Univerza na Primorskem

FAKULTETA ZA MATEMATIKO, NARAVOSLOVJE IN INFORMACIJSKE TEHNOLOGIJE

ZAKLJUČNA NALOGA

Selitev aplikacije iz Oracle Forms v Oracle ADF
(Application migration from Oracle Forms to Oracle ADF)

Ime in priimek: Dejan Ljubič
Študijski program: Računalništvo in informatika, 1. stopnja
Mentor: doc. dr. Iztok Savik

Zahvala

Zahvaljujem se doc. dr. Iztoku Savniku za mentorstvo, napotke, razpoložljivost, prijaznost ter pomoč pri izdelavi zaključne naloge. Zahvalil bi se tudi podjetju In2 Koper za ponujeno možnost uporabe Oraclove programske opreme, ter Študentski organizaciji Univerze na Primorskem za pridobljene delovne navade tekom študija. Iskrena zahvala gre tudi mojim staršem ter vsem bližnjim, ki so me v času študija podpirali ter popestrili študentske dni.

Povzetek

V pričujoči zaključni nalogi sta predstavljena opisni ter praktični del selitve aplikacije iz programskega okolja Oracle Forms v okolje Oracle ADF. V opisnem delu so podrobneje predstavljeni tako obe okolji kot razlogi in dejavniki za selitev ter pristopi za izvedbo same selitve. V praktičnem delu zaključne naloge pa so opisani postopki izdelave osnovnih aplikacij v obeh tehnologijah. Izdelava teh aplikacij je prikazana tudi v slikovnem gradivu. Za izdelavo aplikacij sta uporabljeni orodji Oracle Designer 6i in Oracle JDeveloper 11g z dodatkom JHeadstart in knjižnico OraFormsFaces.

Ključne besede:
informacijski sistemi, podatkovne baze, spletne aplikacije, relacijski sistemi za upravljanje podatkovnih baz, selitev aplikacij.

Abstract:

This thesis presents the theory and practice of application migration from Oracle Forms programming environment to Oracle ADF environment. The descriptive part presents both environments as well as the reasons and factors that influence the migration approaches. The practical part describes how to develop simple application in both technologies. The development steps are shown using screenshots of the application building process. The application development tools used in the study were Oracle Designer 6i and Oracle Jdeveloper 11g tools with the addition of JHeadstart and OraFormsFaces library.

Keywords:
information systems, databases, web applications, relational systems, database management, application migration.

Slovar

Ajax	(ang. Asynchronous JavaScript) asinhroni JavaScript in XML je skupina medsebojno povezanih spletnih razvojnih tehnik, uporabljenih za ustvarjanje interaktivnih spletnih aplikacij.
HTML	(ang. Hyper Text Markup Language) jezik za označevanje nadbesedila je označevalni jezik za izdelavo spletnih strani.
JSF	(ang. JavaServer Face) v Javi osnovano ogrodje za razvoj spletnih uporabniških vmesnikov.
JSP	(ang. JavaServer Pages) je javanska tehnologija za izdelavo spletnih strani z dinamično vsebino.
PL/SQL	(ang. Procedural Language/Structured Query Language) proceduralni jezik/strukturirani poizvedovalni jezik je proceduralen jezik prilagojen za delo s podatkovnimi bazami.
POJO	(angl. plain old Java object) goli Java objekt
SQL	(ang. Structured Query Language) strukturiran poizvedovalni jezik, jezik za dostop, manipuliranje in kontrolo podatkov v podatkovni bazi.
XML	(ang. Extensible Markup Language) razširljiv označevalen jezik, je računalniški jezik podoben HTML-ju, ki omogoča format za opisovanje strukturiranih podatkov.

Kazalo vsebine

Kazalo slik:

Kazalo tabel:

1. UVOD

Vse več podjetij se danes odloča za uporabo spletnih in uporabniku prijaznejših aplikacij, ki hkrati nudijo tako modernejšo podobo ter razne grafične učinke kot tudi širok nabor funkcijskih možnosti. Z izbiro prave tehnologije lahko razvijalcem olajšamo delo in prihranimo čas pri razvoju aplikacij.

Novonastala podjetja se v tem primeru lažje odločijo za tehnologijo, v katero bodo usmerila razvoj aplikacije, medtem ko je ta odločitev zahtevnejša v podjetjih, ki so dlje časa prisotna na trgu in zato uporabljajo starejše tehnologije. V drugem primeru se namreč pojavi težava pri selitvi aplikacij iz starejših delujočih sistemov na tehnološko novejše, še posebej če je bilo v razvoj samih aplikacij vloženih več let dela ter če se podjetje preživlja s prodajo ter vzdrževanjem le-teh.

V zaključni nalogi obravnavamo problem selitve aplikacij iz razvojnega okolja Oracle Forms v tehnološko novejše razvojno okolje Oracle ADF.

V prvem delu naloge sta predstavljena oba sistema. Opisani sta okolji, v katerih razvijamo aplikacije, podani so razlogi in dejavniki, katere moramo upoštevati pri selitvi aplikacije iz enega sistema v drugega ter različni pristopi k selitvi aplikacij.

V drugem delu so predstavljeni praktični rezultati naloge: prikazani so pristopi, ki jih uporabljamo pri selitvi aplikacij iz Oracle Forms v Oracle ADF, sledi krajša predstavitev podatkovne baze, katero uporabljamo pri razvoju testnih aplikacij; podani so še testni primeri aplikacij, kjer so vidne razlike v uporabniškem vmesniku in funkcionalnosti aplikacij.

Cilji zaključne naloge so: opis izdelave aplikacije v Oracle Foms tehnologiji, predstavitev izdelave aplikacije v ADF tehnologiji, prikaz integracije obeh razvojnih okolij in prikaz selitve z uporabo specifičnih orodij.

2. PREDSTAVITEV SISTEMOV

Zaključna naloga temelji na dveh programskih okoljih, ki sta namenjena razvoju aplikacij za poslovne namene. Za lažje razumevanje vsebine zaključne naloge sta v tem poglavju na kratko predstavljena oba razvojna okolja ter razvojna orodja, ki so uporabljena v nadaljevanju za razvoj aplikacij.

2.1. Oracle Forms

Programsko okolje za razvoj aplikacij Oracle Forms temelji na visokonivojskem programskem jeziku PL/SQL, tj. proceduralnem jeziku, ki je prilagojen za delo s podatkovnimi bazami in uporabniškimi vmesniki.

Razvojno okolje Oracle Forms je bilo sprva razvito za vodenje sej na strežniških terminalih, nato je bilo preneseno na druge platforme – tako na operacijski sistem Linux kot tudi na operacijski sistem Microsoft Windows[4]. Te dve platformi sta zamenjali starejše strežniške terminale in trenutno služita kot osnovni platformi za implementacijo odjemalnega okolja aplikacije. Najnovejše verzije Oracle Forms delujejo v Java okolju ter omogočajo integracijo s spletnimi storitvami.

2.1.1. Razvoj uporabniških vmesnikov

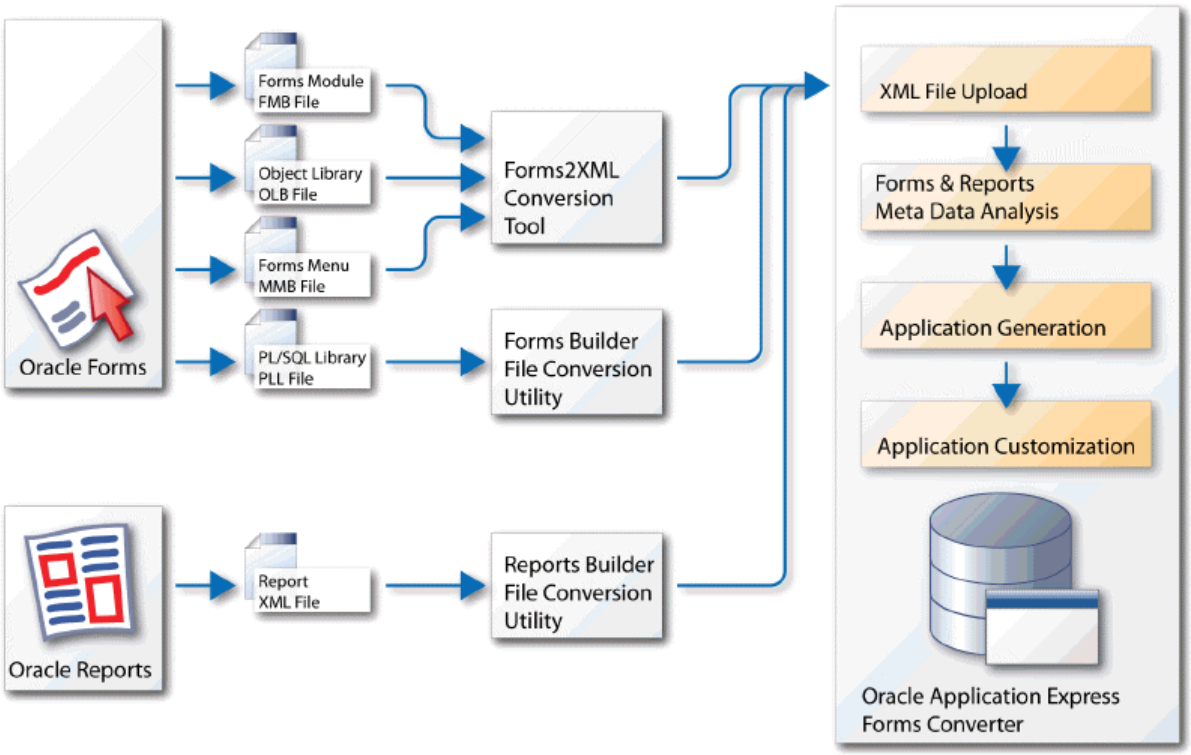
Osnovni namen Oracle Forms je razvoj uporabniških vmesnikov za vnos podatkov in povpraševanje po podatkih, ki so shranjeni v podatkovnih bazah sistema Oracle. Uporabniški vmesniki so definirani na osnovi vnosnih polj, seznamov izbire, tabel za prikaz podrobnosti in gumbov. S temi gradniki je predstavljena konfiguracija uporabniškega vmesnika.

Konfiguracija uporabniškega vmesnika definira interakcijo med podatki, ki so shranjeni v podatkovni bazi, in gradniki uporabniškega vmesnika, kot so na primer grafični elementi: gumbi, meniji in drsniki[4]. Konfiguracija se shrani v datoteko s končnico ».fmb«, katero na namensko postavljenem strežniškem sistemu prevedemo v izvajalno datoteko s končnico

».fmx«. Izvajalne datoteke lahko prikažemo in uporabljamo v izvajalnih modulih razvojnega sistema Oracle Forms.

Razvojno okolje vsebuje množico operacij za ustvarjanje novih zapisov, poizvedovanje ter posodabljanje podatkovnih modelov. Integrirana orodja Oracle Forms omogočajo hitrejše programiranje ponavljajočih se operacij, kot je na primer dinamično ustvarjanje SQL stavkov in zagotavljanje integritetnih omejitev z zaznavanjem spremenjenih vrstic ter preprečitvijo spreminjanja podatkovnih vrstic v primeru napak. Omogočeno je upravljanje kritičnih področij z uporabo prožilcev, ki se samodejno prožijo ob procesiranju vnosov. Vsako kritično področje lahko nadzorujemo z različnimi prožilci, ki se sprožijo pred, med ali za samim ukazom oziroma funkcijo.

Pri programiranju v razvojnem okolju Oracle Forms običajno spreminjamo funkcionalnost prožilcev in s tem nadzorujemo izvajanje aplikacije[4]. Prožilci začetno vsebujejo privzeto kodo, ki jo je potrebno prilagoditi dani aplikaciji. Takšen pristop omogoča izdelovanje obrazcev uporabniškega vmesnika, ki realizirajo potrebno funkcionalnost aplikacije.



Slika 1: Prevajanje Oracle Forms aplikacije [14]

2.1.2. Oracle Forms Developer

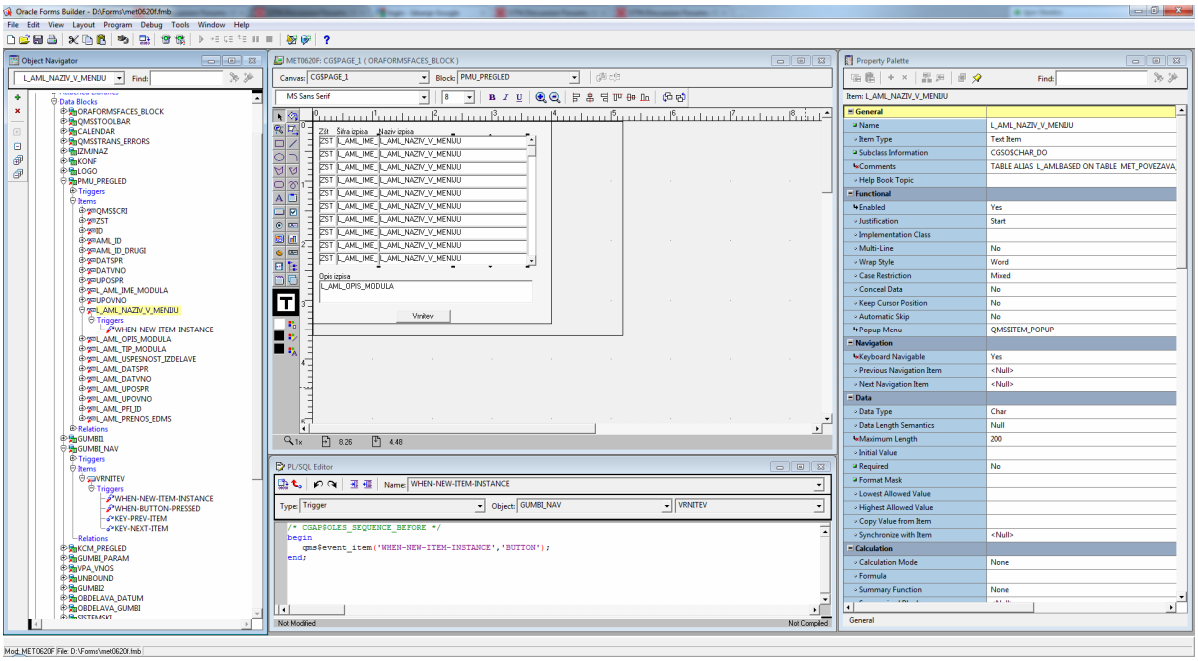
Oracle Forms Developer je razvojno okolje za hiter razvoj aplikacij, ki dostopajo do podatkovne baze. Razvijalci lahko uporabljajo orodje za hitro izdelavo kompleksnih javanskih aplikacij, za ogled, spreminjanje in dodajanje podatkov v podatkovno bazo, brez pisanja Java programske kode.

