Univerza
v Ljubljani

Fakulteta za računalništvo
in informatiko

Tržaška 25
1000 Ljubljana, Slovenija
telefon: 01 476 84 11
faks: 01 426 46 47
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 00509/2010
Datum: 05.04.2010

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **PETER SELAK**

Naslov: **MOŽNOSTI INTEGRACIJE Z ERP SISTEMOM SAP**

OPTIONS FOR INTEGRATION OF SAP AND EXTERNAL APPLICATIONS

Vrsta naloge: Diplomsko delo visokošolskega strokovnega študija

Tematika naloge:

Proučite možnosti integracije ERP sistema SAP z zunanjimi aplikacijami. Pri tem preizkusite tehnologije podjetij SAP in Microsoft. Izdelajte tudi primerjalno analizo različnih pristopov k integraciji z uporabo tehnologij, ki ste jih preizkusili.

Mentor:

doc. dr. Rok Rupnik

Dekan:

prof. dr. Franc Solina

Fakulteta za računalništvo in informatiko
UNIVERZA V LJUBLJANI
2

UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Peter Selak

Možnosti integracije z ERP sistemom SAP

DIPLOMSKO DELO NA VISOKOŠOLSKEM STROKOVNEM ŠTUDIJU

Mentor: doc. dr. Rok Rupnik

Ljubljana 2010

I Z J A V A O A V T O R S T V U

diplomskega dela

Spodaj podpisani/-a Peter Selak,

z vpisno številko 63010301,

sem avtor diplomskega dela z naslovom:

Možnosti integracije z ERP sistemom SAP

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom (naziv, ime in priimek)

  doc. dr. Rok Rupnik

  in somentorstvom (naziv, ime in priimek)

  _____

- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela

- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki »Dela FRI«.

V Ljubljani, dne 11.06.2010 Podpis avtorja: _____

Zahvala

Zahvaljujem se mentorju doc. dr. Roku Rupniku za natančna in strokovna navodila.
Zahvala gre tudi podjetju Genis, še posebej Alešu Šušteršiču za predstavitev njegovega dela.
Še posebna zahvala gre moji ženi Mojci za podporo pri študiju in pisanju diplomske naloge.

Kazalo

Seznam uporabljenih kratic

SOA	Service Oriented Architecture
SAP	System Analysis and Program Development
ABAP	Advanced Business Application Programming, nem. Allgemeiner Berichts-Aufbereitungs-Prozessor
ERP	Enterprise resource planning
GUI	Graphical user interface
SQL	Structured Query Language
IT	Informacijska tehnologija
RPC	Remote procedure call
SOA	Service oriented architecture
API	Application programming interface
SOAP	Simple Object Access Protocol
WSDL	Web Services Description Language
XML	Extensible Markup Language
ESB	Enterprise Service Bus
FTP	File transfer protocol
SFTP	Secure File Transfer Protocol
WCF	Windows Communication Foundation
Idoc	Intermediate Document
XSD	XML Schema Definition
RFC	Remote Function Call

Povzetek

V diplomski nalogi smo na kratko predstavili, kaj so ERP sistemi. Opisali smo glavne lastnosti treh izmed najbolj razširjenih sistemov pri nas. To so SAP, Baan, Microsoft Dynamics NAV (Navision). Izvedli smo primerjavo lastnosti vsakega izmed navedenih. Opisali smo osnovne pristope za integracije včasih in danes.

Kasneje smo za SAP naredili analizo treh načinov integracij s SAP-om. To so:
- tekstovna datoteka v povezavi s posli,
- BAPI funkcije,
- BizTalk strežnik.

Za glavni izziv smo vzeli primer prenosa prejetega računa v SAP sistem. Izvedli smo prenos enega računa na tri različne načine. Opisali smo zahteve za izvedbo določene integracije, ter potrebne korake za izvedbo.

Na koncu smo izvedli primerjavo vseh treh načinov. Primerjali smo glede na ceno programske opreme, času razvoja in kompleksnosti vzdrževanja sistema v produkcijskem delovanju. Iz teh treh primerjav smo izbrali tudi najprimernejši način integracije. Vodilo za izbiro je bila največja gospodarnost sistema.

Abstract

In this thesis, we briefly present ERP systems. We described the main features of three of the most popular systems in our country. These are SAP, Baan, Microsoft Dynamics NAV (Navision). We carried out a comparison of the characteristics of each of these. We described the basic approaches to integration in the past and today.

Later we did an analysis of 3 SAP integration methods with SAP. These are:
-Text file in connection with transactions,
-BAPI function,
-BizTalk Server.

For the main challenge as we take received transfer incoming invoice in the SAP system. We carried out the transfer of one invoice in three different ways. We described the requirements for the performance of a specific integration and the necessary steps for implementation.

Finally, we compare the three methods. We compared the price of software development time and complexity of system maintenance in a production operation. From these three comparisons we have chosen the most appropriate means of integration. Guide for selecting was economy of the system.

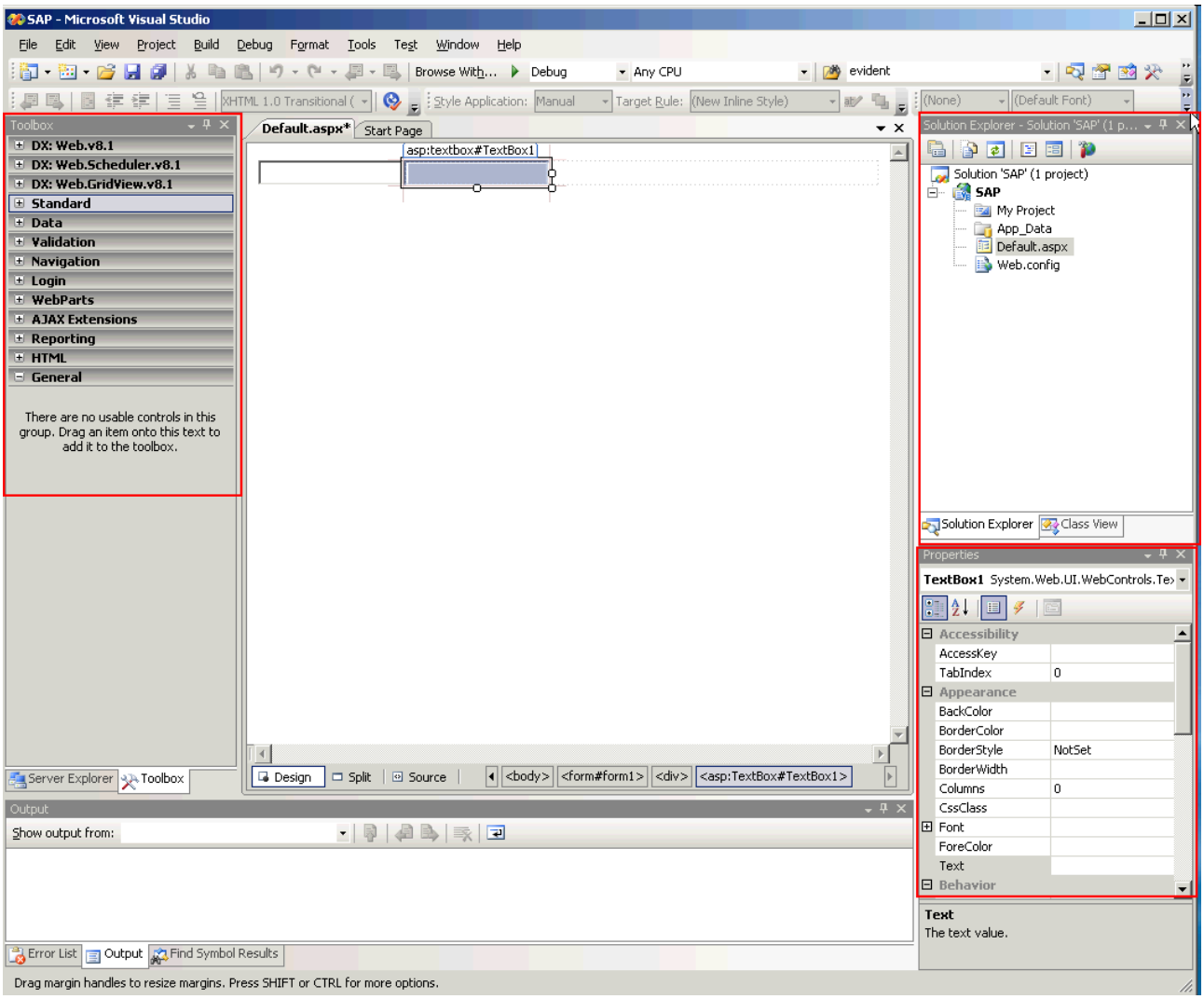
1 Uvod

1.1 Predstavitev uporabljenih tehnologij

Tehnologija v današnjem času, času računalništva, zelo hitro napreduje. Razvoj je zelo hiter tako na strojni opremi, kot na programski opremi. Na kratko bom predstavil glavna orodja, s katerimi sem se srečal pri raziskovanju teme diplomske naloge. Za uspešno izvedene raziskave je bilo potrebno uporabiti kar precej različnih sistemov.

1.1.1 Microsoft Visual studio 2008

Je trenutno najnovejše microsoftovo orodje za razvoj programske opreme. Podpira tako imenovane debele kliente (Client application), kot tudi razvoj spletnih aplikacij. Uporabnik ima možnost izbire med programskima jezikoma c# in vb. Podpira pa tudi skriptni jezik javascript. Meni osebno bolj ugaja vb, saj sem z njim začel programirati. Visual basic ni občutljiv na velike/male črke (case sensitive).
Integriran ima lastni spletni strežnik za razhroščevanje napisane kode (v primeru razvoja spletnih aplikacij). Ima vgrajene standardne kontrole za vse tipe aplikacij. Prav tako omogoča razne razširitve za določene komponente. Kot na primer lahko navedem Developper express (DevX) komponente. Te komponente se ob instalaciji pojavijo na levi strani ob standardnih elementih.



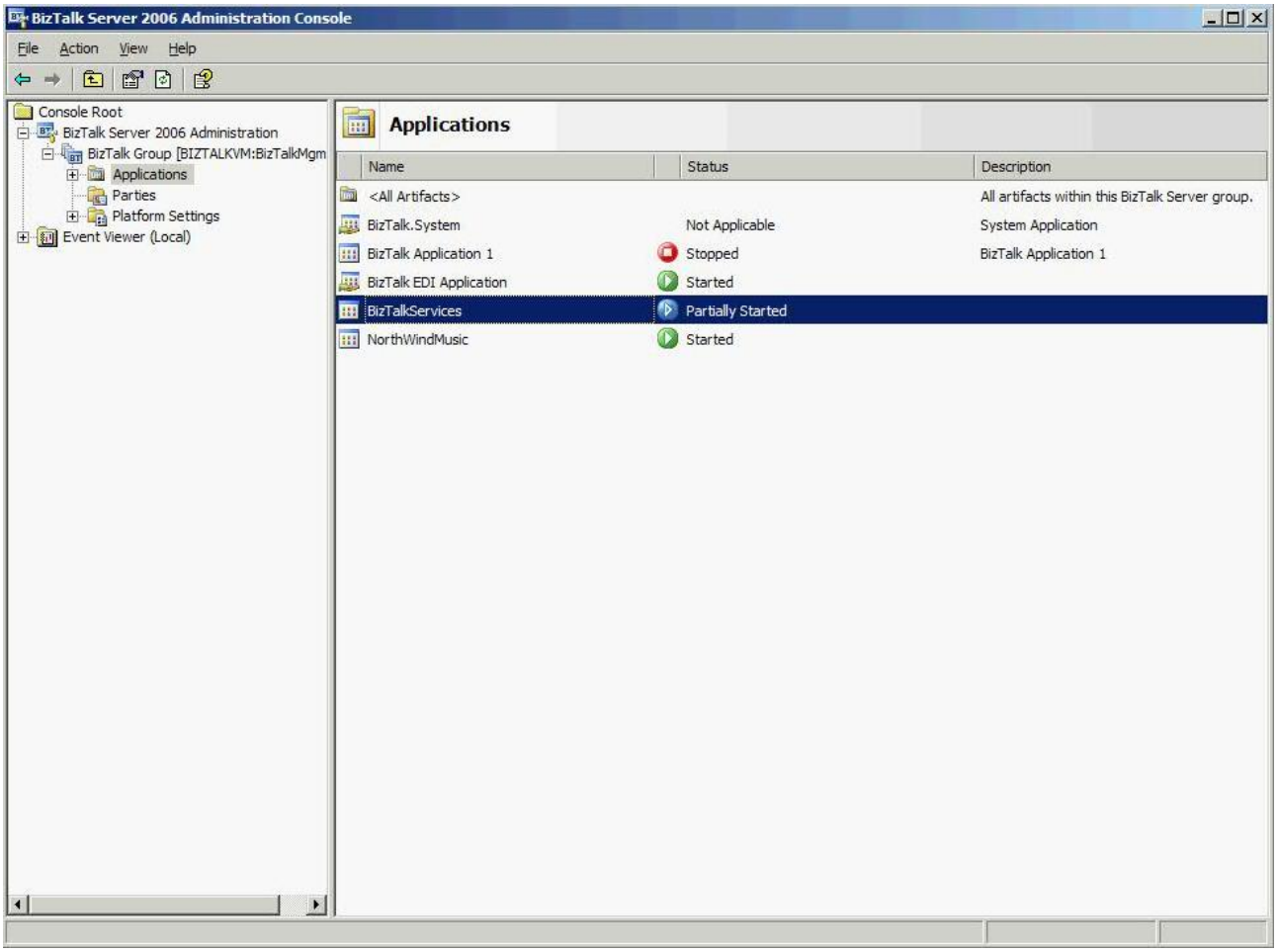
Slika 1: Ekranska slika orodja Microsoft Visual Studio 2008

Na desni strani je t.i. »Solution explorer« oziroma brskalnik rešitve. Tukaj so navedene vse komponente, ki so v projektih, ki so vključeni v rešitev. Spodaj na desni strani lahko spreminjamo lastnosti komponent.
Zelo uporabna funkcionalnost je »debugger« za skriptni jezik javascript. Te funkcionalnosti prejšnja verzija (2005) ni imela.
V primerjavi z verzijo 2005 je programerju bolj prijazno okolje za razvoj.

1.1.2 Microsoft BizTalk server 2006

V vsaki organizaciji, ki uporablja računalniško podporo, se srečujemo z različnimi sistemi. Noben sistem ni tako popoln , da bi podprl celoten poslovni proces. Določen del procesa je podprt v eni aplikaciji, ki pa je smiselno povezan z drugo aplikacijo. Tako se pojavi potreba po povezovanju med sistemi. Neumno bi bilo iste podatke vnašati v dva sistema, če jih lahko dobimo iz prvega (kjer je bil opravljen prvi vnos podatkov). Ker je pa ponavadi druga aplikacija na popolno drugi platformi, naletimo na težavo, kako spraviti podatke iz sistema1 v sistem2. Zelo močno orodje je BizTalk server. Primeren je za integracije med različnimi

bazami, sistemi… Strežnik ponuja predpripravljene adapterje za povezovanje. Izvaja tako imenovano orkestracijo nad sistemi, ki so vključeni v proces.
To je strežnik, ki teče na strežniku Microsoft server 2003 ali višja verzija.



