí k potlačení odpovídajících dynamických hodnot ve zprávě. Pokud MQCIH k dispozici není, tyto informace záhlaví se ve zprávě vytvoří a poté se přidají statické hodnoty definované v importu WebSphere MQ.

Statické hodnoty definované v importu WebSphere MQ jsou specifické pro danou metodu. Pro jiné metody v rámci téhož importu WebSphere MQ můžete zadat odlišné statické hodnoty MQCIH.

Pokud MQCIH specifické informace záhlaví postrádá, tento mechanizmus se k poskytování výchozích hodnot nepoužívá, protože statická hodnota definovaná v importu WebSphere MQ potlačí odpovídající hodnotu obsaženou v příchozí zprávě.

Externí klienti:

Produkt IBM Business Process Manager může odesílat zprávy externím klientům nebo od nich zprávy přijímat pomocí vazeb WebSphere MQ.

Externí klient (například webový portál nebo podnikový informační systém) může odeslat zprávu komponentě SCA v aplikaci prostřednictvím exportu nebo může být vyvolán komponentou SCA v aplikaci prostřednictvím importu.

Vazba exportu WebSphere MQ implementuje objekty typu message-driven bean (MDB) za účelem naslouchání požadavkům přicházejícím do cíle operace příjmu (receive) určeného ve vazbě exportu. Cíl určený v poli odeslání (send) se používá k odeslání odpovědi na příchozí požadavek, pokud vyvolaná aplikace nějakou odpověď poskytne. Proto je externí klient schopen vyvolávat aplikace prostřednictvím vazby exportu.

Importy WebSphere MQ se vážou k externím klientům a mohou jim doručit zprávu. Tato zpráva může, ale nemusí od externího klienta požadovat odpověď.

Další informace o způsobu interakcí s externími klienty pomocí produktu WebSphere MQ naleznete v Informačním centru tohoto produktu WebSphere MQ..

Odstraňování problémů vazeb WebSphere MQ:

Poruchy a podmínky selhání, ke kterým dojde u vazeb WebSphere MQ lze diagnostikovat a opravit.

Primární podmínky selhání

Primární podmínky selhání vazeb WebSphere MQ jsou určeny transakční sémantikou, konfigurací produktu WebSphere MQ, nebo odkazem na existující chování v jiných komponentách. Primární podmínky selhání zahrnují:

- Selhání při připojení ke správci front WebSphere MQ nebo ke frontě.

  Selhání při připojení k produktu WebSphere MQ při přijímání zpráv bude mít za následek selhání spuštění portu modulu listener MDB. Tato podmínka bude zaznamenána do protokolu produktu WebSphere Application Server. Trvalé zprávy zůstanou ve frontě produktu WebSphere MQ, dokud nebudou úspěšně načteny (nebo dokud nevyprší jejich platnost v produktu WebSphere MQ).

Selhání při připojení k produktu WebSphere MQ při odesílání odchozích zpráv způsobí odvolání transakce, která řídí odeslání.

- Selhání při analýze příchozí zprávy nebo při vytvoření odchozí zprávy.

  Selhání ve vázání dat způsobí odvolání transakce, která tuto práci řídí.
- Selhání při odesílání odchozí zprávy.

  Selhání při odesílání zprávy způsobí odvolání příslušné transakce.
- Vícenásobné nebo neočekávané zprávy odezvy.

  Import očekává pouze jednu zprávu odezvy na každou zprávu požadavku. Pokud dorazí více než jedna odezva nebo pokud dorazí pozdní odezva (odezva, pro kterou došlo k vypršení odezvy SCA), je vyvolána výjimka za běhu služby. Transakce je odvolána a zpráva odezvy je vrácena zpět z fronty nebo zpracována správcem událostí se selháním.

Scénáře nesprávného použití: Porovnání s vazbami WebSphere MQ JMS

Import a export WebSphere MQ je v podstatě určen ke spolupráci s nativními aplikacemi produktu WebSphere MQ a vystavuje úplný obsah těla zprávyWebSphere MQ mediacím. Vazba WebSphere MQ JMS je však určena ke spolupráci s aplikacemi JMS implementovanými na produktu WebSphere MQ, který vystavuje zprávy podle modelu zpráv JMS.

Následující scénáře by měly být sestaveny pomocí vazby WebSphere MQ JMS, nikoli vazby WebSphere MQ:

- Vyvolání objektu JMS typu message-driven bean (MDB) z modulu SCA, kde je objekt typu message-driven bean implementován na poskytovateli produktu WebSphere MQ JMS. Použijte import produktu WebSphere MQ JMS.
- Povolení volání modulu SCA ze servletu komponenty Java EE nebo EJB pomocí JMS. Použijte export produktu WebSphere MQ JMS.
- Mediace obsahu JMS MapMessage při přenosu přes produkt WebSphere MQ. Použijte export a import WebSphere MQ JMS ve spojení s příslušným vázáním dat.

Existují situace, ve kterých lze očekávat spolupráci vazeb produktů WebSphere MQ a WebSphere MQ JMS. Zejména při přemostění mezi aplikacemi WebSphere MQ patřícími a nepatřícími k prostředí Java EE použijte export WebSphere MQ a import WebSphere MQ JMS (nebo naopak) ve spojení s příslušnými vázáními dat nebo mediačními moduly (nebo obojím).

Nedoručené zprávy

Pokud produkt WebSphere MQ nemůže doručit zprávu na určené místo určení (například z důvodu chyb konfigurace), odešle namísto toho zprávu do nominované fronty nedoručených zpráv.

Přitom přidá na začátek těla zprávy záhlaví nedoručené zprávy. Toto záhlaví obsahuje příčiny selhání, původní místo určení a další informace.

Zprávy SCA založené na MQ se nezobrazují ve správci událostí se selháním

Pokud dojde k selhání zpráv SCA vytvořených prostřednictvím interakce WebSphere MQ, měly by se tyto zprávy objevit ve správci událostí se selháním. Pokud se tyto zprávy ve správci událostí se selháním neobjeví, zkontrolujte, zda je hodnota vlastnosti maximálního počtu pokusů základního místa určení WebSphere MQ větší než 1. Nastavení této hodnoty na 2 a více umožní interakci se správcem událostí se selháním během vyvolání SCA pro vazby WebSphere MQ.

Události MQ se selháním se přehrávají do chybného správce front

Když se má použít předdefinovaná továrna připojení pro odchozí připojení, vlastnosti připojení musí odpovídat vlastnostem definovaným ve specifikaci aktivace použité pro příchozí připojení.

Předdefinovaná továrna připojení se používá k vytvoření připojení při přehrávání události se selháním, a proto musí být nakonfigurována k použití stejného správce front,ze kterého byla zpráva původně přijata.

Zpracování výjimek:

Způsob, jakým je vazba nakonfigurována, určuje, jak jsou zpracovány výjimky, které jsou způsobeny popisovači dat nebo vázáními dat. Chování systému při vrácení takové výjimky je dále určeno také charakterem mediačního toku.

Když vaše vazba volá popisovač dat nebo vázání dat, může dojít k nejrůznějším problémům. Například může popisovač dat obdržet zprávu s poškozeným informačním obsahem nebo se může pokusit přečíst zprávu v nesprávném formátu.

Způsob, jakým vaše vazba ošetří takovou výjimku, je určen tím, jak implementujete daný popisovač dat nebo vázání dat. Doporučené chování je navrhnout vázání dat tak, aby vracelo výjimku **DataBindingException**.

Podobná situace je i u popisovače dat. Protože je popisovač dat vyvolán vázáním dat, je jakákoli výjimka popisovače dat zabalena do výjimky vázání dat. Proto je výjimka **DataHandlerException** hlášena jako **DataBindingException**.

Když je vrácena jakákoli výjimka za běhu, včetně výjimky **DataBindingException**:

- Pokud je mediační tok nakonfigurován jako transakční, je zpráva JMS standardně uložena do komponenty Failed Event Manager pro ruční přehrání nebo odstranění.

  Poznámka: Režim zotavení pro vazbu je možné změnit tak, aby byla zpráva namísto uložení do komponenty Failed Event Manager odvolána.
- Pokud mediační tok není transakční, je výjimka zaprotokolována a zpráva je ztracena.

Podobná situace je i u popisovače dat. Protože je popisovač dat volán vázáním dat, je výjimka popisovače dat vytvořena uvnitř výjimky vázání dat. Proto je výjimka **DataHandlerException** hlášena jako **DataBindingException**.

Omezení vazeb

Použití vazeb má jistá omezení, která jsou vypsána zde.

Omezení vazby MQ:

Použití vazby MQ má jistá omezení, která jsou vypsána zde.

Žádná distribuce zpráv publikování-odběr

Metoda publikování-odběr pro distribuci zpráv není v současnosti vazbou MQ podporována, ačkoli samotné WMQ publikování-odběr podporuje. Vazba MQ JMS však tuto metodu distribuce podporuje.

Sdílené fronty příjmu

Různé vazby exportu a importu WebSphere MQ očekávají, že jsou veškeré zprávy v jejich nakonfigurované frontě příjmu určeny pro daný export nebo import. Vazby importu a exportu by měly být nakonfigurovány se zohledněním těchto aspektů:

- Každý import MQ musí mít jinou frontu příjmu, protože import MQ předpokládá, že jsou všechny zprávy ve frontě příjmu odpovědi na požadavky, které odeslal. Pokud je fronta příjmu sdílená více importy, mohl by odpovědi přijímat nesprávný import a nebyly by korelovány s původními zprávami požadavků.
- Každý export MQ by měl mít jinou frontu příjmu, protože jinak nelze předpovědět, který export dostane určitou zprávu požadavku.
- Importy a exporty MQ mohou ukazovat na stejnou frontu odesílání.

Omezení vazeb JMS, MQ JMS a generické vazba služby JMS:

Vazby JMS a MQ JMS mají určitá omezení.

Implikace generování výchozích vazeb

O omezeních použití vazeb JMS, MQ JMS a generické vazby JMS pojednávají tyto části:

- Implikace generování výchozích vazeb.
- Schéma korelace odpovědi.
- Podpora obousměrného zápisu.

Pokud se při generování vazby nerozhodnete zadat hodnoty sami, bude několik polí vygenerováno za vás. Bude za vás vytvořen například název továrny připojení. Pokud víte, že svou aplikaci umístíte na server a budete k ní přistupovat vzdáleně prostřednictvím klienta, měli byste při vytvoření vazby místo přijetí výchozích hodnot zadat názvy rozhraní JNDI, protože tyto hodnoty budete pravděpodobně chtít kontrolovat pomocí administrativní konzoly za běhu.

Pokud jste však přijali výchozí nastavení a později zjistili, že k vaší aplikaci nelze přistupovat ze vzdáleného klienta, můžete hodnotu továrny připojení nastavit explicitně v administrativní konzole. V nastavení továrny připojení vyhledejte pole Koncové body poskytovatele a přidejte hodnotu, jako např. <název_hostitele_serveru>:7276 (pokud používáte výchozí číslo portu).

Schéma korelace odpovědi

Pokud používáte schéma korelace odpovědi ID korelace - ID korelace používané ke korelaci zpráv v operaci požadavek-odezva, musíte ve zprávě používat dynamické ID korelace.

Chcete-li vytvořit dynamické ID korelace v mediačním modulu pomocí editoru mediačních toků, přidejte před importem s vazbou JMS mediační primitivum Mapování. Otevřete editor mapování. V cílové zprávě budou dostupná známá záhlaví architektury SCA (Service Component Architecture). Přetáhněte pole obsahující jedinečné ID ve zdrojové zprávě na ID korelace v záhlaví JMS v cílové zprávě.

Podpora obousměrného zápisu

V běhovém prostředí se v názvech rozhraní JNDI (Java Naming and Directory Interface) podporují pouze znaky ASCII.

Sdílené fronty příjmu

Různé vazby exportu a importu očekávají, že jsou veškeré zprávy v jejich nakonfigurované frontě příjmu určeny pro daný export nebo import. Vazby importu a exportu by měly být nakonfigurovány se zohledněním těchto aspektů:

- Každá vazba importu musí mít jinou frontu příjmu, protože vazba importu předpokládá, že jsou všechny zprávy ve frontě příjmu odpovědi na požadavky, které odeslala. Pokud je fronta příjmu sdílená více importy, mohl by odpovědi přijímat nesprávný import a nebyly by korelovány s původními zprávami požadavků.
- Každý export by měl mít jinou frontu příjmu, protože jinak nelze předpovědět, který export dostane kterou zprávu požadavku.
- Importy a exporty mohou ukazovat na stejnou frontu odesílání.

Obchodní objekty

Průmyslové odvětví vývoje počítačového softwaru vyvinulo několik programovacích modelů a rámců, v rámci kterých poskytují *obchodní objekty* přirozenou reprezentaci obchodních dat pro účely zpracování aplikacemi.

Obecně vzato, tyto obchodní objekty:

- Jsou definované pomocí odvětvových standardů.
- Transparentně mapují data na databázové tabulky či podnikové informační systémy.
- Podporují protokoly vzdáleného vyvolání.
- Poskytují základnu modelu datového programování pro účely programování aplikací.

Z perspektivy nástrojů poskytuje produkt Integration Designer vývojářům jeden takový společný model obchodních objektů pro účely reprezentace různých druhů obchodních entit z různých domén. Při vývoji umožňuje tento model vývojářům definovat obchodní objekty jako definice schématu XML.

Za běhu jsou obchodní data definovaná definicemi schématu XML reprezentována jako Obchodní objekty Java. V rámci tohoto modelu jsou obchodní objekty volně založené na raných konceptech specifikace SDO (Service Data Object) a poskytují kompletní sadu aplikačních rozhraní programovacích modelů potřebných k manipulaci s obchodními daty.

Definování obchodních objektů

Obchodní objekty definujete pomocí editoru obchodních objektů z produktu Integration Designer. Editor obchodních objektů ukládá obchodní objekty jako definice schémat XML.