Forms Developer vsebuje orodja, ki omogočajo razvijalcem poslovnih procesov definicijo načrta podatkovne baze in poslovne logike. Razvojno okolje nudi zmogljive deklarativne funkcije, kot so čarovniki za razne komponente ter metode povleci-in-spusti (angl. *drag-and-drop*) za oblikovanje popolnoma funkcionalnih aplikacij, zaradi česar se izognemo pisanju kode same komponente.

Oracle Forms Service omogoča uporabniku Oracle Developer uporabo aplikacijske infrastrukture za delovanje aplikacij v poljubnem omrežju z uporabo spletnega (Web) strežnika, kar dovoljuje delovanje odjemalca v različnih sistemskih okoljih. Storitve omogočajo upravljanje transakcij, začasno shranjevanje zapisov v predpomnilnik, zaklepanje zapisov in rokovanje izjem – vse to olajša razvijalcu delo, saj se ta izogne prekomernemu pisanju ponavljajoče se kode.

Oracle Forms Developer in Oracle Forms Services zagotavljata aplikacijsko ogrodje za učinkovito postavitev Oracle Forms aplikacije na Internet oziroma Intranet in tvorita razvojno okolje za hiter razvoj aplikacije ter aplikacijsko infrastrukturo za delovanje same aplikacije.

Aplikacijsko ogrodje je odprto in razširljivo, kar omogoča: enostavno nadgradnjo aplikacije na novejše verzije, integracijo z Javo, razširitev uporabniškega vmesnika z Java komponentami in XML tehnologijo.

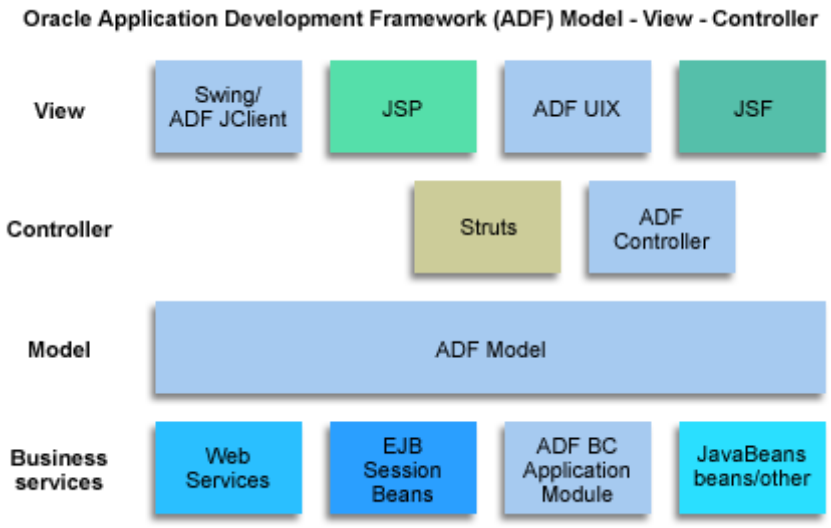


Slika 2: Orodje Oracle Forms Builder

2.2. Oracle ADF

Oracle ADF (ang. Aplication Development Framework) je programsko razvojno ogrodje, zgrajeno na osnovi Java Enterprise platforme, ki omogoča razvijalcem visoko produktivnost pri izdelovanju programskih aplikacij. Programsko razvojno ogrodje omogoča izdelavo in posodobitve različnih slojev aplikacij, kot so na primer aplikacijska logika, uporabniški vmesnik ali aplikacijski moduli.

Arhitektura Oracle ADF je sestavljena iz štirih slojev: poslovnih storitev, modela, krmilnika in vmesnika[13]. Sloj poslovnih storitev zagotavlja dostop do podatkov iz različnih virov ter za rokovanje s poslovno logiko. Naloga modela je zagotoviti abstrakten sloj nad slojem za poslovne storitve, kar omogoča konsistentno delovanje tako krmilnemu sloju kot sloju vmesnika. Krmilni sloj poskrbi za kontrolo toka podatkov in ukazov spletne aplikacije. V sloju vmesnika pa je zgrajen uporabniški vmesnik za dostop do aplikacije.



Slika 3: Arhitektura Oracle ADF [9]

2.2.1. Razvoj spletnih uporabniških vmesnikov

Oracle ADF vključuje izbor več kot 150 komponent, ki temeljijo na JSF standardih in podpirajo funkcionalnost Ajax[3]. Te komponente omogočajo razvoj spletnih uporabniških vmesnikov z visokim nivojem funkcionalnosti in interaktivnosti ter omogočajo vnos in vizualizacijo podatkov z uporabniku enostavnimi koraki, ki zagotavljajo učinkovit razvoj aplikacij.

Oracle ADF razširi osnovni JSF krmilnik tako, da ustvari ADF krmilnik. JSF krmilnik je lahko katerakoli POJO metoda, ki vrača določen niz in se uporablja za nadzorovanje krmarjenja ter akcij med spletnimi stranmi[1]. ADF krmilnik izboljša nadzor nad delovanjem strani in potekom izvajanja operacij ter omogoča ponovno uporabnost komponent v drugih razdelkih in straneh.

ADF ponuja ogrodje, ki poenostavlja povezovanje uporabniškega vmesnika s poslovno logiko preko uporabe preproste operacije povleci-in-spusti. S tem ogrodjem je razvijalec uporabniškega vmesnika izoliran od osnovnih gradnikov, ki delujejo v sloju poslovnih storitev[6]. Tak pristop izdelave uporabniškega vmesnika posledično omogoča povečano učinkovitost izvajane aplikacije v storitveno-orientirani arhitekturi.

ADF ogrodje podpira neposredno implementacijo spletnih vmesnikov, mobilno dostopnih aplikacij in namiznih aplikacij, vključno z integracijo s programom Microsoft Excel-om[3]. Razvijanje je poenostavljeno z uporabo primerne komponente, prikrojene izbrani metodi. V mnogih primerih se lahko komponenta, ki je bila uporabljena za implementacijo določene metode, brez sprememb ponovno uporabi za druge.

Varnost je pri Oracle ADF zagotovljena na osnovi deklarativno definiranih pravic in dovoljenj[6][17]. Za dosego različnih varnostnih zahtev se lahko varnostne kontrole implementira na različnih nivojih aplikacije. Prav tako je mogoče uporabiti že obstoječe varnostne nastavitve, kot na primer nastavitve za dostop do Microsoftovega aktivnega imenika.

Oracle ADF aplikacijo je mogoče prilagoditi potrebam uporabnika z dodajanjem slojev sprememb, ne da bi pri tem spreminjali izvorne kode[16]. Prilagoditev je možno zasnovati neodvisno, in sicer za vsak posamezen sloj ogrodja.

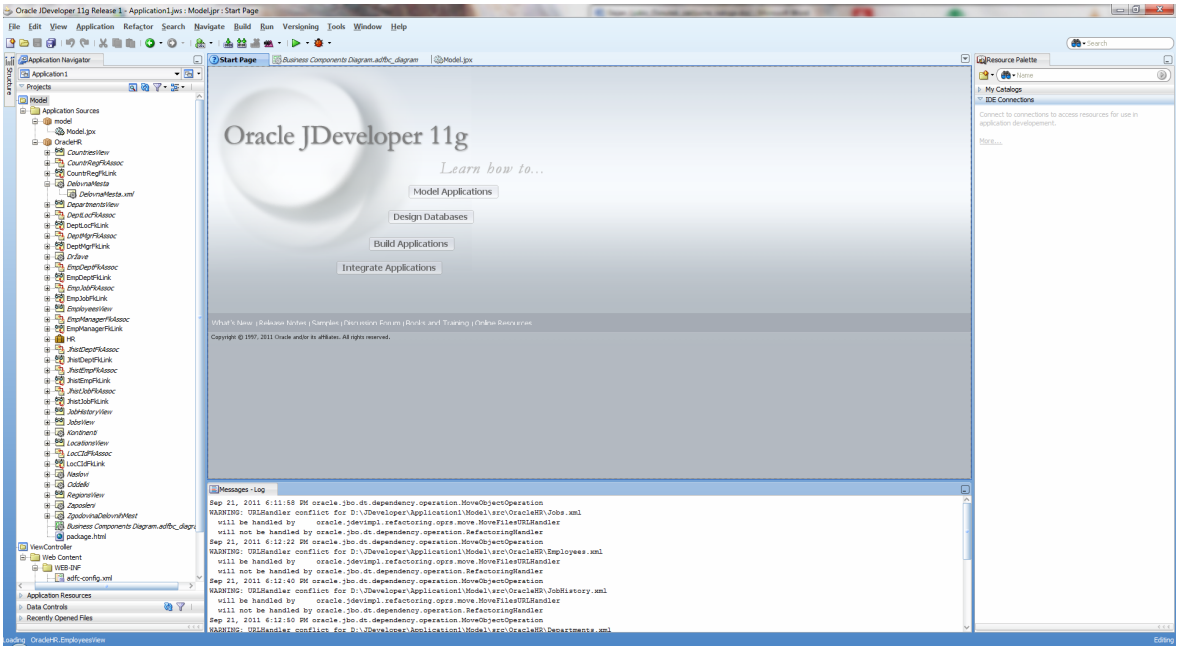
2.2.2. Oracle JDeveloper

Oracle JDeveloper je integrirano razvojno okolje (IDE) podjetja Oracle za izgradnjo aplikacij in spletnih storitev z uporabo najnovejših standardov jezikov Java, XML in SQL. JDeveloper podpira razvoj v vseh fazah življenjskega cikla aplikacije in obsega funkcije za modeliranje, kodiranje, razhroščevanje, testiranje profilov, nastavitev in namestitev aplikacije[2]. JDeveloper omogoča tudi popoln razvoj in modeliranje okolja za izgradnjo podatkovne baze.

Razvoj ADF aplikacij z Oracle JDeveloper omogoča razvijalcu oblikovanje aplikacij z uporabo vizualnih urejevalnikov in diagramov. Lastnosti razvitih komponent se lahko prilagodijo z orodjem za urejanje lastnosti (angl. *Property inspector*). Prav tako omogoča razvijalcu neposredno manipulacijo izvorne kode na kateri koli točki, kar ponuja možnost preklapljanja med različnimi slogi razvijanja, in sicer za lažjo izbiro primerne aplikacije ali izražanja razvijalčevih osebnih preferenc[1].

JDeveloper ponuja popolne rešitve pri popravljanju napak in odpravljanju hroščev, saj omogoča določitev prekinitvene točke znotraj različnih jezikov in ogrodij. Navadno se pri

razvoju aplikacij z JDeveloper uporabljajo sistemi Java, JSP, PL/SQL in ADF. Možno je tudi izvajanje posameznih delov aplikacije.



Slika 4: Orodje Oracle JDeveloper

3. RAZLOGI IN DEJAVNIKI ZA SELITEV APLIKACIJE

V poslovnem svetu je potrebno analizirati vsako spremembo poslovnega procesa, dejavnosti, tehnologije oziroma arhitekturne strukture sistema. Končne odločitve morajo biti temeljito argumentirane in preučene s tržnimi raziskavami ter potrebnimi analizami, v nasprotnem primeru je vsaka izbira lahko napačna, kar lahko, dolgoročno gledano, vodi v napačne smeri.

Pri odločanju o selitvi aplikacije je potrebna preučitev razlogov in dejavnikov, ki pripomorejo k izbiri končne odločitve. V nadaljevanju so opisani nekateri najpomembnejši razlogi in dejavniki, ki lahko vplivajo na samo odločitev.

3.1. Razlogi za selitev aplikacije

Vse več podjetji se danes odloča za uporabo spletnih, uporabniku prijaznejših aplikacij, ki hkrati nudijo tako modernejšo podobo ter razne grafične učinke kakor tudi širok nabor funkcijskih možnosti. Odločitev za tako spremembo zahteva precej sestankov, posvetov in raziskav, saj je nujno potrebno pri tem natančno analizirati vse možnosti, ki nam jih ponuja današnja tehnologija ter seveda upoštevati finančno oziroma stroškovno plat, ki predstavlja v končni fazi pomemben mejnik med uspešnimi in manj uspešnimi podjetji.

Razloge za selitev aplikacije lahko razdelimo na tri sklope, to so:

- razlogi za selitev obstoječega sistema;
- razlogi proti selitvi obstoječega sistema;
- napačni razlogi pri odločanju o selitvi sistema.

3.1.1. Razlogi za selitev obstoječega sistema

Po desetletju in več razvoja, nadgrajevanja in uporabe aplikacije v tehnologiji Oracle Forms, le-ta ne izpolnjuje več pričakovanih zahtev[5]. Pomanjkljivosti se kmalu opazijo, ko se podjetje odloči za širitev poslovanja v svet spletne in mobilne tehnologije, kjer pa Oracle Forms niso prava izbira, saj novejše tehnologije zahtevajo drugačne pristope, ki so prilagojene

različnim napravam, na katerih je omogočeno izvajanje sodobnih spletnih aplikacij. Desetletje je za aplikacijo precej dolgo obdobje, saj se po tem času le-ta bliža koncu svoje življenjske dobe oziroma izgublja na konkurenčnosti, ki jo je obdržala med vodilnimi na trgu. V tem primeru je razmišljanje o prenosu ali prenovi dela funkcionalnosti oziroma celotne aplikacije na sodobnejšo tehnologijo dolgoročno gledano smotrna odločitev.

Med razloge, ki podpirajo selitev aplikacije iz delujočega obstoječega sistema v novega, spada tudi želja vodstva podjetja po širitvi poslovanja v smeri, ki jo prinaša Oracle ADF[5]. Razlog je upravičen, spodbuja inovativnost in konkurenčnost na trgu informacijskih storitev ter ponuja podjetju razvojno smer, ki temelji na novi tehnologiji.

Nenazadnje je možen razlog za selitev tudi zahteva kupcev oziroma končnih uporabnikov po zmogljivejši in sodobnejši aplikaciji, saj pri prodaji moramo upoštevati, da je najboljša stranka zadovoljna stranka. Uspešno podjetje pa si ne sme privoščiti izgube končnih uporabnikov zaradi neizpolnjevanja njihovih želja in zahtev, saj na tak način izgubi tudi na pridobljeni kredibilnosti.

3.1.2. Razlogi proti selitvi obstoječega sistema

Pri sprejemanju odločitev se, poleg razlogov *za*, pojavljajo prav tako pomembni, če ne celo pomembnejši, razlogi *proti*: ti lahko namreč preprečijo prehod na pot, ki vodi v razvoj napačnih tehnologij in dolgoročno gledano tudi v izgubo konkurenčnosti samega podjetja.

Prehod na novo aplikacijo ne sme biti drastičen, saj se morajo razvijalci najprej privaditi na novo programsko okolje. Še posebno v primeru večletnega dela na specifični aplikaciji je smiseln postopen prehod na novo aplikacijo z njenim razvojem po delih. V kolikor bi bilo počasno uvajanje posameznih skupin stroškovno preobsežno, je odločitev o selitvi neprimerna. Čas in trud razvijalcev, ki sta bila vložena v razvoj obstoječe aplikacije in s tem povezane potrebne investicije so dober razlog za nadaljevanje z uporabo obstoječe tehnologije.

