UNIVERZA V LJUBLJANI
EKONOMSKA FAKULTETA

MAGISTRSKO DELO

OBLIKOVANJE METODOLOŠKEGA OKVIRA VPELJAVE STORITVENO USMERJENE ARHITEKTURE V ORGANIZACIJI

LJUBLJANA, oktober 2009 MITJA ŠTURM

IZJAVA

Študent Mitja Šturm izjavljam, da sem avtor tega magistrskega dela, ki sem ga napisal pod mentorstvom prof. dr. Jurija Jakliča in skladno s 1. odstavkom 21. člena Zakona o avtorskih in sorodnih pravicah dovolim objavo magistrskega dela na fakultetnih spletnih straneh.

V Ljubljani, dne 01. 10. 2009

Podpis: _____

Vsebina

UVOD

Uspešnost in rast podjetja ali organizacije sta v današnjem času, ko so spremembe v poslovnem okolju hitre in pogoste, večinoma odvisni od zmožnosti prilaganja in odzivanja organizacije na spremembe poslovnih zahtev in zahtev trga. Tesna povezanost med informacijsko tehnologijo (v nadaljevanju IT) in poslovanjem organizacije je ključ za doseganje takšne prilagodljivosti ali agilnosti organizacije. Iz tega sledi, da je izziv, s katerim se organizacije srečujejo, kako zagotoviti usklajenost in rast okolja IT z rastjo poslovanja organizacije.

V večini današnjih večjih in tudi nekaterih manjših organizacij temelji okolje IT na večjem številu različnih programskih rešitev ali informacijskih sistemov, ki podpirajo glavne poslovne funkcije znotraj organizacije. Glavna značilnost teh sistemov je zapletenost in toga infrastruktura. Poleg tega je taka infrastruktura v večini primerov povezana s slabo povezljivostjo, katere spreminjanje in prilagajanje je drago, zahtevno in včasih tudi nevarno.

Doseganje tesne povezanosti med okoljem IT in poslovnim svetom je mogoče s pomočjo integracije programskih rešitev (Thomson, 2008, str. 1). Pojem integracija programskih rešitev lahko definiramo kot možnost povezovanja programskih rešitev, ki so bile načrtovane in razvite neodvisno druga od druge, ki lahko uporabljajo različne tehnologije in so neodvisno upravljane (Thomson, 2008, str. 3).

Obstajajo različni pristopi in arhitekture, ki nam lahko olajšajo integracijo programskih rešitev. Eden izmed načinov temelji na uporabi storitveno usmerjene arhitekture ali krajše SOA (angl. *Service Oriented Architecture, SOA*). S pomočjo SOA lahko dosežemo, da je integracijo programskih rešitev lažje upravljati, da je lažja za izvedbo in tudi bolj predvidljiva. Ko govorimo o SOA, ne govorimo o tehnologiji ali konkretni rešitvi, ampak o arhitekturnem pristopu, ki vpliva na oblikovanje rešitve. Vpeljava SOA v organizacijo je lahko dolgotrajen postopek, ki ne vpliva samo na informacijsko tehnologijo znotraj organizacije, ampak tudi na poslovne procese in ljudi znotraj organizacije. Glavna prednost SOA je ta, da skrajša čas in napor, potreben za izvedbo sprememb, ki jih narekuje poslovno okolje.

Ker gre pri vpeljavi SOA za dolgotrajen postopek oziroma projekt, je treba natančno vedeti, kako obvladovati posamezne faze, sicer se lahko takšen projekt hitro sprevrže v neuspeh. Zaradi tega potrebujemo neke temelje ali metodološki okvir, s katerim bomo zajeli glavne faze in aktivnosti ter, kar je najpomembnejše, izkoristili prednosti, ki nam jih SOA ponuja (Erl, 2008).

Ko govorimo o SOA, ne govorimo o konkretni rešitvi, ampak o pristopu (Bradley, Schulte, Sholler & Malinverno, 2008, str. 3). Zaradi tega in ker se današnje organizacije zelo razlikujejo med seboj, je težko in do neke mere tudi nesmiselno iskati ali predpisati nek

univerzalen pristop vpeljave SOA v organizacijo. V magistrski nalogi sem na osnovi realnega projekta in literature oblikoval metodološki okvir vpeljave SOA v organizacijo. Definiral in analiziral sem posamezne faze vpeljave tako s semantičnega, kot tudi z organizacijskega in tehnično-tehnološkega vidika. Predvsem pa sem želel poudariti posebnosti vpeljave SOA in razvoja programskih rešitev, ki temeljijo na SOA, ter opozoriti na pasti in težave, ki se lahko v takih postopkih pojavijo in na katere moramo biti še posebno pozorni. S konkretnim projektom vpeljave SOA v večje slovensko podjetje sem metodološki okvir tudi ovrednotil in dokazal smiselnost uporabe takšnega pristopa pri reševanju težav, povezanih z integracijo programskih rešitev. Na osnovi ovrednotenja sem lahko ugotovil, ali je metodološki okvir dovolj dobro zasnovan in ali bi bilo treba določene faze vpeljave še dodatno razširiti in podrobneje definirati.

V prvem delu magistrsko delo vsebuje analitični pregled SOA. Na osnovi strokovne literature, obstoječe najboljše prakse in strokovnih člankov, predvsem tujih strokovnjakov iz področja SOA, so prestavljene glavne značilnosti arhitekturnega pristopa. V drugem delu pa je predstavljen metodološki okvir, ki je nastal na osnovi realnega primera vpeljave SOA v večje slovensko podjetje in najboljše prakse uporabe obstoječih metodologij. Natančno so opisane posamezne faze metodološkega okvira in glavne aktivnosti ter izdelki posamezne faze.

Magistrsko delo je sestavljeno iz štirih glavnih poglavij. V uvodnem poglavju sem predstavil obravnavano problematiko. Ker je za uspešnost podjetij in organizacij integracija programskih rešitev ključnega pomena, sem predstavil, kako lahko vpeljava SOA v organizacijo pomaga pri integraciji programskih rešitev in samo integracijo tudi olajša. V naslednjem poglavju sem najprej predstavil, kaj metodološki okvir sploh je. Sledilo pa je poglavje, kjer sem oblikoval metodološki okvir, ki nam pomaga pri vpeljavi SOA v organizacijo. Najprej sem opredelil in definiral posamezne korake ali faze metodološkega okvira skupaj s pripadajočimi aktivnostmi in izdelki ter predstavil ustrezne metode in diagramske tehnike za izdelavo posameznih izdelkov. Poleg opisa posameznih faz sem v predzadnjem poglavju celoten metodološki okvir tudi ovrednotil s praktičnim primerom. Ovrednotenje metodološkega okvira sem izvedel na osnovi realnega projekta vpeljave SOA v podjetje Mladinska knjiga Založba. Podjetje razpolaga s številnimi informacijskimi rešitvami in zaradi nujne potrebe po integraciji, tako poslovnih procesov kot tudi podatkov, se je za rešitev za probleme odločilo za vpeljavo SOA. Za vsako fazo metodološkega okvira sem na osnovi konkretnega primera opisal, katere aktivnosti so bile izvedene in kateri izdelki so bili izdelani. V zadnjem poglavju sem predstavil še organizacijski vidik metodološkega okvira, v katerem sem opisal, katere so organizacijske posebnosti pri vpeljavi SOA v organizacijo in na kaj je treba biti še posebno pozoren. Prav tako sem z namenom boljšega razumevanja in ovrednotenja organizacijskega vidika podal nekaj praktičnih primerov.

1. STORITVENO USMERJENA ARHITEKTURA

SOA je arhitekturni pristop, kjer so programske rešitve tako načrtovane, da nudijo storitve z zadostno stopnjo strukturiranosti, katere so lahko pozneje uporabljene s strani drugih programskih rešitev (Jurič, 2007). Na pristop SOA lahko gledamo tako z načrtovalskega kot arhitekturnega vidika. Idejni koncept SOA je oblikovanje programskih rešitev ali sistemov, ki imajo dobro načrtovane vmesnike skupaj s storitvami, ki so pozneje vključeni v poslovni proces. Z arhitekturnega vidika pa pristop SOA definira enostavne mehanizme za dostop do teh vmesnikov z namenom integracije poslovanja.

Dandanes obstajajo številne napačne interpretacije, kaj storitev in SOA sploh pomenita. Ker je to bistvenega pomena pri razumevanju celotne arhitekture SOA, si na kratko poglejmo, kako pridemo do besedne zveze storitveno usmerjena arhitektura.

Storitev

Storitev je predstavljena kot enota poslovne funkcionalnosti v organizaciji in je implementirana kot skupina vmesnikov, skupaj z njihovo izvedbo (Natis, 2007). Lahko rečemo tudi, da ima dvojno identiteto; kot program in kot poslovna funkcija. Storitve znotraj SOA vsebujejo poslovno-semantično predstavitev programske rešitve, iz katere storitev izvira. Po tem samo poslovno usmerjene storitve lahko upoštevamo kot gradnike pristopa SOA.

Storitev kot programska rešitev

Storitev v programski rešitvi je načrtovana tako, da do nje lahko dostopajo druge programske rešitve na osnovi vmesnikov, ki jih lahko znova uporabimo. Na sliki 1 je prikazana povezava med storitvijo, procesom in aktivnostjo. Storitev temelji na določenem poslovnem procesu znotraj organizacije. Vsak poslovni proces je sestavljen iz množice aktivnosti, ki se koordinirano izvajajo znotraj organizacijskega in tehničnega okolja ter v celoti (ne posamezno) dosegajo določen poslovni cilj (Weske, 2007).

Slika 1: Grafična predstavitev povezave med storitvijo, poslovnim procesom in aktivnostjo

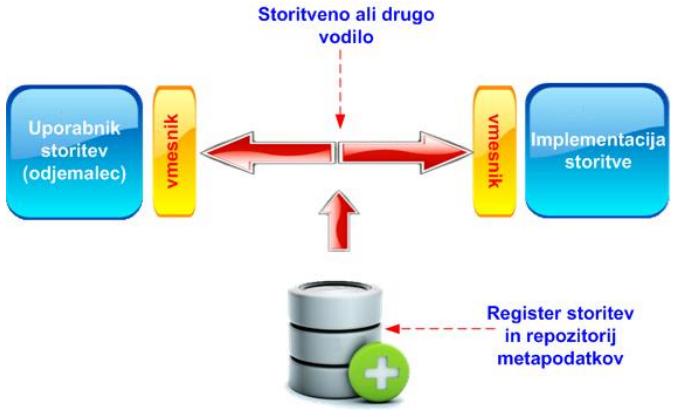
Zaradi tega ima storitev vlogo povezovalne točke med poslovnim in tehničnim načrtom poslovne programske rešitve. Sama semantika storitve je načrtovana z vidika poslovnega modela, njena izvedba pa je del programskega modela. Čeprav se lahko posamezne storitve

uporablja kot povezava točka-točka (gre za povezavo med natanko dvema sistemoma), so storitve tako načrtovane, da jih lahko po potrebi uporabljajo različni procesi, transakcije ali programske rešitve.

Storitveno usmerjena arhitektura

SOA je arhitekturni pristop k razvoju integracijskih rešitev, ki uporablja storitve. Kot je prikazano na sliki 2 so programske rešitve, ki temeljijo na pristopu SOA, sestavljene iz storitev, odjemalcev storitev (angl. *service client*) in po potrebi drugih programskih rešitev (Natis, 2007).

Slika 2: Struktura programske rešitve, ki temelji na pristopu SOA



Vir: Y. Natis, Applied SOA: Transforming Fundamental Principles Into Best Practices, 2007, str. 4

SOA je šibko sklopljena arhitektura (angl. *loose coupling*). Šibka sklopljenost se kaže predvsem v fleksibilni povezavi med storitvijo in njenim odjemalcem. Nov odjemalec lahko dostopa do natančno načrtovane storitve na diskreten način. To pomeni, da ne vpliva na izvajanje drugih odjemalcev. Slabo načrtovana storitev pa po drugi strani lahko vsebuje specifične, nedokumentirane podrobnosti ali pravila za odjemalca, ki onemogočajo klic storitve s strani drugih odjemalcev. Šibka sklopljenost je v arhitekturi SOA omogočena s pomočjo transportnega vmesnika in registra storitev.

Sama šibka sklopljenost pa ni dovolj za pristop SOA, ampak zgolj njena tehnična zahteva. V nadaljevanju so naštete nekatere lastnosti, ki jih ne moremo upoštevati kot funkcionalnosti pristopa SOA in nekatere, ki jih lahko.

Pristop SOA ni identificiran (Natis, 2007, str. 4):

- Z uporabo specifičnih protokolov, kot sta SOAP ali HTTP. Protokola lahko uporabljamo tudi za druge vrste komunikacije, kjer ni nobene sledi o pristopu SOA.
- Z uporabo programskih specifikacij, kot sta WSDL (angl. *Web Service Description Language*) ali BPEL (angl. *Business Process Execution Language*).

- Z uporabo modularnosti; na primer z uporabo EJB (angl. *Enterprise JavaBeans*) ali spletnih storitev (angl. *Web Services*). Modularne so lahko druge programske rešitve, ki podpirajo izvajanje procesov znotraj organizacije.
- Z uporabo orodij, kot so orodja ESB (angl. *Enterprise Service Bus*) ali orodja BPM (angl. *Business Process Management*). Orodja lahko uporabljamo tudi za programske rešitve, ki ne temeljijo na pristopu SOA.

Pristop SOA je identificiran (Natis, 2007, str. 4):

- Z uporabo ločenih, samostojno definiranih vmesnikov programskih komponent. Uporaba slednjih omogoča določeno stopnjo šibke sklopljenosti in dopušča možnost uporabe komponent (kot storitev) zunaj definiranega konteksta. To se izkaže kot glavna sposobnost arhitekture SOA.
- S preprosto granularnostjo poslovnih funkcionalnosti, predstavljenih z vmesnikom. Za pristop SOA ni ustrezna vsaka modularnost. Ko govorimo o modularnosti SOA, mislimo na modularnost poslovnih komponent, ki so uporabljene znotraj različnih programskih rešitev. Zaradi tega morajo biti storitve dovolj razdrobljene in morajo hkrati podpirati celoten poslovni proces.
- Z vidljivostjo posamezne definicije vmesnika in vstopno točko za uporabo storitve s strani drugih sistemov. Govorimo o samostojnih storitvah, ki izvajajo določen poslovni proces, njihov vmesnik je registriran in dokumentiran tako, da do njega lahko dostopajo druge zunanje programske rešitve.

Prihaja pa tudi do ideje, da je storitveno usmerjeni arhitekturni pristop vsebovan že znotraj tehnologije spletnih storitev. To je običajna, ampak napačna trditev, katere posledica so napake, ki jih organizacije naredijo pri vpeljavi SOA v svoje poslovanje. Govorimo o zmotnem prepričanju, da so prednosti in dodana vrednost, ki jih SOA prinaša, enostavno dosežene samo s konkretno investicijo v spletne storitve.

1.1 Smernice informatizacije

Ko govorimo o SOA kot arhitekturnem pristopu, s katerim želimo modernizirati in izboljšati celotno poslovanje organizacije, moramo upoštevati še nekatere druge koncepte. V večini organizacij koncepti, kot so integracija programskih rešitev, SOA, management podatkov in management procesov, delujejo neodvisno. Modernizacija in optimizacija ne vplivata samo na podatke in programske rešitve, ampak tudi na poslovne procese. Govorimo o prepletanju različnih pristopov, na osnovi katerih lahko pričakujemo pozitiven učinek na poslovanje organizacije in pozitivne rezultate vpeljave SOA.

1.1.1 Management podatkov

Management podatkov (angl. *Data Management*) je del iniciative, ki je bolj poznana kot management poslovnih informacij (angl. *Enterprise Information Management, EIM*). V večini primerov so organizacije sledile EIM in managementu podatkov zaradi tehničnih razlogov (kot je recimo izboljšanje kakovosti podatkov) in razlogov, ki niso tehnične narave (kot so na primer nadzor informacij, nadzor in pravilno dodeljevanje vlog pri managementu informacij). Dejstvo je, da lahko slab management podatkov hitro razširi nenatančne podatke v celotni organizaciji. Podatki, ki sploh niso upravljani, pa hitro postanejo zgodovina znotraj organizacije in bolj ko se ti podatki prenašajo znotraj organizacije, težje je dobiti neko smiselno rešitev za njihov management (Thompson, 2008).

Storitve SOA so najpogosteje implementirane na osnovi dejstva, da so podatki in logika programske rešitve pravilni in da je možno vse skupaj zajeti ter združiti v storitev. Takšna strategija samo še razširja slab management informacij. Za reševanje takšnih težav se uporablja postopek, imenovan integracija podatkov, ki je del pristopa managementa podatkov. Glavna naloga integracije podatkov je reševanje semantičnih neskladnosti in urejanje podatkov, tako z vidika semantike kot z vidika sintakse znotraj vseh podatkovnih struktur. Zaradi tega je za uspešno vpeljavo SOA prav tako potreben uspešen management podatkov. V nasprotnem primeru se bo organizacija znašla s še več razdrobljenimi, nekonsistentnimi in nepopolnimi informacijami, kot jih je imela pred vpeljavo.

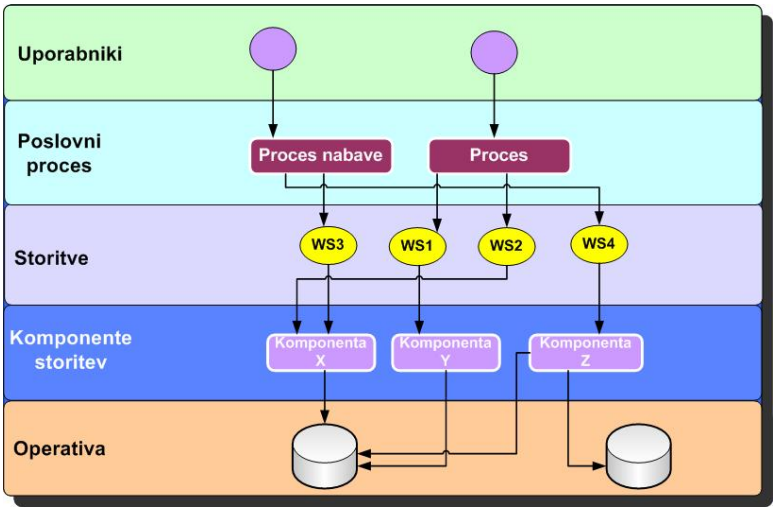
1.1.2 Management poslovnih procesov

Management poslovnih procesov (angl. *Business Process Management BPM*) je disciplina ali pristop, ki zahteva od organizacije, da opusti zanašanje na funkcijska področja znotraj organizacije in začne razmišljati procesno (Thompson, 2008). Od organizacij zahteva in jim omogoča celovit management vseh poslovnih procesov (od načrtovanja, spremljanja, vse do optimizacije poslovnih procesov) in njihovo stalno spreminjanje zaradi sprememb, ki jih narekuje poslovno okolje. Pristop vključuje in povezuje obstoječe ter nove metode, koncepte in orodja na tem področju. Usmerjen je v povezovanje poslovnih partnerjev in neposredno povezljivost njihovih poslovnih procesov (Seničar, 2006).

Čeprav sta SOA in BPM pristopa, ki sta lahko v organizacijo vpeljana ločeno, imata določene skupne cilje, ki so v največji meri povezani z agilnostjo organizacije. Agilnost organizacije lahko opredelimo kot zmožnost organizacije, da se hitro prilagaja spremembam poslovnega okolja. S pomočjo pristopa BPM pridemo do natančnejšega razumevanja poslovnih procesov in v končni fazi do spoznanja, katere procese preoblikovati v storitve SOA (Malinverno, 2007). Zaradi tega lahko rečemo, da združevanje projektov SOA in BPM prinese še večje prednosti kot pa ločeno vključevanje posameznega pristopa v organizacijo. Čeprav velja, da je lahko SOA uspešna tudi brez uporabe ustreznih orodij ali pristopov BPM, obstaja priporočilo, da je pri vpeljavi SOA smiselno vključiti tudi t. i. poslovni management (angl. *Business management*) in poslovne analitike. Samo z njihovo pomočjo lahko pridemo do ugotovitve,

kateri del poslovanja je treba informacijsko podpreti ali pa znova razviti kot storitev SOA. Ko govorimo o storitvah, nam BPM lahko pomaga tudi pri orkestraciji storitev. Kot je prikazano na sliki 3, lahko do posameznih komponent obstoječih sistemov dostopamo s pomočjo storitev, ki jih lahko skupaj povežemo na ravni poslovnega procesa. Storitve so ključ do ponovne uporabljivosti in hitrejšega razvoja, ki postajata za vsako organizacijo dva čedalje pomembnejša dejavnika. Z vidika modeliranja je BPM novi pristop za modeliranje, skrbništvo, konfiguracijo in vzdrževanje poslovnih procesov (Malinverno, Hill, 2007).

Slika 3: Povezava med posameznimi elementi plasti arhitekture SOA



Vir: A. Arsanjani, Service-oriented modeling and architecture, 2004

1.1.3 Integracija programskih rešitev

Pojem integracija programske rešitve lahko definiramo kot možnost povezovanja programskih rešitev, ki so bile načrtovane in razvite neodvisno druga od druge (Thompson, 2008). V zadnjih letih se je pogled na integracijo programskih rešitev s strani organizacij zelo spremenil. V devetdesetih letih je bila integracija zgolj neka posledična aktivnost, katere namen je bil povezati programske rešitve in sisteme, ki niso bili načrtovani za povezovanje z drugimi sistemi. Danes lahko rečemo, da ni samostojne rešitve, ki bi bila neodvisna od drugih sistemov, in da je integracija postala pomembna faza pri načrtovanju informacijskih sistemov. Svetovne raziskave kažejo, da bo več kot 2000 organizacij, med leti 2007 in 2012, vsaj podvojilo medsebojni promet (transakcije, dokumenti) (Biscotti, 2008). To postavlja integracijo kot jedro sestavljenih programskih rešitev in kot glavni razlog za pristope, kot sta SOA in BPM.

Na osnovi ugotovljenih dejstev lahko znotraj integracije programskih rešitev identificiramo tri ključne namene ali vidike integracije (Thompson, Schulte, 2008). Prvi vidik je zagotavljanje in ohranjanje konsistentnosti podatkov znotraj vseh programskih rešitev v organizaciji. Drugi vidik integracije programskih rešitev je podpora sestavljenim, večstopenjskim poslovnim

procesom. Tretji, najpogostejši vidik, pa je podpora sestavljenim programskim rešitvam, ki poleg osnovnih funkcionalnosti v ozadju izvajajo aktivnosti na drugih sistemih.

Ohranjanje konsistentnosti podatkov

Ohranjanje konsistentnosti podatkov pomeni, da so podatki vsebinsko pravilni in enaki v vseh sistemih, v katerih se nahajajo. Neodvisno načrtovani in implementirani sistemi pa večkrat vsebujejo vsebinsko iste podatke, ki se razlikujejo. Zaradi tega je glavna naloga ohranjanja konsistentnosti podatkov definiranje pravil in postopkov, ki bodo zagotavljali, da se vse spremembe nad istimi podatki odražajo tudi v drugih sistemih, kjer se ti podatki nahajajo. Dober primer, kjer je konsistentnost podatkov zelo pomembna, je primer podatkov o poslovnih partnerjih. Podatki o poslovnih partnerjih (naročniki, dobavitelji, kupci ...) so v večjih organizacijah shranjeni v različnih sistemih. Bodisi da so to sistemi CRM bodisi sistemi, ki podpirajo ključne poslovne funkcije znotraj organizacije. V takih primerih je treba zagotoviti, da ko se bo recimo v sistemu CRM spremenil naslov nekega poslovnega partnerja, se bo morala ta sprememba odraziti tudi v drugih sistemih. V nasprotnem primeru lahko rečemo, da so podatki o poslovnih partnerjih vsebinsko nekonsistentni (na enem mestu so lahko pravilni) in zaradi tega nezanesljivi.

Podpora sestavljenim, večstopenjskim poslovnim procesom

Podpora sestavljenim, večstopenjskim poslovnim procesom se uporablja, ko je poslovni proces prek aktivnosti povezan z drugimi sistemi ali programskimi rešitvami. Konec procesa zahteva bodisi konec aktivnosti s samodejno poizvedbo v zunanjem sistemu bodisi konec aktivnosti, ki zahtevajo ročni vnos podatkov na strani uporabnikov. Ker lahko taka interakcija z zunanjimi sistemi traja dlje časa, se lahko tudi izvajanje takega poslovnega procesa zavleče za več sekund, minut, ur ali pa dni. Primer večstopenjskega poslovnega procesa je proces nabave, ki je sestavljen iz več aktivnosti in pri katerem uspešen konec procesa zahteva izvajanje številnih postopkov, ki so najpogosteje podprti z različnimi programskimi rešitvami znotraj organizacije.

Programske rešitve ali komponente, ki nastopajo v večstopenjskem poslovnem procesu, so tehnično neodvisne med seboj in se lahko izvajajo ob različnih časih. Kljub temu pa so do neke mere programske rešitve logično odvisne, saj vsak korak ali aktivnost procesa temelji na predhodnem koraku, ki ga je izvedla predhodna programska rešitev. Uspešnost celotnega procesa pa zahteva uspešen konec zadnjega koraka.

Večstopenjski procesi so praviloma asinhroni, povezovanje z drugimi sistemi in programskimi rešitvami v procesu vedno poteka samo v eno smer. V primeru, ko je poslovni proces implementiran tako, da je povezovanje realizirano v obe smeri, kjer programska rešitev čaka na odgovor druge programske rešitve, pa ne govorimo več o večstopenjskem poslovnem procesu, ampak o sestavljeni programski rešitvi.

Podpora sestavljenim programskih rešitvam

Sestavljene programske rešitve so z vidika integracije najzahtevnejše za implementacijo. Gre za neke vrste sestavo programske opreme, ki implementira določeno poslovno funkcijo in komponente, ki so načrtovane in implementirane neodvisno. Uporabnik vidi sestavljeno programsko rešitev kot en sistem, v resnici pa so določene funkcionalnosti del drugega sistema, ki je uporabniku neznan.

V sestavljeni programski rešitvi so posamezne komponente deloma ali pa popolno sinhronizirane. To pomeni, da komponenta, ki je v določenem trenutku odjemalec, pošlje drugi komponenti sporočilo. Najpogosteje se v takih primerih odjemalec popolnoma ustavi in čaka na posredovan povratni odgovor.

Primer te vrste integracije je recimo uporaba spletnih certifikatov za registracijo avtomobila, kjer se pri registraciji v portal in izbiri spletnega certifikata najprej preveri, ali je spletni certifikat veljaven in ni v t. i. seznamu preklicanih spletnih certifikatov.

1.2 Arhitekturni pristop

Preden začnemo opisovati posamezne arhitekturne pristope, ki jih lahko najdemo v okolju IT, si najprej poglejmo, kakšen pomen sploh ima beseda arhitektura. Seveda v tem primeru govorimo o tehnološki arhitekturi ali arhitekturi programskih rešitev, ki se uporablja, vse odkar obstajajo avtomatizirane rešitve. Arhitekturo programske rešitve sestavlja množica načrtovalskih komponent in najboljših izkušenj, ki omogočajo učinkovito načrtovanje programske rešitve, postavitev in njen konstanten razvoj (Blechar, 2008).

S pojavom večnivojskih informacijskih rešitev se je močno razširil način, kako so lahko programske rešitve sestavljene in umeščene v končno delovno okolje (Erl, 2008). To je bil začetek arhitekture programskih rešitev. Za načrtovalce in razvijalce informacijskih sistemov je aplikacijska arhitektura tehnični načrt. Različne organizacije uporabljajo različne nivoje arhitekture programskih rešitev, ki se razlikujejo predvsem v fizični in logični predstavitvi tehničnega načrta. Značilno je tudi, da imajo organizacije v samem razvoju različne aplikacijske arhitekture. Posamezna arhitektura programskih rešitev se nanaša na posamezno okolje. Kot primer vzemimo organizacijo, katere razvoj temelji na platformi .NET in J2EE ter za posamezno platformo uporablja različno arhitekturo. Glavni namen arhitekture programskih rešitev je, da opredeljuje in predpisuje tako glavne značilnosti in zahteve, kot tudi dolgoročne strateške cilje. To je tudi razlog, da ko v organizaciji obstajajo različne arhitekture programskih rešitev, so te arhitekture programskih rešitev povezane in v skladu s poslovno arhitekturo (angl. *enterprise architecture*), ki predstavlja višji nivo vseh arhitektur znotraj organizacije.

Storitveno usmerjena arhitektura združuje arhitekturo programskih rešitev in poslovno arhitekturo. Prednosti, ki jih SOA ponuja, lahko dosežemo samo takrat, ko jih vključimo v

okolje, ki temelji na različnih informacijskih rešitvah. Govorimo o okolju, v katerem lahko enostavno upravičimo investicijo v izgradnjo ponovno uporabljivih storitev, ki temeljijo na različnih komunikacijskih platformah (Erl, 2008). To še ne pomeni, da mora celotna organizacija postati storitveno usmerjena, ampak lahko takšen pristop uporabimo samo na določenih področjih znotraj organizacije, ki so za tak način komunikacije in poslovanja tudi primerna. Čeprav arhitekturni pristop SOA ne predpisuje obsega, ki ga je treba z vidika arhitekture zajeti, ampak zahteva določeno stopnjo standardizacije in tudi zavedanje različnih ljudi znotraj organizacije o pomembnosti takšnega pristopa.

