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: 00502/2010
Datum: 15.03.2010

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

Kandidat: **ANTON ŠIVIC**

Naslov: **RAZVOJ APLIKACIJ S POMOČJO SQL SERVER 2008 PLATFORME**
SQL SERVER 2008 BASED APPLICATION DEVELOPMENT

Vrsta naloge: Diplomsko delo visokošolskega strokovnega študija

Tematika naloge:

Podrobno proučite SQL server 2008 platformo in dajte poudarek različnim načinom proučitve poslovne logike in načinom proučitve poslovnih podatkov.

Mentor:

doc. dr. Rok Rupnik

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

Dekan:

prof. dr. Franc Solina

UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Anton Šivic

Razvoj s pomočjo SQL Server 2008 platforme

DIPLOMSKO DELO
NA VISOKOŠOLSKEM STROKOVNEM ŠTUDIJU

Mentor: doc. dr. Rok Rupnik

Ljubljanja, 2010

IZJAVA O AVTORSTVU

diplomskega dela

Spodaj podpisan **Anton Šivic**,

z vpisno številko **63020150**,

sem avtor diplomskega dela z naslovom:

Razvoj s pomočjo SQL Server 2008 platforme.

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom doc. dr. Roka Rupnika
- 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 06.07.2010 Podpis avtorja:

Zahvala

Zahvalil bi se predvsem svojemu mentorju doc. dr. Roku Rupniku za vodenje in nasvete pri izdelavi diplomskega dela.

Zahvalil bi se tudi vsem, ki vas je zares veliko, ki ste mi pomagali na moji poti polni številnih vzponov in padcev. Veliko mi pomeni, da ste verjeli vame, mi zaupali in omogočili, da sem danes to kar sem.

Hvala še enkrat!

Kazalo

Seznam uporabljenih kratic

IIS – kratica za Internet Information Services – Microsoft spletni strežnik

SQL – Structured Query Language – jezik za poizvedovanje po podatkovnih bazah

T-SQL – transakcijski SQL – Microsoftova implementacija SQL jezika za Microsoftove podatkovne baze

LINQ – Language Integrated Query

MSDNAA – Microsoft Developer Network Academic Alliance – Microsoftov program, ki akademskim organizacijam omogoča dostop do različnih programov

RFID – radio frekvenčni indikator

AD – aktivni direktorij – Microsoftova tehnologija, ki implementira številne mrežne storitve

BIDS – Business Intelligence Development Studio – Microsoft razvojno orodje za razvoj rešitev poslovne logike

SSRS – SQL Server Reporting Services – poročilni sistem v SQL Server 2008

SSIS – SQL Server Integration Services – integracijski sistem v SQL Server 2008

SSAS – SQL Server Analysis Services – sistem za analiziranje v SQL Server 2008

RDBMS – relational database management systems – sistem za upravljanje z relacijskimi bazami

OLTP – Online transaction processing – razredi ali sistemi, ki skrbijo za transakcijsko usmerjene aplikacije

DTS – Data Transformation Services – nabor objektov in pripomočkov, ki omogočajo avtomatizacijo ekstrahiranje, transformacije in nalaganje podatkov

ETL – postopek ekstrahiranja, transformacije in nalaganja podatkov znotraj podatkovnih skladišč

FTP – file transfer protocol – protokol za kopiranje podatkov iz ene lokacije na drugo

CSV – preprost format za shranjevanje podatkov iz podatkovnih tabel v tekstovne datoteke

PDF – format za shranjevanje dokumentov, ki ga je razvilo podjetje Adobe

XML – nabor pravil za kodiranje podatkov v formatu, ki je brez večjih težav prenosljiv med številnimi aplikacijami

SVN – sistem za verzioniranje

TFS – Microsoftov izdelek, ki omogoča verzioniranje, zbiranje podatkov, poročilni sistem

ADO.NET – del Microsoft.NET Framework platforme za dostopanje do podatkov

Povzetek

V diplomski nalogi sem poiskušal prikazati funkcionalnosti znotraj »SQL Server 2008« platforme in nekaj novosti, ki so se mi zdele najbolj zanimive s prihodom »Microsoft.Net Framework« platforme 3.0, 3.5 sp1 in 4.0. Funkcionalnosti, ki so se meni zdele najbolj zanimive so »SQL Server Integration Services«, »SQL Server Reporting Services« in »SQL Server Analysis services«. »SQL Server Integration services« so se mi zdele zelo zanimive predvsem zato, ker z njimi lahko sestavljamo poljuben tok podatkov in akcij s katerimi lahko zajamemo velik del poslovne logike brez dodatnega programiranja. Za »SQL Server Reporting Services« sem se odločil predvsem zaradi odličnega poročilnega sistema. Z uporabo »SQL Server Reporting Services« imamo lahko odličen poročilni sistem, za katerega ne potrebujemo imeti nameščenih IIS, hkrati pa lahko še vedno vpeljemo številne varnostne nastavitve. S temi varnostnimi nastavitvami pa lahko zagotovimo, da znotraj poslovnega okolja poročila gledajo le tisti, ki imajo temu primerne pravice. Tudi sama poročila, ki jih naredimo so enako dobra kot, če bi uporabili neke aplikacije, katere bi bilo potrebno razviti za te namene. S pomočjo »SQL Server Analysis Services« lahko podatke znotraj naše podatkovne baze analiziramo in nato optimiziramo, kar nam prav tako olajša delo, saj ne potrebujemo dodatnih orodij za analiziranje. Z »SQL Server Analysis Servces« lahko delamo različne vrtilne tabele.

Dodatno sem se odločil, da v moji diplomski predstavim še »Entity Framwork« tehnologijo in s tem povezan »LINQ«. Sama tehnologija se mi je zdela zelo zanimiva, saj lahko s tabelami in drugimi objekti iz podatkovne baze delamo, kot z običajnimi objekti.

Ključne besede:

Poročila, integracija podatkov, analiziranje podatkov, podatkovna baza, objektni model.

Abstract

In my thesis i tried to present some functionality from SQL Server 2008 platform and some new functionality and extensions that were introduced in Microsoft.NET framework version 3.0, 3.5 sp1 and 4.0. Things most interesting to me were SQL Server Integration Services, SQL Server Reporting Services and SQL Server Analysis Services. I think SQL Server Integration Services are interesting because you can create custom work flow diagrams and event handlers with which you can capture a lot of your business logic without any addtional programming. I decided to also use SQL Server Reporting Services because of it's great reporting system. With the use of SQL Server Reporting Services we get a great reporting system without a need to have IIS intalled and at the same time we can still implement a lot of security features. With this security features and settings we can assure that in our business environment only those who have suficient privileges can see the reports. The reports we do with SQL Server Reporting Services are as good as they were made inside applications that were specially designed for that purpose. With SQL Server Analysis Services we can analize our database data and the do different optimizations. This makes our work a bit easier because we do not need any aditional tools for analysing data. With SQL Server Analysis Services we can do different pivot tables.

Additionally I decided to include a chapter about Entity Framework and with that in mind also about LINQ technology. I find this technology very interesting because it makes it possible to work with database table and other database objct as regular objects.

Key words:

Reports, data integration, analysing data, database, object model.

1 Uvod

Ob začetku mojega študija na Fakulteti za Računalništvo in Informatiko nisem nikoli pričakoval, da bom za usmeritev izbral smer programska oprema. Odločitvi za izbiro omenjene smeri je v veliki meri verjetno botrovalo tudi to, da sem ravno takrat začel svojo pot dela preko študentskega servisa in s tem tudi spoznavanje, kako se znanje pridobljeno na predavanjih in vajah, lahko uporabi tudi v praksi. Prvo delo, ki sem ga opravljal je bilo testiranje programske opreme t.i. alfa testiranje. V tistem času sem se tudi prvič spoznal z razvojnim okoljem .NET (»Visual Studio 2005«) in pa »SQL Server« podatkovnim strežnikom (takrat »SQL Server 2005«), podjetje v katerem sem začel mojo delovno pot je namreč za svoje produkte uporabljalo omenjeno tehnologijo.

S tem se je začelo moje zanimanje o povezavi preprostih namiznih, spletnih, mobilnih aplikacijah s poljubnim podatkovnim skladiščem. Okolje, v katerem delam je narekovalo, da je to podatkovno skladišče ostalo »SQL Server« (danes »SQL Server 2008«), .NET pa platforma na kateri sem svoje rešitve lahko razvijal. Za delo z .NET platformo sem potreboval tudi orodje s pomočjo katerega sem te aplikacije lahko gradil. Podobno kot v Javi, bi tudi tu za to lahko uporabljal beležnico (Notepad), vendar pa bi bilo delo na tak način bistveno bolj dolgotrajno, kot če bi za to uporabil temu namenjeno orodje. Orodje, ki sem ga za delo z omenjeno platformo izbral je bil »Visual Studio«. »Visual Studio« namreč omogoča zelo enostavno gradnjo namiznih, spletnih in številnih drugih aplikacij. Omenjeno orodje omogoča tudi enostavno povezavo s podatkovnimi bazami, ki so shranjene na SQL Strežniku. S pomočjo .NET platforme za gradnjo aplikacij lahko uporabimo več programskih jezikov. Najbolj pogosta sta »Visual Basic .NET« in C#. Jaz sem se odločil za slednjega in se lotil raziskovanja njegove uporabe za komunikacijo s podatki shranjenimi nekje drugje.

Ovira, ki sem jo kmalu spoznal in je povezana z Microsoftom, je bila predvsem v plačljivih orodjih za delo z različnimi tehnologijami, ki jih Microsoft nudi svojim strankam. Na srečo je naša fakulteta del MSDNAA programa, preko katerega sem do vseh, drugače zelo dragih orodij, lahko prišel brezplačno. Pred uporabo orodij pridobljenih preko MSDNAA programa pa sem uporabljal brezplačne različice »Visual Studia« (Express) in »SQL Server 2005« (»Express« različica) za skladiščenje svojih podatkov.

2 Nekaj o Microsoft SQL Server 2008 platformi – pregled

Časi shranjevanja podatkov v pisni obliki so za nami. Nič več nimamo ogromnih arhivov nepreglednih kupov številnih dokumentov, na katerih se je vrsto let nabiral prah. V tem poglavju vam bom poskusil predstaviti vizijo kako so si v podjetju Microsoft zamislili, da naj bi bila zgrajena podatkovna platforma. Na podlagi te vizije so trgu ponudili novo podatkovno platformo imenovano »Microsoft SQL Server 2008«.

V zelo hitro razvijajočem se svetu računalništva se pojavljajo številni faktorji, ki povzročajo pravo revolucijo in nove tipe podatkov, kot so digitalizirane slike, video vsebine, podatki, ki jih lahko enostavno črpamo na podlagi branja črtnih kod, RFID oznak ali vse bolj priljubljenih t.i. Microsoft Tag oznak. Hitro lahko ugotovimo, da je teh informacij v podjetjih iz dneva v dan vse več. Zaposleni morajo do teh podatkov dostopati zelo hitro. Vendar pa te podatke zaposleni v veliki večini hranijo preprosto na svojih računalnikih, ki so vse bolj pogosto kar prenosni računalniki. Ob tem se pojavljajo številna varnostna vprašanja, saj se kaj hitro lahko zgodi, da zaposlenega, ki ima v danem trenutku kritično pomembne podatke ni v službi (doma ali na prostem pa nima dostopa do interneta), lahko se prenosnik tudi pokvari ali pa je zaposlenemu celo odtujen. Teh črnih scenarijev pa seveda nočemo.

Podjetja morajo tako poskrbeti, da so podatki zaposlenim na voljo kljub temu, da morebiti skrbnika ali avtorja teh podatkov ni v službi. Glede na to, da so stroški shranjevanja podatkov vse manjši, podjetja lahko zagotovijo rešitve, kjer so ti podatki shranjeni na nekih centralnih lokacijah. Te rešitve pa morajo zadostiti številnim standardom, torej morajo biti ti podatki shranjeni predvsem varno, hkrati pa tudi vedno na voljo za uporabo.

Slika 1 - shematičen prikaz Microsoftove vizije za podatkovne platformo

Microsoftov si je svojo platformo zamisli upoštevajoč vse zgoraj omenjene faktorje, hkrati pa podjetjem in številnim drugim organizacijam omogočil shranjevanje in uporabo podatkov v različnih podatkovnih formatih. Ti formati so lahko XML datoteke, elektronska pošta, datumske vrednosti, dokumente, mape in celo geografski podatki. Na voljo so tudi številne storitve, ki omogočajo enostavno poizvedovanje, analiziranje, integracijo z drugimi sistemi, izdelavo poročil, hkrati pa morajo zagotavljati tudi robustno sinhronizacijo.

Sami SQL strežniki so bili v preteklosti dokaj preprosti za razumevanje, kar se tiče njihove logike delovanja. Njihova osnovna funkcionalnost je bila sestavljena iz podatkovne baze za t.i. OLTP procesiranje (»Online Transaction Processing«), skupaj z možnostmi repliciranja podatkov na različne lokacije. Nekaj verzij produkta kasneje lahko rečemo, da se je SQL Server razvil v obsežno podatkovno platformo, ki je sposobna shranjevanja velikih količin podatkov, upravljanja in predstavitve teh podatkov v malih, srednjih, kot tudi velikih podjetjih, ki imajo veliko dislociranih enot.

Zadnja platforma bi lahko rekli, da ima sledeče glavne prednosti:

- **Zaupanja vredna (»Trusted«)** – podjetja in organizacije lahko uporabljajo svoje aplikacije na zelo visokem varnostnem in zanesljivem nivoju, ki jim omogoča enostavno razširljivost.
- **Produktivnost** – podjetja in organizacije lahko bistveno zmanjšajo stroške povezane z razvojem, implementacijo in vzdrževanjem podatkovne infrastrukture.
- **Inteligenca** – zelo razumljiv in intuitiven pristop k razumevanju same platforme, ki zelo enostavno omogoča vpogled v samo strukturo podatkov in informacij, te pa so na voljo kjer koli jih uporabnik potrebuje.