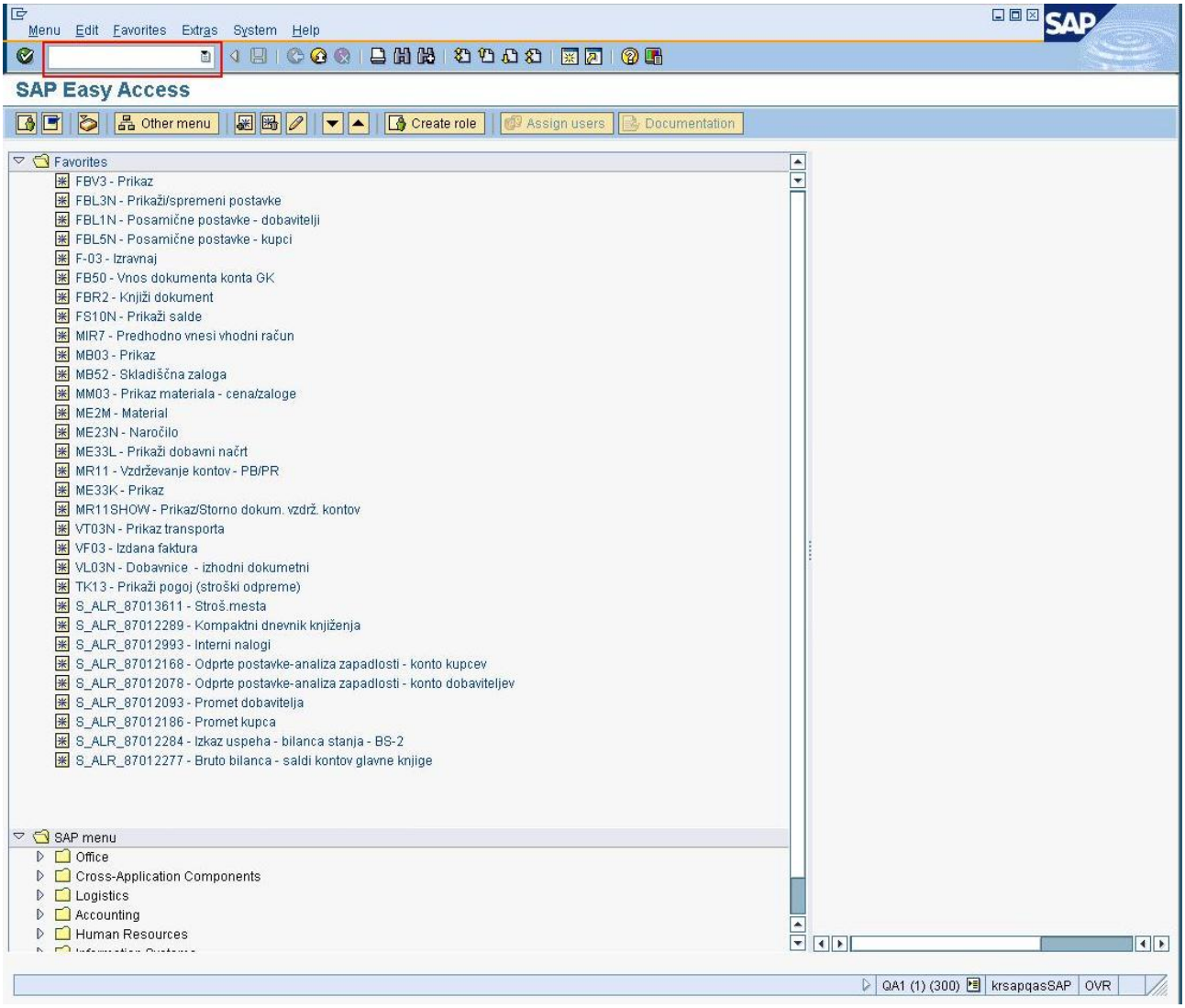
Slika 2: Administratorska konzola BizTalk strežnika

Grobi princip razvoja oz. vzpostavitve orkestracije med sistemi se izvaja v zgoraj omenjenem orodju Microsoft visual studio 2008 (2005). Ob instalaciji BizTalk strežnika se integrirajo novi projekti in objekti, s katerimi lahko nastavimo način komunikacije.
Ko končamo projekt, ga prevedemo in prenesemo (deploy) na BizTalk strežnik. Potem sledi zagon programa na BizTalk strežniku. Ta potem odvisno od načina (urnik, dogodek…) izvaja akcije.

1.1.3 SAP R3 Client

Kot nam že samo ime pove, gre za odjemalca za SAP strežnik. SAP klient je omogoča dostop do sistema SAP. Kdor pozna sistem SAP ve, da je to ogromen ERP sistem. Preko tega klienta lahko dejansko upravljamo strežnik. Je zelo močno orodje za delo s SAP strežnikom. Je pa zelo zanimivo, saj je instalacija za administratorje in navadne uporabnike, čisto ista. Razlikuje se le v tem, da uporabnika zavrne pri določeni akciji, če do le-te nima pravice. Hkrati lahko v tem istem programu napišemo lasten program in ga »debugiramo«. Podpira standarden SAP

programski jezik ABAP. Preko tega klienta se dejansko nadzira cel strežnik in vse akcije, ki se izvajajo.
Sistem temelji na transakcijah. V označen okvir lahko uporabnik vpiše šifro transakcije. Tako se mu odpre nova transakcija. To je v bistvu tako, kot bi pognal nov program. Na ekran dobi novo zaslonsko masko s polji in gumbi, ki so določeni v transakciji, katero je uporabnik pognal.

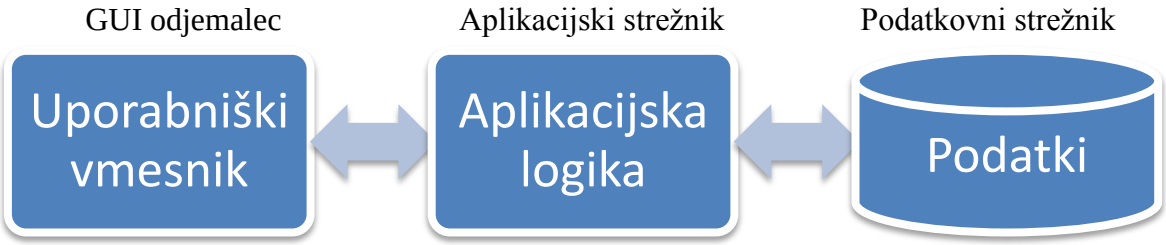


Slika 3: Vhodni ekran SAP klienta

1.2 ERP sistemi

ERP (Enterprise resource planning) je sistem za obvladovanje vseh procesov, resursov in informacij v podjetju. Ponavadi ERP sistemi vključujejo ogromno komponent, ki vključujejo tako programsko kot tudi strojno opremo. ERP sistemi ponavadi uporabljajo relacijsko bazo za shranjevanje podatkov. V začetku so bili taki sistemi namenjeni načrtovanju in obvladovanju večjih večinoma industrijsko naravnanih podjetij. Sedaj je pa razvoj tako napredoval, da sistemi obvladujejo vse vrste podjetij, ne glede na njihovo naravnanost.
Taki sistemi so primerni za srednja in večja podjetja, ki želijo obvladovati proces v čim večji meri. V čim večji meri pa zato, ker noben sistem ni popoln. Oziroma vsako podjetje ima svoje potrebe in želje. Prav tako se pojavi problem pri uvajanju ERP sistema v utečen delovni proces. Ponavadi uvajanje zahteva spremembo delovnega procesa, katere pa noben zaposleni ne sprejme z veseljem. Na dolgi rok se seveda investicija v tak sistem izplača, saj se hitro izkaže kje v delovnem procesu so težave in tako lahko vodilni hitreje ukrepajo. Glavni ERP sistemi v Evropi so: SAP, Baan, Navision…
[2,3]

Vsi trije predstavljeni sistemi uporabljajo 3-nivojsko arhitekturo:
- na najnižjem nivoju se nahaja podatkovni strežnik (relacijska baza),
- na vmesnem nivoju je logični strežnik (Baan server),
- na predstavitvenem nivoju so odjemalci, ki morajo biti nameščeni na vsaki delovni postaji.



Slika 4: 3-nivojska arhitektura

1.2.1 SAP R/3

Podjetje SAP je bilo ustanovljeno leta 1972 pod imenom *Systemanalyse und Programmentwicklung* ("System Analysis and Program Development") v Nemčiji. Ustanovilo ga je 5 bivših inženirjev IBM. Glavni produkt podjetja, ki se sedaj imenuje SAP AG, je SAP ERP sistem. Najnovejša verzija je 6.0. Predhodnik te verzije je R/3. Trenutno je večina slovenskih podjetij na verziji R/3.
SAP ERP je ena izmed petih aplikacij, ki jih podjetje ponuja v paketu »SAP Business suite« Tako, kot se za ERP spodobi, tudi SAP podpira celoten proces podjetja. To vključuje finančno poslovanje, spremljanje zalog, stroškov ...

Ko se podjetje odloči za nakup SAP ERP programskega paketa, pridejo SAP svetovalci in naredijo analizo trenutnega stanja delovnega procesa. Po opravljeni analizi naredijo oz. nastavijo sistem tako, da čim bolj podpira njihov delovni proces. Seveda so pa ponavadi potrebne tudi spremembe delovnega procesa.

SAP sistem vsebuje logični strežnik (SAP server), podatkovni strežnik (katerakoli relacijska baza, ponavadi je to Oracle) in pa kliente. Sistem uporablja tako imenovane »debele kliente«, saj zahteva instalacijo klienta na vsaki delovni postaji. Z instalacijo klienta se namestijo tudi knjižnice, ki omogočajo programski dostop do strežnika (BAPI).

Cena uvedbe SAP sistema v podjetju je zelo velika, zato je ta sistem primeren predvsem za srednja in velika podjetja. Sicer sedaj podjetje pričenja z akcijo, s katero bo prodrlo tudi na trg manjših podjetij. [3]

1.2.2 Baan

Baan korporacijo je ustanovil Jan Baan iz Nizozemske. Prvotni namen podjetja so bile finančne in administrativne storitve. Kasneje sta z bratom Paulom preusmerila razvoj sistemov v ERP. Najbolj popularen je bil program v začetku 90-ih. Bil je resna konkurenca takratnemu vodilnem podjetju na tem področju - SAP-ju. Propad podjetja je sprožila navidezna prodaja velikega števila licenc podjetju, ki je bilo v njihovi lasti. S tem so prikazali ogromne prihodke. Kasneje so podjetje prodali. Trenutno je podjetje v lasti Infor global solutions, ki pa je tudi preimenovalo produkt v SSA ERP LN. Zadnja verzija sistema je SSA ERP LN 6.1.
Trenutno je na milijone uporabnikov tega sistema na svetu, vendar jih ima večina verzijo IV ali V. Prodaja nove rešitve je zelo skromna.
Ciljne organizacije so srednja in velika podjetja.
[3]

1.2.3 Microsoft Dynamics NAV

Podjetje je bilo ustanovljeno leta 1984 na Danskem pod imenom PC&C ApS (Personal Computing and Consulting). Prvotno ime rešitve je bilo »Navision« (po združitvi s podjetjem Navision Software AS). Leta 2002 je Microsoft kupil podjetje in kasneje (leta 2005) spremenil ime produkta v Microsoft dynamics NAV. V Sloveniji je še vedno bolj poznan pod imenom »Navision«. To je Microsoftov ERP sistem. Zadnja verzija je 4.0.

Ciljne organizacije so srednje velika podjetja.
Sistem je v primerjavi z ostalima dvema precej cenejši, kar pa ne pomeni, da je slabši. V Sloveniji je kar precej popularen, saj je cenovno dostopnejši.
Za shranjevanje podatkov uporablja MS SQL 2005/2008 server ali pa lahko uporabnik izbere tudi Microsoft Dynamics NAV Database Server.
[3]

2 Predstavitev problema

Za glavni problem sem si izbral prenos prejete fakture v ERP sistem SAP. Ker ima ERP sistem pomanjkljiv del za odobravanje prejetih računov/faktur, se veliko podjetij odloči, da ta del procesa spelje izven glavnega ERP sistema. Prav tako se lahko odločijo še za kakšen drug del poslovnega procesa, saj ni nujno, da se v celoti lahko porabi ERP sistem. Lahko so tudi določeni moduli izključeni, in je s tem nižja cena sistema. Ta izključeni del pa lahko nadomestijo z lastno aplikacijo (če imajo zadostno IT podporo) ali jo naročijo pri drugem podjetju.

Tukaj nastane situacija, ko oba sistema obdelujeta iste podatke, vendar za različne namene. Te podatke je nesmiselno dvakrat vnašati, saj s tem podvojimo:
- čas za obdelavo podatkov,
-verjetnost človeške napake (pri prepisovanju podatkov, če gre za ročni vnos) ,
-resurse (ljudi) za vnos podatkov.
Zato imajo večji sistemi predvidene »vmesnike« za uvoz podatkov iz zunanjih neodvisnih sistemov. Na konkretnem primeru sem naletel na veliko pomanjkanje dokumentacije glede načinov uvoza podatkov v SAP.

Opis delnega postopka prejema računa:
Prejete fakture prihajajo na vložišče podjetja. Tam izvedejo zajem slike, vnesejo podatke (znesek, porazdelitev davka …) in pošljejo odgovorni osebi v podpis. Ta naredi delitev računa na stroškovna mesta. Ko podpisnik opravi svoje delo, v računovodstvu preverijo pravilnost zneskov. Ko ugotovijo, da je z računom vse vredu, se račun lahko prenese v SAP, od tam naprej gre v plačilo, in nato pa v glavno knjigo ter v ostale evidence. SAP je zelo močen pri raznih evidencah stroškov in računanju faktorjev učinkovitosti določenih enot in tako naprej. Pri prejemu računa je še vedno večina papirnih dokumentov. Nekaj jih je že v elektronski obliki, vendar je tega zelo malo. Če bi se podjetja povezala in si izmenjevala račune v e-Slog obliki, bi računovodstvo imelo precej manj dela z vnosom podatkov, saj bi jih lahko dobili v elektronski obliki. Trenutno je najbolj razširjen način je e-Slog.

2.1 Sodobni načini povezovanja sistemov

Prav je, da omenimo RPC, ki je eden od prvih načinov integracije med sistemi. RPC (remote procedure call) se v današnjem času ne uporablja več. Danes je SOA (Service Oriented Arhitecture) najbolj razširjen način povezovanja.
Nekaj besed o SOA.
SOA je storitveno usmerjena arhitektura. To pomeni, da nas ne zanimajo več API knjižnice, temveč imamo informacijo o storitvi.
O storitvi imamo sledeče informacije:
- kje se nahaja,
- kakšne podatke dobi na vhod,
- kakšne podatke vrne kot odgovor,
- splošno informacijo kaj naj bi določena storitev naredila.

Govorimo o spletnih servisih (Web Service). Pri spletnih servisih srečamo dva izraza:
- SOAP (Simple Object Access Protocol) je protokol izmenjave strukture med servisi,
- WSDL (Web Service Description Language) pa opisuje servis.