Dobra razloga proti selitvi sta tudi zadovoljstvo končnih uporabnikov in izpolnjevanje zahtev samega poslovnega procesa[5]. Za določitev prvega je običajno potrebna raziskava, saj se končni uporabniki po večletnem delu z aplikacijo nanjo navadijo in so zadovoljni z njenim delovanjem. Tudi v tem primeru lahko upoštevamo pravilo, da je najboljša stranka zadovoljna stranka in zato selitve ne izvedemo.

Če obstoječa aplikacija izpolnjuje zahteve poslovnega procesa in ga podjetje ne namerava spreminjati, je selitev programskega okolja neprimerna[5]. Tudi v primeru, ko podjetje ne namerava v doslednem času uporabljati in niti ne uvajati mobilne in internetne tehnologije, je selitev nesmiselna.

Vse tri zgoraj navedene razloge bi lahko zaključili s skupno mislijo, da »delujoče aplikacije ni potrebno zamenjati,« saj takšna odločitev zahteva od podjetja velike stroške in predstavlja prehod na popolnoma novo in nepoznano tehnologijo.

3.1.3. Napačni razlogi pri odločanju o selitvi sistema

Poleg razlogov *proti* selitvi aplikacije je potrebno omeniti tudi *napačne* razloge, ki so praviloma zgolj posledica napačne interpretacije oziroma razmišljanja v procesu sprejemanja odločitev.

Prvi napačen razlog predstavlja dosledno sledenje razvoju novih tehnologij, ki jih razvija proizvajalec razvojnih orodij – v našem primeru Oracle. Proizvajalec namreč običajno podpira razvoj in vzdrževanje starega razvojnega okolja, v našem primeru Oracle Forms, vsaj dokler obstaja za to interes uporabnikov[5].

Z uvedbo novih tehnologij Oracle ne želi siliti v selitev na nove sisteme, temveč zgolj predstaviti možnosti, ki jih ponujajo v primeru, da obstoječa aplikacija ne zadošča zahtevam. Končna odločitev je zato odvisna od posameznikovih želja in potreb.

Med napačne razloge za selitev aplikacije spada tudi izogibanje nadgradnji obstoječih razvojnih sistemov, kot je na primer prehod iz Oracle Forms 6i na Oracle Forms 10g[5]. Tudi če je pri namestitvi oziroma konfiguraciji aplikacije na novem razvojnem sistemu prišlo do

težav, je običajno smiselno izvesti prehod na novo verzijo razvojnega sistema do konca in šele potem izvesti študijo o smiselnosti selitve aplikacije na nov razvojni sistem.

Nazadnje je potrebno opozoriti še na napačno razmišljanje v zvezi s finančnimi razlogi. Programski jezik Java, ki je osnova Oracle ADF, je izredno razširjen, kar precej olajša pridobivanje kadra za razvoj aplikacije. Čeprav je Java programski jezik zelo razširjen in uporabljen v spletnih tehnologijah, je tudi v primeru rabe Oracle ADF potrebno vložiti precej časa v obvladovanje novih tehnologij. Enako velja tudi v primeru razvoja novih funkcionalnosti te aplikacije.

3.2. Dejavniki za selitev aplikacije

Selitev obstoječe Oracle Forms aplikacije v Oracle ADF aplikacijo razumemo kot proces, pri katerem obstoječo aplikacijo preoblikujemo v novo z uporabo novega razvojnega okolja Oracle ADF. Pri takšni selitvi poskušamo izkoristiti čim večji delež že napisane aplikacije. Uspešna selitev aplikacije na novo razvijalno okolje mora upoštevati naslednje tri dejavnike.

1) Temeljito razumevanje na Oracle Forms razvite aplikacije, ki mora biti v vsej svoji kompleksnosti ponovno implementirana v novem okolju[12].

Aplikacija se pogosto iz majhne in enostavne razvije v večjo in precej zahtevnejšo aplikacijo. V času samega razvoja pride velikokrat do zamenjave razvijalcev in do pomanjkljivo napisane dokumentacije, kar lahko novim razvijalcem oteži iskanje odgovorov na ključna vprašanja, ki zadevajo delovanje določenih sklopov aplikacije in zato zahteva dodatno usposabljanje na Oracle Forms arhitekturi.

2) Dobro poznavanje novega razvojnega okolja, kar omogoča korekten prenos vseh funkcionalnosti stare aplikacije v novo arhitekturo in pogosto nudi tudi izboljšanje obstoječih rešitev[12].

Selitev aplikacij iz Oracle Forms v Oracle ADF je zahtevna naloga, saj se ti dve razvojni okolji bistveno razlikujeta. Spremembe se velikokrat obnesejo le, če nova različica aplikacije

ponuja boljše uporabniške možnosti in višje tehnološke zmogljivosti ter modernejši videz. Eden izmed ciljev selitve aplikacije v novo razvojno okolje je namreč tudi njeno izboljšanje.

3) Vzpostavitev procesov projektnega vodenja za nadzorovanje napredka v času razvoja in selitve[12].

Razlogi za neuspeh projektov, katerih naloga je ravno selitev obstoječe aplikacije v novo razvojno okolje, so pogosto posledica nepredvidenih izzivov, pomanjkanja izkušenj in neupoštevanje standardov. Če želimo zagotoviti uspešnost projekta, mora projektno vodenje usmeriti večjo pozornost na spremljanje razvoja: potrebno je identificirati vse potencialne probleme z novo tehnologijo ter standardi in to še preden lahko pride do napačne zasnove nove aplikacije.

4. PRISTOPI SELITVE APLIKACIJE

Razvoj sodobnih aplikaciji je obsežen proces pri katerem običajno sodeluje večje število razvijalcev, svetovalcev in nadzornikov. Vsak projekt mora imeti svojo funkcijsko specifikacijo, ki opisuje naročnikove zahteve ter poslovno logiko aplikacije, katero so dolžni pripraviti sami naročniki. Prav tako pa mora podjetje pri razvoju aplikacije upoštevati standarde za pisanje kode in dokumentacije, ki so potrebni za kasnejše popravljanje, posodabljanje ali samo razumevanje njenega delovanja. Zaradi kasnejšega razumevanja je potrebna natančna dokumentacija, ki vsebuje opis rešitve problemov in uporabljenih pristopov. Prav tako obstaja tudi več načinov oziroma pristopov za selitev aplikacije skupaj z orodji, ki olajšajo omenjeno selitev.

4.1. Pristop ponovne izdelave aplikacije

Ponovno izdelavo aplikacije lahko razumemo kot izdelavo popolnoma nove aplikacije, ki bo imela končno funkcionalnost stare, ne da bi pri tem prišlo do uporabe orodji, ki omogočajo avtomatizirano selitev aplikacij. V nadaljevanju so predstavljeni dejavniki za selitev aplikacije, ki jih lahko razdelimo na tri sklope[12]:

- spretnosti in znanje razvijalcev,
- uporabniški vmesnik,
- poslovna logika.

4.1.1. Spretnosti in znanje razvijalcev

Za uspeh projekta je bistvenega pomena strokovno znanje na področju PL/SQL, Oracle Forms, Java, JavaScript, XML in J2EE. Razvijalce z znanjem na področju Java tehnologij danes enostavno najdemo na trgu, saj je Java programski jezik eden osnovnih predmetov izobraževalnih ustanov na področju računalništva. Ključno je torej znanje oziroma razumevanje kompleksnosti Oracle Forms in PL/SQL, zmogljivosti nove tehnologije ter programske arhitekture v okviru Oracle ADF. Več pozornosti je potrebno torej nameniti

sestavljanju delovne ekipe razvijalcev, ki se morajo spoznati na obe področji. Prav tako je zelo pomembno, da se vodja ekipe spozna na tehnologije, ki bodo uporabljene za razvoj projekta oziroma je vsaj seznanjen funkcionalnostmi, ki jih le-te omogočajo.

4.1.2. Uporabniški vmesnik

Čeprav sodobna spletna tehnologija omogoča večjo dostopnost do storitve, je pri modeliranju novega uporabniškega vmesnika potrebno upoštevati omejitve spletnih brskalnikov. Aplikacija v Oracle Forms ponuja zelo učinkovite podatkovne arhitekture, omogoča poizvedbo in prikaz na tisoče podatkovnih zapisov naenkrat, na stotine polj na eni strani, dostop do računalnika odjemalca (angl. *client computer*) in opravljanje gostiteljskih ukazov (angl. *host command*) ali dostop do datotek. Takšno aplikacijo je za poganjanje v spletih tehnologijah potrebno preoblikovati tako, da bodo prikazani sklopi z dvajsetimi zapisi in da bodo polja, ki so bili prej prikazana na eni strani, razporejena na več straneh. Pri tem je potrebno zagotoviti tudi majhno obremenitev računalnika odjemalca. Če Forms aplikacije ni mogoče preoblikovati na tak način, predstavlja selitev v spletne tehnologije preveliko tveganje. Dokler nam spletne tehnologije ne omogočajo primernejše nadomestne rešitve, je v takem primeru uporaba zadnje različice Oracle Forms najboljša možna rešitev.

4.1.3. Poslovna logika

Oracle Forms aplikacija vsebuje v isti enoti, poleg same poslovne logike, tudi akcije uporabniškega vmesnika, kot so klik z miško na gumb, preverjanje veljavnosti podatkov, manipulacijo podatkov, prožilce, itd. Pri selitvi aplikacije v Oracle ADF je potrebno poslovno logiko ločiti od uporabniškega vmesnika ter ostalih komponent[12]. Možni pristopi za to so:

- Prevesti celotno kodo v Javo, kar zahteva ročno pisanje celotne kode. V tem primeru se izgubi celotna investicija vložena v razvoj poslovne logike v Oracle Forms. Na trgu obstajajo orodja, ki omogočajo samodejno pretvorbo Oracle Forms kode v Java kodo. Rezultat uporabe teh orodij je sicer zaporedna in postopkovna koda (kot je na primer PL/SQL), ki ni objektno usmerjena, kot naj bi bila v Javi, poleg tega pa vsebuje tudi mešanico podatkov za ravnanje s poslovno logiko in uporabniškim vmesnikom, kot je

značilno za Forms ne pa za ADF. Tako kodo je težko razumeti, ker ne izpolnjuje Java standardov in posledično otežuje samo vzdrževanje aplikacije.

Tak pristop ni najprimernejši, saj gre za aplikacije, pri katerih je značilna neposredna interakcija s podatkovno bazo in je zato primernejše, da je večina poslovne logike, če že ne celotna, v bazi in ne v sami aplikaciji.

- Kodo napisano v PL/SQL shraniti v podatkovno bazo. S tem se dostop do podatkov in manipulacija le-teh hitreje izvaja, zaradi manjšega pretoka podatkov pa se zmanjša tudi obremenitev omrežja. PL/SQL že leta izpolnjuje zahteve poslovne logike ne glede na to, ali se nahaja v Oracle Forms ali v podatkovni bazi. Koda uporabniškega vmesnika, kot na primer preverjanje pravilnosti podatkov v vnosnih poljih, ni namenjena uporabi v podatkovni bazi, to pa narekuje, da tudi tak pristop ni najprimernejši.

- Tretji in najprimernejši pristop je združitev zgoraj navedenih pristopov. Koda uporabniškega vmesnika se napiše v Javi, ki bo z uporabo JavaScript in Ajax tehnologije pridobila na hitrosti ter funkcionalnosti, na drugi strani pa se PL/SQL kodo za poizvedovanje in obdelavo samih podatkov shrani v podatkovno bazo[11].

Za določena trivialna preverjanja pravilnosti podatkov, kot so preverjanje vnesenega datumskega zapisa, preverjanje ali polje za vnos števil vsebuje samo numerične vrednosti ter preverjanje pravilnosti dolžine zahtevanega števila (npr. v primeru davčne številke in EMŠO številke), lahko postavimo kontrolne mehanizme v uporabniški vmesnik z uporabo JavaScrpt tehnologije. Na ta način lahko sproti preverjamo in opozarjamo na prisotnost napak pri določenem vnesenem polju.

4.2. Pristop združitve obeh sistemov

Z združitvijo obeh sistemov mislimo na integracijo obstoječe Oracle Froms aplikacije in aplikacije, ki je narejena v Oracle ADF, kar nam omogoča knjižnica OraFormsFaces. To knjižnico je mogoče namestiti kot dodatek orodja Jdeveloper, omogoča nam dodajanje komponent na JSF stran z uporabo metode povleci-in-spusti in z nastavljanjem specifičnih

lastnosti teh komponent. Razvijalcem je tako omogočena integracija Oracle Forms v samo JSF stran, kar zajeme pošiljanje vsebine, dogodkov in parametrov v obe smeri, zmanjšanje časa zagona ter še druge tako vizualne kot funkcionalne možnosti.

Knjižnica OraFormsFaces je ključna tehnologija pri dopolnitvi Oraclove ponudbe za selitev aplikacij na Oracle ADF. Sama tehnologija Oracle Forms namreč ni popolnoma združljiva z Javo in ne omogoča, da bi končni produkt združitve bila ena sama delujoča aplikacija[7].

Uporaba knjižnice OraFormaFaces omogoča postopno selitev aplikacije, napisane v Oracle Forms, v Oracle ADF spletno aplikacijo. Poleg možnosti prenosa parametrov iz ADF spletne aplikacije v Oracle Forms in obratno, imata oba sistema to prednost, da podpirata proženje dogodkov. Integracija Oracle Forms v ADF spletne aplikacije prinaša tudi splošno možnost integracije v nekatere druge spletne aplikacije, torej ne samo v ADF ampak tudi v PHP, ASP.Net, in Oracle WebCenter[7].

Predhodne različice Oracle Forms niso podpirale avtohtonega programskega vmesnika JavaScript. Ta funkcija se uvaja šele z Oracle Forms 11g, vendar se lahko vseeno izvede z uporabo programskega vmesnika LiveConnect, ki se uporablja pri brskalnikih[7]. Ta programski vmesnik omogoča skriptnemu jeziku JavaScript klic metod Java razredov. V primeru, da program Oracle Forms izpostavlja javno metodo (privzeto je sicer ne, vendar jo lahko dodamo), je le-ta lahko poklicana v JavaScriptu.

Pri integraciji je pomembna tudi grafična podoba, saj je lahko prikaz menija aplikacije Oracle Forms znotraj ADF aplikacije moteč. Knjižnica OraFormsFaces tako omogoča prikritje menija, orodnih in statusnih vrstic ter prikaz samega razdelka, v katerem se bo stara aplikacija pognala v želeni resoluciji.

Ne glede na možnosto medsebojne interakcije pa je potrebno omeniti, da sta sistema Oracle Forms in Oracle ADF še vedno dva različna in med seboj ločena sistema, ki zahtevata ločeni povezavi do podatkovne baze, saj je tudi njun način izvajanja transakcij različen[8]. Za Oracle Forms so značilni bazni uporabniki, kar pomeni, da je ima vsak uporabnik aplikacije dostop do podatkovne baze z istim uporabniškim imenom in geslom, s katerim dostopa do same aplikacije. Ob prijavi v aplikacijo se v podatkovni bazi ustvari uporabniška seja, kjer lahko uporabnik uporablja začasne tabele in globalne spremenljivke, ki so vidne samo njemu.