2. 1. Podatkovna baza

Je osnovna funkcionalnost SQL Serverja 2008, ki nam omogoča enostavno shranjevanje, pridobivanje, obdelavo podatkov, hkrati pa so ti podatki tudi zelo varni. S pomočjo te baze lahko izdelujemo zelo učinkovite aplikacije za OLTP procesiranje (»Online Transaction Processing«), dodana pa je seveda tudi podpora za OLAP analizo podatkov (»Online Analytic Processing«).

2. 2. Mehanizem shranjevanja podatkov

V samem jedru SQL Serverja se nahaja mehanizem za shranjevanje podatkov (»Storage Engine«), ki skrbi za to, kako se podatki fizično zapisujejo na disk, ter kako so podatki na voljo aplikacijam, ki jih uporabljajo za svoje delo.

Sama hramba podatkov oziroma mehanizem shranjevanja teh podatkov je nam kot uporabnikom neposredno nedostopen, vendar pa vsebuje ključne komponente za shranjevanje in upravljanje z našimi podatki.

Podatki se shranjujejo na podlagi naših tabel, ki so osnovni objekti za hranjenje podatkov. Poleg samih tabel pa so seveda pomembni tudi stolpci oziroma tipi stolpcev, saj je od tega, kakšnega podatkovnega tipa je stolpec, odvisno tudi kakšne podatke bo stolpec vseboval. Za zmogljivejše in učinkovitejše poizvedovanje lahko ustvarimo in vzdržujemo tudi indekse, ki

zgradijo neke vrste interni katalog podatkov nad katerimi smo jih postavili (nekaj podobnega kot indeksi v knjigi, kjer so različni izrazi razvrščeni leksikografsko, čeprav se lahko nahajajo na različnih straneh). Večje podatkovne tabele in njim pripadajoče indekse lahko nato razdelimo tudi na več različnih particij tako, da imamo naše podatke lahko shranjene na različnih lokacijah, ki navzven delujejo kot celota.

S pomočjo mehanizma za shranjevanje podatkov lahko izdelamo tudi posnetke podatkovne baze (t.i. »Database snapshot«). S temi posnetki ustvarimo kopije podatkovne baze, ki so namenjene zgolj za branje in odražajo stanje podatkovne baze v nekem določenem trenutku. Seveda pa lahko naredimo varnostne kopije celotnih podatkovnih baz in ne zgolj samo njihovih posameznih stanj v določenem trenutku. Tako se lahko zavarujemo pred morebitnimi izgubami podatkov ali pa enostavno obnovimo podatke, ki so bili poškodovani.

2. 3. Varnostni podsistem

»SQL Server 2008« vsebuje izredno učinkovito in prilagodljivo varnostno infrastrukturo s pomočjo katere lahko instance »SQL Server« in podatke shranjene v njih zavarujemo pred morebitnimi vdori in zlorabami.

»SQL Server 2008« lahko nadzira način, kako se klienti vanj prijavljajo. Imamo lahko overovitev na nivoju Windows operacijkega sistema (»Windows credentials«), kjer lahko znotraj poslovnega okolja kar na nivoju aktivnega imenika (»Active directory – AD«) določamo dostopna določila za posamezne uporabnike. Overovitev uporabnikov lahko poteka tudi na nivoju SQL Serverja, kjer moramo znotraj samega SQL Serverja ustvariti novega uporabnika. Temu uporabniku moramo nato nastaviti pravice s pomočjo katerih lahko dostopa do posameznih podatkovnih baz. Varnostne nastavitve lahko spreminjamo na več nivojih (npr. za vsako podatkovno bazo, ki se nahaja v instanci SQL Serverja, posebaj) in s tem uporabnikom omogočamo branje/pisanje podatkov, kot tudi upravljanje z objekti znotraj posameznih instanc.

Kot kombinacija obeh pa lahko uporabljamo tudi kombiniran način dostopanja do podatkovnih baz na SQL Strežniku (t.i. »Mixed mode«), kjer za prijavo lahko uporabimo tako Windows, kot SQL overovitev uporabnika.

»SQL Server« ima tudi sistem sledenja, ki je na voljo za spremljanje uporabe višjih pravic (t.i. »elevated permissions«) s katerim lahko posnemamo uporabnika, ki ima pravice nad objekti (npr. tabelami) do katerih naše uporabniško ime in geslo ne moreta dostopati. Seveda nam mora uporabnik, ki ima dovolj pravic (npr. avtor tabele, administrator, …) omogočiti uporabo višjih pravic, drugače te funkcionalnosti nad nekimi tabelami ne moremo uporabljati.

2. 4. Programski vmesnik

V SQL Serverju 2008 se kot primarni jezik uporablja Transakcijski-SQL (t.i. T-SQL - »Transact Structured Query Language«). SQL je jezik za poizvedovanje podatkov v relacijskih sistemih (t.i. »RDBMS – relational database management systems«). T-SQL pa je Microsoft in Sybase-ova razširitev SQL-a. T-SQL je zelo močan programski jezik za poizvedovanje.

V primeru, da nam T-SQL ne zadošča oziroma ne omogoča funkcionalnosti, ki bi jo želeli implementirati, pa lahko funkcionalnosti SQL Serverja razširimo z uporabo funkcionalnosti, ki se nahajajo znotraj CLR (»Common Language Runtime«) programskih jezikov, kot sta »Microsoft Visual Basic« in »Microsoft Visual C#«. CLR je del .NET Framework platforme, ki nam omogoča izvajanje aplikacij napisanih v omenjenih dveh programskih jezikih.

Sam »SQL Server 2008« vsebuje tudi podporo za XML, ki je neposredno vključena v njegovo osnovno ogrodje. Ta funkcionalnost nam omogoča enostavno poizvedovanje, shranjevanje in tudi izpisovanje podatkov v različnih oblikah XML formatov.

Znotraj samega vmesnika lahko med seboj združujemo številne druge objekte SQL Serverja, kot so pogledi (»views«), shranjene procedure (»stored procedures«), funkcije in prožilce (»triggers«). Koda lahko postane zelo modularna in enostavna za uporabo pri razvoju naših aplikacij.

Podpora za tekstovno iskanje (»Full – text search«) pa nam omogoča gradnjo poizvedb iz potencialno velikih količin zelo nestrukturiranih podatkov.

2. 5. Service Broker

»Service Broker« je mehanizem, ki je bil predstavljen že v »SQL Serverju 2005« in zagotavlja mehanizem za hranjenje sporočil v vrsti (»queuing«). S pomočjo sporočil definiranih s strani uporabnika in raznih dogodkov pri procesiranju podatkov lahko s pomočjo »Service Brokerja« zagotovimo asinhrono procesiranje podatkov.

2. 6. SQL Server Agent

Odlično orodje za spremljanje stanja našega SQL Serverja. Gre za orodje, ki skrbi za različna opravila, ki jih mora SQL Server izvršiti po nekem urniku (»backup« podatkovne baze, izvajanje SSIS paketkov), da se izvedejo. Poleg tega, da skrbi za opravila, pa nam omogoča, da ta opravila lahko ustvarimo tudi sami. Opravila so lahko sestavljena iz več različnih korakov, kjer lahko med seboj kombiniramo izvajanje SSIS paketkov, izvajanje SQL skript in številne druge operacije.

Poleg spremljanja, ustvarjanja in izvajanja številnih opravil, ki se izvajajo na SQL Serverju pa lahko uporabljamo tudi številne mehanizme obveščanja s pomočjo katerih nam »SQL Server Agent« lahko avtomatsko sporoči, kaj se na našem strežniku dogaja. Sporočila lahko vsebujejo podatke o morebitnih napakah ali pa preprosto, da se je določeno opravilo uspešno zaključilo.

2. 7. Replikacija

Je orodje, ki nam omogoča izdelavo in spreminjanje podatkov v različnih podatkovnih bazah, ki se lahko nahajajo na različnih lokacijah. Komunikacija med temi podatkovnimi bazami lahko poteka po konceptu vsak z vsakim (»peer to peer«).

Sposobnost združevalnih replikacij (»Merge replication«) pa omogoča nepovezano obdelavo podatkov. Ti uporabniki so večino svojega časa na poti in nimajo na voljo neposrednega dostopa do podatkov. Uporabniki tako med seboj prenašajo lokalne kopije podatkov, ki jih lahko spreminjajo, ko se vrnejo v podjetje pa te podatke lahko sinhronizirajo z matično podatkovno bazo.

2. 8. Visoka razpoložljivost

Zahtevajo jo najbolj zahtevne aplikacije, ki za svoje delo potrebujejo podatke 24 ur na dan. »SQL Server«, ki mora primarno kot platforma za shranjevanje podatkov vse te zahteve izpolnjevati, implementira več različnih mehanizmov s pomočjo katerih te zahteve lahko izpolnjuje.

2. 9. SQL Server nadomestne instance (»SQL Server failover clustered instances«)

So zgrajene na podlagi WCS (»Windows Clustering Services«). S tem lahko v primeru strojnih napak zavarujemo in ohranimo svoje podatke.

2. 10. Zrcaljenje podatkovne baze

Imamo kopijo produkcijske podatkovne baze. »SQL Server« pri tem uporablja različne mehanizme spremljanja procesov, ki se znotraj instance katero kopiramo izvajajo (zapisovanje podatkov t.i. »logiranje«). Na podlagi teh zapisov potem lahko zrcaljenje (»mirroring«) izvajamo sinhrono brez bojazni, da bi se kakšna transakcija izgubila in s tem ne izvedla.

2. 11. Premikanje zapisov (»Log shipping«)

S pomočjo orodja »SQL Server Agent« lahko nastavimo opravila, ki nam zapise (»Log«) v katere se zapisujejo informacije povezane z našim SQL strežnikom, kopirajo na pomožne sekundarne strežnike. Ti sekundarni strežniki so naš varnostni mehanizem.

3 Integration Services (SSIS)

»Integration Services« je SQL Server platforma, ki jo uporabljajo aplikacije, ki delajo z integracijo podatkov in aplikacijami, ki za svoje delo uporabljajo načrte nekega delovnega toka (»workflow«). SSIS so naslednik »Data Transformation Services« (DTS), katere so se znotraj različnih organizacij uporabljale predvsem za premikanje podatkov.

SSIS vsebuje vse funkcionalnosti, ki jih najdemo v ETL aplikacijah (»Extract, Tranform and Load«). Izluščijo lahko podatke iz številnih zunanjih virov (različne relacijske ali ne relacijske podatkovne baze, datoteke različnih formatov (npr. txt, csv, xml)) te podatke nato transformirajo. S pomočjo transformacije poskrbimo, da se podatki shranjeni v izvornem

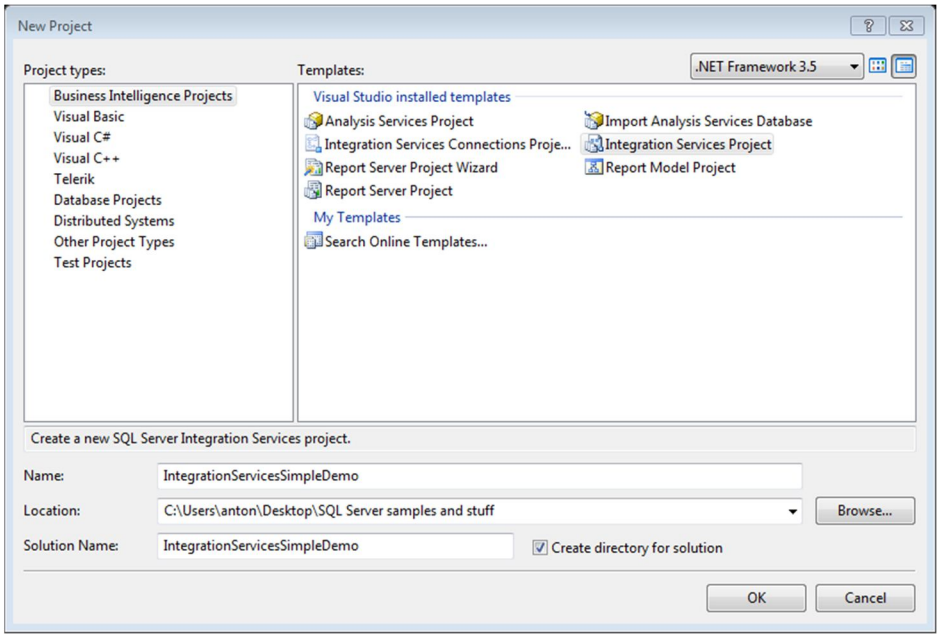
formatu pretvorijo v format primeren za obdelavo. Po zaključeni obdelavi te podatke zapišemo v poljubno ciljno podatkovno bazo ali podatkovno skladišče.

SSIS lahko znotraj poslovnega okolja uporabljamo tudi za upravljanje procesov definiranih v delovnem toku, opravil v podatkovni bazi, ravnanje s sistemskimi viri. Prav tako se lahko odzovejo na številne dogodke (npr. uspešno ali neuspešna transakcija) in temu ustrezno komunicirajo z uporabnikom (kot sprememba v sami shemi delovnega toka).