Vpeljava SOA ni primerna za vsako organizacijo. Okolje, v katerega vpeljujemo SOA, mora biti ustrezno zrelo in mora zagotavljati določeno stopnjo povezanosti posameznih sistemov znotraj organizacije. Poglejmo si, kateri so štirje glavni modeli, ki so potencialni kandidati za vpeljavo SOA (Natis, 2007).

Model 1: Večkanalne programske rešitve (angl. *Multichannel Applications*)
Posamezne programske rešitve znotraj organizacije so lahko načrtovane okrog enotne podatkovne baze ali okrog enotnega sistema, ki skrbi za poslovno logiko. Ko v organizaciji obstajajo različne programske rešitve, ki dostopajo do skupne podatkovne baze ali do skupnega sistema, ki skrbi za poslovno logiko, govorimo o večkanalnih programskih rešitvah. Najpogosteje so to rešitve, ki uporabljajo različne naprave, s katerimi lahko uporabniki dostopajo do posameznih funkcionalnosti rešitve. Čeprav so to različne naprave, je glavna značilnost večkanalnih programskih rešitev ta, da so njene funkcionalnosti dostopne različnim skupinam uporabnikov.

SOA je odlična rešitev za večkanalne programske rešitve. Ločevanje poslovne logike in neodvisnost odjemalcev omogočata posredovanje funkcionalnosti programske rešitve večjemu številu uporabnikov v zelo kratkem času. Seveda to zahteva, da so storitve ustrezno načrtovane in implementirane. Poleg tega so večkanalne programske rešitve zelo dober primer, kjer še posebno pride do izraza ponovna uporabljivost. Večkanalne programske rešitve so tudi najenostavnejši primer za vpeljavo SOA, ker v samem procesu ne vključujejo nobene integracije med programskimi rešitvami in so zaradi tega najboljši kandidat za pilotni projekt.

Model 2: Sestavljene programske rešitve (angl. *Composite Applications*)
Sestavljene programske rešitve so rešitve, katerih komponente ali moduli pripadajo različnim neodvisnim sistemom. Uporaba pristopa SOA je pri takšnih programskih rešitvah nekaj popolnoma običajnega, saj se lahko s takšnim pristopom komponente iz enega sistema uporabljajo tudi pozneje v drugih sistemih. Še več. Namesto, da bi pri vključevanju posamezne komponente v sistem potrebovali razvijalca, ki bi nam komponento vključil, je najboljši pristop ta, da se za posamezno funkcionalnost pripravijo programski vmesniki, ki jih lahko enostavno vključimo v posamezno programsko rešitev.

SOA predpostavlja razdelitev programskih rešitev v sestavljene komponente (storitve). Sestavljene programske rešitve so tako aplikacije, ki na višji ravni izkoriščajo posamezne funkcionalnosti komponent iz različnih sistemov ter tako posamezne vire vključujejo v nove transakcije in procese. Storitve, ki jih v sestavljenih programskih rešitvah uporabljamo, so lahko bodisi novo implementirane storitve bodisi deli nekaterih starih sistemov, ki smo jim opredelili vmesnike in storitve za njihovo zunanjo uporabo.

Model 3: Sestavljanje poslovnih procesov (angl. *Business Process Composition*)
Naslednji model SOA je model, pri katerem se storitve uporabljajo za sestavljanje poteka izvajanja postopka. Ko govorimo o poteku izvajanja, mislimo tako na klic storitve, kot tudi na vključevanje določenih akterjev (ljudi), ki vplivajo na potek izvajanja postopka. Pristop BPM definira orodja, ki omogočajo načrtovanje takih potekov izvajanja, ki temeljijo na pristopu SOA.

1.2.1 Koristi

Vpeljava SOA v organizacijo lahko prinese velike prednosti, ki so povezane predvsem z agilnostjo in s stroški razvoja novih ter spreminjanjem obstoječih rešitev (Natis, 2007). V nadaljevanju je predstavljenih nekaj ključnih področij, kjer so prednosti vpeljave SOA lahko razvidne.

Izboljšanje postopka razvoja in prilagajanja poslovnih procesov
Ena izmed glavnih prednosti vpeljave SOA v organizacijo je v primerjavi z drugimi pristopi prav zmanjševanje časa, napora in stroškov pri izvedbi ali spreminjanju obstoječega poslovnega procesa. Zato postane sam postopek implementacije novega ali spreminjanje obstoječega poslovnega procesa hitrejši in časovno manj potraten. Kljub temu sistemi, ki temeljijo na pristopu SOA, v začetnih fazah zahtevajo nekoliko več časa in truda kot klasični neporazdeljeni sistemi.

Izboljšana integracija
Ena od glavnih prednosti vpeljave SOA je zagotovo razvidna v realizaciji rešitev, ki temeljijo na integraciji storitev. Uporaba obstoječih delujočih rešitev in njihovo povezovanje je del integracije SOA. S takšnim pristopom podjetje pridobi na standardizaciji specifikacij storitev in podatkovnih struktur, ki so vključene v storitve.

Zmanjševanje stroškov
Pristop SOA teži k ponovni uporabljivosti sistemov, ki olajšujejo investicije v obstoječo infrastrukturo IT. Čeprav pristop SOA zahteva veliko investicijo v informacijsko infrastrukturo, se stroški te investicije povrnejo v kratkem času in se odražajo v občutno cenejšem vzdrževanju sistemov (Barner, Scholler, Malinverno, 2005).

Ponovna uporabljivost
Storitvena usmerjenost podpira načrtovanje storitev, ki so ponovno uporabljive. Načrtovanje storitev, ki podpirajo ponovno uporabljivost, omogoča širjenje in uporabo določene poslovne logike na različnih področjih poslovanja. Takšen pristop zahteva dodaten napor pri razvoju in standardizaciji načrtovanja takšnih storitev. Tako kot večina pristopov pristop SOA zahteva, da ljudje znotraj organizacije izkoriščajo ponovno uporabljivost tudi drugih vidikov – ne samo ponovno uporabljivost storitev, ampak tudi ponovno uporabljivost načrtovalskih vzorcev, tehnologij, platform in procesov (Barnes, 2005).

Uvajanje standardiziranega načina predstavitve podatkov
Na najosnovnejši ravni storitve SOA temeljijo na standardu XML (angl. *Extensible Markup Language, XML*), zato je vpeljava SOA odlična priložnost za izboljšanje predstavitve podatkov XML.

Agilnost organizacije
Agilnost organizacije lahko opredelimo kot zmožnost organizacije, da se hitro prilagaja spremembam poslovnega okolja. Agilnost se odraža prav v vseh vidikih poslovanja. Vsaka posamezna komponenta, naj bo to algoritem, programski paket, programska rešitev, platforma ali proces, vsebuje določeno mero agilnosti, ki je povezana s tem, kako je komponenta razvita in pozneje uporabljena. Kako bodo posamezne komponente realizirane in vzdrževane skozi čas ter znotraj določenega finančnega okvira, je definirano z agilnostjo organizacije.

Večji del storitvene usmerjenosti temelji na tem, da se bo to, kar danes izdelamo, skozi čas spreminjalo in razvijalo. Ena izmed glavnih prednosti dobro načrtovane rešitve SOA je zaščita organizacije pred vplivom, ki jih prinašajo spremembe. Ko prilagajanje na spremembe postane pravilo, potem kakovosti, kot sta ponovna uporabljivost in interoperabilnost, postanejo vsakdanjost.

Poleg tega lahko z abstrakcijo posamezne poslovne logike in tehnologije v specializirane plasti storitev s pomočjo pristopa SOA dosežemo šibko sklopljenost (angl. *loose coupling*) med dvema področjema poslovanja. Takšen način povezanosti omogoča posameznemu področju, da se razvija in odziva na spremembe neodvisno od drugih. Zmanjševanje potrebnega napora in stroškov se v sistemih, ki temeljijo na pristopu SOA, zmanjšujejo tudi z vidika odzivanja na tehnološke in poslovne spremembe.

1.2.2 Pogoji

Uspešnost vpeljave SOA v organizacijo je odvisna od številnih dejavnikov. Ti dejavniki so lahko tako tehnični kot tudi organizacijski. V nadaljevanju si poglejmo nekaj dejavnikov.

Postopna vpeljava SOA

Projekta vpeljave SOA v organizacijo se mora zavedati celotna organizacija. Kljub temu pa vpeljava SOA ne sme hkrati zajeti celotnega poslovanja, ampak je treba vpeljavo izvajati postopoma. Mnoge organizacije pri vpeljavi SOA ostanejo razočarane, ker ne upoštevajo vpliva, ki ga ima pristop SOA na razvoj programskih rešitev, vzdrževanje, integracijske pristope in na vpletenost SOA v management in strukturo organizacije, zato se tudi vse iniciative SOA izvajajo počasi in, kar je najpomembnejše, na rezultate vpeljave SOA je treba čakati dolgo časa. Najboljši pristop za preprečevanje takšnih upočasnitev je postopno izvajanje manjših projektov, ki lahko dajejo tako tehnično kot tudi poslovno merljive rezultate in merjenje uspešnosti projekta na osnovi pridobljenih rezultatov. Ti rezultati morajo biti razumljivi tako vodstvu informatike kot tudi vodstvu organizacije same.

Čeprav lahko takšen pristop zahteva počasnejšo vpeljavo SOA pristopa v organizacijo, zagotavlja, da je ta vpeljava manj tvegana in veliko bolj vzdržljiva.

Načrtovanje storitev

Implementacija posameznih storitev ima dolgoročen vpliv na informacijsko okolje v organizaciji. Storitve, ki so načrtovane brez premišljenega načrta, so lahko slabi kandidati za dolgoročne zahteve zaradi spreminjajočega se okolja. Zaradi tega morajo biti storitve premišljeno načrtovane, saj so kot take v dobro načrtovanem okolju SOA dolgoročna pridobitev.

Načrtovanje modela storitev spominja na načrtovanje modela podatkov. V obeh primerih je normalizacija komponent (relacije v sistemih za management relacijskih baz in vmesniki storitev v modelu SOA) pokazatelj kakovosti in zrelosti samega modela. Namen normalizacije je odstraniti redundantnost in doseči nek poslovni pomen razdelitve celote (to so programska rešitev in množice podatkov) v posamezne elemente (kot so storitve in tabele).

Management metapodatkov

Identificiranje in delitev metapodatkov z drugimi sistemi izboljšuje prilagodljivost in agilnost sistemov znotraj okolja IT, saj je enostavneje nadzorovati in spreminjati metapodatke kot razumeti, nadzorovati in spreminjati programirano poslovno logiko in programske module.

Z vidika SOA so metapodatki podatki o programskih rešitvah, ki izvajajo poslovni proces skozi posamezne aktivnosti in definirajo samo vedenje programske rešitve. Najpogosteje najdemo metapodatke razpršene v poslovnih programih v obliki spremenljivk ali drugih vhodnih parametrov. Zato je prvi korak pri vzpostavitvi nadzornega sistema programskih metapodatkov prav njihova identifikacija in priprava dokumentacije metapodatkov le-teh. V naslednjem koraku je treba metapodatke izločiti iz programske kode in jih shraniti v nek repozitorij, do katerega lahko dostopajo tudi drugi sistemi. Takšen repozitorij dviguje stopnjo agilnosti znotraj organizacije, saj je spreminjanje enotnih podatkov znotraj repozitorija hitrejše, cenejše in manj tvegano.

1.2.3 Nevarnosti

Kljub prednostim, ki jih ponuja pristop SOA, in pogojem, ki se jih je smiselno držati, obstajajo tudi določene nevarnosti/tveganja, ki jih je treba upoštevati in se jim je treba poskušati izogniti. Te pasti so povezane predvsem z dejstvom, da gre pri pristopu SOA za spreminjanje tehnologij in načina razmišljanja.

Težji management
Vsaka distribuirana programska rešitev, bodisi da temelji na SOA pristopu ali ne, je težja za načrtovanje, programiranje, testiranje in management (Natis, 2007). Po drugi strani pa je nedistribuirana programska rešitev lahko zelo enostavna. Vzrok za to je povezan predvsem z različnimi komponentami, ki takšno programsko rešitev sestavljajo, in z izzivom, da omenjene komponente prek omrežja komunicirajo med seboj.

Oddelek za informatiko v organizaciji ne zmore sam upravljati celotnega postopka, še posebno ne, če so posamezne komponente znotraj različnih poslovnih funkcij ali področij znotraj organizacije. V takih situacijah prihaja tudi do vprašanja zanesljivosti in kakovosti podatkov.

Neupoštevanje osnov arhitekture XML
»V SOA svetu se vse začne s spletnimi storitvami.« (Erl, 2007) Ta trditev se je znotraj organizacij uveljavila kot nekakšno pravilo. Resnica pa je, da ni vedno popolnoma pravilna. Dejstvo je, da se v svetu SOA vse začne z XML (Erl, 2007). To je standard, iz katerega so se pozneje razvili dopolnilni standardi strukture predstavitve podatkov. Ti standardi so poskrbeli za razvoj specifikacij spletnih storitev, na katerih zdaj temelji pristop SOA.

Veliko pozornosti se posveča temu, kako se podatki prenašajo med posameznimi storitvami. Premalo pozornosti pa se posveča načinu, kako so podatki strukturirani in preverjeni (v nadaljevanju validirani) v ozadju same storitve. Standardizacija načina, kako je lahko standard XML uporabljen za predstavitev, validacijo in procesiranje posameznih podatkov, ki se izmenjujejo med storitvami, je temelj robustnega in optimiziranega pristopa SOA.

Nerazumevanje performančnih zahtev SOA
Če začnemo postopoma in z majhnimi koraki, potem je razvoj rešitev, ki temeljijo na pristopu SOA, lažji in tudi same rešitve so take, kot smo jih pričakovali. Takoj ko obseg naraste in dodamo nove funkcionalnosti, se število sporočil, ki jih pošiljamo, občutno poveča. To je trenutek, v katerem se lahko začnejo pojavljati performančne težave sistema. Pristop SOA temelji na spletnih storitvah, spletne storitve pa temeljijo na predstavitvi podatkov XML, pri čemer je lahko samo procesiranje podatkov XML performančni izziv.

Performanse so tudi eden od razlogov, zakaj se pri razvoju spletnih storitev poudarja uporabo vmesnikov storitev in asinhronega prenosa podatkov. S pomočjo takih in drugih načrtovalskih prijemov se lažje izognemo potencialnim procesorskim ozkim grlom.

1.3 Delitev SOA znotraj organizacije

V prejšnjih razdelkih je bilo predstavljeno, kaj pristop SOA pomeni za organizacijo in katere prednosti prinaša. Ker pristop SOA pomeni večjo fleksibilnost in hitrejši razvoj novih rešitev, se čedalje pogosteje pojavlja potreba po vpeljavi SOA znotraj celotne organizacije, zato organizacije najpogosteje načrtujejo celotno vpeljavo na osnovi centralne infrastrukture SOA. To pomeni enotno storitveno vodilo (angl. *Enterprise Service Bus ESB*) z enotnim repozitorijem storitev. Centralna infrastruktura, in z njo povezan centralni management, olajšuje vpeljavo SOA okrog posameznih poslovnih področij in enot, omogoča tudi hitrejši pretok znanja in potrebnih ljudi. Kljub temu je iz izkušenj razvidno, da je izvedba takšne infrastrukture zelo zapletena in včasih tudi nemogoča tako zaradi tehničnih kot tudi organizacijskih razlogov.

V povezavi s slednjo problematiko se je razvil model delitve SOA po posameznih poslovnih področjih znotraj organizacije (angl. *Federated SOA*). Združen model SOA temelji na metodi »deli in vladaj«, pri čemer se organizacija razdeli na ločene, delno povezane domene SOA (Pezzini, 2008). Domena SOA je definirana kot zaključena skupina elementov, v katero spadajo programske rešitve, poslovni procesi in storitve ter je enotno upravljana tako z organizacijskega kot tudi tehničnega vidika. Lahko rečemo, da posamezna domena temelji na lastni infrastrukturi SOA in na lastnem samostojnem managementu.

V združenem modelu SOA so posamezne domene prek ustrezne infrastrukture povezane med seboj prav z namenom izmenjavanja storitev na osnovi definiranih tehničnih in semantičnih standardov, procesov managementa in drugih organizacijskih predpisov. Iz tega sledi, da je združen model SOA množica semantično integriranih domen SOA (Pezzini, 2008).

Razumljivo je, da so prve težave, s katerimi se srečamo pri postavitvi t. i. združenega modela SOA, povezane prav s tehničnimi zmožnostmi izmenjavanja storitev med posameznimi domenami SOA. To pa ni edini problem in tudi ni nujno, da ga je najtežje odpraviti. Združen model SOA je proces, ki odpira naslednje tri dimenzije (Schulte, 2008, str. 4):

- **interoperabilnost:** Interoperabilnost opredeljuje dva vidika. Prvi vidik je tehnični in je predvsem povezan s postavitvijo ustrezne tehnične infrastrukture, sposobne komunicirati prek različnih storitev z zadostno mero varnosti in kakovosti. To še posebno velja, ko so storitve znotraj različnih domen, ki uporabljajo različna storitvena vodila in različne repozitorije storitev. Drugi vidik pa je semantični in zahteva definiranje skupnih standardnih poslovnih objektov za entitete, ki se najpogosteje uporabljajo (kot so na primer stranka, produkt, plačilo …). Definiranje

standardnih poslovnih objektov je ključnega pomena pri izmenjavi podatkov med različnimi domenami in ponovni uporabljivosti storitev. Semantična interoperabilnost je po navadi težje dosegljiva kot tehnična.

- **management:** V združenem modelu SOA temelji management na definiranju procesov med posameznimi domenami. Ti procesi vključujejo: omogočanje programskih rešitev med domenami in management življenjskega cikla storitve, definiranje jasne politike odgovornosti, omogočanje uporabe storitev med posameznimi domenami s pomočjo registra storitev in drugih orodij.
- **organiziranje:** Podobno kot management je pri organiziranju poudarek na omogočanju sodelovanja posameznih domen skupaj, čeprav so lahko določeni poslovni procesi znotraj posamezne domene različni in nekonsistentni. Čeprav se domene znotraj strukture organizacije spontano pojavljajo, je definiranje njihove obsežnosti ključnega pomena. Če so domene majhne, se zgodi, da se njihovo število hitro poveča, in s tem dosežemo, da postane njihovo združevanje zapleteno, široko zasnovane domene pa postanejo prezapletene za samo implementacijo in zapletene za management.

Zgoraj opisani vidiki so pomembni za vzpostavitev takšnega modela. Čeprav svetovne raziskave kažejo, da bo do leta 2013 model postal ustaljena praksa znotraj večjih organizacij (Sholler, 2008), se organizacije že dolgo časa trudijo za uspešno postavitev tega modela. Pravih rezultatov in najboljših praks pa še ni. Razlogi so povezani tako s pomanjkanjem pravih tehničnih standardov kot tudi z odporom pred spremembami znotraj organizacije. Kljub temu bodo organizacije čedalje bolj prisiljene v razmišljanje o takšnem modelu, zlasti zaradi povezovanja s svojimi poslovnimi partnerji (kupci, dobavitelji, drugi ponudniki storitev).

1.4 Management SOA

Ko govorimo o managementu znotraj organizacije, imamo v mislih načela in pravila, ki definirajo, kako naj se organizacija vede. Z vidika SOA pa je management kombinacija politik, procesov in metapodatkov (podatki, ki definirajo izvor komponent, lastnike komponent in kdo lahko posamezne komponente spreminja). Glavni namen managementa SOA je izogibanje podvajanju spletnih storitev in omogočanju ponovne uporabljivosti storitev (Malinverno, 2006). Zaradi tega razloga management SOA ni možnost, ampak nuja.

Glede na to, da se čedalje več svetovnih organizacij odloča za pristop SOA, se soočamo z dvema pomembnima izzivoma (Malinverno, 2006, str. 1):

- Razširitev obstoječih pilotnih infrastruktur zlasti za podporo pristopa SOA. Obstoječe infrastrukture v glavnem segajo do podpore spletnih storitev, ne zagotavljajo pa dovolj visoke stopnje skalabilnosti, zaupljivosti, varnosti in ostalih performančnih lastnosti.

- Rast pristopa SOA na osnovi ponovne uporabljivosti storitev in izogibanju podvajanja spletnih storitev. To pa lahko dosežemo samo s skrbno načrtovanimi postopki managementa.

Seveda se, ko govorimo o managementu SOA, navezujemo predvsem na drugi izziv. Ta je še posebno pomemben v večjih organizacijah, kjer vpeljava SOA ni toliko odvisna od tehnologije, ki se uporablja za definiranje storitev, ampak od postopkov in organizacijske strukture, ki je povezana z izvajanjem in managementom.

Tri glavne komponente managementa SOA

Svetovne raziskave kažejo, da gre pri managementu pravzaprav za pravice glede sprejemanja odločitev in pravice za vplivanje na te odločitve (Malinverno, 2006). Učinkovit management, gledano z vidika celotnega informacijskega okolja, temelji na kombinaciji odločitev za posamezna področja, ki jih sprejema zaključena skupina ljudi na osnovi definiranih mehanizmov. Dogovori glede sprejemanja odločitev so v glavnem razdeljeni na tri komponente:

- **katere odločitve je treba sprejeti:** Odločitve, povezane s posameznimi področji znotraj okolja IT, in kako strukturirati storitve za dostop do posameznih področij.
- **kdo ima pravico te odločitve sprejeti:** Pravice so porazdeljene na različne načine v posameznih enotah znotraj organizacije.
- **kako so odločitve oblikovane in odrejene:** Učinkovit management temelji na kombinaciji mehanizmov oblikovanja odločitev. V glavnem so ti mehanizmi odvisni od velikosti organizacije in kulture.

Management SOA je specifičen pogled na management IT. Management IT definira pooblastila in odgovornosti sprejemanja odločitev znotraj celotnega okolja IT. Seveda tudi management IT predpisuje neke vrste ogrodje, znotraj katerega je poskrbljeno, da so odločitve v skladu s strateškimi cilji poslovanja. Znotraj managementa IT pa management SOA identificira pooblastila sprejemanja odločitev glede informatizacije poslovnih procesov, na podlagi pristopa SOA. Poleg tega pa management SOA zagotavlja način definiranja, načrtovanja, dostopanja, izvajanja in vzdrževanja ponovno uporabljivih storitev. Natančneje povedano, management SOA z vidika spletnih storitev podpira naslednje tri glavne funkcionalnosti (Thompson & Pezzini, 2009, str. 4):

- **management življenjskega cikla**: Funkcionalnost, ki jo najpogosteje najdemo znotraj repozitorija, ki skrbi za razvoj, verzioniranje in postavitev gradnikov, uporabljenih v implementaciji rešitev SOA. Informacije posreduje ustreznim vlogam (arhitektom, analitikom, razvijalcem) tako, da so lahko posamezni gradniki ponovno uporabljeni. Poleg tega skrbi za sledenje in analizo vpliva.
- **management pravil**: Poudarek je na tehnologiji, ki omogoča razvoj, postavitev in management pravil, ki razširjajo določene vidike managementa SOA, kot so nadzor nad dostopom, management identitete in performans ter dogovori o ravni storitve. Če povzamemo, definira osnove za management poslovnih storitev.

- **management metapodatkov SOA**: Funkcionalnost pokriva management metapodatkov s pomočjo kataloga storitev in s tem povezanih artefaktov, kot so datoteke WSDL (angl. *Web Service Description Language WSDL*), enotna vstopna točka za katalogiziranje in objavljanje informacij o storitvah in programskih rešitvah, sledenje statističnih informacij in performančnih zmožnosti. Govorimo o neke vrste registru, ki skrbi tako za management življenjskega cikla kot tudi za management pravil.

Vpeljava SOA brez ustreznega managementa najpogosteje ne zagotavlja ustrezne donosnosti naložbe (angl. *Return On Investment ROI*) in v večini takih primerov pripelje projekt vpeljave SOA do neuspeha (Thompson & Pezzini, 2009). Obseg managementa SOA znotraj posamezne vpeljave je odvisen od velikosti organizacije, kulturnih in geografskih lastnosti in obstoječega managementa IT (Malinverno, 2006).

1.5 Storitveno vodilo

Pristop SOA temelji na pošiljanju sporočil med različnimi informacijskimi sistemi ali programskimi rešitvami, ki podpirajo določene poslovne funkcije znotraj organizacije. Ker je takšnih sporočil veliko, je pri tem najpomembnejše, da mora biti pošiljanje hitro in predvsem zanesljivo. V nasprotnem primeru lahko pride do izgube informacij in posledično ne moremo več govoriti o integraciji.

Večina organizacij (skoraj tretjina), ki so vpeljale pristop SOA, uporablja tudi storitveno vodilo ali ESB (Sholler, 2008). ESB je tako pomemben del celotnega pristopa, da nekateri celo mislijo, da pristop SOA ne more obstajati brez ESB, in spet drugi, če ima organizacije ESB, potem ima tudi SOA. Seveda nobena od teh trditev ni popolna. ESB ni obvezen gradnik znotraj pristopa SOA, vendar je vseeno treba zagotoviti takšen gradnik, ki bo imel vlogo, ki jo ima ESB znotraj pristopa SOA. Brez ESB je treba večino storitev in konceptov implementirati na modulih programskih rešitev, ki so vključene v SOA kot arhitekturo. V nasprotnem primeru pa lahko večino teh storitev (naslavljanje, mediacija ...) prepustimo ESB in drugim gradnikom, ki so prisotni znotraj ESB (Schulte, 2007).

ESB lahko razumemo kot črno škatlo. Ni nam treba vedeti, kako deluje, bistveno je, da vemo, česa je ESB kot gradnik pristopa SOA zmožen. ESB vsebuje specifične storitve, za katere nam znotraj sistemov, ki jih povezujemo, ni treba skrbeti, ampak jih lahko enostavno uporabimo znotraj ESB. Storitve so naslednje:

- **sporočilne storitve:** Podpirajo številne tipe sporočil, usmerjanje glede na vsebino sporočila in zagotavljajo dostavo posameznih sporočil ciljnemu sistemu.
- **storitve managementa:** Skrbijo za spremljanje in nadzorovanje performansa sporočil. Omogočajo implementacijo prioritet sporočil in podporo za poslovna pravila.

- **storitve vmesnikov:** Omogočajo validacijo sporočil na osnovi definirane sheme. Podpirajo spletne storitve in opredeljujejo vmesnike (adapterje) za nekatere vire, ki niso spletne storitve.
- **storitve mediacije:** Preoblikujejo sporočila v različne formate, ki jih uporabljajo izvori in ponori sporočil.
- **storitve metapodatkov:** V povezavi z mediacijo lahko te storitve (tudi sporočila) preoblikujejo v različne formate na osnovi metapodatkov, ki se hranijo v posebnih registrih.
- **varnostne storitve:** Skrbijo za šifriranje sporočil tam, kjer je to potrebno, in vsebujejo standarden varnostni model avtorizacije in avtentikacije.

Treba se je zavedati, da niso vsi sistemi ESB enaki, vendar so vseeno zgoraj naštete storitve tiste, ki jih najdemo v večini danes največkrat uporabljenih sistemov ESB.

2. METODOLOŠKI OKVIR

2.1 Kaj je metodološki okvir

V prejšnjih poglavjih so bile natančno predstavljene glavne značilnosti pristopa, ki temelji na storitveno usmerjeni arhitekturi. Vse te lastnosti so ključnega pomena pri vpeljavi SOA v podjetje in vsak, ki se z vpeljavo sooča, mora te lastnosti dobro poznati in jih med drugim tudi upoštevati v sami fazi vpeljave. Kljub temu samo upoštevanje zgoraj naštetih lastnosti ni dovolj za uspešno vpeljavo SOA v organizacijo. Potrebni so določeni postopki, koraki in pravila, s katerimi bomo sam postopek vpeljave izvajali in lažje ter hitreje prišli do želenih rezultatov. Potreben je torej metodološki okvir, s katerim bomo zajeli tako posamezne korake vpeljave, kot tudi potrebne diagramske tehnike.

Preden začnemo predstavljati metodološki okvir, si poglejmo, kaj metodološki okvir sploh je. Ko govorimo o metodologiji nasploh, imamo v mislih različne koncepte, kot so teorije, ideje, študije, najboljše izkušnje in različne metode, s pomočjo katerih lahko določeno disciplino reguliramo in upravljamo.

Številni metodološki okviri so se razvijali skozi čas, vsak s svojimi prednostmi in tudi slabostmi. Dejstvo je, da posamezen metodološki okvir ni primeren za vse tipe projektov, saj je treba poleg tehničnega vidika upoštevati tudi organizacijski in projektni vidik, zato tudi projekti, ki temeljijo na vpeljavi SOA v organizacijo, potrebujejo svoj metodološki okvir. Enotna Metodologija Razvoja Informacijskih Sistemov ali krajše EMRIS (Krisper, 2000) je recimo primer metodološkega okvira. Metodologija med drugim predpisuje različne diagramske tehnike in postopke za strukturni razvoj informacijskih sistemov in tudi za objektni razvoj, ki se v določenih pogledih malo razlikuje. Dejstvo pa je, da vpeljava SOA v organizacijo zahteva bistveno drugačen pristop kot klasični strukturni ali objektni razvoj informacijskih sistemov. Seveda obstajajo določeni prijemi, kot so diagramske tehnike in faze razvoja iz omenjene metodologije, s katerimi si lahko pomagamo tudi pri vpeljavi SOA, vendar se razlika pokaže v podrobnostih, ko posamezno fazo razvoja tudi izvajamo. Primer diagramske tehnike, ki jo EMRIS predpisuje v fazi analize, je diagram aktivnosti (angl. *activity diagram*). Z diagramsko tehniko res lahko definiramo posamezne aktivnosti, ki jih določa poslovni proces. Vendar to pri pristopu SOA ni dovolj, ker je v fazi analize treba posvetiti veliko pozornosti tudi sistemom, ki so vključeni v sam proces. Največja razlika z omenjeno metodologijo se pokaže prav pri organizacijskem vidiku, saj je pri vpeljavi SOA zaradi same narave projekta treba vložiti veliko več truda v samo sodelovanje, vodenje in definiranje ustreznih nalog v projektu.