Použití schématu XML k definování obchodních objektů poskytuje několik výhod:

- Schémata XML poskytují standardizovaný model definice dat a základu pro součinnost různorodých systémů a aplikací. Schémata XML se používají ve spojení s jazykem WSDL (Web Services Description Language), a poskytují tak standardizované smlouvy rozhraní mezi komponentami, aplikacemi a systémy.
- Schémata XML definují bohatý model definice dat pro účely představování obchodních dat. Tento model obsahuje mimo jiné funkce pro komplexní typy, jednoduché typy, typy definované uživatelem, dědičnost typů či kardinalitu.
- Obchodní objekty mohou být definovány obchodními rozhraními a daty definovanými v jazyce WSDL (Web Services Description Language), stejně jako schématem XML od organizací odvětvového standardu nebo od jiných organizací či systémů. Produkt Integration Designer dokáže tyto obchodní objekty importovat přímo.

Produkt Integration Designer také poskytuje podporu vyhledávání obchodních dat v databázích a podnikových informačních systémech a následné generování standardizované definice obchodního objektu schématu XML na základě těchto obchodních dat. Tímto způsobem generované obchodní objekty jsou často označovány jako *obchodní objekty specifické pro aplikace*, protože napodobují strukturu obchodních dat definovaných v podnikovém informačním systému.

Když proces manipuluje daty z mnoha různých informačních systémů, může být hodnotným krokem transformovat různorodé reprezentace obchodních dat (např. ZákazníkEIS1 A ZákazníkEIS2 nebo PořadíEIS1 a Pořadí EIS2) do jediné kanonické reprezentace (např. Zákazník a Pořadí). Kanonická reprezentace je často označována jako *generický obchodní objekt*.

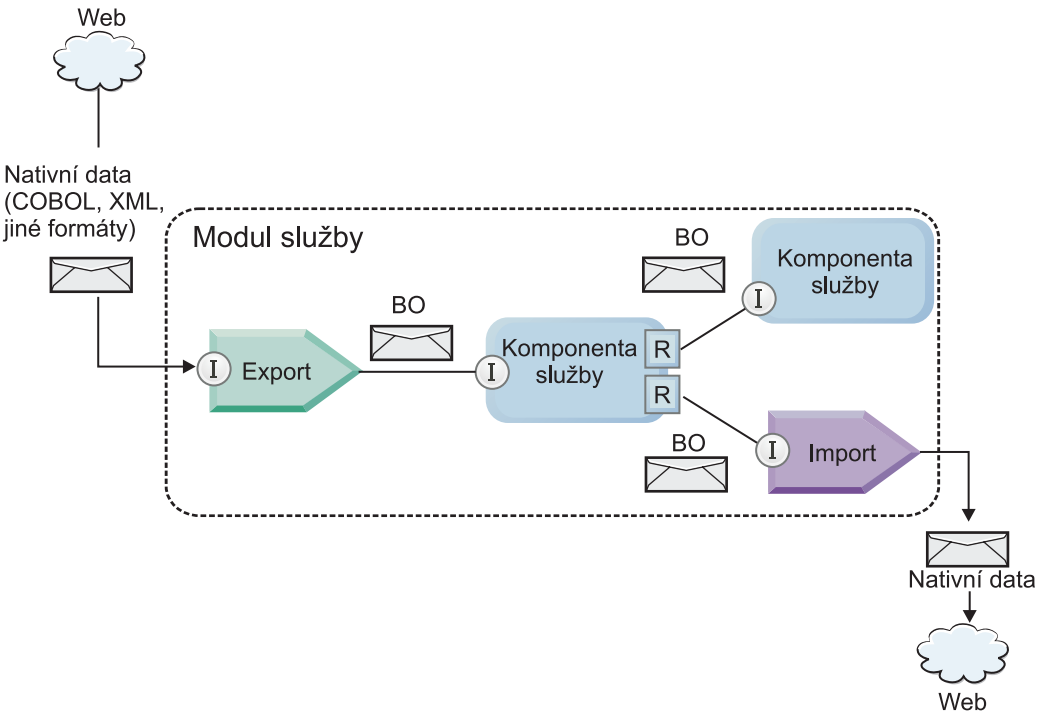
Definice obchodních objektů, obzvláště potom generických obchodních objektů, často používá více než jedna aplikace. V zájmu podpory tohoto opětovného používání umožňuje produkt Integration Designer vytvoření obchodních objektů v knihovnách, které lze přidružit k více modulům aplikací.

Jazyk WSDL (Web Services Description Language) definuje smlouvy ke službám poskytovaným a přijímaným modulem aplikace SCA (Service Component Architecture), stejně jako smlouvy používané k vytvoření komponent v rámci modulu aplikace. V rámci smlouvy může jazyk WSDL reprezentovat jak operace, tak i obchodní objekty (definované schématem XML za účelem reprezentace obchodních dat).

Práce s obchodními objekty

Architektura SCA (Service Component Architecture) poskytuje rámec pro definování modulu aplikace, služeb, které tento modul poskytuje, služeb, které přijímá, stejně jako kompozici komponent poskytujících jeho obchodní logiku. Obchodní objekty hrají v případě aplikací důležitou roli. Definují obchodní data používaná k popisu smluv dané služby a komponenty, a také obchodní data, se kterými tyto komponenty manipulují.

Následující diagram znázorňuje modul aplikace SCA a vykresluje řadu míst, na kterých vývojář pracuje s obchodními objekty.



Obrázek 86. Obchodní objekty reprezentují data tekoucí mezi službami v rámci určité aplikace

Poznámka: Toto téma popisuje, jak moduly aplikace SCA používají obchodní objekty. Pokud používáte rozhraní Java, mohou moduly aplikace SCA zpracovat také objekty jazyka Java.

Programovací model obchodních objektů

Programovací model obchodního objektu sestává ze sady rozhraní Java, která představují:

- Definici obchodního objektu a data instance.
- Sadu služeb podporujících operace s obchodními objekty.

Definice typů obchodních objektů reprezentují rozhraní commonj.sdo.Type a commonj.sdo.Property. Programovací model obchodních objektů poskytuje sadu pravidel mapování informací o komplexním typu schématu XML na rozhraní typu a mapování každého prvku z definice komplexních typů na rozhraní vlastností.

Instance obchodních objektů jsou reprezentována rozhraním commonj.sdo.DataObject. Programovací model obchodních objektů nepoužívá typy, což znamená, že lze stejné rozhraní commonj.sdo.DataObject použít k reprezentaci různých definic obchodních objektů, jako např. Zákazník či Pořadí. Definici, jejíž vlastnosti lze nastavit a získat z každého obchodního objektu, určují informace o typu definované ve schématu XML přidruženému ke každému obchodnímu objektu.

Chování programovacího modelu obchodního objektu je založeno na specifikaci Service Data Object 2.1. Další informace viz objekt SDO 2.1 for Java, výukové programy a dokumenty javadoc na webu: http://osoa.org/display/Main/Service+Data+Objects+Specifications.

Služby obchodních objektů podporují různé operace životního cyklu (jako např. vytvoření, rovnost, syntaktickou analýzu a serializaci) s obchodními objekty.

Specifika modelu programování obchodních objektů viz téma Programování pomocí služeb obchodních objektů a Generovaná dokumentace rozhraní API a SPI týkající se obchodních objektů.

Vazby, vázání dat a popisovače dat

Jak dokládá názorná ukázka, viz Obrázek 41 na stránce 140, jsou obchodní data používaná k vyvolání služeb poskytovaných moduly aplikace SCA transformována v obchodní objekty, aby tak mohly komponenty SCA obchodními daty manipulovat. Podobně jsou obchodní objekty, kterými manipulují komponenty SCA, převedeny do datového formátu, který vyžadují externí služby.

V některých případech, jako např. u vazeb webové služby, vazba používaná k exportu a importu služeb automaticky transformuje data do příslušného formátu. V jiných případech, např. u vazeb služeb JMS, mohou vývojáři poskytnout vázání dat nebo popisovač dat, který převede neintegrované formáty v obchodní objekty reprezentované rozhraním objektu DataObject.

Další informace o vývoji vázání dat a popisovačů dat viz “Popisovače dat” na stránce 55 a “Vázání dat” na stránce 56.

Komponenty

Komponenty SCA definují své smlouvy služeb spotřeby a zajišťování kombinací jazyka WSDL (Web Services Description Language) a schématu XML. Obchodní data, která architektura SCA předává mezi komponentami, jsou prostřednictvím rozhraní objektu DataObject reprezentovány jako obchodní objekty. Architektura SCA ověřuje, zda jsou tyto typy obchodních objektů kompatibilní se smlouvou rozhraní, kterou definovala vyvolávaná komponenta.

Abstrakce programovacího modelu za účelem manipulace obchodními objekty se u jednotlivých komponent liší. Komponenta POJO a vlastní primitivum komponenty mediačního toku poskytují přímou manipulaci obchodními objekty tím, že prostřednictvím programovacích rozhraní a služeb obchodních objektů umožňují přímé programování v jazyce Java. Většina komponent poskytuje abstrakce vyšší úrovně pro účely manipulace obchodními objekty, ale také úseky kódu jazyka Java pro účely definování vlastního chování v rámci rozhraní a služeb obchodních objektů.

Obchodní objekty lze transformovat buď pomocí kombinace komponent Mediace toku rozhraní a Mapa obchodních objektů, nebo pomocí kombinace komponenty mediačního toku a primitiva Mapa XML. Tyto schopnosti transformace obchodních objektů jsou užitečné při převodu obchodních objektů specifických pro aplikace do generických obchodních objektů a z nich.

Speciální obchodní objekty

Objekty SMO a obchodní grafy jsou dva specializované typy obchodních objektů používané pro specifické účely aplikací.

Objekt SMO

Objekt SMO (Service Message Object) je specializovaný obchodní objekt, který používají komponenty mediačního toku k reprezentaci kolekce dat přidružených k vyvolání služby.

Objekt SMO má pevnou strukturu nejvyšší úrovně sestávající ze záhlaví, kontextu, těla a příloh (pokud existují).

- Záhlaví nesou informace týkající se vyvolání služby prostřednictvím určitého protokolu nebo vazby. Příkladem jsou záhlaví zpráv SOAP a platformy JMS.
- Při zpracování prostřednictvím komponenty mediačního toku nesou kontextová data další logické informace přidružené k danému vyvolání. Tyto informace zpravidla nejsou součástí dat aplikace, která odesílají či přijímají klienti.
- Tělo objektu SMO nese obchodní data informačního obsahu zastupující zprávu základní aplikace nebo data vyvolání ve formě standardního obchodního objektu.

Objekt SMO může také nést data příloh pro účely vyvolání webových služeb pomocí zprávy SOAP s přílohami.

Mediační tok provádí úlohy, jako je směrování požadavku či transformace dat, a objekt SMO poskytuje kombinovaný pohled na obsahy záhlaví a informační obsahy v rámci jediné unifikované struktury.

Obchodní graf

Obchodní graf je speciální obchodní objekt používaný za účelem poskytování podpory synchronizace dat v rámci scénářů integrace.

Zvažte příklad, kdy dva podnikové informační systémy obsahují reprezentaci specifické objednávky. Pokud se v jednom ze systémů daná objednávka změní, lze do druhého systému odeslat zprávu za účelem synchronizace dat této objednávky. Obchodní grafy podporují takové pojetí, kdy je do druhého systému odeslána pouze změněná část objednávky společně s anotací v podobě souhrnných informací o změně, které definují její typ.

V tomto příkladu by obchodní graf Objednávka druhému systému sdělil, že došlo k odstranění jedné z položek a že byla aktualizována vlastnost předpokládaný datum expedice.

Obchodní grafy lze snadno přidávat do existujících obchodních objektů v produktu Integration Designer. Nejčastěji se s nimi setkáte ve scénářích, ve kterých jsou používány adaptéry WebSphere a také v případě podpory migrace aplikací serveru WebSphere InterChange Server.

Režim syntaktické analýzy obchodního objektu

Produkt Integration Designer poskytuje vlastnost modulů a knihoven, kterou lze použít k nakonfigurování režimu syntaktické analýzy XML pro obchodní objekty na hodnotu Urychleně nebo Pomalu.

- Pokud je tato volba nastavena na hodnotu *Urychleně*, dojde k vytvoření obchodního objektu prostřednictvím urychlené analýzy proudů bajtů XML.
- Pokud je tato volba nastavena na hodnotu *Pomalu*, je obchodní objekt vytvořen normálně, ale skutečná analýza proudu bajtů XML je odložena a dojde pouze k částečné analýze při přístupu k vlastnostem daného obchodního objektu.

V obou režimech syntaktické analýzy XML jsou za účelem vytvoření obchodního objektu data jiná než XML vždy urychleně analyzována.

Aspekty volby režimu syntaktické analýzy obchodního objektu

Režim syntaktické analýzy obchodního objektu určuje, jakým způsobem je prováděna syntaktická analýza XML dat v době běhu. Režim syntaktické analýzy obchodního objektu je definován v modulu nebo knihovně při jejich vytvoření. Režim syntaktické analýzy modulu nebo knihovny můžete změnit, avšak měli byste si uvědomit možné důsledky.

Režim syntaktické analýzy obchodního objektu je nastaven na úrovni modulu a knihovny. Moduly, které byly vytvořeny v produktu IBM Integration Designer nižší verze než verze 7, poběží v režimu časné syntaktické analýzy bez nutnosti jakýchkoli změn. Standardně bude modulům a knihovnám vytvořeným v produktu IBM Integration Designer verze 7 a novějších poskytnut nejvhodnější režim syntaktické analýzy v závislosti na počtu faktorů, například režim syntaktické analýzy stávajících projektů ve vašem pracovním prostoru nebo režim syntaktické analýzy závislých projektů nebo jiných projektů ve stejném řešení atd. Režim syntaktické analýzy obchodního objektu pro modul nebo knihovnu můžete změnit tak, aby vyhovoval vaší implementaci, avšak měli byste vzít do úvahy následující aspekty.