Slabost takšnega načina prijavljanja je v tem, da je sočasno lahko na podatkovno bazo prijavljenih več uporabnikov, ki si lastijo po eno ali več sej (če je pognanih več aplikacij). Pri Oracle ADF so uporabniški računi shranjeni v samem aplikacijskem strežniku, ki z enim uporabniškem računom dostopa do podatkovne baze, na kateri odpre določeno število sej. S takim pristopom se število odprtih sej zmanjša, saj so dodeljene uporabnikom na zahtevo: ko uporabnik zahteva dostop do podatkovne baze, dobi prosto sejo, po uporabi pa se ta dodeli drugemu uporabniku oziroma se sprosti.

V tem pristopu globalne spremenljivke in začasne tabele ne pridejo v poštev, saj lahko eno sejo uporablja več uporabnikov, ki običajno hranijo različne vrednosti. To težavo lahko rešimo tako, da namesto začasnih uporabljamo navadne tabele, kjer pri vsakem zapisu hranimo identifikacijski podatek o uporabniku.

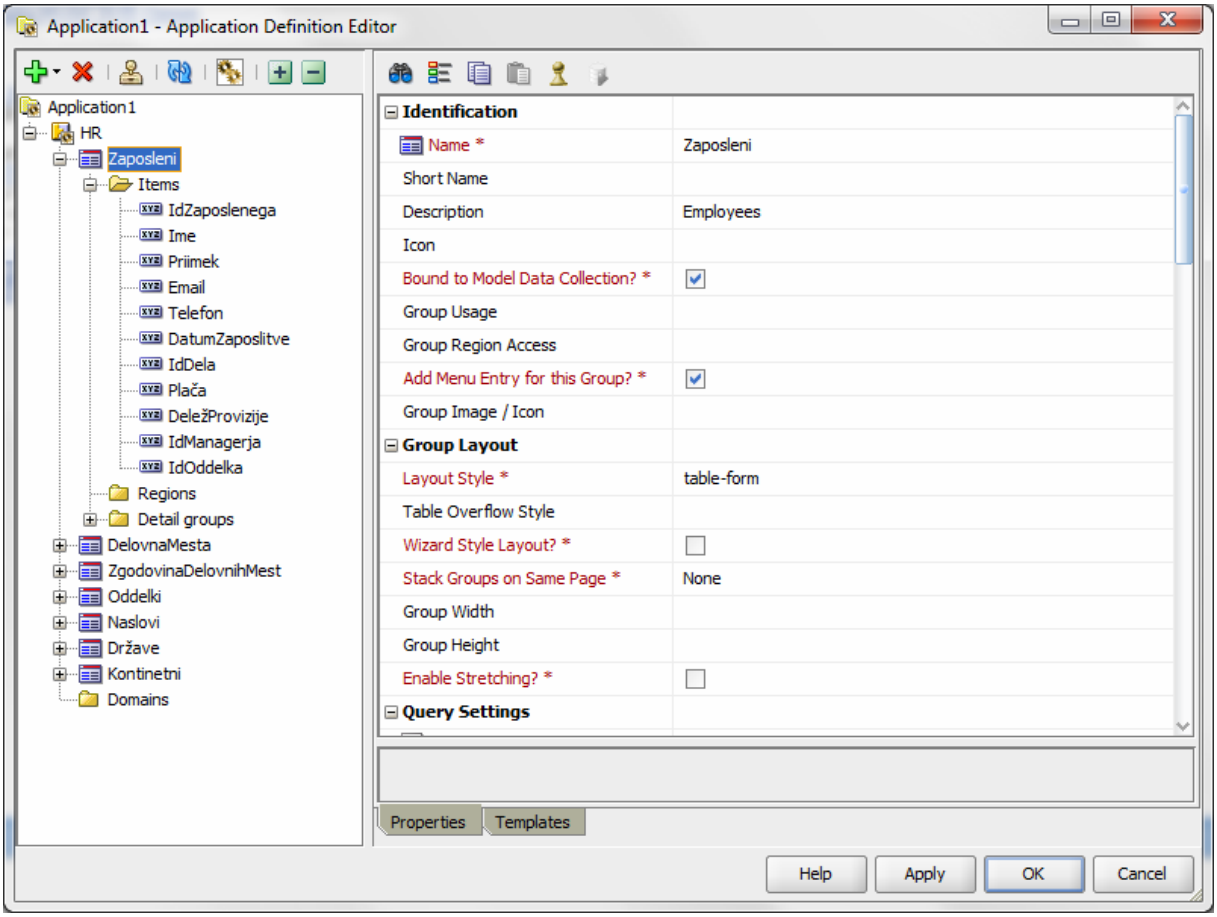
4.3. Pristop selitve z uporabo orodja JHeastart

Oracle JHeadstart 11g izboljša razvijalčevo produktivnost pri Oracle ADF, omogoča organizacijam hitro uvedbo Java EE (JEE), ne da bi hkrati potrebovali skupine JEE gurujev za gradnjo kompleksnih transakcijskih JEE aplikacij[10].

Z deklarativno specifikacijo aplikacij preko uporabe funkcionalnih meta podatkov, preprostih urejevalnikov lastnosti ter JHeadstart aplikacijskega generatorja, slednji lahko ustvari zelo kompleksne ADF aplikacije. Med ostalimi funkcijami vključuje tudi drevesa, sezname vrednosti, hiter in napreden iskalnik, čarovnike, izbirne bloke, hierarhično povezovanje, dinamične zavihke, pogojno odvisne elemente, specifično nivojsko varnost, ob zagonu definirane dinamične elemente, dinamično generirane menijske strukture in celostno večjezično podporo[18].

JHeadstart samodejno izvaja številna načela in pravila, ki so povezana z ADF funkcijami, kot na primer tokovi opravil, fragmenti strani, predloge strani, dinamična področja strani, strukturirani meniji in modelirani seznami vrednosti. JHeadstart je neopazno integriran z Oracle JDeveloper IDE in je osnovan na najboljših svetovnih primerih Oracle Consulting-a.

JHeadstart vključuje tudi Forms2ADF generator, ki omogoča neposredno transformacijo Oracle Forms aplikacij v ADF aplikacije, posledično pa na tak način ščiti naložbe pri premiku v JEE (Java Enterprise Edition) razvojno platformo ter OraFormsFaces generator[18]. Slednji omogoča integracijo obstoječih Oracle Forms oken v ADF aplikacijo z možnostjo prenosa parametrov, ki so potrebni za inicializacijo aplikacije oziroma dela aplikacije.



Slika 5: JHeadstart aplikacijski urejevalnik

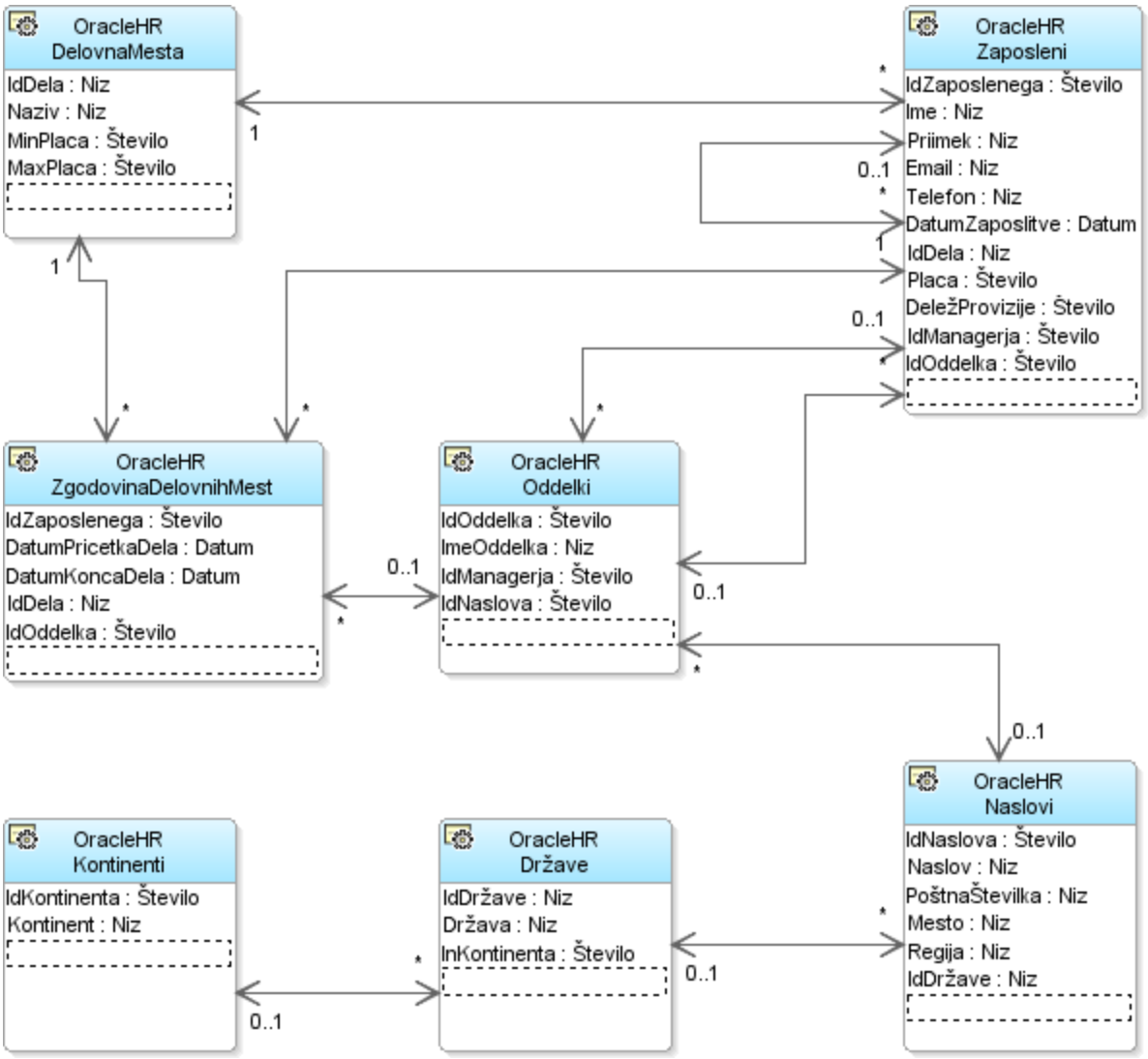
5. PRIMERJAVA APLIKACIJ

Pričujoče poglavje predstavlja drugi, praktični del zaključne naloge, v katerem bo sprva predstavljena podatkovna baza Oracle Database XE, na osnovi katere bodo zgrajene aplikacije, za boljšo predstavitev povezanosti in odvisnosti podatkov v podatkovnih tabelah. Predstavitvi podatkovne baze sledi predstavitev razvoja aplikacije v Oracle Forms tehnologiji. Podrobneje bo predstavljena izdelava .fmb datotek, ki se nato na strežniku prevedejo v izvršilne .fmx datoteke. Sledita še predstavitev razvoja aplikacije z uporabo Oracle ADF tehnologije in primer integracije predhodne Oracle Forms aplikacije vanjo.

5.1. Podatkovna baza

Oracle Database XE je Oraclova podatkovna baza, namenjena razvijalcem oziroma institucijam za izobraževalne namene in razvoj manj zahtevnih aplikacij[15]. Podatkovno bazo je mogoče namestiti na operacijske sisteme Windows in Linux ne glede na strojno opremo samega računalnika, saj koristi do enajst gigabajtov diskovnega spomina, do en gigabajt RAM spomina ter eno procesorsko enoto tudi v primeru več procesorskega računalnika.

Z namestitvijo podatkovne baze pridobimo pet privzeto nastavljenih shem s podatki. V nadaljevanju opisane in predstavljene aplikacije se osredotočajo na preprosto shemo HR (angl. *Human Resources*), ki vsebuje sedem med seboj povezanih tabel (slika 6).



Slika 6: UML diagram podatkovne baze

V tabeli »Zaposleni« so shranjeni podatki o delavcih. Podatki o podatkovnem tipu posameznega atributa so prikazani v prilogi, in sicer v tabeli 1. Tabela »Zaposleni« je povezana s tabelami »Oddelki«, »Delovno mesto« in »Zgodovina delovnih mest« ter sama s sabo.

V tabeli »Delovna mesta« so shranjeni zapisi o vseh poklicih oziroma delovna mesta znotraj podjetja. Podatki o potrebnih atributih vsakega zapisa so razvidni v tabeli 2, ki je povezana s tabelama »Zaposleni in pretekla delovna mesta«.

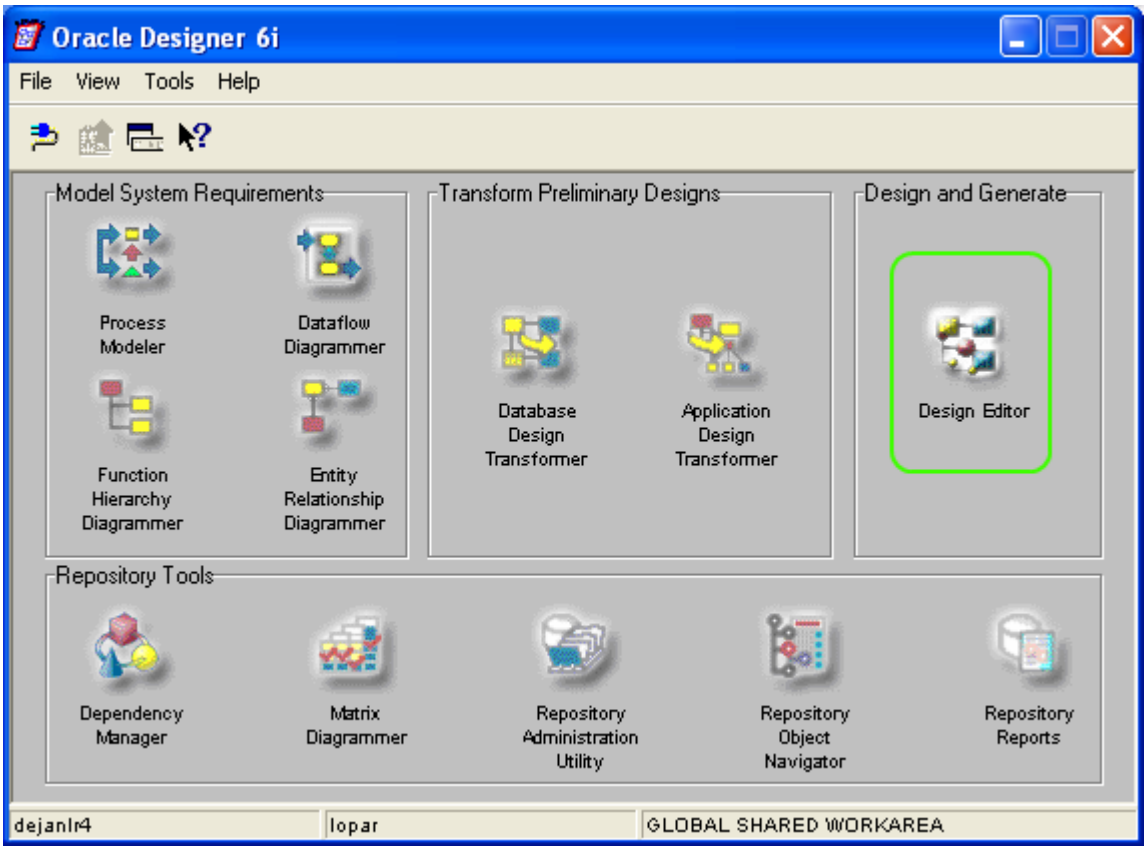
Tabela »Zgodovina delovnih mest« hrani zapise o tem, katera delovna mesta je v preteklosti zasedal določen zaposlen. Ti podatki so koristni pri izdelavi statistike napredovanja posameznega zaposlenega. Podrobnosti so prikazane v tabeli 3.