V obeh primerih govorimo o XML datotekah, ki so strukturirane tako, da so lepo berljive v navadnem tekstovnem urejevalniku.
[3]
Tretji način je ESB (Enterprise Service Bus). Ta način naj bi celo prehitel oz. nadomestil SOA. ESB izvaja orkestracijo izmenjave podatkov med različnimi sistemi. Tudi ta arhitektura temelji na spletnih servisih. Trend razvoja integracij med sistemi vsekakor kaže, da so spletni servisi zelo učinkoviti pri izvedbi le-te.

2.2 Cilj diplome

Glavni cilj diplome je skupna ocena primernosti posamezne metode integracije. Zgoraj opisan problem bo izveden na tri različne načine. Kasneje bo izvedena primerjava metod med sabo glede na različne parametre.

Primerjava bo izvedena po sledečih parametrih:
- cena potrebne programske in strojne opreme,
- čas razvoja integracije,
- zahtevnost vzdrževanja sistema.

Podjetja se pred vsakim nakupom večjega sistema srečujejo s prodajalci, ki po navadi zagotavljajo vse mogoče integracije. Ko pa pridejo v situacijo, ko je določeno integracijo potrebno izvesti, se dostikrat ustavi. Lahko pride do težave, da je določen vmesnik potrebno doplačati, ker pač ni bil vključen v to verzijo programa. Potem je težava, ker se morajo izvajalci dogovoriti o načinu in izvedbi integracije. Nato lahko še pride problem pomanjkanja dokumentacije o integracijah.
Ker je takih težav še veliko, sem se odločil, da naredim analizo za primer SAP integracije z ostalimi sistemi. Tako bo lahko razvidno, na kaj morajo biti kupci previdni pri izbiri sistema.

3 Opis načinov povezovanja do sistema SAP

3.1 Tekstovna datoteka – SAP Job

Prvi način, tekstovna datoteka, lahko rečemo, da izvira iz samih začetkov nove dobe računalništva. Temelji na strukturiranih zapisih v tekstovni datoteki. Datoteka služi za vhod v sistem SAP. Za prvi način - t. i. tekstovna datoteka - lahko rečemo, da …
Prednosti:
- uporabniku ni potrebno čakati na povratno informacijo (sistem v ozadju skrbi),
- uporabnik ne potrebuje dodatnih komponent za proženje prenosa.

Slabosti:
- zakasnitve pri prenosu iz enega sistema v drugega – npr. po urniku se prenaša enkrat na dve uri,
-uporabnik mora spremljati, kaj se dogaja z zapisi za prenos (lahko pride do napak pri prenosu).

SAP ima že od samega začetka t. i. »event«-e oz. dogodke, ki jih lahko prožimo z določenim ukazom. Drug ključen del uvoza podatkov so t. i. »job«-i oz. posli. Posli potem lahko naprej kreirajo dogodke, kličejo podprograme …
Torej moramo na SAP-u definirati dogodek, ki bo pognal določen posel, ki bo potem izvedel pognal podprogram za preverjanje strukture datoteke in naprej program za uvoz podatkov v sistem.
To je grob opis delovanja. Podrobneje bomo spoznali način v nadaljevanju.

3.1.1 Tekstovna datoteka

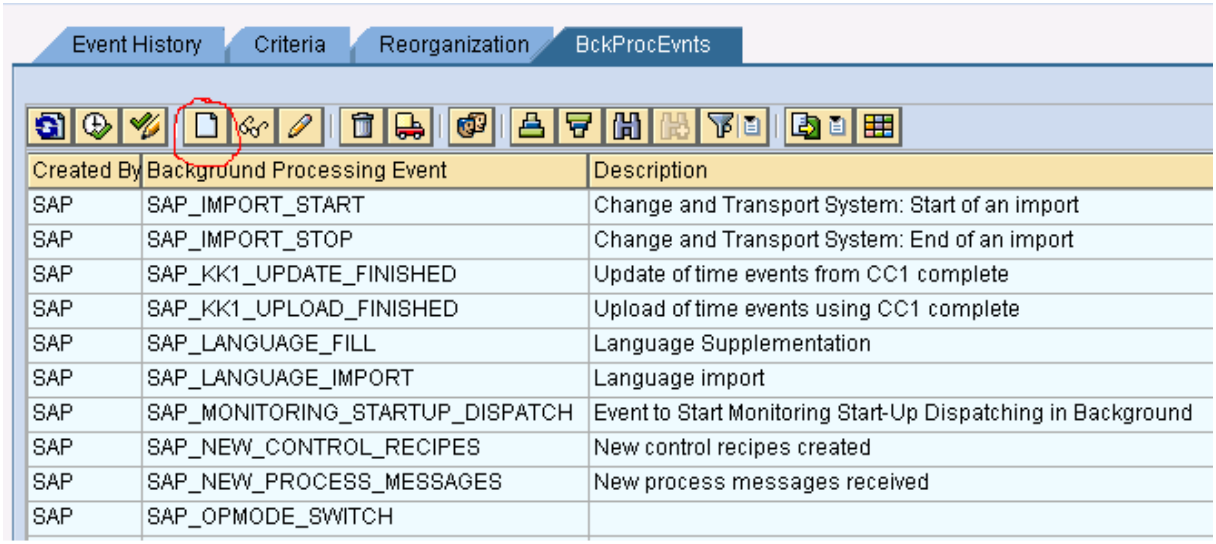
Zunanji sistem kreira tekstovno datoteko. Torej so vsi podatki zapisani v eni tekstovni datoteki. Kot smo omenili že zgoraj gre za paketen prenos podatkov. Torej se naenkrat lahko prenese več zapisov.
Datoteka ima predpisano strukturo podatkov. Določeno je zaporedje vrstic v datoteki – npr. najprej mora biti glava računa, in nato vknjižbe, katere pripadajo računu. Prav tako je predpisano zaporedje podatkov v eni vrstici. Podatki v vrstici so ločeni z znakom »/«. Za vsak podatek je določeno število znakov. Če določen podatek ne obstaja v zunanjem sistemu, moramo namesto podatka v datoteko zapisati toliko praznih znakov, kot je določena dolžina podatka.

Ko enkrat sestavimo datoteko, jo je potrebno na nek način prenesti na SAP strežnik. Ta postopek se lahko razlikuje glede na sistem, na katerem teče SAP, in na okolje na katerem imamo zunanji sistem. Če sta oba sistema na Windows platformi, lahko to storimo enostavno preko dovoljenj na datotečnem sistemu. Drug, bolj standarden način je ftp (file transfer protocol). To je protokol, ki je kar precej uveljavljen na spletu. Pozna ga večina platform (Windows, Linux …). Navaden FTP je lahko nevaren, če imamo omrežje povezano v svetovni splet. Navaden FTP je nevaren zaradi načina prenosa gesel. Protokol gesel ne kriptira, in jih lahko kdorkoli prestreže. Rešitev za to je SFTP ali FTPS, katera uporabljata

kriptiranje pri prenosu gesel. Čeprav so po navadi taki sistemi zaprti za svetovni splet, dodatna varnost ni odveč.

3.1.2 Dogodek (»Event«) na SAP

Dogodek nam služi za tako imenovan (t. i.) »trigger«. Preko dogodka lahko poženemo posel in potem naprej program.
Kreiranje dogodka:
Dogodek kreiramo s pomočjo transakcije SM62



Created By	Background Processing Event	Description
SAP	SAP_IMPORT_START	Change and Transport System: Start of an import
SAP	SAP_IMPORT_STOP	Change and Transport System: End of an import
SAP	SAP_KK1_UPDATE_FINISHED	Update of time events from CC1 complete
SAP	SAP_KK1_UPLOAD_FINISHED	Upload of time events using CC1 complete
SAP	SAP_LANGUAGE_FILL	Language Supplementation
SAP	SAP_LANGUAGE_IMPORT	Language import
SAP	SAP_MONITORING_STARTUP_DISPATCH	Event to Start Monitoring Start-Up Dispatching in Background
SAP	SAP_NEW_CONTROL_RECIPES	New control recipes created
SAP	SAP_NEW_PROCESS_MESSAGES	New process messages received
SAP	SAP_OPMODE_SWITCH	

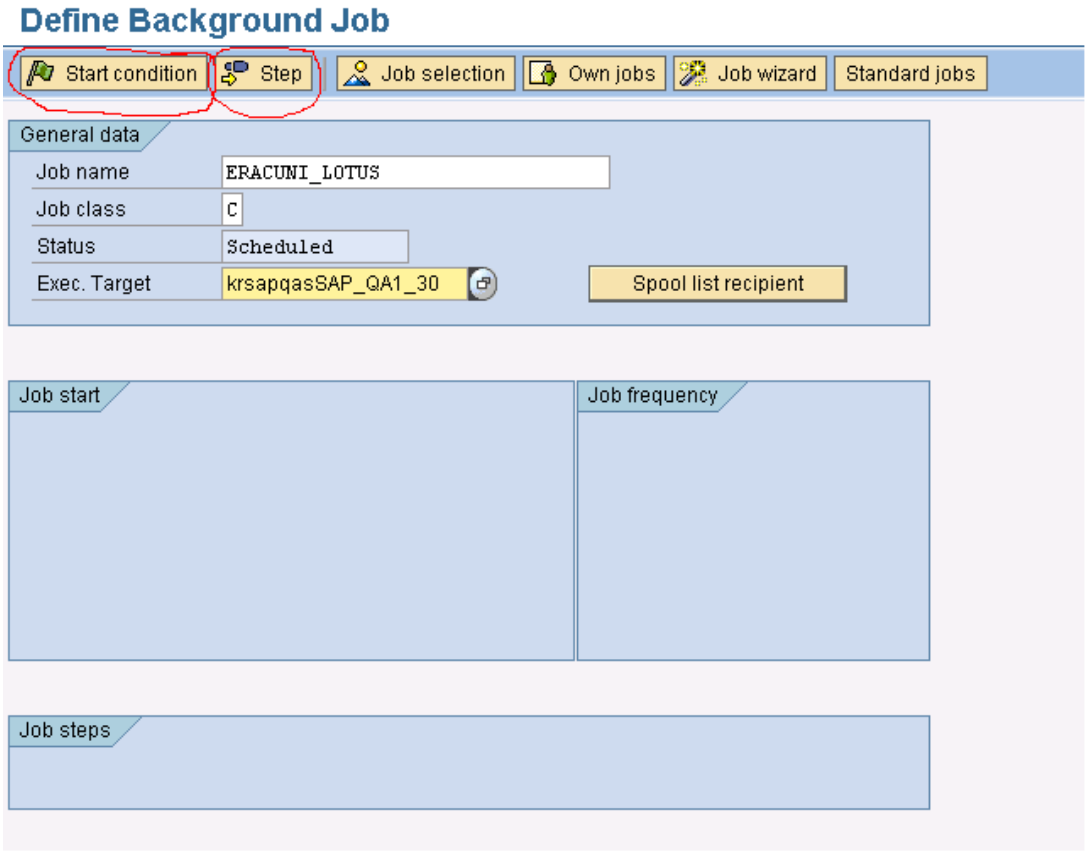
Slika 5: Ekran za definiranje dogodkov

Tukaj lahko tudi ročno prožimo dogodek za potrebe testiranja. Dogodek nam služi kasneje pri definiciji posla (job-a), na katerem določimo, kdaj se posel zažene.
Sedaj ko imamo kreiran dogodek, ga je potrebno še sprožiti. Kot je bilo že prej omenjeno, se datoteka prenese na strežnik na SAP. V praksi se kreira tudi cmd (izvršilna datoteka), v katero zapišemo ukaz za izvedbo dogodka. To storimo zaradi težav s pravicami. Sedaj pridemo do težave, kako pognati ukazno datoteko, ki smo jo prenesli na SAP. Tukaj si lahko pomagamo s programom »psexec«. Ta program omogoča proženje izvedljivih datotek na oddaljenem strežniku.

Programu lahko predamo različne parametre (uporabniško ime, geslo, delovna mapa …)
Parametri klica so definirani v dokumentaciji in pomoči programa psexec in jih ne bi razlagali v podrobnosti.
Torej moramo v datoteki »primer.cmd« zapisati klic dogodka, ki bo sprožil prenos. To storimo s pomočjo programa SAPEVT, ki je del SAP strežnika.
Ko smo torej sprožili dogodek, moramo definirati posel (job), ki bo poklical program, ki bo izvajal uvoz podatkov.

3.1.3 Posel (»Job«) na SAP

Posle kreiramo s pomočjo transakcije SM37.



Slika 6: Ekran za definiranje poslov

V definiciji posla definiramo podatke kot so:

- naziv posla,
- tip posla,
- strežnik na katerem se bo nahajal posel,
- pogoj, kdaj se sproži določen posel:

tukaj lahko navedemo prej kreiran dogodek, določen čas, lahko rečemo, naj se takoj izvede (enkratne operacije) ali po določenem poslu,

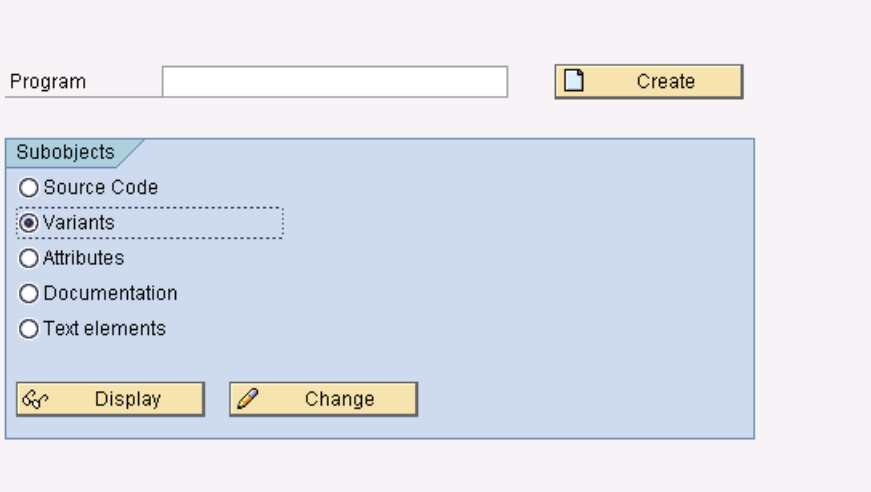
- korak (step):

Za korak lahko navedemo zunanji program, zunanji ukaz ali ABAP program, ki je opisan v naslednjem razdelku.