Aspekty

- Režim zpožděné syntaktické analýzy obchodního objektu rychleji zpracovává XML data; avšak mezi režimem časné a zpožděné syntaktické analýzy existují rozdíly v kompatibilitě, které musíte vzít do úvahy před změnou konfigurace modulu nebo knihovny. Tyto rozdíly ovlivní chování modulů v době běhu. Informace o tom, který režim syntaktické analýzy je optimální pro vaši aplikaci, naleznete v tématu "Výhody použití režimu zpožděné syntaktické analýzy oproti použití režimu časné syntaktické analýzy".
- Modul lze konfigurovat pro provoz pouze v jednom z režimů syntaktické analýzy. Knihovny mohou být konfigurovány tak, aby podporovaly jen některý z režimů syntaktické analýzy nebo oba režimy. Knihovna nakonfigurovaná pro podporu obou režimů syntaktické analýzy může být odkazována modulem využívajícím režim časné syntaktické analýzy i modulem, který využívá režim zpožděné syntaktické analýzy. Režim syntaktické analýzy pro danou knihovnu v době běhu je určen moduly, které na tuto knihovnu odkazují. V době běhu modul deklaruje svůj režim syntaktické analýzy a tento režim je využíván modulem a všemi knihovnami, které modul používá.

- Moduly a knihovny, které jsou nakonfigurovány pro různé režimy syntaktické analýzy, jsou kompatibilní v následujících případech:
  - Moduly a knihovny nakonfigurované s režimem zpožděné syntaktické analýzy jsou kompatibilní s knihovnami, které využívají buďto režim zpožděné syntaktické analýzy, nebo režim zpožděné i režim časné syntaktické analýzy.
  - Moduly a knihovny nakonfigurované s režimem časné syntaktické analýzy jsou kompatibilní s knihovnami, které využívají buďto režim časné syntaktické analýzy, nebo režim zpožděné i režim časné syntaktické analýzy.
  - Knihovny nakonfigurované s režimy časné i zpožděné syntaktické analýzy jsou kompatibilní pouze s knihovnami, které používají režim časné i režim zpožděné syntaktické analýzy.
- Používejte stejný režim syntaktické analýzy pro moduly ve vzájemné interakci, které komunikují pomocí vazby SCA. Jestliže moduly komunikují pomocí různých režimů syntaktické analýzy, mohou se vyskytnout problémy s výkonností.

Související pojmy:

"Výhody používání režimu pomalé versus urychlené syntaktické analýzy" na stránce 143
Pro některé aplikace je výhodný režim pomalé syntaktické analýzy XML, zatímco jiné dosáhnou lepšího výkonu v režimu časné syntaktické analýzy. Doporučujeme vaši aplikaci otestovat typovou úlohou v obou režimech syntaktické analýzy, abyste mohli určit, který z daných režimů nejlépe vyhovuje specifické charakteristice vaší aplikace.

Výhody používání režimu pomalé versus urychlené syntaktické analýzy

Pro některé aplikace je výhodný režim pomalé syntaktické analýzy XML, zatímco jiné dosáhnou lepšího výkonu v režimu časné syntaktické analýzy. Doporučujeme vaši aplikaci otestovat typovou úlohou v obou režimech syntaktické analýzy, abyste mohli určit, který z daných režimů nejlépe vyhovuje specifické charakteristice vaší aplikace.

Aplikace provádějící analýzu rozsáhlých datových proudů XML budou pravděpodobně moci těžit ze zlepšení výkonu při použití režimu pomalé syntaktické analýzy XML. Výhody vyššího výkonu se zvyšují úměrně se zvyšováním velikosti proudu bajtů XML a snižováním množství dat z daného proudu dat, ke kterým aplikace přistupuje.

Poznámka: Režim pomalé syntaktické analýzy obchodních objektů podporuje server WebSphere Process Server verze 7.0.0.3 a novější. Také jej podporuje server IBM Process Server. Moduly a mediační moduly obsahující komponenty mediačního toku podporovány nejsou.

Následující aplikace pravděpodobně podají lepší výkon v režimu časné syntaktické analýzy:

- Aplikace analyzující datové proudy jiné než XML.
- Aplikace, které používají zprávy vytvořené pomocí služby BOFactory.
- Aplikace analyzující velmi malé zprávy XML.

Související odkazy:

"Aspekty volby režimu syntaktické analýzy obchodního objektu" na stránce 142
Režim syntaktické analýzy obchodního objektu určuje, jakým způsobem je prováděna syntaktická analýza XML dat v době běhu. Režim syntaktické analýzy obchodního objektu je definován v modulu nebo knihovně při jejich vytvoření. Režim syntaktické analýzy modulu nebo knihovny můžete změnit, avšak měli byste si uvědomit možné důsledky.

Aspekty migrace a vývoje aplikací

Pokud konfigurujete aplikaci původně vyvinutou pro využití režimu urychlené syntaktické analýzy tak, aby nyní používala režim pomalé syntaktické analýzy, nebo pokud plánujete přepnout aplikaci mezi režimem pomalé a urychlené analýzy, mějte na paměti rozdíly mezi těmito režimy a aspekty přepínání režimů.

Ošetřování chyb

Pokud je analyzovaný proud bajtů XML chybně vytvořený, dojde k výjimkám analýzy.

- V režimu urychlené syntaktické analýzy XML k těmto výjimkám dojde, jakmile je obchodní objekt analyzován z příchozího proudu XML.
- Pokud je nakonfigurován režim pomalé syntaktické analýzy XML, vyskytují se výjimky analýzy latentně při přístupu k vlastnostem obchodních objektů, kdy je analyzována chybně vytvořená část kódu XML.

Chcete-li se s chybně vytvořeným kódem XML vypořádat, vyberte jednu z následujících voleb:

- Implementujte sběrnici ESB na hranách, a ověřte tak příchozí XML.
- Autorizujte logiku pomalé detekce chyb v bodě, kde dochází k přístupu k vlastnostem obchodního objektu.

Zásobník a zprávy výjimek

Vzhledem k tomu, že má pomalý a urychlený režim syntaktické analýzy XML každý jiné základní implementace, obsahují sice trasování zásobníku vrácené programovacími rozhraními a službami obchodních objektů stejný název třídy výjimek, ale nemusí obsahovat stejnou zprávu výjimky nebo zabalenou sadu tříd výjimek specifických pro implementaci.

Formát serializace XML

Režim pomalé syntaktické analýzy XML poskytuje optimalizaci výkonu, která se při serializaci pokouší zkopírovat nepozměněné XML z příchozího proudu bajtů do odchozího proudu bajtů. Výsledkem je zvýšení výkonu, ale formát serializace odchozího proudu bajtů XML se může lišit, pokud byl celý obchodní objekt aktualizován v režimu pomalé syntaktické analýzy XML, nebo pokud tento objekt běžel v režimu urychlené syntaktické analýzy XML.

Ačkoli formát serializace XML nemůže být přesně syntakticky ekvivalentní, je sémantická hodnota poskytnutá obchodním objektem ekvivalentní bez ohledu na zvolený režim analýzy. XML lze tedy bezpečně předat mezi aplikacemi spuštěnými v různých režimech syntaktické analýzy se sémantickou ekvivalencí.

Validátor instance obchodního objektu

Validátor instancí režimu pomalé syntaktické analýzy obchodního objektu poskytuje vyšší přesnost ověřování obchodních objektů, zejména ověřování fazet hodnot vlastností. Díky těmto vylepšením zachytává validátor instancí režimu pomalé syntaktické analýzy další problémy, které urychlený režim analýzy nezachytí a poskytuje podrobnější chybové zprávy.

Mapy XML verze 602

Mediační tok původně vyvinutý před verzí 6.1 produktu WebSphere Integration Developer může obsahovat primitiva Mapování obsahující mapu či šablonu stylů, kterou nelze přímo v režimu pomalé syntaktické analýzy XML provést. Když je aplikace převáděna migrací za účelem jejího použití v režimu pomalé syntaktické analýzy XML, může soubory mapování přidružené k primitivům Mapování automaticky aktualizovat průvodce migrací, aby mohly být spuštěny v novém režimu. Pokud ovšem primitivum Mapování odkazuje přímo na ručně upravenou šablonu stylů, daná šablona převedena migrací není a nemůže tudíž být spuštěna v režimu pomalé syntaktické analýzy XML.

Soukromá nepublikovaná rozhraní API

Pokud určitá aplikace využívá nepublikovaná, soukromá, pro implementaci specifická programovací rozhraní obchodních objektů, pravděpodobně v případě přepnutí režimu analýzy selže její kompilace. V režimu urychlené syntaktické analýzy jsou tato soukromá rozhraní zpravidla třídy implementace obchodních objektů definovaná rámcem EMF (Eclipse Modeling Framework).

Ve všech případech doporučujeme z aplikace odebrat soukromá rozhraní API.

Rozhraní API rámce EMF objektu SMO

Mediační komponenta produktu IBM Integration Designer poskytuje schopnost manipulovat obsahem zpráv pomocí tříd a rozhraní Java poskytnutých v balíku com.ibm.websphere.sibx.smobo. V případě režimu pomalé syntaktické analýzy XML lze stále použít rozhraní Java z balíku com.ibm.websphere.sibx.smobo, ale metody odkazující přímo na třídy a rozhraní rámce EMF (Eclipse Modeling Framework) nebo metody zděděné z rozhraní EMF pravděpodobně selžou.

V režimu pomalé syntaktické analýzy XML nelze objekt ServiceMessageObject a jeho obsah přetypovat na objekty rámce EMF.

Služba BOMode

Služba BOMode se používá k určení, zda je aktuálně prováděný režim syntaktické analýzy XML pomalý nebo urychlený.

Migrace

Všechny aplikace předcházející verzi 7.0.0.0 běží v režimu urychlené syntaktické analýzy XML. Když jsou migrovány za běhu prostřednictvím nástrojů migrace za běhu řízení BPM, jsou nadále spuštěny v režimu urychlené syntaktické analýzy XML.

Pokud chcete umožnit konfiguraci aplikace verze předcházející verzi 7.0.0.0 pro použití režimu pomalé syntaktické analýzy XML, musíte nejprve pomocí produktu Integration Designer migrovat její artefakty. Po migraci potom aplikaci nakonfigurujte pro použití pomalé syntaktické analýzy XML.

V tématu Migrace zdrojových artefaktů najdete informace o migraci artefaktů pomocí produktu Integration Designer a v tématu Konfigurace režimu syntaktické analýzy obchodního objektu pro moduly a knihovny najdete informace o nastavení režimu syntaktické analýzy.

Vztahy

Vztah je přidružení mezi dvěma nebo více datovými entitami, zpravidla obchodními objekty. V produktu IBM Business Process Manager Advanced lze vztahy použít k transformaci dat, která jsou ekvivalentní napříč obchodními objekty, a jiných dat, která jsou však reprezentována odlišně. Lze je také použít ke kreslení přidružení napříč různými objekty v různých aplikacích. Vztahy mohou být sdíleny mezi aplikacemi, řešeními a dokonce i mezi produkty.

Infrastrukturu a operace pro správu vztahů v produktu poskytuje IBM Business Process Manager Advanced služba vztahů. Protože tato služba umožňuje zacházet s obchodními objekty bez ohledu na to, kde se nacházejí, může poskytnout komplexní pohled na všechny aplikace v podniku a sloužit jako stavební kámen řešení řízení BPM. Jelikož vztahy jsou rozšiřitelné a spravovatelné, mohou být použity v komplexních řešeních integrace.

Co jsou vztahy?

Vztah je přidružení mezi obchodními objekty. Jednotlivé obchodní objekty ve vztahu se nazývají *účastníci* vztahu. Každý z účastníků vztahu je odlišen od ostatních účastníků funkcí neboli *rolí*, kterou v daném vztahu plní. Vztah obsahuje seznam rolí.

Jednotlivé role a jejich vzájemné vztahy popisuje *definice* vztahu. Definice rovněž popisuje celkový "tvar" vztahu. Například jedna role může mít pouze jediného účastníka, zatímco jiná role může mít libovolný potřebný počet účastníků. Můžete například definovat vztah *automobil-vlastník*, v němž může jeden vlastník vlastnit více automobilů. Například jedna instance může mít v jednotlivých rolích tyto účastníky:

- Automobil (Ferrari).
- Vlastník (John).

Definice vztahu představuje šablonu pro *instanci* vztahu. Instance je běhová konkretizace vztahu. V uvedeném příkladu vztahu *automobil-vlastník* může instance popisovat kterékoli z následujících přidružení:

- John vlastní automobil Ferrari.
- Sara vlastní automobil Mazda.
- Bob vlastní automobil Ferrari.

Použití vztahů odstraňuje potřebu sestavení vlastní perzistence sledování vztahů v rámci vaší obchodní logiky. U některých scénářů za vás dělá veškerou práci služba vztahů. Viz příklad popsaný v oddílu Vztahy identity.

Scénáře

Toto je typický příklad situace, ve které může řešení integrace používat vztahy. Velká společnost kupuje více společností neboli obchodních jednotek. Každá z těchto obchodních jednotek používá jiný software ke sledování pracovníků a laptopů. Společnost potřebuje způsob, jak sledovat zaměstnance a jejich laptopy. Potřebuje řešení, které jí umožní:

- Zobrazit všechny zaměstnance v různých obchodních jednotkách, jako kdyby se nacházeli v jediné databázi.
- Získat jediný pohled na všechny jejich laptopy.
- Povolit zaměstnancům přihlášení do systému a koupi laptopu.
- Pojmout různé systémy podnikových aplikací v různých obchodních jednotkách.

Za tímto účelem potřebuje společnost zajistit, aby například John Smith a John A. Smith v různých aplikacích byli považováni za téhož zaměstnance. Potřebuje způsob, jak konsolidovat jedinou entitu napříč prostory několika aplikací.

Složitější scénáře vztahů zahrnují sestavení procesů BPEL, které kreslí vztahy napříč různými objekty v několika aplikacích. U složitých scénářů vztahů se obchodní objekty nacházejí v integračním řešení, nikoli tedy v aplikacích. Platformu pro trvalou správu vztahů poskytuje služba vztahů. Před službou vztahů byste měli sestavit vlastní službu perzistence objektů. Dva příklady složitých scénářů vztahů:

- Máte obchodní objekt **car** s číslem VIN v aplikaci SAP a chcete sledovat skutečnost, že tento automobil vlastní někdo jiný. Nicméně vztah vlastnictví je s někým v aplikaci PeopleSoft. V takovém vzorci vztahů máte dvě řešení a potřebujete mezi nimi vybudovat most.
- Velká maloobchodní společnost chce monitorovat vratky zboží výměnou za hotovost nebo kredit. Zapojeny jsou dvě aplikace: systém správy objednávek (OMS) pro nákupy a systém správy vratek (RMS) pro vratky. Obchodní objekty se nacházejí ve více než jedné aplikaci, a proto potřebujete způsob, jak znázornit vztahy, které mezi nimi existují.

obecné vzory využívání

Nejběžnějším vzorcem vztahů jsou *ekvivalence*. Ty jsou založeny na křížovém odkazování či korelaci. Tomuto vzorku odpovídají dva typy vztahů: *bez identity* a *identita*.