Podobna metodologija je metodologija RUP (angl. R*ational Unified Process*), ki omogoča definiranje šibko sklopljenih storitev, vendar je metodologija RUP bolj projektno usmerjena

(Matsumura, 2006). Pri vpeljavi SOA v organizacijo ne moremo govoriti samo o projektu, ker vpeljava zajema celotno organizacijo. Poleg tega metodologija ne opredeljuje postopkov za identifikacijo posameznih storitev.

2.2 Namen in prednosti metodološkega okvira

Če smo zdaj ugotovili, da ni metodološkega okvira, ki bi ga lahko neposredno uporabili pri vpeljavi SOA, si poglejmo še razloge, zakaj metodološki okvir sploh potrebujemo, katere so tiste glavne prednosti metodološkega okvira in zakaj nam lahko metodološki okvir olajša delo pri vpeljavi SOA, skrajša čas same vpeljave in zmanjša stopnjo tveganja.

Namen metodološkega okvira vpeljave SOA ni izdelati splošnega primera vpeljave, na katerem bi lahko organizacije temeljile in ga uporabljale. Izdelava takšnega primera enostavno ni mogoča, saj je SOA pristop in ne rešitev (Bradley, Schulte, Sholler, Malinverno, 2008). Razloge je treba iskati drugod.

Prvi razlog je ta, da je metodološki okvir osnova za pripravo projektnega plana in z njim povezane časovne opredelitve projekta. Projekti, ki temeljijo na pristopu SOA, so v večini primerov projekti na daljši rok in zato lahko tudi posamezne faze trajajo dlje časa. Brez natančne opredelitve vsebine posamezne faze in izdelkov, ki so za to fazo pomembni, je zelo težko opredeliti časovne okvire tako posamezne faze kot tudi projekta v celoti. Zaradi tega je ključna lastnost metodološkega okvira ta, da je razdeljen na faze, ki imajo poleg natančno opredeljenega namena tudi natančno opredeljene izdelke, ki v sami fazi nastanejo. Ko govorimo o izdelkih, imamo v mislih različne dokumente, diagrame in sheme, ki temeljijo tako na novih kot tudi na že poznanih diagramskih tehnikah. Pri pripravi projektnega plana pa je z vidika organizacije pomembno tudi načrtovanje različnih skupin ljudi, ki so ključne v posameznih korakih vpeljave. Tako so na primer v začetnih fazah bolj pomembne skupine analitikov, v zaključnih fazah pa skupine razvijalcev, ki rešitev dejansko implementirajo. To pomeni, da mora metodološki okvir poleg natančno definiranih nalog in izdelkov opredeliti tudi skupine ljudi, ki imajo v posamezni fazi ključno vlogo. S takšno opredelitvijo lahko tudi znotraj samega projekta vpeljave lažje načrtujemo in razporejamo vire.

Skozi celoten proces vpeljave SOA govorimo o poslovnem procesu, ki ga želimo informatizirati. Ker je poslovni proces glavni element vpeljave SOA, mora biti tudi metodološki okvir tako strukturiran, da vsebuje vse ključne faze, ki se nanašajo na poslovni proces. Znotraj posameznih faz pa morajo biti definirane tako diagramske tehnike kot druga orodja za izvedbo faze.

Življenjski cikel poslovnega procesa je sestavljen iz faz, ki so predstavljene na sliki 4 (Weske, 2007).

Slika 4: Življenjski cikel poslovnega procesa

Vir: M. Weske, Business Process Management, 2007, str. 12

Faze so predstavljene v ciklični strukturi, ki prikazuje tudi medsebojno odvisnost posameznih faz. Ta odvisnost pa ne narekuje natančnih časovnih okvirjev, v katerih mora biti določena faza izvedena. Načrtovanje in razvojne aktivnosti se lahko na primer pojavljajo tudi v vseh fazah.

Modeliranje in analiza

Življenjski cikel poslovnega procesa se začne s fazo modeliranja in analize, v kateri se izvaja pregledovanje podrobnosti poslovnega procesa tako z organizacijskega kot tudi s tehničnega vidika. Na osnovi podrobnosti nastane model poslovnega procesa. Model poslovnega procesa uporabljamo za lažje razumevanje same vsebine in namena poslovnega procesa med različnimi vlogami ljudi znotraj organizacije. Ključni proces v tej fazi je uporaba tehnike modeliranja poslovnih procesov, kjer se s pomočjo ustrezne notacije identificira in prikaže aktivnosti poslovnega procesa skupaj s pripadajočim zaporedjem izvajanja posameznih aktivnosti.

Za modeliranje poslovnih procesov se večinoma uporabljata dve notaciji. Prva je notacija UML (angl. *Unified Modeling Language*). Znotraj notacije UML obstaja t. i. diagram aktivnosti, znotraj katerega je poslovni proces modeliran kot zaporedje aktivnosti (White, 2007). Slabost diagramske tehnike je povezana z možnostjo uporabe v poznejših fazah življenjskega cikla. Med arhitekti poslovnih procesov in analitiki so najbolj razširjena orodja, ki temeljijo na notaciji BPMN (angl. *Business Process Modeling Notation*) (Rosser, 2008). Glavna prednost notacije BPMN je uporaba notacije, ki je zelo poznana vsem poslovnim uporabnikom, bodisi poslovnim analitikom, ki pripravijo začetni osnutek, bodisi tehničnim razvijalcem, ki so odgovorni za izvedbo poslovnega procesa. Razumljiva je tudi poslovnim ljudem, ki proces analizirajo in upravljajo (Weske, 2007). Njena pomembna prednost je povezana tudi z možnostjo enoumne preslikave poslovnega modela v izvajalno različico (Rozman, 2006). Podrobnosti notacije BPMN niso predmet naloge, zato jih ne bom podrobneje opisoval.

Konfiguracija

Ko je enkrat poslovni proces modeliran in analiziran, lahko začnemo z informatizacijo poslovnega procesa. Obstajajo različni načini, kako implementirati poslovni proces. Proces je lahko implementiran kot množica pravil in procedur, s katerimi se morajo zaposleni v organizaciji strinjati in jih upoštevati. V takem primeru ne potrebujemo nobenega dodatnega sistema za management poslovnih procesov. Najpogosteje pa implementacija poslovnega procesa temelji na prehodu iz poslovnega modela (notacija BPMN) v model BPEL (notacija BPEL). BPEL je programski jezik, ki temelji na strukturi XML in omogoča standarden procesno usmerjen način integracije različnih sistemov (Gaur, Zirn, 2006).

Pomemben gradnik, ki ga mora metodološki okvir vpeljave SOA upoštevati in ki igra poglavitno vlogo pri povezovanju spletnih storitev s procesom BPEL in orkestracijo nasploh, je register spletnih storitev (angl. *Universal Description, Discovery and Integration Registry*) ali na kratko register UDDI. V fazi izvedbe sem na realnih projektih prišel do ugotovitve, da je register UDDI pomemben zlasti zaradi dveh razlogov:

- omogoča management spletnih storitev (iskanje, analiziranje) in vzdrževanje verzij posameznih storitev,
- omogoča enostavno orkestracijo storitev v posameznih fazah razvoja (postopek RTP).

Register UDDI se uporablja kot register vseh spletnih storitev znotraj organizacije. S pomočjo registra lahko enostavneje poiščemo in najdemo določeno spletno storitev, ki bi lahko bila potencialni kandidat za novi ali obstoječi poslovni proces, strukturo storitve podrobneje analiziramo in če je ustrezna, lahko storitev tudi vključimo v poslovni proces. S pomočjo registra UDDI skrbimo za ponovno uporabljivost storitev. Poleg enostavnega iskanja in managementa spletnih storitev ima register UDDI pomembno vlogo pri vzdrževanju različic posameznih storitev.

Postavitev

Ko govorimo o postavitvi poslovnega procesa, imamo v mislih predvsem inicializacijo in izvajanje posameznih instanc (primerkov) poslovnega procesa. Sistem za management poslovnih procesov neprestano spremlja izvajanje posameznih instanc, ki jih definira poslovni model. Za spremljanje izvajanja se uporabljajo specifična grafična orodja, ki omogočajo grafično prikazovanje statusov posameznih instanc, posamezne vrednosti podatkov znotraj opredeljenih aktivnosti, uspešno ter neuspešno zaključenih instanc in podobne statistične preglede.

Med samim izvajanjem instanc poslovnih procesov se pridobivajo številni podatki, ki se hranijo v posebnih dnevniških zapisih in omogočajo beleženje izvajanja določenih poslovnih procesov. Ti podatki so osnova za naslednjo fazo.

Spremljanje in vrednotenje
V fazi spremljanja in vrednotenja lahko na osnovi podatkov, pridobljenih v fazi postavitve, izvajamo različne analize in na njihovi osnovi poslovni proces tudi izboljšamo. V določenem poslovnem procesu lahko na primer ugotovimo, da se posamezna aktivnost zaradi določenega razloga predolgo izvaja. To je informacija, ki lahko narekuje zamenjavo določene aktivnosti ali izvedbo ukrepa, ki bo izboljšal izvajanje celotnega poslovnega procesa. Večina današnjih orodij za modeliranje poslovnih procesov že vsebuje funkcionalnost, s pomočjo katere lahko rezultate izvajanja instanc poslovnih procesov uporabimo na obstoječem modelu poslovnega procesa in tako poskušamo sam proces iz različnih vidikov tudi optimizirati.

Skrbništvo nad izdelki
V celotnem življenjskem ciklu je treba upoštevati tudi številne izdelke, diagrame in modele z različnimi stopnjami abstrakcije, ki nastanejo v posameznih fazah. Strukturirano mesto hranjenja (repozitorij) in učinkovit način iskanja posameznih izdelkov, ki se navezujejo tako na poslovni proces kot na posamezne instance procesov, ima lahko strateški pomen. Še zlasti je to pomembno v večjih organizacijah s številnimi poslovnimi procesi.

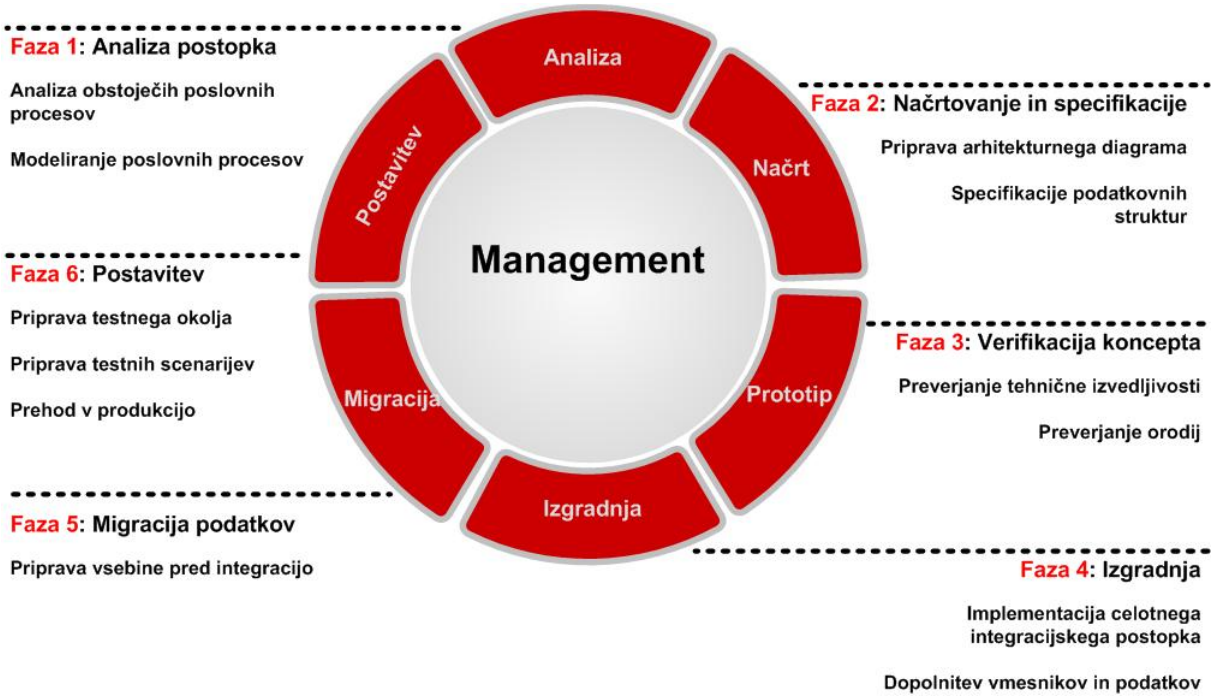
Gledano z vidika obstoječih metodologij ni metodologije ali metodološkega okvira, s katerim bi si lahko pri vpeljavi SOA v celoti pomagali. Razlog se skriva predvsem v tem, da pri vpeljavi SOA ne gre za klasičen razvoj novih rešitev, ampak predvsem za povezovanje tako obstoječih sistemov kot tudi ljudi in poslovnih procesov znotraj organizacije.

Na osnovi izkušenj, pridobljenih na realnih projektih, in uporabe različnih diagramskih tehnik je nastal metodološki okvir, struktura katerega je predstavljena v poglavju 2.3, podrobnejši opis posameznih faz pa v poglavju 3.

2.3 Struktura metodološkega okvira

Kot je razvidno iz slike 5, je metodološki okvir sestavljen iz šestih osnovnih faz. Poleg teh faz pa metodološki okvir opredeljuje še uvodno fazo, ki na sliki ni predstavljena, ker fazo izvajamo po želji in samo pred informatizacijo prvega poslovnega procesa.

Slika 5: Faze metodološkega okvira



Analiza postopka

Prva faza integracijskega projekta je analiza postopka. V poglavju 1 je bila predstavljena povezava med storitvijo, poslovnim procesom in aktivnostjo. Kot je razvidno na sliki 6, je v metodološkem okviru zaradi lažje implementacije in razumevanja poslovni proces dodatno razdeljen na integracijske postopke. Vsak posamezen postopek pa naprej na aktivnosti, pri čemer lahko dva integracijska postopka uporabljata isto aktivnost. Glavna naloga faze analize je modeliranje enega ali več poslovnih procesov oziroma pripadajočih integracijskih postopkov.

Slika 6: Grafični prikaz povezave storitve, poslovnega procesa, integracijskega postopka in aktivnosti

Načrtovanje in specifikacije
Glavni namen faze načrtovanja je podrobno določanje elementov (vmesniki in spletne storitve), ki so del integracijskega postopka, in kar je najpomembnejše, določanje podatkovnih struktur ali sporočil, ki se prek poslovnega vodila prenašajo med sistemi.

Verifikacija koncepta
Verifikacija koncepta je faza, ki je ne izvajamo pri vsakem integracijskem postopku. Glavni namen je izdelava postopka, s katerim bodisi ugotovimo ustreznost vpeljave SOA bodisi identificiramo morebitne težave pri izvedbi integracijskega postopka in druge ključne dejavnike uspeha.

Izgradnja
Sledi faza izgradnje, v kateri implementiramo tako določene elemente kot tudi poslovni proces, na katerem temelji integracijski postopek.

Migracija podatkov
Pri vpeljavi SOA in implementaciji integracijskih postopkov je včasih potrebna tudi faza migracije podatkov. Migracijo podatkov izvajamo takrat, ko je treba pred prehodom v testiranje in produkcijo določene podatke bodisi združevati bodisi prestaviti v drugo podatkovno strukturo ali okolje.

Postavitev
Faza postavitve opredeljuje korake prehoda integracijskega postopka najprej na testno okolje in pozneje, po izvedbi testiranja, še na produkcijsko okolje.

Poleg osnovnih faz se v metodološkem okviru pogosto uporablja tudi faza, katere namen je organizacijo seznaniti s koncepti SOA in definirati začetni poslovni proces, na katerem vpeljavo SOA tudi začnemo. Fazo imenujemo uvodna faza vpeljave SOA.

2.3.1 Uvodna faza vpeljave SOA

Ko imamo enkrat identificiran poslovni proces, ki ga bomo informatizirali na osnovi pristopa SOA, lahko začnemo izvajati posamezne faze. Ker je identifikacija takšnega poslovnega procesa včasih zapletena, si lahko pomagamo z uvodno fazo vpeljave SOA. Uvodno fazo lahko razumemo kot dodatno aktivnost vpeljave in kot dodatno fazo metodološkega okvira, ki jo izvajamo po želji in poleg tega samo pred informatizacijo prvega poslovnega procesa. Namen uvodne faze je, da odgovorni ljudje v organizaciji skupaj s strokovnjaki s področja SOA spoznajo koncepte pristopa SOA in skupaj identificirajo področja znotraj organizacije, ki predstavljajo kandidate za vpeljavo SOA. Uvodna faza je sestavljena iz korakov, ki so na kratko predstavljeni v nadaljevanju.

Razumevanje konceptov pristopa SOA

Uspešnost projekta vpeljave SOA v organizacijo je večinoma odvisna od ljudi, ki so v projekt vključeni. Če se odgovorni ljudje v organizaciji ne zavedajo pomembnosti in prednosti pristopa SOA in kaj bo takšen projekt organizaciji prinesel, potem je projekt že od samega začetka obsojen na propad. Da pa lahko ljudje spoznajo te prednosti in pomembnosti, je nujno potrebno, da se najprej soočijo s koncepti, funkcionalnostmi in najboljšimi izkušnjami pristopa SOA.

Identificiranje neučinkovitih procesov poslovanja

Sledi korak identifikacije neučinkovitih procesov poslovanja. Kaj je neučinkovit proces poslovanja, bomo predstavili na primeru. Vzemimo zavarovalniško podjetje. To podjetje ima neučinkovit proces lansiranja neke nove storitve na trg (angl. *Time to Market*). Razlogov, zakaj je proces neučinkovit, je lahko veliko, bistveno pa je to, da zaradi takšnega procesa podjetje ni konkurenčno na trgu. Sama identifikacija takšnega procesa pa ne zadostuje. Ta proces je treba dodatno razčleniti na aktivnosti in identificirati aktivnosti, zaradi katerih je celoten proces neučinkovit. Iz prejšnjega primera bi lahko kot neučinkovito aktivnost identificirali pomanjkanje fleksibilnosti informacijskih sistemov, ki podpirajo celoten postopek od izdelave novega produkta pa vse do lansiranja novega produkta na trg. Ker pa je lahko takšnih aktivnosti več, moramo iz vseh teh izbrati take aktivnosti, ki jih lahko SOA kot pristop naslavlja. Seveda bo brez poznavanja konceptov SOA in pomoči strokovnjakov zadeva spet otežena.

Izbira začetnega postopka vpeljave

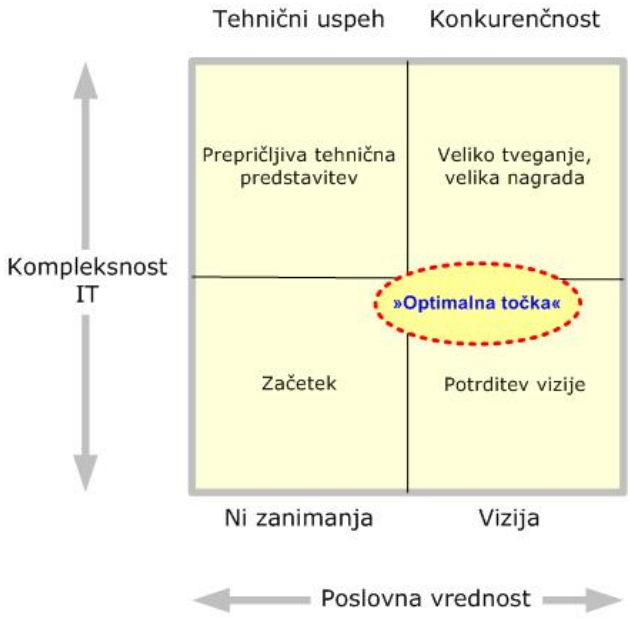
Ko smo tako razumeli koncepte pristopa SOA in identificirali glavne neučinkovite procese poslovanja znotraj organizacije in pripadajoče aktivnosti, je naslednji korak izbira začetnega postopka vpeljave. Začetni postopek mora biti eden izmed procesov, ki smo jih identificirali v prejšnjem koraku. S tem procesom bomo začeli samo vpeljavo SOA v organizacijo in s pomočjo faz vpeljave SOA neučinkovit proces poslovanja tudi izboljšali in optimizirali.

Ker pa je lahko takšnih neučinkovitih procesov veliko, je zelo pomembna izbira ustreznega. Začetni proces mora biti viden (poznan) in rezultati izvedbe procesa morajo biti dosegljivi skrbnikom procesov (Olding, Rosser, 2007). Kar pa je najpomembnejše, proces mora biti tako izbran, da je tveganje za neuspeh majhno, vendar mora biti kljub temu dovolj značilen, da predstavi pravo vrednost pristopa SOA.

Obstaja enostavno pravilo, kako takšen proces izbrati. Začetni proces ima dve lastnosti; visoka stopnja vidljivosti in najnižja stopnja tveganja (Malinverno, Barnes, 2006).Z izrazom visoka stopnja vidljivosti mislimo na to, da je proces dovolj pomemben tako za zagotovitev začetne investicije kot tudi za vzbuditev pozornosti sponzorjev. Z izrazom najnižja stopnja tveganja pa mislimo na to, da proces ne prinese večjih sprememb ali razširitev obstoječega poslovnega procesa, med drugim tudi, da zmanjša število uporabljenih tehnologij (ne

vključujemo vseh potrebnih tehnologij pristopa SOA na enkrat). Za izbiro začetnega procesa si lahko pomagamo s shemo na sliki 7.

Slika 7: Shema izbire začetnega poslovnega procesa



Vir: P. Malinverno, SOA: Where Do I Start, 2006, str. 3

Kot je razvidno s slike 7, mora biti začetni proces izbran tam (optimalna točka), kjer je kompleksnost IT pod povprečjem, poslovna vrednost pa visoka. Zadeti to točko ni enostavno, lahko pa prinese prepričljive rezultate.

Ko imamo enkrat izbran začetni postopek vpeljave, lahko začnemo izvajati posamezne faze vpeljave SOA in tako slediti metodološkemu okviru. Preden začnemo s predstavitvijo posameznih faz vpeljave SOA, je treba poudariti dve pomembni lastnosti metodološkega okvira, opaženi na osnovi izkušenj, pridobljenih na projektih. Prva lastnost je ta, da se faze izvajajo izključno zaporedno in da je samo v posebnih primerih, odvisno od postopka, določeno fazo dovoljeno in tudi smiselno preskočiti. Druga pomembna lastnost pa se navezuje na samo naravo projekta vpeljave SOA v organizacijo. Projekt vpeljave je lahko dolgotrajen projekt in najboljše izkušnje nam povedo, da vpeljava SOA ne sme na enkrat zajeti celotnega poslovanja organizacije, ampak je treba vpeljavo izvesti skozi posamezne procese, ki jih v končni fazi zlasti z vidika projektnega vodenja lahko razumemo kot podprojekte celotne vpeljave. Zaradi tega je tudi metodološki okvir zasnovan tako, da vse faze izvedemo za vsak posamezen postopek.

2.4 Management SOA

Kot je razvidno s slike 5 v poglavju (2.3), je poleg posameznih faz za metodološki okvir zelo pomemben tudi vidik managementa. Govorimo o vidiku, ki v bistvu predstavlja jedro metodološkega okvira. Posamezne faze okvira so lahko še tako dobro izvedene, a bo brez ustreznega managementa celotna vpeljava SOA v organizacijo neuspešna. V pomoč so nam tudi številna orodja, ki lahko postopek managementa bistveno olajšajo (Kenney, 2007).

2.4.1 Problematika vključenosti različnih institucij

Prva problematika, s katero se je treba soočiti, ko govorimo o managementu, je povezana z različnimi notranjimi (poslovne funkcije znotraj organizacije) ali zunanjimi (druge organizacije) partnerji, ki so vključeni v izvedbo integracijskega postopka. Metodološki okvir obe skupini partnerjev opredeljuje kot inštitucije. V obeh primerih lahko hitro pride do nerazumevanja in slabih odnosov med posamezniki, še posebno, če cilji niso pravilno zastavljeni in zato tudi rezultati niso ustrezni. Pri samem vodenju projekta sem prišel do naslednjih problemov, ki so povezani predvsem:

- Z nezainteresiranostjo inštitucij za pripravo ustreznih integracijskih elementov (spletnih storitev) za izvedbo integracijskega postopka. Ta nezainteresiranost je najpogosteje povezana s sredstvi, ki so namenjena za pripravo takšnih elementov, in slabo motiviranostjo odgovornih v inštituciji.
- Z lastninjenjem sistemov. Znotraj organizacij še vedno obstajajo skrbniki sistemov, ki sprejemajo določene programske rešitve ali sisteme kot svojo last in ponos ter nočejo nič slišati o tem, da bi imeli dostop do teh sistemov. Da ne govorimo o dejstvu, da bi na te sisteme postavili določene spletne storitve.
- S slabo pripravljenimi navodili za izdelavo ustreznih integracijskih elementov. Če inštitucije ne dobijo natančnih navodil o tem, kakšna je njihova vloga v postopku informatizacije poslovnega procesa in kaj je treba narediti, bo prišlo do tega, da bodo elementi sicer pripravljeni, vendar bodo zaradi slabih specifikacij nepopolni ali celo neuporabni.

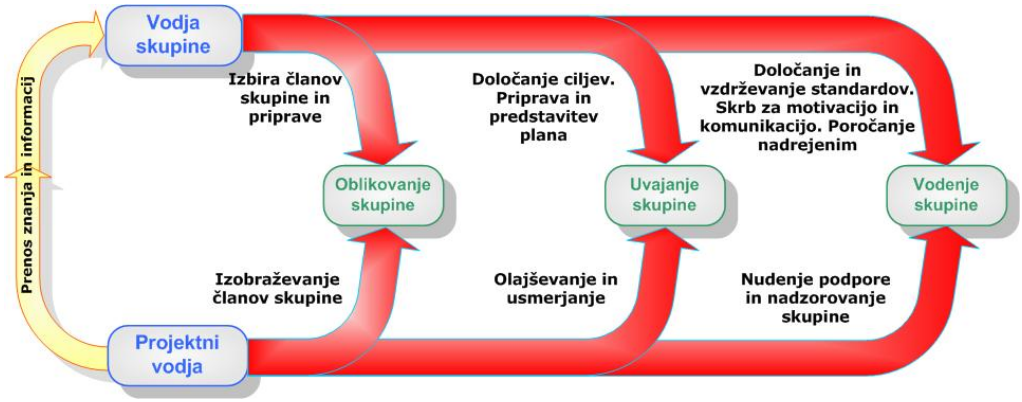
S to problematiko se je treba čim prej soočiti in, če je mogoče, že v začetnih fazah ugotoviti, kje so potencialne nevarnosti, in tja usmeriti največ pozornosti in truda. Določene probleme lahko rešimo enostavno. Problemom, povezanim s slabo pripravljenimi navodili, se lahko izognemo s pravilnim pristopom in uporabo standardov. Dober primer je recimo uporaba pristopa od zgoraj navzdol in specifikacij v obliki datoteke WSDL za implementacijo spletnih storitev. Tukaj so zadeve jasno definirane in verjetnosti, da bi bila storitev napačno implementirana, skoraj ni. Za reševanje drugih, zahtevnejših problemov (predvsem ko govorimo o zunanjih inštitucijah) pa je najpogosteje potrebna pomoč nadrejenih oziroma vodstva. V takih primerih morajo biti pravila igre natančno definirana in, kar je

najpomembnejše, natančno morajo biti opredeljene naloge, odgovornosti in tudi kazni v primeru različnih težav ali zaostajanja v okviru projektnega plana.

2.4.2 Oblikovanje projektne skupine

Uspešnost vpeljave SOA v organizacijo je odvisna tudi od projektne skupine, ki je bila za projekt oblikovana. Za oblikovanje projektne skupine sta najpogosteje odgovorni dve osebi, ki sta pozneje tudi sami del skupine. To sta projektni vodja in vodja skupine.

Slika 8: Aktivnosti oblikovanja projektne skupine



Vir: W. Humphrey, TSP(SM)–Coaching Development Teams, 2006, str. 16

Kot je razvidno na sliki 8, sta projektni vodja in vodja skupine neprestano vključena v življenjski cikel projektne skupine. Življenjski cikel se začne z oblikovanjem skupine, kjer vodja skupine izbere ustrezne profile ljudi glede na njihovo znanje in razpoložljivost v času projekta. Naloga projektnega vodje pa je njihovo izobraževanje. Sledi uvajanje skupine, kjer mora vodja skupine skupino seznaniti s cilji projekta in projektnim planom. Projektni vodja pa poskrbi za usmerjanje in posredovanje tako vsebinskih kot tehničnih nejasnosti. V času izvajanja projekta skrbi vodja skupine za določanje in vzdrževanje standardov ter njihovo upoštevanje, skrbi za motivacijo skupin in nemoteno komunikacijo med člani projektne skupine. Pomembna naloga vodje skupine je tudi poročanje rezultatov izvajanja projekta nadrejenim. V tem času projektni vodja, ki skupino nadzoruje, nudi skupini ustrezno podporo. Skozi celoten življenjski cikel projektne skupine med drugim prihaja tudi do stalnega prenosa znanja in informacij od projektnega vodje do vodje skupine. Ta prenos znanja se navezuje predvsem na dogovore in ugotovitve na statusnih sestankih ter drugih koordinacijskih dogodkih.