Delo s SSIS storitvami poteka preko tako imenovanih SSIS paketkov znotraj katerih lahko implementiramo številne delovne toke. Te pakete lahko shranimo neposredno na disk, kot fizične datoteke, lahko pa jih shranimo tudi v samo podatkovno bazo. Moram reči, da v primeru, kjer imamo testno okolje znotraj katerega najprej te paketke testiramo, nato pa jih moramo prenesti v produkcijo najbolje te paketke že v osnovi shraniti kot samostojne datoteke. Drugače bi morali paketke iz podatkovne baze najprej izvoziti in nato ponovno uvoziti na produkcijskem sistemu, tako pa jih zgolj samo uvozimo v podatkovno bazo.

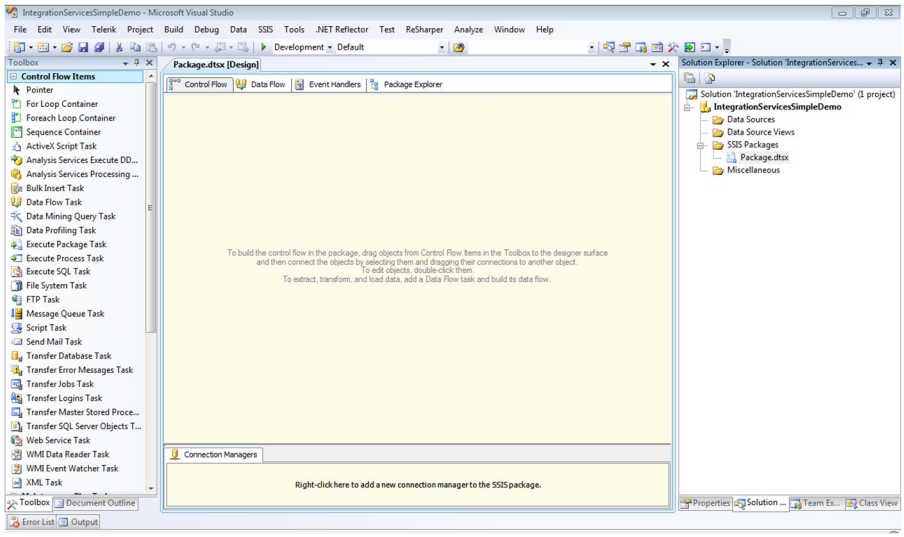
Pri izdelavi SSIS paketov imamo na voljo kar nekaj gradnikov, ki nam omogočajo izvajati številne naloge. Te naloge lahko predstavljajo prenašanje datotek preko FTP protokola, upravljanje z datotekami v imenikih, uvažanje datotek v podatkovno bazo, izvoz podatkov iz podatkovne baze v nepovezane datoteke (»flat file«).

3. 1. Primer izdelave preprostega SSIS projekta

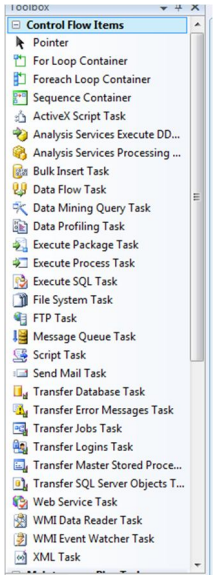


Slika 2 - začnemo nov SSIS projekt

Nov SSIS projekt začnemo tako, da ko zaženemo BIDS, izberemo File → New → Project in znotraj »Business Intelligence Projects« izberemo predlogo »Integration Services Project«.

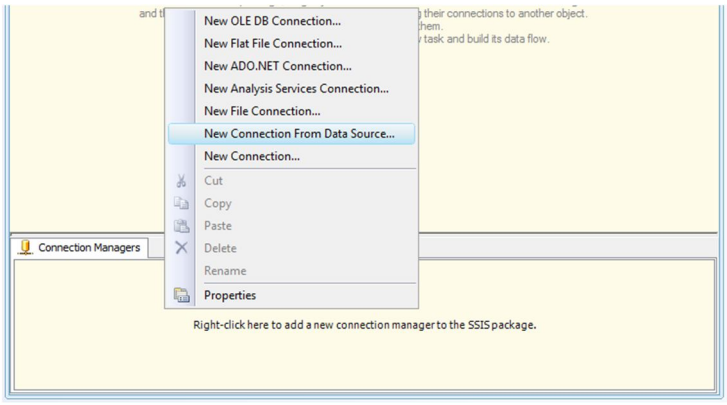


Slika 3 - znotraj "Package Designer" okna implemetiramo tok naših podatkov



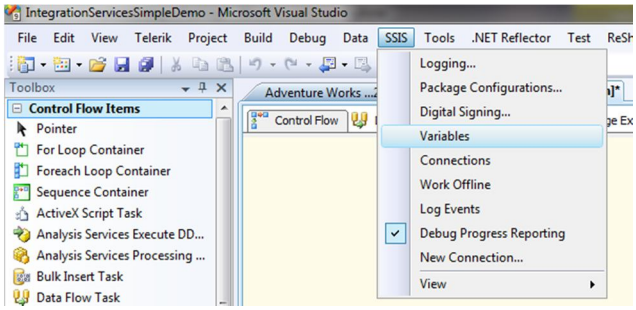
Slika 4 - Toolbox okno

»Toolbox« okno vsebuje številne gradnike za gradnjo našega kontrolnega toka izvajanja.



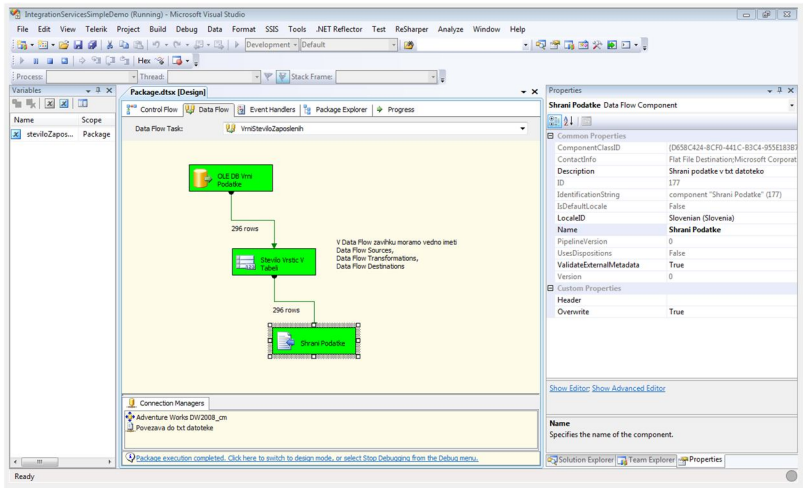
Slika 5 - Connection Managers

Znotraj »Connection Managers« okna naredimo povezavo na naš vir podatkov, v mojem primeru bom izbral »New Connection From Data Source…«.



Slika 6 - menijska vrstica SSIS meni

Ker smo izbrali SSIS paket, imamo sedaj v menijski vrstici SSIS meni, preko katerega lahko SSIS paketu določimo npr. spremenljivke, itd.



Slika 7 - shema preprostega diagrama

4 Reporting Services (SSRS) – poročilni sistem znotraj SQL Server 2008 platforme

S stališča razvoja bi najraje videli, da bi vsi znali pisati poizvedbe, ki nam vrnejo podatke, ki jih v danem trenutku potrebujemo. V primeru, da to ni mogoče, bi želeli imeti na voljo programerje, ki bi bili pripravljeni narediti vmesnike s pomočjo katerih bi lahko v podjetju zagotovili podatke, ki jih podjetje potrebuje.

V večini podjetij je to zelo težko zagotoviti, saj podjetja v večini primerov nimajo na voljo virov, ki bi jih lahko temu namenili. Končni uporabniki tako potrebujejo orodje, ki jim je na voljo in s katerim lahko izdelajo in oblikujejo standardizirana poročila. Te uporabniki imajo tako tudi možnost ustvariti poročila v trenutku takrat, ko jih potrebujejo in jih ni potrebno pripraviti vnaprej. Ta poročila so na voljo znotraj celotne organizacije.

Tako imamo na voljo močno platformo za delo s poročili, ki omogoča enostavno načrtovanje in seveda prikazovanje podatkov znotraj celotne organizacije.

V primeru, da imamo v podjetju na voljo IT oddelek, ki skrbi za naše podatkovno skladišče, lahko IT oddelek poskrbi za izdelavo poročil. Ta poročila, kot smo že omenili, pa lahko izdelajo uporabniki sami. Ob tem jim ni potrebno poznati celotne strukture in logike, ki smo jo implementirali v ozadju naše podatkovne baze.

Poleg samega poročilnega sistema lahko znotraj »SQL Server 2008 Reporting Services« definiramo tudi številne dodatne funkcionalnosti. Med drugim si lahko uporabniki pripravijo opravila, ki se na SSRS izvajajo po urniku.

Uporabniki si namreč lahko naredijo urnik izdelave poročil, ki so lahko poljubnih oblik. Ta poročila so lahko poslana prejemnikom preko različnih distribucijskih kanalov.

Primer uporabe je lahko scenarij, kjer želi vodja prodaje spremljati prodajo izdelkov na dnevni bazi. Preprosto se nastavi opravilo na SSRS, ki se izvede ob koncu delovnega dne, ko je podjetje že zaprlo svoja prodajna mesta. Ob tem se izdela poročilo v PDF formatu, katero se pošlje vodji prodaje, ki ga zjutraj lahko vidi kot novo elektronsko sporočilo s prilogo v svojem poštnem predalu.

SSRS je v resnici sestavljen iz dveh komponent. Ti dve komponenti sta strežnik za poročila in »Report Designer«.

Strežnik za poročila je odgovoren za gostovanje poročil in vse varnostne nastavitve. Ko uporabnik zahteva poročilo, mora strežnik za poročila poskrbeti za vzpostavitev povezave med poročilom in podatki, generiranje in prikaz.

Poročilo lahko izdelamo na dva osnovna načina:

- na zahtevo uporabnika se poročilo izdela takrat, ko ga potrebuje,
- preko urnika opravil, ko vnaprej nastavimo časovno oznako kdaj želimo, da se poročilo izdela.

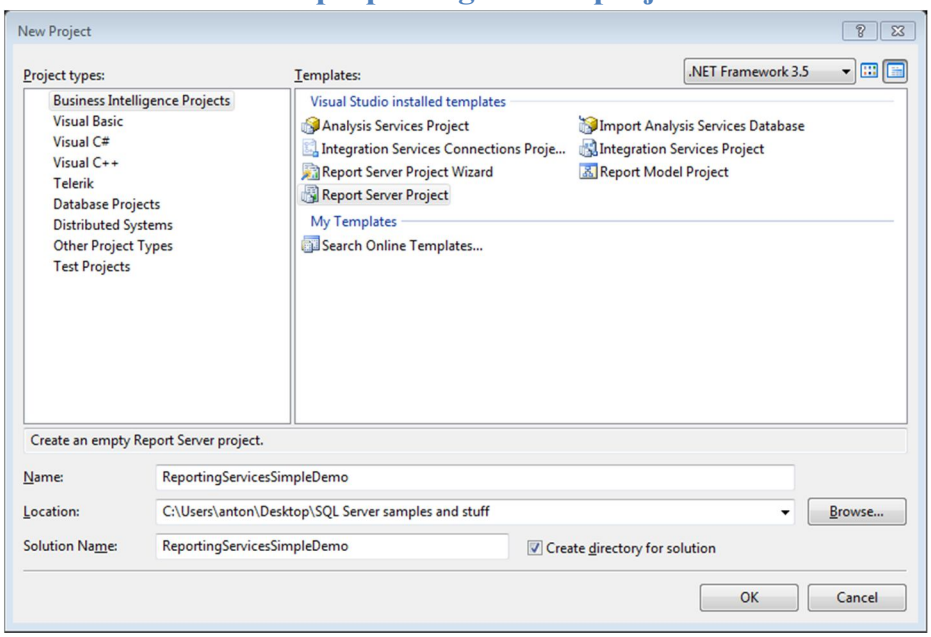
»Report Designer« je orodje, ki nam omogoča izdelavo in razhroščevanje poročil. Komponente, ki jih »Report Designer« vsebuje omogočajo izdelavo različnih vrst poročil. Poročila so lahko enstavne tabelarične ali matrične oblike, vendar pa lahko izdelamo tudi bolj kompleksna poročila, ki prikazujejo več nivojev informacij in podporočil, različne diagrame.