- **Vztahy bez identity** navazují přidružení mezi obchodními objekty či jinými daty na bázi jedna ku n nebo n ku n. Pro každou instanci vztahu může existovat jedna či více instancí každého účastníka. Jedním typem vztahu bez identity je statický vyhledávací vztah. Příkladem může být vztah, ve kterém **CA** v aplikaci SAP souvisí s objektem **California** v aplikaci Siebel.
- **Vztahy identity** navazují přidružení mezi obchodními objekty či jinými daty na bázi jedna ku jedné. Pro každou instanci vztahu může existovat pouze jedna instance každého účastníka. Vztahy identity zachycují křížové odkazy mezi obchodními objekty, které jsou sice sémanticky ekvivalentní, ale v různých aplikacích jsou identifikovány různě. Každý účastník vztahu je přidružen k obchodnímu objektu, jenž má určitou hodnotu (nebo kombinaci hodnot), která jej jedinečně identifikuje. Vztahy identity většinou transformují klíčové atributy obchodních objektů, jako jsou číselné identifikátory a kódy produktů.

  Máte-li například v aplikacích SAP, PeopleSoft a Siebel obchodní objekty **car** a chcete-li vytvořit řešení, které je synchronizuje, zpravidla by bylo třeba zavést ručně vytvořenou logiku synchronizace vztahů v šesti mapách:

  SAP -> generický

  generický -> SAP

  PeopleSoft-> generický

  generický-> PeopleSoft

  Siebel-> generický

  generický-> Siebel

  Pokud však ve svém řešení použijete vztahy, poskytne služba vztahů předem sestavené implementace vzorců, které vám zachovají všechny tyto instance vztahů.

Nástroje pro práci se vztahy

Nástrojem pro modelování a návrh vztahů a rolí obchodní integrace je *editor vztahů* v produktu Integration Designer. Podrobné informace o pozadí a úlohách souvisejících s vytvářením vztahů a použitím editoru vztahů najdete v tématu Vytvoření vztahů.

Službou infrastruktury v produktu IBM Business Process Manager, která udržuje vztahy a role v systému a poskytuje operace pro správu vztahů a rolí, je *služba vztahů*.

Rozhraním pro správu vztahů je *Správce vztahů*. Lze k němu přistupovat prostřednictvím stránek Správce vztahů v administrativní konzole.

Vztahy lze programově vyvolávat prostřednictvím rozhraní API služby vztahů.

Služba vztahů

Služba vztahů ukládá data vztahů v tabulkách vztahů, kde napříč aplikacemi a řešeními sleduje hodnoty specifické pro různé aplikace. Služba vztahů poskytuje operace pro správu vztahů a rolí.

Princip činnosti vztahů

Vztahy a role jsou definovány pomocí grafického rozhraní nástroje editoru vztahů v produktu Integration Designer. Služba vztahů ukládá korelační data do tabulek v databázi vztahů ve výchozím zdroji dat, který určíte při konfiguraci služby vztahů. Oddělená tabulka (někdy se nazývá tabulka účastníků) ukládá informace o jednotlivých účastnících vztahu. Na základě těchto tabulek vztahů služba vztahů sleduje související hodnoty specifické pro různé aplikace a šíří aktualizované informace napříč všemi řešeními.

Vztahy, což jsou obchodní artefakty, jsou implementovány v rámci projektu nebo ve sdílené knihovně. Při první implementaci služba vztahů provede naplnění daty.

Pokud mapy či jiné komponenty produktu IBM Business Process Manager požadují za běhu instanci vztahu, jsou v závislosti na scénáři aktualizovány nebo načteny instance vztahu.

Manipulace s daty instance role či vztahu je možná třemi způsoby:

- Vyvoláním rozhraní API služby vztahů úryvkem kódu Java komponenty produktu IBM Business Process Manager.
- Transformací vztahu ve službě mapování obchodních objektů produktu IBM Business Process Manager.
- Nástrojem Správce vztahů.

Podrobné informace o pozadí a úlohách souvisejících s vytvářením vztahů, identifikací typů vztahů a použitím editoru vztahů viz tématu Vytvoření vztahů.

Správce vztahů

Správce vztahů představuje rozhraní pro správu vztahů. Lze k němu přistupovat prostřednictvím stránek Správce vztahů v administrativní konzole.

Správce vztahů poskytuje grafické uživatelské rozhraní pro vytváření a manipulaci se vztahy a daty rolí za běhu. Entity vztahů můžete spravovat na všech úrovních: instance vztahu, instance role a úrovně dat atributu a dat vlastností. Správce vztahů umožňuje:

- Zobrazit seznam vztahů v systému a podrobné informace o jednotlivých vztazích.
- Spravovat instance vztahů:
  - Dotazovat se na data vztahů za účelem zobrazení podmnožin dat instancí.
  - Dotazovat se na data vztahů za účelem zobrazení podmnožin dat instancí pomocí pohledů databáze.
  - Zobrazit seznam instancí vztahů, které odpovídají dotazu na vztah, včetně podrobných informací o instanci.
  - Upravit hodnoty vlastností pro instanci vztahu.

  - Vytvořit a odstranit instance vztahu.
- Spravovat role a instance role:
  - Zobrazit podrobnosti o roli nebo instanci role.
  - Upravit vlastnosti instance role.
  - Vytvořit a odstranit instance role pro určitý vztah.
  - Odvolat data instance vztahu do časového bodu, o němž víte, že v něm byla data spolehlivá.
- Importovat data z existujícího statického vztahu do systému nebo exportovat data z existujícího statického vztahu do souboru RI nebo CSV.
- Odebrat data a schéma vztahů z úložiště po odinstalaci aplikace, která je používá.

Vztahy v prostředích síťové implementace

V prostředích síťové implementace (ND) lze vztahy používat bez jakékoli další konfigurace.

V prostředích síťové implementace (ND) jsou vztahy nainstalovány v klastru aplikací. Poté jsou vztahy viditelné v rámci klastru, přičemž k datům instancí uloženým v databázi vztahů mají přístup všechny servery v tomto klastru. Schopnost spuštění služby vztahů v prostředí ND zajišťuje rozšiřitelnost a vysokou dostupnost.

Správce vztahů umožňuje spravovat vztahy napříč různými klastry prostřednictvím centralizovaného rozhraní pro správu. Správce vztahů připojíte k serveru v klastru pomocí výběru jeho objektu MBean vztahu.

Rozhraní API služby vztahů

Prostřednictvím rozhraní API služby vztahů lze programově vyvolávat vztahy, a to v rámci map obchodních objektů i mimo ně.

K dispozici jsou tři typy rozhraní API:

- Rozhraní API pro manipulaci s instancí vztahu (včetně přímého vytvoření, aktualizace a odstranění dat instance).
- Rozhraní API pro podporu vzorců vztahů (včetně correlate() a correlateforeignKeyLookup).
- Vzorce pro vyhledání vztahu (rozhraní API pro vyhledání).

Sběrnice Enterprise Service Bus v produktu IBM Business Process Manager

Produkt IBM Business Process Manager podporuje integraci služeb aplikací, včetně stejných schopností jako WebSphere Enterprise Service Bus.

Spojování služeb prostřednictvím sběrnice Enterprise Service Bus

Pomocí sběrnice Enterprise Service Bus (ESB) můžete maximalizovat flexibilitu architektury SOA. Účastníci interakce služeb nejsou připojeni k sobě navzájem, ale ke sběrnici ESB.

Když se klient služby připojí ke sběrnici ESB, tato přebírá zodpovědnost za doručování (prostřednictvím zpráv) jeho požadavků na poskytovatele služeb, který nabízí požadovanou funkci a kvalitu služeb. Sběrnice ESB usnadňuje interakce mezi žadateli a poskytovateli a adresuje neshody v protokolech, vzorcích interakcí a schopnostech služeb. Sběrnice ESB může také povolit nebo rozšířit monitorování a správu. Sběrnice ESB poskytuje funkce virtualizace a správy, které implementují a rozšiřují schopnosti jádra architektury SOA.

Sběrnice ESB abstrahuje tyto funkce:

Umístění a identita

Účastníci nemusí znát umístění ani identitu jiných účastníků. Žadatelé například nemusí vědět, že by určitý požadavek mohl obsloužit kterýkoli z několika poskytovatelů. Poskytovatele služeb lze přidávat nebo odebírat bez přerušení.

Protokol interakce
Účastníci nemusejí sdílet stejný komunikační protokol či styl interakce. Například požadavek vyjádřený jako protokol SOAP přes protokol HTTP může obsloužit poskytovatel, který rozumí pouze protokolu SOAP přes službu JMS (Java Message Service).

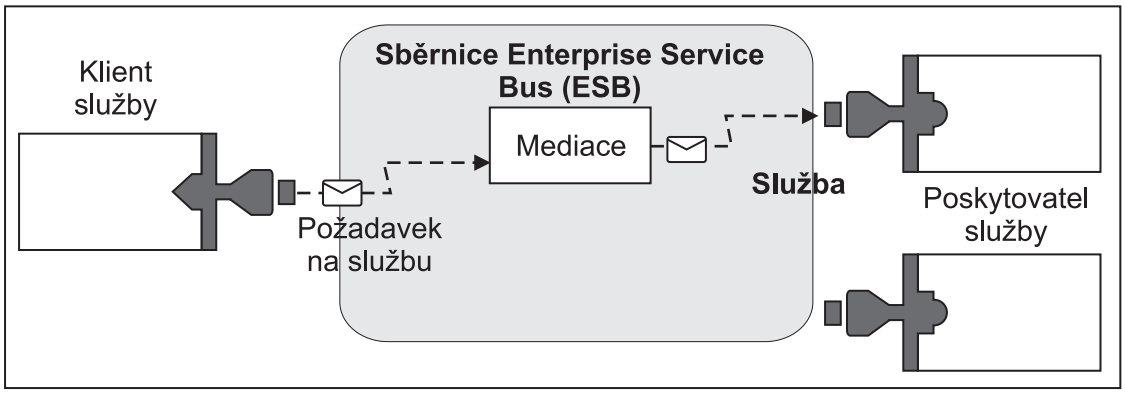
Rozhraní
Žadatelé a poskytovatelé se nemusí shodnout na společném rozhraní. Sběrnice ESB srovnává rozdíly tím, že transformuje zprávy požadavků a odpovědí do formy, kterou poskytovatel očekává.

Kvalita služeb (interakcí)
Účastníci nebo také administrátoři systému deklarují své požadavky na kvalitu služeb, včetně autorizace požadavků, šifrování a dešifrování obsahů zpráv, automatické auditování interakcí služeb, a také způsob, a jakým by jejich požadavky měly být trasovány (např. optimalizace pro rychlost či náklady).

Vložení sběrnice ESB mezi účastníky vám umožní modulovat interakce těchto účastníků pomocí logické konstrukce nazvané *mediace*. Mediace fungují u zpráv mezi žadateli a poskytovateli průběžně. Mediační toky lze například použít k nalezení služeb se specifickými charakteristikami, které hledá žadatel a k interpretaci rozdílů rozhraní mezi žadateli a poskytovateli. Pro účely komplexních interakcí lze vytvořit postupný řetězec mediací.

Pomocí mediací provádí sběrnice Enterprise Service Bus následující akce mezi žadatelem a službou:

- *Trasování* zpráv mezi službami. Sběrnice Enterprise Service Bus nabízí společnou komunikační infrastrukturu, kterou lze použít k propojení služeb, a tím také obchodních funkcí, které tyto služby reprezentují, a to bez potřeby toho, aby programátoři zapisovali a udržovali komplexní logiku konektivity.
- *Převod* přenosových protokolů mezi žadatelem a službou. Sběrnice Enterprise Service Bus poskytuje konzistentní, standardizovaný způsob integrace obchodních funkcí využívajících různé standardy IT. To umožňuje integraci obchodních funkcí, které by normálně komunikovat nemohly. Např. připojení aplikací do zásobníků v rámci různých oddělení, nebo umožnění podílení se na interakcích služeb aplikacím z různých společností.
- *Transformování* formátu zpráv mezi žadatelem a službou. Sběrnice Enterprise Service Bus umožňuje obchodním funkcím vyměňovat informace v různých formátech, přičemž sběrnice zajišťuje, že informace dodané obchodní funkci jsou ve formátu, který daná aplikace požaduje.
- *Obsluha* obchodních událostí z různorodých zdrojů. Sběrnice Enterprise Service Bus podporuje pro účely zpracování požadavků na službu vedle výměny zpráv také interakce založené na událostech.



Obrázek 87. Sběrnice Enterprise Service Bus. Sběrnice Enterprise Service Bus trasuje zprávy mezi aplikacemi, které jsou žadateli nebo poskytovateli služeb. Sběrnice převádí přenosové protokoly a transformuje formát zpráv mezi žadateli a poskytovateli. Na tomto obrázku používá každá aplikace jiný protokol (reprezentovaný jiným geometrickým tvarem konektoru) a jiný formát zpráv.

Při použití sběrnice Enterprise Service Bus se můžete zaměřit na jádro svého podnikání a nemusíte věnovat pozornost svým počítačovým systémům. Služby můžete měnit či přidávat v případě potřeby - například jako odezvu na změny v obchodních požadavcích, chcete-li přidat další kapacitu služeb nebo nové schopnosti. Požadované změny můžete provést předkonfigurováním sběrnice, což bude mít minimální, nebo vůbec žádný, dopad na existující služby a aplikace, které tuto sběrnici používají.

Infrastruktura systému zpráv sběrnice Enterprise Service Bus

Produkt IBM Business Process Manager zahrnuje schopnosti sběrnice Enterprise Service Bus. Produkt IBM Business Process Manager podporuje integraci technologií řízených událostmi a orientovaných na služby či na zprávy za účelem poskytnutí infrastruktury systému zpráv založené na standardech v integrované sběrnici Enterprise Service Bus.