Pri oblikovanju skupine je zelo pomembno, da se v začetni fazi definirajo tudi osnovna pravila, ki jih mora v nadaljnjih fazah upoštevati vsak posameznik znotraj skupine (Rosen, 2004).

2.4.3 Priprava projektnega plana

Pogoj za uspešno izvedbo integracijskega postopka in izvedbo projekta vpeljave SOA v organizacijo je skrbno definiran projektni plan. Dober projektni plan definira aktivnosti, bodisi za izvedbo integracijskega postopka bodisi za pripravo posameznih elementov, ki so za izvedbo integracijskega postopka potrebni.

2.4.3.1 Struktura projektnega plana

Vpeljava SOA v organizacijo je, gledano s projektnega vidika, dolgoročen projekt, ki opredeljuje več manjših in krajših podprojektov. Podprojekti se nanašajo na izvedbo posameznega integracijskega postopka, zato je smiselno ločeno pripraviti projektni plan vpeljave SOA in projektni plan za izvedbo posameznega integracijskega postopka.

Projektni plan vpeljave SOA je širše in bolj grobo zastavljen. Aktivnosti projektnega plana definirajo posamezna integracijska področja, ki bodo izvedena v okviru vpeljave SOA v organizacijo. Tukaj seveda ne govorimo o vseh integracijskih postopkih, ampak samo o tistih začetnih (najpogosteje 3 ali 4), saj je večje število postopkov vnaprej težko opredeliti. Poleg tega pa se moramo spomniti, da uspešne izvedbe prvih postopkov (pozitiven vtis) bistveno vplivajo na izvedbo naslednjih. Glavna značilnost projektnega plana vpeljave SOA je, da se njegove aktivnosti, vsaj tiste začetne, v glavnem izvajajo zaporedno. Najboljše prakse priporočajo izvajanje enega integracijskega postopka naenkrat. Pred začetkom nove aktivnosti moramo torej najprej končati prejšnjo aktivnost. Vse aktivnosti so na isti ravni.

Po drugi strani pa se lahko aktivnosti projektnega plana integracijskega postopka izvajajo tudi vzporedno, ampak spet ne vse hkrati in vsaj ne na prvi ravni. Prvo raven aktivnosti predstavljajo faze metodološkega okvira, ki so podrobneje razdeljene v posamezne aktivnosti na drugi ravni. Aktivnosti druge ravni se lahko izvajajo vzporedno in njihovo število je odvisno predvsem od zapletenosti in vsebine integracijskega postopka. Če na primer za izvedbo integracijskega postopka ne potrebujemo spletnih storitev, je v fazi načrtovanja aktivnost priprave specifikacij nepotrebna.
Aktivnosti projektnega plana integracijskega postopka morajo poleg datuma začetka in konca aktivnosti vsebovati tudi seznam ljudi, ki so potrebni za izvedbo aktivnosti in, kar je najpomembnejše, vedno je treba določiti tudi odgovorno osebo, ki poroča o izvedbi aktivnosti in za aktivnost tudi odgovarja.

2.4.3.2 Planiranje prioritet na projektu

Prav zaradi vključenosti številnih inštitucij v projektno skupino in v sam projekt je treba znotraj projektnega plana upoštevati tudi prioritete. Ko govorimo o prioritetah, imamo v

mislih to, da moramo določenim aktivnostim enostavno dati prednost pred drugimi in jim že v začetnih fazah posvetiti največ pozornosti. V to skupino sodijo:

- Aktivnosti, ki za samo izvedbo potrebujejo več časa in bi lahko ogrozile časovne okvire izvedbe integracijskega postopka. Včasih potrebuje implementacija nekega integracijskega elementa zaradi različnih vzrokov veliko več časa in zato je treba aktivnosti dati prednost in jo začeti izvajati čim prej.
- Aktivnosti, od katerih so odvisne druge aktivnosti. Primer take aktivnosti je lahko izdelava neke spletne storitve. Izvedba te aktivnosti je nujno potrebna recimo za orkestracijo znotraj procesa BPEL, zato je tej aktivnosti treba dati prednost.
- Aktivnosti, ki jih izvajajo zunanji izvajalci (zunanje inštitucije), in je njihova izvedba odvisna od razpoložljivih virov zunanjih izvajalcev. Moramo se zavedati, da imajo tudi zunanji izvajalci, ki nam recimo pripravljajo določene spletne storitve, določene načrte, ki jih je treba upoštevati znotraj projektnega plana integracijskega postopka.

Uspešna izvedba integracijskega postopka temelji na pripravi dobrega projektnega plana. Glede na to, da je pri pristopu SOA najpogosteje treba upoštevati tudi določene zunanje dejavnike, pa je za pripravo uspešnega projektnega plana nujno sodelovanje celotne projektne skupine, ki vključuje tudi zunanje izvajalce. Projektni vodja lahko sam pripravi projektni plan, ki pa ga mora nujno pregledati in potrditi celotna projektna skupina.

2.4.4 Vodenje projektne skupine

Vodenje je ena izmed najtežjih nalog znotraj metodološkega okvira in vpeljave SOA v organizacijo. Tudi pri vodenju se zadeve še toliko bolj zaostrijo in zapletejo zaradi vključenosti različnih inštitucij, ki jih je treba koordinirati in voditi. Problem pa niso samo inštitucije, ampak po navadi tudi ljudje znotraj organizacije, v katero SOA vpeljujemo. Ljudje znotraj organizacije so navajeni na svoj način dela in med drugim so jim novejši koncepti razvoja (kot je recimo SOA pristop) tuji in če je le možno, se jim poskušajo izogniti (primer lastninjenja sistemov, ki smo ga opisali v prejšnjem poglavju). Ker pa pristop SOA zahteva novejše prijeme, predvsem tehnološke, je treba te ljudi neprestano usmerjati in, kar je najpomembnejše, ustrezno motivirati.

Sredstva, s katerimi si lahko pomagamo pri vodenju, so dober projektni plan in različni delovni ter usklajevalni sestanki.

2.4.4.1 Usklajevalni sestanki in kontrolne točke

Koordinacija in usklajevanje je pri vpeljavi SOA in implementaciji integracijskih postopkov nujna in potrebna pri vseh fazah razvoja. Zato je priporočljivo, da se na ravni projektne

skupine organizirajo redni usklajevalni sestanki, ki morajo vsebovati naslednje točke dnevnega reda:

- pregled projektnega plana in odstopanja od plana,
- pregled aktivnosti, ki se trenutno izvajajo,
- razpoložljivost ljudi znotraj projektne skupine,
- različna odprta vprašanja,
- potencialni problemi in nevarnosti na projektu (tehnični ali organizacijski),
- kontrolne točke.

V okviru projektnega plana je smiselno načrtovati (to ni nujno konkretno opredeliti znotraj projektnega plana) tudi kontrolne točke, katerih namen je pregled nad določeno celoto in mogoče tudi izvedba manjšega testiranja ali pregled delovanja manjše funkcionalnosti. Dober primer kontrolne točke je recimo ta, da ko so vse spletne storitve izdelane, lahko te storitve pregledamo, testiramo (obstajajo številni programi za testiranje) in skupaj ugotovimo ustreznost storitev.

Zelo pomembna naloga pri samem vodenju projekta in vodenju usklajevalnih sestankov je priprava zapisnika. Za vsak usklajevalni sestanek in še posebno za kontrolne točke je nujna priprava zapisnika. V zapisniku morajo biti opredeljene vse naloge in dogovori, ki so bili na sestanku sprejeti in ugotovljeni. V primeru naloge pa mora zapisnik vsebovati tudi odgovorno osebo in datum opravljanja naloge. Zapisnik mora biti posredovan vsem članom projektne skupine (tistim, ki se udeležujejo usklajevalnih sestankov), ki ga morajo nato temeljito pregledati, če je treba, kaj dopolniti, in potem tudi potrditi.

Z vidika vodenja projekta so lahko zapisniki zelo pomembno orodje za uspešno izvedbo samega postopka. Med usklajevalnimi sestanki in predvsem kontrolnimi točkami pridejo večkrat na dan ugotovitve, ki jih v začetnih fazah razvoja ni bilo mogoče predvideti ali pa so bile spregledane. Vse te ugotovitve so predmet zapisnikov, ki omogočajo sledenje in njihovo upoštevanje v nadaljnjih fazah. V kriznih situacijah pa omogočajo uspešno reševanje neskladij med člani projektne skupine.

2.4.4.2 Skupen razvoj in priprava t. i. »delavnic«

Med izvedbo integracijskega postopka samo usklajevalni sestanki niso dovolj, ampak je potrebno tudi skupno delo pri določeni aktivnosti. Ko govorimo o skupnem delu, imamo v mislih to, da manjša skupina razvijalcev skupaj izvede določeno aktivnost. Tukaj ne moremo govoriti o sestankih, saj gre za konkretno izvedbo neke naloge, ampak govorimo o t. i. delavnicah. Delavnice so posebno pomembne in nujne zaradi dveh razlogov:

- Pristop SOA prinaša številna nova znanja in nove tehnične/tehnološke prijeme. Včasih znanje znotraj organizacije, v katero SOA vpeljujemo, ni na ustrezni ravni, zato je potrebno skupno sodelovanje in skupna priprava določenih elementov integracijskega

postopka. V takem primeru so delavnice še posebno pomembne, ker razvijalci hitreje pridobivajo nova znanja in so aktivnosti izvedene v krajšem času.

- Izvedba določenih aktivnosti zahteva uskladitev številnih ljudi projektne skupine in takšna uskladitev lahko terja veliko časa ter vpliva na projektni plan. S pomočjo delavnice lahko hitro zberemo skupaj vse potrebne ljudi in aktivnost s skupnimi močmi končamo. Primer take delavnice je recimo spet aktivnost orkestracije procesa BPEL. Od posameznih inštitucij bi lahko zahtevali samo naslove spletnih storitev (prek elektronske pošte) in potem orkestrirali proces BPEL. Lahko pa organiziramo delavnico, kjer zberemo vse odgovorne za spletne storitve, skupaj zadeve povežemo in na koncu storitve proces tudi testiramo. V primeru napak pa lahko hitro rešimo določene težave.

Pozitivna stvar delavnic je tudi ta, da se na delavnicah lahko razvijalci med seboj dogovorijo o konkretnih tehničnih/tehnoloških rešitvah in si izmenjujejo različna mnenja. Poleg tega pa prihaja na delavnicah do prenosa znanja in pridobivanja novih idej.

2.4.4.3 Motiviranje

Motiviranje projektne skupine je zelo pomemben vidik vodenja. Vpeljava SOA in izvajanje integracijskih postopkov nujno potrebuje motivirano ekipo. Če so člani ekipe motivirani, potem sami čutijo neko pripadnost tej ekipi in sami naredijo vse, kar je potrebno (mogoče še več), da je določena naloga opravljena. Med seboj si delijo neko obvezo do uresničitve skupnega cilja. Vprašanje, ki se lahko tukaj pojavi, pa je, kako zagotoviti visoko stopnjo motiviranosti ekipe skozi celoten projekt vpeljave SOA.

Za doseganje ustrezne stopnje motiviranosti ekipe je zelo pomembno, da ekipa stalno pridobiva povratne informacije o delu, ki ga izvaja, in nalogah, ki jih opravlja. Ni zmagovalne ekipe, ki ne bi poznala svojih rezultatov (Humphrey, 2006). Ekipa mora biti stalno obveščena, ali je recimo učinkovita in celo prehiteva s projektnim planom ali pa ni učinkovita in zaostaja. Neprestano mora pridobivati informacije o napredovanju in o statusu projekta samega. Povratne informacije so najpomembnejši dejavnik pri ohranjanju motiviranosti ekipe.
Drugi pomemben vidik je povezan z občutkom pripadnosti projektu. Ekipa, ki je motivirana, sprejema projekt za svojega in se za sam projekt počuti odgovorna. O tem projektu celo govori z določeno mero ponosa. Zaradi tega morajo biti člani ekipe neprestano vključeni v različna usklajevanja, še posebno, ko se sprejemajo pomembne tehnične ali organizacijske odločitve okrog samega projekta. S takim načinom dela dobijo člani ekipe občutek pomembnosti in se začnejo zavedati, da lahko določene naloge naredijo samo oni. To jim daje nov elan in povečuje njihovo motivacijo.

Tretji vidik pa se navezuje na naloge, ki jih morajo člani ekipe opraviti. Če so aktivnosti in naloge dobro definirane in načrtovane v nekem realnem delovnem času, potem ni nobenih

težav in so naloge v predvidenih časovnih okvirih tudi izvedene. Takoj pa, ko ekipa začuti neko negotovost pri opredelitvi posameznih nalog in ne ve, kje bi lahko dobila posamezne napotke, kako nalogo opraviti, lahko to negativno vpliva na njeno počutje in motivacijo. Ekipa bo začela delovati zmedeno in brez kakršnegakoli medsebojnega sodelovanja.

Za zaključek tega poglavja povejmo, da je vodenje in koordinacija bodisi projektne skupine bodisi projekta samega pomemben vidik uspešne vpeljave SOA v organizacijo. Sama narava projekta zahteva številne prijeme in orodja za uspešno izvajanje posameznih aktivnosti in, kar je bistveno, za ohranjanje ustrezne stopnje motiviranosti projektne skupine.

3. OPREDELITEV POSAMEZNIH FAZ VPELJAVE

3.1 Analiza postopka

Podobno kot pri drugih metodoloških okvirih se tudi pri vpeljavi SOA v organizacijo postopek začne z analizo. Ko govorimo o analizi, mislimo na analizo poslovnega procesa ali, bolje rečeno, enega ali več integracijskih postopkov znotraj procesa. Pri vpeljavi SOA gre v večini primerov za izvajanje določenih integracijskih postopkov bodisi na osnovi nekih obstoječih poslovnih procesov, ko te poslovne procese na nek način optimiziramo, bodisi na osnovi novih poslovnih procesov, s katerimi želimo izboljšati poslovanje, zato je glavna naloga faze analize modeliranje enega ali več poslovnih procesov in definiranje posameznih integracijskih postopkov, ki nastopajo znotraj procesa.

3.1.1 Oblikovanje razvojne skupine SOA

Trenutno je bila največja evolucija v razvoju programske opreme prehod iz klasične arhitekture programske opreme v pristop SOA (Mittal, 2006). Pri klasični arhitekturi gledamo na projekt kot na proces, rezultat katerega je programska rešitev. Pri pristopu SOA pa je rezultat množica integriranih storitev (obstoječih in novih). Ker se je sam pristop razvoja močno spremenil, potrebujemo tudi različne skupine ljudi, ki lahko takemu pristopu sledijo, zato je uspešnost vpeljave SOA v organizacijo močno odvisna tudi od razvojne ali projektne skupine.

Uspešna razvojna skupina SOA mora vključevati vsaj naslednje skupine ljudi (Mittal 2006, str. 1):

- arhitekta,
- razvijalca,
- poslovnega analitika,
- projektnega vodjo in vodjo skupine.

Število ljudi znotraj posamezne skupine je odvisno od velikosti projekta.

Arhitekt
V večjih organizacijah običajno najdemo dve skupini arhitektov. Glavni arhitekti določajo pravila, najboljše izkušnje in postopke, ki se jih morajo držati vsi znotraj organizacije. Arhitekti programskih rešitev pa skrbijo za arhitekture posameznih programskih rešitev in za to, da programske rešitve zadovoljujejo trenutne in prihodnje poslovne potrebe.

Vloga glavnega arhitekta pri pristopu SOA je pospeševanje osvojitve samega pristopa. Arhitektom programskih rešitev in razvijalcem pomagajo razumeti osnove pristopa SOA in pomagajo preslikati poslovne zahteve v storitve, ki jih te skupine pozneje razvijejo in dajo na voljo drugim sistemom. Odgovorni so tudi za definiranje pravil v zvezi z managementom in varnostjo ter sprejemajo odločitve glede managementa spletnih storitev (verzioniranje, ukinjanje), orkestracije in storitvenega vodila.

Arhitekt programskih rešitev pri pristopu SOA mora zagotavljati, da napisana programska koda sledi določenim najboljšim izkušnjam in je storitveno usmerjena. Poslovne zahteve mora znati preslikati v storitve in skupaj z glavnim arhitektom skrbi, da je izkoriščena vsaka možnost ponovne uporabljivosti posameznih storitev.

Razvijalec
Razvijalec najpogosteje izvaja vlogo načrtovalca storitev, ki ima znotraj pristopa SOA ključni pomen. Načrtovanje storitev je naloga, ki zahteva tako poslovno kot tehnično znanje (Tilkov, 2007). Naloge so povezane predvsem z grupiranjem operacij znotraj storitve, odločanjem o nazivu storitve in granularnostjo storitve, delom z napakami in z drugimi vidiki, ki vplivajo na samo strukturo storitve.

Poslovni analitik
Vloga poslovnega analitika je ključnega pomena pri pristopu SOA, saj se njegove naloge najbolj spreminjajo. Dve pomembni nalogi, ki jih poslovni analitik opravlja:

- komunicira z nadrejenimi in ljudmi na strateškem nivoju z namenom razumevanja njihovih potreb in zahtev sistema ter
- komunicira s tehničnimi osebjem, da zahteve pretvorijo v tehnične specifikacije, ki so potrebne za razvoj.

Ko govorimo o pristopu SOA pa mora poslovni analitik še:

- sodelovati s celotno razvojno skupino in jo usmerjati v storitveno usmerjen način razmišljanja (katere storitve potrebujejo, katere storitve že imajo) ter
- sodelovati s tehnično skupino za načrtovanje in postavitev ustreznih storitev.

Projektni vodja in vodja skupine
Vlogi sta podrobneje opisani v poglavju 2.4.2.

Za doseganje ustrezne stopnje kvalificiranosti posameznikov znotraj skupine SOA so nujno potrebna usposabljanja in izobraževanja. Čeprav izobraževanje v začetnih fazah zahteva določen čas in napor, se dolgoročno povračilo skriva v hitrejšem doseganju kompetentnosti in učinkovitosti, ko se začnejo zahteve povečevati (Olding, Rosser, 2007).

Posamezne skupine s pripadajočimi člani so znotraj metodološkega okvira predstavljene z organizacijsko shemo.

3.1.2 Analiza obstoječih poslovnih procesov

Ko želimo s pomočjo pristopa SOA izboljšati in optimizirati določen poslovni proces znotraj organizacije, ga moramo najprej dobro razumeti. Kot smo definirali že v uvodnih poglavjih, je vsak poslovni proces sestavljen iz množice aktivnosti, ki si sledijo v nekem določenem zaporedju, odjemalcu pa vrača določen rezultat izvajanja. Če hočemo razumeti delovanje poslovnega procesa, ga moramo najprej razdeliti na aktivnosti in za vsako posamezno aktivnost določiti funkcijo, ki jo opravlja. Poslovni proces in aktivnosti, ki ga sestavljajo, lahko opišemo v besedilu in natančno naštejemo glavne lastnosti. To je sicer podroben opis, ki pa nam žal ne daje nekega pregleda nad celoto. Učinkovitejše in natančneje je opredeliti poslovni proces s poslovnim modelom, ki smo ga predstavili že v prejšnjih poglavjih.

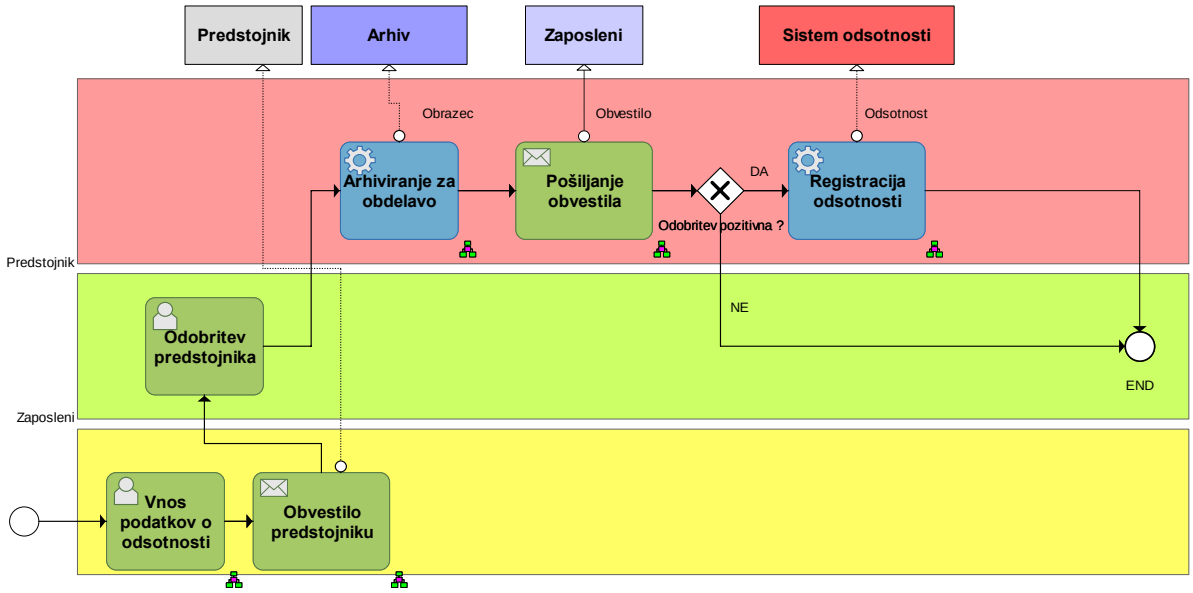
Takoj, ko govorimo o poslovnem modelu, se srečamo z modeliranjem. Z dobrim modeliranjem lahko izdelamo tak poslovni model, ki bo razumljiv tako poslovnim analitikom kot pozneje tudi razvijalcem, ki bodo poslovni model ali v končni fazi poslovni proces tudi implementirali. Prva faza metodološkega okvira vpeljave SOA torej temelji na modeliranju poslovnih procesov na osnovi notacije BPMN.

V fazi modeliranja sem na projektih, glede lastnosti dobrega poslovnega modela, prišel do naslednjih ugotovitev:

- Model mora biti kronološko urejen. To pomeni, da si morajo aktivnosti na modelu slediti v kronološkem časovnem vrstnem redu najpogosteje od leve proti desni.
- Model se sproži z nekim dogodkom in na koncu, ko se izvedejo določene aktivnosti, vrne nek poslovni rezultat.
- Vse aktivnosti in naloge na modelu pripadajo določeni vlogi, ki je pomembna za določene ljudi v organizaciji. Pri modeliranju je treba zajeti vse potrebne vloge, ki pa so včasih locirane tudi zunaj meja organizacije.
- Potek izvajanja aktivnosti mora natančno določati tudi, kako se podatki prenašajo in predvsem, kam (v katere sisteme, podsisteme). Večina problemov, s katerimi se organizacije soočajo, temelji na medsebojnih odvisnostih, ki jih najenostavneje identificiramo s pomočjo diagrama poteka izvajanja.
- Poslovni proces je lahko modeliran tudi hierarhično in ga lahko opazujemo na različnih ravneh. To pomeni, da imamo lahko model znotraj modela.
- Možnosti izbire pri odločitvah znotraj modela definirajo in pogojujejo, katera pot bo izbrana (katere aktivnosti bodo izvedene).

Na sliki 9 je predstavljen primer poslovnega modela, na katerem so opredeljene posamezne aktivnosti skupaj z odgovornimi za posamezno aktivnost.

Slika 9: Grafična predstavitev poslovnega modela



3.1.3 Definiranje ciljev vpeljave SOA

Ko govorimo o ciljih vpeljave SOA v organizacijo, imamo v mislih predvsem definiranje ciljev, bodisi optimizacije obstoječega poslovnega procesa znotraj organizacije bodisi implementacije novega poslovnega procesa. Najpogosteje se srečujemo z optimizacijo obstoječega poslovnega procesa, ki smo ga v začetku faze analize najprej identificirali in ga zdaj želimo optimizirati.

V tem koraku se še ne sprašujemo, KAKO bomo proces optimizirali, ampak si najprej odgovorimo, KAJ bomo optimizirali in ZAKAJ. To pomeni, da moramo najprej identificirati tiste neuspešne aktivnosti znotraj procesa (KAJ) in ugotoviti, ZAKAJ so te aktivnosti neuspešne. Šele ko odgovorimo na ti dve vprašanji, lahko začnemo razmišljati o tem, kako bomo zadeve optimizirali.

Da bi do teh odgovorov lažje prišli, si lahko v fazi analize pomagamo tudi z merjenjem in simuliranjem izvajanja poslovnega procesa. S pomočjo simulacij lahko opazujemo izvajanje poslovnega procesa (modela) in ovrednotimo rezultate, ki jih proces vrne. Na osnovi rezultatov enostavneje identificiramo neuspešne aktivnosti in s spreminjanjem neuspešnih aktivnosti hitreje pridemo do prednosti, ki jih taka sprememba prinese v poslovnem procesu.

3.1.4 Identifikacija integracijskih postopkov

Zelo pomemben korak v fazi analize je identifikacija integracijskih postopkov za določen poslovni proces. Posamezen poslovni proces ima lahko enega ali več integracijskih postopkov, odvisno od tega, kako so sam proces in sistemi, ki proces podpirajo, zamišljeni.

Kot primer vzemimo obdelavo prejetega računa v podjetju. Poslovni proces je tako zasnovan, da se novo prejeti račun najprej s pomočjo skenerja zajame in shrani v sistem X. V sistemu X se račun opremi z metapodatki in pošlje v sistem Y. V sistemu Y se izvede likvidacija računa in nato se podrobnosti o računu (posamezne postavke po stroškovnih mestih) spet posredujejo v sistem X. V opisanem primeru imamo dva integracijska postopka:

- prenos računa iz sistema X v sistem Y skupaj z metapodatki,
- prenos posameznih postavk računa iz sistema Y v sistem X.

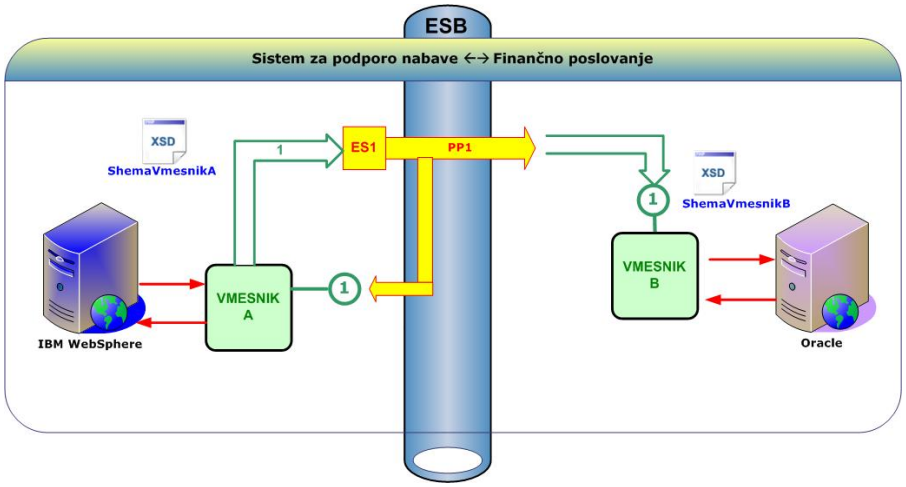
Ta dva postopka sta z vidika poslovnega procesa povezana in soodvisna, z vidika implementacije pa ju lahko obravnavamo in implementiramo neodvisno.

Za vsak integracijski postopek je treba opredeliti naslednje elemente:

- **izvorni sistem:** Sistem, ki sproži izvajanje integracijskega postopka in se uporablja kot izvor podatkov, ki jih je treba prenesti.
- **ponorni sistem:** Sistem, ki sprejme podatke, prenesene s pomočjo integracijskega postopka.
- **povratna informacija:** Definirati je treba, ali izvorni sistem potrebuje povratno informacijo o uspešnosti prenosa podatkov v ponorni sistem.
- **način prenosa**: Integracijski postopek se lahko izvede sinhrono ali asinhrono (podrobnosti posameznega prenosa so opisane v drugi fazi metodološkega okvira).
- **groba struktura podatkov**: Definirati je treba grobo strukturo podatkov in identificirati posamezne skupine podatkov (v našem primeru je to račun in postavke računa).

Na osnovi identifikacije integracijskih postopkov poslovnega procesa in opredelitve podrobnosti posameznega postopka pridemo do t. i. integracijske sheme.

Slika 10: Grafična predstavitev integracijske sheme



Na sliki 10 je prikazan primer integracijske sheme, na kateri sta predstavljena dva sistema, ki sta povezana prek storitvenega vodila SOA. Sistem za podporo nabave s pomočjo vmesnika A (ta tvori ustrezno zahtevo za storitveno vodilo) posreduje podatke na storitveno vodilo, ki podatke sprejme in preusmeri na sistem za finančno poslovanje prek spletne storitve na vmesniku B. Po uspešnem prenosu storitveno vodilo s klicem spletne storitve vmesnika A sporoči rezultat prenosa (uspešen prenos ali napaka).