3.1.4 ABAP program

ABAP (*Advanced Business Application Programming*) je programski jezik, ki ga je razvil SAP za interno uporabo. Posebnost tega programskega jezika je način shranjevanja izvorne in preveden kode. Vsa koda je shranjena direktno v bazi in ne v ločenih datotekah. Podoben način shranjevanja kode uporablja tudi Lotus Notes. S pomočjo programov lahko izvajamo poljubne operacije na strežniku. Lahko dostopamo do baze podatkov, programsko kreiramo, prožimo posle in podobno.
Program navadno poženejo posli z določeno varianto. [1]

Programe urejamo s transakcijo NSE38:



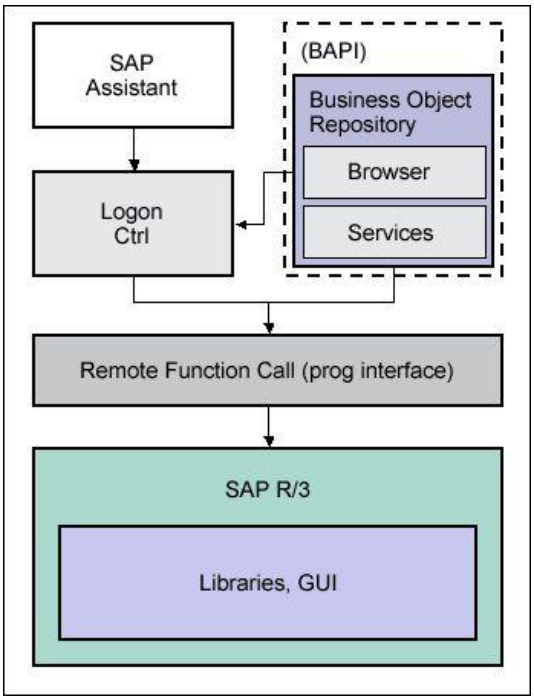
Slika 7: Ekran za urejanje ABAP programov

Za vsak program definiramo ustrezne variante programa. V variantah definiramo določene parametre za program. Definiramo lahko tudi variante za obstoječe SAP-ove programe. Eden izmed njih je RFBIBL00. Konkretno temu programu se bomo posvetili še kasneje.

3.2 BAPI (Business Application Programming Interface)

Ker ima prvi način kar nekaj pomanjkljivosti, so v podjetju SAP AG omogočili drug način za vnos podatkov iz zunanjega sistema. Ta način so poimenovali BAPI.
S pomočjo knjižnic, ki jih dobimo z namestitvijo SAP klienta, lahko dostopamo do poslovno povezanih funkcionalnosti. Torej imamo za določen poslovni proces že pripravljen t. i. BAPI, s katerim izvajamo operacije.
BAPI predstavlja odjemalca z objektno orientiranim pogledom na poslovne procese SAP-a. Torej se uporabniku ni potrebno ukvarjati z vsebinskim delom (pravilnost datumov, zneskov ...), ampak program (BAPI) sam skrbi za to.
Omogoča nam integracijo na poslovnem nivoju. Kako je to tehnično (podrobnosti) izvedeno, nas ne zanima, saj za to poskrbi SAP. Zaradi tega je taka integracija bolj zanesljiva.
[5]

V praksi so to knjižnice (dll datoteke), ki jih vključimo v projekt, kjer prevajamo izvorno kodo. Seveda je potrebno kasneje poskrbeti, da se bodo vse potrebne knjižnice prenesle na okolje, kjer se bo koda izvajala. Knjižnice je možno dobiti na njihovi uradni strani, ali jih pa pridobimo z instalacijo odjemalca za SAP.



Slika 8: Arhitektura povezovanja na SAP z BAPI-jem

Slabost tega načina povezovanja je pomanjkanje dokumentacije in pomoči za delo s temi objekti. Sicer lahko dostopamo do pomoči, ki obstaja na SAP strežniku, kar preko odjemalca za SAP in sicer s transakcijo »bapi«. Tukaj najdemo seznam vseh že obstoječih BAPI-jev. Prav tako najdemo grob opis parametrov, ki so potrebni za izvajanje določenega BAPI-ja.

SAP nam omogoča, da napišemo svoj BAPI, če nam noben izmed že napisanih ne ustreza. Postopek je kar zapleten in dolgotrajen. Prav tako moramo potem skrbeti za prenos našega

novega BAPI-ja na vse sisteme. Torej razvoj poteka na testnem sistemu (razvojno okolje), kasneje ko je program dobro pretestiran, pa se prenese na produkcijo. Na ta prenos morajo biti pozorni administratorji sistema. Če napišemo uporaben BAPI, ga lahko pošljemo na SAP in ga ta vključi v posodobitve sistema. Lahko pa naredimo le spremembo obstoječega BAPI-ja. Torej vzamemo del kode, ki ga lahko uporabimo, drug del pa napišemo sami.

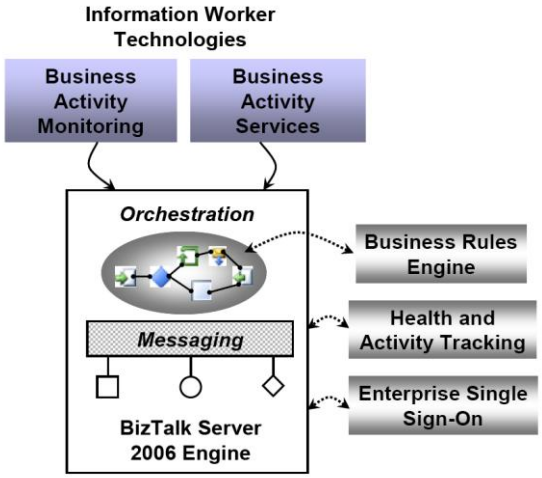
Za posamezen BAPI lahko rečemo da je to »črna škatla« (»black box«). Ne poznamo načina implementacije določene funkcije oz. akcije v SAP. Poznamo samo ime funkcije ter vhodne in izhodne parametre.
Glavni del pri implementaciji te integracije je poznavanje funkcij in njihovih parametrov. Potrebno je poznati proces v SAP-u**,** kamor želimo uvažati podatke oziroma jih črpati.

3.3 BizTalk server

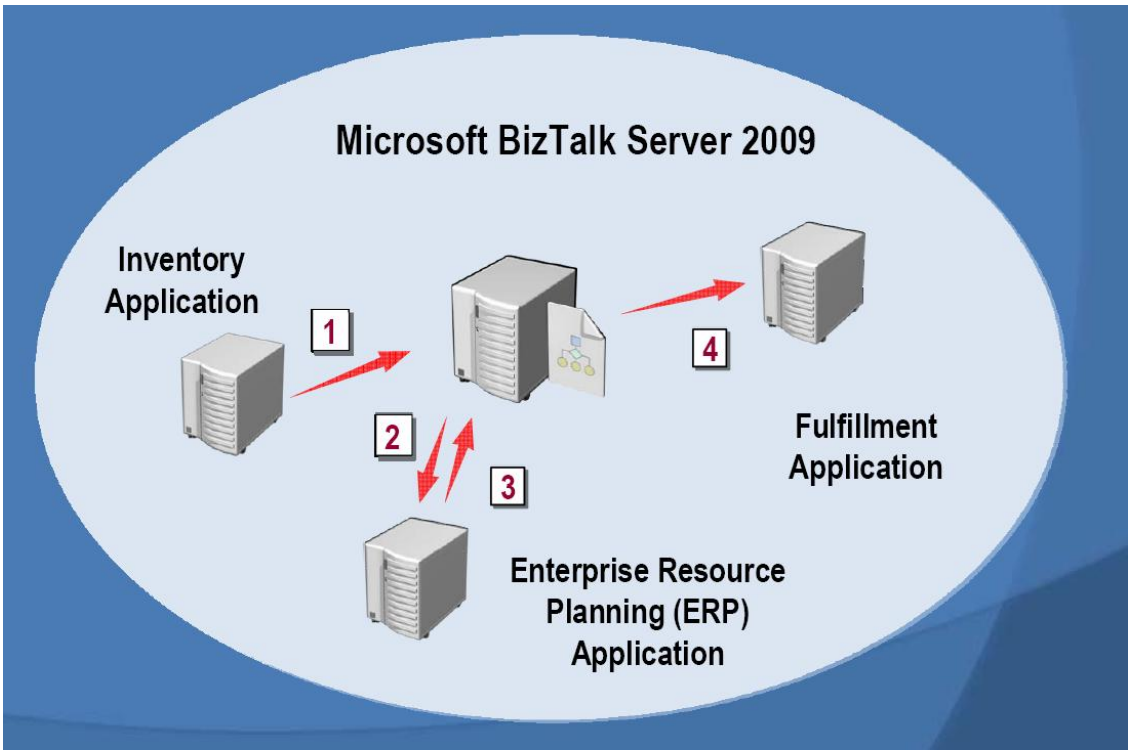
Kot tretji možni način lahko izberemo BizTalk strežnik. BizTalk strežnik lahko uporabljamo kot ESB (Enterprise Service Bus). Podpira povezovanje na različne podatkovne baze, datotečni sistem, elektronsko pošto in omogoča tudi povezavo na SAP. Omogoča nam grafično oblikovanje in nadziranje procesov.

Strežnik je sestavljen iz dveh pomembnejših komponent:
- Komponenta za sporočila nam omogoča komunikacijo z veliko različnimi sistemi. Za vsako različno povezavo moramo namestiti ustrezen adapter, ki nam omogoča komuniciranje.
-Drugi del je pa komponenta, ki nam omogoča grafično kreiranje in nadziranje procesov. To imenujemo orkestracija. V orkestraciji implementiramo logiko, ki vodi cel delovni proces ali pa le določen del.
[4]

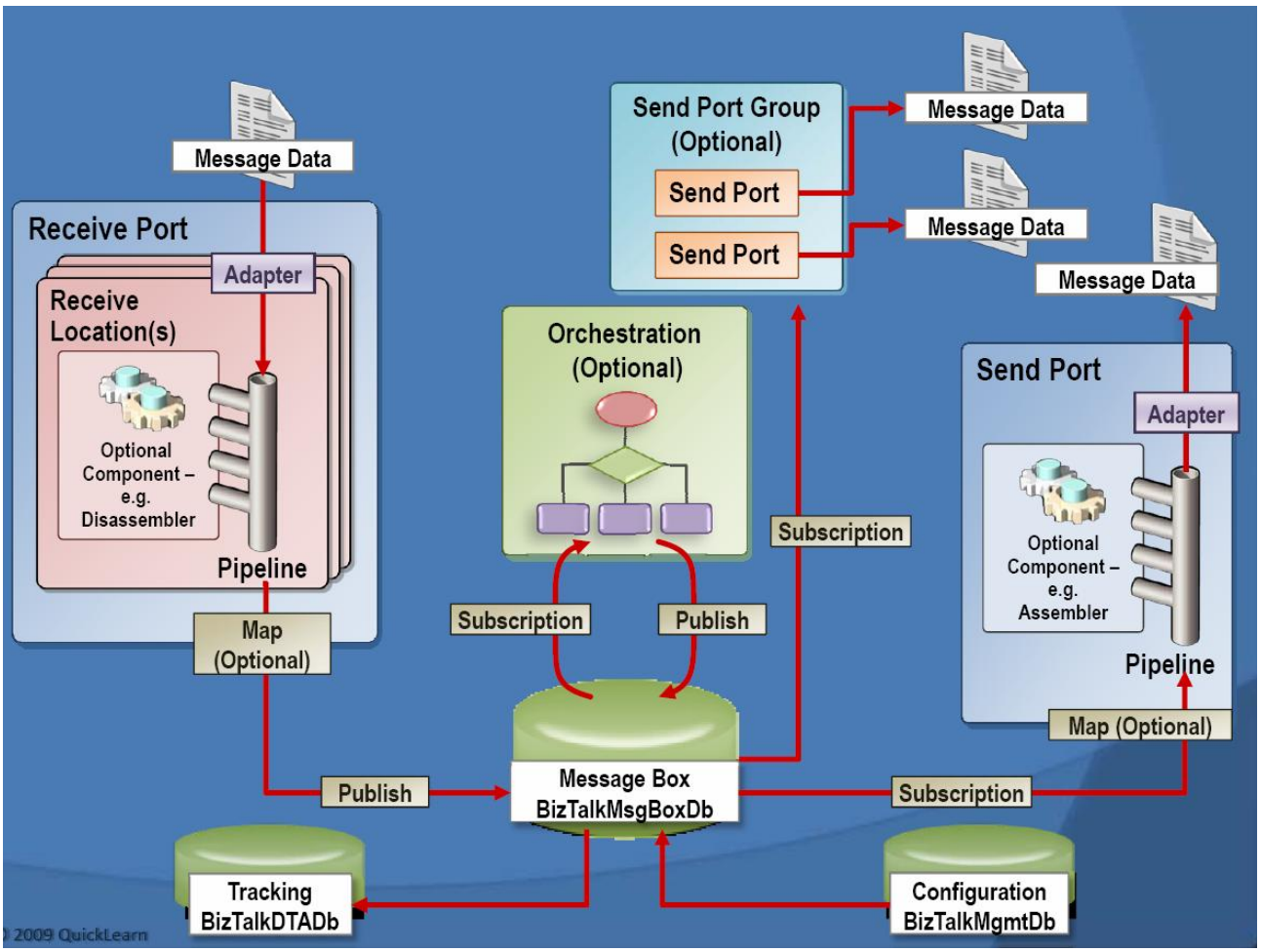
Strežnik se lahko povezuje tudi s t. i. Single Sign-On, ki omogoča enkratno avtentikacijo uporabnika, povezovanje s poslovnimi pravili, ki nam omogočajo kompleksno obravnavo poslovnih pravil.



Slika 9: Delovanje BizTalk strežnika



Slika 10: Preprosta skica integracije med različnimi sistemi s pomočjo BizTalk strežnika



Slika 11: Kompleksnejša skica delovanja sistema:

Razlaga delovanja strežnika:
Strežnik pošilja in prejema sporočila. Tako lahko rečemo, da preko *receive adapterja* pride sporočilo v sistem. Različni adapterji omogočajo različne načine komunikacije. Torej lahko vhodno sporočilo pride preko spletnega servisa, iz datoteke ali na kakšen drug način. Sporočilo se potem pretransformira v *receive pipeline-u.* Transformacija se izvede iz njegove izvirne oblike v XML dokument. Dokument se potem prenese v podatkovno bazo imenovano *MessageBox*, ki je implementirana v SQL strežniku.

Logika, ki vodi poslovni proces, je implemementirana v eni ali več orkestracij. V nasprotju s pričakovanjem tukaj ne pišemo programske kode v programskem jeziku kot npr. VB, C#, ampak celoten poslovni proces modeliramo z grafičnim vmesnikom. Lahko definiramo zanke, pogoje in ostalo obnašanje procesa.
Vsaka orkestracija kreira *subscription*. S tem definira, kakšna sporočila bo orkestracija sprejemala. Ko določeno sporočilo prispe v bazo sporočil, sistem preusmeri sporočilo pravilni orkestraciji. Orkestracija potem obdela sporočilo in po navadi je rezultat obdelave novo sporočilo, ki se shrani v bazo. To sporočilo se pretvori iz XML oblike, ki jo uporablja BizTalk strežnik, v obliko, ki je definirana za izhod iz orkestracije. Prav tako lahko doda podpis itd.
To preoblikovanje naredi *pipeline.* Samo pošiljanje pa izvede *send adapter*.
[4]

3.3.1 BizTalk Adapters

Odgovorni člen za komunikacijo med različnimi sistemi so adapterji. Za vsak tip komunikacije moramo namestiti ustrezen adapter. Le-ti so na voljo na spletu in so brezplačni, če imamo postavljen strežnik.
Zadnjo verzijo (3.0) je možno uporabljati preko WCF (Windows Communication Foundation). WCF lahko uporablja SOAP sporočila med dvema procesoma, zato pravimo da je WCF interoperabilen s procesi, ki uporabljajo sporočila v SOAP obliki.

4 Primeri različnih integracij

Kot sem že na začetku omenil, bom poizkušal izvesti prenos prejetega računa iz nekega zunanjega sistema v ERP sistem SAP.

