Název	**STANDARDY PRO TVORBU APLIKACÍ PRO MICROSOFT BIZTALK SERVER**
Typ	**Standardy a normy**
Verze	**1.00**
Datum vydání	**27. 9. 2007**
ID	**BizTalkDevelopmentStandard.v1.00.doc**
Popis	Standardy pro vývoj, postupy pro nasazení a provozní postupy aplikací pro Microsoft BizTalk Server v prostředí datového centra ČSSZ
Správce	Jiří Bramburek - Microsoft
Vydává	Úsek IKT

Microsoft® Consulting Services

SIEMENS

Změny:

Datum vydání	Verze	Změna proti předchozí verzi	Změnil (jméno)
4.7.2007	0.30	Úvodní draft	Jiří Brambůrek
23.8.2007	0.40	Rozšíření dokumentu	Jiří Binko
25.8.2007	0.41	Doplnění a úpravy dle jednání z 18.7.2007	Jiří Brambůrek
30.7.2007	0.42	Úpravy jmenných konvencí	Stanislav Dvořák
2.8.2007	0.43	Úpravy dle jednání z 1.8.2007	David Mikulík
15.8.2007	0.44	Úpravy dle jednání z 1.8.2007 – doplnění kapitoly 3	Miloš Maryška
15.8.2007	0.45	Doplnění popisu konstrukce portů	Stanislav Dvořák
20.8.2007	0.46	Konsolidace popisu logování s metodikou vývoje; Doporučení uložení konfiguračních parametrů; Úprava doporučení pro postup instalace; Částečná gramatická a formulační korektura	Roman Antoň
20.8.2007	0.47	Konsolidace revizí do verze pro externí doplnění (MS)	Roman Antoň
24.8.2007	0.48	Úpravy dle jednání z 1.8.2007	Jiří Brambůrek
11.9.2007	0.49	Úprava doporučení pro instalace (kapitola 3) podle výstupů jednání 29.8.2007 Doplněny kapitoly s návrhem struktury instalační a provozní dokumentace procesního řešení (kapitoly 5.6 a 5.7)	Roman Antoň Miloš Maryška
17.9.2007	0.50	Doplnění vzorů a příkladu	Jiří Brambůrek
18.9.	0.51	Sloučená verze 0.49 a 0.50	Roman Antoň
19.9.2007	0.52	Doplnění informací pro nasazování stavových databází, doplnění hlaviček tabulek, nahrazení obrázků konfigurace textem	Jiří Brambůrek
26.9.2007	1.00	Závěrečná a schválená verze k vydání	Jiří Brambůrek Milan Shrbený

Obsah

Seznam obrázků

Seznam tabulek

1. ÚVOD

2. STANDARDY PRO VÝVOJ

Vývoj aplikací pro BizTalk Server 2006 v mnohém vychází z obecných principů moderního vývoje a architektury aplikací (OOP, SOA, EAI ...), tudíž všechny principy definované v obecných metodických pokynech a principy popsané v pokynech pro vývoj pro platformu .NET platí i pro vývoj aplikací pro platformu BizTalk Server 2006. Vývoj aplikací pro BizTalk má však svá specifika, která vedou k nutnosti definovat konkrétní požadované postupy či vymezit některé přístupy, které (ač za určitých podmínek plně validní) nejsou vhodné pro typy aplikací a prostředí ČSSZ.

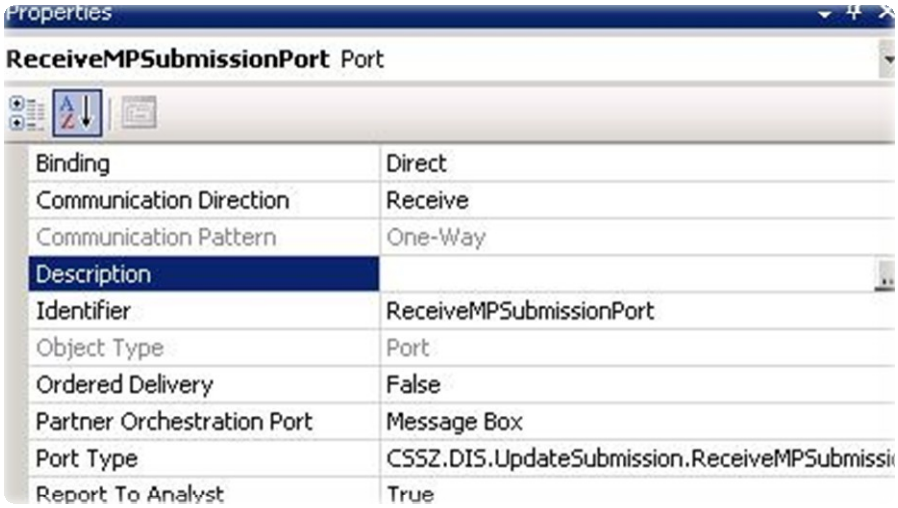
Jakkoliv je možné použít libovolnou kombinaci a množství stavebních bloků (shapes) orchestrací, vývojář by vždy měl mít na paměti, že technologii by měl využívat k implementaci „business procesů". V konečném řešení by neměly převládat prvky ošetřující či překonávající specifické vlastnosti použité technologie, naopak by měl být na první pohled zřejmý ne-technický cíl dané části řešení. Takový postup usnadní další vývoj, správu a provozování řešení postavených na platformě BizTalk Serveru 2006.

2.1 Základní pravidla pro návrh a vývoj

2.1.1 Direct binding

Preferovaným způsobem komunikace mezi jednotlivými orchestracemi je „direct binding", tj. konfigurace messagebox-direct-bound receive a send portů s příslušnými filtry. Tento způsob komunikace umožňuje omezit reference mezi orchestracemi.

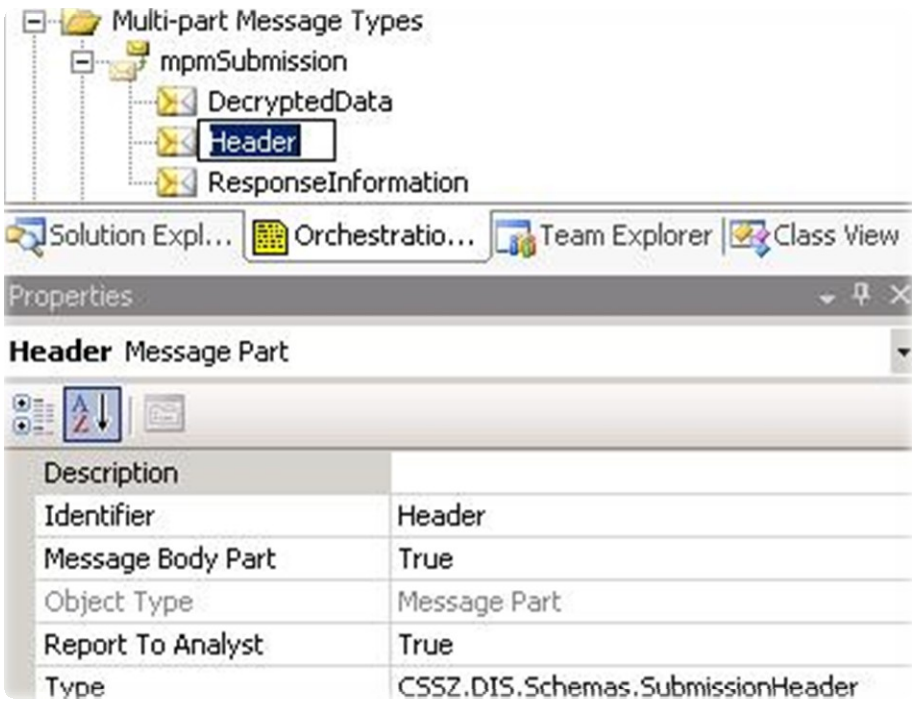
Výjimkou z tohoto pravidla je použití mechanismu callback (kdy mateřská instance orchestrace předá zprávu dceřiné orchestraci a ta poté vrací zprávu původní instanci mateřské orchestrace), v takovém případě je možné použít bloky Call či Start a předat potřebné parametry (jako např. reference na port apod.).



Obrázek 2-1 Konfigurace message box direct bound portu

2.1.2 Multipart messages

Všechny zprávy by měly být navrhovány jako tzv. multipart message types, i tehdy, pokud má zpráva právě jednu část (single part message type). BizTalk pracuje se zprávami typu singlepart stejným způsobem, jako se zprávami typu multipart s jednou částí, tudíž použití mutlipart message types nenese riziko snížení výkonu, na druhou stranu přináší snazší implementaci změn v případě nutnosti úprav schématu.



Obrázek 2-2 Multipart message type

2.1.3 Interní schémata

Pro zprávy (a jejich části) musí být používána pouze interní schémata (tj. schémata specifická pro dané řešení), při komunikaci s externími systémy musí být prováděna transformace na úrovni portu. Interní schémata mohou být vytvořena zkopírováním části nebo celého externího schématu a následnou úpravou namespaces.

Toto opatření je klíčové pro nasazení více aplikací používajících stejnou webovou službu na jeden BizTalk cluster.

Není dovoleno pro zprávy používat schémata či části schémat externích systémů (webových služeb). Taková závislost by vedla k řetězení nutnosti změn aplikací při změnách externích schémat.

2.1.4 Konstrukce portu a napojení na operace ve webových službách

Aby fungovala transformační mapa na portech (převádí interní schéma na externí a naopak.) je nutné, aby porty v orchestraci měly jen jednu operaci. Pokud by tom tak nebylo, nebylo by možné na portu nastavit transformační mapu. Pokud tedy bude orchestrace komunikovat s webovou službou, která má víc operací, musí se v orchestraci vytvořit samostatný port pro každou operaci zvlášť. Propojení těchto portů na operace webové služby se nastaví až při administraci BTS

2.1.5 Orchestrace – princip koheze

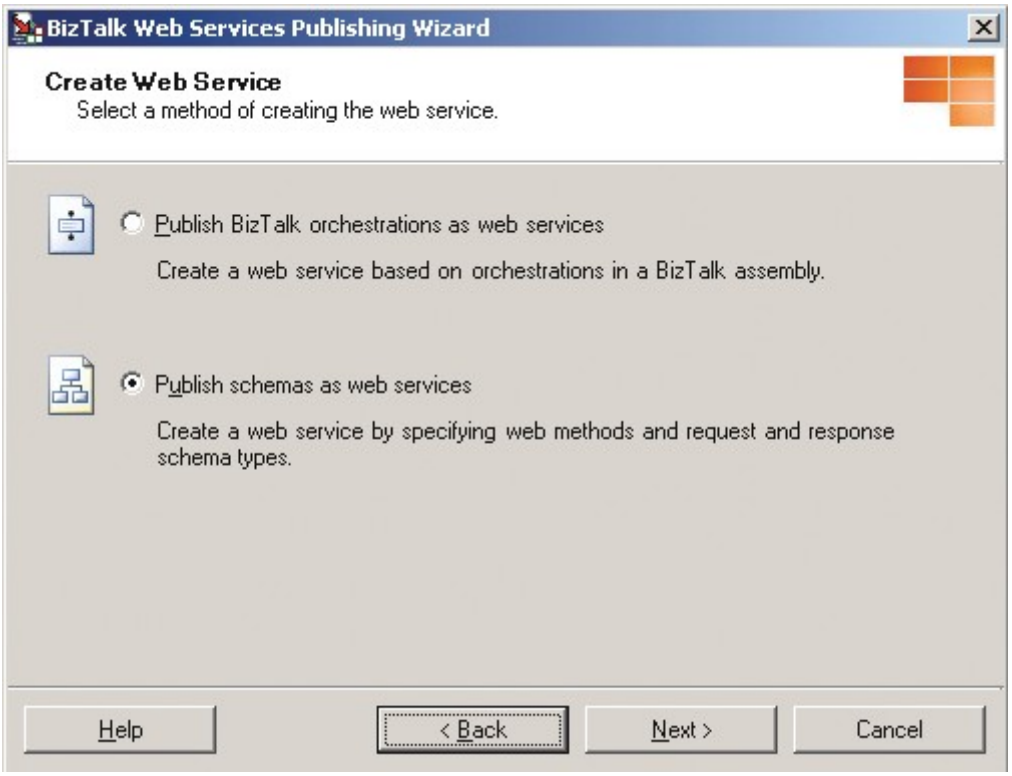
Orchestrace je analogie třídy a pro to by pro ni měla platit pravidla návrhu jako pro jakoukoliv třídu dle principů OOP (objektově orientované programování). Tzn. orchestrace by měla být od začátku navržena tak, aby nepřebírala více zodpovědností než je nezbytně nutné. Orchestrace, které tomuto požadavku neodpovídají, by měly být rozděleny do menších logických celků a řešit jen jeden z problémů, předchozí rozsáhlejší orchestrace.