Znotraj poročil lahko implementiramo tudi različne funkcije in kalkulacije. Seznam nekaj najbolj zanimivih novosti znotraj SSRS:

- administratorji lahko določijo koliko spomina ima SSRS na voljo,
- IIS za strežnik za poročila ni več potreben za gostovanje in prikaz poročil,
- strežnik za poročila poskrbi tudi za vse overovitvene zahteve,
- podporočila in vgnezdene podatke lahko prikazujemo znotraj Excela,

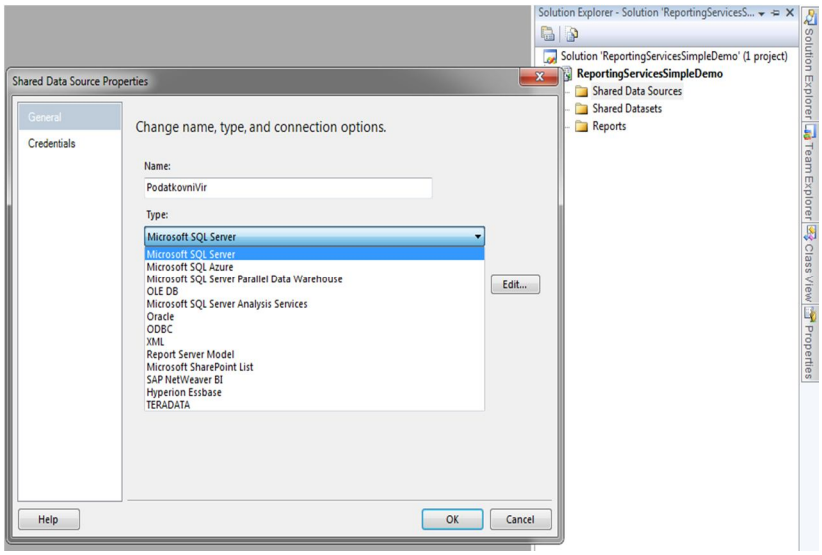
- poročila lahko prikazujemo znotraj »Windows Forms«, »Web Forms«, CSV, PDF, Images, Excel, Word, XML formatov,
- podprte so tudi »Dundas« kontrole za poročila,
- izboljšane kontrole za grafe
- poročila lahko izdelamo znotraj BIDS – »Business Intelligence Development Studia« ali znotraj »Report Designer« orodja

4. 1. Primer izdelave preprostega SSRS projekta

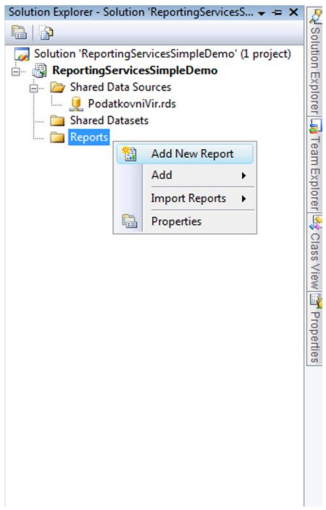


Slika 8 – začnemo nov SSRS projekt

Nov SSRS projekt začnemo tako, da ko zaženemo BIDS izberemo File → New → Project in znotraj »Business Intelligence Projects« izberemo predlogo »Report Server Project«.



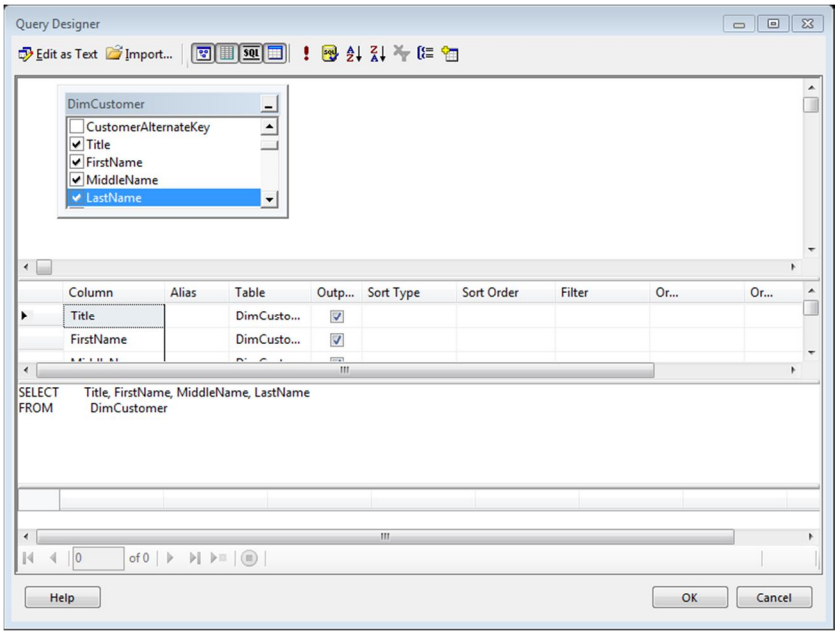
Slika 9 - nov podatkovni vir



Slika 10 - novo poročilo

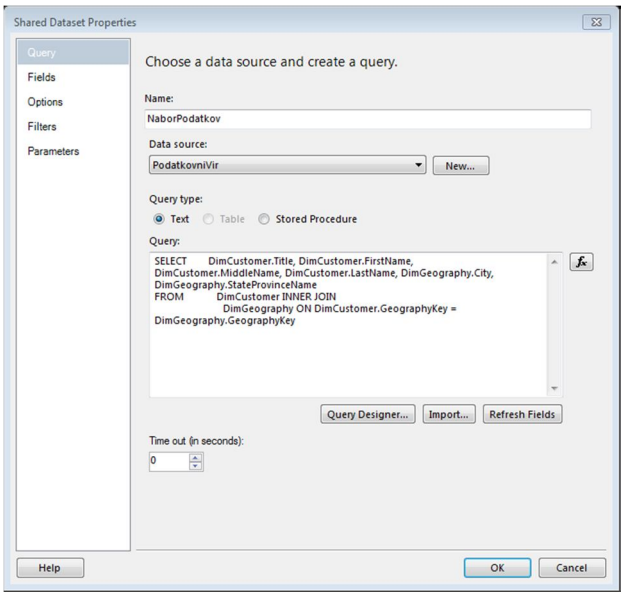
Nov podatkovni vir dodamo tako, da kliknemo na mapo »Shared Data Sources« in nato z desnim klikom iz prikazanih možnosti izberemo "Add New Data Source".

Novo poročilo najlažje izdelamo tako, da kliknemo na mapo "Reports" in nato z desnim klikom na miški, iz ponujenih možnosti izberemo "Add New Report" – zažene se čarovnik za gradnjo novega poročila.



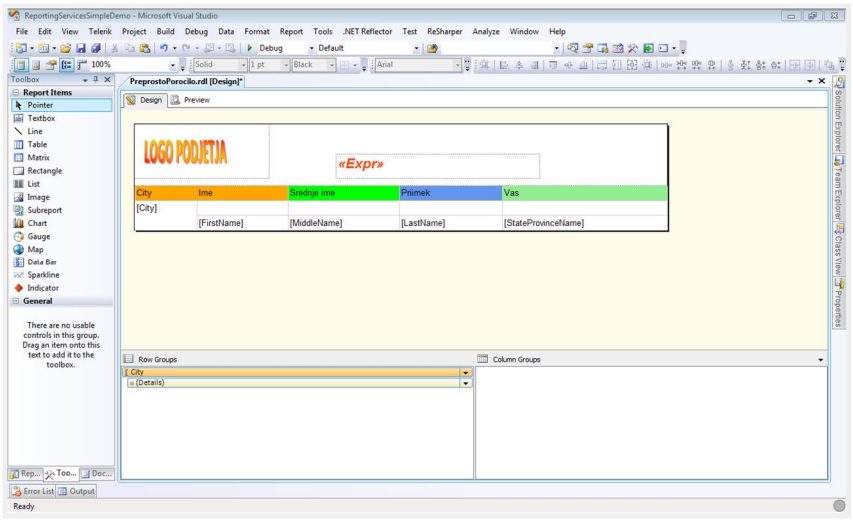
Slika 11 - "Shared Dataset« okno

Nov "Shared Dataset" dodamo na enak način, kot smo dodali nov podatkovni vir in poročilo - kliknemo na mapo "Shared Datasets" z desnim klikom iz možnosti izberemo možnost za dodajanje novega nabora podatkov. Odpre se nam okno za gradnjo SQL-a.



Slika 12 – okno z lastnostmi podatkov za poročilo

Ko smo končali z gradnjo SQL poizvedbe lahko s klikom na »Fiels«, »Options«, »Filters« in »Parameters« nastavimo še dodatne lastnosti našega nabora podatkov.



Slika 13 - oblikovanje poročila znotraj "Report Designer" orodja, ki je del BIDS-a

5 Analysis Services (SSAS) – platforma za analizo podatkov znotraj SQL Server 2008

Namen SQL Server Analysis Services (SSAS) je predvsem v tem, da zapolnijo praznino med potrebami poslovnih uporabnikov po podatkih, ki jih potrebujejo za svoje delo in IT oddelki, ki morajo te podatke zagotoviti.

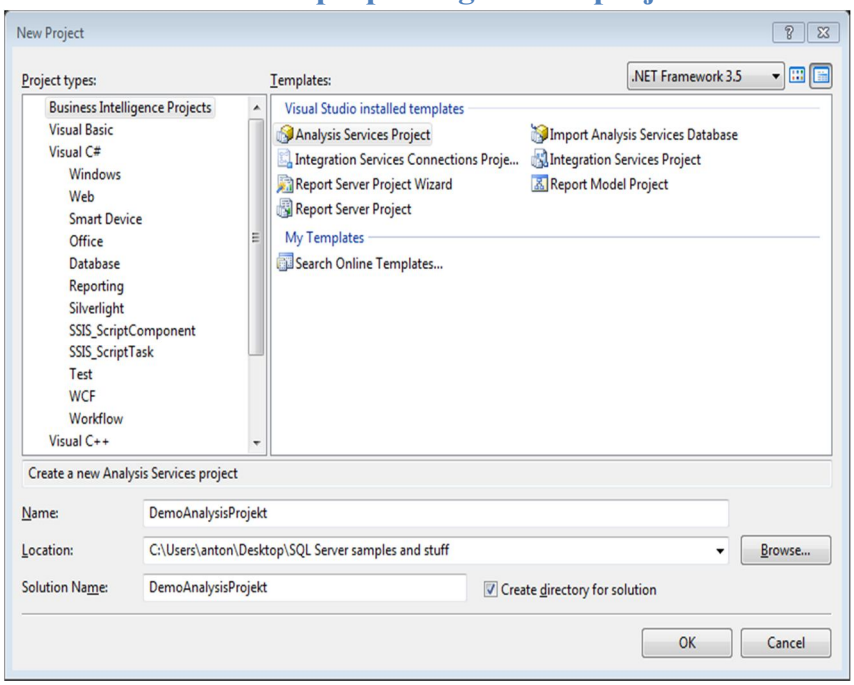
Funkcionalnost SSAS obsega predvsem OLAP in »Data Mining« (rudarjenje podatkov).

OLAP sistem omogoča postavitev, poizvedovanje in upravljanje podatkovnih kock, ki smo jih naredili z BIDS. Vključimo lahko več dimenzij znotraj katerih lahko zgradimo več hierarhij. Izberemo lahko tudi katere atribute želimo prikazovati in kako naj bodo podatki sortirani.

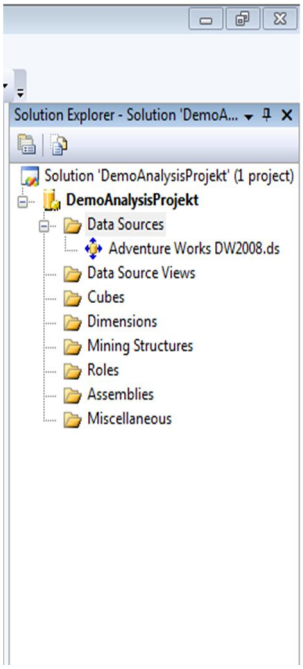
»Data Mining« sistem nam omogoča odkrivanje zakonitosti v podatkih. Ob tem lahko razširimo analizo poslovanja in s tem uporabniku omogočimo iskanje različnih vzorcev in lažje predvidevanje znotraj analiziranja teh podatkov.

Z uporabo katerega izmed algoritmov, ki so vgrajeni znotraj SQL Serverja 2008 lahko podatke poslovanja analiziramo skozi daljše časovno obdobje. Na podlagi teh analiz lahko izluščimo faktorje, ki bistveno vplivajo na prodajo. Te faktorje nato prilagodimo in s tem maksimiziramo prodajo.

5. 1. Primer izdelave preprostega SSAS projekta

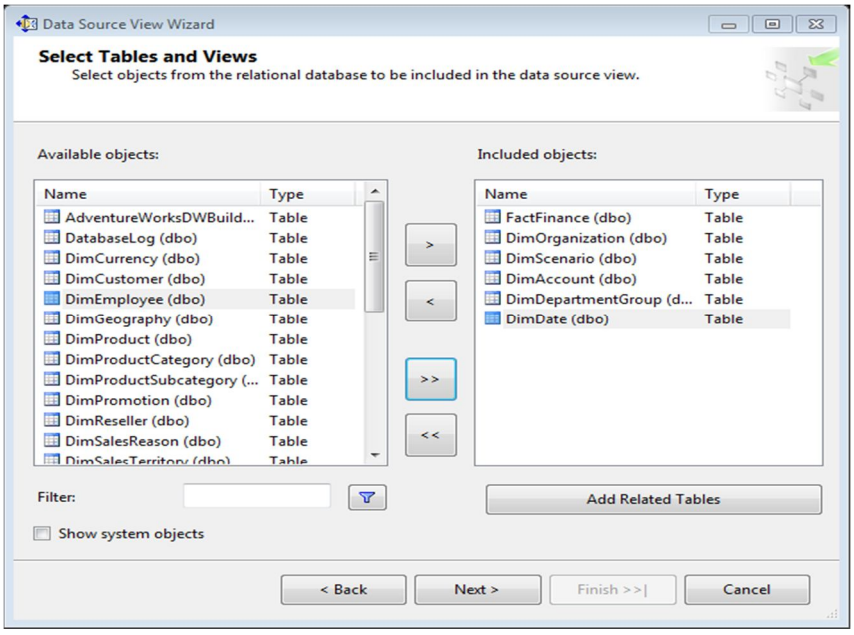


Slika 14 - nov SSAS projekt

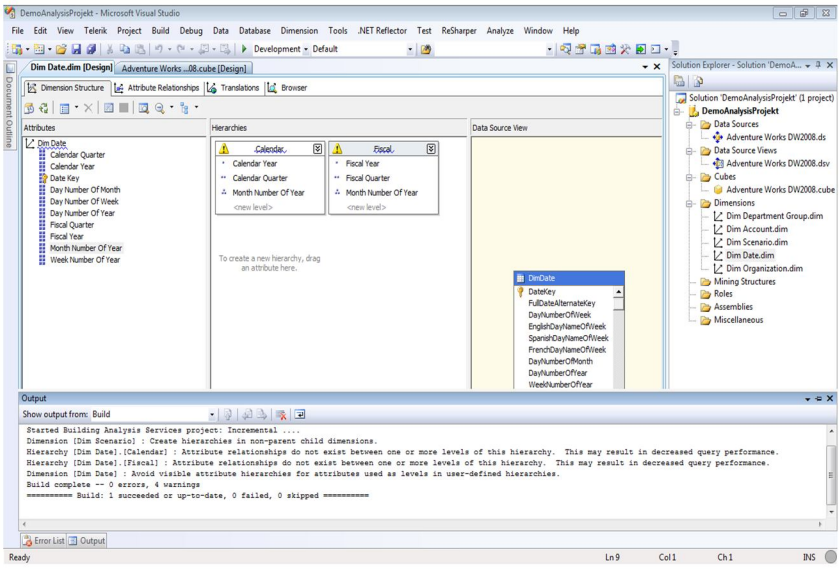


Slika 15 - z desnim klikom na mapo Data Sources našemu projektu dodamo nov podatkovni vir

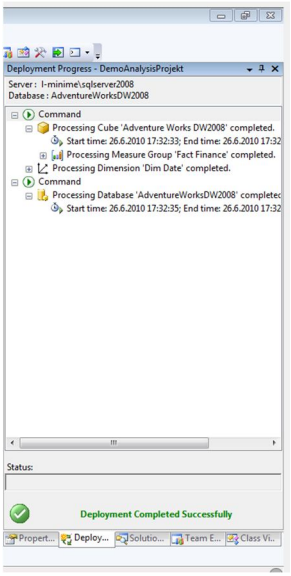
Nov SSAS projekt začnemo tako, da potem ko zaženemo BIDS izberemo File → New → Project in znotraj »Business Intelligence Projects« izberemo predlogo »Analysis Services Project«.