4.1 Prenos prejetega računa preko txt datoteke in Job-a

Kot že sam naslov načina prenosa pove, gre za prenos preko tekstovne datoteke. Torej nek zunanji program kreira datoteko po pravilih, kot jih določa SAP.

4.1.1 Datoteka

Primer datoteke:
1.vrstica:
0FI-DOC 500UPORABNIK 00000000//

Vrstica je *batch* vrstica in vanjo vpišemo: tip dokumentov, ciljni sistem in uporabnika.
2.vrstica:
1FB01 03082009DS000104122009/ EUR / 7120017043/ 0900004 / 03.08.2009/10.08.2009 / / / // / / / / // / ///*
V to vrstico vpišemo podatke iz glave računa. Nekaj podatkov iz glave računa: datum izstavitve računa, datum valute, valuta, sklic, opis ...

3.vrstica:
*2BBSEG 21/ // 92,10 / / / //// / / / / / / / / / / / / / / / / / / 03082009/ / / /*3000-0900004* // / // / N060/ / / / // / / // /// / / / // / // /*
Nadaljujejo se vrstice tipa 2BBSEG. To so vknjižbe računa. Na primeru manjka precej podatkov, saj jih je ogromno. Definiramo šifro knjiženja, konto, znesek, opis ...
Teh vrstic je lahko več, odvisno od tega, na koliko različnih stroškovnih mest je porazdeljen račun.

4.vrstica:
2BBTAX 15,35 R250/ / / / / / /
Ta vrstica ni nujno, da obstaja. To je vrstica**,** v kateri navedemo davek, ki je naveden na računu.

Kot smo že v začetku povedali, gre v tem primeru za paketno obdelavo. Torej bomo hkrati prenašali več računov. Torej se bodo vrstice 1FB01, 2BBSEG in 2BBTAX ponavljale. Seveda odvisno od tega, koliko računov želimo prenesti naenkrat.

4.1.2 Dogodki

Ko imamo datoteko kreirano, jo moramo prenesti na strežnik. Kot smo že v samem opisu povedali, je najbolj univerzalen način preko FTP protokola.

Ko enkrat prenesemo datoteko na strežnik, kjer teče SAP, je potrebno sprožiti dogodek, ki bo pričel z izvajanjem prenosa. Seveda je prej potrebno definirati ustrezen dogodek na SAP strežniku.
Dogodke kreiramo s transakcijo SM62. Za primer prenosa računa kreiramo dva dogodka: *ERACUNI* in *ERACUNI_SAP*:

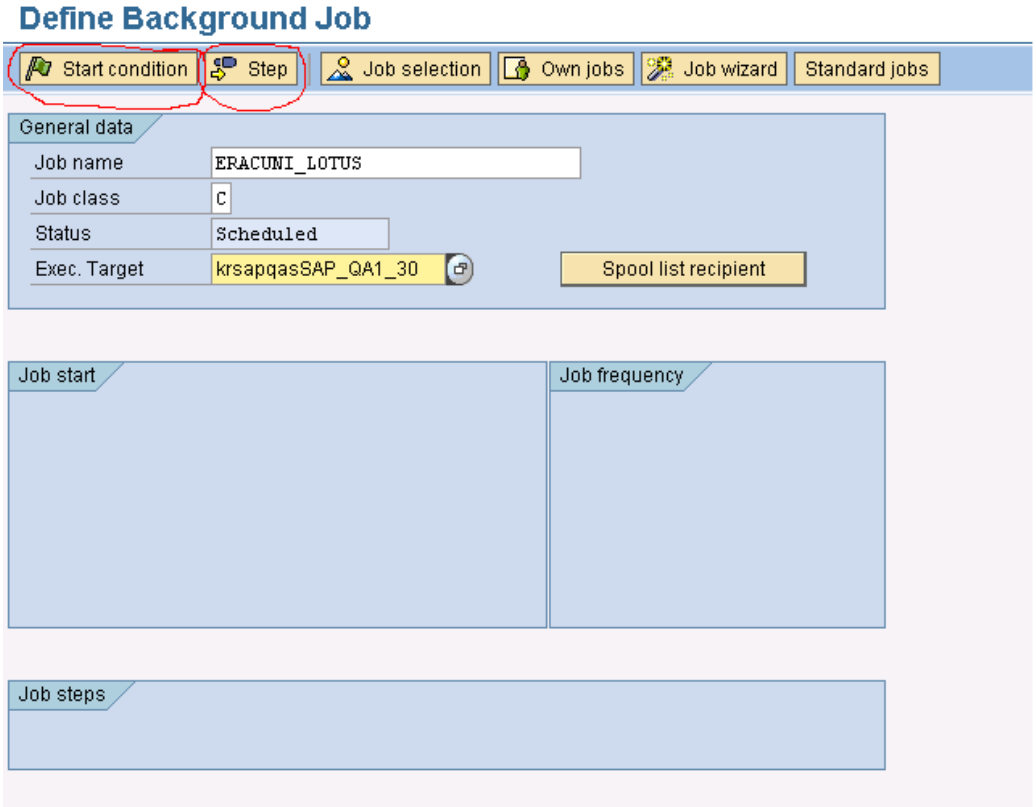
Slika 12: Vnos novega dogodka

Dogodek sprožimo s pomočjo SAPEVT programa, ki je del SAP-a.
Primer klica:
G:\usr\sap\QA1\sys\exe\run\SAPEVT.EXE /COMPANY/ERACUNI_LOTUS -p '200609131520' name=QA1 nr=30
Ker se SAP strežnik po navadi nahaja nekje na drugem sistemu, moramo proženje dogodka izvesti malo naokoli. Na strežnik pošljemo ukazno datoteko, kjer smo navedli vse parametre za klic dogodka. Torej vse, kar moramo narediti, je to, da s pomočjo programa *psexec* izvedemo ukazno datoteko, ki smo jo prej prenesli na strežnik.
Primer klica ukazne datoteke primer.cmd na strežniku krsapqasap

c:\rexec\psexec.exe \\krsapqassap -u domain\qa1adm -p simsap -d -w g:\temp g:\temp\prenos300.bat

4.1.3 Posli

Definicije poslov izvajamo s transakcijo SM36. Torej dogodek ERACUNI sproži posel ERACUNI_LOTUS. Oglejmo si ekransko sliko definicije posla:



Slika 13: Defnicija prvega posla

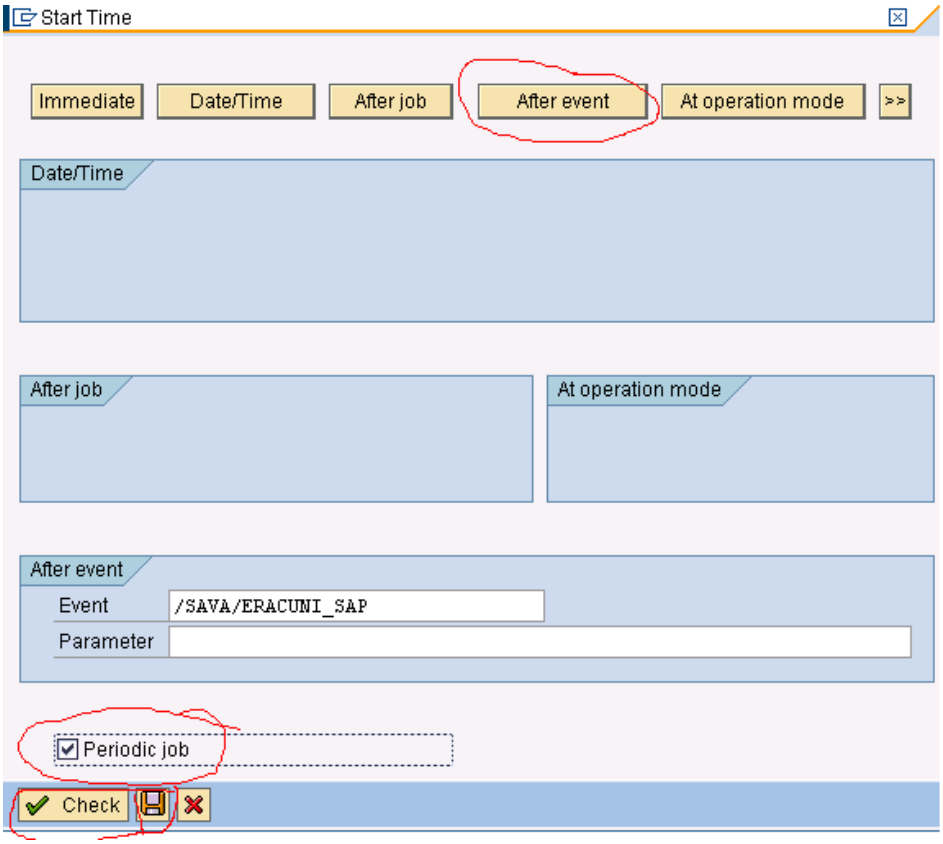
Definiramo potrebne parametre:
Ime posla, strežnik kjer se bo izvajal posel.
Za začetni pogoj izberemo dogodek ERACUNI, ki smo ga prej definirali. Izberemo tudi možnost, da dovoljujemo periodično izvajanje (*periodic job*).
Za korak izberemo ABAP program ERACUNI z ustrezno varianto. V parametrih vpišemo ustreznega uporabnika, ime in varianto. Ta program bomo razložili kasneje.
Ko smo vnesli vse zahtevane parametre, moramo posel še shraniti.

Prvi posel izvede preverjanje datoteke. Torej, ko smo uspešno definirali prvi posel, moramo narediti definicijo še drugega, ki pa bo dejansko izvedel prenos podatkov.
Definicija drugega posla je precej podobna. Definicija drugega posla se izvaja preko druge transakcije kot za prvi posel. Razlog za to je napaka na SAP-u. Definicijo izvajamo s transakcijo BMV0.

Slika 14: Definicija drugega posla

Ko izberemo »define job« vpišemo podatke o poslu:
Ime direct input joba: "ERACUNI"
Program: "RFBIBL00"
Varianta: "ERACUNI"
Upor.ime: "SNOVAK"
Dovol. periodično: "DA"

Ko smo vnesli podatke, izberemo »start job«. Potem je potrebno vnesti parametre za zagon posla ...



Slika 15: Določanje pogojev za zagon posla

Dejansko sta oba posla precej podobna, le definicija drugega je narejena malce naokoli. Za usklajeno delovanje skrbi zgoraj omenjeni ABAP program ERACUNI.

4.1.4 Program ERACUNI

Da bi delovanje prenosa računov potekalo nadzorovano skrbi program, ki se izvaja na SAP strežniku. Napisan je v programskem jeziku ABAP.
Razlaga delovanja programa:
Program najprej kreira nov posel, ki preveri strukturo datoteke. Če je struktura vredu, se potem pokliče posel, ki dejansko prenese podatke v sistem. Oziroma se izvede dogodek, ki potem prične izvajati posel za uvoz podatkov.
Preverjanje se izvaja zato, ker se v primeru napačne datoteke posli zaklenejo. Ko so posli enkrat zaklenjeni, jih mora najprej administrator ustaviti in izbrisati. Potem jih je potrebno na novo kreirati. [1]

Programski jezik je zelo specifičen, zato tudi ne bomo razlagali celotne kode programa. Pojasnili bi le nekaj pomembnejših delov programa.

Kreiranje novega posla:

```
DATA:
    jobname LIKE bapixmjob-jobname,
    extuser LIKE bapixmlogr-extuser,
    jobcount LIKE bapixmjob-jobcount,
    reportid LIKE bapixmrep-reportid,
    variantnam LIKE bapixmrep-variantnam,
    execserver LIKE bapixmjob-execserver,
    status LIKE bapixmjob-status,
    sapuser LIKE bapixmstep-authcknam.
CONCATENATE 'ERACUNI_' eventpar(12)   INTO jobname.
extuser = 'SNOVAK'.
sapuser = 'SNOVAK'.
CALL FUNCTION 'BAPI_XBP_JOB_OPEN'
    EXPORTING
        jobname          = jobname
        external_user_name = extuser
    IMPORTING
        jobcount         = jobcount
        return           = bapiret2.
IF NOT bapiret2-type IS INITIAL.
  PERFORM write_bapiret2.
  EXIT.
ENDIF.
reportid = 'RFBIBL00'.
variantnam = 'ERACUNI400'.

CALL FUNCTION 'BAPI_XBP_JOB_ADD_ABAP_STEP'
```

```
  EXPORTING
    jobname                = jobname
    jobcount               = jobcount
    external_user_name     = extuser
    abap_program_name      = reportid
    abap_variant_name      = variantnam
    sap_user_name          = sapuser
  IMPORTING
    return                 = bapiret2
```

Proženje dogodka:

```
  CALL FUNCTION 'BP_EVENT_RAISE'
    EXPORTING
      eventid                = 'ERACUNI_SAP'
    EXCEPTIONS
      bad_eventid            = 1
      eventid_does_not_exist = 2
      eventid_missing        = 3
      raise_failed           = 4
      OTHERS                 = 5
          .
  IF sy-subrc <> 0.
    WRITE: /'Napaka pri startanju dogodka  ERACUNI_SAP!'.
    EXIT.
  ENDIF.
```

Torej ta program skrbi za usklajeno delovanje prenosov. Ker pa vedno potrebujemo še informacijo, ali so se računi prenesli brez napak, potrebujemo povratno informacijo. Povratna informacija se nahaja v log-u posla, ki izvaja prenos v SAP. To je posel ERACUNI.
Poslu pa posreduje podatke program RFBIBL00. Vidne so opombe in opozorila pri izvajanju programa. V izpisu je vidna SAP-ova interna številka računa in akcija, ki je bila izvedena. Torej imamo zapisane vse pravilno prenesene račune. Prav tako pa sistem javi napako, če npr. določen konto ni odprt ali podobno.
Ker moramo te zapise nekako spraviti v zunanji sistem, smo na SAP-u kreirali novo tabelo, kamor potem s pomočjo programa ERACUNI zapisujemo podatke. To je ločena tabela od SAP-ovih. Seveda je s tem lahko kar nekaj težav, saj nihče noče kreirati dodatnih tabel, ki jih je potrebno vzdrževati ročno. Torej lahko nastane težava ob morebitni nadgradnji sistema. Vendar druge, »elegantnejše« rešitve ni, oziroma jo ne poznamo.