Jedná se o aplikaci principu koheze (resp. jeho části, nazývané též „single responsibility principle"), dogmatické dodržování tohoto principu není nutné, je však nutné zabránit „přetížení odpovědnostmi", tj. vzniku orchestrace, která je kvůli zavlečené složitosti (složitost způsobená konkrétním řešením problému, nikoliv jeho podstatou (srovnej s inherentní složitostí = složitost vlastní danému problému)) příliš komplexní, což vede k problémům s udržovatelností, prováděním změn, oprav a provozováním takového řešení.

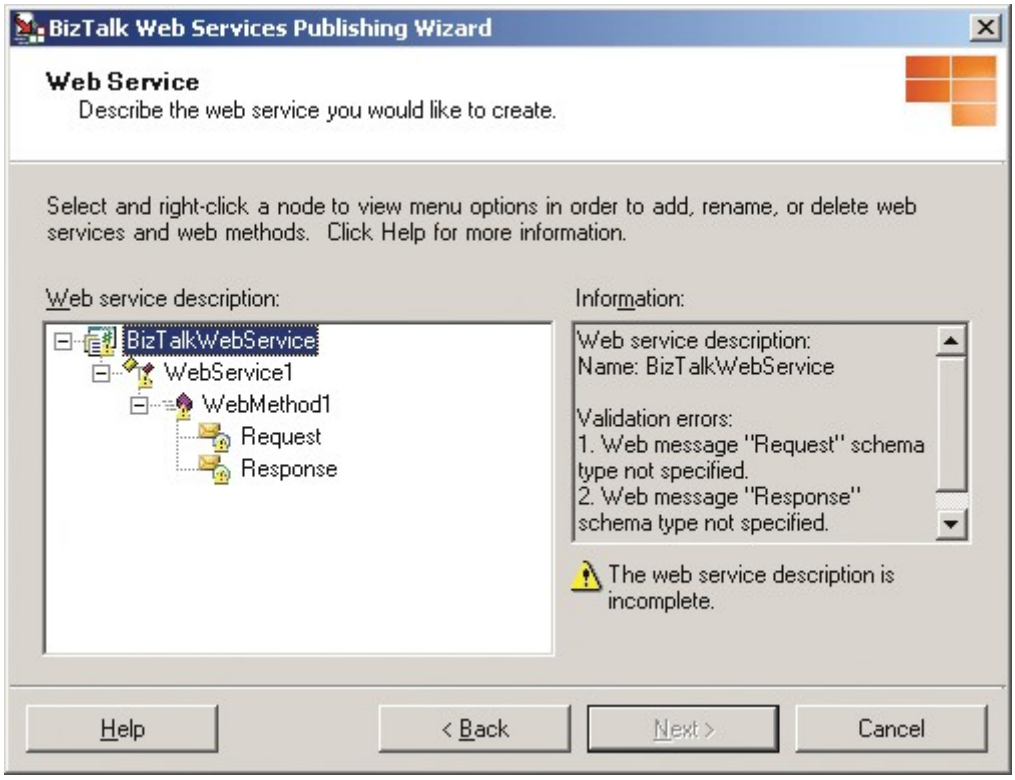
2.1.6 Webové služby

Webové služby použité jako receive či send porty orchestrací musí odpovídat obecným pravidlům definovaným v dokumentech „Standard metodiky vývoje" a „Programátorské konvence .NET 2.0" včetně jmenných konvencí pro namespace rozhraní a verze rozhraní.

Při definování webových služeb, jež jsou součástí řešení pro BizTalk, by měla být vždy publikována (externí) schémata, nikdy ne orchestrace. Toto doporučení umožňuje rozvolnit vazbu mezi schématem a procesem (orchestrací).



Obrázek 2-3 Web Services Publishing Wizard 1



Obrázek 2-4 Web Services Publishing Wizard 2

2.1.7Jazykové verze

Práce s jazykovými verzemi musí odpovídat principům popsaným v dokumentu „Standard metodiky vývoje".

2.2 Assemblies, struktura projektů

Projekty musí být členěny do assemblies dle logiky aplikace z pohledu závislostí jednotlivých částí řešení. Samostatné assemblies musí být vytvořeny pro externí schémata, interní schémata, pipelines, maps (transformace) a pipeline components, pokud jsou tyto prvky v řešení použity.

Orchestrace musí být v samostatné assembly či více assemblies. Při rozdělování řešení na jednotlivé assemblies je nutné minimalizovat závislosti dané vzájemnými referencemi – navzájem referencované orchestrace by měly být v jedné assembly.

Výjimkou jsou transformace použité v orchestraci pro sestavení jedné zprávy z několika vstupních zpráv (částí), takové transformace by měly být součástí assembly s orchestrací, neboť takovou transformaci je možné použít pouze z orchestrace a orchestrace bez takové transformace nemůže pracovat, funkční vazba je zde tedy těsná a rozdělení těchto artefaktů do samostatných assemblies nic neřeší.

Členění do assemblies dle dodavatele či autora částí řešení je nepřípustné.

2.3 Verzování

Přípustné jsou následující dva způsoby verzování, které je možné pro různé iterace a cykly jednoho řešení kombinovat.

Při nasazení nové verze aplikace je možné verzovat i send a receive porty resp. receive locations, pokud je to třeba pro paralelní běh původní a nové verze, není to však vyžadováno.

Volba strategie verzování artefaktů pro BizTalk a její dopady závisí vždy na konkrétní aplikaci. Pro snadné nasazování nových verzí je vhodnější verzování FileVersion. Pokud je nutný paralelní běh různých verzí binárních souborů, je jediným řešením verzování AssemblyVersion a FileVersion, to však vnáší nutnost pečlivě odladit závislosti (verzování schémat, transformací a rozhraní webových služeb atd.). Paralelní běh různých verzí procesů je možné docílit s jednou verzí binárních souborů (a tím zjednodušit nasazování pomocí verzování FileVersion) tak, že nová verze orchestrací obsahuje větve pro zpracování původního procesu (pro dokončení zpracování rozpracovaných procesů) i pro nový typ zpracování, to však vyžaduje včasné rozdělení odpovědností do samostatných orchestrací.

2.3.1 Verzování FileVersion

Při tomto způsobu verzování je měněn atribut FileVersion, atribut AssemblyVersion zůstává stejný. Toto verzování umožňuje při nasazení nové verze ponechat nasazené assemblies (typicky schémata, pipelines apod.) v původní verzi (pokud v nich není třeba provést změny) a nasadit upravené assemblies (orchestrace, transformace) bez

nutnosti rekompilování a nasazení závislých assemblies (neboť závislosti jsou vztažené k assembly strong name, které obsahuje AssemblyVersion).

Tento způsob nesmí být použit, pokud je nutné po nasazení nové verze dokončit zpracování instancí, které byly iniciovány před updatem.

Při tomto způsobu verzování může dojít k tomu, že BizTalk Server runtime může načíst nesprávnou verzi, pokud je nainstalováno více různých verzí stejné assembly. Nepoužívané verze musí být odstraňovány a po nasazení nové verze musí být restartovány host instances, které danou assembly používají.

Pro rozlišení konkrétních verzí je naprosto nezbytné důsledně navyšovat file version s každým buildem.

2.3.2 Verzování AssemblyVersion a FileVersion

Při tomto způsobu verzování jsou měněny atribut FileVersion i AssemblyVersion. Tento způsob verzování umožňuje paralelní běh více různých verzí instancí orchestrací, pro nové instance je použita nejnovější verze.

Tento způsob může vést k nutnosti nasazení nových verzí závislých assemblies, které referencují změněnou assembly, a tím zvyšovat náročnost postupu nasazení.

Verzování AssemblyVersion vyžaduje změnu bindings a změnu rozhraní webových služeb (v kódu vygenerovaném Web Services Publishing Wizardem je referencována AssemblyVersion assembly orchestrace, která byla publikována). Pokud jsou publikována schémata, je nutné vygenerovat novou verzi webové služby i v případě, že se změní verze AssemblyVersion assembly se schématy (na druhou stranu verze assembly se schématy by se měla měnit pouze tehdy, pokud se mění schémata, a pokud se mění schémata, měla by být změněna verze schématu v atributu targetNamespace atd., čímž dojde k vytvoření nové verze interface).

2.4 Jmenné konvence

Jmenné konvence musí odpovídat obecným jmenným konvencím a jmenným konvencím pro .NET. Názvy proměnných a konstant musí být vytvářeny dle obecných jmenných konvencí.

2.4.1 Assemblies

Assemblies jednotlivých částí řešení musí být pojmenovávány dle následujícího schématu:

AMCSSZ.ProductName.BizTalk.package

Jmenné konvence pro assemblies (Tabulka 2-1)

Šablona názvu souboru assembly	Obsah assembly
AMCSSZ.ProductName.BizTalk.InternalSchemas.dll	Interní schémata

AMCSSZ.ProductName.BizTalk.Pipelines.dll	Pipelines
AMCSSZ.ProductName.BizTalk.Pipeline.ComponentName.dll	Pipeline komponent
AMCSSZ.ProductName.BizTalk.Orchestrations.OrchestrationName.dll	Orchestrace
AMCSSZ.ProductName.BizTalk.Constants.dll	Konstanty použité v orchestraci

Pokud jsou konstanty sdílené s částí řešení mimo BizTalk (výčtové typy používané ve webovových službách i v orchestracích apod.), musí být umístěny v obecné assembly AMCSSZ.ProductName.Constants.dll. Pokud se jedná o konstanty používané pouze v části řešení využívající BizTalk server (např. Řetězcové konstanty s hodnotami pro binding filter pro direct binding apod.), musí být konstanty umístěny ve specifické assembly AMCSSZ.ProductName.BizTalk.Constants.dll.

2.4.2 Jmenné prostory

Jmenné prostory datových typů musí odpovídat názvům assembly:

AMCSSZ.ProductName.BizTalk.package

Jmenné prostory XML musí odpovídat následujícímu schématu

http://schemas.cssz.cz/ProductName/ProductVersion/SchemaName/SchemaVersion

2.4.3 Aplikace

Aplikace musí být pojmenovávány dle následujícího schématu:

AMCSSZ.ProductName

2.4.4 Artefakty

Názvy pro jednotlivé objekty BizTalku a design-time objekty (shapes) jsou shrnuty v následující tabulce:

Jmenné konvence pro názvy souborů v projektu (Tabulka 2-2)

Typ souboru v projektu	šablona názvu souboru	příklady
Soubor schéma	[RootNode].xsd	PO.xsd
Soubor property schema	[název schématu]_Properties.xsd	PO_Properties.xsd
Soubor transformace	[název schématu]_to_[SchemaName].btm	PO_to_ZDV.btm
Soubor orchestrace	[funkce orchestrace].odx	BatchReceivePO.odx
Soubor pipeline	[funkce pipeline].btp	ReceivePOBatch.