3.1.5 Identifikacija integracijskih elementov

V fazi analize je poleg integracijskih postopkov treba identificirati tudi vse integracijske elemente, ki so vključeni v postopek. Integracijski elementi so vsi ključni gradniki, ki omogočajo implementacijo integracijskega postopka in ki jih je treba upoštevati v vseh fazah vpeljave metodološkega okvira. Metodološki okvir opredeljuje naslednje skupine integracijskih elementov:

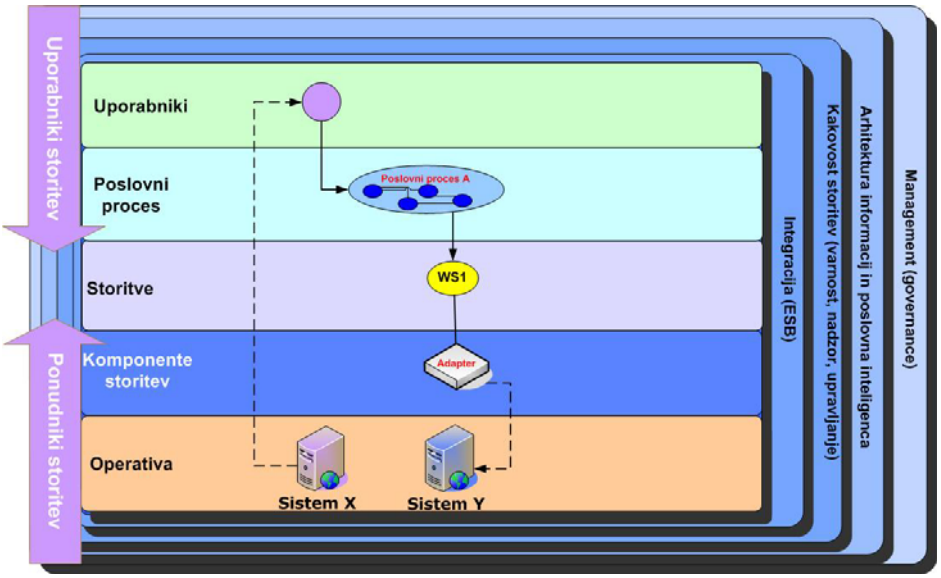
- vključeni sistemi (programske rešitve, informacijski sistemi, podatkovne strukture),
- storitvene komponente (vmesniki in adapterji).

Integracijski postopek lahko razumemo tudi kot mehanizem, ki poskrbi za prenos podatkov iz enega sistema (izvor) v drugi sistem (ponor). Ta mehanizem pa mora biti sposoben pridobiti tako podatke iz izvornega sistema kot tudi predati podatke ponornemu sistemu. Torej so potrebni neke vrste povezovalni moduli, ki poskrbijo za povezovanje integracijskega postopka z izvornim in ponornim sistemom. Te povezovalne module imenujemo storitvene komponente (sem spadajo različni vmesniki in adapterji).

Poleg poslovnega modela in integracijske sheme je v fazi analize pomembna tudi shema integracijskih plasti. V shemi so opredeljeni posamezni integracijski elementi ter njihova

medsebojna odvisnost. Primer sheme skupaj z opisom posameznih plasti je opredeljen na sliki 11.

Slika 11: Grafična prestavitev integracijske plasti



Operativa

Opredeljene so vse programske rešitve, moduli in drugi sistemi, ki podpirajo poslovne funkcije znotraj organizacije in so vključeni v integracijski postopek. Razdelitev arhitekture SOA na različne plasti povečuje vpliv obstoječih sistemov in omogoča integracijo med njimi na osnovi storitveno usmerjenih integracijskih tehnik (Arsanjani, 2004).

Komponente storitev

Posamezne programske komponente so definirane z implementacijo določenih operacij, ki omogočajo realizacijo aktivnosti določenih storitev. Komponente storitev odražajo definicijo storitve tako s funkcionalnega vidika kot tudi z vidika kakovosti same storitve.

Storitve

Vse storitve posameznega integracijskega postopka so definirane znotraj plasti storitev. Arhitektura SOA definira storitev kot abstraktno zbirko poslovnih funkcij (Arsanjani, 2007). Specifikacije storitev morajo nuditi dovolj informacij, da jih lahko drugi sistemi uporabijo na tehnološko neodvisen način.

Poslovni proces

Kompozicija in orkestracija storitev iz prejšnje plasti so definirane v tej plasti. Kompozicijo uporabimo za združevanje skupin storitev v neko zaporedje, orkestracijo pa za združevanje posameznih storitev v zaporedje. Tako s pomočjo storitev gradimo programsko rešitev, ki temelji na definiranem poslovnem procesu.

Uporabniki
Plast uporabnikov ali predstavitvena plast poskrbi za vse potrebne zahteve pri posredovanju podatkov končnemu uporabniku. Prav tako omogoča realizacijo specifičnih uporabniških vmesnikov, kot so portali, debeli odjemalci, spletne programske rešitve in drugi sistemi, ki končnemu uporabniku omogočajo dostop do poslovnih storitev.

Integracija
Ključna plast pristopa SOA je plast integracije, ki omogoča mediacijo, preusmerjanje in prenos zahtev od odjemalca do ustreznega ponudnika storitve. Integracija se navezuje predvsem na integracijo elementov, predstavljenih od plasti komponent do plasti poslovnih procesov.

Kakovost storitev
Znotraj pristopa SOA so posebnosti, ki lahko dodatno vplivajo na kakovost storitev. Primer takih posebnosti je šibka sklopljenost, virtualizacija, uporaba struktur XML, združevanje sestavljenih storitev. Te lastnosti lahko povzročajo težave, zato jih je treba še toliko bolj upoštevati in nadzirati.

Arhitektura informacij in poslovna inteligenca
Plast arhitekture informacij in poslovna inteligenca opozarjata na arhitekturo podatkov in arhitekturo informacij, na katere je tudi treba biti pozoren. Te se lahko v končni fazi uporabljajo kot osnova za razvoj poslovne inteligence skozi področna in osrednja podatkovna skladišča.

Management
Smernice in politika sprejemanja odločitev glede pristopa SOA so zbrane v plasti managementa. Plast managementa je prisotna v vseh drugih plasteh in lahko bistveno pospeši izvedbo določenega integracijskega postopka kot tudi celotne vpeljave SOA.

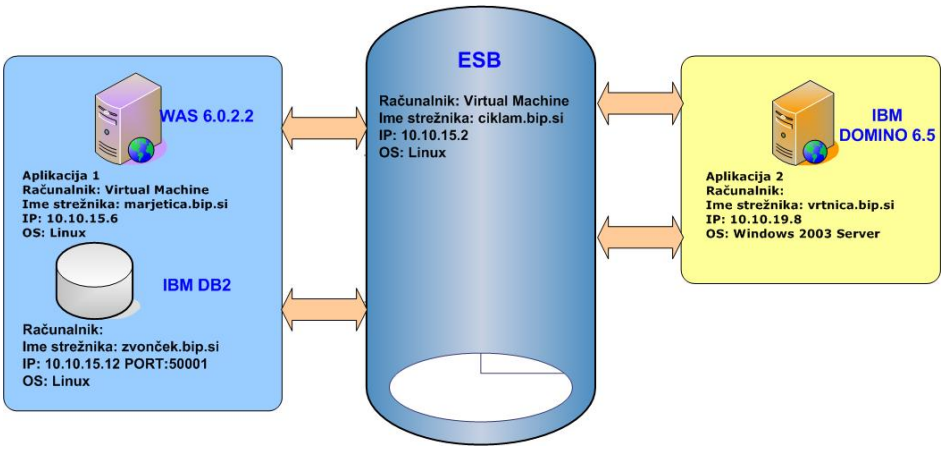
3.2 Načrtovanje in specifikacije

Ko smo enkrat dobro definirali integracijski postopek in opredelili glavne integracijske elemente postopka, je treba posamezne elemente tudi natančno specificirati. Še posebno so specifikacije pomembne v primeru, ko so za implementacijo posameznih integracijskih elementov zadolženi ljudje ali podjetja zunaj organizacije. Specifikacije morajo biti torej tako pripravljene, da so nedvoumne in da niso odvisne od tehnološke platforme.

3.2.1 Priprava arhitekturnega diagrama

Prva stvar, s katero se je v fazi načrtovanja treba soočiti, so posamezni informacijski sistemi in programske rešitve (v nadaljevanju sistemi), ki so vključeni v integracijski postopek. Te sisteme je treba že v začetku identificirati in opredeliti platformo, na katerih sistemi potekajo. Znotraj metodološkega okvira vpeljave SOA je to zagotovljeno s pomočjo arhitekturnega diagrama. Na sliki 12 je prestavljen primer arhitekturnega diagrama, ki vključuje dva sistema.

Slika 12: Grafična prestavitev arhitekturnega diagrama



Glede na to, da so za vsak sistem znotraj arhitekturnega diagrama definirani konkretni naslovi, je smiselno pripraviti arhitekturni diagram za vse faze razvoja. Ker razvoj na osnovi pristopa SOA tako kot klasični razvoj temelji na postopku RTP (razvoj, test, produkcija), je smiselno pripraviti arhitekturni diagram za vse tri faze.

3.2.2 Identifikacija in priprava specifikacij spletnih storitev

Kot smo ugotovili v prejšnjih poglavjih, je ena izmed glavnih aktivnosti pri implementaciji integracijskega postopka in pristopa SOA nasploh orkestracija spletnih storitev. Orkestracijo si lahko predstavljamo kot centralni proces, ki koordinira izvajanje operacij znotraj spletnih storitev, ki so vključene v proces (Jurič, 2007). Preden aktivnost orkestracije sploh izvedemo, je treba zagotoviti vse spletne storitve, ki jih integracijski postopek potrebuje. Če spletna storitev za določen postopek že obstaja, jo lahko znova uporabimo. V nasprotnem primeru pa je treba izdelati novo. Vprašanje, ki se tukaj pojavlja, je, kako pripraviti natančna navodila za implementacijo spletne storitve, zlasti v primeru, ko je za implementacijo določene storitve zadolženo neko zunanje podjetje. Če hočemo zagotoviti določeno stopnjo standardizacije in enostavnega managementa, morajo tudi specifikacije spletnih storitev ustrezati določenim standardom in po teh standardih morajo biti tudi implementirane storitve.

Specifikacija spletne storitve mora vsebovati elemente, predstavljene na sliki 13 (Pijanovski, 2008).

Slika 13: Elementi spletne storitve

Vir: K. Pijanovski, The UDDI API Sets, 2008

Najprimernejši način specifikacije spletnih storitev se doseže z uporabo strukture WSDL (angl. *Web Service Description Language*). WSDL je zapis v formatu XML, ki opisuje spletne storitve kot množico končnih operacij s pripadajočimi podatkovnimi strukturami (sporočili). Operacije in sporočila so predstavljena abstraktno in so vezana na konkreten komunikacijski protokol.

3.2.3 Specifikacije podatkovnih struktur

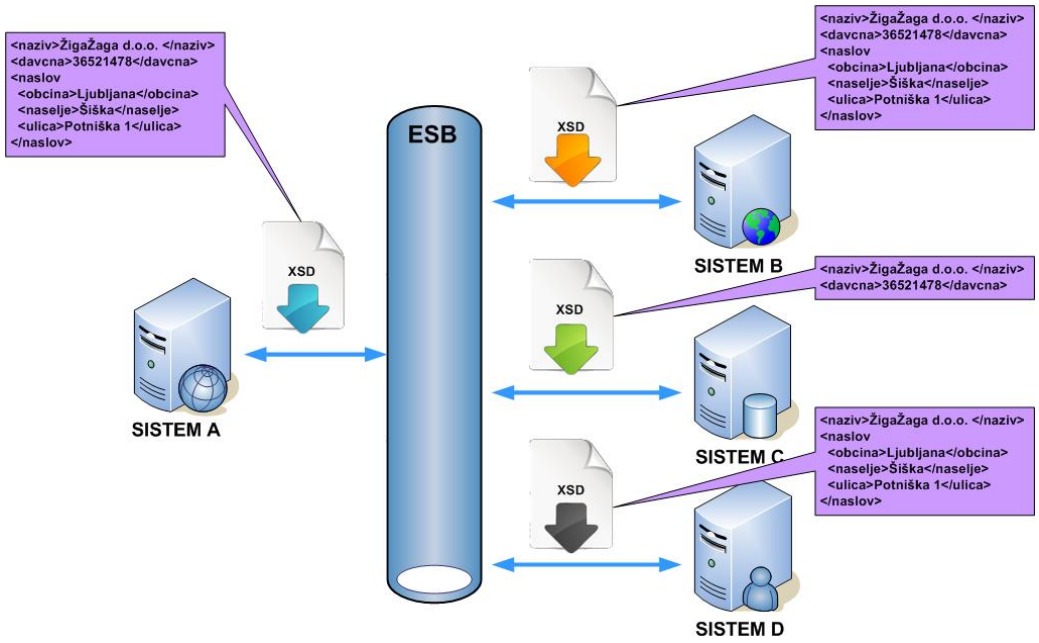
Specifikacija podatkovnih struktur, ki se uporabljajo pri integracijskem postopku, je ena izmed najpomembnejših aktivnosti v fazi načrtovanja. Dobro definirane strukture v fazi načrtovanja lahko bistveno skrajšajo čas implementacije postopka in zmanjšajo možnost poznejših napak pri samem izvajanju postopka.

Tako kot pri podatkovnem modeliranju (priprava konceptualnega modela) gre tudi pri pripravi specifikacij podatkovnih struktur za identifikacijo posameznih elementov (atributov). Te atribute grupiramo v različne skupine in jim določamo različne metapodatke (tip, dolžino in omejitve). Najprimernejši način za specifikacijo podatkovnih struktur je uporaba strukture XSD (angl. *XML Schema Definition*). Podobno kot WSDL je tudi XSD zapis v formatu XML, s katerim opisujemo strukturo in organizacijo podatkov. Uporablja se za definiranje katerihkoli tipov podatkov: števil, datumov, tekstovnih polj, seznamov itd.

V času priprave podatkovnih struktur je zelo pomembno, da so v to aktivnost vključeni tudi poslovni analitiki, ki dobro poznajo sisteme, ki so vključeni v integracijski postopek, in tudi podatke, ki jih želimo med sistemi prenašati. Sama aktivnost se začne s pripravo seznama vseh struktur, v katerih so opredeljeni vsi potrebni podatki za prenos. Če recimo govorimo o integraciji poslovnih partnerjev, potem bo ta seznam vseboval podatke, kot so naziv, davčna številka, naslov itd. V tej fazi se nam pogosto pojavlja vprašanje, ali mora seznam vsebovati samo podatke, ki jih potrebuje ciljni sistem, ali vse podatke, ki jih izvorni sistem lahko ponudi. Metodološki okvir predvideva drugo možnost. Razlog je preprost in smiseln. Zaradi ponovne uporabljivosti in drugih prednosti, ki jih pristop SOA nudi, je treba v storitveno

vodilo pripeljati čim več podatkov. Proces pa mora potem poskrbeti, da posameznemu ponoru dodeli samo tiste podatke, ki jih potrebuje. Če imamo torej izvorni sistem A, znotraj katerega upravljamo s poslovnimi partnerji, in te partnerje prenašamo v ponorne sisteme B in C, potem je naloga procesa in storitvenega vodila, da v sistem B ne pošilja naslova poslovnega partnerja, ker ga sistem B ne potrebuje (niti nima ustreznih struktur v podatkovni bazi za beleženje tega podatka). Opisani primer je ilustriran na sliki 14.

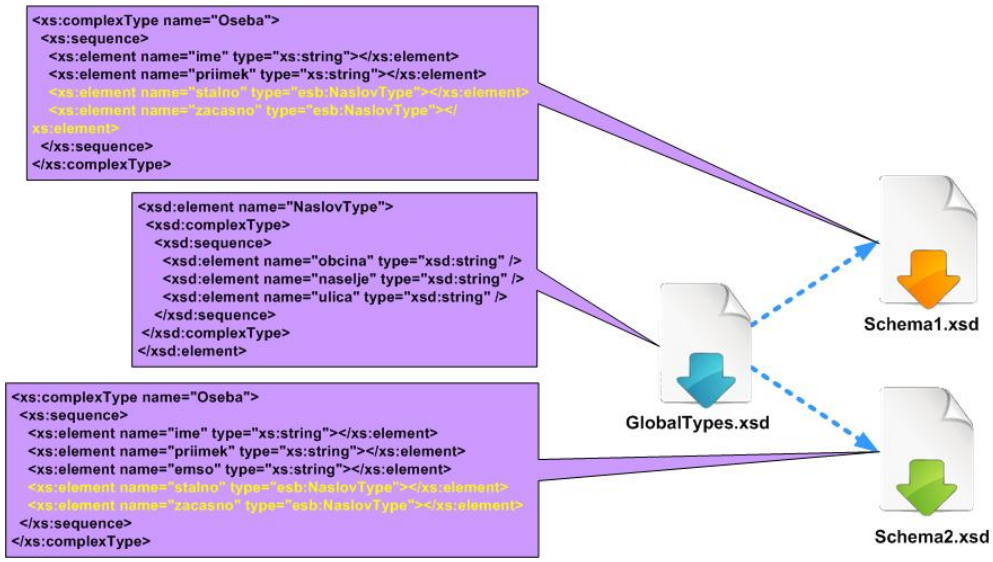
Slika 14: Grafični prikaz vsebine sporočil integracijskega postopka



Ta izbira se še posebno izkaže kot uporabna v primeru, ko je treba pozneje v integracijski postopek vključiti še dodatni sistem, denimo sistem D. Z načinom definiranja podatkov, kot smo ga opisali v zgornjem odstavku, lahko sistem D priključimo na storitveno vodilo in s pomočjo preslikovanja podatkov na storitvenem vodilu pripeljemo v sistem D samo potrebne podatke. To pomeni, da v izvornem sistemu nič ne spreminjamo ali dodajamo.

Druga pomembna lastnost metodološkega okvira, ko govorimo o specifikaciji podatkovnih struktur, je povezana s ponovno uporabljivostjo sestavljenih struktur. Za primer vzemimo spet podatke o osebi, vendar v tem primeru fizični osebi. Vsaka oseba ima vsaj dva sklopa podatkov za naslov. To sta naslov stalnega prebivališča in naslov začasnega prebivališča (začasnega prebivališča nima vsaka oseba). Ti dve skupini podatkov sta popolnoma enaki, saj tako začasno kot stalno prebivališče vsebuje podatke o občini, naselju, ulici in hišni številki. Torej bomo specifikacije pripravili tako, da bomo najprej specificirali globalno strukturo (kompleksni tip) s podatki o naslovu ter to strukturo uporabili za začasno in stalno prebivališče. Metodološki okvir in najboljša praksa priporočata, da se takšne globalne strukture hrani v ločenih (skupnih) shemah, ki jih je mogoče pozneje vključiti v posamezne sheme za konkreten integracijski postopek.

Slika 15: Grafični prikaz razdelitve shem podatkov



Kot je razvidno iz slike 15, je globalna struktura definirana v ločeni shemi XSD (*GlobalTypes.xsd*), ki jo lahko pozneje vključijo posamezne sheme. S tem dosežemo, da imamo globalne strukture enkrat definirane in konsistentne.

Dobro definirane podatkovne strukture, ki jih pregledajo poslovni analitiki, so glavni pogoj za nadaljevanje integracijskega postopka in prehod v fazo verifikacije koncepta ali izgradnje. Če strukture niso dobro definirane in je treba v fazi izgradnje dodajati podatke, lahko to bistveno podaljša čas izdelave samega postopka ter prinese veliko dodatnega in nepotrebnega spreminjanja integracijskih elementov.

3.2.4 Definiranje nalog in odgovornosti

Ko smo enkrat specificirali spletne storitve in podatkovne strukture, je treba opredeliti posamezne projektne naloge in določiti odgovorne ljudi. Ta aktivnost je še zlasti pomembna pri integracijskih postopkih, pri katerih je projektna skupina sestavljena ne samo iz ljudi znotraj organizacije in izvajalca samega, ampak so v projektno skupino vključeni tudi nekateri zunanji izvajalci. Brez natančno opredeljenih nalog in predvsem definiranja odgovornih ljudi sta management in vodenje takšnega projekta težka in postaneta kmalu neobvladljiva.

Za izvedbo slednje aktivnosti je znotraj metodološkega okvira definiran t. i. seznam nalog in odgovornosti, ki za vsak integracijski element natančno definira nalogo, ki jo je treba opraviti, rok za izvedbo in odgovorno osebo. Posamezni integracijski vmesniki so predstavljeni v integracijski shemi faze analize. Primer enega elementa seznama je podan v tabeli 1.

Tabela 1: Primer definirane naloge znotraj vmesnika

Integ. element	**Vmesnik A**
Naziv naloge	**Priprava agenta za posredovanje podatkov na ESB**
Odgovoren	**Podjetje Miška d.o.o.**
Rok izvedbe	**12 .6. 2009**

Opis naloge
Izdelati je treba programsko rešitev JEE, ki s pomočjo časovnega razporejevalnika ob določeni uri prebere podatke iz podatkovne baze in jih posreduje na storitveno vodilo ESB. Podrobnosti programske rešitve so natančneje specificirane v dokumentu »*Specifikacija vmesnika A.doc*«.

Na takšen način lahko zagotovimo, da se pred fazo izgradnje vsi člani projektne skupine zavedajo svojih nalog in tako lahko lažje načrtujemo naslednje faze.

3.2.5 Planiranje – časovna opredelitev

Planiranje je tako kot pri klasičnem razvoju ena izmed najpomembnejših aktivnosti. Zaradi vključenosti različnih ljudi v projektno skupino je tako kot pri definiranju nalog pri pristopu SOA planiranje zelo pomembno in nujno. Seveda je planiranje nesmiselno, če nimamo vnaprej definiranih posameznih nalog in če za posamezno nalogo nimamo neke ocene, v kakšnem času je nalogo treba ali jo je mogoče opraviti.

Pri vpeljavi SOA je treba planiranje obravnavati z dveh vidikov. Glede na to, da je vpeljava SOA dolgotrajen proces, sem ugotovil, da moramo najprej pripraviti grob plan vpeljave SOA. V grobem planu so opredeljene vse aktivnosti za vpeljavo SOA v organizacijo, skupaj z aktivnostmi informatizacije posameznih poslovnih procesov na osnovi SOA. Poleg grobega plana je potrebno tudi podrobno planiranje. Podrobno planiranje pa izvajamo za posamezno informatizacijo izbranega poslovnega procesa. Za vsak cikel, od analize do postavitve, izdelamo podroben projektni plan.

Planiranje je znotraj metodološkega okvira definirano s pomočjo diagrama Gantt, znotraj katerega naštejemo naloge, potrebne za izvedbo integracijskega postopka, po potrebi naloge razdelimo v podnaloge (v primeru, ko je za izvedbo določene naloge odgovornih več ljudi ali skupin) in definiramo časovne okvire. Pri planiranju je treba tudi upoštevati medsebojno odvisnost nalog. Če imamo možnost, da se dve nalogi vzporedno izvajata (to se najpogosteje pojavlja, ker so za naloge odgovorni različni ljudje), jih je treba tako tudi vnesti v projektni plan.

Ko je projektni plan izdelan, ga morajo vsi vključeni v projektno skupino skupaj pregledati in na koncu tudi potrditi. Od tega trenutka naprej je ta plan pravilo in vodja projekta mora poskrbeti, da se od projektnega plana ne sme odstopati.

3.3 Verifikacija koncepta

Verifikacija koncepta je faza, ki je ne izvajamo pri vsakem integracijskem postopku. Verifikacijo koncepta izvajamo:

- pri prvem integracijskem postopku vpeljave SOA v organizacijo,
- pri zahtevnejših integracijskih postopkih.

V prvem primeru želimo z verifikacijo koncepta predvsem ugotoviti smiselnost vpeljave SOA v organizacijo in potrditi arhitekturni pristop. Pri drugem pa poskušamo ugotoviti morebitne težave pri izvedbi integracijskega postopka, še preden gremo v natančno izvedbo.

3.3.1 Izvedba verifikacije koncepta

Ker gre pri verifikaciji koncepta za izdelavo pilotnega integracijskega postopka, je treba postopek že konkretno implementirati. Več je o sami implementaciji oziroma izgradnji postopka opisano v naslednji fazi metodološkega okvira. Tukaj omenimo samo, da se v fazi verifikacije koncepta sama implementacija razlikuje v tem, da se ne osredotočamo preveč na samo vsebino postopka, ampak bolj na tehnološki vidik. To pomeni, da ne bo nič narobe, če v fazi verifikacije koncepta ugotovimo, da določena podatkovna struktura (shema XSD, definirana v fazi načrtovanja) nima vseh potrebnih podatkov ali pa, da je določen podatek napačnega tipa.

V fazi verifikacije koncepta je treba zagotoviti, da:

- se verifikacija koncepta izvaja na osnovi specificiranih podatkovnih struktur in spletnih storitev iz faze načrtovanja,
- okolje, v katerem se verifikacija koncepta izvaja, ustreza ciljnemu okolju, v katerega bo integracijski postopek implementiran (arhitekturni diagram),
- so v pilotni integracijski postopek vključeni vsi izvorni in ponorni sistemi,
- so v pilotni integracijski postopek vključeni vsi vmesniki in adapterji,
- se podatki uspešno prenašajo med sistemi in uspešno shranjujejo v ustrezne podatkovne strukture,
- se poslovni proces znotraj storitvenega vodila pravilno izvaja ter se pravilno zapisujejo vsi metapodatki procesa,
- so metrike izvajanja integracijskega postopka v nekem sprejemljivem merilu.

3.3.2 Analiza izvedbe in priprava poročila

Ko je pilotni integracijski postopek implementiran in je izvedena verifikacija koncepta, sledi analiza izvedbe, v kateri je treba pripraviti poročilo, ki vsebuje vse ugotovitve izvedbe, ki jih moramo upoštevati v naslednjih fazah.

Poročilo mora vsebovati:

- specifikacijo pilotnega integracijskega postopka, specifikacijo spletnih storitev in specifikacijo podatkovnih struktur,
- seznam podatkovnih struktur, ki jih je treba dopolniti, in opredelitev novih podatkov za posamezno strukturo,
- seznam tehničnih ugotovitev izvajanja postopka (tehnične napake pri integraciji podatkovnih shem na različnih platformah, napake ali omejitve na vmesnikih …),
- seznam ljudi iz projektne skupine, ki so sodelovali pri izvedbi testiranja verifikacije koncepta,
- kratek opis rezultata izvedbe (statusno poročilo); zaradi lažjega razumevanja se pripravi tudi grafična shema.

Z verifikacijo koncepta želimo doseči predvsem dve stvari. V prvi fazi želimo preveriti ustreznost specifikacij, pripravljenih v predhodnih fazah, in ugotoviti kakovost izdelave specifikacij. S tem lahko hitro ugotovimo, ali smo bili pri specificiranju dovolj natančni in ali je treba pri naslednjih integracijskih postopkih vložiti še dodaten trud v specifikacije. Faza verifikacija koncepta pa je posebno pomembna pri prvih integracijskih postopkih, ker lahko v tej fazi dobro preizkusimo in preverimo razvojna orodja, ki jih uporabljamo pri izdelavi integracijskih postopkov in vpeljavi SOA nasploh. Faza verifikacije koncepta pa je zelo pomembna tudi z vidika kredibilnosti. Glede na to, da je ta faza praviloma časovno kratka, lahko hitro pridemo do rezultatov in tako do zaupanja višjega managementa.

3.4 Izgradnja

V fazi izgradnje gre za implementacijo vseh elementov integracijskega postopka. Na osnovi izdelanih specifikacij implementiramo spletne storitve, druge integracijske elemente in poslovni proces znotraj integracijskega postopka. Ko so vsi elementi implementirali, sledi aktivnost povezovanja poslovnega procesa z integracijskimi elementi, kar imenujemo tudi orkestracija.

3.4.1 Implementacija integracijskih elementov

Kot smo že povedali v prejšnjih poglavjih, sodijo med integracijske elemente tako spletne storitve kot tudi različni adapterji, s pomočjo katerih lahko enostavno dostopamo do podatkovnih struktur različnih proizvajalcev (Oracle, DB2, SAP …). Glede na to, da so adapterji že implementirani in pripravljeni za konfiguracijo ter povezovanje z ustreznimi podatkovnimi strukturami, se pri tej aktivnosti v glavnem osredotočamo na spletne storitve, ki jih je treba implementirati.

Osnova za implementacijo spletne storitve je v fazi analize pripravljena specifikacija v obliki dokumenta WSDL. Lastnost dokumenta WSDL je opredeljevanje temeljnih elementov spletne storitve (operacije, parametri), razvijalcem pa pušča proste roke pri izbiri tehnologije in platforme, na kateri se bo storitev izvajala. Seveda sta pri pristopu SOA tehnologija in platforma določeni s sistemom, ki bo spletno storitev implementiral in na katerem se bo storitev tudi izvajala. Če bomo določene podatke pošiljali v celoviti informacijski sistem Navision, bo ta spletno storitev implementiral v okolju .NET in se bo izvajala na platformi Windows.

Pristop od zgoraj navzdol

Z vidika implementacije spletnih storitev temelji metodološki okvir vpeljave SOA na pristopu od zgoraj navzdol pristopu. Ta pristop je sestavljen iz aktivnosti, ki se začnejo z izdelavo datoteke WSDL (specifikacije) in nato z implementacijo storitve na ciljni platformi. Pristop zagovarja izvedbo analize storitve pred njeno izgradnjo in postavitvijo (Erl, 2008). Druga pomembna lastnost pa je povezana z implementacijo spletne storitve, saj nas pri pristopu od zgoraj navzdol ne zanima, kako bo razvijalec storitev implementiral, kako se bodo metode imenovale in katere parametre bo posamezna metoda (storitev) imela. Vse te podrobnosti so opredeljene v datoteki WSDL in tega se more razvijalec striktno držati.