Slika 16 - z desnim klikom na mapo Data Source Views našemu projektu dodamo nov pogled za prikaz podatkov

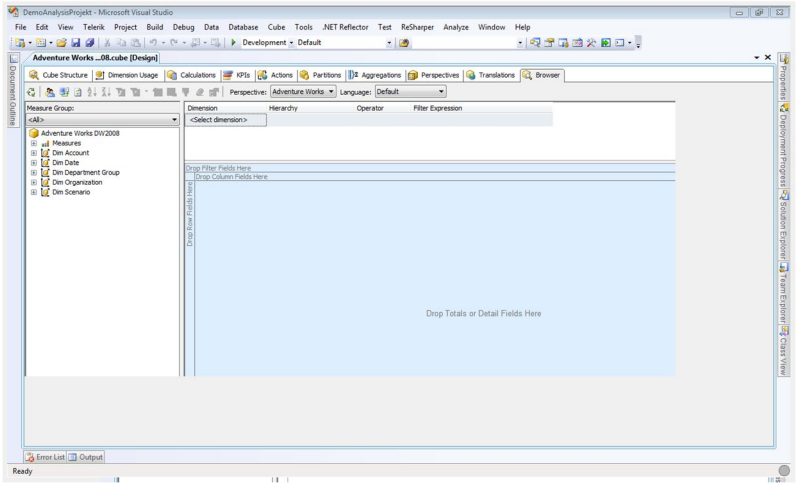


Slika 17 - izdelamo herarhično predstavitev znotraj posamezne dimenzije

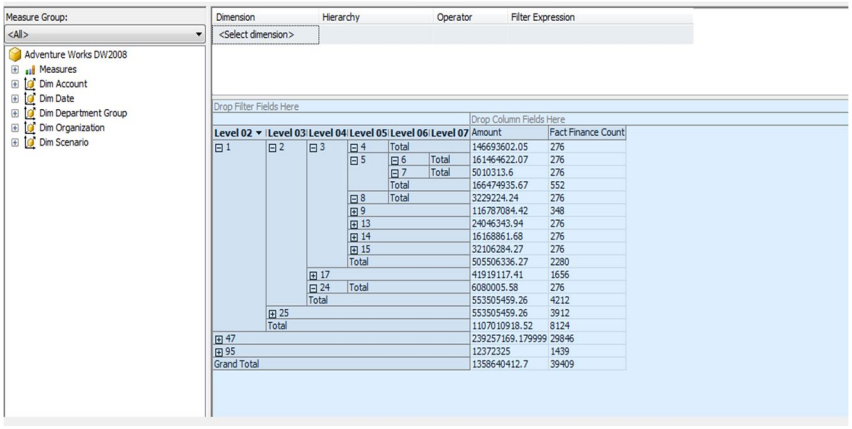


Slika 18 - "Deployment Completed Successfully"

Če smo vse storili pravilno, se ob »Deploy« operacij znotraj »Deployment Process« okna pojavi sporočilo "Deployment Completed Successfully".



Slika 19 - Cube Browser okno



Slika 20 – podatki znotraj Cube Browser okna

Ko določimo katere podatke podamo v "Drop Totals or Detail Fields Here" in katera polja določimo za vrstice, dobimo izpis podoben vrtilnim tabelam.

Ko delamo z »Analysis Services«, moramo biti zares pozorni, da imamo vse nastavitve pravilno nastavljene, saj drugače postavitev projekta ne uspe in kot rezultat dobimo napako.

6 BIDS – Business Intelligence Development Studio

Upravljanje z SSIS, SSAS in SSRS poteka znotraj »SQL Server Management Studia«. Razvoj poročil, OLAP kock in ostalih različnih modelov pa poteka znotraj orodja »Business Intelligence Development Studio«.

»Business Intelligence Development Studio« je v osnovi lupina razvojnega orodja »Visual Studio 2008«, ki podpira »SQL Server 2008 Business Intelligence« projekte.

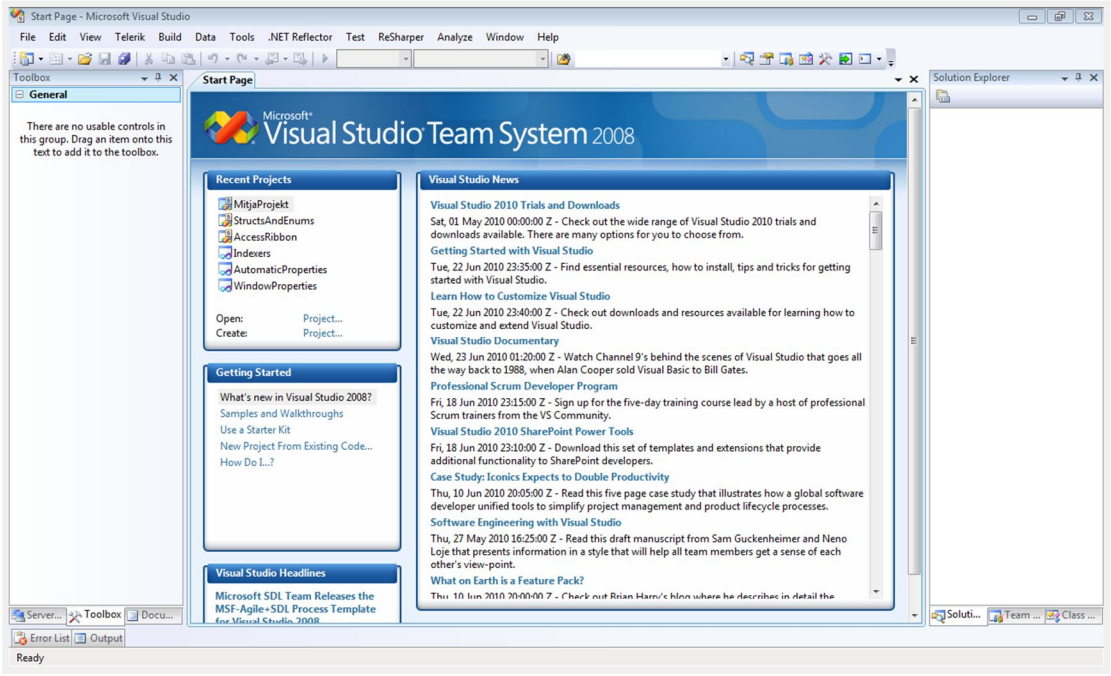
»Business Intelligence Development Studio« je primarno orodje za razvoj poslovnih rešitev, ki vključujejo »Integration Services«, »Reporting Services« in »Analysis Services«. Vsak izmed tipov projektov, ki jih izberemo vsebuje številne predloge za ustvarjanje objektov, ki jih potrebujemo za izdelavo specifične poslovne rešitve. Znotraj razvojnega orodja imamo na voljo številna orodja oblikovanja, čarovnike in druga orodja za delo z objekti, ki smo jih naredili.

6. 1. Začetna stran

Ob prvem zagonu BIDS se znotraj orodja, sredi uporabniškega vmesnika, naloži začetna stran. Na tej strani je navedenih zadnjih nekaj projektov, ki smo jih spreminjali. Navedene so tudi teme, ki so nam lahko v pomoč pri delu in iskanju informacij, vsebuje tudi naslove številnih spletnih strani, tehničnih člankov in drugih virov, povezave do novih produktov in drugih informacij.

Privzet je RSS kanal informacij in novic, nastavljen na Microsoftovo privzeto mesto preko katerega nas Microsoft obvešča o številnih novostih. Če si želimo vir novic prilagoditi, si lahko nastavimo poljuben vir, katerega novice želimo spremljati znotraj začetne strani orodja.

V primeru, da imamo na svojem sistemu že nameščeno orodje »Visual Studio 2008« in želimo uporabljati BIDS, se nam zažene »Visual Studio 2008«, kjer imamo poleg projektov za delo z »SQL Server 2008« platformo še ostale projekte, ki jih lahko razvijamo.

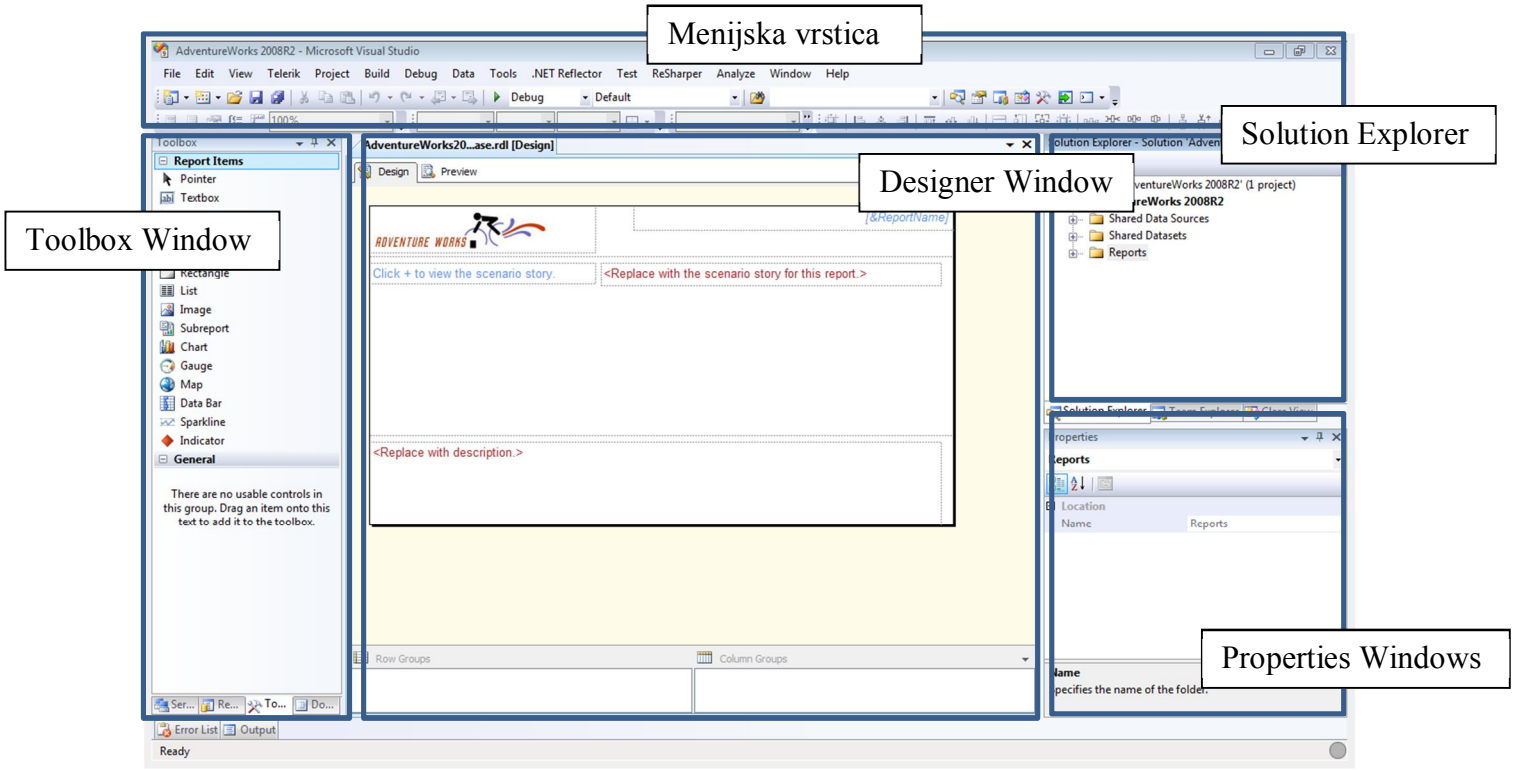


Slika 21 - začetna stran v BIDS (enaka začetni strani znotraj Visual Studia 2008)

BIDS orodje vsebuje številna okna, ki nam olajšajo delo na vseh stopnjah razvoja. S pomočjo teh oken lahko enostavno uporavljamo več projektov hkrati, ki navzven delujejo kot ena enota. Zelo enostavno lahko vidimo lastnosti posameznih projektov in njihovih objektov, ki jih po potrebi lahko tudi enostavno spremenimo.