Jmenné konvence pro názvy artefaktů v administraci BizTalk Serveru (Tabulka 2-3)

Typ artefaktu	šablona názvu	příklad
Fyzický send port	[funkce portu][Adapter]Port	SendToArchiveFilePort
Send port group	[funkce portu]PortGroup	SendToArchivePortGroup
Receive location	[funkce portu][Adapter]Loc	ReceiveBatchFileLoc
Fyzický receive port	[funkce portu]Port	ReceiveBatchPort

Jmenné konvence pro názvy bloků v orchestraci (Tabulka 2-4)

Blok (shape)	šablona názvu	příklad
Receive shape	rcv[název zprávy]from[název systému či komponenty]	rcvPOfromBUS rcvPOContinueFromMSGBOX
Send shape	snd[název zprávy]to[název systému či komponenty]	sndPOAckToBUS sndPOUpdateToMSGBOX
Scope shape	scp[popisné jméno]	scpCallWebService
Transacion	ltx[název long running transaction] atx[název atomické transakce]	ltxConfirmation atxSavePO
Expression shape	ex[popisné jméno]	exLogException
Decide block	dcd[rozhodovací kritéria]	dcdOrganizationSize
If branch	If[popis podmínky]	IfInvalidSignature
Else branch	Else	Else
Construct message block	cm[název zprávy]	cmMsgPOBAMRequest
Asign shape	am[název zprávy]	amMsgPOBAMRequest
Transform shape	xm[název schématu]To[název schématu]	xm PO_to_ZDV
Call shape	call[název orchestrace]	callBatchReceivePO
Start shape	start[název orchestrace]	startBatchReceivePO

Throw shape	throw[typ výjimky]	throwInvalidOperationEx
Parallel shape	ll[popisné jméno]	llReceiveActivatingMessage
Loop block	loop[podmínky ukončení]	loopUntilAllReceived
Suspend shape	suspend[důvod]	suspendTargetUnreachable
Terminate shape	terminate[důvod]	terminateInvalidStatus
Call rules shape	rules[jméno Policy]	rulesBNOValidation
Compensate block	comp[názvy kompenzovaných transakcí]	compLtxConfirmation
Catch block	catch[typ výjimky]	catchTimeOutEx
Delay	delay[důvod]	delayTimeOut
Listen	lsn[popisné jméno]	lsnReceiveResponsesOrTimeout

Jmenné konvence pro názvy datových typů (Tabulka 2-5)

Datový typ	**šablona názvu**	**název**
Multipart message type	mpt[logický typ]	mpmPOType
Multipart message part	p[název schématu]	pPO
Message	msg[název schématu či typu] mpm[název schématu] msg[WS][Operation]Rq msg[WS][Operation]Rsp	msgPO mpmPO msgBAMSetActivityRq msgBAMSetActivityRsp
Port type	[funcke portu]PortType	ReceivePOBatchPortType
Logický port	[funcke portu]Port	prtReceivePOBatch
Correlation type	corr[název vlastností]Type	corrMessageIDBatchGuidType
Correlation set	corr[účel]	corrSendAllChildRecords
Název orchestrace	[Funkce orchestrace]	BatchReceivePO
Název typu orchestrace	[Funkce orchestrace]Type	BatchReceivePOType

Jmenný prostor (.NET namespace) orchestrace	AMCSSZ.ProductName. BizTalk.Orchestrations. [Funkce orchestrace] AMCSSZ.ProductName. BizTalk.Orchestrations. [Funkce sady orchestrací]	AMCSSZ.POProcessing. BizTalk.Orchestrations. BatchReceivePO AMCSSZ.POProcessing. BizTalk.Orchestrations. ReceivePO

Pokud je v jedné assembly jedna orchestrace, bude namespace orchestrace končit jménem orchestrace, a tím by maskovalo typ orchestrace, proto je nutné typ orchestrace vždy doplňovat koncovkou Type.

2.4.5 Reference na webové služby

Reference na webové služby je nutné udržovat v samostatných sdílených knihovnách a v rámci jednotlivých řešení referencovat tyto sdílené knihovny a to buď z orchestrace, nebo z mapy (transformace).

2.5 Logování, trasování

Pro logování je třeba používat Enterprise Library Logging Application Block. Enterprise Library Logging Application Block umožňuje konfiguraci cíle logování bez nutnosti změny kódu aplikace s využitím tzv. providerů. K dispozici jsou providers pro EventLog (protokol událostí systému Windows), soubory, databáze (ODBC, OLEDB, čili MS SQL, ale i ORACLE), ale i syslog. Je možné vytvářet vlastní providery.

Pro trasování (detailní výpisy zpracování) je možné používat třídu System.Diagnostics.Trace nebo trace level logování Enterprise Library Logging Blocku.

2.5.1 Konvence pro logování

Základní konvence zaznamenávání událostí je popsána v dokumentu „Standard metodiky vývoje". Vzhledem k tomu, že BizTalk aplikace je nasazena v prostředí webových služeb, vztahuje se na ni i příslušné požadavky na logování komunikace s webovými službami.

2.6 Aplikační konfigurace

Konfiguraci aplikace je možné udržovat

- V konfiguračních (XML) souborech, referenci na tento soubor je nutné konfigurovat plně kvalifikovanou cestou k souboru v BTSNtSvc.exe.config. Název soboru musí začínat jménem aplikace (AMCSSZ.ProductName), může obsahovat verzi, může obsahovat název modulu či komponenty (pokud se jedná o část konfigurace specifickou pro určitý fragment řešení), musí mít poslední příponu .config

- V konfiguraci pipeline komponent pro jednotlivé porty (per instance pipeline configuration)
- V úložišti SSO – výhodou je, že parametry jsou automaticky dostupné pro všechny BizTalk servery. Není potřeba již žádná další distribuce konfiguračních souborů. Konfigurační klíče a hodnoty jsou automaticky zašifrované.
- V úložišti Business Rules Engine (BRE) – konfigurace může být popsána i za pomoci business rules. To je obzvlášť vhodné v případech, kdy je konfigurace používána pro nastavení parametrů obchodní logiky (např. počet dní na vyřízení žádosti)

Není dovoleno používat následující úložiště konfiguračních dat

- Registry
- Konfigurační soubory v kořenovém adresáři systémového disku či v adresářích OS (C:\Windows) či platformy (C:\Program Files\Microsoft BizTalk Server 2006) apod., konfigurační soubory nesmí referencovány „implicitně" či částí cesty (cesta ke konfiguračnímu souboru musí být upravitelná správcem bez zásahu do kódu aplikace)

Pro umístění konfiguračních informací existuje následující doporučení

- „obchodní parametry řešení" – tj. hodnoty, které ovlivňují průběh zpracování z pohledu obchodní logiky (jako například relativní termíny, minimální, maximální a prahové hodnoty, intervaly, atd.) mají být umístěny v úložišti Business Rule Engine a spravovány jeho prostředky.
- „technické parametry řešení" – všechny informace technického charakteru, které souvisí s konfigurací řešení v daném prostředí (jako jsou například adresy spolupracujících systémů) nebo s provozovanou verzí daného řešení, mají být umístěny v konfiguračních souborech.

3. STANDARD PRO NASAZENÍ APLIKACÍ PRO BIZTALK DO PROVOZU

Nasazení BizTalk řešení se skládá z několika kroků, které jsou závislé na architektuře daného řešení. Kroky zahrnují

- Import informací o nových verzích assemblies do management databáze BizTalku
- umístění správně verzovaných DLLs do GAC na všech nodech BizTalk group
    - restart BizTalk host instancí (nutné z důvodu zavedení právě nainstalovaných dlls do paměti)
- nastavení nových nebo změněných binding informací (URI, nastavení security)
- konfigurace send portů

  - konfigurace příslušných adresářů systému souborů, nastavení ACL

- konfigurace receive portů a receive locations
  - konfigurace příslušných adresářů systému souborů, nastavení ACL
  - konfigurace virtuálních adresářů a application pools IIS, nasazení webových služeb
- konfigurace hosts a host instances
- konfigurace handlerů adaptérů pro nové hosts (SMTP, WCF, SOAP)

Cílem je zdokumentovaný a opakovaně použitelný postup nasazení. V případě potřeby lze nasadit jen určité (pouze změněné) části nebo knihovny projektu.

Pro nasazení BizTalk řešení je preferovaným způsobem instalace použití sady instalačních skriptů, které využívají nástroje příkazové řádky – BTSTask , GacUtil, InstallUtil atd. Celkový průběh instalace je řízen hlavním skriptem. Přípustné jsou skripty Windows Scripting Host (VBSript, JScript) a dávkové soubory (BAT, CMD). Preferovaným jazykem je VBScript.

Instalace bude zahrnovat zejména následující součásti:

a. Skript pro konfiguraci hosts, host instances a adapter handlerů prostředky WMI.
b. BTSTTask pro vytvoření aplikace a import assembly do BizTalk management DB.
c. GACUtil pro instalaci assembly do GAC.
d. Příkazy pro vytváření adresářových struktur, kopírování souborů (soubory webových služeb, konfigurační soubory).
e. CACLS pro nastaveni oprávnění (ACL).
f. Konfigurace virtuálních adresářů, webů a application pools IIS prostředky ADSI a WMI.
g. Skripty pro vytvoření receive portů a receive locations prostředky WMI.
h. Případné další nástroje a skripty pro specifickou konfiguraci.

Ke skriptům bude k dispozici dokumentace, která umožni správci platformy modifikovat postup, v jakém mají být skripty spouštěny (a zda vůbec). Kód ve skriptech musí být pečlivě okomentován a odzkoušen. Pokud jsou pro dokončení konfigurace nutné manuální kroky, musí být přesně popsány a doplněny snímky obrazovek.

Konfigurace jednotlivých prostředí BizTalk Serveru (integrační, školící a produkční) se vzájemně odlišují, proto pro ně musí existovat specifická nastavení. Tato nastavení budou uložena v souborech s konfigurací pro dané procesní řešení (BizTalk aplikaci) a

příslušné cílové prostředí instalace. Z hlediska nasazení BizTalk aplikace je jedním z nejdůležitějších tzv. binding file.

Instalační zdroje mohou obsahovat MSI balíčky, které budou provádět dílčí instalační úlohy. MSI balíčky budou obsahovat binární součásti instalace.a vzniknou vygenerováním z MS Visual Studia a/nebo exportem řešení z BizTalk Serveru. Součástí MSI jsou BizTalk artifacts (orchestrace, schémata, pipelines, maps), vlastní a související externí dlls, business rules. Je nutné, aby veškeré projekty, které jsou součástí řešení byly zahrnuty v jednom MSI balíčku. MSI balíček se pak postupně nasazuje jednou pro instalaci do BizTalk DB (Import v BizTalk MMC) a dále se spustí na každém nodu BizTalk group (Instalace do GAC).

Při tvorbě MSI balíčku je vhodné, aby na stroji, na němž je MSI balíček vytvářet, byl proveden deployment assembly tak, aby byla zaškrtnuty následující volby na první obrazovce importu assembly do GAC. Volby jsou:

- Add to the global assembly cache on add resource
- Add to the global assembly cache on MSI file import
- Add to the global assembly cache on MSI file install

Jsou-li všechny volby zaškrtnuty, je omezena možnost vzniku problémů při importu řešení, které může být provedeno více způsoby.

Při tvorbě MSI balíčku procesního řešení BizTalk je nutné vybrat zdroje, které do něj budou zahrnuty. Může jít o:

- Samotné workFlow řešení (assembly)
- Pravidla (rules)

Jednoznačným doporučením je do exportovaného řešení zahrnout všechny součásti, které jsou exportovacím nástrojem nabídnuty (zpravidla jde o export všech součástí procesního řešení).

Před exportem je vhodné nastavit všechny orchestrace na „BizTalkServerApplication". Pokud bude proveden export se specifickým hostem pro danou aplikaci, bude třeba před importem řešení z MSI balíčku mít v cílovém systému tohoto specifického hosta s naprosto totožnými parametry.