Primer izpisa log-a posla:

Date	Time	Message text	Msg class	Msg no.	Msg type
04.12.2009	11:07:12	Job started	00	516	S
04.12.2009	11:07:12	Step 001 started (program RFBIBL00, variant	00	550	S

		ERACUNI500, user ID SNOVAK)			
04.12.2009	11:07:13	ERACUNI: This job was started periodically or directly from SM36/SM37	BD	076	I
04.12.2009	11:07:14	Session 1: Special character for 'empty field' is /	FB	012	I
04.12.2009	11:07:14	Doc. : Record end indicator in table BBKPF was not supplied with /	FB	174	I
04.12.2009	11:07:14	... Table BBKPF was extended	FB	023	I
04.12.2009	11:07:14	... Before the next put,	FB	024	I
04.12.2009	11:07:14	... also maintain the new fields in table BBKPF	FB	025	I
04.12.2009	11:07:15	Period 011/2009 is not open for account type K and G/L 221000	F5	286	I
04.12.2009	11:07:15	Period 011/2009 is not open for account type K and G/L 221000	F5	286	I
04.12.2009	11:07:15	Period 011/2009 is not open for account type K and G/L 221000	F5	286	I
04.12.2009	11:07:15	Session 1: Special character for 'empty field' is /	FB	012	I
04.12.2009	11:07:15	Session 1 session name FI-DOC was opened	FB	007	I
04.12.2009	11:07:15	Session 1 session name FI-DOC was created	FB	008	I
04.12.2009	11:07:15	Job finished	00	517	S

4.2 Prenos prejetega računa z vmesnikom BAPI

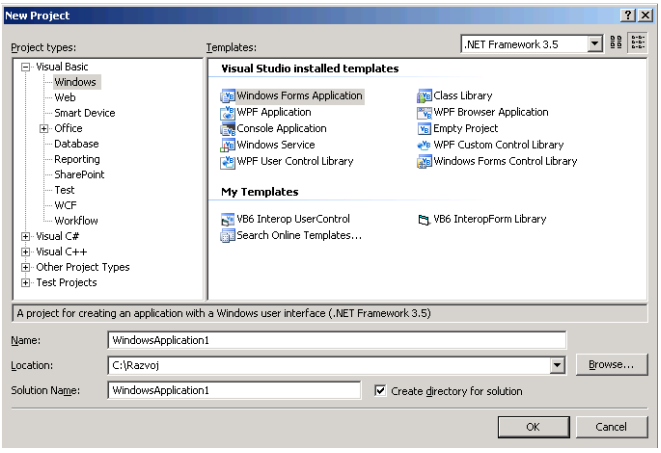
Kot smo že v opisu samega načina integracije omenili, gre tukaj za direktni dostop do akcij, dokumentov preko objektnega modela. Seveda obstajajo določene omejitve, katere bomo spoznali skozi podrobno analizo. Že na začetku pa lahko rečemo, da je precej bolj enostaven za uporabo in administracijo. To trditev pa lahko potrdimo z dejstvom, da za uporabo BAPI-ja ni potreben noben administrativni poseg na strežniku. Torej če primerjamo to metodo s prejšnjo vidimo, da nam ni potrebno nastavljati nobenega posla, programov ...

4.2.1 Priprava okolja

Razvoj integracij s pomočjo BAPI-jev je najbolj razširjen na Microsoftovi platformi. Jezik, ki je uporabljen, je Microsoft visual basic. Torej potrebujemo Microsoft Visual studio za pisanje in prevajanje potrebne kode. Ker pa uporabljamo SAP-ove knjižnice, moramo imeti nameščen še SAP klient. Prav tako moramo imeti dostop do nekega testnega SAP strežnika. Zaradi pomanjkanja dokumentacije nam t. i. debugiranje zelo olajša razvoj. Tako lahko spremljamo izvajanje programa in morebitne lokacije napak. Prav tako so zelo lepo razvidne vrednosti, ki jih SAP vrača ob izvedbi določene akcije/transakcije.

4.2.2 Kreiranje projekta v Visual studio

Za potrebe testiranja prenosa, kreiramo nov projekt tipa »windows forms application«. V dejanski uporabi bi morali kreirati nek vmesni modul med aplikacijo in SAP-om. Najprimernejša bi bila spletna storitev (WEB service). Vendar, ker naš namen ni izdelati aplikacijo, ki bi bila 100% uporabna v realnem svetu, izberemo lažjo varianto z Windows forms aplikacijo.



Slika 16: Kreiranje novega projekta v Visual Studiu 2008

Ko smo ustvarili nov projekt, moramo dodati še knjižnice za dostop do SAP strežnika. Knjižnice se nahajajo v programskih datotekah nameščenega SAP klienta. Za sam razvoj ni nujno, da imamo nameščen SAP klient, lahko samo prekopiramo zahtevane knjižnice. Za testiranje in povezovanje na strežnik pa je potrebna instalacija.

4.2.3 Priprava povezave na SAP

Ko smo enkrat pripravili projekt, lahko pričnemo s programiranjem. Za začetek je potrebno vzpostaviti povezavo na objekt SAP.

```
oBapiControl = CreateObject("SAP.BAPI.1")
```

To je najvišji element, preko katerega dostopamo do BAPI-jev. Torej če imamo okolje ustrezno pripravljeno, bi se program, ki vsebuje le to vrstico kode, moral izvesti, brez da sistem javi napako.

Preverjanje, če je prišlo do napake:

```
If Err.Number <> 0 Then
   Potek = "BAPI kontrola ne obstaja"
   oBapiControl = Nothing
   Exit Sub
End If
```

Err je standardni objekt v windows aplikaciji. Če pride do napake, se v Err.Number napolni številka napake.

Ko smo uspešno kreirali objekt SAP, moramo vzpostaviti povezavo s strežnikom.

```
Dim oLogonCtrl As Object
oConnection = oBapiControl.Connection
oLogonCtrl = CreateObject("SAP.Logoncontrol.1")
oConnection = oLogonCtrl.NewConnection

oConnection.ApplicationServer = "krsapqassap"
oConnection.Client = "300"
```

```
oConnection.Language = "E"
oConnection.User = "user"
oConnection.Password = "password"
oConnection.SystemNumber = "30"

If Not oConnection.Logon(1, True) Then
   oConnection = Nothing
   bLoggedOn = False
Else
   bLoggedOn = True
End If
Logon = bLoggedOn
oBapiControl.Connection = oConnection
```

4.2.4 Kreiranje potrebnih objektov fakture

Za prenos računa/fakture se uporablja objekt *Incoming invoice*. Objekt kreiramo po spodnjem primeru.

```
SapOrder = oBapiControl.GetSAPObject("IncomingInvoice")
```

Kreirati je potrebno še glavo dokumenta, kamor bomo vnesli osnovne podatke o računu.

```
SapOrderHeader = oBapiControl.DimAs(SapOrder, "CreateFromData", "Headerdata")
```

Objekt davkov

```
SapTaxItems = oBapiControl.DimAs(SapOrder, "CreateFromData", "Taxdata")
```

Objekt knjižb

```
SapCountItems = oBapiControl.DimAs(SapOrder, "CreateFromData", "GlacCountdata")
```

4.2.5 Polnjenje struktur

Ko imamo enkrat definirane strukture, potrebne za prenos, jih je potrebno še pravilno napolniti. To zna včasiih predstavljati problem, saj ni točne dokumentacije, kaj in v kakšni obliki mora biti napolnjeno. Precej problemov izhaja tudi iz vsebine podatkov. Lahko določena šifra dobavitelja ne obstaja, stroškovno mesto ne obstaja. Poznati moramo tudi obliko podatkov, število mest šifer ...

Nekaj primerov polnjenja podatkov v strukturi:

```
SapOrderHeader.Value("HEADER_TXT") = "Besedilo glave"
SapOrderHeader.Value("PAYMT_REF") = "0123456" 'model + sklic
```

Davek:

```
SapTaxRowNew = SapTaxItems.rows.Add
SapTaxRowNew("TAX_CODE") = "D1" 'šifra davka
```

Knjižbe:
```
SapCountRowNew = SapCountItems.rows.Add
SapCountRowNew("INVOICE_DOC_ITEM") = "1" 'Zaporedna številka
SapCountRowNew("GL_ACCOUNT") = "000040000" 'Konto
SapCountRowNew("ITEM_AMOUNT") = "12,34" 'znesek
SapCountRowNew("DB_CR_IND") = "H" 'indikator debet/kredit
```

4.2.6 Izvajanje transakcije

Ko imamo vse strukture napolnjene, sledi klic oziroma vstavljanje transakcije.

```
SapOrder.CreateFromData(Headerdata:=SapOrderHeader, _
ItemData:=SapTabItems, taxdata:=SapTaxItems, glaccountdata:=SapCountItems, _
Accountingdata:=SapAccountingItems, Return:=oReturn)


If Err.Number <> 0 Then
   oBapiControl = Nothing
   Exit Sub
End If
```

Ker pri kreiranju transakcije lahko pride do vsebinskih ali ostalih, moramo preveriti strukturo, ki jo vrne metoda *CreateFromData.*

```
If oReturn.rows.Count > 0 Then
   IzpisNapake = oReturn.rows.Parent.Data
   Exit Sub
End If
```

V objekt, ki ga vrača ta metoda, dobimo izpis napake v več vrsticah. Sporočila so zelo podobno kot tista, ki jih dobi uporabnik, če vnaša preko SAP klienta. Torej pri samem razvoju programa potrebujemo tudi nekoga, ki pozna SAP oz. tisti del SAP-a, katerega hočemo integrirati.

```
oCom = oBapiControl.GetSAPObject("BapiService")
oCom.TransactionCommit("0", oReturn)
```

Ko je izvajanje programa uspešno končano, sledi še odjava iz sistema:
```
oConnection.Logoff()
oConnection = Nothing
```

Glavna kontrola pravilnega izvajanja programa, je na programskem nivoju. Torej če pri nobenem od klicu funkcij ni prišlo do napake, je račun pravilno prenešen v SAP.
Pri tej metodi se pojavi težava, če uporabljamo interno številčenje. SAP pozna dve vrsti številčenja.

Interno številčenje:
Sap samodejno dodeljuje interne številke dokumentom. To pomeni, da se zunanjemu programu ni potrebno ukvarjati s številčenjem. Problem nastane, ko želimo v zunanjem sistemu dobiti številko dokumenta v SAP-u. Te številke ni mogoče dobiti, če izvajamo prenos z BAPI-jem. Sistem ne vrne interne številke. To je velik minus, ker povezavo preko številke potrebujemo. Obstajajo tudi drugi podatki, preko katerih bi bila možna povezava, vendar niso tako zanesljivi kot interna številka. To se sicer da rešiti na drug način, vendar je precej več dela in tudi performančno ni ravno ugodno. Rešitev bi bila ta, da bi naredili poizvedbo v bazo po nekem drugem unikatnem podatku. Potem bi pa lahko pridobili tudi interno SAP številko računa. Vendar smo spet pri bazi, kjer potrebujemo bralni dostop do SAP tabel.

Eksterno (zunanje) številčenje:
V tem primeru za številčenje skrbi zunanji sistem. Pri polnjenju podatkov je potrebno napolniti podatek o interni številki. Vendar moramo paziti, saj ne smemo dvakrat poslati iste številke v SAP. V primeru, da ključ že obstaja, nam program vrne napako. Vendar nam ob takem načinu številčenja ni potrebno skrbeti za interno številko, saj jo že imamo.

Če pogledamo programsko kodo, ki je potrebna za prenos računa, lahko hitro ugotovimo, da jo je zelo malo (čeprav zgoraj ni napisan cel program). Prav tako je zelo enostaven za administracijo. Prehod med testom in produkcijo je čisto preprost. Le popravimo strežnik in številko klienta in že pošiljamo podatke na produkcijski strežnik.

4.3 Prenos prejetega računa z BizTalk strežnikom

Če hočemo narediti še tretji način prenosa, je potrebno pripraviti kar precej programske opreme.

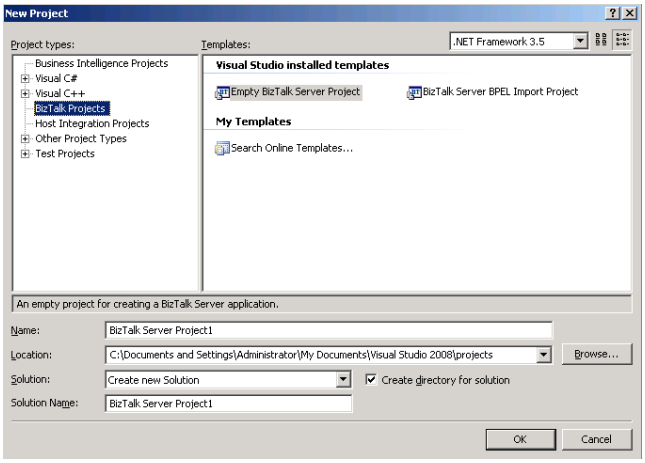
4.3.1 Zahteve za izvedbo integracije

Potrebujemo delujoč BizTalk strežnik. Prav tako mora biti na strežniku nameščeno glavno orodje za razvoj. To je Microsoft Visual Studio (najmanj verzija 2008). Nameščen mora biti paket vtičnikov (adapter pack). Za konkreten primer mora biti nameščen SAP vtičnik.
Pri podrobni raziskavi je SAP adapter razvit tako, da načeloma niti ne potrebujemo BizTalk strežnika za povezavo na SAP. Če izberemo ta način, je sistem dejansko brez orkestracije Adapter za SAP nam omogoča izmenjavo t. i. IDoc dokumentov in pa klic BAPI-jev, ki smo jih obdelali že v prejšnjem poglavju. Torej je ta adapter le nekakšna ovojnica za BAPI-je, ki jih že poznamo.
Lahko bi se lotili tudi prenosa računa preko IDoc-a.

IDoc *(intermediate document)* je standardna podatkovna struktura za elektronsko izmenjavo podatkov med SAP sistemom in zunanjimi programi. Način je uporabljen za asinhron način prenosa podatkov, za razliko od BAPI-ja ki je sinhron način prenosa. [5]

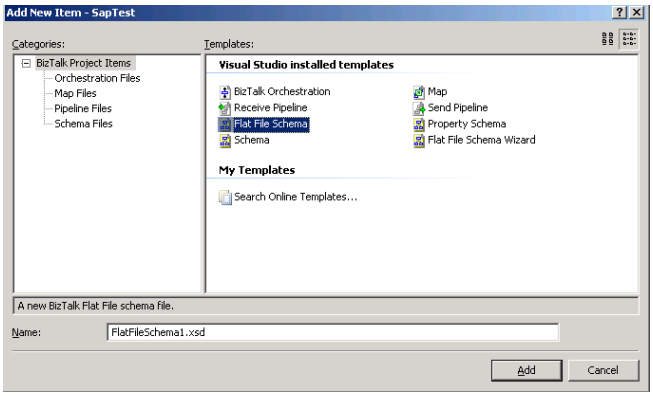
4.3.2 Kreiranje strukture vhodnih podatkov

Najprej moramo kreirati nov projekt v Visual Studiu 2008. Tip izberemo »BizTalk Project«



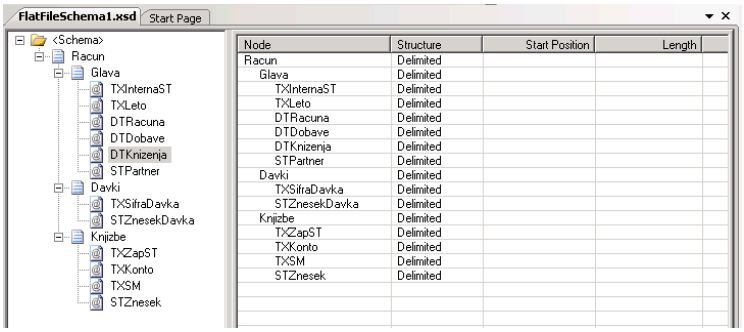
Slika 17: Kreiranje BizTalk projekta

Ko imamo definiran nov projekt, je potrebno dodati strukturo vhodne datoteke:



Slika 18: Dodajanje strukture vhodne datoteke

Definicija strukture (niso vpisani vsi potrebni atributi):