Glavna okna, ki sestavljajo BIDS vmesnik so:

- »Solution Explorer«
- »Properties Window«
- »Designer Window«
- »Toolbox Window«



Slika 22 - različna okna znotraj BIDS vmesnika

6. 2. Solution Explorer

Znotraj »Solution Explorer« okna lahko upravljamo z različnim številom projektov in njim pripadajoče objekte znotraj trenutno odprte rešitve (ang. solution). Te projekte in objekte lahko odpremo za urejanje ali upravljanje neposredno iz tega okna. Ker različni tipi projektov pripadajoče objekte shranjujejo na različne načine ni nujno, da se direktorijska struktura ujema s fizično strukturo objektov znotraj posameznega projekta.

6. 3. Properties Window

Znotraj »Properties Windows« okna lahko spreminjamo lastnosti trenutno odprtega objekta. Pri tem moramo biti pozorni na to, da imajo različni objekti različne lastnosti. Za nekatere lastnosti lahko vrednosti izberemo iz različnih spustnih seznamov, nekatere vrednosti vnesemo ročno – pri tem moramo biti pozorni na pravilno formatiranje teh vrednosti.

6. 4. Designer Window

Znotraj »Design Window« okna lahko izdelamo ali spreminjamo obstoječe BI objekte. Samo okno sta v resnici dva okna v enem. Okno se namreč razdeli na del, ki prikazuje programsko kodo in na del, kjer imamo grafični prikaz objektov.

6. 5. Toolbox Window

Znotraj »Tookbox Window« okna imamo na voljo številne gradnike za uporabo znotraj BI projektov. Skupine zavihkov znotraj okna se spreminjajo odvisno od urejevalnika ali okna za dizajniranje, ki ga imamo trenutno aktivnega.

6. 6. Meniji

Meniji znotraj »Business Intelligence Development Studia« so v osnovi identični menijem, ki se nahajajo znotraj »Visual Studia 2008«. Tako kot je značilno za primer okna z lastnostmi kjer se lastnosti spreminjanjo odvisno od trenutno izbranega objekta, se tudi menijska vrstica spreminja glede na objekt, ki ga imamo izbranega.

6. 6. 1. File

Menijska možnost, ki nam omogoča delo z datotečnim sistemom. Znotraj te možnosti lahko odpiramo obstoječe ali izdelamo nove rešitve in projekte različnih tipov. Nekatere možnosti so sicer privzeto onemogočene in jih lahko uporablimo šele zatem, ko smo začeli z delom.

6. 6. 2. Edit

Menijska možnost, ki nam omogoča urejanje besedila in kode v datotekah.

6. 6. 3. View

Menijska možnost, ki nam omogoča urejanje samega uporabniškega vmesnika znotraj BIDS. Znotraj podmenijev lahko izbiramo med možnostmi, ki nam ponovno prikažejo različna okna, ki smo jih morda med samim delom zaprli, da bi pridobili na delovni površini.

6. 6. 4. Tools

Znotraj te menijske možnosti imamo možnost nastaviti številne nastavitve s katerimi lahko vplivamo na obnašanje razvojnega orodja (upravljanje z različnimi vtičniki, delo z zunanjimi orodji, itd.).

6. 6. 5. Window

Nam omogoča enostavno upravljanje z okni in raziskovalci znotraj »Business Intelligence Development Studia«.

6. 6. 6. Community

Enostavno nam omogoča zastavljanje vprašanj znotraj različnih skupnosti, iskanje po različnih virih, itd.. Prav tako lahko preko te možnosti enostavno prenesemo povratne informacije Microsoftu.

6. 6. 7. Help

Ta možnost nam omogoča enostavno delo s pomočjo in enostaven dostop do številnih enostavnih primerov uporabe posameznih funkcionalnosti.

6. 7. Verzioniranje

»Business Intelligence Development Studio« ima osnovo »Visual Studia«. Na voljo imam tako tudi funkcionalnost, ki nam omogoča uporabo programa za verzioniranje. V primeru, da imamo na sistemu nameščenega katerega izmed omenjenih programov za verzioniranje (»Subversion SVN, Visual Studio Team System in Team Foundation Server, Visual SourceSafe«, itd.) lahko naše rešitve in projekte dodamo v program za verzioniranje, katerega smo povezali z BIDS.

Iz programa za verzioniranje lahko te rešitve in projekte enostavno odpiramo znotraj »Business Intelligence Development Studia 2008«. S tem, ko se odločimo za verzioniranje naših rešitev ali projektov si svoj razvoj lahko zelo poenostavimo. Znotraj poslovnega okolja je verzioniranje skoraj obvezni del razvoja različnih rešitev.

Dandanes v veliki meri razvoj ni zgolj delo samo enega posameznika, temveč je sestavljeno iz večjega števila ljudi, ki imajo različne vloge. Verzioniranje nam omogoča enostavno povezavo z razvijalci. Ti razvijalci se lahko nahajajo na različnih lokacijah in vseeno delajo z istimi projekti in rešitvami.

V primeru, da za naše verzioniranje uporabimo brezplačno platformo »Subversion SVN«, nam platforma omogoča enstavno, a klub temu učinkovito verzioniranje.

Lahko pa se odločimo za uporabo »Team Foundation Server« platforme za verzioniranje. Z uporabo te platforme lahko združimo vse stopnje razvoja projekta. »Team Foundation Server« (imenovan tudi TFS) nam tako kot Subversion SVN, omogoča verzioniranje vseh objektov znotraj rešitve in projektov. Prav tako lahko zelo učinkovito zbiramo številne podatke, ki so povezani s projekti (napredek razvoja projektov, poročila, morebitne napake, rezultate testov).

Slika 23 - preprosta shema tri nivojske arhitekture TFS-ja

7 Entitiy Framework

Zakaj je Entitiy Framwork sploh prišel v uporabo, je več razlogov. Danes imamo na voljo veliko obstoječih in že dobro uveljavljenih tehnologij, ki nam omogočajo dostop do naših podatkovnih virov. Menim, da je Microsoft vložil veliko truda, časa in s tem tudi denarja v razvoj »ADO.Net« tehnologije. Tehnologijo »ADO.Net« razvijalci že vrsto let uporabljamo za dostopanje do podatkovnih virov. Sama tehnologija se je z vsako novo verzijo ».NET Framwork« platforme izboljševala in nadgrajevala in si tako pridobila vse večje zaupanje. Znotraj »ADO.Net« tehnologije so razvijalci za delo s podatki večinoma uporabljali »DataReader« in »DataSet« razreda.

Kljub vsem izboljšavam in nadgradnjam »ADO.Net« tehnologije je še vedno obstajala neka praznina med samimi aplikacijami, ki so jih razvijalci pisali in podatkovnimi bazami, katere so te aplikacije potrebovale.

Vendar pa so se ob tem pojavile tudi številne težave in vprašanja. Razvijalci so morali namreč skrbeti, da so se vse spremembe, ki so se odvijale na podatkovni bazi odražali tudi v sami aplikaciji. Tako so morali v primeru, da se je v tabeli znotraj podatkovne baze spremenilo lastnosti njene strukture (sprememba imena stolpca, sprememba podatkovnega tipa stolpca, izbris celotnega stolpca, itd.) razvijalci poskrbeti, da so bile spremembe narejene tudi znotraj same kode njihovih aplikacij.

To delo pa je razvijalcem vzelo veliko časa. Namesto, da bi nadaljevali z razvojem, so morali poskrbeti za te spremembe, zaradi katerih bi drugače aplikacije lahko prenehale delovati.

Koda v spodnji metodi nam prikazuje uporabo klasičnega »ADO.Net pristopa« (preprosta konzolna aplikacija, ki izpiše vrednosti stolpcev »FirstName«, »MiddleName«, »LastName« v tabeli »Person«. »Contact«, ki ima vrednost stoplca »ContactID« enako devet):

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Data.SqlClient;
using System.Data;

namespace SimpleConsoleConnectionToDatabaseTest
{
   class Program
   {
      static void Main()
      {
         string povezava  = GetConnectionString();
         string poizvedba =
            "SELECT pc.FirstName, " +
            "pc.MiddleName, " +
            "pc.LastName " +
            "FROM Person.Contact as pc " +
            "WHERE pc.ContactID = @contact";
         using (SqlConnection connection = new SqlConnection(povezava))
```