3.1Aplikace

Jednotlivá řešení musí být nasazována jako samostatné aplikace.

Pokud jsou současně nasazeny 2 či více verzí jednoho řešení, mohou být nasazeny jako jedna aplikace, či mohou být pro každou verzi vytvořeny samostatné aplikace. Název aplikace by v druhém případě měl obsahovat hlavní číslo verze aplikace.

3.2 Hosts a Host Instances

Jednotlivá řešení musí používat samostatné hosts (in-process i out of process (isolated)). In process host instance odpovídá instanci Windows služby (Win32 service) BizTalk Serveru, isolated host instance odpovídá application pool IIS. Host jakožto konfigurační kontejner umožňuje nastavení parametrů ovlivňujících výkon (throttling) i distribuci zátěže (škálování mezi jednotlivými nody BizTalk group).

Jednotlivé aplikace mohou používat více hosts a host instances pro lepší škálovatelnost (oddělení logických celků aplikace, oddělení hosts pro receive adaptéry, send adaptéry a orchestrace).

Pokud aplikace vyžaduje specifická nastavení výkonnostních parametrů (throttling), musí být tato nastavení dodána jako součást instalačního postupu.

V každém z prostředí (integrační, školící, produkční) bude vždy existovat aplikační host pojmenovaný „BizTalkServerApplication", kterému byly nastaveny následující parametry:

- Tracking: FALSE
- Trusted: YES

Tento host má nastavení totožná pro všechna prostředí. Jde o základní aplikační host, pod kterým budou dodávána všechna řešení s cílem zajištění snadného přenosu z jednoho prostředí do jiného prostředí. Následně, je-li třeba zajistit specifická nastavení výkonnostních parametrů (throttling), je možné vytvořit další hosty, které budou specifické např. pro jednotlivé aplikace/jednotlivé orchestrace atd.

3.3 GAC

Při reinstalaci řešení je vhodné v GAC stroje, na který jsou importována BizTalk řešení ověřit, zda v ní není importováno více již neplatných assembly daného projektu. Jsou umístěny v: %SYSTEMROOT%\assembly

Tento postup platí zejména v případě, že došlo k odebrání stávající verze orchestrace. Ne vždy je řádně odebrána assembly i z GAC. Je tedy nutné vždy ověřit, zda jsou v GAC pouze platné assembly daného řešení.

Správnost assembly je možné ověřit prostřednictvím atributu „Version", který musí mít nastaven každá assembly. V případě vzniku nového řešení musí být vždy atribut „Version" povýšen.

3.4 Databáze

Pokud aplikace pro prostředí BizTalk Serveru používá vlastní databázi („stavová" či „procesní" databáze apod.), je možné tuto databázi nasadit na samostatný databázový server (cluster) či je možné nasadit databázi na existující či novou instanci SQL Serveru 2005 existujícího clusteru, který je využíván BizTalk servery.

Pokud je databáze nasazena na samostatném serveru či clusteru, je pro fungující autentizaci a autorizaci nutné buď zajistit integrovanou autentizaci příslušného účtu

BizTalk hostu (tj. SQL server musí být ve stejné doméně, jako BizTalk servery, a v databázi musí být nastavena pro účet potřebná oprávnění), či je nutné v connectionstringu uvést platné přihlašovací údaje (pro standalone SQL server).

Pokud je databáze nasazena na instanci clusteru, který je využíván BizTalk servery, je nutné, aby byla nasazena v instanci, která není používána BizTalk serverem (tj. žádná z instancí, která slouží pro BizTalk Server, nesmí zároveň obsluhovat jiné databáze, než databáze produktu Microsoft BizTalk Server). Konfigurace oprávnění je stejná jako v případě samostatného serveru ve stejné doméně, tj. účet BizTalk hostu musí mít oprávnění pro přístup k datům v databázi.

4. STANDARD PRO PROVOZOVÁNÍ APLIKACÍ PRO BIZTALK

4.1 Monitorování správnosti běhu

Zásadním druhem monitorování správného běhu aplikací pro BizTalk jsou správné výsledky zpracování předávaných zpráv. V této oblasti je nezastupitelná úloha uživatelů systému a odborných garantů jednotlivých aplikací, kteří mohou být první, kdo zachytí nesprávné chování systému.

BizTalk spolehlivě loguje do protokolu událostí operačního systému jakékoliv problémy při zpracování (runtime výjimky, chyby v síťové komunikaci, problémy s konfigurací apod.). Pravidelné sledování protokolu (např. Pomocí monitorovacích nástrojů) je základem úspěšného dlouhodobého provozování stabilního řešení.

Při chybách ve zpracování jsou instance procesů, ve kterých došlo k výjimce, suspendovány. Suspendované instance je možné vyhledat a zobrazit pomoci BizTalk Server Administration Console a aplikace HAT.

Pokud nasazené aplikace využívají trasování, je možné na produkčním prostředí bez ovlivnění výkonu zachytit podrobné ladicí informace.

Tracking (konfigurovatelný z BizTalk Server Administration Console) je možné vypnout pro zvýšení výkonu platformy, ale pokud je zapnutý, může poskytnout cenná data pro řešení problémů aplikací, ale i pro monitorování business procesů. Tracking je možné konfigurovat pro jednotlivé orchestrace a porty včetně zaznamenávání dat zpráv (audit, debug), je však vždy nutné předem zajistit dostatečnou kapacitu pro zpracování těchto dat (disková kapacita, kapacita CPU instance s tracking databází).

Specifické informace o průběhu zpracování mohou poskytovat logy jednotlivých aplikací. Dle konfigurace a implementace aplikací se může jednat o protokoly událostí či o textové soubory logu.

Dalším zdrojem informací pro sledování běžného provozu jsou HTTP logy BizTalk serverů (pro hledání chyb, ale i rutinní monitoring) a dále log SQL serverů.

Pro monitorování výkonu jsou k dispozici kromě systémových performance counters (disk, CPU, RAM, network) a performance counters SQL Serveru (cache, indexes, compilations) i specifické performance counters BizTalk Serveru.

4.2 Tracking, DTA a HAT

Informace o průběhu zpracování dokončených instancí jsou ukládány v DTA databázi a mohou být zobrazeny pomocí nástroje HAT.

Tyto informace musí být odstraňovány z produkční databáze, součástí dokumentace BizTalku jsou postupy pro archivaci a odstraňování takových dat – tyto postupy je možné automatizovat či je provádět ručně http://msdn2.microsoft.com/en-us/library/aa560754.aspx . Dále existují postupy pro opětovné načtení archivovaných dat - http://www.gotdotnet.com/codegallery/codegallery.aspx?id=67bbd6ea-850e-4d93-be87-df6788976cab .

4.3 Zpracování suspendovaných zpráv

Zpracování suspendovaných zpráv je specifické pro danou aplikaci. Dodavatel aplikace musí definovat, které instance je možné terminovat, které musí být resumovány. Všechny ostatní případy mimo definované musí být řešeny v rámci podpory či jako chyby dodaného řešení.

Při rutinním provozu je nezbytné pravidelně monitorovat počty suspendovaných instancí.

4.4 Zálohování BizTalk

Pro zálohování databází BizTalk Serveru jsou připraveny a zdokumentovány postupy, které zajistí konzistenci záloh všech databází k danému časovému okamžiku, včetně postupů pro obnovu systému ze zálohy http://msdn2.microsoft.com/en-us/library/aa562140.aspx . Konfigurace zálohování je podrobně popsaná v dokumentaci http://msdn2.microsoft.com/en-us/library/aa546765.aspx .

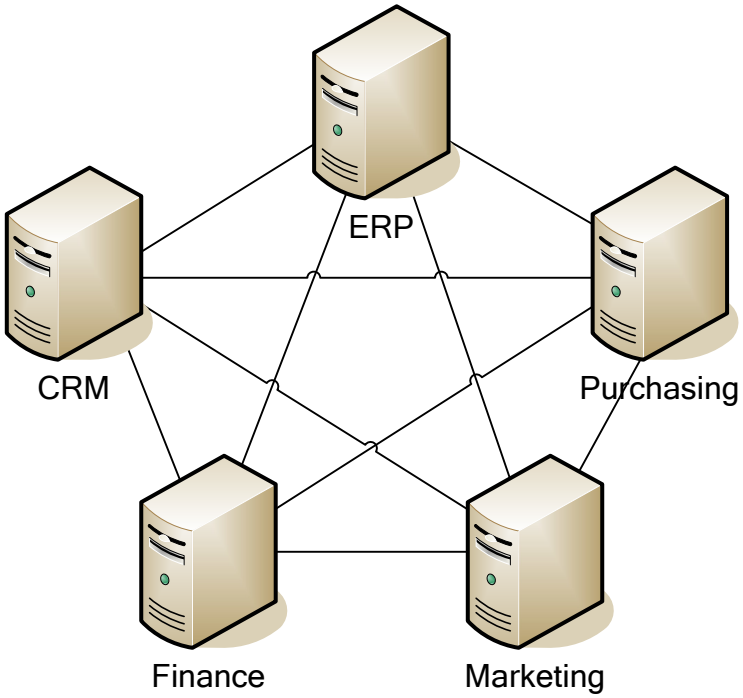
Zálohování dalších databází na společném clusteru není součástí zálohovacích postupů pro zálohování databází BizTalk Serveru a musí být řešeno ve spolupráci s dodavatelem aplikace.

5. PŘÍLOHY

5.1 Nedoporučené vzory pro integraci aplikací

5.1.1 Point-to-point

V prvotní implementaci je poměrně výkonné řešení a nevyžaduje velké počáteční investice. Proto se může jevit jako snadné a vhodné, ale takovéto řešení se brzy stane neudržovatelné, nepřehledné a náročné na změny. Největší nevýhodou je, že podporuje tvorbu těsných vazeb mezi systémy. Nepracuje se s jedním obecným formátem zprávy, ale naopak každá zpráva je překládána z výstupního formátu jednoho systému do vstupního formátu druhého systému a naopak. Složitost propojování těchto systémů pak roste kvadratickou závislostí.



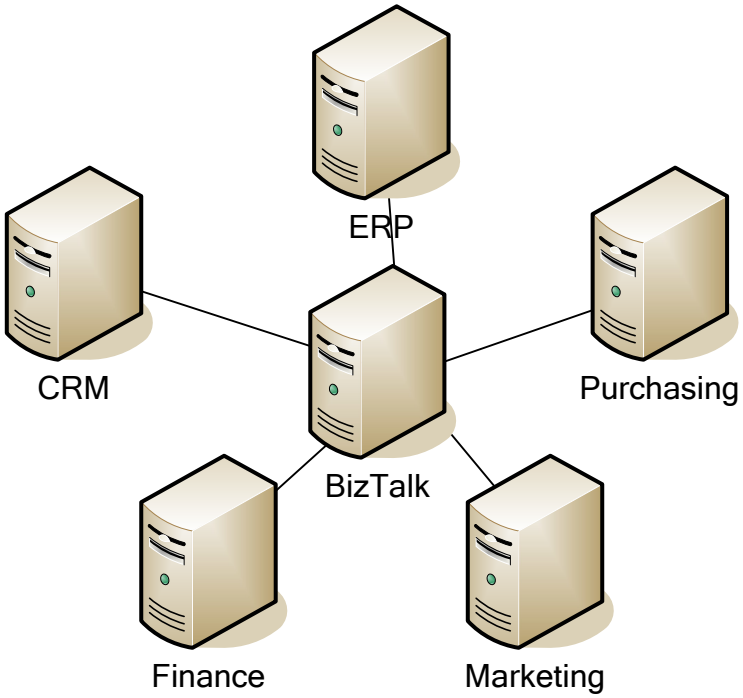
Obrázek 5-1 Point to point integrace

5.2 Doporučené vzory pro integraci aplikací

5.2.1 Hub-and-Spoke

V tomto uspořádání je mezi systémy prostředník, který rozumí jednomu společnému formátu zprávy. Pro ostatní systémy je pak nutné zabezpečit překlad z a do tohoto obecného formátu. Tímto způsobem odpadá přímá komunikace systémů mezi sebou navzájem. Počáteční investice je však vyšší. Obrovskou výhodou však je (kromě snížení komunikačních uzlů), že je pod vaší úplnou kontrolou formát zprávy, který je verzovatelný. V praxi tento formát může vycházet z nějakého standardizovaného formátu, ale vždy jej máte možnost přizpůsobit tak, aby zpráva mohla obsahovat vše potřebné pro to, aby mohl proběhnout úplný přenos dat do všech systému v řešení. Pokud dojde ke změně požadavku pak se, formát vámi vlastněné zprávy upraví, a změní se interní číslo verze formátu. Relevantní cílové systémy jsou pak poupraveny tak aby dokázaly zpracovat nově přidané údaje zprávy.