V podatkovni tabeli »Oddelki« lahko najdemo zapise, ki vsebujejo splošne podatke o samem oddelku, se pravi ime oddelka, vodjo ter naslov (v primeru, da ima podjetje več oddelkov v različnih stavbah). S to tabelo so povezane tabele »Zaposleni«, »Zgodovina delovnih mest« in »Naslovi«. Struktura tabele z vsemi atributi in podatkovnimi tipi je opisana v tabeli 4.

Preostale tri tabele »Kontinenti«, »Države« in »Naslovi« so enostavnejše, saj imajo, za razliko od ostalih zgoraj naštetih, manj relacij. Tabela »Kontinenti« vsebuje imena vseh kontinentov in je povezana s tabelo »Države«, v kateri so naštete pripadajoče države. Slednja je povezana še s tabelo »Naslovi«, v kateri so shranjeni splošni podatki o naslovih oddelkov. Strukture teh podatkovni tabel so opisane v tabelah 5, 6, 7.

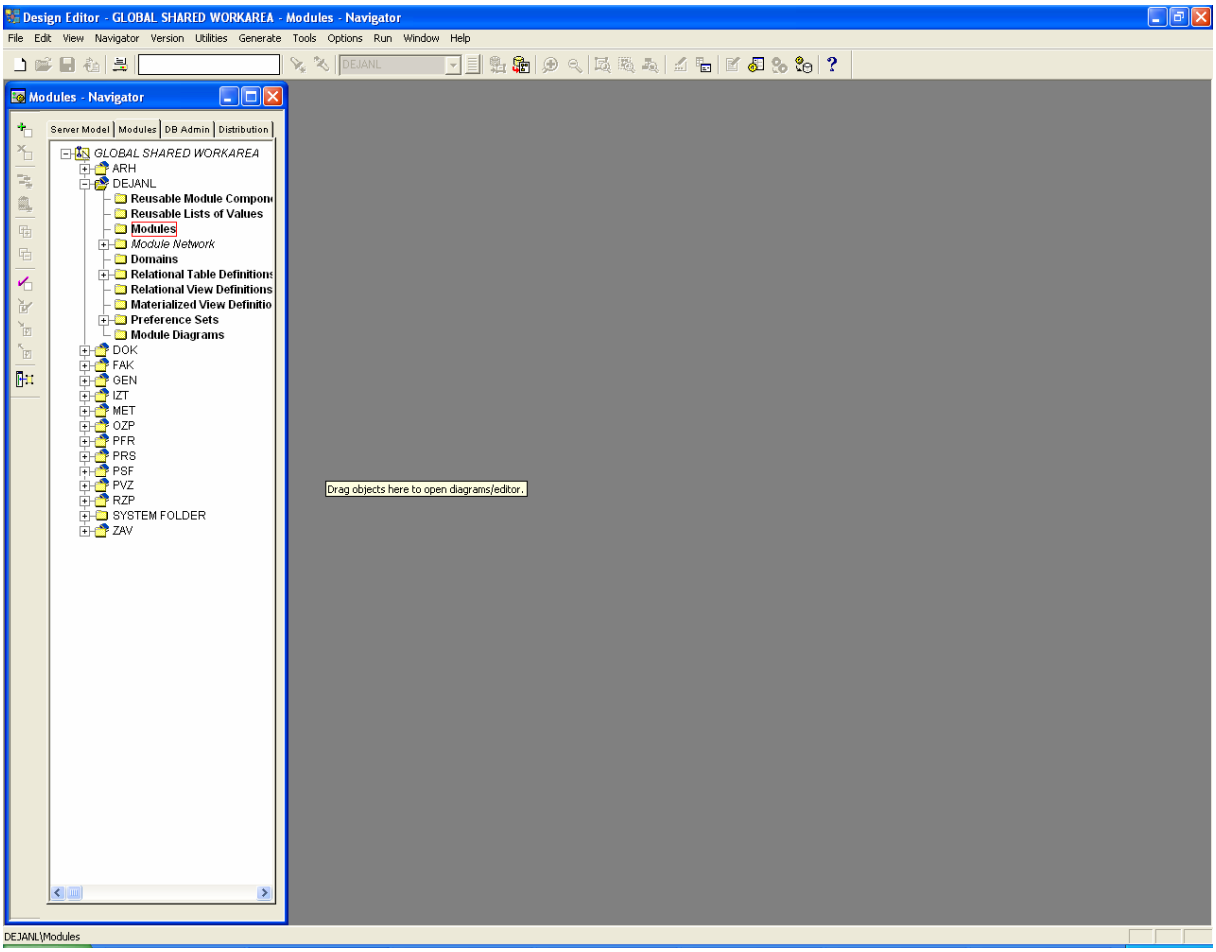
5.2. Oracle Forms aplikacija

Za izdelavo Oracle Forms aplikacij ponuja Oracle orodji Oracle Forms Builder in Oracle Designer. Slednje je kompleksnejše, vendar omogoča avtomatsko generiranje določenih delov kode v ozadju, kar programerju prihrani čas. V nadaljevanju bomo prikazali, kako lahko izdelamo Forms datoteke, ki omogočajo poizvedovanje po baznih tabelah. Natančneje bo prikazano, kako lahko ustvarimo osnovne tabele, ki prikazujejo bazne zapise, *master detail* tabele, liste vrednosti nad določenim poljem in vnosne obrazce z gumbi.



Slika 7: Zagonsko okno programa Oracle Designer

Za izdelavo form je bilo uporabljeno orodje Oracle Designer 6i, ki je povezano s podatkovno bazo sheme, o kateri smo pisali v prejšnjem poglavju. Razvoj aplikacije bomo prikazali od izdelave Forms datoteke z že vzpostavljeno povezavo do podatkovne baze in dostopom do vseh tabel HR sheme.



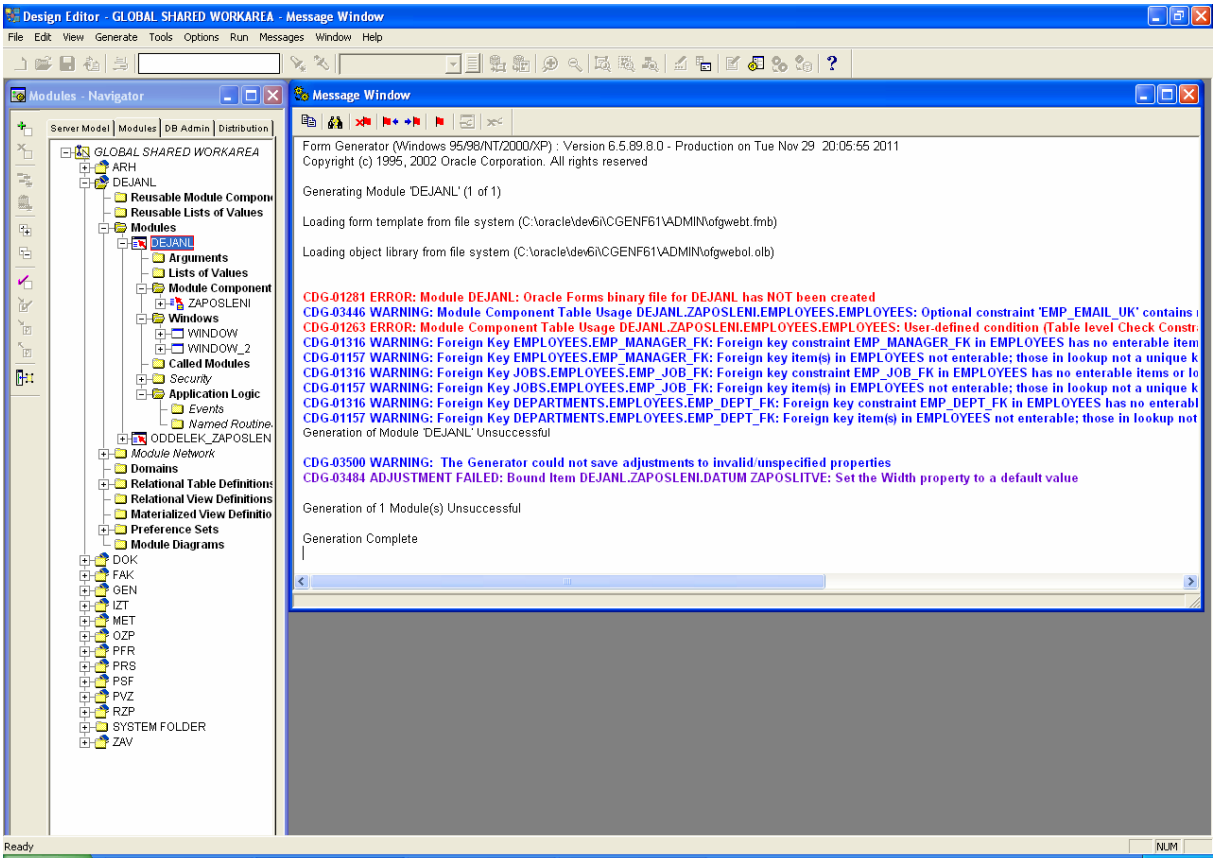
Slika 8: Delovno okolje orodja Oracle Designer za urejanje form

Zgornja slika prikazuje sklop, v katerega bomo dodali novo Formo. Pri izdelavi le-te označimo najprej razdelek »Module« in nato kliknemo na ikono za dodajanje. Odpre se novo okno za izdelavo modula z zahtevkom podatkov kot so na primer ime modula in programski jezik, kjer moramo paziti, da izberemo Orecle Forms. Ko je modul izdelan, lahko pričnemo z dodajanjem komponent in baznih tabel, ki se bodo uporabljale za prikaz podatkov. To storimo tako, da se znotraj novo ustvarjenega modula premaknemo na razdelek Module Component in kliknemo na ikono za dodajanje. Takrat se bo odprl čarovnik, ki bo po naslednjih korakih vodil postopek za dodajanje komponent:

- vnos imena in naslova komponente,
- izbira podatkovne tabele
- izbira osnovnih operacij nad tabelo ter tip vira podatkov,
- izbira stolpcev podatkovne tabele,
- vnos prikaznih imen stolpcev,
- nastavitev osnovnih operacij za posamezen stolpec,

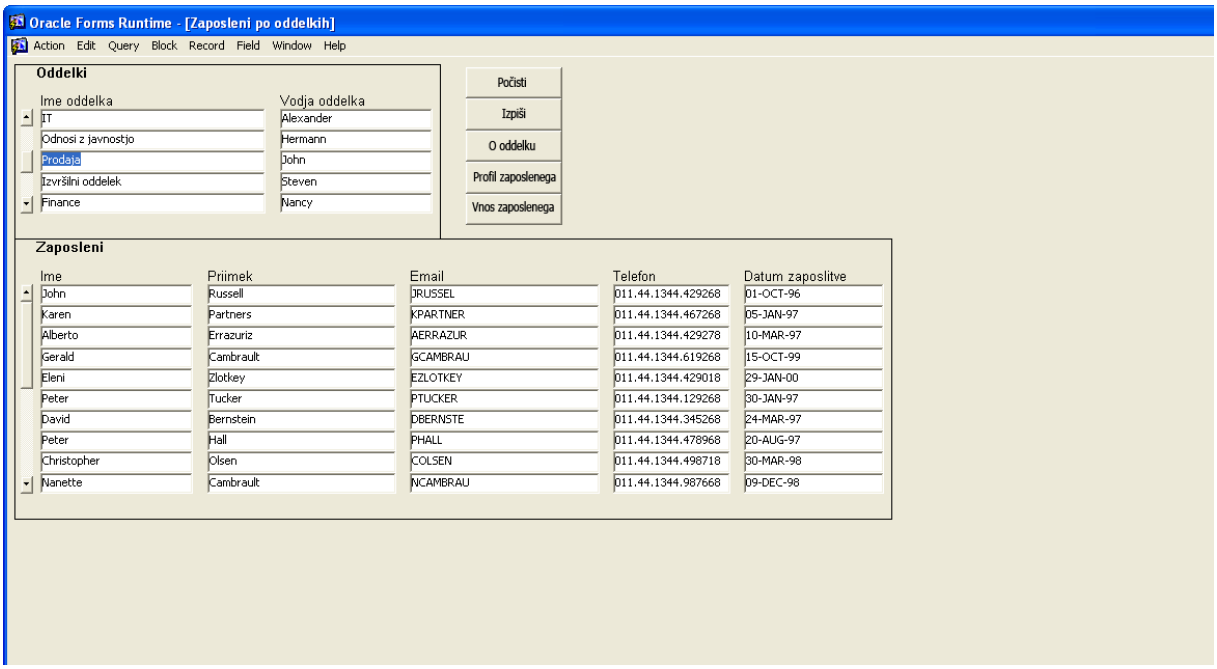
- izbira obveznih stolpcev,
- izbira tujih ključev, ki referencirajo druge tabele,
- napredne lastnosti forme.

Po končanem postopku lahko izberemo možnost za generiranje Forms datoteke. V primeru uspešnega prevajanja bo datoteka formirana, v primeru napake pa se nam bo le-ta izpisala na zaslonu.



Slika 9: Izpis napake pri generiranju Forms datoteke.

S ponavljanjem postopka lahko v modul dodamo več komponent, kar nam omogoča izdelavo vse kompleksnejših Forms datotek, ki bodo služile za prikaz kompleksnejših, med seboj odvisnih podatkov iz različnih podatkovnih tabel. Na sliki 10 je prikazana Forma, ki prikazuje podatke o zaposlenih na določenem oddelku. Sestavljena je iz dveh komponent, očetovske tabele »Oddelki« ter hčerinske tabele »Zaposleni«, ki sta med seboj povezane s tujim ključem. S premikanjem po zapisih očetovske tabele se nam v hčerinski tabeli prikazujejo ustrezni zapisi.