```
        {
            SqlCommand sqlUkaz = connection.CreateCommand();
            sqlUkaz.CommandText = poizvedba;
            SqlParameter parameter = new SqlParameter("@contact",SqlDbType.Int,10,"ContactID");
            parameter.Value = 9;
            sqlUkaz.Parameters.Add(parameter);
            try
            {
                connection.Open();
                SqlDataReader reader = sqlUkaz.ExecuteReader();
                while (reader.Read())
                {
                    Console.WriteLine("{0}, {1},{2}", reader[0], reader[1], reader[2]);
                }
                reader.Close();
            }
            catch (Exception ex)
            {
                Console.WriteLine(ex.Message);
            }
        }
    }
    static private string GetConnectionString()
    {
        return "Data Source=l-minime\\sqlserver2008 ;Initial Catalog=AdventureWorks;"
            + "Integrated Security=SSPI";
    }
  }
}
```

Koda sama po sebi ni nič posebnega, vendar pa že na takem enostavnem primeru vidimo možne težave, do katerih bi lahko prišlo v prihodnosti že, če bi naredili preprosto spremembo kot je sprememba imena stolpca (torej, če bi namesto »FirstName« pisalo »FistName« med samim prevajanjem kode ne bi dobili obvestila o napaki, napaka bi se pojavila šele, ko bi aplikacijo začeli uporabljati).

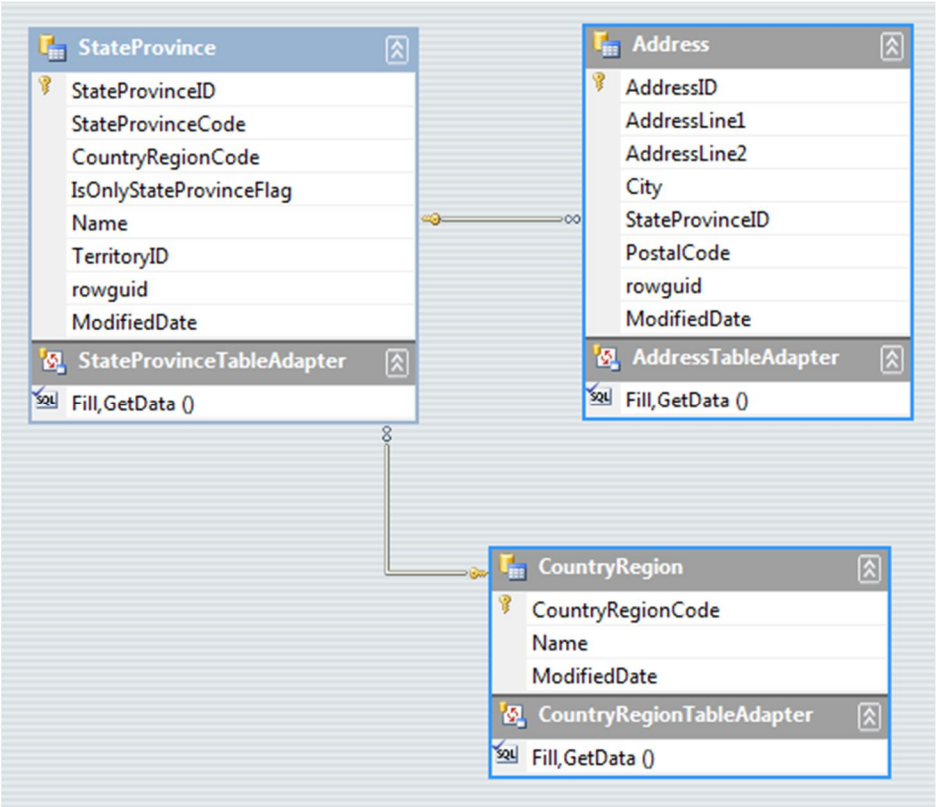
Tako se je pojavila potreba po modelu, kjer se lahko na najbolj enostaven način aplikacije, podatkovne baze in same podatke poveže skupaj. Ravno temu seveda služi »Entity Framework«. S tem v mislih ne smemo jemati »Entity Framwork«, kot nadomestilo »ADO.Net« tehnologiji, temveč kot dodatek k omenjeni tehnologiji.

»Entity Framework« je konceptualni model, ki deluje s podatkovnimi bazami in aplikacijami in pomaga premostiti ovire, ki smo jih razvijalci imeli s tehnologijami, kot so »DataReader« in ostale. Je skupek tehnologij znotraj »ADO.Net«, ki nam pomaga pri reševanju ovir z objektno orientiranim razvojem in podatkovnimi bazami.

7. 1. Podatkovni entiteni model in entitetni model znotraj Entity Frameworka

Sedaj imamo na voljo dva načina za dostopanje do podatkov. Na eni strani imamo dobro uveljavljene »DataReader« in »DataSet« objekte, na drugi strani pa imamo »Entity Framework«. Kaj izbrati je odvisno od scenarija, ki ga moramo implementirati.

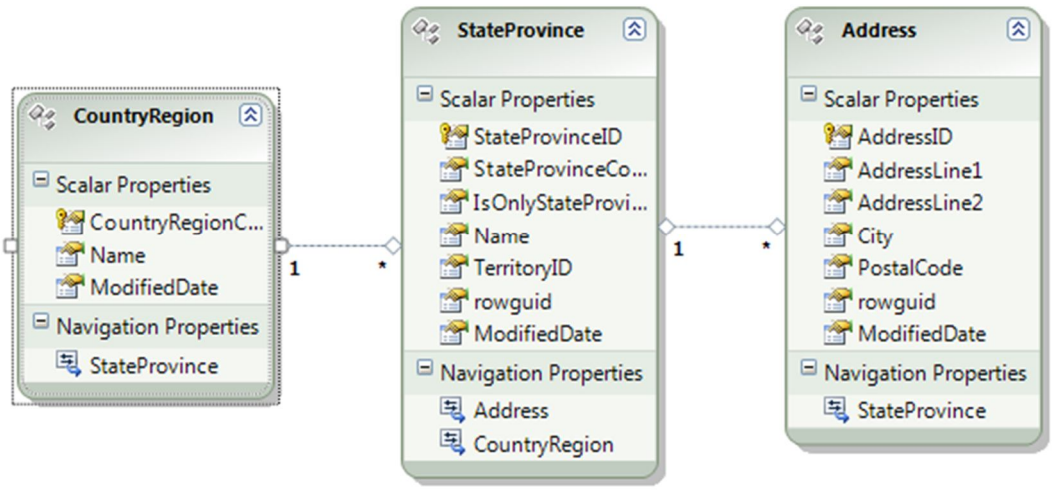
Pri klasičnem »DataReader« in »DataSet« pristopu moramo kodo aplikacije združiti s podatki.



Slika 24 - del podatkovnega modela

Programerji moramo s pomočjo kode še vedno poskrbeti za pravilno ujemanje stolpcev in vrstic, ki jih dobimo vrnjene kot rezultate in njihovo mapiranje v objekte znotraj naše aplikacije. V primeru, da rezultati iz podatkovne baze niso taki kot bi si jih želeli, moramo programerji sami spisati poizvedbe, ki nam vrnejo željeni rezultat.

To pa pomeni tudi to, da moramo dovolj dobro poznati tudi sintakso in vsaj osnovne funkcionalnosti SQL jezika za učinkovito in optimalno poizvedovanje.



Slika 25 - del entitetnega podatkovnega modela

Entitetni model »EntityFrameworka« ni enak entitetnem modelu podatkovne baze. Diagrami »Entity Frameworka« opisujejo strukturo vaših poslovnih objektov, kjer diagrami podatkovne baze opisujejo sheme znotraj baze. Z »Entity Frameworkom« sedaj delamo na nivoju objektnega programiranja, ki je različen od tabelaričnega nivoja, ki smo ga navajeni pri klasičnem »ADO.Net« pristopu.

Ni nam več potrebno skrbeti za transformacije podatkov k objektom, saj so naše entitete že predstavljene kot objekti s katerimi lahko delamo po klasičnem konceptu objektnega programiranja.

Entitete so tako v mnogo pogledih enake objektom, saj:

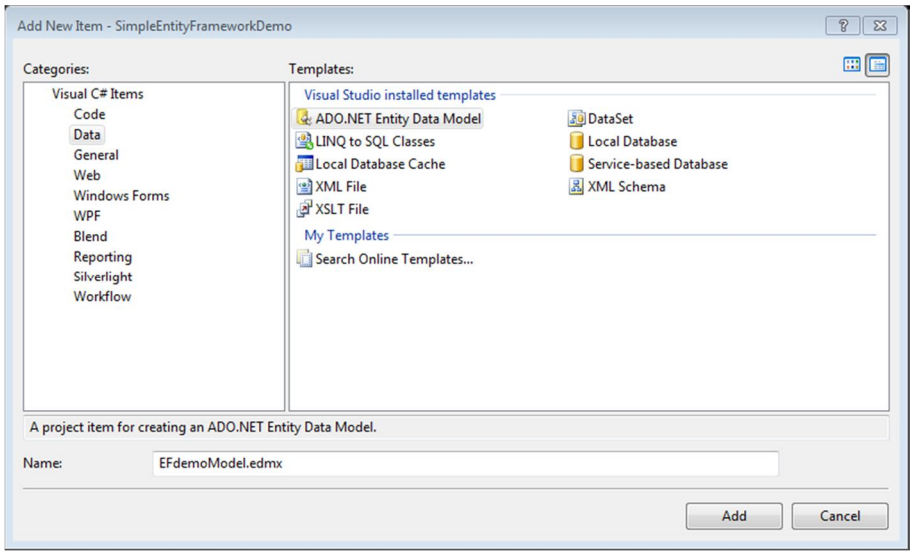
- imajo znan podatkovni tip,
- imajo lastnosti, te lastnosti pa lahko hranijo skalarne vrednosti,
- lastnosti lahko hranijo tudi reference na druge entitete,
- vsaka entiteta ima enolično določilo.

Potem, ko imamo entitni model z uporabo »Entity Framworka« že narejen, moramo narediti tudi ustrezne poizvedbe. Sedaj se moramo zavedati, da poizvedovanje z uporabo »Entity Frameworka« ne poteka neposredno po podatkovni bazi znotraj katere hranimo podatke, temveč poizvedbe gradimo na podlagi objektnega modela.

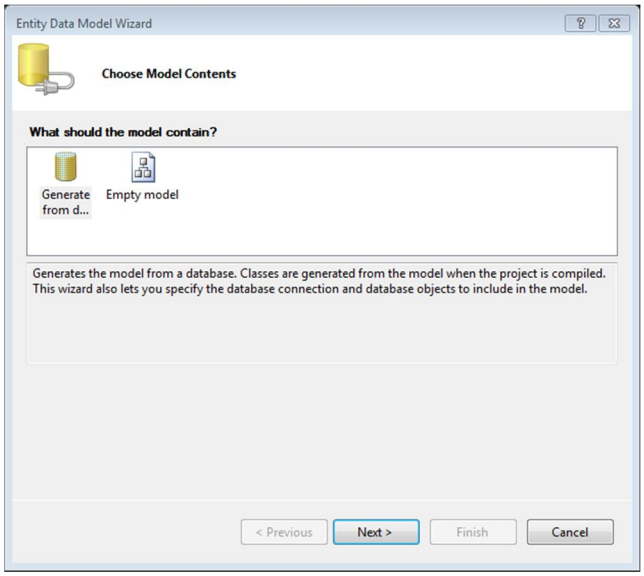
Poizvedovanje sedaj poteka na podlagi objektnega modela, kar s seboj prinese tudi nekaj sprememb. Sama sintaksa poizvedb je bistveno drugačna. Namesto pisanja klasičnih T-SQL poizvedb sedaj uporabljamo LINQ (Language Integrated Query).

»Entity Framework« za procesiranje poizvedb, ki smo jih sedaj naredili z uporabo LINQ-a, uporabi funkcionalnost »ADO.Net« ponudnikov (predvsem ponudnika

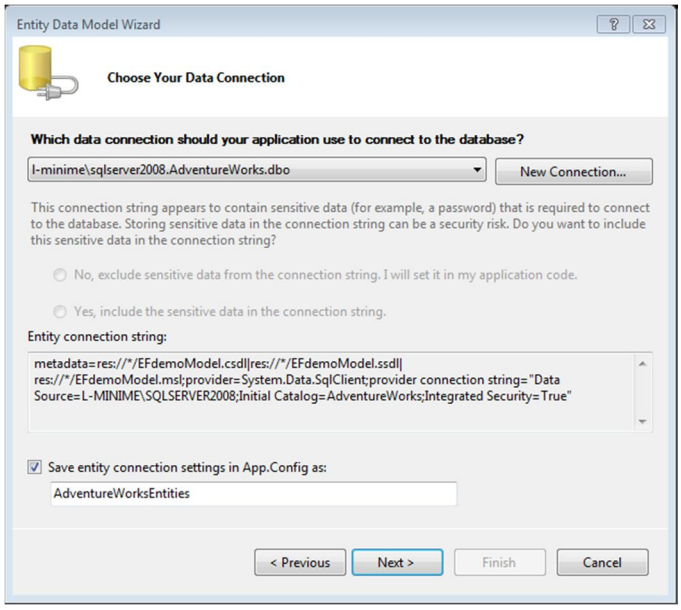
»System.Data.SqlClient«, ki LINQ poizvedbe pretvori v nekaj, kar »SQL Server 2008« razume in zna obdelati).



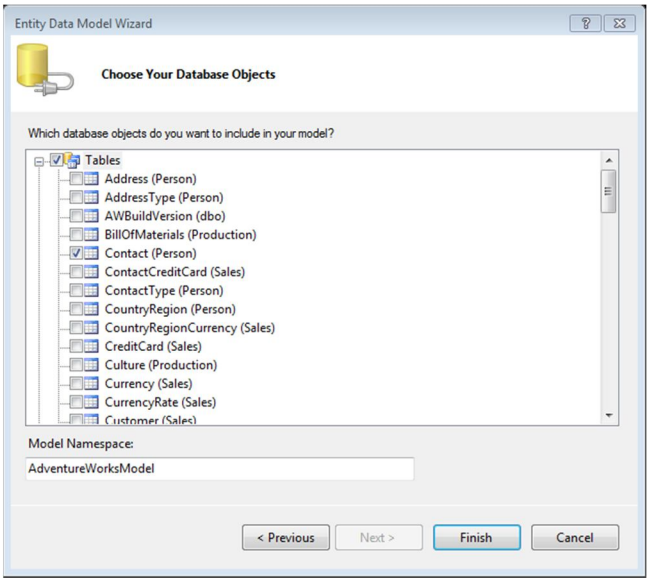
Slika 26 - za uporabo EntityFrameworka moramo izdelati podatkovni model - izberemo ADO.NET Entity Data Model predlogo



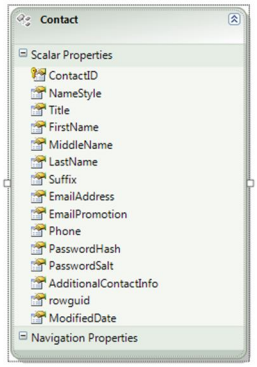
Slika 27 - izberemo Generate From Database



Slika 28 - v primeru, da nimo še narejene povezave na našo podatkovno bazo, to storimo v tem koraku čarovnika



Slika 29 - izberemo tiste tabele, poglede in procedure katere potrebujemo za delo (v mojem primeru izberem samo tabelo Contact - znotraj sheme Person)



Slika 30 - dobim preprost objektni model izbrane entitete oz. tabele

Znotraj preproste konzolne aplikacije sedaj lahko poizvedujemo s pomočjo LINQ-a:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
namespace SimpleEntityFrameworkDemo
{
   class Program
   {
      static void Main(string[] args)
      {
         int vrednost = 9;
         var kotekstPodatkov = new AdventureWorksEntities();
         var linqPoizvedba = from oseba in kotekstPodatkov.Contact
                             where oseba.ContactID == vrednost
                             select oseba;

foreach (var oseba in linqPoizvedba)

         {
            Console.WriteLine("{0}, {1}, {2}", oseba.FirstName, oseba.MiddleName, oseba.LastName);
         }
      }
   }
}
```

Vidimo, da je LINQ sintaksa podobna T-SQL sintaksi:

T-SQL

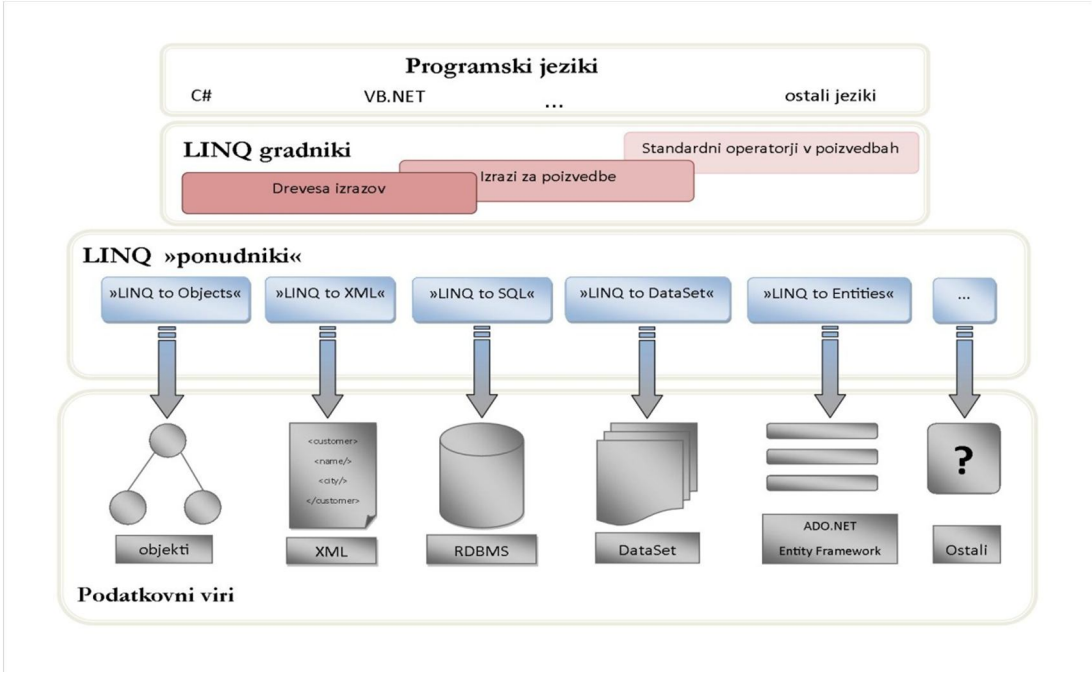
```
string poizvedba =
    "SELECT pc.FirstName, " +
    "pc.MiddleName, " +
    "pc.LastName " +
    "FROM Person.Contact as pc " +
    "WHERE pc.ContactID = @contact";
```

```
var linqPoizvedba = from oseba in kotekstPodatkov.Contact
                    where oseba.ContactID == vrednost
                    select oseba;
```

LINQ

Slika 31 - levi del prikazuje T-SQL sintakso poizvedbe - enstaven string, desno pa imamo poizvedbo z uporabo LINQ-a - objektni pristop

7. 2. Nekaj o LINQ-u



Slika 32 - shema LINQ-a

LINQ – »Language Integrated Query« je veliko več kot samo model za pretvarjanje SQL-a v objektni model.