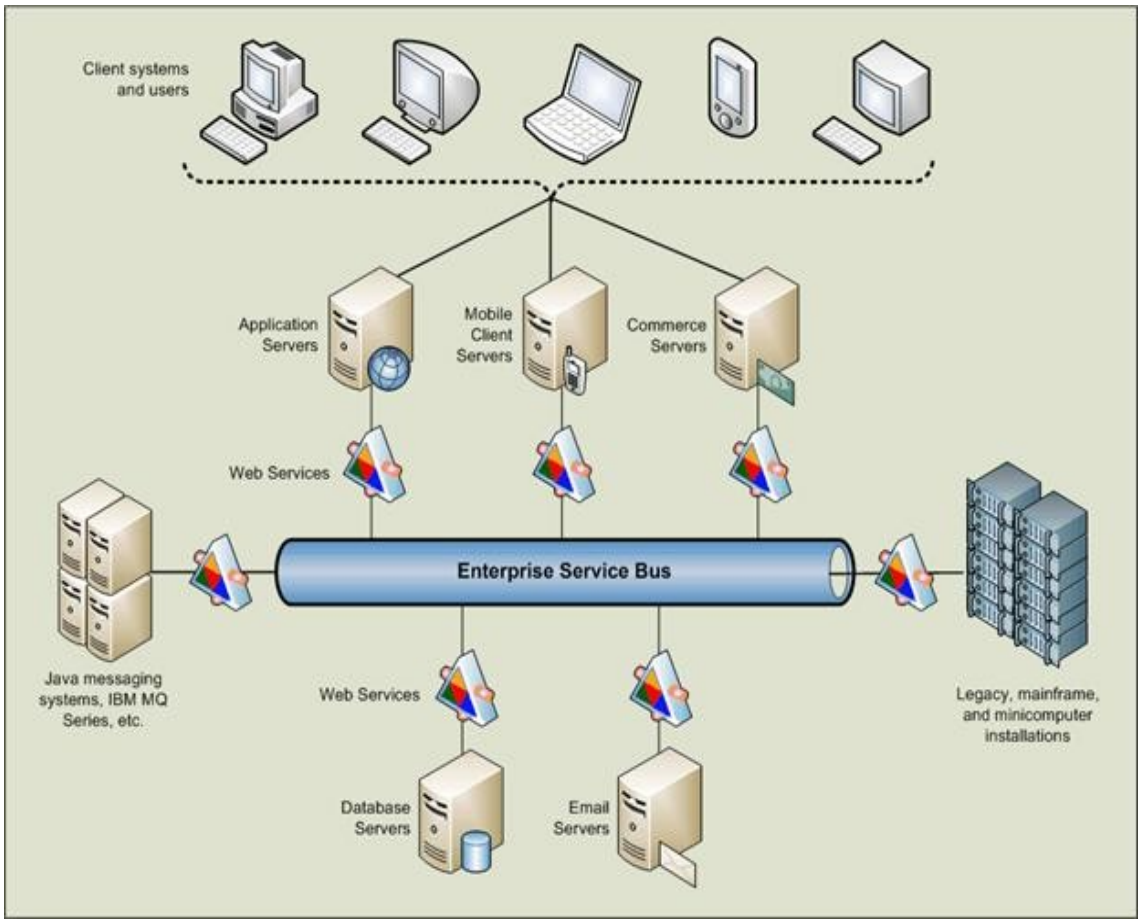
Tento vzor přináší větší volnost mezi vazbami, lépe se udržuje a je snazší provádět zásahy do projektu. Další nezanedbatelnou výhodou pak je, že díky centrálnímu uzlu (přes který prochází veškerá komunikace) lze zabezpečit monitorování, audit, lepší zabezpečení toku dat, atd.



Obrázek 5-2 Hub and spoke integrace

5.2.2 Enterprise Service Bus

Jedná se o vzor, který pomáhá zajistit lepší škálovatelnost řešení spolu se snazší rozšiřitelností a udržovatelností. Je velmi podobný vzoru hub-and-spoke. Myšlenka je, že jednotlivé systémy neadresují cílové systémy přímo, ale jen odešlou zprávu na sběrnici (reprezentovanou v tomto případě BizTalk Serverem). Ty systémy, které potřebují určitý druh zprávy se pak mohou přihlásit k odběru této zprávy (může jich být i více) a ostatní systémy naopak mohou zprávu ignorovat. Není tedy vůbec nutné, aby jednotlivé systémy o sobě musely vědět a už vůbec není nutné znát formát cílového systému. To umožňuje velmi dobře přidávat a odebírat jednotlivé části z nebo do sběrnice (zapojování nových systémů a nahrazování stávajících systémů novými verzemi).



Obrázek 5-3 ESB integrace aplikací

Dokumentace včetně způsobu implementace je popsána na http://www.codeplex.com/esb/Release/ProjectReleases.aspx?ReleaseId=5176

5.3 Nedoporučené vzory pro aplikace pro BizTalk

5.3.1 God object

V případě BizTalku je tento vzor popisován též jako God Orchestration. Řešení sestává z jedné či několika málo orchestrací, které jsou přetíženy odpovědnostmi. V důsledku zavlečené složitosti pak obsahují větvené rozhodovací bloky a opakující se sady stavebních bloků (shapes). Přidáním další funkčnosti jsou kopírovány celé větve zpracování či vytváření proměnných pro nesení hodnot pro další rozhodovací bloky. Řešení pak často obsahuje více shapes pro zajištění toku zpracování požadovaným směrem, než těch, které jsou určeny pro vlastní zpracování (práce se zprávami).

Problematická je nejenom údržba takového řešení, ale i reálný provoz, neboť je nutné rozlišovat mezi suspendovanými instancemi v různých místech zpracování, což je v praxi při ev. vyšším počtu suspendovaných instancí a malých zkušenostech operátorů první úrovně podpory často zásadní nevýhodou.

5.4 Doporučené vzory pro aplikace pro BizTalk

Vzory pro BizTalk mají stejné charakteristiky, jako kterékoliv jiné návrhové vzory. Z nich jednou z nejdůležitějších je ta, že vzor vychází z osvědčeného řešení opakovatelného problému. Rozpoznání vzoru problému je tedy nutnou podmínkou pro správnou aplikaci vzoru řešení, selhání v této oblasti často vede k situaci, že aplikace nevykazuje pozitivní znaky vzorů, které jsou v ní použity, neboť tyto jsou použity špatně – a to připadá vždy na vrub dodavatele aplikace.

Vzory řešení nejsou často jedinou možností pro algoritmické řešení problému a nemusí být ani možností ideální, ani optimální. Mnohdy je možné pro jednu konkrétní situaci volit několik vzorů či jejich kombinací, které mohou mít v daném použití různé výhody i nevýhody.

Přínos vzorů je při jejich správném použití nesporný (zkrácení vývoje, opakovaná použitelnost částí řešení), za správné použití však vždy odpovídá architekt a vývojář konkrétní aplikace.

5.4.1 Kombinace direct binding, rozdělení řešení do více orchestrací, používání multipart messages, interních schémat, mapování na portech a verzování file version

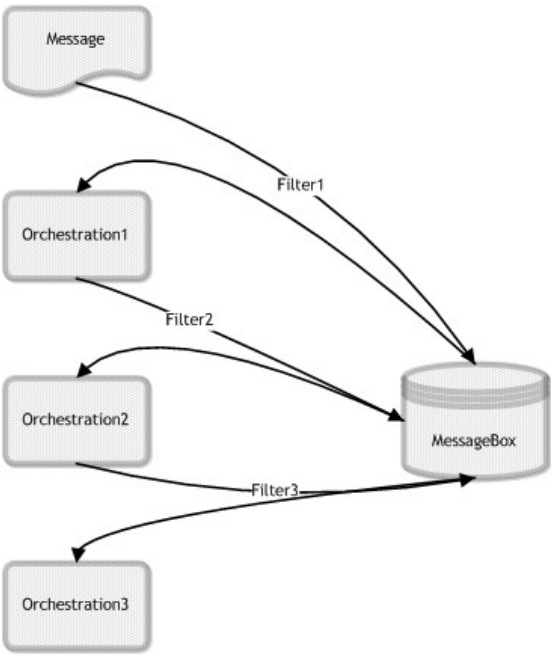
Kombinací postupů určených v předcházejících oddílech tohoto dokumentu je možné dosáhnout značné flexibility architektury řešení.

Multipart messages umožňují přidávání dalších parts (částí) s rozšiřováním informací přenášených zprávami bez nutnosti přepojování všech send/receive shapes a portů, které změněný typ zprávy používají.

Messagebox direct binding je obdobou loose coupling, odstraňuje závislosti mezi orchestracemi, čímž umožňuje verzování jednotlivých assemblies samostatně.

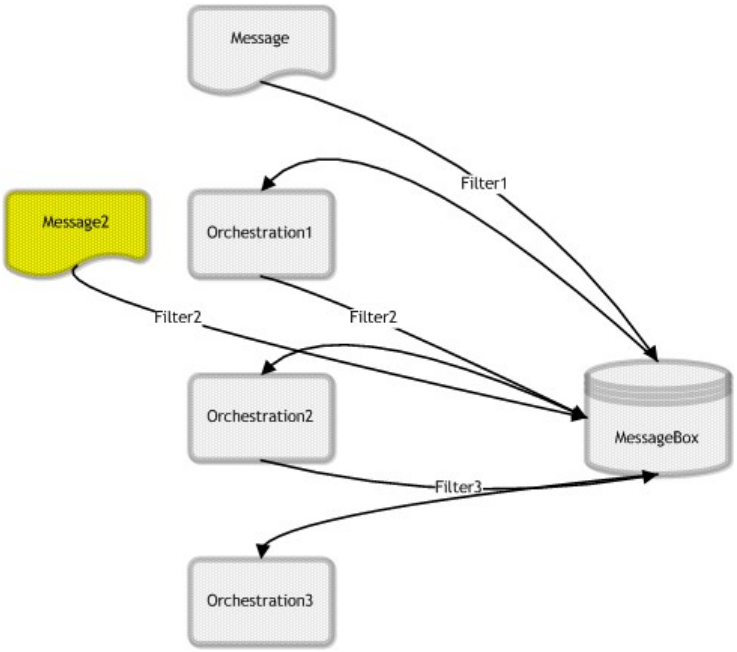
Používání interních schémat vede k menší náchylnosti řešení k fragilitě na základě změny schématu externího systému (obdoba problému typu fragile base class). V případě změny externího schématu není nutné měnit interní schémata a orchestrace, mění se pouze externí schémata a transformace (mapy) (samozřejmě pokud změna nevyžaduje též zpracovávání rozšířených informací, které nová verze rozhraní externího systému přináší).

Verzování pomocí FileVersion attributu zajišťuje snadný způsob nasazování úprav a přitom stále nese informaci o konkrétním buildu, což umožní udržovat konzistenci verzí v různých prostředích.