Slika 16: Grafični prikaz glavnih aktivnosti pristopa TOP-DOWN

Na sliki 16 so prikazane glavne tri aktivnosti pristopa TOP-DOWN. Kot smo že povedali, se vse začne s pripravo datoteke WSDL. Na osnovi WSDL se pripravi vmesnik, ki je že odvisen od končne platforme. Najpogosteje se ta vmesnik avtomatsko generira s pomočjo razvojnih orodij. Na koncu sledi še izvedba, v kateri razvijalec pripravi poslovno logiko (vnos v podatkovno bazo, različne logične funkcije, poslovna pravila) in jo poveže z vmesnikom.

Preslikovanje podatkov

Ko je enkrat vmesnik spletne storitve pripravljen, je treba izdelati poslovno logiko. Glede na to, da gre pri integracijskih postopkih v glavnem za prenos podatkov med sistemi, se po poslovni logiki pričakuje shranjevanje podatkov v ustrezne podatkovne strukture sistema. Ker se struktura podatkov izvornega sistema najpogosteje ne ujema popolnoma s strukturo podatkov ciljnega sistema, je treba pripraviti določeno preslikovalno logiko, ki bo na osnovi specifikacij preslikala izvorne podatke v ponorne. Spomnimo se primera prenosa podatkov o fizični osebi (poglavje 3.2.3). Lahko recimo naletimo na primer, ko imamo v izvornem sistemu ločena podatka za ime in priimek, v ciljnem sistemu pa sta ta dva podatka shranjena kot isti atribut tabele znotraj podatkovne baze. V tem primeru potrebujemo navodila, kako ta dva podatka združiti v enega (recimo, ali damo na prvo mesto ime ali priimek).

Za reševanje slednje problematike metodološki okvir predvideva uporabo t. i. preslikovalne tabele. Preslikovalna tabela je tabela, v kateri so na levi strani podatki iz izvornega sistema, na desni strani pa podatki ponornega sistema. Med podatki so natančno opisani algoritmi ali funkcije, ki opredeljujejo, kako so posamezni podatki povezani.

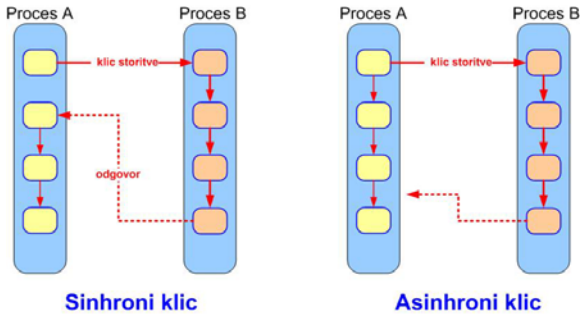
3.4.2 Implementacija poslovnega procesa

Kot smo že v uvodnih poglavjih metodološkega okvira povedali, temelji integracijski postopek na poslovnem procesu, ki ga v fazi analize modeliramo in v fazi izgradnje na osnovi modela tudi implementiramo ter preslikamo v proces BPEL. Proces BPEL je lahko predstavljen kot spletna storitev, do katere lahko dostopajo bodisi drugi poslovni procesi bodisi druge programske rešitve in sistemi zunaj in znotraj organizacije. Ker podroben opis jezika BPEL ni predmet te naloge, povejmo samo še to, da lahko poslovni proces, implementiran v jeziku BPEL, z uporabo orkestracije komunicira z drugimi spletnimi storitvami drugih sistemov.

Asinhroni in sinhroni proces

Proces BPEL je sestavljen iz množice aktivnosti, ki se izvajajo v natančno določenem vrstnem redu. Sam proces se drugim sistemom in procesom predstavlja kot spletna storitev. Z vidika, kako poslovni proces uporabljajo drugi procesi ali sistemi, je lahko proces BPEL implementiran kot asinhroni proces ali kot sinhroni proces. Za primer vzemimo procesa A, ki kliče proces B. V primeru, ko je proces B implementiran kot sinhroni proces, mora proces A po klicu čakati, da se proces B do konca izvede in šele nato dobi proces A odgovor procesa B. V tem času je proces A blokiran, ker čaka na odgovor in se ne more izvajati naprej. V primeru, ko je proces B implementiran kot asinhroni proces, pa proces A po klicu ne čaka na odgovor, ampak se lahko naprej izvaja. V tem primeru je treba zagotoviti mehanizme za posredovanje odgovora s procesa B do procesa A. Seveda, če je odgovor sploh potreben. Opisan primer prikazuje slika 17.

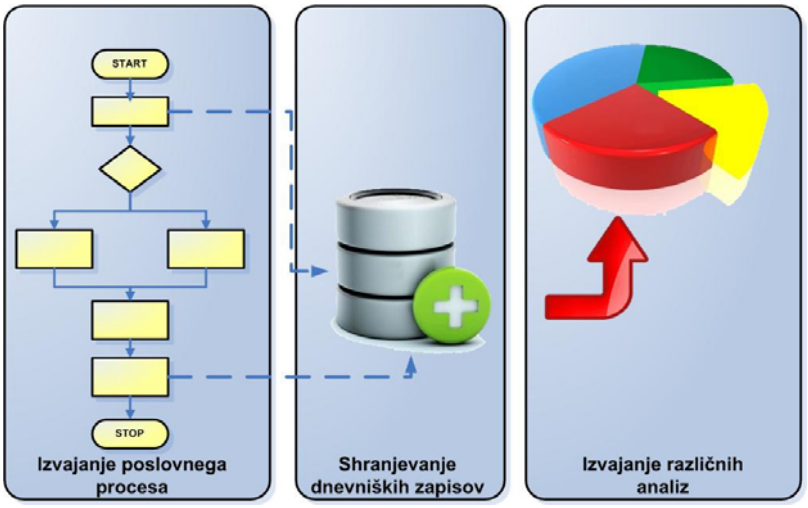
Slika 17: Grafični prikaz asinhronega in sinhronega klica



Spremljanje izvajanja procesa

V fazi implementacije je smiselno razmišljati tudi o sistemu, s pomočjo katerega bomo lahko v fazi postavitve in testiranja pregledovali posamezno izvajanje integracijskega postopka in analizirali uspešnosti posameznega prenosa. Nekateri sistemi, ki skrbijo za izvajanje poslovnih procesov (*IBM Process Server, Oracle BPEL Process Manager*), že imajo specifična orodja, s katerimi je mogoče pregledovati in slediti posameznemu izvajanju poslovnega procesa. V nasprotnem primeru je to treba posebej implementirati in predvsem proces BPEL je odličen element, znotraj katerega lahko tako spremljanje implementirano. To pomeni, da bomo v sam proces BPEL vključili določene aktivnosti, s pomočjo katerih bomo v posebne podatkovne strukture shranjevali statuse ter metapodatke, ki jih bomo lahko pozneje v fazi testiranj spremljali in na osnovi podatkov izvajali različne analize. V bistvu gre za beleženje dnevniških zapisov o izvajanju integracijskih postopkov. Grafična predstavitev opisane rešitve je predstavljena na sliki 18.

Slika 18: Grafični prikaz spremljanja izvajanja integracijskega postopka



V metodološkem okviru je koncept spremljanja izvajanja integracijskega postopka imenovan monitor SOA, ki definira zmožnost pregledovanja in spremljanja izvajanja integracijskih

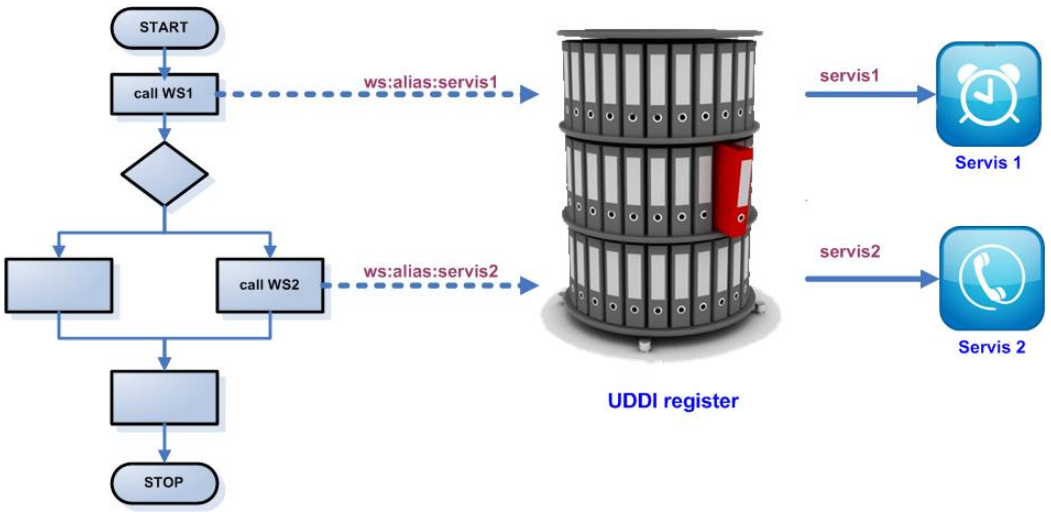
postopkov. Monitor SOA je funkcija in enotna rešitev, ki spremlja izvajanje vseh integracijskih postopkov, ki so bili implementirani v okviru vpeljave SOA v organizacijo.

3.4.3 Povezovanje integracijskih elementov in poslovnega procesa

Ko so proces BPEL in drugi integracijski elementi (spletne storitve in adapterji) implementirani, lahko te elemente povežemo med seboj Povezovanje poslovnega procesa BPEL z drugimi integracijskimi elementi z namenom izvajanja poslovnega procesa imenujemo orkestracija. Orkestracijo lahko najdemo in implementiramo tudi v drugih arhitekturnih pristopih, saj je vse, kar potrebujemo, povezava, s katero lahko izvedemo klic spletne storitve. Orkestracija je pri pristopu SOA še toliko bolj eksplicitna in poudarjena, saj gre pri integracijskih postopkih SOA za povezovanje različnih sistemov, najpogosteje prek spletnih storitev ali drugih integracijskih elementov. Povezovanje teh storitev je izvedeno s pomočjo poslovnega procesa, znotraj katerega se izvajajo aktivnosti, ki v definiranem vrstnem redu kličejo posamezne storitve ter si z zunanjimi sistemi izmenjujejo množico podatkov.

Pomembna lastnost registra UDDI, uporabljena znotraj metodološkega okvira, je povezana z orkestracijo storitev v posameznih fazah razvoja (razvoj, test in produkcija). Ker se naslov spletne storitve na razvojnem testnem in produkcijskem strežniku po navadi razlikuje, bi bilo treba pri vsakem prehodu iz ene faze v drugo skrbeti tudi za naslov storitve, na katero se v posamezni fazi sklicujemo. Če imamo ta naslov zapisan v procesu BPEL, pomeni, da je treba pred prehodom na testno okolje ta naslov ročno spremeniti. Ker pa želimo imel proces BPEL, ki je neodvisen od faze razvoja, nam pri tem lahko pomaga register UDDI.

Slika 19: Povezovanje procesa BPEL z registrom UDDI



Register UDDI, ki je sicer ločeno definiran za posamezno fazo razvoja (razvoj, test, produkcija), skrbi za vzdrževanje in management posamezne spletne storitve. Na sliki 19 je

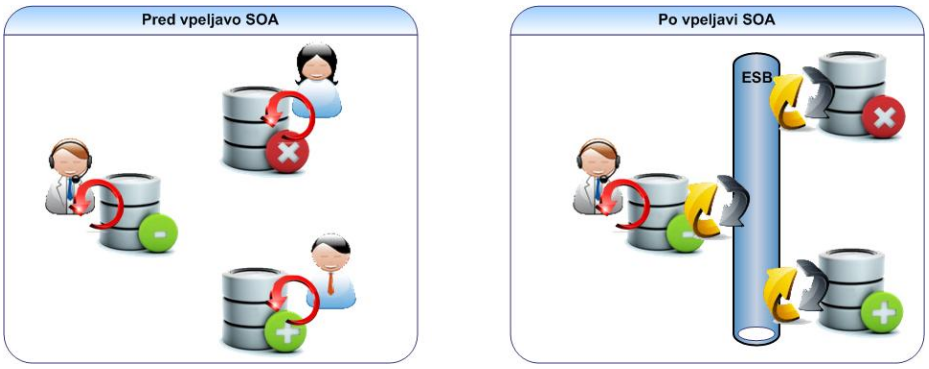
predstavljena rešitev, kjer proces BPEL pri klicu spletne storitve ne kliče direktno naslova storitve, ampak najprej pokliče register UDDI, kateremu posreduje ključ. Register UDDI na osnovi ključa identificira želeno storitev in procesu BPEL posreduje naslov spletne storitve.

3.5 Migracija podatkov

Predzadnja faza metodološkega okvira je faza migracije podatkov, ki jo tako kot fazo verifikacije koncepta ne izvajamo pri vsakem integracijskem postopku.

V poglavju 1.1.3 so bile opredeljene tri smernice razvoja informacijskih rešitev. Ena od treh smernic je tudi management podatkov, pri katerem je poudarek na postopkih in aktivnostih, ki so potrebne za uskladitev različnih, vsebinsko istih podatkov iz številnih podatkovnih struktur in okolij znotraj organizacije. Primer takih podatkov so podatki o poslovnih partnerjih, ki so znotraj organizacije najpogosteje shranjeni v različnih sistemih. Tako v organizacijah večkrat naletimo na to, da recimo kadrovska služba vzdržuje svoj šifrant poslovnih partnerjev, finančna služba svoj šifrant (z drugimi podatki) in prodaja spet svoj šifrant. Takšni podatki o poslovnih partnerjih so nekonsistentni in vzdrževanje teh podatkov je praktično nemogoče. S pristopom SOA in z ustreznim integracijskim postopkom lahko takšno situacijo bistveno izboljšamo. V tem primeru je faza migracije podatkov neizogibna faza.

Slika 20: Stanje pred vpeljavo in po vpeljavi pristopa SOA



Slika 20 prikazuje stanje pred in po vpeljavi pristopa SOA. Pred vpeljavo se dogaja to, da različne službe skrbijo za svoje podatke, ki pa so po vsebini enaki (primer poslovnih partnerjev). Z vpeljavo SOA in izvedbo faze migracije podatkov pa poskrbimo, da se podatki upravljajo na enem mestu, ustrezni mehanizmi (ESB) pa poskrbijo za vzdrževanje in osveževanje podatkov drugih sistemov.

3.5.1 Analiza obstoječih podatkovnih struktur

Preden začnemo urejati samo vsebino (podatke), je treba najprej analizirati podatkovne strukture znotraj posameznih sistemov. Moramo ugotoviti, kateri so tisti podatki (atributi), ki so skupni (na primer davčna številka poslovnega partnerja), in kateri se v sami strukturi razlikujejo (recimo naslov poslovnega partnerja). Glede na to, da cilj migracije podatkov ni kakršnokoli spreminjanje podatkovnih struktur v sistemih, je treba natančno definirati, kako se bodo med samo migracijo podatki preslikovali v ciljno podatkovno strukturo. Ta struktura bo po migraciji hranila vse podatke ter bo edina struktura za vzdrževanje in management podatkov. Vsa pravila preslikovanja podatkov so znotraj metodološkega okvira opredeljena v specifikaciji preslikovanja.

Specifikacija preslikovanja je dokument, v katerem so opredeljena vsa pravila, kako se podatki iz izvornega sistema preslikajo v strukture ponornega sistema. Zaradi enostavnosti in ponovne uporabljivosti je specifikacija podatkov najpogosteje implementirana kar v obliki različnih poizvedb in ukazov nad podatkovnimi strukturami. Specifikacijo preslikovanja si lahko predstavljamo kot podprogram, ki ga lahko večkrat izvedemo nad podatki in ki poskrbi za migracijo podatkov po definiranih pravilih v ciljni sistem. Osnova za pripravo dobre specifikacije preslikovanja je poznavanje vsebine podatkovnih struktur izvornega in ponornega sistema. Zaradi tega je v fazi migracije podatkov nujno potrebno tudi sodelovanje poslovnih analitikov.

3.5.2 Definiranje cilja migracije podatkov

Še preden začnemo s pripravo specifikacije preslikovanja, je treba definirati, kakšen je cilj migracije podatkov. Identificirati moramo vse sisteme, ki bodo v migracijo vključeni, in v končni fazi je treba definirati tudi ponorni sistem. Ko govorimo o ponornem sistemu, imamo dve možnosti:

- **podatke migriramo v novi sistem**: V tem primeru je ponorni sistem neka nova programska rešitev, katere namen bo tudi vzdrževanje migriranih podatkov. Pri tej možnosti lahko pridemo na osnovi struktur podatkov izvornih sistemov do podatkovne strukture ciljnega sistema, ki vsebuje presek strukture podatkov iz izvornih sistemov.
- **podatke migriramo v obstoječi sistem:** V tem primeru je ponorni sistem kar eden od izvornih sistemov. Pri tej možnosti nimamo svobode pri oblikovanju strukture ciljnega sistema, ampak je treba še toliko bolj natančno opredeliti posamezna pravila preslikovanja. Če hranimo v ponornem sistemu naslov v enem stolpcu tabele v izvornem sistemu pa imamo za občino, naslov in ulico svoj stolpec, je treba definirati, kako se naslov iz izvornega sistema preslika v ponor.

Faza migracije podatkov se znotraj metodološkega okvira izvaja po vnaprej predpisanih korakih, ki pa se zaradi večje kakovosti podatkov lahko večkrat ponovijo. Koraki migracije so naslednji:

- **izvoz podatkov iz izvora:** Podatke je treba najprej izvoziti iz posameznih sistemov. Glede na to, da so izvorni sistemi različni, moramo tukaj upoštevati tudi različne izvorne strukture (nepovezane datoteke, različni formati izvozov, Excel datoteke …).
- **kontrolne skripte:** Včasih je smiselno pripraviti tudi kontrolne skripte, ki preverijo vsebino izvoženih podatkov. Tukaj ne govorimo o vsebini podatkov, ampak o vsebini izvoženih datotek. Včasih se na primer lahko zgodi, da določena izvozna datoteka vsebuje neke čudne znake, ki jih je treba preslikati v pravilne ali pa te znake izbrisati.
- **polnjenje delovnih tabel:** Izvožene podatke se naprej naloži v t. i. delovne tabele, znotraj katerih se bo migracija podatkov tudi izvedla. Ker se ta postopek lahko večkrat ponovi, je smiselno pripraviti določene avtomatizme, ki poskrbijo za polnjenje izvoženih podatkov delovne tabele.
- **izvajanje migracije:** Sledi korak, ko izvedemo migracijo podatkov, ki na osnovi opredeljene specifikacije preslikovanja podatke preslika v ustrezne strukture znotraj delovnih tabel.
- **izvoz podatkov iz delovnih tabel:** Migrirane podatke spet izvozimo iz delovnih tabel v določene strukture, s pomočjo katerih bomo napolnili tabele ciljnega sistema.
- **polnjenje ciljnega sistema:** Na osnovi izvoženih tabel napolnimo ustrezne strukture ciljnega sistema migracije.

Migracija podatkov ni enostavna faza in za uspešno izvedbo zahteva veliko veščin in poznavanja dela z različnimi podatkovnimi bazami. Ker integracijski postopki temeljijo na izmenjavanju podatkov med različnimi sistemi, je najpogosteje migracija podatkov nujna in zaradi tega sta lahko uspešnost in kakovost integracijskega postopka odvisni prav od same migracije.

3.6 Postavitev

Zadnja faza metodološkega okvira se nanaša na postavitev integracijskega postopka najprej na testno okolje in pozneje, po uspešni izvedbi testiranja, še na produkcijsko okolje. V fazi postavitve je zelo pomembno testiranje in prav testiranju se znotraj te faze posveča največ pozornosti. Testiranje lahko pokaže različne rezultate.

Slika 21: Grafični prikaz aktivnosti faze postavitve

Kot je prikazano na sliki 21, se celoten postopek začne s pripravo testnih scenarijev in nato sledi sama izvedba testiranja, ki je spet razdeljena na določene korake. V primeru uspešne izvedbe testiranja in pozitivnih rezultatov nimamo nobenih ovir za postavitev integracijskega postopka v produkcijsko okolje. V primeru negativnih rezultatov pa je treba najprej odpraviti vse napake in popravke ter nato testiranje znova izvesti. To med drugim narekuje daljši čas izvedbe postopka in mogoče tudi podaljšanje projekta. Tega pa si seveda ne želimo.

3.6.1 Priprava testnih scenarijev

V nasprotju s klasičnim razvojem informacijskih sistemov je pri integracijskih postopkih testiranje malo bolj zapleteno ali, bolje rečeno, bolj porazdeljeno. Vpeljava SOA najpogosteje narekuje tudi določene spremembe obstoječih postopkov znotraj organizacije (prenova poslovnih procesov). Zaradi slednjih lahko prihaja tudi do sprememb in dopolnitev obstoječih sistemov, ki skrbijo za poslovanje in podpirajo poslovne funkcije znotraj organizacije. Zaradi tega je testiranje nujno potrebno razširiti tudi na sisteme, ki so v integracijski postopek vključeni. Metodološki okvir opredeljuje tri vrste testiranja:

- **integracijski test:** Namen integracijskega testa je preverjanje izvajanja integracijskega postopka. Uspešno izveden integracijski test z uspešnimi rezultati pomeni, da se podatki po specificiranih pravilih pravilno prenašajo med sistemi.
- **aplikativni test:** Namen aplikativnega testa je testiranje funkcionalnosti sistemov, ki so vključeni v integracijski postopek. Uspešno izveden aplikativni test z uspešnimi rezultati pomeni, da sistemi delujejo v skladu s specifikacijami in da izvajanje integracijskega postopka ne vpliva na njihovo delovanje.
- **tehnični test:** S tehničnim testom preverjamo uspešnost migracije podatkov. To pomeni, da tehnično testiranje izvajamo samo v primeru, ko smo izvedli tudi fazo migracije podatkov. Uspešno izveden tehnični test z uspešnimi rezultati pomeni, da so bili podatki pravilno migrirani v ciljni sistem.

Glede na to, da je izvedba testiranja zelo pomembna aktivnost, je nujno sam postopek testiranja čim bolj podrobno opredeliti in opisati. S tem se bomo izognili temu, da bi bil postopek (ne glede na vrsto testiranja) nepravilno ali napačno stestiran. V ta namen si pomagamo s t. i. testnimi scenariji. Testne scenarije pripravijo poslovni analitiki, ki dobro poznajo izvajanje posameznih postopkov. Testni scenarij mora opredeliti:

- **osnovne podatke scenarija:** Opredeliti je treba številko scenarija, datum testiranja in kdo bo test izvedel.
- **postopek testiranja:** Scenarij mora vsebovati natančen opis postopka, ki ga je treba stestirati v okviru scenarija.
- **rezultate testiranja:** Vsako testiranje zahteva tudi poročilo o uspešnosti izvedbe testiranja in o rezultatih samega testa. Vsi komentarji in napake so osnova za morebitne dopolnitve na sistemih.

Testni scenariji so pomembni zlasti z dveh vidikov. Scenarij opredeljuje določeno nalogo, ki jo mora oseba, ki testira, sprejeti in je za izvedbo te naloge tudi odgovorna. Drugi vidik pa je povezan z odpravljanjem napak. Določene napake pri izvedbi se pokažejo šele v fazi testiranja. Rezultati testiranja na osnovi testnih scenarijev so lahko odlično sredstvo za iskanje vzroka napake (glede na to, da je postopek opredeljen) in odpravljanje napake.

3.6.2 Vključevanje odgovornih oseb in izvedba testiranja

Posebno poglavje je v okviru metodološkega okvira rezervirano prav za izvedbo testiranja. Namen poglavja ni opredeliti, kako izvesti tehnično testiranje, ampak kako zagotoviti, da bo testiranje v celoti izvedeno.

Kot smo povedali že v prejšnjem poglavju, je testiranje znotraj pristopa SOA širše opredeljeno in je treba samo testiranje razširiti na več sistemov znotraj organizacije. To pomeni, da moramo v samo testiranje vključiti tudi ljudi (uporabniki) iz posameznih oddelkov, ki postopke sistema dobro poznajo. Aktivno testiranje je zaradi tega treba začeti najprej z opredelitvijo posameznih uporabnikov, ki bodo določen sklop testiranja tudi izvedli. Izberemo najmanj 2 osebi za določen sistem ali sklop testiranja. Upoštevati moramo njihovo znanje in predvsem njihovo motivacijo za tovrstno delo. Testiranje bo neučinkovito, če bo test izvajal nekdo, ki testiranje močno sovraži. Vse, kar bomo dosegli, bo nepopolno izveden test. Ko so enkrat uporabniki izbrani, je treba uporabnike seznaniti s celotno vsebino in jim predstaviti testne scenarije, ki jih bodo uporabljali pri testiranju. Uporabniki morajo biti poleg tega seznanjeni z odgovornostjo, ki jo pri tem nosijo, in se morajo odgovornosti tudi zavedati. Sledi sama izvedba testiranja in pri tem je zelo pomembno, da imajo uporabniki natančen načrt, kdaj mora biti testiranje opravljeno in do kdaj morajo vrniti izpolnjene testne scenarije. Načrt testiranja je še posebno pomemben v primeru, ko je treba pri testiranju upoštevati določeno zaporedje izvajanja testiranja posameznih sistemov. Primer je recimo integracija dveh sistemov A in B, pri čemer moramo najprej izvesti aplikativni test sistema A, nato

integracijski test (podatki se prenesejo v sistem B) in nato spet aplikativni test sistema B. Ko je enkrat testiranje izvedeno in smo s strani uporabnikov pridobili vse izpolnjene testne scenarije, lahko testiranje še ocenimo. Na osnovi ocene se odločimo, ali je integracijski postopek pripravljen za prehod v produkcijo.

Faze metodološkega okvira, opisane v poglavju, imajo to skupno značilnost, da morajo biti dobro načrtovane (časovna opredelitev) in da morajo biti natančno ter v celoti izvedene (razen seveda verifikacije koncepta in migracije podatkov, ki niso obvezne faze). Uspešna vpeljava SOA v organizacijo na osnovi metodološkega okvira pa ni odvisna samo od posameznih faz, tukaj je treba upoštevati še organizacijski vidik oziroma vidik ali koncept, ki ga v svetu SOA imenujmo tudi management.

4. OVREDNOTENJE

Primer, s katerim bom metodološki okvir ovrednotil, je projekt vpeljave SOA v podjetje Mladinska knjiga Založba. Podjetje uporablja številne informacijske rešitve. Za nekatere skrbi razvojna skupina znotraj podjetja (lasten razvoj). Obstajajo pa tudi informacijske rešitve, za katere skrbijo zunanji izvajalci. Zaradi potrebe po informatizaciji poslovnih procesov se je podjetje odločilo za vpeljavo SOA.

Cilja ovrednotenja metodološkega okvira sta predvsem dva:

- Na praktičnem primeru predstaviti posamezne faze oblikovanega metodološkega okvira skupaj z izdelki, ki so v posamezni fazi nastali.
- Podati nekaj predlogov za izboljšanje metodološkega okvira. Metodološki okvir je nastal na osnovi lastnih izkušenj pri vodenju in izvajanju projektov, ki temeljijo na pristopu SOA. Glede na to, da se določeni projekti še vedno izvajajo, neprestano prihaja do novih ugotovitev in idej, ki vplivajo na strukturo in vsebino metodološkega okvira in so kandidati za izboljšanje okvira samega.

4.1 Uporaba metodološkega okvira na primeru

Metodološki okvir bom predstavil na praktičnem primeru integracijskega postopka urejanja poslovnih partnerjev (v nadaljevanju komitenti) v podjetju Mladinska knjiga Založba.

4.1.1 Analiza postopka

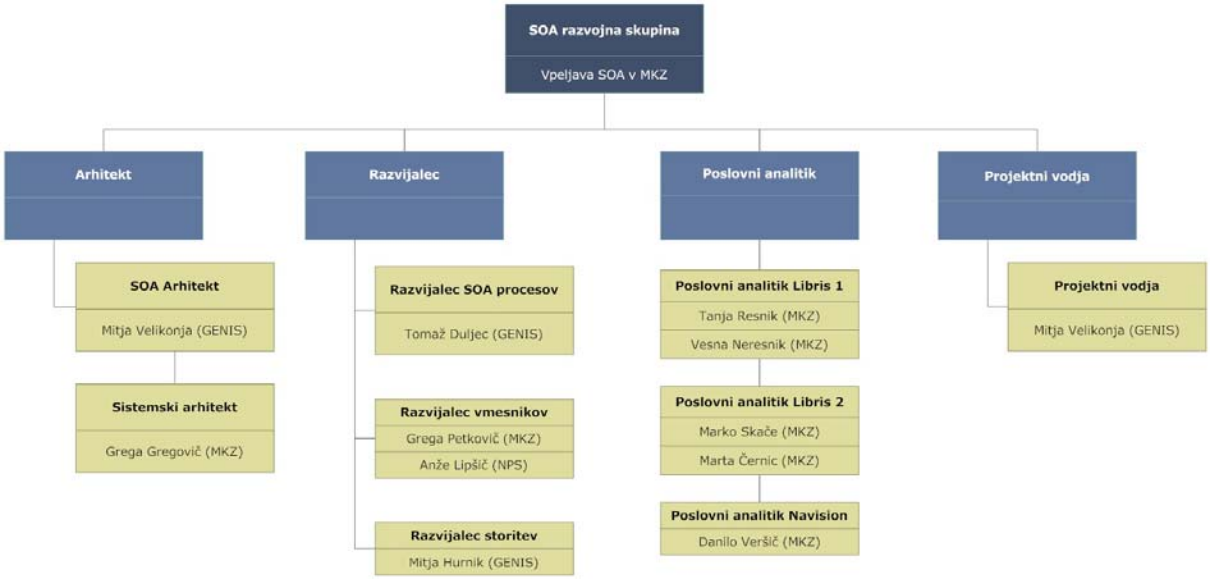
Oblikovanje razvojne skupine SOA

V okviru vpeljave SOA v podjetje Mladinska knjiga in izvedbe integracijskega postopka urejanja komitentov je bila oblikovana razvojna skupina, ki opredeljuje profile ljudi iz naslednjih inštitucij oziroma podjetij, ki so vključena v integracijski postopek:

- Mladinska knjiga Založba,
- podjetje NPS – razvoj in podpora celovitega informacijskega sistema Navision,
- podjetje Genis – vpeljava SOA.