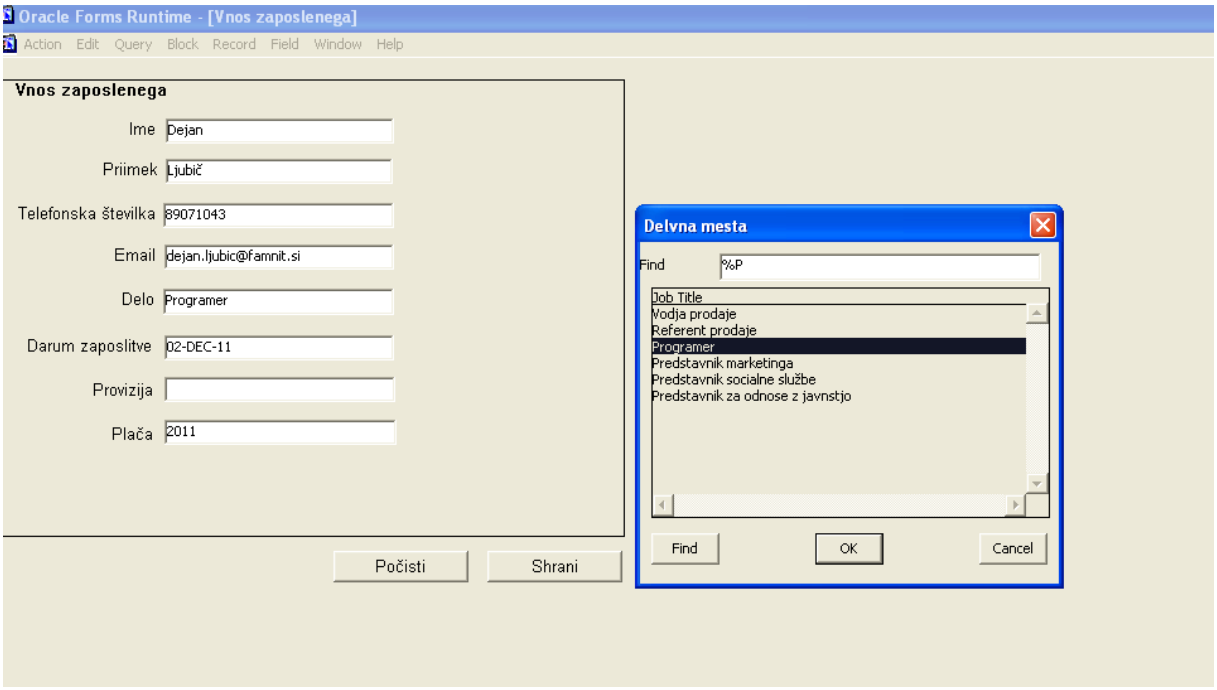


Slika 10: Obrazca za pregled zaposlenih v Oracle Forms.

Pri vnašanju novih podatkov lahko pri določenih vnosnih poljih izberemo eno od vnaprej ponujenih možnosti, na primer na katerem oddelku bo delala na novo zaposlena oseba oziroma katero službo bo opravljala. Ko lahko izbiramo med manjšim številom možnosti, pride v poštev domena, ko pa je teh možnosti več, je primernejša lista vrednosti. V nadaljevanju bomo prikazali, kako lahko dodamo listo vrednosti na polje »delovno mesto«.

Za izdelavo liste vrednosti je v podatkovnem pogledu na voljo čarovnik, ki po sledečih korakih vodi po potrebnih nastavitvah:

- vnos imena in naslova komponente,
- izbira podatkovne tabele,
- izbira imena tabele,
- izbira stolpcev podatkovne tabele,
- vnos prikaznih imen stolpcev,
- izbira preslikave izbranega stolpca,
- vnos dodatnih omejitev,
- napredne lastnosti (npr. višina oziroma širina polj).

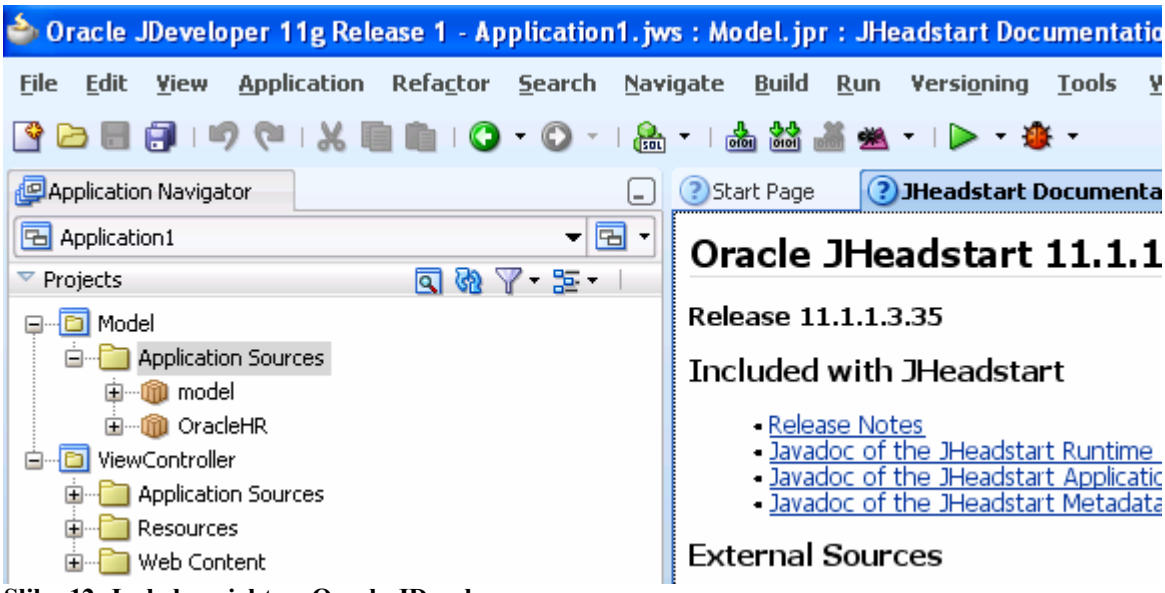


Slika 11: Obrazec za vnos zaposlene osebe v Oracle Forms z listo vrednosti.

Iz slike je razvidno, da se pri izbiri vrednosti v polje zapiše naziv dela, ne pa identifikacijska številka izbranega dela, ki ustreza delu v tabeli »Delovna mesta« in katero hranimo v tabeli Zaposleni. Vrnjene vrednosti je mogoče nastavljati v nastavitvah same liste vrednosti. V vseh zgoraj navedenih formah so bili gumbi dodani z orodjem Oracle Forms Builde, in sicer s kasnejšo obdelavo .fmb datotek, ki smo jih dobili pri generiranju v Oracle Designer okolju.

5.3. Oracle ADF aplikacija

V nadaljevanju bo predstavljen primer izdelave ADF aplikacije, ki je funkcijsko enakovredna Forms aplikaciji. Tako kot v prejšnjem poglavju tudi tu predpostavljamo, da je v JDeveloperju ustvarjen nov projekt s povezavo do podatkovne baze in nameščenim dodatkom JHeadstart. ADF aplikacija je razdeljena na dva gradnika, to sta Model in ViewController. V prvem delu se nahaja vsa logika aplikacije, v drugem pa se ta poveže z grafičnim vmesnikom. Testiranje pravilnega delovanja funkcij in metod je tako možno izpeljati še preden je pripravljen grafični vmesnik.

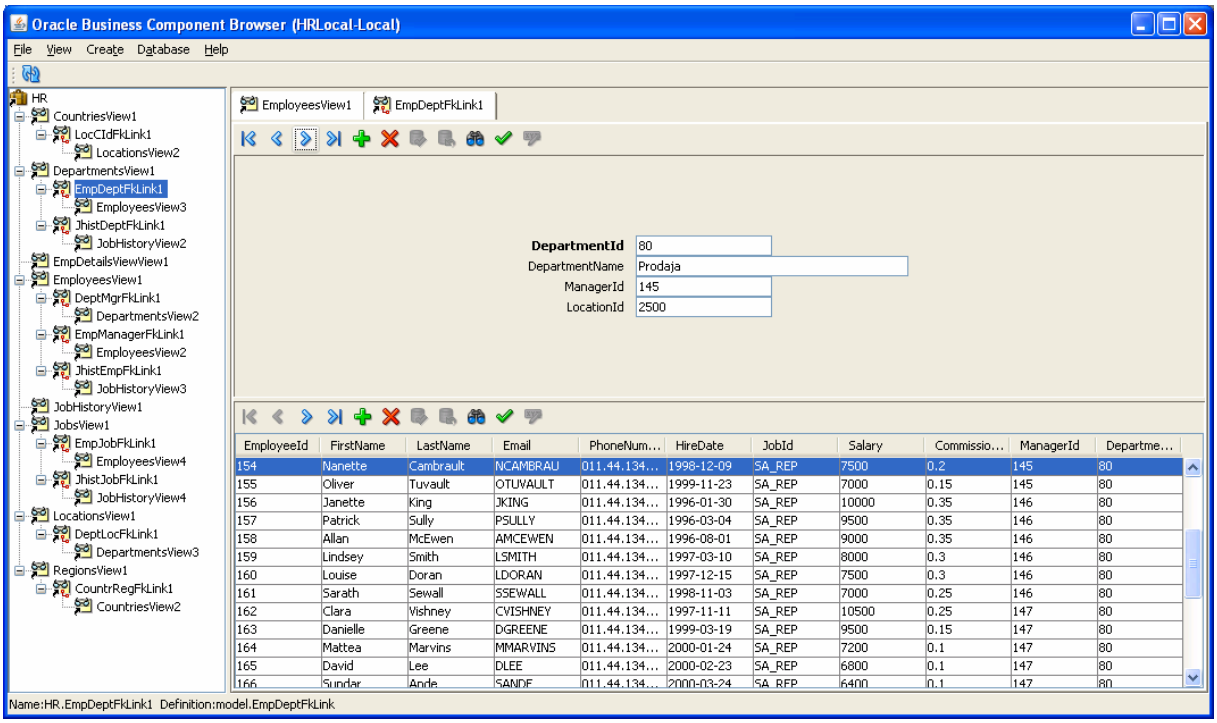


Slika 12: Izgled projekta v Oracle JDeveloper

Ko je projekt ustvarjen in povezava do podatkovne baze deluje, je v aplikacijo potrebno dodati entitete za vse podatkovne tabele, ki jih bomo uporabili. To storimo tako, da poženemo čarovnik Create Business Component from Tables, ki nas po korakih vodi do pravilne namestitve:

- izbira entitet,
- izbira objektov katere bo mogoče spreminjati,
- izbira objektov samo za branje,
- izbira aplikacijskega modula,
- izdelava ER diagrama.

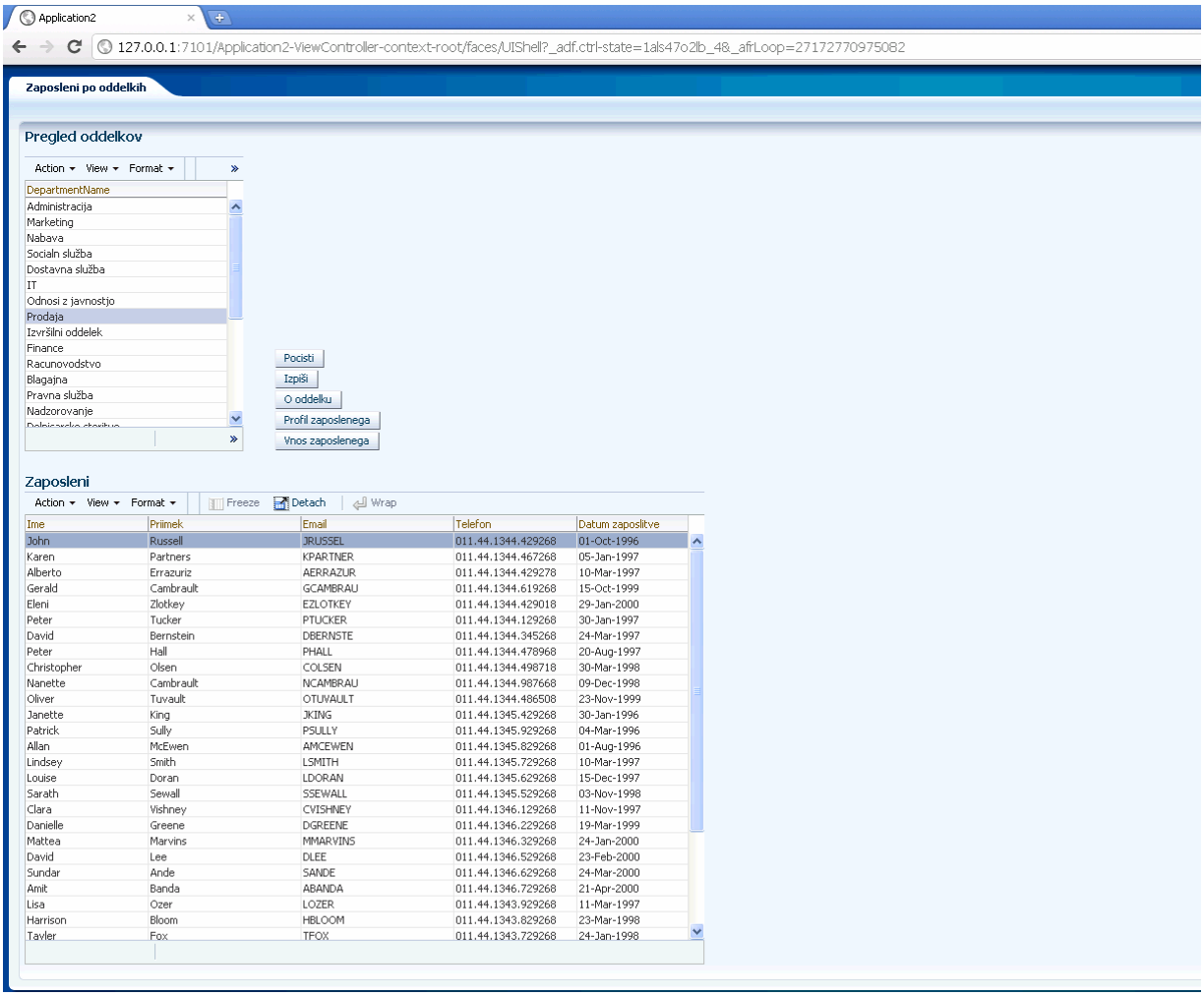
Po zaključenem postopku nam JDeveloper ustvari potrebne XML znotraj Modela, katerega lahko poženemo in takoj preverimo, ali aplikacijski model podpira želene operacije kot so brisanje, dodajanje in urejanje zapisov, poizvedovanje oziroma klici lastnih funkcij in metod.