Schopnosti Enterprise Services, které můžete používat v rámci podnikových aplikací, poskytují nejen transportní vrstvu, ale také podporu mediace usnadňující interakce služeb. Sběrnice Enterprise Service Bus je postavena na otevřených standardech a architektuře SOA (service-oriented architecture). Je založena na robustní infrastruktuře Java EE a přidružených službách platformy poskytované serverem IBM WebSphere Application Server Network Deployment.

Produkt IBM Business Process Manager je založen na stejné technologii dostupné se sběrnicí IBM WebSphere Enterprise Service Bus. Tato schopnost je součástí základní funkčnosti produktu IBM Business Process Manager a k využití těchto schopností není vyžadována žádná další licence na WebSphere Enterprise Service Bus.

Nicméně, kolem svého podniku můžete implementovat další samostatné licence sběrnice WebSphere Enterprise Service Bus, a tím rozšířit dosah konektivity u řešení integrace procesů založených na produktu IBM Business Process Manager. Například je možné nainstalovat sběrnici WebSphere Enterprise Service Bus blíže k aplikaci SAP za účelem hostování adaptéru IBM WebSphere Adapter for SAP a transformace zpráv SAP, předtím než budou tyto informace odeslány po síti do obchodního procesu, jehož choreografem je produkt IBM Business Process Manager.

Hostitelé systému zpráv nebo cíle fronty

Hostitel systému zpráv nebo cíle fronty poskytuje funkci zasílání zpráv v rámci serveru. Server se stává cílovým hostitelem systému zpráv, když jej nakonfigurujete jako cíl systému zpráv.

Stroj systému zpráv je spuštěn v rámci serveru. Stroj systému zpráv poskytuje funkce zasílání zpráv a bod připojení, který aplikace používají k připojení ke sběrnici. Asynchronní komunikace architektury SCA (Service Component Architecture), importy a exporty služby JMS i asynchronní interní zpracování používají v rámci stroje systému zpráv fronty zpráv.

Při implementaci modulů aplikace připojuje prostředí implementace zdroj zprávy k cíli zprávy prostřednictvím sběrnice. Když budete znát zdroj a cíl zprávy, můžete snadněji určit potřebný typ prostředí implementace.

Aplikace mohou ukládat trvalá data v datovém úložišti, což je sada tabulek databáze či schématu, nebo v úložišti souborů. Pro účely interakce s touto databází používá stroj systému zpráv instanci zdroje dat JDBC.

Cíl systému zpráv nakonfigurujte při definování prostředí implementace prostřednictvím volby **Server** v administrativní konzole nebo server určete jako cílového hostitele během instalace softwaru.

Datová úložiště:

Každý stroj systému zpráv může používat datové úložiště, což je sada tabulek databáze či schématu, ve kterých jsou uložena trvalá data.

Všechny tabulky datového úložiště jsou zadržovány ve stejném schématu databáze. Každé datové úložiště můžete vytvořit v oddělené databázi. Můžete také vytvořit více datových úložišť v jedné databázi, ale každé z nich musí používat jiné schéma.

Stroj systému zpráv používá instanci zdroje dat JDBC k zajištění interakce s databází obsahující dané datové úložiště pro daný stroj systému zpráv.

Poskytovatelé JDBC

Pomocí poskytovatelů JDBC lze umožnit interakci aplikací s relačními databázemi.

Aplikace využívají poskytovatele JDBC k zajištění interakce s relačními databázemi. Poskytovatel JDBC dodává specifickou implementační třídu ovladačů JDBC pro přístup ke specifickému typu databáze. Přidružením zdroje dat k

poskytovateli JDBC můžete vytvořit fond připojení k této databázi. Poskytovatel JDBC spolu s objekty zdroje dat jsou funkčně shodné s továrnou připojení architektury Java EE Connector Architecture (JCA), která poskytuje konektivitu k nerelační databázi.

Viz příklady nastavení typického samostatného prostředí a nastavení typického prostředí implementace v předchozím tématu.

Podrobnější informace o poskytovatelích JDBC viz téma “Poskytovatelé JDBC” v Informačním centru aplikačního serveru WebSphere Application Server.

Sběrnice SIBus produktu IBM Business Process Manager

Sběrnice SIBus je spravovaný mechanizmus komunikace podporující integraci služeb prostřednictvím synchronního a asynchronního zasílání zpráv. Sběrnice sestává z propojujících se strojů systému zpráv, které spravují prostředky této sběrnice. Jedná se o jednu z technologií serveru WebSphere Application Server, na kterých je založený produkt IBM Business Process Manager.

Některé používané sběrnice vytváří automaticky systém, implementované aplikace architektury SCA (Service Component Architecture), či další komponenty. Sběrnice můžete také vytvořit za účelem podpory logiky integrace služeb nebo jiných aplikací. Například kvůli podpoře aplikací, které fungují jako klienti a poskytovatelé služeb v rámci produktu IBM Business Process Manager, nebo kvůli odkazování na produkt WebSphere MQ.

Cíl sběrnice je logická adresa, ke které se mohou aplikace připojovat jako producenti, odběratelé nebo obojí. Cíl fronty je cíl sběrnice používaný pro účely systému zpráv typu point-to-point.

Každá sběrnice může mít minimálně jeden člen sběrnice, přičemž každý takový člen je buď server, nebo klastr.

Topologie sběrnice je fyzické uspořádání aplikačních serverů, strojů systému zpráv a správců front produktu WebSphere MQ, stejně jako šablona připojení sběrnice a odkazy mezi těmito připojeními, ze kterých sestává vaše sběrnice Enterprise Service Bus.

Některé sběrnice SIBus jsou vytvořeny za účelem podpory produktu IBM Business Process Manager automaticky. Když vytváříte prostředí implementace nebo konfigurujete server či klastr za účelem podpory aplikací SCA, jsou vytvořeny až šest sběrnic. Každá z těchto sběrnic má pět aliasů ověřování, které je třeba nakonfigurovat.

Systémová sběrnice SCA:

Systémová sběrnice architektury SCA je sběrnice SIBus používaná jako hostitel cílů fronty modulů architektury SCA (Service Component Architecture). Běhové prostředí architektury SCA, které podporuje mediační moduly, používá cíle systémové sběrnice jako infrastrukturu podporující asynchronní interakce mezi komponentami a moduly.

Aplikační sběrnice se vytváří automaticky při vytvoření prostředí implementace nebo při konfiguraci serveru či klastru za účelem podpory aplikací SCA. Systémová sběrnice poskytuje rozsah, v rámci kterého jsou konfigurovány prostředky, jako např. cíle fronty, pro účely mediačních modulů a koncových bodů interakcí. Sběrnice umožňuje směrování zpráv mezi koncovými body. Pro danou sběrnici můžete určit kvalitu služby, včetně její priority a spolehlivosti.

Název sběrnice je SCA.SYSTEM.busID.Bus. Alias ověřování používaný k zabezpečení této sběrnice je SCA_Auth_Alias.

Aplikační sběrnice SCA:

Cíle aplikační sběrnice podporují asynchronní komunikaci adaptérů WebSphere Business Integration Adapters a dalších komponent architektury System Component Architecture.

Aplikační sběrnice se vytváří automaticky při vytvoření prostředí implementace nebo při konfiguraci serveru či klastru za účelem podpory aplikací SCA. Aplikační sběrnice se podobá sběrnicím SIBus, které lze vytvořit za účelem podpory logiky integrace služeb nebo jiných aplikací.

Název sběrnice je SCA.APPLICATION.busID.Bus. Alias ověřování používaný k zabezpečení této sběrnice je SCA_Auth_Alias.

Sběrnice infrastruktury Common Event Infrastructure:

Sběrnice infrastruktury Common Event Infrastructure se používá k asynchronnímu přenosu událostí Common Base Event do nakonfigurovaného serveru infrastruktury Common Event Infrastructure.

Název této sběrnice je CommonEventInfrastructure_Bus. Alias ověřování používaný k zabezpečení této sběrnice je CommonEventInfrastructureJMSAuthAlias.

Sběrnice komponenty Business Process Choreographer:

Pro účely interního přenosu zpráv použijte název a ověření sběrnice komponenty Business Process Choreographer.

Sběrnice komponenty Business Process Choreographer se používá k internímu přenosu zpráv a také pro účely rozhraní API platformy JMS (Java Messaging Service) komponenty Business Flow Manager.

Název této sběrnice je BPC.cellMane.Bus. Alias ověřování je BPC_Auth_Alias

Sběrnice komponenty Performance Data Warehouse:

Sběrnice komponenty Performance Data Warehouse se používá k internímu přenosu zpráv infrastrukturou a ke komunikaci s klienty produktu IBM Business Process Manager.

Sběrnice komponenty Performance Data Warehouse se vytváří automaticky při vytvoření prostředí implementace.

Název sběrnice je PERFDW.busID.Bus. Alias ověřování používaný k zabezpečený této sběrnice je PERFDWME_Auth_Alias.

Sběrnice serveru Process Server:

Sběrnice serveru Process Server se používá k internímu přenosu zpráv infrastrukturou a ke komunikaci s klienty produktu IBM Business Process Manager.

Sběrnice serveru Process Server se vytváří automaticky při vytvoření prostředí implementace.

Název sběrnice je PROCSVR.busID.Bus. Alias ověřování používaný k zabezpečení této sběrnice je PROCSVRME_Auth_Alias.

Aplikace služeb a moduly služeb

Modul služby je modul architektury SCA(Service Component Architecture), který poskytuje služby v rámci běhového prostředí. Když implementujete modul služby do produktu IBM Business Process Manager, sestavujete přidruženou aplikaci služeb, která je zabalená v podobě souboru EAR.

Moduly služby jsou základní jednotky implementace, které mohou obsahovat komponenty, knihovny a moduly fázování používané přidruženými aplikacemi služeb. Moduly služby obsahují exporty a volitelně také importy, které definují vztahy mezi moduly a klienty i poskytovateli služeb. Produkt WebSphere Process Server podporuje moduly obchodních služeb a mediačních modulů. Moduly o mediační moduly jsou typy modulů SCA. Mediační modul umožňuje komunikace mezi aplikacemi, neboť převádí vyvolání služby do formátu, kterému cíl rozumí, předá požadavek na cíl a vrátí výsledek původci. Modul pro obchodní službu implementuje logiku obchodního procesu. Modul však může také obsahovat stejnou mediační logiku, která se balí do mediačního modulu.

Implementace aplikace služeb

Proces implementace souboru EAR obsahujícího aplikaci služeb je shodný s procesem implementace libovolného souboru EAR. Hodnoty parametrů mediace lze změnit v době implementace. Po implementaci souboru EAR obsahujícího modul SCA si můžete nechat zobrazit podrobnosti o dané aplikaci služeb a o modulu, který je k ní přidružený. Můžete vidět, jak je modul služby připojený ke klientům služby (prostřednictvím exportů) a poskytovatelům služby (prostřednictvím importů).

Zobrazení podrobností o modulu SCA

To, jaké podrobnosti o modulu služby si můžete nechat zobrazit, závisí na daném modulu SCA. Podrobnosti mohou obsahovat následující atributy.

- Název modulu SCA.
- Popis modulu SCA.
- Název přidružené aplikace.
- Informace o verzi modulu SCA, pokud je tento opatřen verzí.
- Importy modulu SCA:
  - Rozhraní importu jsou abstraktní definice popisující způsob přístupu modulu SCA ke službě.
  - Vazby exportu jsou konkrétní definice určující fyzický mechanizmus, který modul SCA používá pro přístup ke službě. Například prostřednictvím vazby SOAP/HTTP.
- Exporty modulu SCA:
  - Rozhraní exportu jsou abstraktní definice popisující způsob přístupu klienta služby k modulu SCA.
  - Vazby exportu jsou konkrétní definice určující fyzický mechanizmus, který modul klient služby používá pro přístup k modulu SCA a nepřímo také ke službě.
- Vlastnosti modulu SCA.

Importy a vazby importu

Importy definují interakce mezi moduly SCA a poskytovateli služeb. Moduly SCA používají importy za účelem povolení přístupu komponent k externím službám (službám vyskytujícím se mimo modul SCA) za použití lokální reprezentace. Vazby importu definují specifický způsob přístupu k externí službě.

Pokud moduly SCA nepotřebují přistupovat k externím službám, nemusí mít importy. Mediační moduly mají obvykle minimálně jeden import používaný k předávání zpráv či požadavků určeným cílům.

Rozhraní a vazby

Import modulu SCA musí mít minimálně jedno rozhraní a má jedinou vazbu.

- Rozhraní exportu jsou abstraktní definice, které definují sadu operací za použití jazyka WSDL (Web Services Description Language), jazyka XML pro popis webových služeb. Modul SCA může mít řadu rozhraní importu.
- Vazby importu jsou konkrétní definice určující fyzický mechanizmus, který moduly architektury SCA používají pro přístup k externí službě.

Podporované vazby importu

Produkt IBM Business Process Manager podporuje následující vazby importu:

- Vazby SCA spojují moduly SCA s jinými moduly SCA. Vazby SCA bývají také označovány jako výchozí vazby.
- Vazby webové služby povolují komponentám vyvolávat webové služby. Podporované jsou protokoly SOAP1.1/HTTP, SOAP1.2/HTTP a SOAP1.1/JMS.

  Můžete použít vazbu SOAP1.1/HTTP nebo SOAP1.2/HTTP založenou na rozhraní Java API for XML Web Services (JAX-WS), která umožňuje interakce se službami používajícími vazby dokumentu nebo RPC/literal a která používá

k úpravě vyvolání obslužné rutiny JAX-WS. Oddělená vazba SOAP1.1/HTTP je poskytována za účelem povolení interakcí se službami používajícími vazbu RPC/encoded nebo také když existuje požadavek na úpravu vyvolání za použití obslužných rutin JAX-RPC.