Obrázek 5-4 Architektura procesu implemenovaného pomocí direct binding a více orchestrací

Architektura založená na direct binding také umožňuje vstupy do zpracování v definovaných bodech. Pokud jsme schopni sestavit zprávu (všechny požadované části zprávy) tak, aby na základě definovaných kritérií filtru vstoupila do zpracování ve zvoleném bodě, může takové řešení sloužit ke zpracování neočekávaných stavů a k ev. opakování zpracování některých částí procesu (tam, kde je to z hlediska business procesu přípustné): při výjimce je zpracování ukončeno, po odhalení a odstranění příčiny (nedostupnost částí systému, oprava chyby v orchestraci apod.) je sestavena vstupní zpráva pro konkrétní bod zpracování a ta je přes definovaný port publikována do messageboxu. Na základě definovaných filtrů je zpráva předána nové instanci přislušné orchestrace. Tato architektura vhodně doplňuje kompenzace transakcí tam, kde opravdu obchodní proces kompenzace používá.



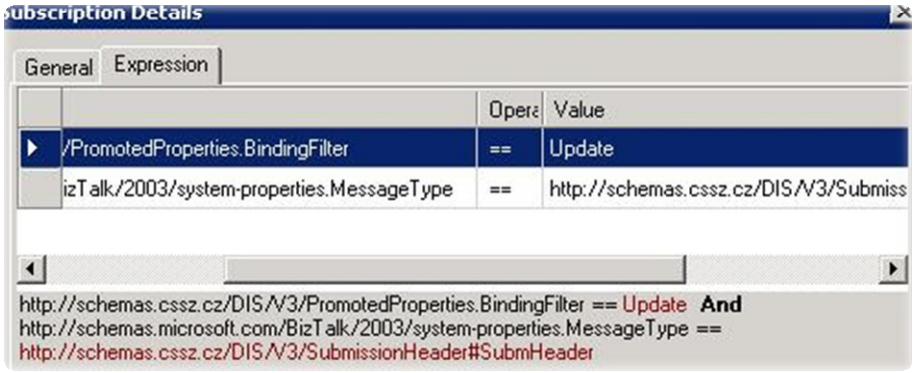
Obrázek 5-5 Alternativní vstup zprávy do zpracování (procesu)

Popis: Přes fyzický port, který není navázán (binding) na žádnou ochestraci, je přijata zpráva (a pomocí custom pipeline jsou promotovány příslušné vlastnosti, včetně vlastnosti BindingFilter – hodnota Filter1), tj. zpráva je publikována do messageboxu.

Orchestrace Orchestration1 má receive port nakonfigurován s nastavením binding na hodnotu direct a partner orchestration port na hodnotu messagebox. Odpovídající receive shape má nastaven filtr pro vlastnost BindingFilter na hodnotu Filter1. Po publikování zprávy je nalezena tato subskripce, proto je vytvořena instance orchestrace a ta je spuštěna pro danou zprávu. Orchestrace provede zpracování (volaní webových služeb, zápis do stavové databáze apod.) a vytvoři novou zprávu jako kopii vstupní zprávy, ve které změní hodnotu BindingFilter. Tuto novou zprávu odešle přes messagebox direct bound port.

Orchestrace Orchestration2 má receive port nakonfigurován s nastavením binding na hodnotu direct a partner orchestration port na hodnotu messagebox. Odpovídající receive shape má nastaven filtr pro vlastnost BindingFilter na hodnotu Filter2. To zajistí vytvoření subskripce, inicializaci orchestrace a zpracování zprávy. Řetěz takto pokračuje dle počtu zapojených orchestrací.

Pokud je do messageboxu přes fyzický receive port bez explicitního bindování publikována taková zpráva, která odpovídá typu vstupní zprávy pro orchestraci Orchestration2, a má vlastnost BindingFilter nastavenu na hodnotu Filter2, dojde k inicializaci nové instance orchestrace Orchestration2 a ke zpracování zprávy touto orchestrací.



Obrázek 5-6 Subskripce pro orchestraci s messagebox direct bound receive portem a nastaveným filtrem

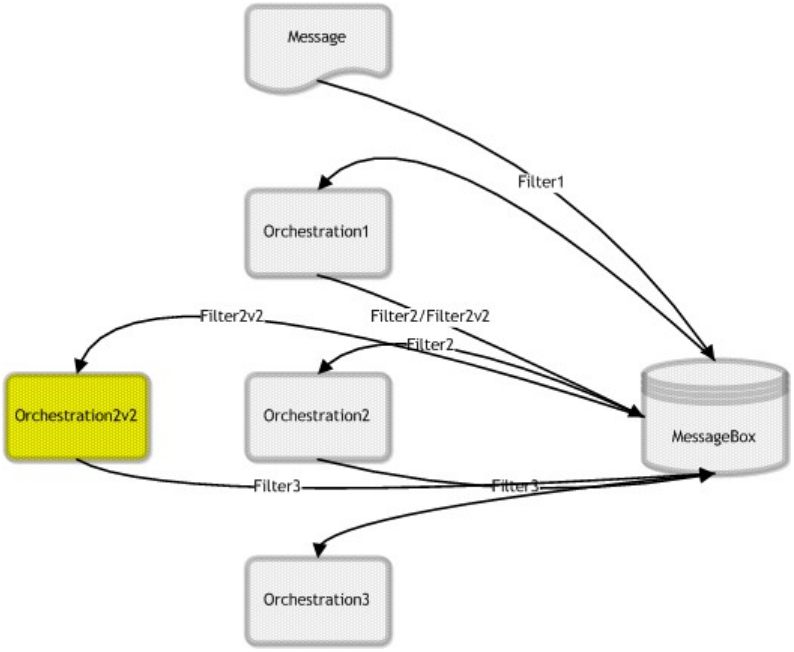
5.4.2 Paralelní běh různých verzí procesu

Paralelní běh různých verzí procesu je možné zajistit několika způsoby.

Ne vždy je vhodné mít přímou implementaci, tj. verze procesu odpovídající verzi sady orchestrací. Toto řešení vyžaduje verzování AssemblyVersion. V komplexních řešeních zahrnující více schémat a externí webové služby toto řešení vede k vytváření závislostí napříč verzemi aplikací (pokud se s každou verzí nemění zároveň i schémata a rozhraní webových služeb) a rychle se stává nespravovatelným, jeho další rozvoj je omezen a každá další verze výrazně zvyšuje komplexnost celého řešení a tím i náchylnost k chybám.

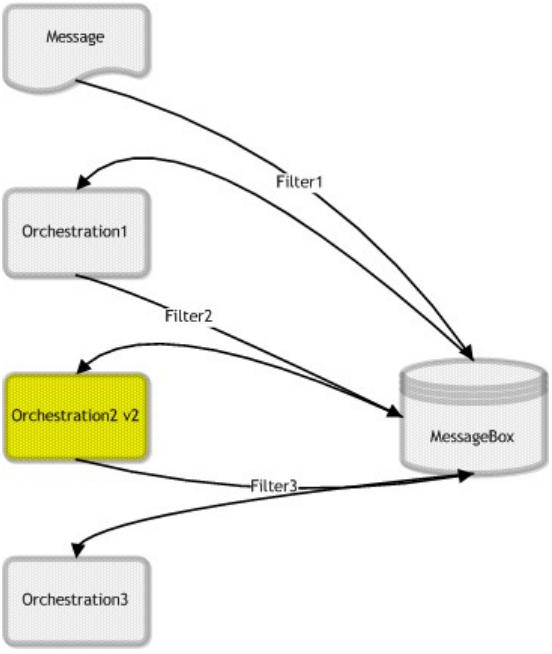
Naopak vhodným řešením se jeví kombinace interních schémat, mapování na portech, rozdělení do více orchestrací a direct binding. Nová verze procesu znamená nasazení nových verzí některých orchestrací v řešení, které zajistí routování dle verze procesu buď na původní nezměněné orchestrace nebo na nové orchestrace s novou funkčností. Takové řešení lze dlouhodobě efektivně rozvíjet a spravovat a zároveň si nutně nevynucuje verzování AssemblyVersion, takže nasazování nových částí nevytváří závislosti.

Použití architektury řešení rozděleného na jednotlivé orchestrace podle odpovědnosti, které spolu komunikují mechanismem direct binding přes messagebox, umožňuje různé možnosti řešení požadavků na verzování obchodního procesu. Oddělení artefaktů do samostatných assemblies v takovém případě umožňuje při nasazování změn ponechat schémata a tím též rozpracované instance (pochopitelně pouze ty před měněným blokem a po něm).



Obrázek 5-7 Verzování procesu vložením orchestrace pro novou verzi části procesu

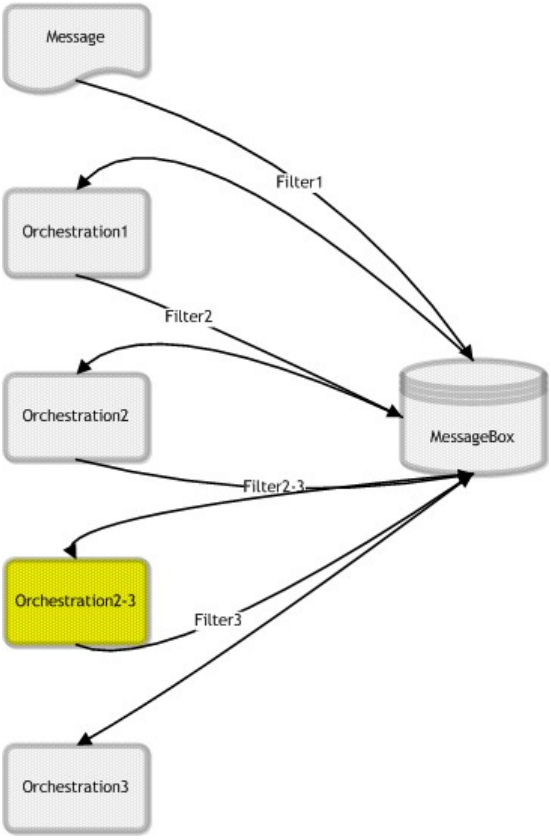
Nasazením nové orchestrace pro novou verzi části procesu paralelně k původní orchestraci a úpravou předcházející orchestrace tak, aby na základě obchodní logiky správně nastavila filtr pro původní verzi či novou verzi, je možné zajistit zpracovávání zpráv odpovídající verzí procesu.



Obrázek 5-8 Verzování procesu nahrazením orchestrace pro část procesu

Výměnou orchestrace za novou verzi je možné zajistit zpracování zpráv podle nové verze procesu. Pokud bude nová verze obsahovat rozhodovací bloky a různé větve

zpracování, může zajistit i zpracování zpráv odpovídající verzí procesu, přitom nasazení této verze může znamenat pouze výměnu assembly s upravenou orchestrací.



Obrázek 5-9 Verzování procesu vložením orchestrace pro novou část procesu

Nasazením nové orchestrace pro novou část procesu a úpravou následující orchestrace (kritéria filtru zpráv) je možné zařadit do procesu novu logiku. Pokud bude nová orchestrace obsahovat rozhodovací bloky a větve zpracování pro různé verze procesu, může toto řešení zajistit zpracování zpráv odpovídající verzí procesu.

5.4.3 Vzor rodič – potomek (parent - child)

Vzor rodič – potomek je typickým vzorem pro zpracování sady (dávky) datových vět. Vstupem je zpráva obsahující sadu datových vět, každou větu je třeba zpracovat samostatně a zajistit zpracování všech vět v dávce.

V prostředí BizTalk Serveru je toto možné řešit různými způsoby a závisí na dodatečných požadavcích, jaký způsob je vhodný.

Pokud není nutné zachovat obálku a po zpracování všech vět pokračovat ve zpracování dávky, je vhodným a výkonným způsobem použití disassembleru v pipeline. Tento výkonný způsob neumožňuje sesbírání výsledků zpracování jednotlivých vět a následné pokračování jednoho procesu pro celou dávku, pokud není zároveň použita stavová databáze (pro dávku i jednotlivé věty).