Slika 13: Testiranje modela aplikacije v Oracle ADF

Naslednja stopnja gradnje aplikacij zajema izdelavo grafičnega uporabniškega vmesnika, pri čemer si bomo pomagali z orodjem JHeadstart. V Application Definition Editorju dodamo nov servis, ki predstavlja prvi nivo menija. Znotraj njega pa ustvarimo grupe, ki se v meniju prikažejo kot drugi nivo. V nadaljevanju bomo prikazali, kako lahko izdelamo spletno aplikacijo za prikazovanje zaposlenih po oddelkih in kako dodati nove zaposlene osebe.

V servisu ustvarimo novo grupo »Oddelki«, ki bo imela pogled na tabelo »Oddelki«. Znotraj grupe lahko sedaj vidimo naštete vse atribute podatkovne tabele »Oddelki«, ki jih lahko poljubno prikažemo na strani tako, da za vsakega posebej nastavimo ali naj bo prikazan ali ne. Ustvarjeni grupi sedaj dodamo podgrupo »zaposleni« ter nastavimo primerno podatkovno zbirko, za vsak atribut pa posebej nastavimo želene lastnosti. Podobno kot v Forms aplikaciji moramo tudi v tem primeru posebej dodati gumbe, ki bodo poklicali metode oziroma odpre novo stan. Na koncu ponovno generiramo aplikacijo, tako da se spremembe shranijo v XML datoteke.

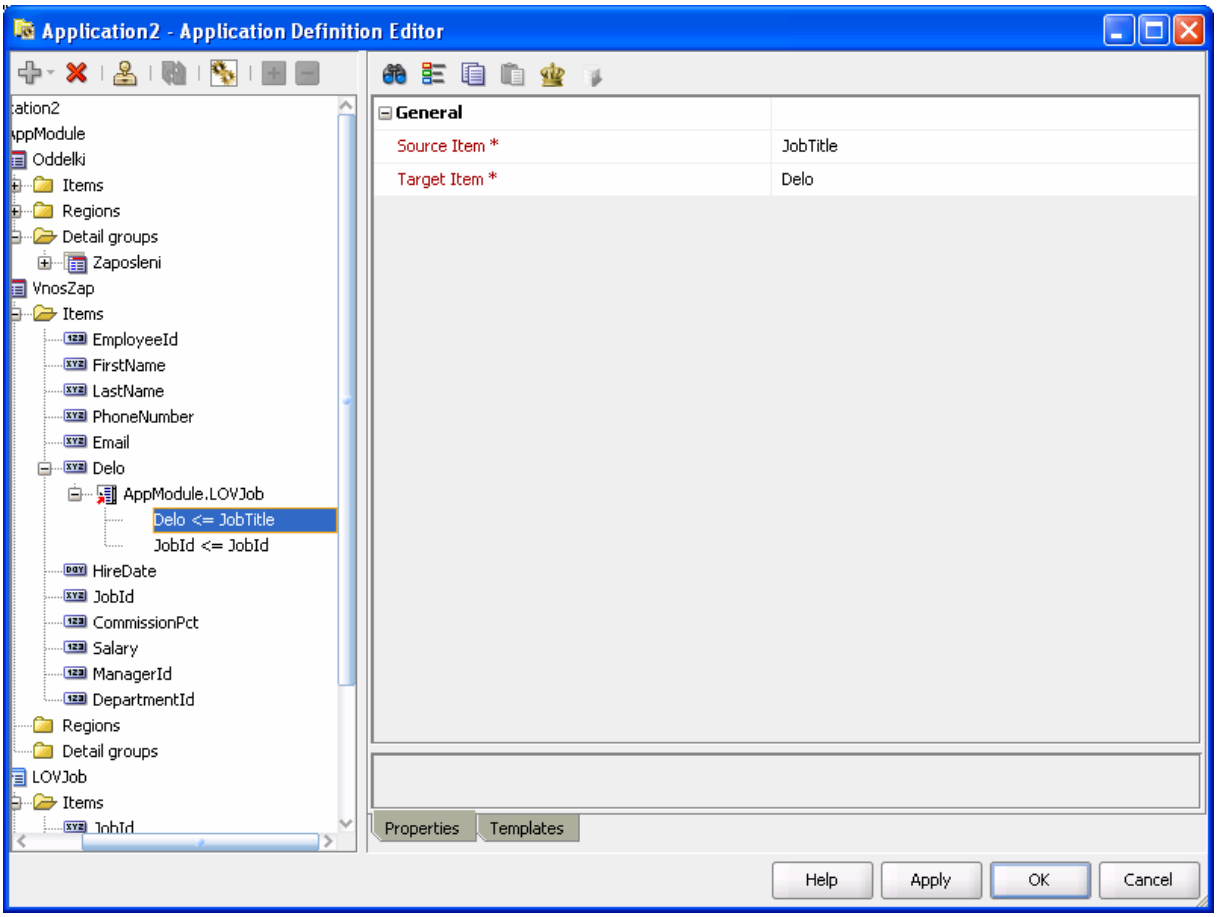


Slika 14; Obrazec za pregled zaposlenih v oracle ADF

Za izdelavo strani za dodajanje zaposlene osebe moramo ustvariti novo grupo. Ta temelji na tabeli »Zaposleni«, kateri nastavimo stil razporeditve na vnosni obrazec. V Forms aplikaciji je bila nad poljem »Delo« postavljena lista vrednosti, ki je preslikala identifikacijsko številko izbranega dela v tabelo »Zaposleni«. Listo vrednosti nad določenem polju v ADF tehnologiji lahko ustvarimo na več načinov. Eden izmed možnih pristopov predvideva listo vrednosti v entitetnem modelu aplikacije nad izbranim atributom, drugi pristop pa je ustvarjen na osnovi nove grupe v oknu Application Definition Editor. Obema pristopoma je skupna podatkovna zbirka, ki bo poizvedovala nad želeno tabelo. Novi grupi za delovna mesta moramo nastaviti namen uporabe in stil razporeditve, saj privzeto nista nastavljena. Lista vrednosti je sedaj ustvarjena in jo lahko po lastni izbiri uporabimo nad poljem na naslednji način:

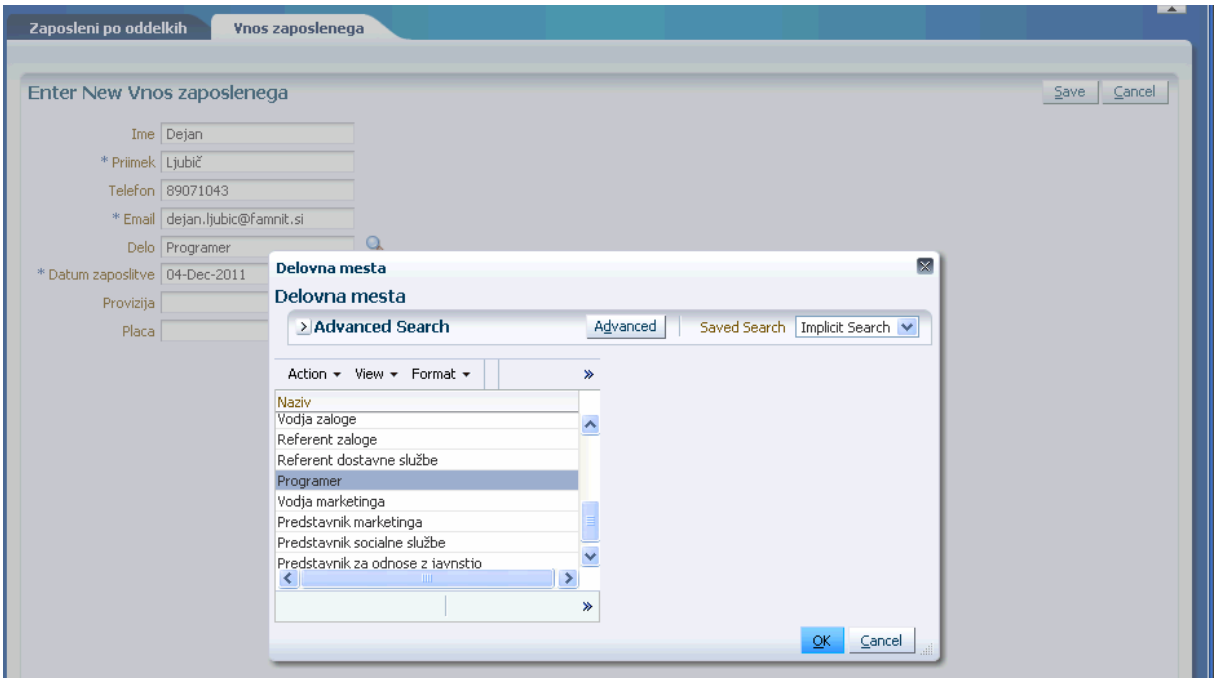
- v želeni grupi ustvarimo nov element,
- z uporabo liste vrednosti elementu nastavimo polje za validacijo,
- nad poljem dodamo prej ustvarjeno listo vrednosti,

- dodajamo vrnjene vrednosti.



Slika 15: Vračanje atributov liste vrednosti v želena polja

Aplikacijo je sedaj potrebno shraniti in ponovno generirati ter pognati.

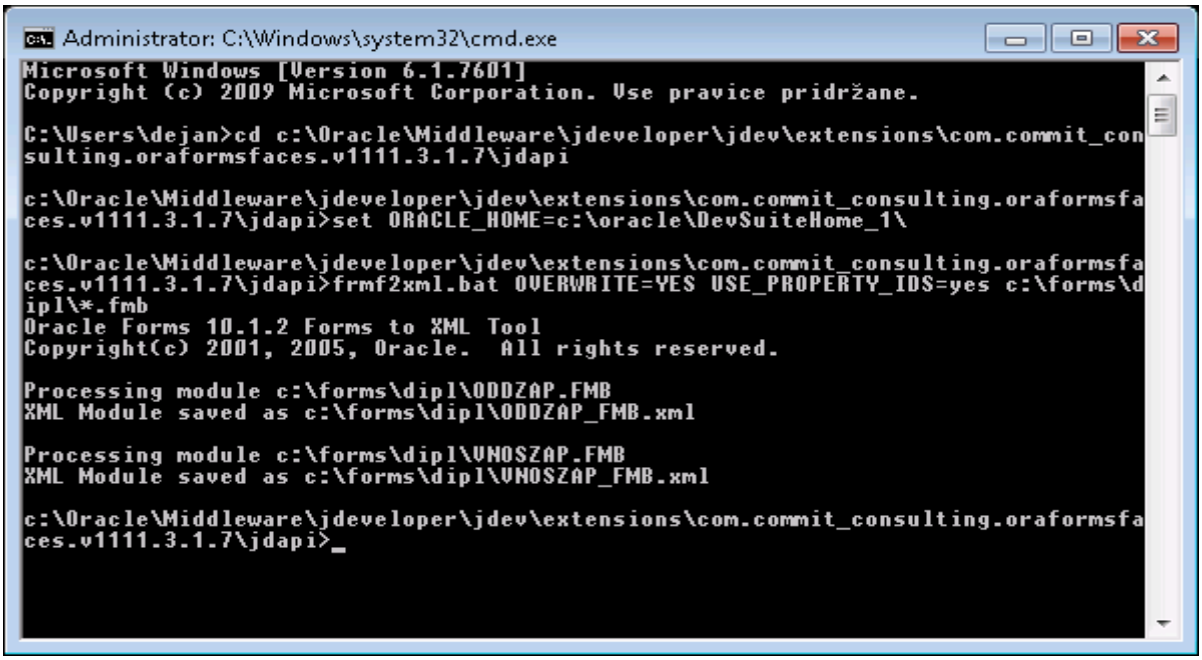


Slika 16: Obrazec za vnos zaposlene osebe v Oracle ADF z listo vrednosti

Dodatek JHeadstart omogoča avtomatizirano izdelavo ADF aplikacije z uporabo orodja Forms2ADF. Preden uporabimo to orodje moramo pripraviti XML verzije vseh forms datotek, ki jih želimo preseliti v ADF tehnologijo. To lahko storimo tako, da v konzoli uporabimo ukaz:

frmf2xml.bat OVERWRITE=YES USE_PROPERTY_IDS=yes [pot do form datotek]

Pri tem opozarjamo, da je za izvajanje ukaza potrebno imeti nameščen Oracle Forms strežnik.