- Vazby HTTP povolují přístup k aplikacím za použití protokolu HTTP.
- Vazby importu objektu EJB (Enterprise JavaBeans) umožňují komponentám SCA vyvolávat služby poskytované obchodní logikou prostředí Java EE na serveru Java EE.
- Vazby podnikových informačních systémů (EIS) zajišťují konektivitu mezi komponentami SCA a externím podnikovým informačním systémem. Tuto komunikaci umožňuje použití adaptérů prostředku.
- Vazby platformy JMS (Java Message Service) verze 1.1 povolují s výchozím poskytovatelem systému zpráv serveru WebSphere Application Server. Služba JMS dokáže využívat různé typy transportu, včetně protokolu TCP/IP a HTTP či HTTPS. Automaticky je podporována třída zpráv JMS a jejích pět podtypů (text, bajty, objekt, proud a mapa).
- Generické vazby služby JMS povolují interoperabilitu s poskytovateli služby JMS od jiných dodavatelů, integrovanými do serveru WebSphere Application Server pomocí zařízení ASF (Application Server Facility) služby JMS.
- Vazby služby produktu WebSphere MQ JMS povolují interoperabilitu s poskytovateli služby JMS založenými na produktu WebSphere MQ. Automaticky je podporována třída zpráv JMS a jejích pět podtypů (text, bajty, objekt, proud a mapa). Pokud chcete použít produkt WebSphere MQ jako poskytovatele služby JMS, použijte vazby WebSphere MQ JMS.
- Vazby WebSphere MQ povolují interoperabilitu s produktem WebSphere MQ. Vazby produktu WebSphere MQ lze použít pouze pomocí vzdálených správců front, a to za použití připojení klienta produktu WebSphere. Pomocí lokálních správců front je použít nelze. Vazby produktu WebSphere MQ použijte pouze v případě, že chcete komunikovat s nativními aplikacemi produktu WebSphere MQ.

Dynamické vyvolání služeb

Služby lze vyvolat prostřednictvím jakékoli podporované vazby importu. Služba se normálně nachází v koncovém bodu určeném v importu. Tento koncový bod se nazývá statický koncový bod. Potlačením statického koncového bodu je možné vyvolat jinou vazbu. Dynamické potlačení statických koncových bodů vám umožňuje vyvolat službu v jiném koncovém bodu, a to prostřednictvím libovolné podporované vazby importu. Dynamické vyvolání služeb vám také umožňuje vyvolat službu v případě, kdy podporované vazba importu nemá statický koncový bod.

Import s přidruženou vazbou se používá k určení protokolu a jeho konfigurace pro účely dynamického vyvolání. Import používaný pro účely dynamického vyvolání lze s volající komponentou spojit nebo jej lze dynamicky vybrat za běhu.

Pro účely vyvolání webových služeb a architektury SCA je možné provést dynamické vyvolání také bez importu a protokol a konfigurace budou odečteny u adresy URL koncového bodu. Cílový typ invokace se identifikuje na základě adresy URL koncového bodu. Pokud se použije import, musí být adresa URL kompatibilní s protokolem dané vazby importu.

- Adresa URL architektury SCA označuje vyvolání jiného modulu SCA.
- Adresa URL protokolu HTTP nebo služby JMS standardně označuje vyvolání webové služby. V případě těchto adres URL je možné poskytnout další hodnotu typu vazby, která označuje, že tato adresa URL představuje vyvolání prostřednictvím vazby HTTP nebo JMS.
- Adresa URL protokolu HTTP webové služby standardně používá protokol SOAP 1.1 a lze určit hodnotu typu vazby, která bude označovat použití protokolu SOAP 1.2.

Exporty a vazby exportu

Exporty definují interakce mezi moduly SCA a klienty služby. Moduly SCA nabízejí služby ostatním prostřednictvím exportů. Vazby exportu definují specifický způsob přístupu klientů služby k modulu SCA.

Rozhraní a vazby

Export modulu SCA vyžaduje nejméně jedno rozhraní.

- Rozhraní exportu jsou abstraktní definice, které definují sadu operací za použití jazyka WSDL (Web Services Description Language), jazyka XML pro popis webových služeb. Modul SCA může mít řadu rozhraní exportu.
- Vazby exportu jsou konkrétní definice určující fyzický mechanizmus, který klienti služby používají pro přístup ke službě. Obvykle je pro export modulu SCA určena jedna vazba. Export, který nemá určenou žádnou vazbu, běhové prostředí interpretuje jako export s vazbou SCA.

Podporované vazby exportu

Produkt IBM Business Process Manager podporuje následující vazby exportu:

- Vazby SCA spojují moduly SCA s jinými moduly SCA. Vazby SCA bývají také označovány jako výchozí vazby.
- Vazby webových služeb povolují vyvolání exportů jako webových služeb. Podporované jsou protokoly SOAP1.1/HTTP, SOAP1.2/HTTP a SOAP1.1/JMS.

  Můžete použít vazbu SOAP1.1/HTTP nebo SOAP1.2/HTTP založenou na rozhraní Java API for XML Web Services (JAX-WS), která umožňuje interakce se službami používajícími vazby dokumentu nebo RPC/literal a která používá k úpravě vyvolání obslužné rutiny JAX-WS. Oddělená vazba SOAP1.1/HTTP je poskytována za účelem povolení interakcí se službami používajícími vazbu RPC/encoded nebo také když existuje požadavek na úpravu vyvolání za použití obslužných rutin JAX-RPC.
- Vazby HTTP povolují přístup k exportům za použití protokolu HTTP.
- Vazby exportu EJB (Enterprise JavaBeans) umožňují vystavit komponenty SCA jako objekty EJB. Obchodní logika prostředí Java EE tak může vyvolat komponenty, které jsou jinak pro tyto vazby nedostupné.
- Vazby podnikových informačních systémů (EIS) zajišťují konektivitu mezi komponentami SCA a externím podnikovým informačním systémem. Tuto komunikaci umožňuje použití adaptérů prostředku.
- Vazby platformy JMS (Java Message Service) verze 1.1 povolují s výchozím poskytovatelem systému zpráv serveru WebSphere Application Server. Služba JMS dokáže využívat různé typy transportu, včetně protokolu TCP/IP a HTTP či HTTPS. Automaticky je podporována třída zpráv JMS a jejích pět podtypů (text, bajty, objekt, proud a mapa).
- Generické vazby služby JMS povolují interoperabilitu s poskytovateli služby JMS od jiných dodavatelů, integrovanými do serveru WebSphere Application Server pomocí zařízení ASF (Application Server Facility) služby JMS.
- Vazby služby produktu WebSphere MQ JMS povolují interoperabilitu s poskytovateli služby JMS založenými na produktu WebSphere MQ. Automaticky je podporována třída zpráv JMS a jejích pět podtypů (text, bajty, objekt, proud a mapa). Pokud chcete použít produkt WebSphere MQ jako poskytovatele služby JMS, použijte vazby WebSphere MQ JMS.
- Vazby WebSphere MQ povolují interoperabilitu s produktem WebSphere MQ. Ke správci front MQ nebo vzdálenému počítači se připojujete pomocí vzdáleného (či klientského) připojení. Místní připojení (nebo vazby) je přímé připojení k produktu WebSphere MQ. To lze použít pouze k připojení ke správci front MQ v rámci stejného počítače. Produkt WebSphere MQ povolí oba typy připojení, ale vazby MQ podporují pouze "vzdálené" (nebo "klientské") připojení.

Mediační moduly

Mediační moduly jsou moduly architektury SCA (Service Component Architecture), které mohou měnit formát, obsah nebo cíl požadavků na službu.

Mediační moduly pracují se zprávami, které jsou v přípravě mezi klienty služeb a poskytovateli služeb. Můžete směrovat zprávy k jiným poskytovatelům služeb a rovněž upravit obsah či strukturu zprávy. Mediační moduly mohou poskytovat funkce, jako např. protokolování zpráv a zpracování chyb, které jsou přizpůsobeny vašim požadavkům.

Z administrativní konzoly můžete měnit určité aspekty mediačních modulů bez nutnosti jejich nové implementace.

Komponenty mediačních modulů

Mediační moduly obsahují následující položky:

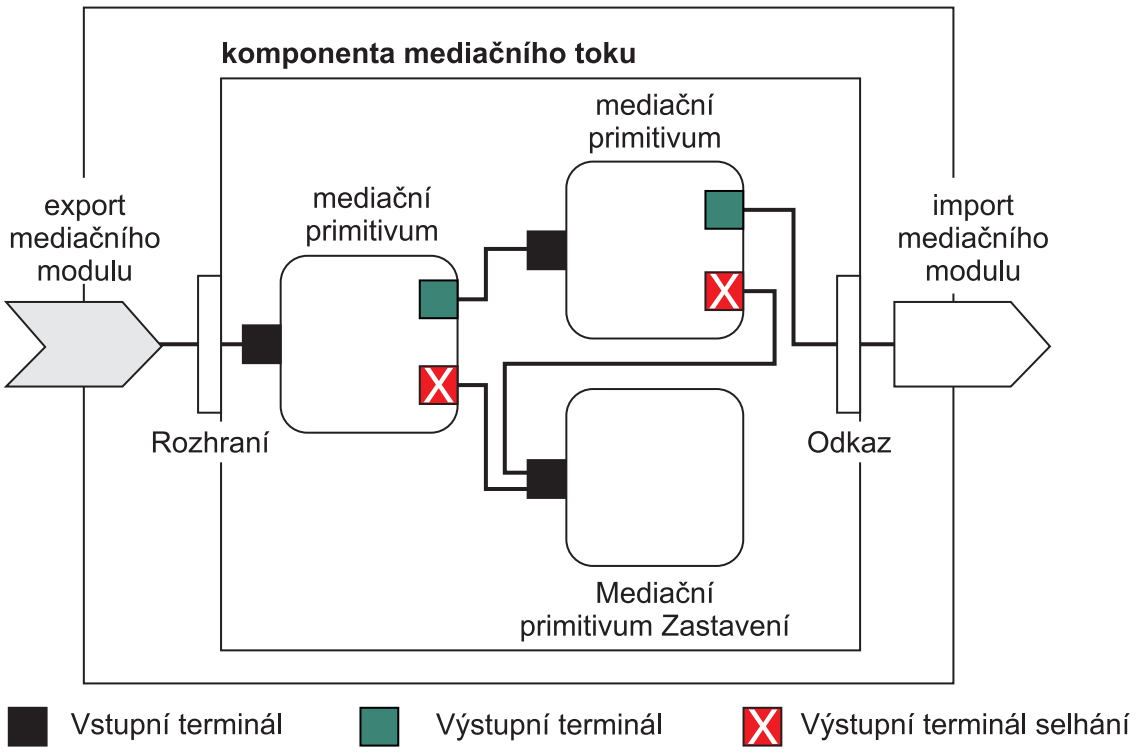
- Importy, které definují interakce mezi moduly SCA a poskytovateli služeb. Importy umožňují modulům SCA volat externí služby, jako kdyby byly lokální. Můžete si nechat zobrazit importy mediačního modulu a upravit jeho vazbu.
- Exporty, které definují interakce mezi moduly SCA a klienty služby. Umožňují modulu SCA nabídnout službu a definovat externí rozhraní (přístupový bod) modulu SCA. Můžete si nechat zobrazit exporty mediačního modulu.
- Komponenty SCA, které jsou stavebními bloky modulů SCA, jako jsou např. mediační moduly. Moduly a komponenty SCA lze pomocí produktu Integration Designer upravovat graficky. Po implementaci mediačního modulu můžete prostřednictvím administrativní konzoly upravovat určité jeho aspekty, a to bez nutnosti jeho nové implementace.

  Mediační moduly obvykle obsahují specifické typy komponent SCA nazvané *komponenta mediačního toku*. Komponenty mediačního toku definují mediační toky.

  Komponenta mediačního toku může obsahovat jedno, žádné nebo několik mediačních primitiv. Produkt IBM Business Process Manager podporuje dodanou sadu mediačních primitiv, která poskytuje funkce trasování a transformace zpráv. Pro další flexibilitu mediačních primitiv můžete pomocí primitiva Vlastní mediace volat vlastní logiku.

  Účelem mediačního modulu, který neobsahuje komponentu mediačního toku, je transformovat požadavky na službu z jednoho protokolu do jiného. Požadavek na službu může být například proveden pomocí SOAP/JMS, ale před odesláním jej může být potřeba transformovat na SOAP/HTTP.

  Poznámka: Mediační moduly si můžete nechat zobrazit a také na nich provádět určité změny prostřednictvím produktu IBM Business Process Manager. V produktu IBM Business Process Manager nicméně nemůžete zobrazovat ani měnit komponenty SCA určitého modulu. Komponenty SCA upravujte pomocí produktu Integration Designer.



Obrázek 88. Zjednodušený příklad mediačního modulu. Mediační modul obsahuje jednu komponentu mediačního toku, která obsahuje mediační primitiva.

- Vlastnosti

  Mediační primitiva mají vlastnosti, z nichž některé lze zobrazit v administrativní konzole jako další vlastnosti modulu SCA.

Aby byly v administrativní konzole produktu IBM Business Process Manager vlastnosti mediačního primitiva viditelné, musí vývojář Integration Developer zvýšit jejich úroveň. Určité vlastnosti se hodí k tomu, aby byly konfigurovány administrativně a produkt Integration Designer takové vlastnosti popisuje jako podporovatelné vlastnosti, protože je lze povýšit z cyklu integrace do cyklu administrace. Jiné vlastnosti zase pro administrativní konfiguraci vhodné nejsou, protože jejich změna může ovlivnit mediační tok takovým způsobem, že by bylo nutné nově implementovat mediační modul. Produkt Integration Designer uvádí seznam vlastností, které můžete zařadit pod význačné vlastnosti mediačního primitiva.

Hodnotu povýšených vlastností můžete pomocí administrativní konzoly produktu IBM Business Process Manager měnit bez nutnosti nové implementace mediačního modulu či restartování serveru či modulu.

Obecně vzato mediační toky použijí změny vlastností okamžitě. Pokud nicméně dojde ke změně vlastnosti v buňce správce implementace, projeví se tyto změny u každého uzlu při jeho synchronizaci. Také mediační toky, které jsou v přípravě, používají dále své předchozí hodnoty.

Poznámka: Prostřednictvím administrativní konzoly můžete měnit pouze hodnoty vlastností, nikoli skupiny vlastností, jejich názvy či typy. Chcete-li změnit skupiny vlastností, jejich názvy nebo typy, musíte použít produkt Integration Designer.