Organizacijska shema projektne skupine je prikazana na sliki 22.

Slika 22: Organizacijske shema integracijskega postopka urejanja komitentov



Analiza obstoječih poslovnih procesov

Podjetje Mladinska knjiga hrani glavne podatke o komitentih v treh različnih sistemih, ki jih med drugim tudi ločeno vzdržuje. Ti sistemi podpirajo glavne poslovne funkcije znotraj podjetja:

- **sistem Libris 1:** logistika založništva MKZ,
- **sistem Libris 2:** prodaja (pravne osebe in fizične osebe) in nabava izključno za MKZ,
- **sistem Libris Navision:** knjigovodstvo (saldakonti kupcev, dobaviteljev, glavna knjiga …).

Proces, ki ga želimo informatizirati, omogoča urejanje podatkov o komitentih na enem mestu (sistemu). V proces so vključuje vsi trije sistemi (Libris 1, Libris 2 in Navision), pri čemer je Libris 2 izvor podatkov, Libris 1 in Navision pa ponora. Vsi podatki o komitentih se urejajo v sistemu Libris 2. Ko se določen podatek spremeni ali pride do vnosa novega komitenta, se podatki prek arhitekture SOA posredujejo na vmesnika Libris 1 in Navision. Vmesnika podatke sprejmeta in jih vneseta v pripadajoče podatkovne strukture. Za prenos podatkov prek arhitekture SOA poskrbi proces BPEL, ki najprej ugotovi, v kateri sistem se podatki prenesejo (ne prenašajo se vedno v oba sistema), in shrani posamezne vmesne podatke o prenosu v tabele dnevniških zapisov.

V prilogah je predstavljen poslovni model urejanja podatkov o komitentih. Sistemi, ki so vključeni v postopek, so predstavljeni kot vodoravne steze (angl. *lanes*), v katerih so pripadajoče aktivnosti. Na takšen način lahko hitro ugotovimo, znotraj katerega sistema se določena aktivnost izvede. Postopek urejanja komitentov se začne z vnosom ali spremembo podatkov o komitentu. Aktivnost je definirana kot človeška interakcija (angl. *human task*), ker vnos ali spremembo podatka izvede uporabnik. Po spremembi ali vnosu se podatki samodejno napolnijo v ustrezne strukture in spletni servis poskrbi za prenos podatkov na vodilo SOA.

Logika znotraj vodila ugotovi, v kateri sistem so podatki namenjeni, in na osnovi te ugotovitve pokliče ustrezne servise na vmesnikih obeh sistemov. Posamezni vmesniki poskrbijo za vnos podatkov v končne sisteme.

Identifikacija integracijskih postopkov

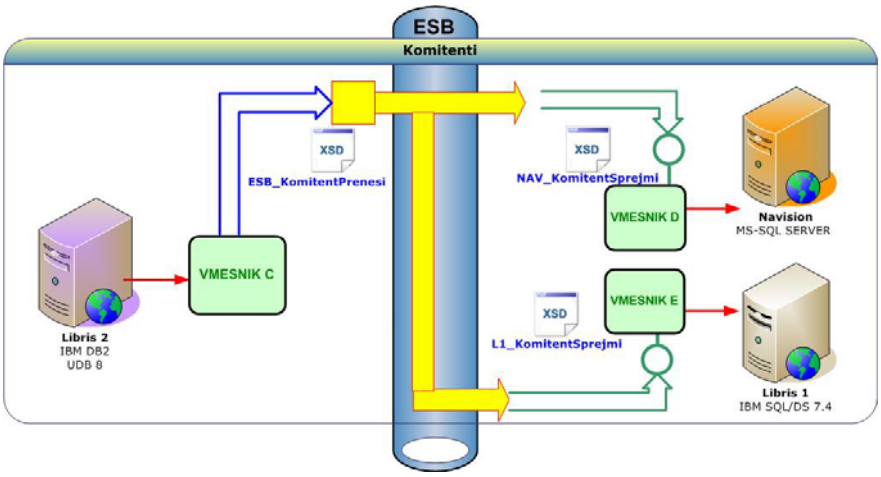
Prikazan poslovni model ima naslednje značilnosti, ki so predstavljene v tabeli 2.

Tabela 2: Karakteristike poslovnega modela postopka urejanja komitentov

Izvorni sistem	Libris 2
Ponorni sistem	Libris 1 in Navision
Povratna informacija	Povratna informacija o uspešnosti prenosa ni potrebna.
Način prenosa	Asinhroni prenos
Groba struktura podatkov	KomitentType

Lastnosti integracijskega postopka so prikazane v spodnji integracijski shemi na sliki 23.

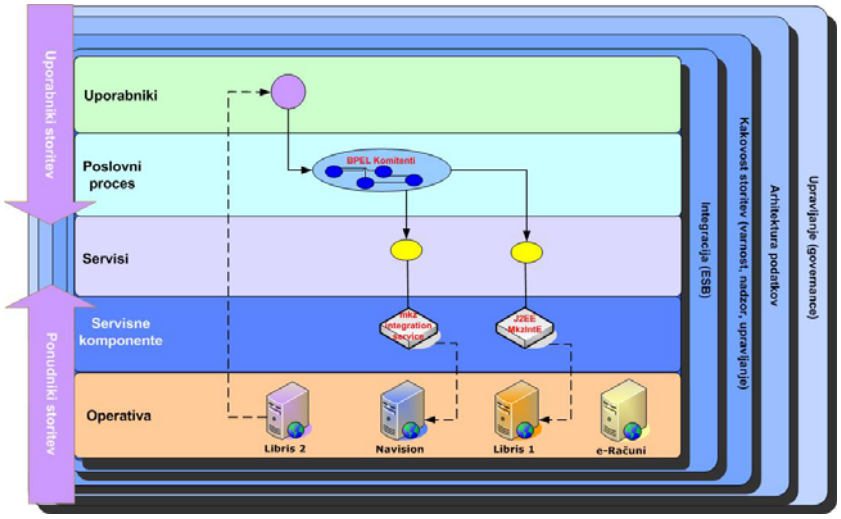
Slika 23: Integracijska shema postopka urejanja komitentov



Identifikacija integracijskih elementov

Kot je prikazano na sliki 24, predstavljajo raven operative obstoječi sistemi Libris 1, Libris 2 in Navision. Na ravni sistemov Libris 1 in Navision (ponora) sta definirani spletni komponenti s pripadajočima spletnima storitvama za vnos podatkov. Definiran je poslovni proces BPEL_Komitenti, ki na osnovi orkestracije posameznih storitev zagotavlja prenos podatkov o komitentih v sistema Libris 1 in Navision. Poslovni proces kot storitev uporablja sistem Libris 2, ki podatke o komitentih pošilja na poslovno vodilo.

Slika 24: Shema integracijskih plasti postopka urejanja komitentov

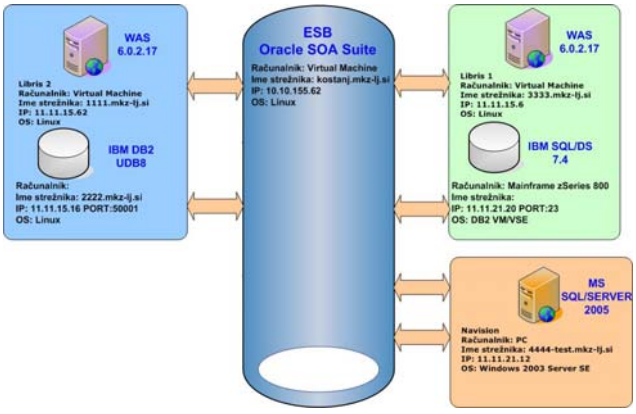


4.1.2 Načrtovanje in specifikacije

Priprava arhitekturnega diagrama

Arhitekturni diagram na sliki 25 predstavlja podrobnosti sistemov, ki so vključeni v integracijski postopek. Primer prestavlja tri sisteme in morebitne relacijske podatkovne baze. Arhitekturni diagram pripravimo za vsako fazo razvoja (razvoj test, produkcija) posebej, saj se sistemi v posamezni fazi razlikujejo. Govorimo o istih sistemih, ki se lahko izvajajo na drugih platformah.

Slika 25: Arhitekturni diagram postopka urejanja komitentov



Identifikacija in priprava specifikacij spletnih storitev

Specifikacije spletnih storitev so podane v obliki dokumenta WSDL. Za vsako spletno storitev je pripravljen ustrezen dokument WSDL. Integracijski postopek uporablja za prenos podatkov

dve spletni storitvi, ki sta razviti na ravni vmesnika C in vmesnika D (glej integracijsko shemo na sliki 23).

Tabela 3: Seznam specifikacij spletnih storitev postopka urejanja komitentov

Naziv	**Vnos podatkov v Navision**
Vmesnik	Vmesnik D
Datoteka WSDL	mkzintegrationservice.wsdl
Shema XSD	mkzintegrationservice0.xsd mkzintegrationservice1.xsd mkzintegrationservice2.xsd

Naziv	**Vnos podatkov v Libris 1**
Vmesnik	Vmesnik E
Datoteka WSDL	L1SprejemKomitentov.wsdl
Shema XSD	L1_KomitentSprejmi.xsd

Vsaka specifikacija spletne storitve definira tudi ustrezno datoteko XSD. Znotraj datoteke XSD je opisana struktura podatkov, ki jih metode storitve uporabljajo kot parameter. Primer specifikacij je predstavljen v tabeli 3.

Specifikacija podatkovnih struktur

Specifikacije podatkovnih struktur so podane v obliki dokumentov XSD za posamezno sporočilo, ki se prenaša prek integracijskega postopka. Za primer integracije podatkov o komitentih imamo samo eno glavno strukturo (v seznamu imenovana ESB_GlobalTypes) podatkov o poslovnem partnerju, ki jo vključujejo posamezni dokumenti XSD na posameznem integracijskem elementu.

Dokumenti XSD so razdeljeni v tri sklope:

- ***glava*** – vsebuje podatke o datumu prenosa in identifikacijsko številko prenosa;
- ***podatki o ciljnem sistemu*** – za vsako sporočilo definiramo, v kateri sistem naj se podatki pošljejo;
- ***telo*** – glavni podatki o komitentih.

V tabeli 4 so opisani dokumenti XSD za integracijski postopek.

Tabela 4: Seznam shem podatkov postopka urejanja komitentov

Shema	ESB_KomitentPrenesi
Uporaba	Shema se uporablja za prenos podatkov o komitentu iz vmesnika C na vodilo SOA. Shema vključuje shemo ESB_GlobalTypes.

Shema	NAV_KomitentSprejmi
Uporaba	Shema se uporablja za prenos podatkov o komitentu iz vodila SOA na vmesnik D (sistem Navision). Shema vključuje shemo ESB_GlobalTypes.

Shema	L1_KomitentSprejmi
Uporaba	Shema se uporablja za prenos podatkov o komitentu iz vodila SOA na vmesnik E (sistem Libris 1). Shema vključuje shemo ESB_GlobalTypes.

Shema	ESB_GlobalTypes
Uporaba	Shema vsebuje vse skupne značke, ki se uporabljajo v integracijskem postopku KOMITENTI.

Definiranje nalog in odgovornosti

Naloge in odgovornosti so razdeljene glede na integracijski element, ki ga je treba izdelati. Primeri nalog so opisani v tabeli 5.

Tabela 5: Seznam nalog in odgovornosti za izvedbo integracijskega elementa

Integ. element	**Vmesnik C**
Naziv naloge	**Priprava agenta za pripravo podatkov iz L2**
Odgovoren	**MKZ**
Rok izvedbe	**10. 10. 2008**
Opis naloge	

Aktivnost lahko razdelimo v dve fazi. V prvi fazi je treba na programski rešitvi L2 zagotoviti, da se ob vsaki spremembi komitenta (vnos novega komitenta in popravljanje obstoječega) sproži postopek prenosa.

Druga faza pa zahteva pripravo podatkov o komitentu (podatke o komitentu je treba prebrati iz podatkovne baze Libris 2), mapiranje podatkov v vnaprej pripravljene strukture Java (Java razredi iz knjižnice) ter klic prenosa s pomočjo knjižnice.

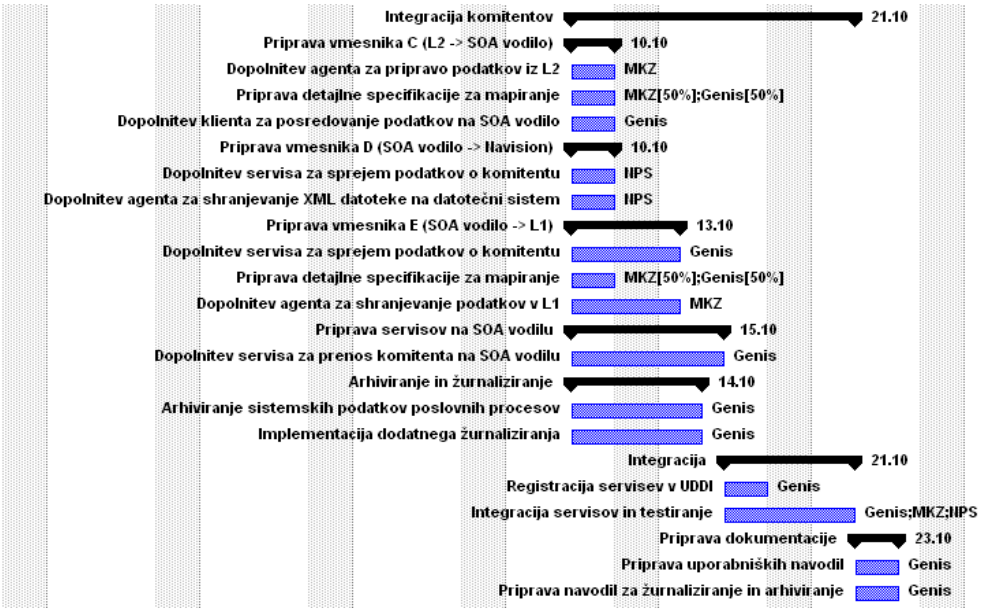
Integ. element	**Vmesnik C**
Naziv naloge	**Priprava klienta za posredovanje podatkov na vodilo SOA**
Odgovoren	**Genis**
Rok izvedbe	**10. 10. 2008**
Opis naloge	

Za potrebe posredovanja podatkov o komitentu bo Genis pripravil knjižnico Java, ki bo poleg klienta za posredovanje podatkov na vodilo vsebovala tudi potrebe razrede Java, v katere bo moral MKZ napolniti podatke o komitentu. Knjižnica bo na MKZ posredovana v obliki datoteke ».jar«.

Planiranje

Podobno kot so porazdeljene naloge in odgovornosti pri izvedbi nalog, je razdeljen tudi projektni plan, prikazan na diagramu Gantt (slika 26). Diagram natančno opredeljuje datum začetka in konca posamezne aktivnosti ter ljudi (podjetja), ki so odgovorni za izvedbo aktivnosti.

Slika 26: Projektni plan postopka urejanja komitentov



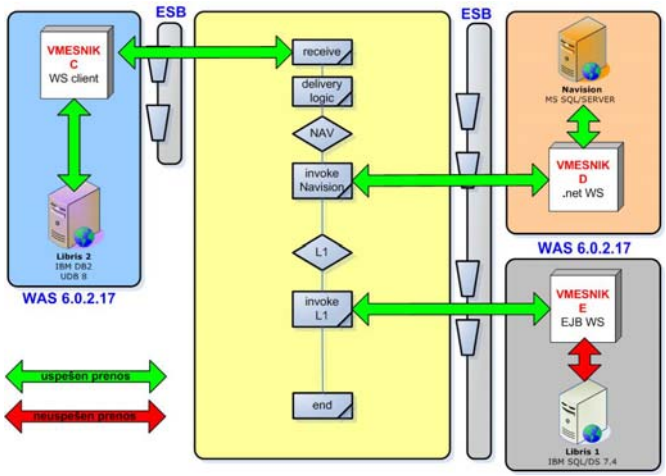
Aktivnosti projektnega plana ne smemo mešati z nalogami, ki so opredeljene v zgornji tabeli. V projektnem planu so definirane posamezne aktivnosti tako za fazo izvedbe kot tudi aktivnosti, ki se nanašajo na organizacijsko plat. Naloge iz zgornjih tabel pa so bolj tehnične in definirano je, kaj je treba narediti. V določenih primerih pa so naloge podane tudi kot aktivnosti projektnega plana.

4.1.3 Verifikacija koncepta

Aktivnost verifikacije koncepta je bila izvedena, ker je bil to prvi integracijski postopek, ki je bil razvit v podjetju na osnovi pristopa SOA. Integracijski postopek, ki je bil pripravljen za verifikacijo koncepta, temelji na specifikacijah, pripravljenih v fazi analize in načrtovanja.

Rezultat faze verifikacije koncepta je poročilo, katerega glavna elementa sta shema verifikacije koncepta in ugotovitve ter problemi pri izvajanju postopka skupaj s predlogi za dopolnitev.

Slika 27: Shema verifikacije koncepta postopka urejanja komitentov



Na osnovi sheme (slika 27), na kateri so prikazani rezultati izvajanja postopka, je mogoče hitro ugotoviti, kjer se je pri verifikaciji koncepta zalomilo in kateremu delu integracijskega postopka je treba posvetiti več časa.

Ugotovitve in problemi

Primer izdelka iz faze verifikacije koncepta je tudi seznam ugotovitev in problemov, ki so nastali pri izvajanju postopka. Predlogi se bodo lahko pozneje upoštevali v fazi izgradnje. V tabeli 6 je predstavljenih nekaj primerov.

Tabela 6: Seznam ugotovitev in problemov po izvedbi faze verifikacije koncepta

Polja z omejitvami na vnosu
Ugotovili smo, da lahko prihaja do težav v primeru, ko znotraj sheme XSD uporabljamo omejitve na vnosu (enumeration). Do težave pride, ker .NET drugače pretvori številska enumeration polja kot pa Java.
Predlog
Omejitve na poljih bi bilo treba odstraniti in jih ne uporabljati.
Vsebina in obveznost podatkov
V primeru, ko pri pošiljanju komitenta na vodilo SOA ne vnesemo vseh obveznih podatkov, prihaja do napake pri pošiljanju, ko se preverja veljavnost sheme. Na osnovi napake je težko ugotoviti (razen iz log-a), kje je prišlo do napake.
Predlog
Na shemi XSD bo treba natančno definirati, kateri podatki so pri vnosu komitenta obvezni za vnos in kateri ne. Vmesnik D bo moral pred pošiljanjem na vodilo SOA preverjati, ali so bili vsi obvezni podatki vneseni. V nasprotnem primeru bo prišlo do napake pri pošiljanju.

4.1.4 Izgradnja

Izvedba verifikacije koncepta je pokazala določene pomanjkljivosti integracijskega postopka in pripadajočih specifikacij, ki so bile odpravljene.

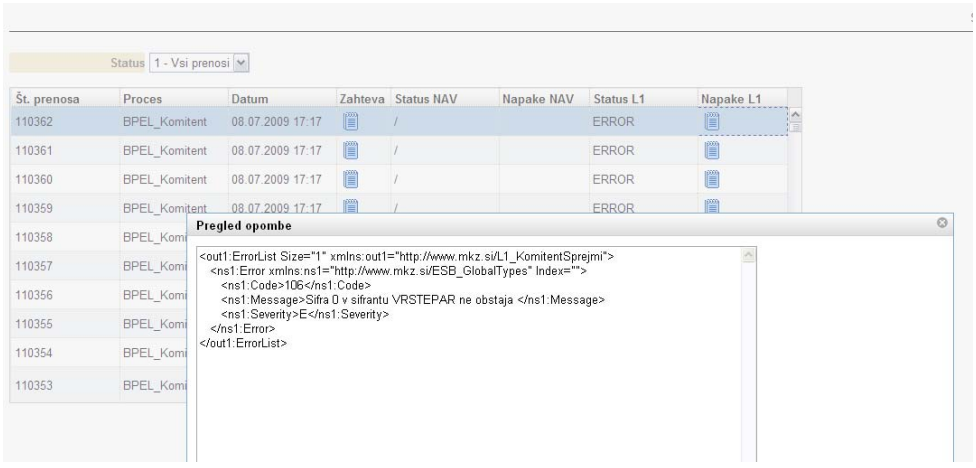
Implementacija integracijskih elementov
V fazi izgradnje sta bila najprej izdelana vmesnika na osnovi specifikacij (dokumentov WSDL) in pristopa od zgoraj navzdol. Rezultat pristopa TOP-DOWN je bilo ogrodje vmesnika D in E s pripadajočima spletnima storitvama. Treba je bilo še razviti logiko za vnos podatkov v sisteme Libris 1 in Navision. V ta namen je bila pripravljena preslikovalna tabela, katere primer za prvih nekaj polj je prestavljen v prilogah.

Implementacija poslovnega procesa
Poslovni proces integracijskega postopka je bil implementiran kot asinhroni poslovni proces iz naslednjih razlogov:

- Mehanizem pošiljanja podatkov iz sistema Libris 2 se sproži ob vsaki spremembi podatkov o komitentu. Zaradi tega sistem ne sme čakati, da se podatki najprej pošljejo v ustrezne sisteme, in šele nato omogočiti uporabniku nadaljevanje dela.
- Ni potrebe po posredovanju odgovora o pravilnosti shranjevanja podatkov v ponorni sistem. V primeru napake poslovno vodilo poskrbi za obveščanje skrbnika podatkov.

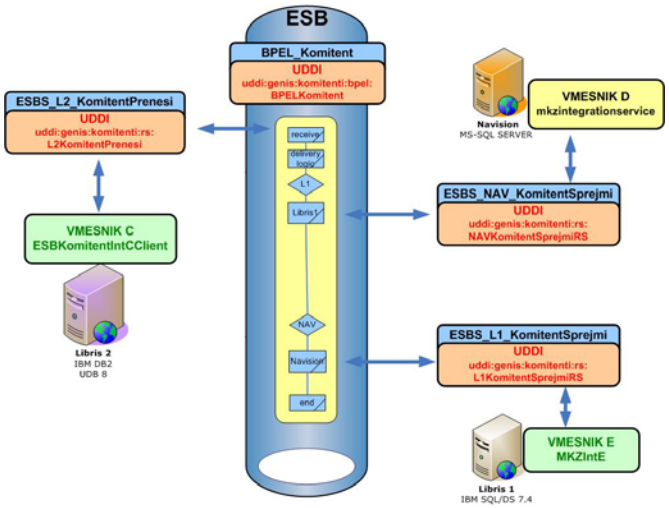
Za spremljanje izvajanja procesa skrbi sistem monitor SOA, ki omogoča skrbniku ugotavljanje napak pri prenosu in vnovično izvajanja samega postopka. Na sliki 28 je predstavljena zaslonska maska programske rešitve monitor SOA, kjer je prikazana napaka, ki je nastala pri prenosu podatkov o komitentu. Iz opisa napake lahko skrbnik hitro ugotovi, zakaj je prišlo do napake, in po potrebi znova sproži prenos.

Slika 28: Primer uporabniškega vmesnika sistema monitor SOA



Povezovanje integracijskih elementov in poslovnega procesa
Povezovanje integracijski elementov in poslovnega procesa je izvedeno s pomočjo registra UDDI. Na sliki 29 je razvidno, kateri ključi se uporabljajo za posamezen integracijski element. Na osnovi ključev lahko register UDDI pridobi naslov posamezne spletne storitve znotraj poslovnega procesa.

Slika 29: Grafični prikaz uporabe ključev znotraj registra UDDI



4.1.5 Migracija podatkov

Analiza obstoječih podatkovnih struktur
Pogoj za izvajanje integracijskega postopka so urejeni podatki o komitentih v vseh treh sistemih. To je doseženo na osnovi izvedbe faze migracije podatkov.

Za potrebe migracije podatkov so bili najprej analizirani podatkovni modeli sistemov Libris 1 in Libris 2. Za Navision analiza ni bila potrebna, ker so se podatki o komitentih v Navision predhodno polnili iz sistema Libris 1.

Definiranje cilja migracije podatkov
Namen migracije podatkov je enotna točka za vnos in spreminjanje podatkov o komitentu in uporaba integracijskega postopka za prenos podatkov v druge sisteme. Glede na to, da je migracija podatkov pogoj za uspešno izvajanje integracijskega postopka, je bilo treba fazo migracije temeljito definirati in izvesti.

V prilogah je predstavljen seznam korakov faze migracije podatkov. Naštetih je nekaj začetnih točk za posamezen korak migracije podatkov. Koraki so tako optimizirani, da je večkratno izvajanje migracije podatkov (zaradi ugotovljenih napak) enostavno in hitro.

4.1.6 Postavitev

Priprava testnih scenarijev
Testiranje integracijskega postopka je razdeljeno v tri sklope. Za posamezen sklop obstaja večje število posameznih testnih scenarijev (predstavljeno v tabeli 7).

Tabela 7: Seznam različnih sklopov testiranja in število posameznih testnih scenarijev

Sklop	Opis	Št. scenarijev
Aplikativni test	testiranje aplikacije Libris 1 testiranje aplikacije Libris 2 testiranje aplikacije Navision (ni potrebno)	8 15 0
Integracijski test	testiranje prenosa podatkov iz sistema Libris 2 v Libris 1 in Navision	22
Tehnični test	testiranje migracije podatkov	1

Primer testnega scenarija je prikazan v prilogah.

Vključevanje odgovornih oseb in izvedba testiranja
Pred izvedbo testiranja (vsi trije sklopi) so bile najprej definirane testne skupine in posamezni testni uporabniki. Organiziran je bil delovni sestanek, na katerem je bil vsakemu posamezniku predstavljen namen testiranja in predana so bila navodila za izvedbo testiranja. Med uporabnike so bili razdeljeni tudi testni scenariji.

V prvem ciklu testiranja so bile ugotovljene določene napake tako na aplikativni ravni (sistem Libris 2) kot tudi na integracijski. Napake so bile odpravljene in sledil je nov cikel testiranja, ki je dal pozitivne rezultate. Po uspešnem testiranju so sledile aktivnosti postavitve rešitve v produkcijsko okolje.

4.2 Predlogi in spremembe

Vpeljava pristopa SOA v organizacijo v glavnem temelji na informatizaciji poslovnih procesov. Zaradi tega tudi metodološki okvir pokriva vse ključne faze življenjskega cikla poslovnega procesa. Metodološki okvir zelo dobro pokriva začetne faze vpeljave (analiza postopka in načrtovanje in specifikacije), ki so tudi med najpomembnejšimi fazami pri sami vpeljavi. Bistveno manj pa metodološki okvir pokriva zaključne faze (postavitev).

Kot sem v okviru naloge že večkrat omenil, se je vpeljave SOA treba lotevati postopoma in postopoma tudi informatizirati posamezne poslovne procese. Ker je znotraj organizacije lahko takih poslovnih procesov veliko, postane sčasoma postavitev rutinska faza. Tega seveda ne moremo govoriti pri prvih informatizacijah. Na projektu vpeljave SOA v podjetje Mladinska knjiga, katerega primer je tudi opisan v poglavju 4.1, smo izvedli prvi prehod informatiziranega poslovnega procesa na osnovi pristopa SOA v produkcijsko okolje.

Ugotovil sem, da v sami fazi postavitve obstajajo številne pasti, ki lahko samo postavitev rešitve ogrozijo in tako bodisi podaljšajo čas izvedbe bodisi vplivajo na zadovoljstvo projektne skupine in, kar je ključnega pomena, nadrejenih. V nadaljevanju navajam nekaj ugotovljenih pasti, na katere je treba biti pozoren v fazi postavitve:

- **vrstni red postavitve programskih rešitev:** V primeru, ko zahteva faza postavitve namestitev več programskih rešitev v produkcijsko okolje naenkrat, je treba opredeliti vrstni red, po katerem se bodo programske rešitve nameščale. S tem se lahko izognemo nepravilnemu izvajanju informatiziranega poslovnega procesa.
- **časovna opredelitev postavitve:** Fazo postavitve je treba načrtovati tako, da namestitev informatiziranega poslovnega procesa ne vpliva na izvajanje obstoječih poslovnih procesov. Kot primer navajam informatiziran poslovni proces v podjetju Mladinska knjiga, katerega namestitev je bila izvedena v nedeljo, ko so trgovine podjetja zaprte in sistemi ugasnjeni.
- **rezervni scenarij:** Čeprav je lahko informatiziran poslovni proces popolnoma stestiran na testnem okolju, lahko pride pri postavitvi v produkcijsko okolje do nepričakovanih težav. Zaradi tega je vedno treba opredeliti tudi rezervni scenarij, s katerim bomo lahko vzpostavili prejšnje stanje produkcijskega okolja.