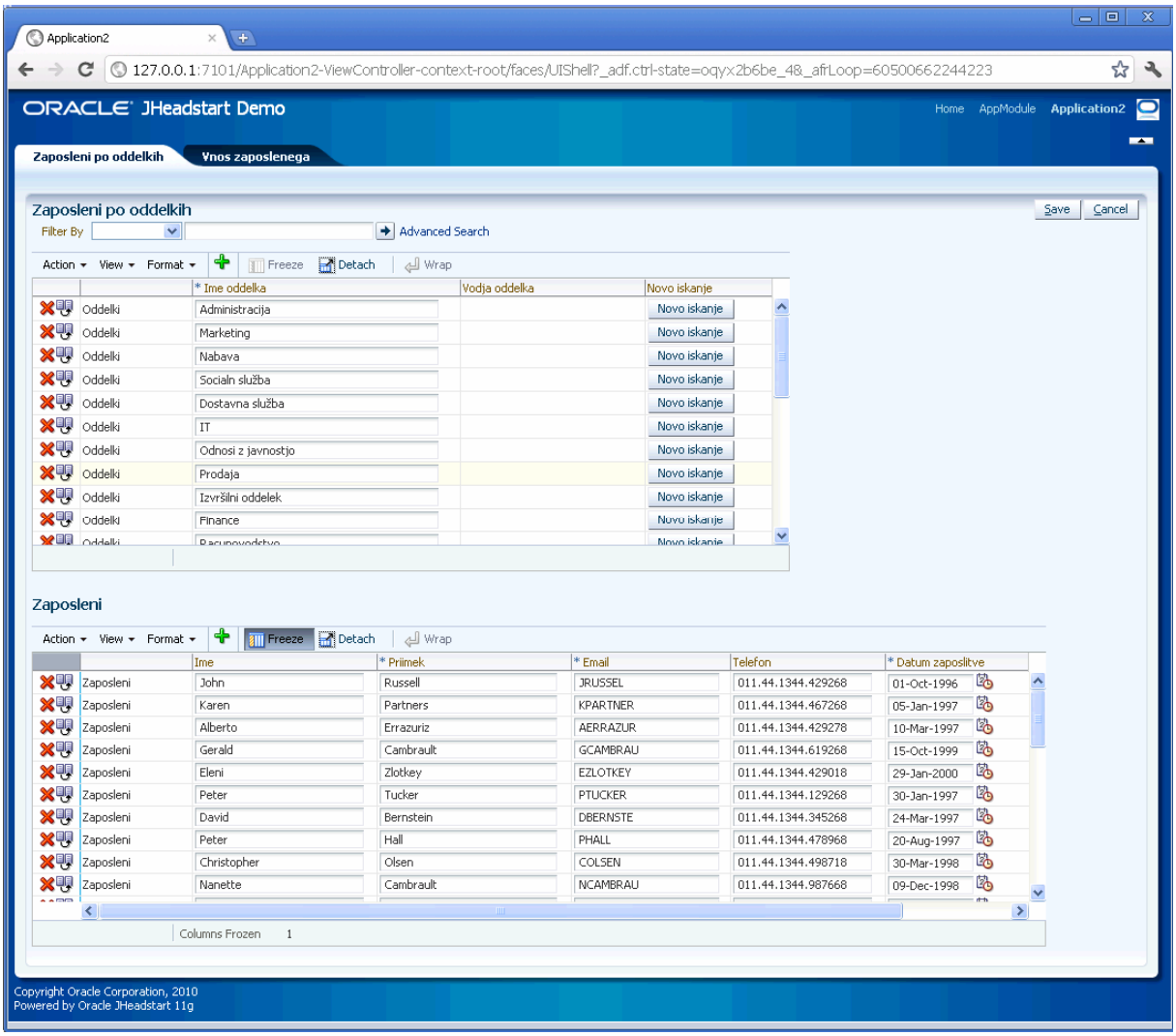


Slika 17: Izdelava XML verzij forms datotek

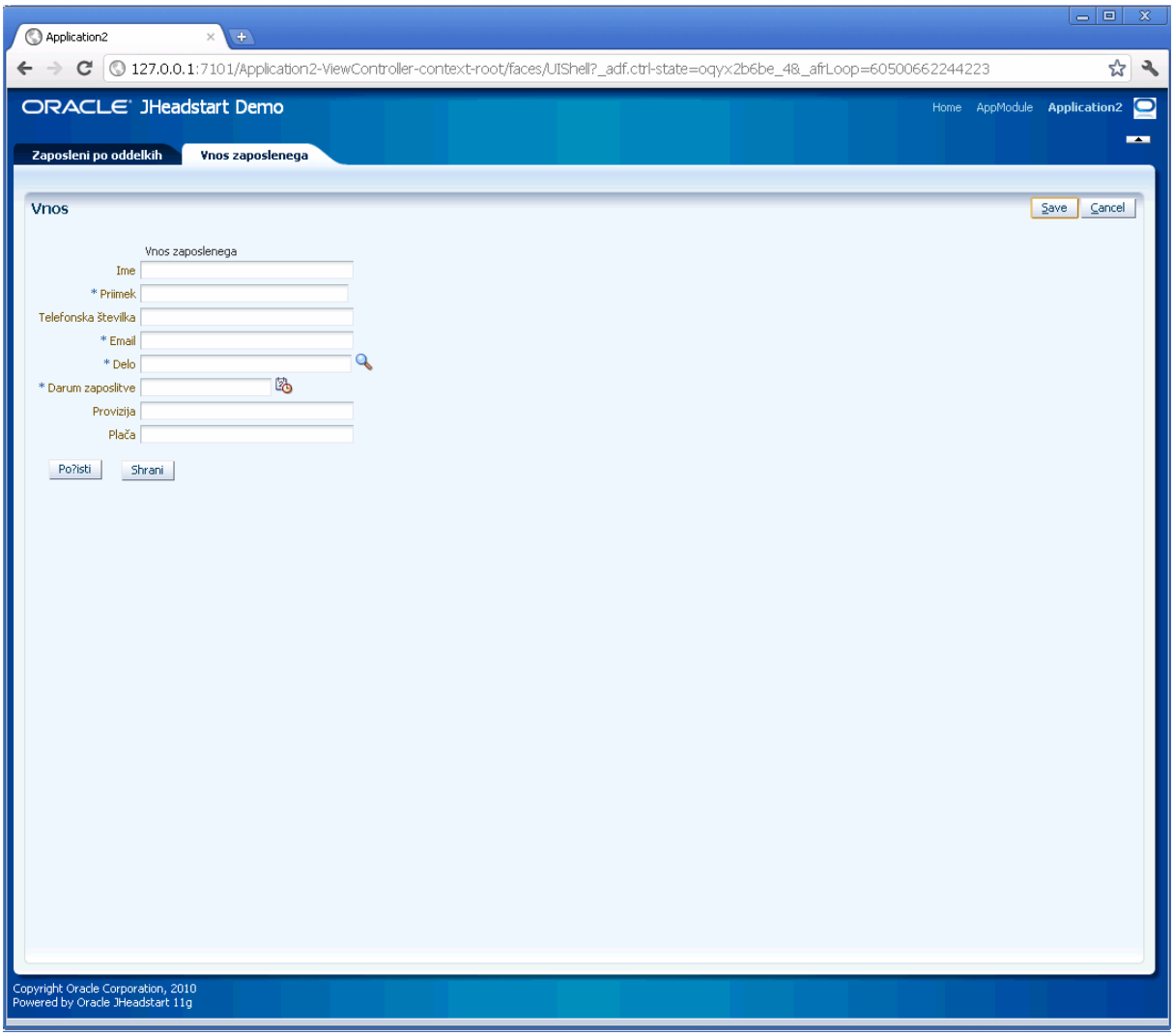
Potem ko smo ustvarili XML datoteke, lahko poženemo čarovnik, ki nas bo vodil pri uporabi orodja Forms2AD. V aplikaciji z desnim klikom nad Modelom izberemo opcijo »New« in poiščemo JHeadstart Forms2ADF Generator ter sledimo korakom:

- izbira XML datoteke želene forme,
- izbira elementov, ki jih želimo izločiti (npr. orodne vrstice),
- izbira povezave do podatkovne baze, ki hrani uporabljene tabele in podatke,
- dodatne nastavitve za ADF in
- dodatne nastavitve za JHeadstart.

Ob zaključku postopka aplikacijo ponovno generiramo in poženemo. V sliki 18 in 19 lahko vidimo obrazca za pregled zaposlenih in vnos zaposlene osebe. Če primerjamo rezultate s slikama 15 in 16, lahko opazimo razlike pri prikazu tabel in postavitvi gumbov. Te razlike lahko popravimo v JHedstartu.



Slika 18: Forms2ADF obrazec za pregled zaposlenih oseb



Slika 19: Forms2ADF obrazec za vnos zaposlene osebe

V prvem delu zaključne naloge je opisan pristop združitve obeh sistemov oziroma integracija obstoječe Oracle Froms v Oracle ADF aplikacijo, kar nam omogoča knjižnica OraFormsFaces. Ko je v JDeveloperju nameščen dodatek OraFormsFaces, lahko v grafičnem urejevalniku spletne strani dodamo komponento Form, kateri kot parameter nastavimo ime forme, ki se bo pognala. Prav tako je možno dodajati razne parametre, ki jih potrebujemo pri interakcijo obeh sistemov s komponento Form Parameter. Obe komponenti dodamo tako, da jih z metodo povleci-in-spusti prenesemo na želen razdelek strani.

Preden zaženemo stran je potrebno še preveriti, ali imamo nastavljeno pravilno pot do Forms strežnika, kjer se nahajajo forms datoteke in vse potrebne knjižnice. Te nastavitve najdemo v datoteki web.xml.

6. ZAKLJUČEK

Zaključna naloga obravnava problem selitve aplikacij iz razvojnega okolja Oracle Forms v tehnološko novejše razvojno okolje Oracle ADF. Vsebinsko je naloga zgrajena iz dveh delov – teoretičnega in praktičnega. Teoretični del podrobneje opisuje obe razvojni okolji ter predstavlja dejavnike, ki vplivajo na odločitve o selitvi aplikacije med tehnologijama. Našteti in opisani so tako pravilni kot napačni razlogi, ki vplivajo oziroma ki bi jih bilo potrebno upoštevati pri odločitvi v zvezi s selitvijo aplikacije. V praktičnem delu so prikazani pristopi, ki jih uporabljamo pri selitvi aplikacij iz Oracle Forms v Oracle ADF. Predstavljena je tudi podatkovna baza, nad katero sta razviti aplikaciji. Sam razvoj je opisan po korakih in prikazan s pomočjo slikovnega gradiva, katerega smo se poslužili tudi pri prikazu razlik v uporabniških vmesnikih.

Pri pisanju zaključne naloge mi je največjo težavo predstavljal praktični del, saj do tedaj nisem poznal Oracleovi razvojni okolji. Težave so se zato pojavljale tudi pri uporabi razvojnega orodja Oracle Designer, medtem ko je bilo delo z JDeveloper prijaznejše, predvsem zaradi podobnosti z orodjem Eclipse, katerega sem uporabljal med študijem fakulteti.

Možnost nadaljnjega dela vidim v primerjavi okolja Oracle ADF z okoljem Microsoft.NET za izdelavo specifične spletne aplikacije. Takšna primerjava bi lahko zajemala podobnosti in razlike v funkcionalnosti, grafični podobi ter zahtevnosti razširitve določenih komponent z implementacijo lastnih metod.

LITERATURA

[1] Koletzke P., Mills D., 2006, Oracle JDeveloper 10g for Forms & PL/SQL Developers: A Guide to Web Development with Oracle ADF, 1. izd, ZDA: McGraw-Hill Osborne Media

[2] Mills, D, Koletzke P, Roy-Federman A., 2009, Oracle JDeveloper 11g Handbook: A Guide to Fusion Web Development, 1. izd, ZDA: McGraw-Hill Osborne Media

[3] Nimphius F., Munsinger L., 2010, Oracle Fusion Developer Guide: Building Rich Internet Applications with Oracle ADF Business Components and Oracle ADF Faces, ZDA: McGraw-Hill Osborne Media

[4] Palinski J., 2003, Oracle9i Developer: Developing Web Applications with Forms Builder, 1. izd, Course Technology

[5] Ronald G., From Forms to ADF – When Why, and How? Dostopno 1. 8. 2011 20:42 na Intenetu: http://download.oracle.com/otndocs/OTNVDD-REA/ppt/FormstoADF2.pdf

[6] Ronald G., 2010, Quick Start Guide to Oracle Fusion Development: Oracle JDeveloper and Oracle ADF, 1. izd, ZDA: McGraw-Hill Osborne Media

[7] Van der Deijl W., Integrating Oracle Forms and ADF Faces? Dostopno 26. 8. 2011 21:24 na Intenetu: http://www.oracle.com/technetwork/articles/wilfred-adf-forms-099635.html

[8] Van Der Deijl W., OraFormsFaces Developer's guide v 3.1.7. Dostopno 20. 6. 2011 21:14 na Intenetu: http://static.commit-consulting.com/oraformsfaces/OraFormsFaces-devguide.pdf

SPLETNI VIRI

[9] Arhitektura Oracle ADF. Dostopno 21. 9. 2011 21:40 na Intenetu: http://download.oracle.com/docs/cd/B25006_02/tour/qt/j2ee009.htm

[10] Building Rich Enterprise JSF Applications with Oracle JHeadstart for ADF (11.1.1). Dostopno 20. 6. 2011 21:22 na Intenetu: http://download.oracle.com/consulting/jhstutorial1111.pdf

[11] Case Study: Redeveloping an Oracle Forms application using Oracle JDeveloper 11g and Oracle ADF 11g. Dostopno 20. 6. 2011 20:40 na Intenetu: http://www.oracle.com/technetwork/developer-tools/jdev/redeveloping-forms-in-adf-11g-1-133095.pdf

[12] From Oracle Forms to Oracle ADF and J2EE, Modernizing Oracle Forms applications to Oracle Application Development Framework and the J2EE Architecture. Dostopno 26. 8. 2011 21:24 na Intenetu: http://www.pitss.com/fileadmin/pitss/images/de/White_Papers/2009-06_PITSS_White_Paper_ADF.pdf

[13] Oracle Application Development Framework Overview? Dostopno 15. 10. 2011 21:24 na Intenetu: http://download.oracle.com/otndocs/tech/ias/portal/files/RG/WhitePaper/ADF_overview.pdf

[14] Oracle Application Express Application Migration Guide. Dostopno 21. 9. 2011 21:27 na Intenetu: http://download.oracle.com/docs/cd/E14373_01/migrate.32/e13368/appmgr_forms.htm

[15] Oracle Database Sample Schemas Dostopno 5. 9. 2011 18:30 na Intenetu: http://download.oracle.com/docs/cd/B28359_01/server.111/b28328.pdf

[16] Oracle Fusion Middleware: Developer's Guide for Oracle Business Intelligence Enterprise Edition 11g Release (11.1.1). Dostopno 20. 6. 2011 20:33 na Internetu: http://download.oracle.com/docs/cd/E14571_01/bi.1111/e10545.pdf

[17] Oracle Fusion Middleware: Fusion Developer's Guide for Oracle Application Development Framework 11g Release 1 (11.1.1.4.0). Dostopno 20. 6. 2011 21:05 na Intenetu: http://download.oracle.com/docs/cd/E17904_01/web.1111/b31974/toc.htm

[18] Oracle JHeadstart 11g for ADF release 11.1.1.3 Developer's guide. Dostopno 20. 6. 2011 21:11 na Intenetu: http://download.oracle.com/consulting/jhsdevguide1111.pdf

PRILOGE

Ime stolpca	Začetna vrednost	Podatkovni tip
IdZaposlenega	Obvezna	Število (6)
Ime		Niz(20)
Priimek	Obvezna	Niz (25)
Email	Obvezna	Niz (20)
Telefon		Niz (20)
DaumZaposlitve	Obvezna	Datum
IdDela	Obvezna	Niz (10)
Plača		Število (8,2)
DeležProvizije		Število (2,2)
IdManagerja		Število (6)
IdOddelka		Število(4)

Tabela 1: Struktura tebele Zaposleni [15]

Ime stolpca	Začetna vrednost	Tip
IdDela	Obvezna	Niz(10)
Naziv	Obvezna	Niz(35)
MinPlača		Število(6)
MaxPlača		Število(6)

Tabela 2: Struktura tabele Delovna mesta[15]

Ime stolpca	Začetna vrednost	Podatkovni tip
IdZaposlenega	Obvezna	Število(6)
DatumPričetkaDela	Obvezna	Datum
DatumKoncaDela	Obvezna	Datum
IdDela	Obvezna	Niz(10)
IdOddelka		Število(4)

Tabela 3: Struktura tabele Zgodovina delovnih mest[15]

Ime stolpca	Začetna vrednost	Podatkovni tip
IdOddelka	Obvezna	Število(4)
ImeOddelka	Obvezna	Niz(30)
IdManagerja		Število(6)
IdNaslova		Število(4)

Tabela 4: Struktura tabele Oddelki[15]

Ime stolpca	Začetna vrednost	Podatkovni tip
IdNaslova	Obvezna	Število(4)
Naslov		Niz(40)
PoštnaŠtevilka		Niz(12)
Mesto	Obvezna	Niz(30)
Regija		Niz(25)
IdDržave		Znak(2)

Tabela 5: Struktura tabele Naslovi[15]

Ime stolpca	Začetna vrednost	Podatkovni tip
IdDržave	Obvezna	Znak(2)
Država		Niz(40)
IdKontinenta		Število

Tabela 6: Struktura tabele Države[15]

Ime stolpca	Začetna vrednost	Podatkovni tip
IdKontineneta	Obvezna	Število
Kontinent		Niz(25)

Tabela 7: Struktura tabele Kontinenti[15]