Slika 19: Prikaz atributov vhodne datoteke

Primer generirane vhodne datoteke

```
<ns0:Racun xmlns:ns0="http://SapTest.FlatFileSchema1">
    <Glava TXInternaST="6" TXLeto="TXLeto_1" DTRacuna="1999-05-31" DTDobave="1999-05-31" DTKnizenja="1999-05-31" STPartner="STPartner_5" />
    <Davki TXSifraDavka="TXSifraDavka_0" STZnesekDavka="STZnesekDavka_1" />
    <Knjizbe TXZapST="TXZapST_0" TXKonto="TXKonto_1" TXSM="TXSM_2" STZnesek="STZnesek_3" />
</ns0:Racun>
```

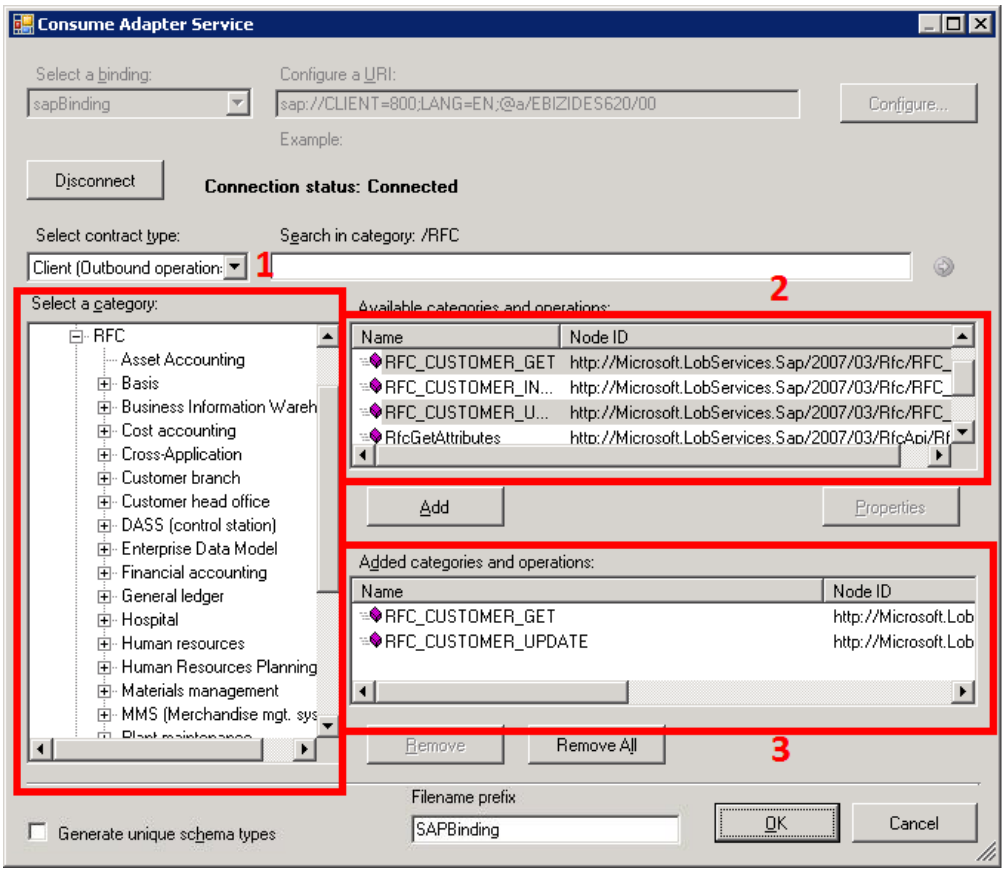
Zgoraj je prikazan primer vhodne datoteke. Sedaj moramo pridobiti še shemo podatkov za BAPI za parkiranje računa.

Ker pa za ta korak potrebujemo direktno povezavo na SAP, tega koraka praktično ni bilo mogoče izvesti. Težava je bila v povezavi iz strežnika, kjer teče BizTalk preko določenih vrat na testni SAP strežnik. Zaradi varnostnih razlogov mi tega v podjetju, kjer so mi omogočili testni dostop do SAP, niso mogli odobriti. Zato bomo od tukaj naprej govorili o predvidenih postopkih za izvedbo integracije. Vsi nadaljnji postopki so povzeti iz primera na spletu [6]

4.3.3 Generiranje sheme izhodnih podatkov

Za pridobitev sheme za prenos računa bi uporabili dodatek v Visual studiu: Consume Adapter Service BizTalk Project Add-in. S tem dodatkom poiščemo pravi BAPI in generiramo XSD shemo.

Ker poznamo BAPI za vnos vhodnega računa, ne bo problem generirati pravo strukturo.



Slika 20: Ekranska slika dodatka za generiranje strukture izhodnih podatkov

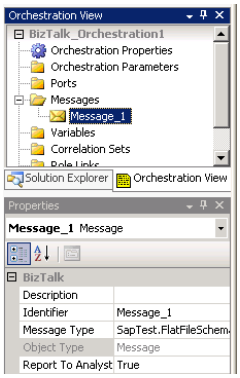
Potrebno je najprej nastaviti povezavo s SAP strežnikom – definirati številko klienta, jezik, itd. Ko imamo enkrat povezavo s strežnikom vzpostavljeno, lahko na levi strani brskamo po BAPI-jih (Okvir številka 1). Vendar za uspešno brskanje moramo kar dobro poznati kategorije, ki jih določa sistem. Lahko pa tudi uporabimo iskanje, ki je omogočeno.
V okvirju številka 2 izberemo pravi BAPI oz RFC (Remote Function Call). Spodaj (okvir številka 3) se nam prikažejo možni klici funkcij, npr. Get, Update ...
Pri generiranju shem se hitro pojavi problem podvojenosti podatkov. Ker ponavadi ne generiramo le ene sheme, tukaj nastane težava, ker imata dve funkciji enake parametre. Zato moramo paziti, da generiranje izvedemo le enkrat (ne zapiramo Consume Adapter) in obvezno izberemo možnost »Generate unique schema types«. To pomeni, da bo pred vsakim elementom podal unikaten »namespace«.
Ko smo končali z dodajanjem želenih funkcij, potrdimo generiranje datoteke.

Zgornji dodatek lahko uporabimo tudi za generiranje kode za povezovanje na SAP preko WCF. Kot smo že omenili, lahko BizTalkAdapter uporabimo kot WCF protokola povezovanja.

Torej sedaj imamo dve shemi, vhodno in izhodno shemo. Potrebno bo definirati še shemi za t. i. COMMIT in ROLLBACK. To naredimo isto s Consume Adapter dodatkom.

4.3.4 Definiranje sporočil v sistemu

Najprej je potrebno dodati datoteko orkestracije. Torej imamo v projektu definirano orkestracijo. Za delovanje sistema je potrebno določiti sporočila, ki se prenašajo v sistemu. Za vsako sporočilo je potrebno definirati shemo, kateri pripada.



Slika 21: Pregled sporočil v projektu

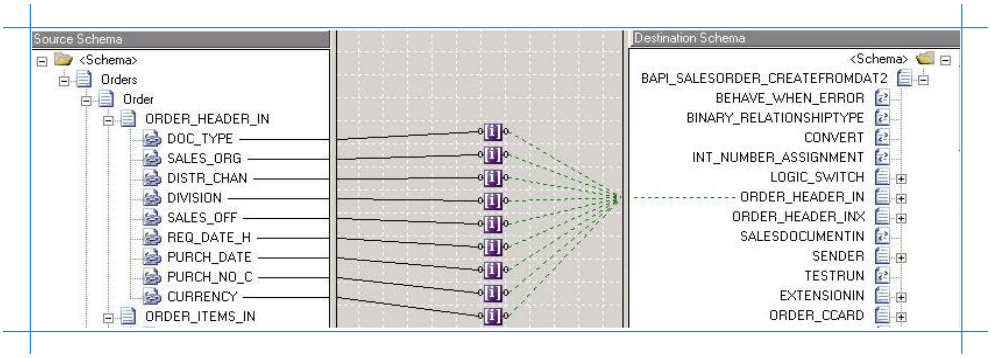
Sporočila, definirana v sistemu

Sporočilo	Opis sporočila
SendToAdapter	Sporočilo ki pride na vhod (vhodna datoteka)
BAPIMessage	Sporočilo ki se pošlje BAPI-ju
BAPIResponse	Odgovor ki ga BAPI vrne
BAPICommitMessage	Sporočilo BAPI-ju naj izvede COMMIT

BAPICommitResponse	Odgovor BAPI-ja glede na COMMIT
BAPIRollbackMessage	Sporočilo BAPI-ju naj izvede ROLLBACK
BAPIRollbackResponse	Odgovor BAPI-ja glede na ROLLBACK

4.3.5 Mapiranje podatkov

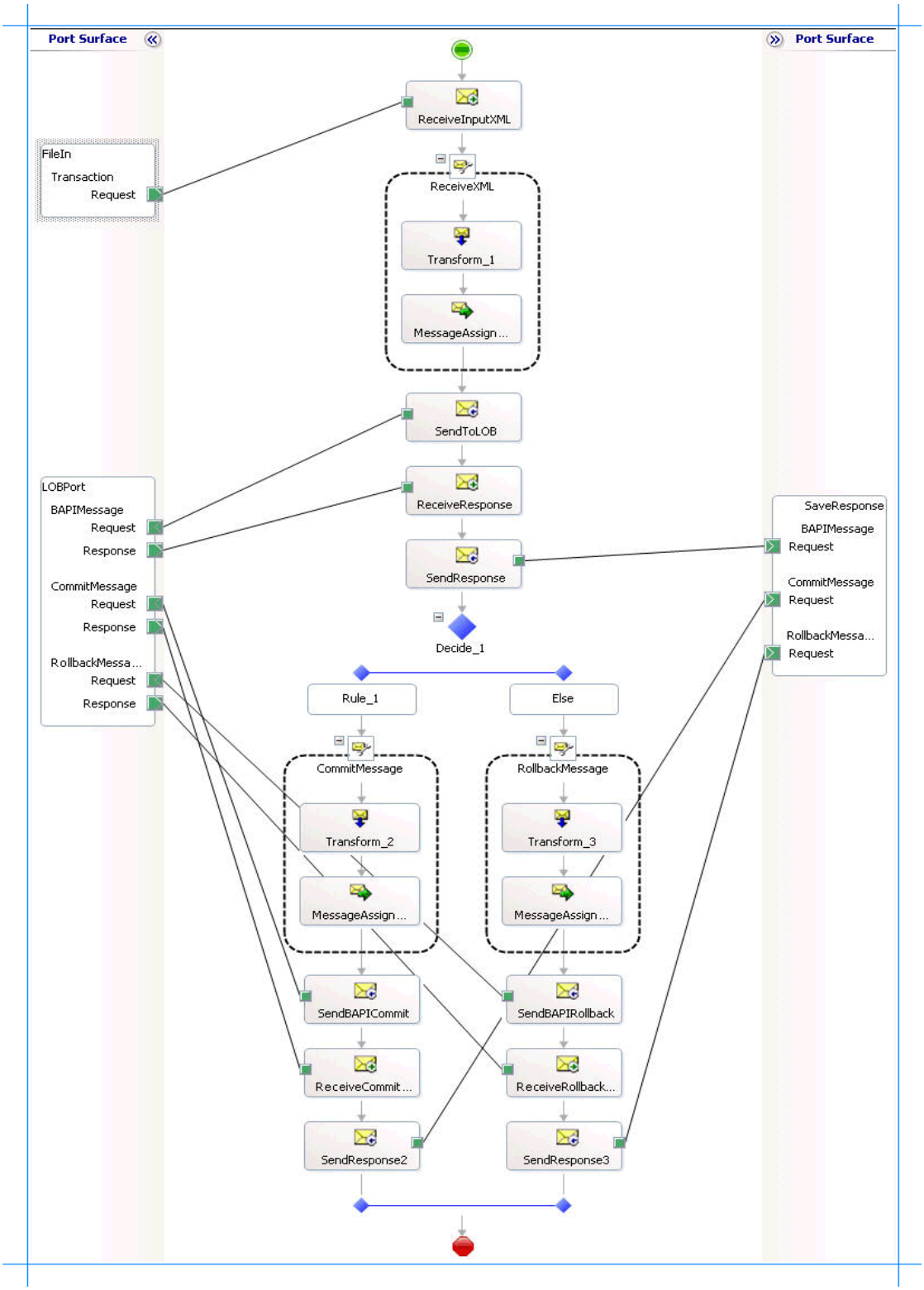
Ker so podatki v vhodni datoteki v drugačni obliki kot jih pošiljamo BAPI-ju, moramo izvesti mapiranje podatkov. Za mapiranje obstaja svoj tip datoteke v Visual Studiu (Biz Talk Map File).
Grafični vmesnik nam omogoča, da lahko vizualno povezujemo elemente med dvema shemama. Lahko si pomagamo tudi s funkcijami, da nam sistem sam mapira glede na podobna imena ali pa glede na strukturo v shemi.
(slika je simbolična)



Slika 22: Pregled mapiranja podatkov

4.3.6 Orkestracija sistema

Orkestracija skrbi in nadzira izvajanje in prenos sporočil med sistemi.
Opis naše orkestracije:
Na določeno lokacijo zunanji program odloži datoteko s podatki o računih. Sistem prevzame datoteko in jo preko mapirnih tabel pretransformira v obliko, ki je primerna za SAP. V taki obliki jo potem pošlje v SAP. Odgovor o uspešnosti transakcije shrani na drugo lokacijo. Sistem glede na odgovor o uspešnosti izvede commit oziroma rollback transakcije v SAP sistemu. Odgovor o uspešnosti commit-a/rollback-a prav tako shrani na drugo lokacijo.



Slika 23: Primer orkestracije

4.3.7 Definicija sporočil v sistemu

V orkestracijo je potrebno dodati sporočila, ki se bodo pošiljala v sistemu. Spodaj je prikazana tabela sporočil. Ker je za uspešen prenos potrebno poslati dve sporočili (podatki + commit/rollback), moramo definirati še ostala sporočila za vse tipe pošiljanja sporočil)

Sporočilo	Tip sporočila
ReceiveInputXML	Receive
SendToLOB	Send
ReceiveResponse	Receive
SendResponse	Send
SendBAPICommit	Send
ReceiveCommitResponse	Receive
SendResponse2	Send
SendBAPIRollback	Send
ReceiveRollbackResponse	Receive
SendResponse3	Send

4.3.8 Definicija odločitvenega pravila

V orkestracijo smo vključili tudi odločitveno pravilo. Pravilo glede na uspešnost izvedbe prve akcije, izvede eno ali drugo akcijo. Torej če so bili podatki na računu pravilni, se izvede commit, v nasprotnem primeru se pa izvede rollback. Za pogoj definiramo sledeče:
SendToAdapter.isCommit == true
Da lahko tak pogoj definiramo, moramo v shemi za SendAdapter določiti, da se ta lastnost pošlje na adapter.

4.3.9 Dodajanje vrat (port)

V našem primeru bomo rabili tri različna vrata:

Vrata	Opis
FileIn	• Prebere datoteko iz lokacije na disku. Komunicira le v eni smeri in to tako, da sprejema podatke.
LOBPort	• Ta vrata komunicirajo s SAP strežnikom. Komunicira v obe smeri, tako da pošilja in sprejema podatke. Izvaja vpis podatkov in commit/rollback.
SaveResponse	• Shranjuje povratno informacijo od vpisa podatkov do commit/rollback akcije. Komunicira le v eni smeri in to tako, da oddaja podatke.