Na osnovi tega sem prišel do ugotovitve, da je lahko slednja faza ena izmed najzahtevnejših faz, zato bi bilo treba pasti in predvsem mehanizme, kako se tem pastem izogniti, bolje opredeliti znotraj samega metodološkega okvira.

Področje, ki mu je treba znotraj metodološkega okvira posvetiti še veliko časa, je spremljanje izvajanja poslovnih procesov (angl. *monitoring*) in merjenje. Gre za področje, ki ga pri klasičnem razvoju programske opreme le redkokdaj obravnavamo. Pri informatizaciji poslovnih procesov se lahko s pomočjo spremljanja izvajanja in merjenja pride do podatkov, ki jih je mogoče uporabiti za optimizacijo izvajanja poslovnega procesa. Kot je definirano v življenjskem ciklu poslovnega procesa (glej poglavje 2.2), obstaja posebna faza, v kateri izvajanje procesa spremljamo in na osnovi pridobljenih podatkov lahko izvajamo različne analize in rezultate analiz uporabimo bodisi za reševanje napak na procesu bodisi za njegovo optimizacijo. Metodološki okvir sicer definira mehanizme, kako spremljati izvajanje poslovnih procesov, ampak zgolj za odpravljanje napak, nastalih pri izvajanju. Predvideni so torej določeni vpogledi v izvajanje in ugotavljanje nastalih napak in njihovo odpravljanje. Bolj se je v metodološkega okviru treba posvetiti tudi pridobivanju kazalcev in ključnih kazalnikov poslovanja (angl. *Key Performance Indicators*), s katerimi bomo lahko izvajanje procesov dejansko ovrednotili in na osnovi teh podatkov proces tudi izboljšali.

Malinverno (2006) definira management znotraj pristopa SOA ne kot možnost, ampak kot nujnost. Še posebno to velja takrat, kadar je organizacija, v katero SOA vpeljujemo, velika in razdeljena na posamezne enote. Management je znotraj SOA eden izmed glavnih mehanizmov, s katerimi lahko zagotovimo eno izmed glavnih prednosti pristopa SOA, to je ponovno uporabljivost. Metodološki okvir vključuje tudi management in treba se ga je

zavedati v vseh fazah vpeljave, sicer je lahko projekt vpeljave v veliki nevarnosti. Znotraj okvira so opredeljeni določeni mehanizmi, ki jih lahko uvrščamo v management. Opaziti pa je, da so premalo definirani postopki ponovne uporabljivosti storitev. Razlog je mogoče iskati v tem, da je metodološki okvir nastal na osnovi izkušenj, pridobljenih na konkretnih projektih. Ker so projekti še v zgodnjih fazah vpeljave pristopa SOA, prava vrednost ponovne uporabljivosti še ni prišla na dan. Z informatizacijami drugih poslovnih procesov se je šele pokazala prednost ponovne uporabljivosti, ki bo vplivala tudi na aktivnosti znotraj faz metodološkega okvira.

ZAKLJUČEK

Hitre in pogoste spremembe v poslovnem okolju zahtevajo od današnjih organizacij zmožnost prilagajanja na spremembe. Takšno prilagodljivost lahko dosežemo s tesno povezanostjo IT in poslovanjem organizacije in to na osnovi integracije programskih rešitev. Ker v večini današnjih organizacij temelji okolje IT na večjem številu programskih rešitev, potrebujemo arhitekturni pristop in metodološki okvir, ki nam olajša integracijo programskih rešitev. Integracija aplikacij na osnovi pristopa SOA je zagotovo pravi pristop, ki brez ustreznega metodološkega okvira ne prinese pravih rezultatov. V okolju IT obstajajo številni metodološki okviri, vendar nobeden od slednjih v celoti ne opredeljuje in upošteva lastnosti, ki so značilne za arhitekturni pristop SOA.

Kot rešitev za slednjo problematiko sem v magistrski nalogi predstavil arhitekturni pristop SOA in metodološki okvir vpeljave SOA v organizacijo. Metodološki okvir je nastal na osnovi izkušenj, pridobljenih pri vodenju realnega projekta vpeljave, in na osnovi faz življenjskega cikla poslovnega procesa, ki je pri sami vpeljavi SOA eden od ključnih poslovnih elementov. V posameznih fazah sem opredelil tudi posamezne izdelke, ki so nastali med izvajanjem projekta. Ugotovil sem, da ti izdelki dajejo metodološkemu okviru dodatno pomembnost, saj bistveno vplivajo na kakovost implementacije poslovnega procesa in izboljšujejo komunikacijo in način sodelovanja med različnimi skupinami, ki so vključene v projektno skupino. Poleg tega omogočajo hitrejše spoznavanje in razumevanje konceptov pristopa SOA. V metodološkem okviru sem opredelil tudi določene organizacijske posebnosti, ki so pri informatizaciji poslovnih procesov na osnovi pristopa SOA nekaj posebnega in na katere je treba biti še posebno pozoren. Te posebnosti so povezane predvsem z načinom vodenja tako projekta, ki temlji na pristopu SOA, kot tudi projektne skupine. Ti projekti se od klasičnih projektov razvoja programskih rešitev razlikujejo predvsem v arhitekturi programske rešitve, ki jo je treba razviti, kompleksnosti same rešitve, ki najpogosteje zahteva visoko stopnjo razumevanja in povezovanja konceptov s strani razvijalcev, in strukturi projektne skupine. Struktura projektne skupine lahko vključuje tudi posameznike iz zunanjih organizacij, kar dodatno vpliva na zahtevnost vodenja takšne skupine in med drugim tudi na kakovost programske rešitve kot celote.

V drugem delu naloge sem predstavil primer uporabe metodološkega okvira na konkretnem projektu vpeljave in na osnovi tega metodološki okvir tudi ovrednotil. Ugotovil sem, da vsebuje metodološki okvir vse potrebne faze in da lahko s pomočjo metodološkega okvira pridemo do pozitivnih rezultatov pri informatizaciji poslovnih procesov na osnovi pristopa SOA. Vsi izdelki, ki so definirani znotraj posameznih faz, so za samo izvedbo faze zelo pomembni in vplivajo na kakovost programske rešitve in na verjetnost pojavljanja napak ali pomanjkljivosti v nadaljnjih fazah vpeljave. Poleg slednjega sem na konkretnem projektu tudi ugotovil, da obstajajo določena področja, ki so znotraj okvira premalo opredeljena in upoštevana. Ta področja so povezana predvsem z managementom, ki je ključnega pomena pri

vpeljavi SOA v organizacijo. Čeprav so lahko vse faze vpeljave metodološkega okvira natančno opredeljene in so pri njegovi uporabi vse faze tudi upoštevane, se lahko brez ustreznega managementa hitro prenaglimo in kršimo osnovna načela pristopa SOA. Ponovno uporabljivost storitev kot eno izmed ključnih načel pristopa SOA lahko dosežemo samo z ustreznim managementom, ki mora biti prisoten v vseh fazah metodološkega okvira vpeljave SOA v organizacijo.

Glede na to, da so vsi projekti vpeljave SOA, v katerih sodelujem, še vedno v izvajanju in bodo zaradi informatizacije novih poslovnih procesov v izvajanju še kar nekaj časa, bom lahko pridobil še veliko novih izkušenj in spoznanj, s pomočjo katerih bom lahko metodološki okvir dopolnil in dodatno izboljšal.

LITERATURA IN VIRI

1. Arsanjani, Ali. (2004, 9. november). Service-oriented modeling and architecture. *E-revir*. Najdeno 31. januarja 2009 na spletnem naslovu http://www.ibm.com/developerworks/library/ws-soa-design1/
2. Arsanjani, Ali (2007, 28. marec). Design on SOA solution using a reference architecture. *E-revir*. Najdeno 14. julija 2009 na spletnem naslovu http://www.ibm.com/developerworks/library/ar-archtemp/
3. Barnes, M., Scholler, D. & Malinverno, P. (2005, 6. september). Benefits and Challenges of SOA in Business Terms. *E-revir*. Najdeno 26. decembra 2008 na spletnem naslovu http://my.gartner.com/portal/server.pt?open=512&objID=218&mode=2&PageID=466502&resId=485146&ref=QuickSearch
4. Barnes, M. (2005, 1. november). Leveraging Reuse to Drive Adoption of Service-Oriented Architecture. *E-revir*. Najdeno 26. decembra 2008 na spletnem naslovu http://my.gartner.com/portal/server.pt?open=512&objID=218&mode=2&PageID=466502&resId=495686&ref=QuickSearch
5. Biscotti, F. (2008, 10. marec). Report Highlight for Market Trends: Multienterprise/B2B Infrastructure Market, Worldwide, 2008. *E-revir*. Najdeno 26. decembra 2008 na spletnem naslovu http://my.gartner.com/portal/server.pt?open=512&objID=218&mode=2&PageID=466502&resId=619310&ref=QuickSearch
6. Blechar, M. (2008, 12. junij). Defining the Discipline of Application Architecture. *E-revir*. Najdeno 27. decembra 2008 na spletnem naslovu http://my.gartner.com/portal/server.pt?open=512&objID=218&mode=2&PageID=466502&resId=693308&ref=QuickSearch
7. Bradley, A., Schulte, W., Sholler, D. & Malinverno, P. (2008, 2. december). Building an SOA Business Case: A Gartner Framework. *E-revir*. Najdeno 29. decembra 2008 na spletnem naslovu http://my.gartner.com/portal/server.pt?open=512&objID=218&mode=2&PageID=466502&resId=824819&ref=QuickSearch
8. Erl, T. (2008). *Service-Oriented Architecture, Concepts, Technology, and Design.* Crawfordsville: Prentice Hall.
9. Erl, T. (2008). SOA Methodology. Najdeno 31. 1. 2009 na spletnem naslovu http://www.soamethodology.com/
10. Gaur, H. & Zirn M. (2006). *BPEL Cookbook, Best practice for SOA-based integration and composite application development*. Birmingham: Packt Publishing.
11. Hill, J. & Melenovsky, M. (2006, 11. april). Achieving Agility: BPM Delivers Business Agility Through New Management Practices. *E-revir*. Najdeno 27. decembra 2008 na spletnem naslovu http://my.gartner.com/portal/server.pt?open=512&objID=218&mode=2&PageID=466502&resId=490856&ref=QuickSearch
12. Humphrey, W. (2006). *TSP(SM)–Coaching Development Teams*. Indianapolis: Addison-Wesley Professional.

13. Jurič, M., Sarang, P., Loganathan, R. & Jennings, F. (2007). *SOA Approach to Integration*. Birmingham: Packt Publishing.
14. Kenney, L. (2007, 31. december). Magic Quadrant for Integrated SOA Governance Technology Sets, 2007. *E-revir.* Najdeno 27. decembra 2008 na spletnem naslovu http://my.gartner.com/portal/server.pt?open=512&objID=218&mode=2&PageID=466502&resId=572713&ref=QuickSearch
15. Krisper, M., Rozman, I. & Silič, M. (2000). Enotna metodologija razvoja informacijskih sistemov, tretji zvezek, Ljubljana, 2000.
16. Lheureux, B. & Malinverno, P. (2008, 28. maj). Magic Quadrant for Integration Service Providers. *E-revir.* Najdeno 27. decembra 2008 na spletnem naslovu http://my.gartner.com/portal/server.pt?open=512&objID=218&mode=2&PageID=466502&resId=680713&ref=QuickSearch
17. Malinverno, P. (2006, 6. januar). Service-Oriented Architecture Craves Governance. *E-revir.* Najdeno 26. decembra 2008 na spletnem naslovu http://my.gartner.com/portal/server.pt?open=512&objID=218&mode=2&PageID=466502&resId=488180&ref=QuickSearch
18. Malinverno, P. (2006, 27. april). Sample Governance Mechanisms for a Service-Oriented Architecture. *E-revir.* Najdeno 26. decembra 2008 na spletnem naslovu http://my.gartner.com/portal/server.pt?open=512&objID=218&mode=2&PageID=466502&resId=491623&ref=QuickSearch
19. Malinverno, P. & Barnes, M. (2006, 16. avgust). SOA: Where Do I Start? *E-revir.* Najdeno 21. decembra 2008 na spletnem naslovu http://my.gartner.com/portal/server.pt?open=512&objID=218&mode=2&PageID=466502&resId=495226&ref=QuickSearch
20. Malinverno, P. (2006, 16. avgust). Learn the Key Success Factors for SOA Deployments. *E-revir.* Najdeno 26. decembra 2008 na spletnem naslovu http://my.gartner.com/portal/server.pt?open=512&objID=218&mode=2&PageID=466502&resId=495686&ref=QuickSearch
21. Matsumura, M. (2006, 24. julij). SOA Integration and Methodologies. *E-revir.* Najdeno 1. septembra 2009 na spletnem naslovu http://www.infoq.com/news/SOA-Methodologies
22. Mittal, K. (2006, 18 julij). Create the ideal SOA team. *E-revir.* Najdeno 4. januarja 2009 na spletnem naslovu http://www.ibm.com/developerworks/architecture/library/ar-soateam/index.html
23. Natis, Y. & Pezzini M. (2007, 26. oktober). Twelve Common SOA Mistakes and How to Avoid Them. *E-revir.* Najdeno 25. decembra 2008 na spletnem naslovu http://my.gartner.com/portal/server.pt?open=512&objID=218&mode=2&PageID=466502&resId=537409&ref=QuickSearch
24. Natis, Y. (2007, 4. april). Applied SOA: Transforming Fundamental Principles Into Best Practices. *E-revir.* Najdeno 25. decembra 2008 na spletnem naslovu http://my.gartner.com/portal/server.pt?open=512&objID=218&mode=2&PageID=466502&resId=503238&ref=QuickSearch

25. Olding, E. & Rosser, B. (2007, 4. oktober). Getting Started With BPM, Part 1: Assessing Readiness. *E-revir*. Najdeno 27. decembra 2008 na spletnem naslovu http://my.gartner.com/portal/server.pt?open=512&objID=218&mode=2&PageID=466502&resId=528510&ref=QuickSearch
26. Olding, E. & Rosser, B. (2007, 4. oktober). Getting Started With BPM, Part 3: Understanding Critical Success Factors. *E-revir*. Najdeno 27. decembra 2008 na spletnem naslovu http://my.gartner.com/portal/server.pt?open=512&objID=218&mode=2&PageID=466502&resId=528512&ref=QuickSearch
27. Pezzini, M. (2008, 26. september). Divide and Conquer: Taming Complexity Through Federated SOA. *E-revir*. Najdeno 20. decembra 2008 na spletnem naslovu http://my.gartner.com/portal/server.pt?open=512&objID=218&mode=2&PageID=466502&resId=765713&ref=QuickSearch
28. Pijanovski, K. (2008, 6. oktober). The UDDI API Sets. *E-revir*. Najdeno 25. avgusta 2008 na spletnem naslovu http://www.keithpij.com/Home/tabid/36/EntryID/16/Default.aspx
29. Rosen, A. (2004). *Effective IT Project Management: Using Teams to Get Projects Completed on Time and Under Budget*. New York: AMACON.
30. Rosser, B. (2008, 7. april). Taking Advantage of User-Friendly Business Process Modeling. *E-revir*. Najdeno 27. decembra 2008 na spletnem naslovu http://my.gartner.com/portal/server.pt?open=512&objID=218&mode=2&PageID=466502&resId=641217&ref=QuickSearch
31. Rozman, T. (2006). *Metoda za modeliranje in predstavitev obsežnih delovnih procesov*. Maribor: Fakulteta za elektrotehniko, računalništvo in informatiko.
32. Schulte, W. (2007, 3. maj). The Enterprise Service Bus: Communication Backbone for SOA. *E-revir*. Najdeno 27. decembra 2008 na spletnem naslovu http://my.gartner.com/portal/server.pt?open=512&objID=218&mode=2&PageID=466502&resId=504645&ref=QuickSearch
33. Schulte, W. (2008, 7. januar). Predicts 2008: Business Applications' Architectural Styles Are Changing. *E-revir*. Najdeno 26. decembra 2008 na spletnem naslovu http://my.gartner.com/portal/server.pt?open=512&objID=218&mode=2&PageID=466502&resId=577607&ref=QuickSearch
34. Seničar, D. (2006). *Modeliranje in avtomatizacija poslovnih procesov v podjetju*. Ljubljana: Ekonomska fakulteta.
35. Sholler, D. (2008, 3. julij). Hype Cycle for Application Architecture, 2008. *E-revir*. Najdeno 28. decembra 2008 na spletnem naslovu http://my.gartner.com/portal/server.pt?open=512&objID=218&mode=2&PageID=466502&resId=713410&ref=QuickSearch
36. Sholler, D. (2008, 26. september). 2008 SOA User Survey: Adoption Trends and Characteristics. *E-revir*. Najdeno 23. decembra 2008 na spletnem naslovu http://my.gartner.com/portal/server.pt?open=512&objID=218&mode=2&PageID=466502&resId=765720&ref=QuickSearch
37. Thompson, J. (2008, 11. marec). Key Issues for Application Integration, 2008. *E-revir*. Najdeno 22. decembra 2008 na spletnem naslovu http://my.gartner.com/p-

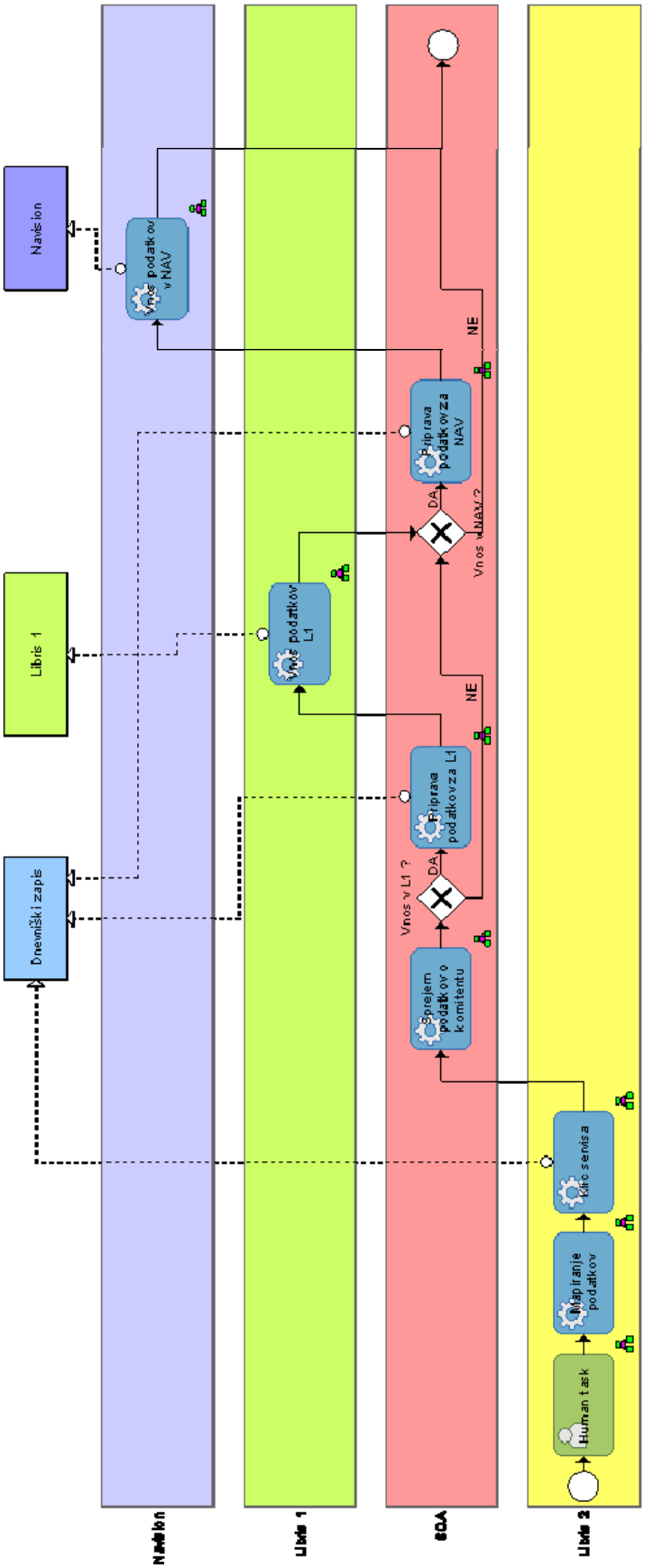
ortal/server.pt?open=512&objID=218&mode=2&PageID=466502&resId=619612&ref=QuickSearch

38. Thompson, J. & Schulte, W. (2008, 29. julij). Understanding the Three Patterns of Application Integration. *E-revir*. Najdeno 26. decembra 2008 na spletnem naslovu http://my.gartner.com/portal/server.pt?open=512&objID=218&mode=2&PageID=466502&resId=734222&ref=QuickSearch
39. Thompson, J. (2008, 5. september). Modernizing IT by Federating IT Disciplines. *E-revir*. Najdeno 22. decembra 2008 na spletnem naslovu http://my.gartner.com/portal/server.pt?open=512&objID=218&mode=2&PageID=466502&resId=753513&ref=QuickSearch
40. Thompson, J. & Pezzini M. (2009, 24. marec). Gartner's Reference Architecture for SOA Application Infrastructure. *E-revir*. Najdeno 17. maja 2009 na spletnem naslovu http://my.gartner.com/portal/server.pt?open=512&objID=218&mode=2&PageID=466502&resId=919112&ref=QuickSearch
41. Tilkov, S. (2007, 18. julij). Roles in SOA Governance. *E-revir*. Najdeno 04. januarja 2009 na spletnem naslovu http://www.infoq.com/articles/tilkov-soa-roles
42. Weske, M. (2007) . *Business Process Management*. Potsdam, Germany: Springer
43. White, S.(2007, 22. februar). Process Modeling Notations and Workflow Patterns. *E-revir*. Najdeno 04. januarja 2009 na spletnem naslovu http://www.modernanalyst.com/Resources/Articles/tabid/115/articleType/ArticleView/articleId/56/Process-Modeling-Notations-and-Workflow-Patterns.aspx

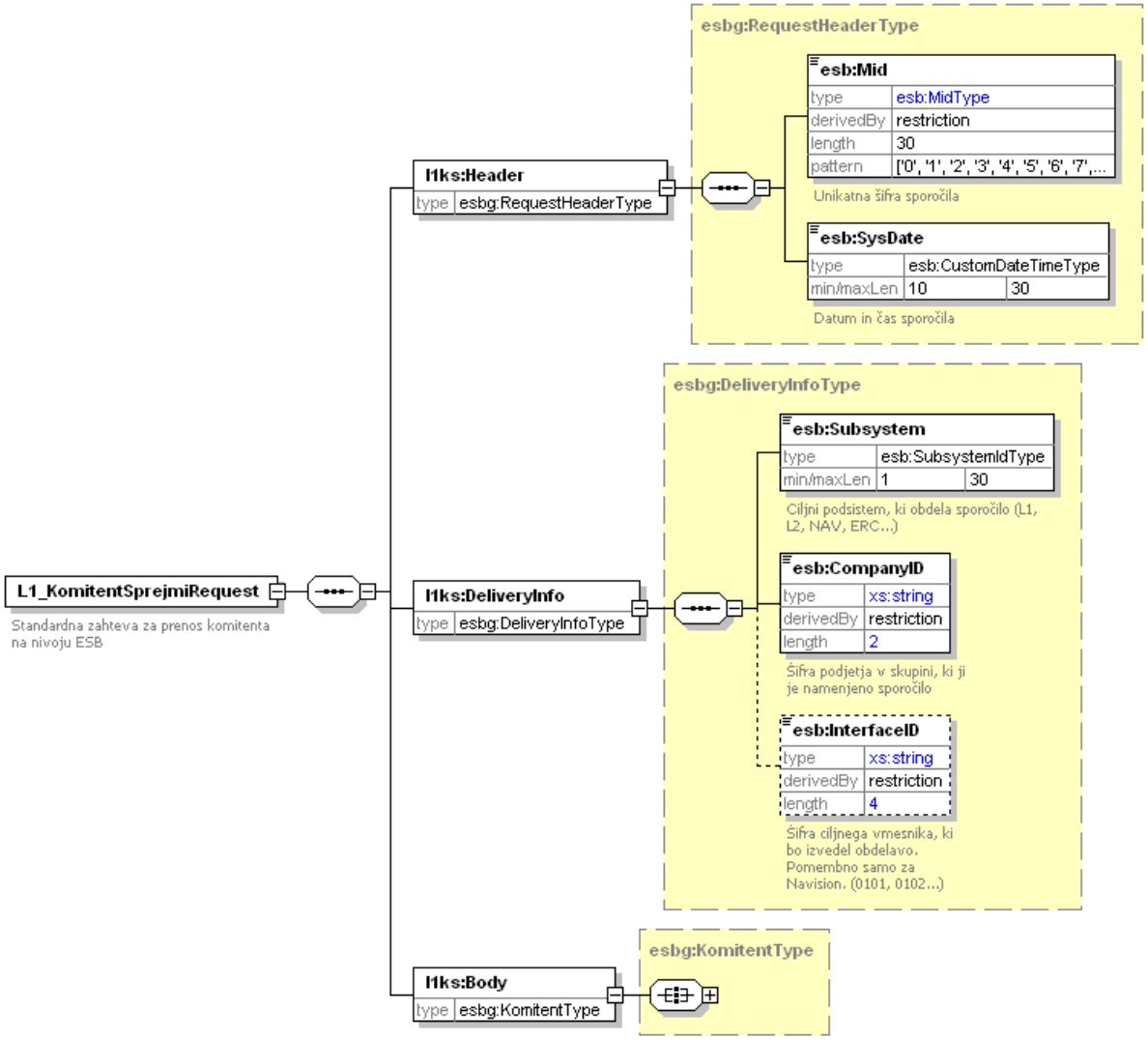
PRILOGE

Priloga 1: Model integracijskega postopka urejanja komitentov
Priloga 2: Primer XSD sheme na vmesniku E
Priloga 3: Izvleček iz preslikovalne tabele postopka urejanja komitentov
Priloga 4: Seznam korakov faze migracije podatkov znotraj postopka urejanja komitentov
Priloga 5: Primer testnega scenarija

Priloga 1: Model integracijskega postopka urejanja komitentov



Priloga 2: Primer XSD sheme na vmesniku E



Priloga 3: Izvleček iz preslikovalne tabele postopka urejanja komitentov

Ponor: LIBRIS1 → PRAVOPLAC			**Izvor: ESB hrbtenica – XSD shema partnerja**	
Zap. št.	**Ime polja**	**Vrsta**	**Ime Type**	**Ime značke**
***1**	**PRAVOSEBA**	**INTEGER**	**KomitentType**	**No**
	INT(No)			
***2**	**NAZIV1**	**CHAR 42**	**KomitentType**	**Name**
	UPPER(Name + No) Opomba: Zadnjih 7 mest je šifra partnerja.			
5	**NAZIVK2**	**CHAR 15**		
	Se ne prenaša iz hrbtenice.			
***6**	**PTT NAZIV**	**CHAR 10**	**AdressList / AdressItem**	**Postname**
	Naslov mora imeti vrednost v znački AdressList/AdressItem/ Priority = 1 (Prioriteta). Na osnovi tega Item se dela primerjava za iskanje DRŽAVE in POŠTE. Prvih 10 znakov iz značke Postname.			

Priloga 4: Seznam korakov faze migracije podatkov znotraj postopka urejanja komitentov

Korak	**Uvoz podatkov iz Libris 1**
Postopek	• Na podatkovno bazo IWOJIMI si v delovni direktorij prenesemo skripte in podatke. • Na delovnih tabelah se pobriše vsebina tabel Libris1 s skriptami: LIBRIS_TABELE.L1_*. • Poženemo naslednje bazne procedure o libris_tabele.l1_disable_constraints; • libris_tabele.L1_truncate_tables; • …….

Korak	**Uvoz podatkov iz Libris 2**

Postopek	• Zbrišemo vsebino L2 (tabele n_*) 2x!!!
	• Poženemo naslednje bazne procedure
	  o libris_tabele.l2_disable_constraints;
	  o libris_tabele.l2_truncate_tables.
	• Preverimo z libris_tabele.l2_table_counts.
	• ……

Korak	**Migracija**
Postopek	• Poženemo bazno proceduro prenos_l1_l2.vnesi_popravi_vse.

Korak	**Izvoz podatkov za Libris 2**
Postopek	• V delovni direktorij skopiramo datoteko LIBRIS_structure_only in jo popravimo v LIBRIS_OUT. • V LIBRIS_OUT ustvarimo povezave na Oraclove tabele N_* • ….

Priloga 5: Primer testnega scenarija

Genis
okolju prijazne informacijske rešitve

Projekt: MKSOA

TESTNI SCENARIJ MKSOA

OSNOVNI PODATKI SCENARIJA

Številka testa:	0001		
Datum testa:	20.2.009	Sistem:	Libris 1
Test izvedel:	Jana Bolta		

PODROBNOSTI SCENARIJA

Naziv testnega scenarija

Vnos naročila tuje PO –ni davčni zavezanec v EU

Test: pravilnost fakture in prenos v Navision

Podroben opis testnega scenarija

vpišemo šifro kupca, datum naročila,oznako valute, rok plačila
oznako davka (način obračuna davka je vpisan v referenčni tabeli- EU, OSTALI SVET)PO ki ni davčni zavezanec-nima DŠ plača davek
vnos naročenih artiklov, količina, %rabata
naročilo spustimo v obdelavo - nastane dobavnica, vnos poštnine
odprema, potrditev dobavnice, nastane račun

REZULTATI TESTIRANJA

Test izveden DA Test opravljen DA

Komentarji / napake

Podatki so bili uspešno vnešeni