Pokud část procesu zpracování zahrnuje práci s celou dávkou a část s jednotlivými větami, je možné použít řešení, ve kterém jsou jednotlivé datové věty ve smyčce hlavní (rodič) orchestrace odeslány podřízené orchestraci, následně jednotlivě zpracovány podřízenou orchestrací (potomek), která odešle odpovědi zpět hlavní orchestraci. Jednoduchá implementace s korelací je možná s využitím direct-bound portů s nastavením partner orchestration port na porty odpovídající orchestrace či předáváním callback portu jako parametru. Taková implementace vyžaduje sdílení stejné assembly pro orchestrace či reference mezi assemblies. Implementace s využitím direct boud portů s vazbou na messagebox odstraňuje toto omezení, bez nutnosti referencí mezi assemblies je možné orchestraci potomka nahradit libovolně dlouhým řetězem odchestrací (opět s využitím direct binging), a přitom zachovat callback.

5.4.4 Ošetření výjimek

Ošetření výjimek v aplikacích implementovaných na platformě Microsoft BizTalk Server 2006 se neliší od odšetřování výjimek v jakémkoliv obdobném prostředí (serverová aplikace bez uživatelského rozhraní).

Výjimky způsobené chybějícím popisem větví zpracování či neúplným výčtem možností v analýze procesu jsou odchyceny uživateli či správcem systému a reportovány jako nesprávné zpracování. Při analýze problému je třeba zajistit definici požadovaného chování aplikace pro takové případy, následně vyvinout, otestovat a nasadit upravenou verzi aplikace, která již řeší takové případy správně. Obecný způsob zpracování nespecifických výjimek (odchycení, zalogování, opakované vyvolání výjimky (rethrow – neboť nebyla zpracována)) v takovém případě pomůže dohledat příčinu (chybějící větev pro zpracování speciálního případu apod.), pokud je logování dostatečně podrobné.

Výjímky způsobené chybu vývojáře jsou odchyceny uživateli či správcem systému a reportovány jako nesprávné zpracování či nesprávné chování aplikace. Obecný způsob zpracování nespecifických výjimek (odchycení, zalogování, opakované vyvolání výjimky (rethrow – neboť nebyla zpracována)) v takovém případě pomůže dohledat příčinu (logická chyba v implementaci procesu apod.), pokud je logování dostatečně podrobné.

Výjimky způsobené stavem okolního prostředí a aplikací mimo BizTalk Server pomáhá prostředí Microsoft BizTalk Server řešit mimo jiné možností konfigurace opakování komunikace na portech (retry count a interval, backup transport atd.). V orchestraci je možné definovat odchytávání výjimek komunikace na portech (DeliveryNotificationException) konfigurací vlastností logického portu (DeliveryNotification = Transmitted). Způsob ošetření takové výjimky po zachycení není specifický pro BizTalk, ale pro jednotlivé aplikace – v některých procesech je implementováno opakovatelné zpracování, takže je možné ukončit běh orchestrace (samozřejmě po korektním zalogování informací o výjimce).

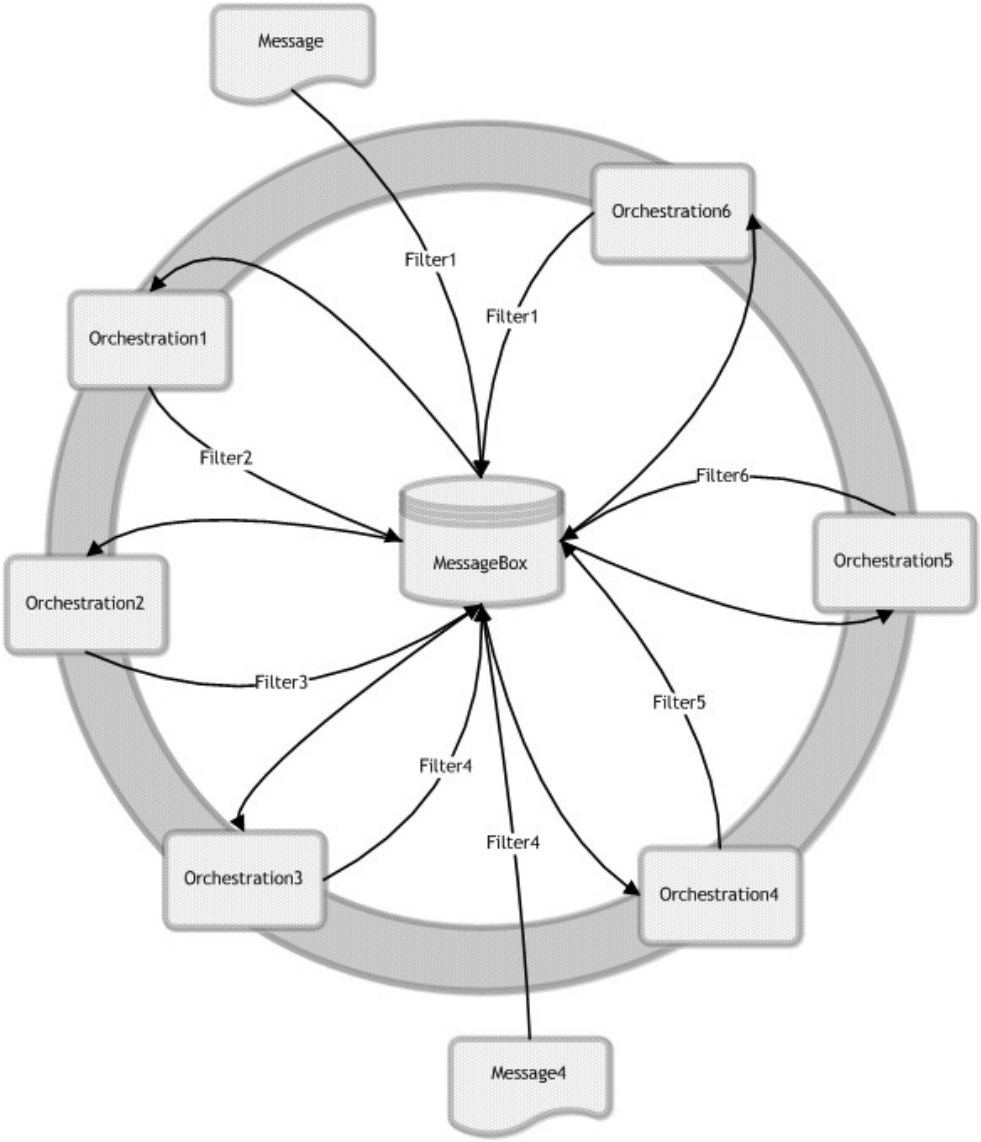
Výjimky způsobené stavem prostředí BizTalk Serveru (nedostatek paměti, chybějící konektivita k systémovým databázím BizTalk Serveru apod.) není možné v aplikacích řešit (handlovat). BizTalk Server loguje takové výjimky do logů operačního systému. Řešení suspendovaných instancí orchestrací po takové události spočívá v analýze stavů jednotlivých instancí a dle implementace procesu v terminování instancí a opakovaném spuštění procesu či v pokusu o pokračování zpracování. Pro tento postup je nutná

spolupráce správce prostředí se skupinou podpory dané aplikace (se znalostí procesu zpracování a možností terminování , opakování a existujících vstupních bodů).

5.4.5 Cykly a smyčky

Pravé cykly (smyčky) jsou v obchodních procesech poměrně vzácné, většinou se jedná o nepravé cykly, kdy v opakovaném zpracování nejsou data zprávy či kontext (v rozšířeném smyslu slova – tj. i stavy systémů včetně těch mimo BizTalk, či např. verze procesu či knihoven) shodné s původní zprávou (opakované zpracování po opravě chybných dat, opravě procesu či opravě chyby v aplikaci apod.). Takové řešení lze dobře implementovat pomocí rozdělení do více orchestrací a direct binding, neboť je možné odkudkoliv publikovat takovou zprávu, že bude zpracována některou z orchestrací předcházející v pořadí té aktuální, a tím dojde k vytvoření smyčky procesu. Stejně tak umožňuje architektura založená na direct binding a rozdělení do více orchestrací implemetovat pravé cyklické procesy.

Rozdělení procesu do více orchestrací navíc přidá flexibilitu pro zastavení procesu v konkrétním definovaném bodě (zastavením konkrétní orchestrace) pro nasazení nové verze některých částí. Tím lze dosáhnout cyklického procesu s možností přerušení a aktualizace. Stejně jako u necyklické obdoby této architektury je možné do procesu vstoupit v různých bodech, pokud je možné sestavit odpovídající vstupní zprávu.



Obrázek 5-10 Architektura využívající direct binding pro cyklický process implementovaný v samostatných orchestracích

5.4.6 Kód příkladu

Pro snazší pochopení popsaných principů výše uvedených vzorů je přiložen kód příkladu, který implementuje některé z nich. Ačkoliv kód příkladu vychází z řešení, které splňuje požadavky daného procesu (funkční ale i výkonnostní parametry), je příklad bez jakýchkoliv záruk, jedná se o ukázku implementace. Při použití části či celku kódu je nutné provést veškeré požadované testy (výkonnostní, integrační apod.) a vyhodnotit je v kontextu požadavků konkrétního řešení. Jedná se o příklad implementace některých vzorů, pro správnou funkčnost v rámci konkrétního řešení je nutné jej uzpůsobit (schémata, multipart message types apod.) a optimalizovat.

Vzory a principy použité v příkladu:

- Direct binding na messagebox pro komunikaci mezi orchestracemi, použití vlastnosti BindingFilter (definované v property schématu) pro řízení zpracování

- Orchestrace v samostatných assemblies bez vzájemných referencí, schémata oddělená v samostatné assembly (příklad používá pouze interní schémata)

- Logování s využitím Enterprise Library Logging blocku

- Odchycení výjimky a její zpracování (zalogování a odeslání zprávy na port). Pozn.: součástí řešení není MIME send pipeline pro zápis multipart message do jednoho souboru ani MIME receive pipeline pro čtení multipart message ze souboru, potřebné pipeline komponenty jsou součástí instalace Microsoft BizTalk Serveru 2006 a jejich použití je dokumentováno v nápovědě produktu.

- Komunikace rodič – potomek (parent - child) se zpětným voláním (callback) v samostatných orchestracích s využitím direct binding. Pozn.: vzor nepokrývá vlastní rozpad těla dávky na jednotlivé datové věty, to je možné provést několika způsoby dle konkrétního procesu, jedním z nich je volání pipeline z orchestrace a zpracování zpráv takto vzniklých ve smyčce v orchestraci, což je technika, které je popsána v příkladech v dokumentaci produktu (SDK). Pro inicializaci correlation setu je použito odeslání zprávy na bindovaný port, tato zpráva neslouží k dalšímu zpracování, v produkčním prostředí je tedy nutné použít pro tento port null adaptér nebo pravidelně odstraňovat zprávy z adresáře tohoto portu.

Popis příkladu:

Orchestrace BeginProcess přijme vstupní zprávu (filtr „BeginProcess"), zaloguje informaci o zahájení zpracování, a pokud je identifikátor v těle zprávy platný, odešle zprávu do messageboxu s filtrem „Splitter", pokud identifikátor platný není, vyhodí výjimku. Případná výjimka je odchycena, je sestavena zpráva z původní zprávy a informace o výjimce a tato zpráva je odeslána na port.

Orchestrace Splitter přijme vstupní zprávu (filtr „Splitter") a provede inicializaci correlation setu s filtrem „WorkerCorrelated" (pro vytvoření per instance subskripce pro receive port). Následně ve smyčce publikuje 10 zpráv s filterm „Worker" a informacemi pro callback (partner port a partner service) do messageboxu. Poté spočítá čas pro timeout a následuje smyčka s receive portem či timeoutem pro příjem odpovědí. Receive port je nastaven jako direct-bound self-correlated kvůli vyvtoření per instance subscriptions.