- Mediační modul nebo závislá knihovna mohou také definovat dílčí toky. Dílčí tok zapouzdřuje sadu mediačních primitiv spojených do znovupoužitelné části logiky integrace. Do mediačního toku lze přidat primitivum za účelem vyvolání dílčího toku.

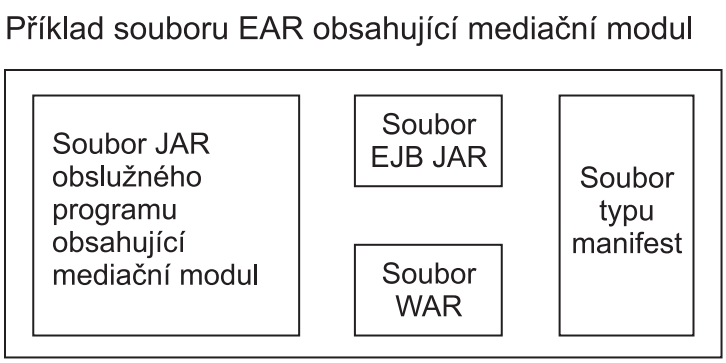
Implementace mediačních modulů

Mediační moduly se vytvářejí prostřednictvím produktu Integration Designer a obecně se implementují do produktu IBM Business Process Manager, do souboru EAR.

V době implementace lze změnit hodnotu povýšených vlastností.

Z produktu Integration Designer můžete mediační modul exportovat, a nechat jej tak zabalit do archivu JAR a ten zase do souboru EAR. Tento soubor EAR můžete potom implementovat tak, že prostřednictvím administrativní konzoly nainstalujete novou aplikaci.

Mediační moduly lze brát jako jednu entitu. Moduly SCA nicméně definuje řada souborů XML uložených v souboru JAR.



Obrázek 89. Zjednodušený příklad souboru EAR obsahujícího mediační modul. Soubor EAR obsahuje soubory JAR. Soubor JAR s obslužným programem obsahuje mediační modul.

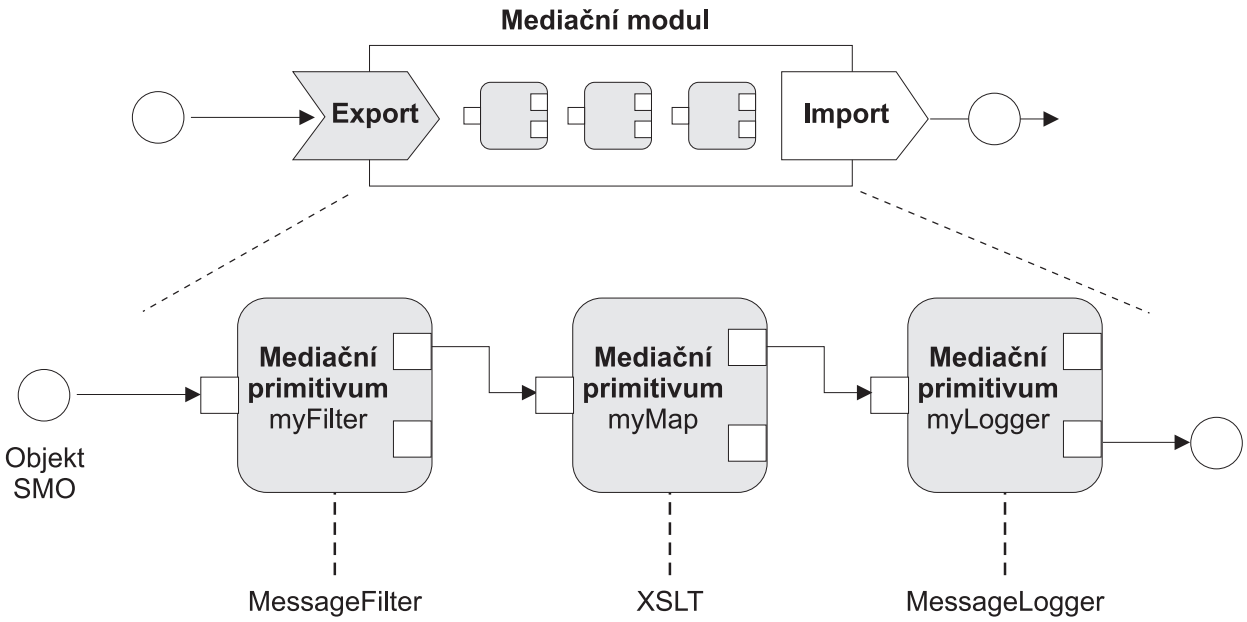
Mediační primitiva

Komponenty mediačního toku operují v rámci toků zpráv mezi komponentami služby. Schopnosti mediačních komponent implementují *mediační primitiva*, která implementují standardní typy implementace služeb.

Komponenta mediačního toku obsahuje minimálně jeden tok. Například jeden pro požadavek a jeden pro odpověď.

Produkt IBM Business Process Manager podporuje dodanou sadu mediačních primitiv, které implementují standardní schopnosti mediace mediačních modulů nebo modulů implementovaných do produktu IBM Business Process Manager. Potřebujete-li speciální schopnosti mediace, můžete vyvinou svá primitiva Vlastní mediace.

Mediační primitivum definuje “vnitřní” operaci, která zpracovává zprávy reprezentované objekty SMO. Mediační primitivum může také definovat “vnější” operace, které odesílají zprávy do jiné komponenty nebo modulu.



Obrázek 90. Mediační modul obsahující tři mediační primitiva

Nakonfigurovat mediační primitiva a nastavit jejich vlastnosti můžete pomocí produktu Integration Designer. Některé z těchto vlastností lze zviditelnit pro administrátora běhového prostředí tím, že zvýšíte jejich úroveň. Jakákoli vlastnost mediačního primitiva, jejíž úroveň lze zvýšit, může být také dynamickou vlastností. Dynamickou vlastnost lze za běhu potlačit pomocí souboru zásad.

Produkt Integration Designer také umožňuje grafické modelování a sestavení komponent mediačního toku z mediačních primitiv a sestavení mediačních modulů nebo modulů z komponent mediačního toku. Administrativní konzola označuje mediační moduly a moduly jako moduly SCA.

Produkt Integration Designer umožňuje také definovat dílčí toky v rámci modulů nebo jejich závislých knihoven. Dílčí tok může obsahovat jakékoli mediační primitivum, s výjimkou mediačního primitiva Vyhodnocení zásady. Dílčí tok se vyvolává z požadavku nebo toku odezvy nebo z jiného dílčího toku za pomoci mediační primitiva Dílčí tok. Vlastnosti povýšené z mediačních primitiv dílčího proudu jsou vystavené jako vlastnosti mediačního primitiva Dílčí tok. Úroveň těchto vlastností lze dále zvyšovat, až dokud nedosáhnou úrovně modulu, což je bod, kdy je již může upravovat administrátor běhového prostředí.

Podporovaná mediační primitiva

Produkt IBM Business Process Manager podporuje následující mediační primitiva:

Mapa obchodních objektů
Transformuje zprávy.

- Definuje transformace zpráv pomocí mapy obchodních objektů, kterou lze používat opakovaně.

- Prostřednictvím editoru map obchodních objektů umožňuje grafické definování transformací zpráv.
- Může změnit obsah zprávy.
- Může transformovat vstupní typ zprávy na jiný výstupní typ zprávy.

Vlastní mediace

Umožňuje vám implementaci vlastní logiky mediace do kódu jazyka Java. Primitivum Vlastní mediace kombinuje flexibilitu mediačního primitiva definovaného uživatelem a jednoduchost předdefinovaného mediačního primitiva. Následujícími způsoby můžete vytvořit komplexní šablony transformací a trasování:

- Vytvořením kódu jazyka Java.
- Vytvořením svých vlastních vlastností.
- Přidáním nových terminálů.

Službu můžete vyvolat prostřednictvím primitiva Vlastní mediace, ale mediační primitivum Vyvolání služby je navrženo tak, aby volalo služby a poskytovalo další funkčnosti, jako např. opakovaný pokus.

Popisovač dat

Umožňuje vám transformovat část zprávy. Používá se k převodu prvku zprávy z fyzického formátu na logickou strukturu nebo k převodu logické struktury do fyzického formátu. Toto primitivum se primárně používá k převodu fyzického formátu, jako např. textového řetězce z objektu Textová zpráva JMS, do logické struktury obchodních objektů a zpátky. Tato mediace se běžně používá na:

- Transformování sekce vstupní zprávy z definované struktury na strukturu jinou. Příkladem může být situace, kdy objekt SMO obsahuje hodnotu řetězce oddělenou čárkami, kterou chcete analyzovat pro potřeby specifického obchodního objektu.
- Pozměnění typu zprávy. Příkladem by mohla být situace, kdy byl export služby JMS konfigurován pro použití základního typu vázání dat služby JMS a v rámci mediačního modulu se vývojář Integration Developer rozhodne, že by měl být obsah předán do specifické struktury obchodních objektů.

Vyhledávání v databázi

Upravuje zprávy za použití informací z databáze dodané uživatelem.

- Musíte nastavit databázi, zdroj dat a všechna nastavení ověření serveru, která má mediační primitivum Vyhledávání v databázi použít. S tímto úkolem vám pomůže administrativní konzola.
- Mediační primitivum Vyhledávání v databázi může číst pouze z jedné tabulky.
- Určený sloupec klíče musí obsahovat jedinečnou hodnotu.
- Data ve sloupci hodnot musí být buď typ jednoduchého schématu XML, nebo typ schématu XML, který rozšiřuje jednoduchý typ schématu XML.

Vyhledávání koncového bodu

Umožňuje dynamické směrování požadavků díky vyhledávání koncových bodů služeb v úložišti.

- Informace o koncovém bodu služby se získává z produktu WSSR (WebSphere Service Registry and Repository). Registr produktu WSRR může být buď místní, nebo vzdálený.
- Změny v registru se provádí prostřednictvím administrativní konzoly produktu WSRR.
- Produkt IBM Business Process Manager musí vědět, jaký registr má používat. Z toho důvodu musíte pomocí administrativní konzoly produktu IBM Business Process Manager vytvořit definice přístupu produktu WSRR.

Emitor událostí

Rozšiřuje monitorování tím, že vám umožní odesílat události z komponenty mediačního toku.

- Zrušením zaškrtnutí příslušného políčka můžete akci mediace pozastavit.
- Události emitoru událostí si můžete nechat zobrazit pomocí prohlížeče událostí CBE (Common Base Event)v produktu IBM Business Process Manager.
- Události byste měli odesílat pouze v případě významné události v rámci mediačního toku nebo z důvodu zajištění výkonu.
- Můžete nadefinovat části zprávy, které daná událost obsahuje.
- Události jsou odesílány ve formě událostí Common Base Event na server Common Event Infrastructure.

- Aby bylo možné informace z emitoru událostí využít plně, musí odběratelé událostí chápat strukturu událostí Common Base Event. Událost Common Base Event má celkové schéma, které ale nemodeluje data specifická pro aplikace, ta jsou obsažena v rozšířených datových prvcích. Pro účely modelování rozšířených datových prvků generují nástroje produktu Integration Designer pro každé nakonfigurované mediační primitivum Emitor událostí soubor definice katalogu událostí infrastruktury Common Event Infrastructure. Definiční soubory katalogu událostí jsou artefakty exportu poskytované jako asistence. Produkt Integration Designer ani běhové prostředí produktu IBM Business Process Manager je nepoužívají. Aplikace spotřebovávající události emitoru událostí byste měli vytvářet s ohledem na soubory definic katalogu událostí.
- V produktu IBM Business Process Manager můžete určit jiné monitorování. Můžete například monitorovat události emitované z importů či exportů.

Selhání

Zastaví konkrétní cestu toku a vygeneruje výjimku.

Vstupní větvení

Pomáhá agregovat (sloučit) zprávy.

- Lze použít pouze ve spojení s mediačním primitivem Výstupní větvení.
- Společné použití mediačních primitiv Vstupní větvení a Výstupní větvení umožňuje agregovat data do jedné výstupní zprávy.
- Mediační primitivum Vstupní větvení přijímá zprávy až do dosažení bodu rozhodování, kdy se výstupem stává jedna zpráva.
- Data by měla být zadržována pomocí sdíleného kontextu.

Výstupní větvení

Pomáhá dělit a agregovat (slučovat) zprávy.

- Společné použití mediačních primitiv Vstupní větvení a Výstupní větvení umožňuje agregovat data do jedné výstupní zprávy.
- V režimu iterace umožňuje mediační primitivum Výstupní větvení iterovat na základě jediné vstupní zprávy obsahující opakující se prvek. Pro každý výskyt tohoto opakujícího se prvku je odeslána zpráva.
- Data by měla být zadržována pomocí sdíleného kontextu.

Modul nastavení záhlaví HTTP

Poskytuje mechanizmus správy záhlaví zpráv HTTP.

- Může vytvořit, nastavit, zkopírovat nebo odstranit záhlaví zpráv HTTP.
- Může nastavit více akcí za účelem změny více záhlaví HTTP.

Mapování

Transformuje zprávy.

- Umožňuje provádět transformace jazyka XSL (Extensible Stylesheet Language) nebo transformace mapy obchodních objektů.
- Zprávy transformujete pomocí transformace XSLT 1.0 nebo XSLT 2.0, případně pomocí transformace mapy obchodních objektů. Transformace XSL operují na serializaci XML zprávy, kdežto transformace mapy obchodních objektů operují na objektech SDO (Service Data Object).

Modul nastavení prvku zprávy

Poskytuje jednoduchý mechanizmus nastavení obsahu zpráv.

- Může změnit, přidat nebo odstranit prvky zprávy.
- Nemění typ zprávy.
- Data ve sloupci hodnot musí být buď typ jednoduchého schématu XML, nebo typ schématu XML, který rozšiřuje jednoduchý typ schématu XML.

Filtr zpráv

Směruje zprávy po různých cestách na základě jejich obsahu.

- Zrušením zaškrtnutí příslušného políčka můžete akci mediace pozastavit.

Modul protokolování zpráv

Protokoluje zprávy v relační databázi nebo prostřednictvím vašeho vlastního modulu protokolování. Tyto zprávy se ukládají jako XML, a jejich data lze tedy následně zpracovat v aplikacích, které s XML pracují.