Je metodologija s pomočjo katere lahko bistveno poenostavimo, predvsem pa poenotimo dostopanje do naših podatkov. Poleg tega je tudi programski model s pomočjo katerega poizvedbe (kakršne koli vrste že so), znotraj katerega koli Microsoft.Net programeskega jezika, lahko predstavimo kot razrede. Samo obvladovanje LINQ-a sicer zahteva nekaj novega učenja sintakse, vendar pa si lahko na koncu s tem bistveno povečamo samo produktivnost našega dela in s tem čas odkrivanja in odpravljanja napak (ob spremembi podatkovne baze).

»Language Integrated Query« (LINQ) obstaja v več različicah. Sama razširljivost ga naredi zelo primernega za uporabo pri vseh scenarijih, kjer se srečamo z manipulacijo s podatki.

V sledečih poglavjih bom na kratko predstavil nekaj teh različic.

7. 2. 1. LINQ to Entities

Kadar delamo z »Entity Framoworkom« smo videli, da nad fizičnimi podatki relacijske podatkovne baze lahko naredimo abstrakcijski nivo. Z »LINQ to Entites« lahko poizvedujemo po entitetah na tem nivoju namesto nad fizičnimi podatki.

7. 2. 2. LINQ to SQL

Najbolj očitna in najbolj razširjena različica LINQ-a nam služi za poizvedovanje po relacijskih podatkovnih bazah. Rezultat takih poizvedb je objektni model na podlagi obstoječih entitet.

7. 2. 3. LINQ to DataSet

V osnovi je »System.Data.DataSet« nabor podatkov, ki jih aplikacija hrani v spominu. Zelo prav nam pride v primeru, ko nimamo aktivne povezave do naše podatkovne baze, vseeno pa imamo tako na voljo kopijo podatkov s katerimi lahko delamo. Interno so DataSet-i predstavljeni kot interna relacijska podatkovna baza tako, da so zato zelo primerni za implementacijo LINQ-a.

7. 2. 4. LINQ to XML

Z uporabo XML (»Extensible Markup Language«) formata si lahko olajšamo prenosljivost naših podatkov med različnimi sistemi. S stališča razvijalcev je XML še vedno ne preveč priljubljena tehnologija, predvsem zaradi svoje stroge sintakse. Včasih je zelo težko implementirati logiko, kjer so podatki shranjeni znotraj XML datoteke. Na voljo imamo številne specifikacije (XSD - XML Schema Definition – shema, ki določa strukturo XML dokumenta, XSLT – Extensible Stylesheet Language for Transformation – navodila za mapiranje podatkov med različnima shemama, XPath inXQuery - za poizvedovanje po XML dokumentu, DOM – Document Object Model – delo z dokumentom, ki je shranjen v spominu, itd.), ki nam to delo lahko precej olajšajo, vendar pa je to lahko časovno zelo potratno.

»LINQ to XML« nam s pomočjo Microsoft.Net programske kode omogoča poizvedovanje po XML datotekah brez učenja nekih dodatnih specifik. Tu sta združeni moči DOM tehnologije in učinkovitost XPath in XQuery tehnologije.

8 Primerjave tehnologij – morebitne alternative

»Microsoft SQL Server 2008« je na voljo v različnih distribucijah. V moji diplomski nalogi sem uporabljal »Microsoft SQL Server 2008 Enterprise Edition«. V to verzijo platforme je vključena celotna platforma za implementacijo poslovne logike. Del te plaforme so tako tudi SSIS, SSRS in SSAS.

Za Microsoftovo platformo sem se odločil predvsem zato, ker ob verziji platforme, ki sem jo izbral, ni potrebno nameščati dodatnih orodij.

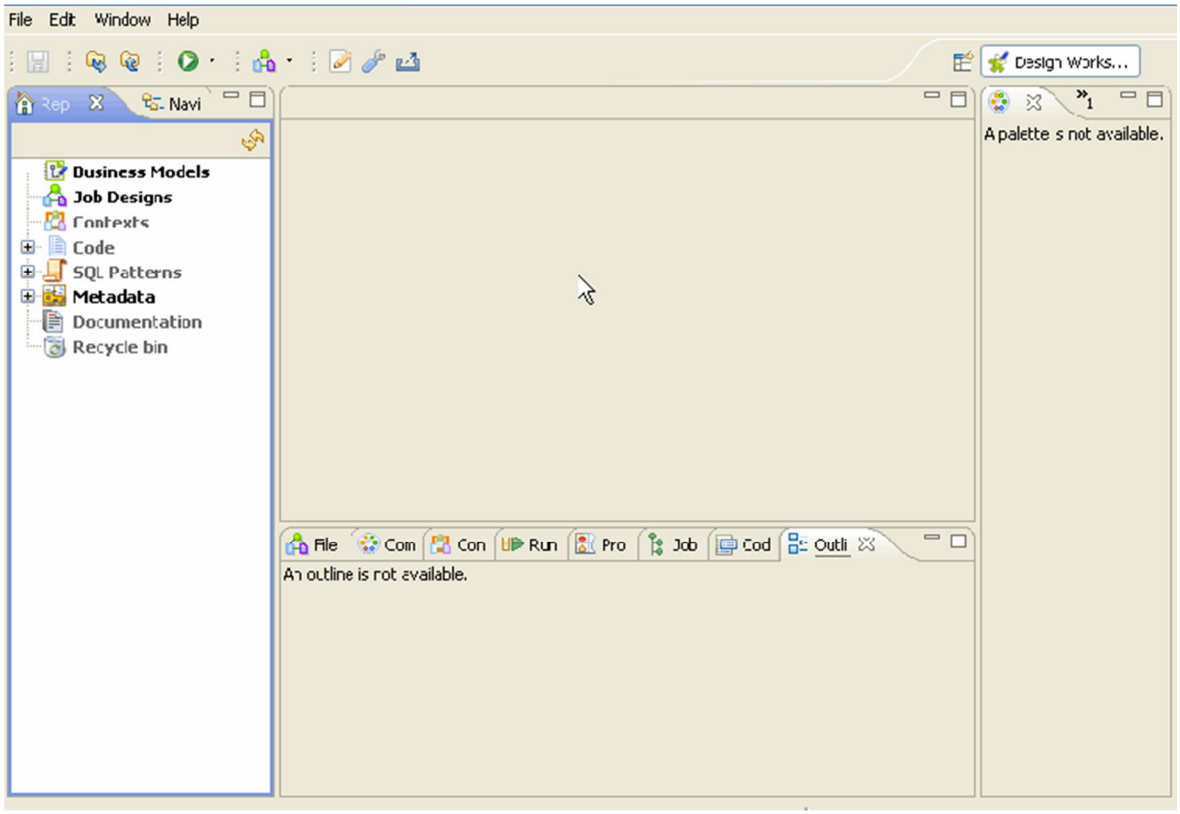
Vseeno sem za implementacijo rešitev poslovne logike našel odprtokodno rešitev, ki se mi je zdela kot najboljša alternativa, imenovano »Talend«. »Talend« je prav tako platforma, ki nam izredno olajša implementacijo poslovne logike v naših podatkih. »Talend« je odprtokodna rešitev, ki podpira ogromno raznovrstnih podatkovnih baz (Oracle, MS SQL, MySQL, PostgreSQL, itd.) in drugih podatkovnih virov (aplikacije, elektronska pošta, spletne storitve, itd.) iz katerih lahko pobiramo podatke. Del te platforme je tudi razvojno okolje imenovano »Talend Open Studio«, znotraj katerega lahko te rešitve razvijamo. Znotraj »Talend Open Studia« je na voljo ogromno gradnikov, s katerimi lahko izdelamo raznolike scenarije, ki so naš tok podatkov znotraj poslovne logike.

»Talend« deluje znotraj Java okolja. To, da deluje znotraj Java okolja pomeni, da uporaba ni omejena zgolja na Microsoftov operacijski sistem. »Talend Open Studio« je zgrajen na platformi razvojnega okolja »Eclipse«. Za projekte in rešitve lahko uporabljamo bodisi programski jezik Java ali pa se odločimo za programski jezik »Perl«. Sama platforma je s tem na voljo številnim organizacijam različnih velikosti in organizacijskih oblik, ki imajo posameznike z različnimi stopnjami predznanja. Predvsem je primerna tudi za podjetja z zelo majhnim proračunom.

Žal pa pri platformi »Talend« nisem našel primernega poročilnega sistema, ki ga »SQL Server 2008« ima. »SQL Server 2008« namreč namestimo s svojim strežnikom za poročila in ne potrebuje dodatnega spletnega strežnika za prikaz poročil. Vendar pa so alternative SSRS vse naše aplikacije, ki nam omogočajo izdelavo in prikaz poročil s podatki, ki jih aplikacije izpisujejo.

Tudi analize podatkov lahko delamo v drugih orodjih. Eden izmed takih orodij, ki nam za analizo naših podatkov pride zelo prav, je »Microsoft Excel«. Mnogi uporabniki podcenjujejo Office orodja, vendar pa je ravno »Excel« tisto orodje, ki nam omogoča analiziranje podatkov.

V različici Microsoft SQL Serverja 2008, ki sem jo uporabil sem za razvoj uporabil BIDS. BIDS je lupina Visual Studia 2008, katerega sem imel že nameščenega in zato nisem imel občutka, da pri svojem delu uporabljam kakšno drugo orodje. »Visual Studio 2008« je razvojno orodje, ki ga uporabljam pri svojem delu. Samega učenja in spoznavanja novega razvojnega okolja zato skoraj ni bilo.



Slika 33 - grafični vmesnik Talend Open Studia

9 Sklepna ugotovitev

Poslovne aplikacije so danes »nujno zlo« v vseh srednjih in velikih podjetjih. »SQL Server 2008« nam nudi visoko razpoložljivo platformo, ki jo lahko enostavno razširimo glede na potrebe, ki jih ima podjetje.

Same tri tehnologije SSIS, SSAS in SSRS končnim uporabnikom omogočajo izdelavo številnih rešitev s pomočjo katerih lahko podatke predstavimo na uporabiku prijazen način. Predvsem bi tu rad izpostavil poročilni sistem, ki nam zagotovo omogoča izdelavo poročil v trenutku, ko le-te potrebujemo. Ni potrebno, da smo odvisno od tretje stranke, ki ta poročila naredil namesto nas. S tem lahko tudi veliko privarčujemo.

Res je, da je sprva potrebno kar nekaj časa za to, da omenjene tri tehnologije osvojimo, vendar pa s tem lahko kasneje dobro izkoristimo celoten nabor omenjenih tehnologij za implementacijo poslovne logike.

Tudi »Entity Framwork« in LINQ nam končno nudita platformo, ki je sicer na voljo že nekaj časa. Razvijalci lahko z objekti podatkovne baze delamo kot z navadnimi objekti znotraj .NET programskega jezika.

PRILOGE

Primeri aplikacij in rešitev prikazanih v poglavjih diplomske naloge, se nahajajo na priloženem CD-ju.

KAZALO SLIK

VIRI IN LITERATURA

[1] Microsoft SQL Server 2008 platform overview. Dostopno na spletni strani: http://www.microsoft.com/sqlserver/2008/en/us/overview.aspx

[2] Whitepaper owerview. Dostopno na spletni strani: http://download.microsoft.com/download/6/9/d/69d1fea7-5b42-437a-b3ba-a4ad13e34ef6/SQL2008_ProductOverview.docx

[3] Microsoft TAG. Dostopno na spletni strani: http://www.microsoft.com/tag/content/download/

[4] Definicija T-SQL. Dostopna na spletni strani: http://en.wikipedia.org/wiki/Transact-SQL

[5] Definicija SQL. Dostopna na spletni strani: http://en.wikipedia.org/wiki/SQL

[6] SQL SERVER 2008 Application development. Dostopno na spletni strani: http://www.microsoft.com/sqlserver/2008/en/us/app-dev.aspx

[7] AdventureWorks 2008 R2 podatkovne baze. Dostopno na spletni strani: http://msftdbprodsamples.codeplex.com/releases/view/45907

[8] O Entity FrameWorku. Dostopno na spletni strani: http://msdn.microsoft.com/en-us/library/bb399572.aspx

[9] SQL Server Reporting Services tutorials. Dostopno na spletni strani: http://msdn.microsoft.com/en-us/library/ms167031.aspx

[10] O ADO.NET Entity Framework. Dostopno na spletni strani: http://msdn.microsoft.com/en-us/library/bb399572.aspx

[11] Kako začeti delo v Business Intelligence Development Studiu. Dostopno na spletni strani: http://msdn.microsoft.com/en-us/library/ms173767.aspx

[12] O razvoju aplikacij s pomočjo SQL Serverja 2008. Dostopno na spletni strani: http://www.microsoft.com/sqlserver/2008/en/us/app-dev.aspx

[13] Hotek Mike, »Microsoft SQL Server 2008 Step by Step«, Microsoft Press, 1st edition, November 12, 2008

[14] Russo Marco, »Introducing Microsoft LINQ«, Microsoft Press, May 16, 2007

[15] Talend, odprtokodna rešitev Microsoftovim Integration Services. Dostopno na spletni strani:
http://talend.com/index.php