4.3.10 Definicija sporočil za akcije in povezava le teh

Potrebno je definirati sporočila in povezave sporočil na vrata.

ReceiveInputXML
SendToLOB
ReceiveResponse
SendResponse
SendBAPICommit
ReceiveCommitResponse
SendResponse2
SendBAPIRollback
ReceiveRollbackResponse
SendResponse3

Ko smo uspešno definirali in povezali sporočila, je urejanje orkestracije končano.

4.3.11 Prenos aplikacije na BizTalk strežnik

Najprej moramo na BizTalk strežnik kopirati referenco na shemo za sap adapter. Shema se nahaja v knjižnici *Microsoft.Adapters.SAP.BiztalkPropertySchema.dll*.
Ko smo dodali knjižnico za adapter, moramo dodati še *Assembly* kot resurs v BizTalku. To storimo v administracijski konzoli strežnika in sicer odpremo želeno aplikacijo (najbrž kreiramo novo) in pod *Resources* lahko dodamo *BizTalk Assembly*.
Ko imamo to definirano, moramo še prevesti aplikacijo v Visual Studiu. Kot rezultat vidimo novo orkestracijo v administracijski konzoli.

4.3.12 Nastavitve aplikacije na BizTalk strežniku

Zopet odpremo administracijsko konzolo in poiščemo našo orkestracijo. Določiti moramo gostiteljski strežnik (host), fizične adapterje, sprejemna/oddajna vrata ...
Nastaviti je potrebno SAP adapter, mu podati povezavo na pravi SAP strežnik ...

4.3.13 Zagon aplikacije

Zagon se izvede v administracijski konzoli in sicer med orkestracijami najdemo našo orkestracijo in preko menija izberemo *Start*.
Potem na vnaprej določeno lokacijo odložimo XML datoteko z vhodnimi podatki. Če smo celoten proces definirali pravilno, bi se morala na lokaciji, kjer odlagamo povratno informacijo, pojaviti datoteka z vsebino o obdelavi računa.

4.4 Primerjava vseh treh načinov obdelav

Kriteriji za primerjavo vseh treh načinov integracij:

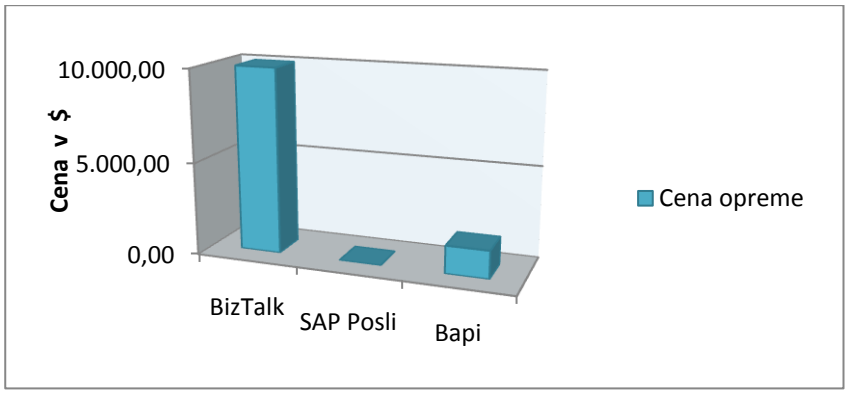
4.4.1 Cena potrebne programske opreme

Če primerjamo načine med seboj glede na ceno programske opreme, dobimo zelo zanimive rezultate. Zraven vsakega načina sem navedel tudi potrebno programsko opremo za razvoj, ki je ne moremo šteti h končni ceni integracije. To je pač investicija, ki jo mora vsako razvojno podjetje izvesti.

SAP Posli
Za razvoj potrebujemo le eno licenco za SAP Odjemalca. Te licence kasneje za uporabo v produkciji ne potrebujemo. Zato lahko rečemo, da je integracija na ta način dejansko brezplačna. SAP odjemalca lahko uporabimo na katerikoli postaji kjer ljudje uporabljajo SAP. Zato ni potrebno kupovati licenc samo za ta namen.

BAPI:
Za razvoj potrebujemo program Visual Studio 2006 in novejši in SAP odjemalca. Za produkcijsko delovanje bi po zagotovilih SAP-a bilo dovolj, če bi v programski paket vključili le določene knjižnice od SAP klienta, ki jih je mogoče dobiti brezplačno.
Torej lahko zaključimo, da je tudi ta način za uporabnike (podjetja) zelo ugoden. Če predpostavimo, da bo integracijski modul tekel na Windows 2003 strežniku, je to strošek 1500 $. Načeloma pa tudi to ni potrebno, saj se lahko integracijski modul namesti na tiste delovne postaje, kjer se bo izvajal prenos.
[8]

BizTalk:
Za razvoj potrebujemo Visual Studio 2006 in novejši, SAP odjemalca, BizTalk strežnik, Windows server 2003 (za potrebe BizTalk strežnika). Za kasnejše delovanje v produkcijskem okolju potrebujemo knjižnice SAP odjemalca, BizTalk strežnik in Windows server 2003 oz. novejši.
Okvirna cena strežnika je 1500 $, BizTalk pa okvirno 8,500 $, tako da skupni stroški nanesejo približno 10.000,00 $.
[7, 8]



Slika 24: Graf primerjave cen

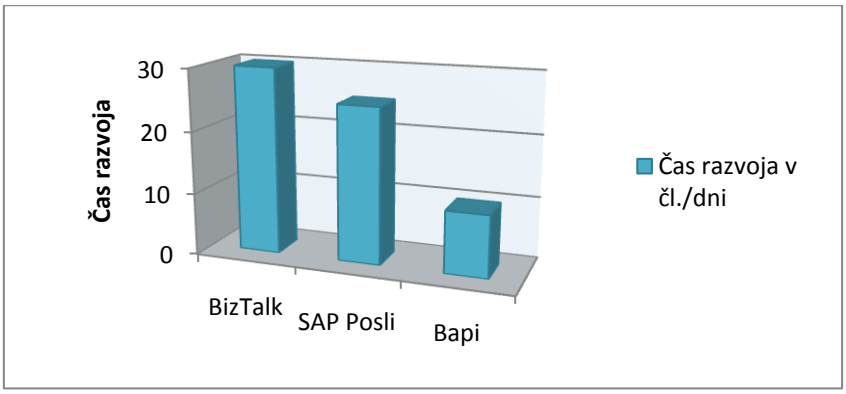
4.4.2 Čas razvoja integracije

Govoriti o času razvoja posamezne integracije je zelo relativno. Tukaj lahko pride do večjih odstopanj, saj lahko nekdo s predznanjem določenega področja precej hitreje naredi neko akcijo. Čas sem »meril« glede na moje raziskovanje, z upoštevanjem določenega predznanja. Časovna ocena je izražena v človek/dnevih. 1 človek/dan pomeni 8 ur dela.

SAP Posli
Glavni razvoj poteka v SAP odjemalcu. Sama logika je precej zapletena in temu primeren je tudi čas razvoja. Potrebno je bilo precej študije SAP-ove dokumentacije glede poslov. S predpostavko da bi moral cel sistem razviti iz ničle, bi lahko na grobo ocenil delo na 25 človek/dni.

BAPI
V tem primeru poteka razvoj v orodju Visual Studio. Sam razvoj je precej hiter. Je pa potrebno veliko vsebinskega znanja na področju, kjer izvajamo integracijo. Potrebno je poznati proces in podatke, katere prenašamo iz sistema v sistem. Tukaj lahko sistem hitro vrne napako, ki je lahko vsebinska, ali pa tudi sistemska. Ocena dela 10 človek/dni

BizTalk
Tudi v tem primeru je glavni del razvoja v Visual Studiu. Vendar je programiranje specifično, saj moramo poznati princip delovanja BizTalk strežnika. Tukaj bi zelo težko podal oceno dela, saj mi ni uspelo v praksi izdelati tega primera integracije. Vendar bi glede na navodila in potek razvoja lahko ocenili, da bi izdelava takega sistema trajala 30 človek/dni.



Slika 25: Graf primerjave časa razvoja

4.4.3 Kompleksnost vzdrževanja sistema

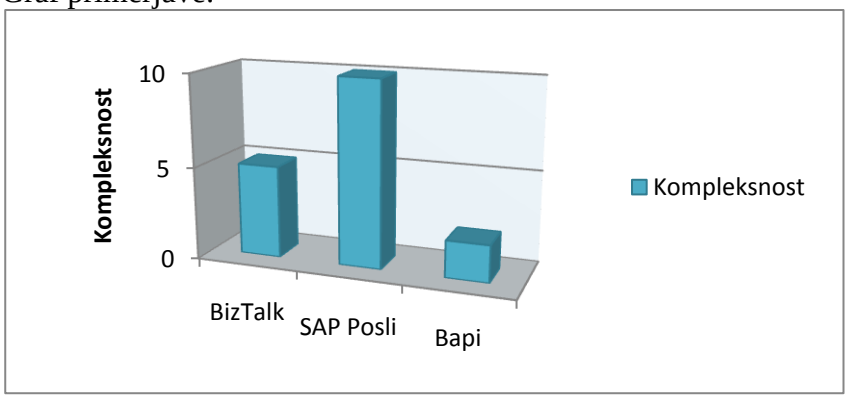
Vsak sistem potrebuje vzdrževanje. Pod vzdrževanje smatramo razne dopolnitve prenosa podatkov. Lahko podjetje uvede dodaten indikator/podatek in je potrebno ta podatek prav tako prenesti v SAP. Tukaj je pomemben tudi način lociranja in odprave težave na sistemu. Dobro vemo, da noben sistem ni popoln in 100% zanesljiv, zato je zaželeno, da težavo čim lažje najdemo in tudi odpravimo. Kompleksnost bomo ocenjevali kar z lestvico od 1-10

SAP Posli
Ta način je zelo kompleksen za vzdrževanje. Problematično je kreiranje poslov, ki se lahko že ob napačni kombinaciji proženja zaklenejo. Tudi sama lokacija napake je zelo težavna, saj je veliko različnih programov in točk, kjer lahko pride do napake. Ocena težavnosti: 10

BAPI
Sistem je lažje obvladljiv. Prav tako pri samem vzdrževanju ni veliko komponent, zato je napaka hitro locirana. Omogočeno je tudi »debugiranje«, tako da je sistem precej obvladljiv. Za dodajanje novega parametra prenosa je potrebno ponovno prevesti program. Tukaj velja omeniti še to, da mora za dodajanje novega parametra BAPI omogočati vnos tega parametra. Kar pa ni vedno podprto. Ocena težavnosti: 2

BizTalk
Tukaj zopet težko podamo realno oceno, saj sistema v praksi ne poznamo. Sklepamo pa, da ni tako zelo kompleksen, saj nam že sam strežnik omogoča izpis zgodovine dogajanja na strežniku. Prav tako je z dodajanjem novih parametrov. Potrebno je dopolniti sheme in ponovno prevesti program, ter ga namestiti na strežnik. Ocena težavnosti: 5

Graf primerjave:



Slika 26: Graf primerjave kompleksnosti vzdrževanja

5 Sklepne ugotovitve

Eden izmed najpomembnejših faktorjev pri poslovanju podjetja je dobiček. Dobiček podjetje ustvarja z razliko med prodajo in nabavo, z opravljanjem storitev ali na kakšen drug način. Če je podjetje uspešno, potem ustvarja dobiček. Ker je pa vodenje podjetja kar precej velik zalogaj za vodilne, je tukaj informacijska podpora. Današnji sistemi omogočajo veliko podporo pri poslovanju podjetja. Vendar moramo omeniti, da morajo biti različni sistemi med sabo povezani. Če te povezave ni, potem lahko hitro nastanejo nepotrebni stroški. Prav tako lahko pride do napačne slike stanja financ, zalog ... v podjetju, posledično uprava nima pregleda stanja. Zato so integracije med sistemi v podjetju zelo pomembne.

Če pogledamo zgornje primerjave, hitro ugotovimo, kateri način je »najboljši«. Sicer je beseda najboljši zelo relativna, saj so določeni faktorji, preko katerih ne moremo. Podjetje ima lahko svoje predpise in standarde pri implementaciji in tukaj potem lahko nastanejo razlike pri oceni posameznih načinov. Npr. če podjetje želi, da se vsi dokumenti pošiljajo podpisani, da določena oseba dobi obvestilo o izvedeni akciji itd., potem ima tukaj absolutno prednost BizTalk.
Če pa podjetje nima posebnih potreb in je glavni cilj izvesti integracijo, je najprimernejši način preko BAPI-jev.

BAPI nam omogoča nizko ceno razvoja. Ko pa enkrat integracijo prenesemo v produkcijsko delovanje, ta način ne povzroča nobenih fiksnih stroškov. Torej ni dodatnih licenc za odjemalca ... Tako lahko rečemo, da nas nič ne stane. Enostaven je za obvladovanje, dopolnitve ... Prav tako je razvoj te integracije zelo hiter.
Moramo pa tukaj omeniti, da je bila analiza usmerjena na prenos računa v SAP. Glavni cilj je bil prenesti fakturo iz zunanjega sistema v SAP. Če bi se zahteve po prenosih povečale, se pravi, da bi prenašali podatke še v kakšen drug sistem, bi se prav gotovo BizTalk boljše odrezal. Tudi cenovno bi bil BizTalk v tem primeru boljši. Dopuščam možnost, da je moja ocena dela malce pristranska za BizTalk, saj ga v praksi nisem uspel zagnati.

V današnjem času dobiva integracija sistemov vse večji pomen. V preteklosti integracije ni bilo v taki meri kot je sedaj. Razlog za to je zelo enostaven – podjetja niso imela toliko različnih sistemov, oz. procesi niso bili informatizirani. Prav tako je danes veliko spletnih storitev (WEB services), ki so dostopne na spletu in preko njih je možno dobiti razno razne podatke. Npr. status davčnega zavezanca (evropski servis), razni finančni podatki ...

6 Viri

[1] Aleš Šuštaršič, *Integracija standardnega poslovno-informacijskega sistema SAP R/3 s sistemom za podporo skupinskemu delu*, Ljubljana, 2002

Spletne strani:

[2] Splošno o ERP sistemih
Dostopno na http://www.tech-faq.com/erp.shtml, 8.12.2009

[3] Wikipedija
Dostopno na http://en.wikipedia.org, 7.1.2010

[4] Primeri vzpostavljanja orkestracij
BizTalk Server 2009 Developer Immersion, dostopno na www.QuickLearn.com, 8.1.2010

[5] SAP dokumentacija in primeri
Dostopno na http://searchsap.techtarget.com, 10.11.2009

[6] Uporaba BizTalk adapterja za SAP
Dostopno na http://msdn.microsoft.com/en-us/library/cc185462%28BTS.10%29.aspx, 2.2.2010

[7] Splošno o BizTalk strežniku
Dostopno na http://www.microsoft.com/biztalk/en/us/default.aspx, 1.2.2010

[8] Cene programske opreme
Dostopno na http://www.microsoft.com/slovenija/windowsserver2003/SBS/Pricing.mspx, 1.2.2010

7 Priloge

7.1 Kazalo slik