- Zrušením zaškrtnutí příslušného políčka můžete akci mediace pozastavit.
- Schéma databáze Rational (strukturu tabulek) definuje IBM.
- Standardně používá mediační primitivum Modul protokolování zpráv obecnou databázi. Běhové prostředí mapuje zdroj dat v protokolu **jdbc/mediation/messageLog** k obecné databázi.
- Chcete-li upravit chování vlastního modulu protokolování, můžete nastavit implementační třídy obslužné rutiny. Dále můžete chování vlastního modulu protokolování upravit poskytnutím implementačních tříd formátovače.

Modul nastavení záhlaví MQ

Poskytuje mechanizmus správy záhlaví zpráv MQ.

- Může vytvořit, nastavit, zkopírovat nebo odstranit záhlaví zpráv MQ.
- Může nastavit více akcí za účelem změny více záhlaví MQ.

Vyhodnocení zásady

Umožňuje dynamickou konfiguraci požadavků díky vyhledávání koncových bodů služeb a přidružených souborů zásad v úložišti.

- Soubor zásad můžete použít k dynamickému potlačení povýšených nastavení jiných mediačních primitiv.
- Informace o koncovém bodu služby a informace o zásadách se získávají z produktu WSSR (WebSphere Service Registry and Repository). Registr produktu WSRR může být buď místní, nebo vzdálený.
- Změny v registru se provádí prostřednictvím administrativní konzoly produktu WSRR.
- Produkt IBM Business Process Manager musí vědět, jaký registr má používat. Z toho důvodu musíte pomocí administrativní konzoly produktu IBM Business Process Manager vytvořit definice přístupu produktu WSRR.

Vyvolání služby

Volá službu z mediačního toku, takže není třeba čekat na konec mediačního toku a používat mechanizmus volání.

- Pokud služba vrátí poruchu, můžete opakovat stejnou službu nebo volat službu jinou.
- Mediační primitivum Vyvolání služby je výkonné mediační primitivum, které lze pro jednoduchá volání služby použít samotné, nebo jej lze pro účely komplexních mediací použít v kombinaci s dalšími mediačními primitivy.

Nastavení typu zprávy

Při vývoji integrace umožňuje považovat pole zpráv slabého typu za pole zpráv silného typu. Pole je slabého typu, když může obsahovat více než jeden typ dat. Pole je silného typu, pokud je znám jeho typ a interní struktura.

- Za běhu umožňuje mediační primitivum Nastavení typu zprávy kontrolovat, zda obsah zprávy odpovídá očekávanému typu dat.

Modul nastavení záhlaví SOAP

Poskytuje mechanizmus správy záhlaví zpráv SOAP.

- Může vytvořit, nastavit, zkopírovat nebo odstranit záhlaví zpráv SOAP.
- Může nastavit více akcí za účelem změny více záhlaví SOAP.

Zastavení

Zastaví konkrétní cestu toku bez generování výjimky.

Filtr typů

Umožňuje vám směrovat zprávy do různých cest toku na základě typu zprávy.

Načtení z produktu WebSphere eXtreme Scale

Můžete načíst informace z prostředí mezipaměti serveru eXtreme Scale.

- Můžete vyhledat hodnoty v mezipaměti a uložit je jako prvky do zprávy pomocí klíče.

- Kombinací mediačních primitiv Uložení do produktu eXtreme Scale a Načtení z produktu eXtreme Scale lze uložit do mezipaměti odezvu z back-endového systému. Budoucí požadavky již nebudou vyžadovat přístup k back-endovému systému.
- Je třeba vytvořit definice produktu eXtreme Scale pomocí administrativní konzoly produktu WebSphere ESB, abyste mohli určit, který server eXtreme Scale se má použít.

Uložení do produktu WebSphere eXtreme Scale

Můžete uložit informace do prostředí mezipaměti serveru eXtreme Scale.

- Informace lze uložit do mezipaměti produktu eXtreme Scale pomocí klíče a objektu.
- Můžete využít kombinaci mediačních primitiv Uložení do produktu eXtreme Scale a Načtení z produktu eXtreme Scale; pomocí mediačního primitiva Uložení uložíte data do mezipaměti a pomocí mediačního primitiva Načtení dříve uložená data načtete.
- Je třeba vytvořit definice produktu eXtreme Scale pomocí administrativní konzoly produktu WebSphere ESB, abyste mohli určit, který server eXtreme Scale se má použít.

Dynamické směrování

Zprávy můžete směrovat různými způsoby za použití koncových bodů definovaných v době integrace nebo dynamicky za běhu určených koncových bodů.

Dynamické směrování pokrývá dva případy směrování zpráv:

- Směrování zpráv, kdy je tok dynamický, ale všechny možné koncové body jsou předdefinované v modulu architektury SCA (Service Component Architecture).
- Směrování zpráv, kdy je dynamický nejen tok, ale také výběr koncového bodu. Koncové body služeb se vybírají z externího zdroje za běhu.

Dynamický výběr koncového bodu

Běhové prostředí má schopnost směrovat zprávy požadavků a odezev na adresu koncového bodu identifikovanou prvkem záhlaví zprávy. Tento prvek záhlaví zprávy lze aktualizovat pomocí mediačních primitiv v mediačním toku. Adresu koncového bodu lze aktualizovat pomocí informací z registru, databáze nebo pomocí informací ze samotné zprávy. Směrování zpráv odpovědí přichází v úvahu pouze, když je odezva odesílána prostřednictvím exportu rozhraní JAX-WS webové služby.

Aby běhové prostředí mohlo dynamické směrování na základě požadavku nebo odezvy implementovat, musí být pro modul SCA nastavena vlastnost Použít dynamický koncový bod, pokud je nastavený v záhlaví zprávy. Vývojáři integrace mohou vlastnost Použít dynamický koncový bod, pokud je nastavený v záhlaví zprávy nastavit nebo mohou zvýšit její úroveň (zviditelnit ji za běhu), aby ji mohl nastavit administrátor běhového prostředí. Vlastnosti modulu si lze nechat zobrazit v okně Vlastnosti modulu. Okno se zobrazí po klepnutí na položku **Aplikace > Moduly SCA > Vlastnosti modulu**. Vývojář Integration Developer přidělí každé povýšené vlastnosti název aliasu, což budou názvy zobrazované v administrativní konzole.

Registr

Informace o koncových bodech služby můžete uložit v produktu WSRR (IBM WebSphere Service Registry and Repository) a následně můžete vytvořit moduly SCA, které koncové body z registru WSRR načtou.

Když vyvíjíte moduly SCA, umožňujete mediačnímu toku dotazovat se registru WSRR na koncový bod služby či na sadu koncových bodů služeb prostřednictvím mediačního primitiva Vyhledávání koncového bodu. Pokud modul SCA obdrží sadu koncových bodů, musí preferovaný koncový bod vybrat pomocí dalšího mediačního primitiva.

Řízení zásad mediace požadavků na službu

Zásady mediace můžete použít k řízení mediačních toků mezi klienty služeb a poskytovateli služeb.

Mediační toky lze řídit pomocí zásad mediace uložených v produktu IBM WebSphere Service Registry and Repository (WSRR). Implementace správy zásad služeb v produktu WSRR je založena na rámci zásad WS-Policy (Web Services Policy Framework).

Abyste mohli řídit požadavky na službu pomocí zásad mediace, musíte mít v registru WSRR vhodné moduly SCA (Service Component Architecture) a dokumenty zásad služeb.

Postup připojení zásady mediace k požadavku na službu

Při vyvíjení modulu SCA, který musí využívat zásadu mediace, je třeba do mediačního toku zahrnout mediační primitivum Vyhodnocení zásady. Za běhu toto mediační primitivum Vyhodnocení zásady získá z registru informace o zásadě mediace. Kvůli podpoře řízení zásad mediace u požadavků na služby proto modul SCA musí obsahovat komponentu mediačního toku.

V registru můžete připojit jednu či více zásad mediace k modulu SCA nebo k cílové službě, kterou modul SCA používá. Připojené zásady mediace lze použít (tj. jsou v dosahu) pro všechny zprávy služby zpracovávaných daným modulem SCA. Zásady mediace mohou obsahovat připojení zásady, která definují podmínky. Podmínky zásady mediace umožňují použít v různých kontextech různé zásady mediace. Kromě toho mohou zásady mediace obsahovat klasifikace, které lze použít k určení stavu řízení.

WebSphere Service Registry and Repository

Produkt WSRR (WebSphere Service Registry and Repository) vám umožňuje uložit a spravovat informace o koncových bodech služeb a zásadách mediace, stejně jako k těmto informacím přistupovat. Pomocí produktu WSRR mohou být vaše aplikace služeb dynamičtější a přizpůsobitelnější vůči měnícím se obchodním podmínkám.

Úvod

Mediační toky mohou produkt WSRR používat jako mechanizmus dynamického vyhledávání, a získávat tak informace o koncových bodech služeb a zásadách mediace.

Za účelem získání přístupu k produktu WSRR se prostřednictvím administrativní konzoly vytváří dokumenty definic WSRR. Můžete také použít příkazy administrace produktu WSRR prostřednictvím skriptovacího klienta wsadmin. Mechanizmem používaným pro připojení k instanci registru a získání koncového bodu služby či zásady mediace jsou definice produktu WSRR a vlastnosti jejich připojení.

Koncové body služeb

Úložiště WSRR můžete používat k uchovávání informací o službách, které již používáte, plánujete používat nebo o kterých chcete být informováni. Tyto služby se mohou nacházet ve vašich systémech nebo i v jiných systémech. Určitá aplikace může například použít produkt WSRR k vyhledání nejvhodnější služby, která by splňovala potřeby jejího fungování a výkonu.

Když vyvíjíte modul SCA, který musí přistupovat ke koncovým bodům služeb z produktu WSRR, musíte do daného mediačního toku zahrnout mediační primitivum Vyhledávání koncového bodu. Za běhu získává mediační primitivum Vyhledávání koncového bodu koncové body služeb z registru.

Zásady mediace

Produkt WSRR můžete také použít k uložení informací o zásadách mediace. Zásady mediace vám mohou pomoci s řízením požadavků na služby tím, že potlačí vlastnosti modulu. Pokud produkt WSRR obsahuje zásady mediace, které jsou přiložené k objektu reprezentujícímu buď vás modul SCA, nebo vaši cílovou službu, mohou zásady mediace potlačit vlastnosti modulu. Pokud chcete v rámci různých kontextů aplikovat různé zásady mediace, můžete vytvořit podmínky zásad mediace.

Poznámka: Zásady mediace se zabývají řízením mediačních toků, nikoliv zabezpečením.

Při vyvíjení modulu SCA, který musí využívat zásadu mediace, je třeba do mediačního toku zahrnout mediační primitivum Vyhodnocení zásady. Za běhu toto mediační primitivum Vyhodnocení zásady získá z registru informace o zásadě mediace.

WebSphere eXtreme Scale

Produkt WebSphere eXtreme Scale (eXtreme Scale) umožňuje poskytovat systém mezipaměti, který lze integrovat s aplikací produktu IBM Business Process Manager. Použití produktu eXtreme Scale s produktem IBM Business Process Manager může zlepšit spolehlivost a dobu odezvy služeb a zajistit další funkčnost integrace.

Produkt eXtreme Scale funguje jako elastická, přizpůsobitelná datová mřížka v paměti. Tato datová mřížka dynamicky ukládá do mezipaměti, segmentuje, replikuje a spravuje data aplikací a obchodní logiku v rámci více serverů. Pomocí produktu eXtreme Scale lze rovněž získat kvality služby, jako např. transakční integritu, vysokou dostupnost a předvídatelnou dobu odezvy.

Pro přístup k funkci ukládání do mezipaměti produktu eXtreme Scale lze použít mediační toky obsahující mediační primitiva produktu WebSphere eXtreme Scale. Při vyvíjení modulu SCA, který musí ukládat informace do mezipaměti produktu eXtreme Scale, je třeba do mediačního toku zahrnout mediační primitivum Uložení do produktu WebSphere eXtreme Scale. Chcete-li načítat informace z mezipaměti produktu eXtreme Scale, je třeba zahrnout mediační primitivum Načtení z produktu WebSphere eXtreme Scale. Kombinací těchto dvou mediačních primitiv v mediačním toku lze uložit do mezipaměti odezvu z back-endového systému, odkud ji mohou načítat budoucí požadavky.

Chcete-li konfigurovat přístup k produktu eXtreme Scale, musíte pomocí administrativní konzoly vytvořit definici produktu WebSphere eXtreme Scale. Volitelně můžete použít příkazy administrace produktu WebSphere eXtreme Scale ve skriptovacím klientu wsadmin. Definice produktu eXtreme Scale je mechanizmus, s jehož pomocí se mediační primitiva Načtení z produktu WebSphere eXtreme Scale a Uložení do produktu WebSphere eXtreme Scale připojují k serveru eXtreme Scale.

Klient Message Service

Klienti Message Service jsou k dispozici pro jazyk C/C++ a .NET za účelem povolení připojení aplikací nezaložených na platformě Java ke sběrnici Enterprise Service Bus.

Klienti Message Service Clients pro C/C++ a .NET poskytují rozhraní API nazvané XMS, které má stejnou sadu rozhraní jako rozhraní API služby JMS (Java Message Service). Klient Message Service Client for C/C++ obsahuje dvě implementace XMS, jednu pro použití aplikacemi v jazyce C a druhou pro použití aplikacemi v jazyce C++. Klient Message Service Client pro .NET obsahuje plně spravovanou implementaci XMS, kterou může používat libovolný jazyk vyhovující rámci .NET.

Klienty Message Service Client for .NET lze získat na adrese http://www-01.ibm.com/support/docview.wss?rs=0&q1=IA9H&uid=swg24011756&loc=en_US&cs=utf-8&cc=us&lang=en

Klienty Message Service Clients for C/C++ lze získat http://www-01.ibm.com/support/docview.wss?rs=0&q1=ia94&uid=swg24007092&loc=en_US&cs=utf-8&cc=us&lang=en.

Můžete také nainstalovat a používat podporu klienta prostředí Java EE z produktu WebSphere Application Server Network Deployment, včetně klienta webových služeb, klienta objektu EJB a klienta služby JMS.