Orchestrace Worker přijme vstupní zprávu (filtr „Worker") a sestaví odpověď s filtrem „WorkerCorrelated" a nastavenými informacemi pro callback. Tuto zprávu publikuje do messageboxu, při odesílání je inicializován correlation set, který není nikde sledován, což je technika pro promotování vlastností (properties) z orchestrace (jinak je to možné pouze z pipeline či adaptéru).

Orchestrace Splitter a Worker navzájem nejsou referencovány, což umožňuje rozšíření řešení např. tak, že orchestrace Worker bude publikovat zprávu s filtrem „Worker1" pro

další orchestraci a až ta bude publikovat zprávu s příslušným filtrem a nastavením pro callback, to vše opět bez vzájemných referencí (obdoba loose coupling mechanismu).

Instalace příkladu:

1. Nainstalujte Enterprise Library 3.1 a zaregistrujte assemblies v GAC
2. Nakonfigurujte Enterprise Library Logging Block pro aplikaci
    a. Do konfiguračního souboru BTSNTSvc.exe.config přidejte definici handleru Microsoft.XLANGs.BizTalk.CrossProcess.XmlSerializationConfigurationSectionHandler a konfigurační sekci pro konfiguraci AppDomains s nastavením konfiguračního souboru pro appdomain
    b. Do konfiguračního souboru aplikace přidejte definici handleru konfigurace logging blocku Microsoft.Practices.EnterpriseLibrary.Logging.Configuration.LoggingSettings a vlastní konfiguraci logování (je možno vytvořit pomocí nástroje Enterprise Library Configuration)
3. Nasaďte a nakonfigurujte aplikaci AMCSSZ.Patterns
    a. Proveďte deployment schémat a orchestrací a nakopírujte knihovnu pomocných funkcí do GAC. Pro tento krok je připraven pomocný skript deploy.cmd.
    b. V administrační konzoli BizTalku vytvořte receive port, receive location a send porty
        i. receive port a receive location s FILE adaptérem a XMLReceive pipeline pro vstup zpráv (prtReceive)
        ii. send port pro zápis zpráv, jejichž zpracování selhalo (prtSendErr) s FILE adaptérem a pass-through (či custom MIME encode) pipeline
        iii. send port pro odchytávání zpráv pro inicializaci correlation setu a pro odchytávání zpráv callbacku po timeoutu v hlavní orchestraci (prtSendOut) s FILE (či NULL) adaptérem, pass-through pipeline a filtry nastavenými na AMCSSZ.Patterns.BizTalk.InternalSchemas.BindingFilter == WorkerCorrelated
    c. Nabindujte orchestrace na hosty a porty, nastartuje aplikaci a restartujte host instance
4. Vyzkoušejte následující scénáře zasláním zpráv dle schématu
    a. Zpráva s BindingFilter=BeginProcess
    b. Zpráva s BindingFilter=Splitter

c. Zpráva s BindingFilter=BeginProcess a prázdným elementem MsgId

5.5 Odkazy

Scénáře pro orchestrace a referenční použití vzorů

http://msdn2.microsoft.com/en-us/library/aa561965.aspx

Business Process Management Solution

Scénář vzorového řešení s řídící orchestrací, umožňující rozšiřitelnost procesu přidáním dalších orchestrací a úpravou konfigurace, s řízením přerušení zpracování.

http://msdn2.microsoft.com/en-us/library/aa559473.aspx

Pattern Wizard a některé vzory pro orchestrace

http://www.codeplex.com/PatternWizard

Brožura Moderní integrace aplikací – Michal Juřek, Microsoft

http://download.microsoft.com/download/8/6/c/86c09926-affc-4e14-bec0-3c45cd989436/Moderni_integrace.pdf

5.6Struktura instalační dokumentace

1 Účel dokumentu
2 Obsah dokumentu
  2.1 Mapa instalačních kroků
3 Pojmy a zkratky
4 Požadavky řešení
  4.1 Požadavky na prostředí
  4.2 Cílové prostředí (HW a SW)
    4.2.1 Software:
    4.2.2 Hardware (vytvořen cluster):
    4.2.3 Konfigurace:
      4.2.3.1 účty,
      4.2.3.2 oprávnění
  4.3 Cílové prostředí – okolní systémy
5 Architektura a prostředí dávkových úloh
  5.1 Architektura a prostředí
  5.2 Architektura procesního řešení
6 Formát dodaných komponent
  6.1 Doporučené znalosti pro instalaci jednotlivých komponent
7 Instalace řešení na platformu BizTalk Server 2006
  7.1 Informace před zahájením instalace - definované proměnné
  7.2 Základní instalační proměnné

7.3 Další instalační proměnné:
7.4 Zdroje pro instalaci, instalační média
7.5 Předpoklady a požadavky pro instalaci proceního řešení
7.6 Orchestrace vyžadují pro instalaci následující předpoklady:
7.7 Nastavení prostředí pro instalaci
7.8 Konfigurace
8 Instalace Webových služeb
8.1 Přípravné kroky
8.2 Webové služeby umístěné na BizTalk Serveru
8.3 Tvorba WebSite/VirtualDirectory
8.4 Postup instalace a konfigurace Webových služeb
8.4.1 MSI
8.4.2 binární podoba
8.5 Instalace a konfigurace dalších webových služeb
8.6 Po-instalační kroky
8.7 Update Webových služeb
9 Instalace workflow
9.1 Všeobecné informace
9.2 Postup instalace (bodově)
9.3 Konfigurace receive a send portů – platí pouze pro případ instalace z assembly a portbindins souborů (nikoliv pro MSI balíčky)
9.4 Správnost nastavení portů
9.5 Implementace business pravidel
9.5.1 Instalace knihovny pro vyhodnocování pravidel
9.5.2 Nasazení a konfigurace business pravidel
9.5.3 Nasazení a konfigurace nové verze pravidel
9.6 Instalace orchestrací
9.6.1 Postup instalace pomocí MSI balíčku
9.6.2 Postup ruční instalace procesního řešení
9.6.3 Společné kroky následující po importu procesního řešení využitím MSI balíčku/ruční implementace
9.6.4 Spuštění orchestrací
9.6.5 Restart služby BizTalk serveru
9.6.6 Update workflow
9.7 Instalace databáze
9.7.1 Instalace využitím interpreta osql.exe
9.7.2 Instalace prostřednictvím SQL Management studia
9.8 Činnosti po ukončení instalace BTS řešení
9.8.1 Restart služby BizTalk serveru
9.8.2 Restart IIS
9.8.3 Spuštění orchestrací
9.8.4 Otestování procesního řešení

5.7Struktura Provozní dokumentace

1 Úvod a účel – provozní dokumentace
1.1 Popis prostředí řešení
1.2 Schéma okolí aplikace
1.3 Popis spolupracujících modulů
2 Provozní dokumentace
2.1 Nástroje
2.2 Sledované ukazatele a jejich vyhodnocení (podrobněji viz samostatná kapitola)
2.3 Administrační zásahy (podrobněji viz samostatná kapitola)
3 Provozní příručka
3.1 Všeobecné pokyny

3.2 Logování systémů souvisejících s APV
3.3 Stručný přehled použitelných správcovských nástrojů
3.4 Logování APV a okolních systémů
3.4.1 Logování webových služeb
3.4.1.1 Popis
3.4.1.2 Použití
3.4.2 Logování aplikačního serveru IIS
3.4.2.1 Popis
3.4.2.2 Použití
3.4.3 Logování BizTalk serveru
3.4.3.1 Popis
3.4.3.2 Použití
3.5 Relevantní správcovské nástroje
3.5.1 Správcovská konzole „Services And Applications"
3.5.2 Správcovská konzole „Internet Information Services"
3.5.3 Správcovská konzole „BizTalk Administration Console"
3.5.4 Správcovský nástroj „Health and Activity Tracking"
3.5.4.1 Zobrazení běžících orchestrací
3.5.4.2 Debuging orchestrací
3.6 Správa prostřední
3.6.1 Přednastavení receive/send portů
3.6.2 Konfigurace řešení
3.6.2.1 Konfigurace portů
3.6.2.2 Konfigurace orchestrací
3.6.2.3 Konfigurace IIS
3.6.3 Tvorba uživatelského účtu
3.6.4 sledování LOGů
3.6.5 sledování HATu
3.6.6 sledování procesní databáze (je-li k dispozici)
3.6.7 sledování EventViewer
3.6.8 Ověření správnosti procesu
3.7 Sledované ukazatele
3.8 Dislokace a umístění APV
3.9 Určení úrovně řešení
3.9.1 denní administrace
3.9.2 expertní podpora
3.10 Instalace nových verzí
3.10.1 Instalace WorkFlow
3.10.2 Implementace nové verze Bindings
3.10.3 Instalace nové verze WebService
3.11 Řešení problémů
3.11.1 zásahy v případě suspend (resumable)
3.11.2 zásahy v případě suspend (not resumable)
3.11.3 stavy active a dehydrated
3.12 Chyby, které mohou nastat
3.12.1 Chyby odhalitelné v EventViewer (např. ve spojení s IIS)
3.12.2 Stručný potup instalace z MSI vytvořeného v BizTalk Serveru
3.13 Kontaktní osoby

5.8 Příklady konfigurace

5.8.1 Referencování konfiguračního souboru aplikace v BTSNTSvc.exe.config

```
<?xml version="1.0" ?>
```

```
<configuration>
<configSections>
<section name="xlangs"
type="Microsoft.XLANGs.BizTalk.CrossProcess.XmlSerializationConfigurationSectionHandler,
Microsoft.XLANGs.BizTalk.CrossProcess" />
</configSections>
<xlangs>
<Configuration>
<AppDomains AssembliesPerDomain="10">
<AppDomainSpecs>
<AppDomainSpec Name="AMCSSZ" SecondsIdleBeforeShutdown="60"
SecondsEmptyBeforeShutdown="60">
<BaseSetup>
<ConfigurationFile>C:\FileDrop\LoggingApp.config</ConfigurationFile>
</BaseSetup>
</AppDomainSpec>
</AppDomainSpecs>
<PatternAssignmentRules>
<PatternAssignmentRule AssemblyNamePattern="AMCSSZ.*" AppDomainName="AMCSSZ" />
</PatternAssignmentRules>
</AppDomains>
</Configuration>
</xlangs>
</configuration>
```

5.8.2 Konfigurace logování Enterprise Library

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
<configSections>
<section name="loggingConfiguration"
type="Microsoft.Practices.EnterpriseLibrary.Logging.Configuration.LoggingSettings,
Microsoft.Practices.EnterpriseLibrary.Logging, Version=3.1.0.0, Culture=neutral,
PublicKeyToken=b03f5f7f11d50a3a" />
<section name="dataConfiguration"
type="Microsoft.Practices.EnterpriseLibrary.Data.Configuration.DatabaseSettings,
Microsoft.Practices.EnterpriseLibrary.Data, Version=3.1.0.0, Culture=neutral,
PublicKeyToken=b03f5f7f11d50a3a" />
</configSections>
<loggingConfiguration name="Logging Application Block" tracingEnabled="true"
defaultCategory="General" logWarningsWhenNoCategoriesMatch="false">
<listeners>
<!-- -->
</listeners>
<formatters>
 <!-- -->
</formatters>
<categorySources>
<add switchValue="Error" name="Exception">
<listeners>
<add name="Error Rolling Flat File Trace Listener" />
</listeners>
</add>
<add switchValue="All" name="General" />
</categorySources>
<specialSources>
<allEvents switchValue="All" name="All Events">
```

```
<listeners>
<add name="Info Rolling Flat File Trace Listener" />
<add name="System.Diagnostics TraceListener" />
</listeners>
</allEvents>
<notProcessed switchValue="All" name="Unprocessed Category" />
<errors switchValue="All" name="Logging Errors &amp;amp; Warnings" />
</specialSources>
</loggingConfiguration>
</configuration>
```