# OBJEKTI IN INTRANET ZA IZDELAVO SISTEMA ZA OBRAČUN ZEMELJSKEGA PLINA

**Janko Mivšek*, Matjaž Wiegele****

*Eranova d.o.o., Pod hribom 55, Ljubljana

e-pošta: janko.mivsek@eranova.si

URL: http://www.eranova.si

*TRIS A d.o.o., Pod hribom 55, Ljubljana

e-pošta: mwiegele@tris-a.si

URL: http://www.tris-a.si

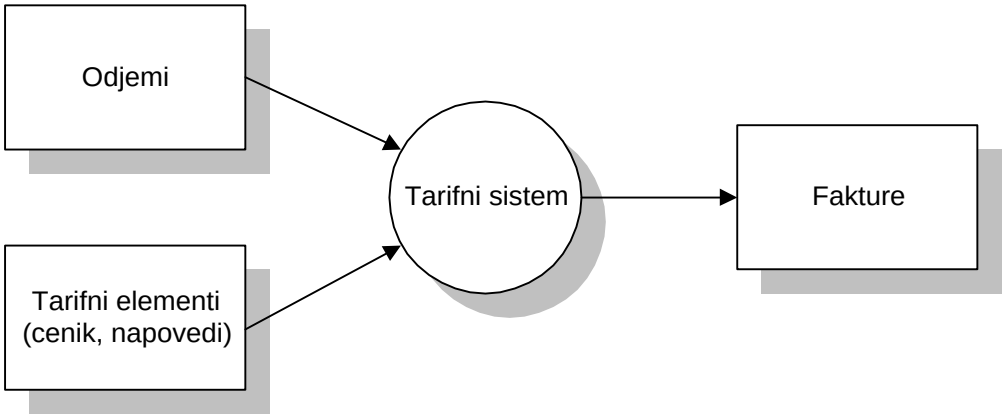
*Povzetek*

*Lansko leto smo dokaj nepričakovano dobili zahtevek za ponudbo izdelave paketa za obračun zemeljskega plina na podjetju Geoplin d.o.o. Ljubljana. Zahtevek je prišel sredi avgusta, rok za izdelavo je bil konec leta. Ali smo sposobni ponuditi rešitev v tako kratkem času? Rešitev, ki bo pokrila dokaj kompleksen problem, ki bo sodobna, zanesljiva, lahko vzdržljiva in razširljiva in ki naročniku ne bo zaprla poti pri nadaljnjem razvoju informacijskega sistema? Rešitev z uporabo navadnih spletnih pregledovalnikih pri uporabnikih ter spletnim strežnikom, izvedenim v okolju Smalltalk skupaj z objektno bazo? Odločili smo se za vstop v visoko tvegan projekt in uspeli. V prispevku opisujemo najbolj zanimive trenutke projekta, uporabljeno tehnologijo, arhitekturo sistema, način dela ter rezultate.*

## 1. UVOD

Sistem za obračun zemeljskega plina AIDA/GBS je namenjen za spremljanje in obračunavanje odjema plina. Poenostavljeno gledano mora sistem iz podatkov o odjemu plina na merilnih mestih ter iz tarifnih elementov (cenik, napovedi porabe ipd.) po pravilniku o tarifnem sistemu pravilno obračunati porabo plina in izdati fakture (slika 1).

Naročnik se je bil primoran odločiti za izdelavo novega sistema zaradi korenite spremembe tarifnega sistema obračuna zemeljskega plina, ki ga je sprejela vlada v sredini lanskega leta.




slika 1: Poenostavljen delovni tok sistema

## 2. IZVEDBA PROJEKTA

Kar je bil rok za izvedbo izredno kratek (3 mesece, oziroma do novega leta), smo projekt razdelili na 4.faze:

*1 faza: Analiza specifikacije*. V tej fazi smo analizirali problem, obstoječe stanje ter naredili objektni model novega sistema. Pri tem smo si pomagali s tako imenovanimi primeri uporabe. S pomočjo CRC (Class-Responsibility-Collaboration) kartic smo definicirali ključne objekte, njihove zadolžitve ter interakcije med njimi. Model smo že med analizo prototipno izvedli v Smalltalk okolju. Model je narejen po metodologiji UML, ki postaja de-facto standard na področju objektega modeliranja. Analizo smo izvedli v tesnem sodelovanju z naročnikom v enem tednu, kajti želeli smo čimprej postaviti osnovni model sistema in se ne spuščati v podrobnosti že na začetku projekta.

*2. faza Prototip sistema*. Z izvedbo prototipa smo tako mi kot naročnik preizkusili in še dovolj zgodaj ugotovili, da je projekt na zastavljen način možno izvesti. V tej fazi smo do konca izvedli samo en del sistema - vnos odjemov plina iz merilnih mest. Vnos je precej zapleten in je bila zato ta uspešno izvedena funkcija dovolj trdno zagotovilo za izvedbo tudi ostalih funkcij. V tej fazi smo morali izvesti vse faze življenjskega cikla od načrtovanja, implementacije, testiranja do namestitve. Naročnik je imel ves čas možnost spremljati rezultate našega dela preko Interneta, tako da je na koncu prototip v celoti ustrezal zahtevam. Prototip smo izvedli v enem mesecu.

*3.faza: Dokončna izdelava sistema*. V tej fazi smo izvedli še preostale funkcije sistema, ki so ob koncu leta omogočale obračun plina po novem tarifnem sistemu. S tesnim sodelovanjem z naročnikom smo tako 15.januarja uspešno obračunali in izdali fakture za prvo obdobje letošnjega leta.

*4. faza: Uvajanje paketa v uporabo, vmesniki do drugih sistemov, dokumentacija*. Ker se hudič vedno skriva v podrobnostih, smo to dejstvo upoštevali in že v pogodbi definirali to

končno fazo, v kateri smo sistem 'izpilili', izdelali dodatne preglede, izboljšali spletne strani uporabniškega vmesnika in pripravili vso potrebno dokumentacijo. Ravno tako se je v tem času pokazala potreba po množici sprememb in dopolnitev, ki jih v fazi analize nismo zaznali, ker enostavno takrat nismo imeli (tako mi kot naročnik) dovolj znanja in iskušenj z novim sistemom. Vendar pa v tej pozni fazi nismo imeli zahvaljujoč dobro postavljenemu modelu v fazi analize ter dinamičnem razvojnem okolju problemov pri spreminjanju in dodelavah sistema.

Za podporo razvojnemu ciklu smo uporabili CASE orodje ADVANCE nemškega proizvajalca IC&C. To orodje je pisano v Smalltalk okolju, zato je z njim tesno povezano, kar bistveno olajša predvsem vzdrževanje in nadgradnje poslovnega modela. Vsaka sprememba v poslovnem modelu se takoj odraža v programski kodi in tudi neposredna sprememba programske kode se takoj pozna v modelu (povratni inženiring).

# 3. ARHITEKTURA SISTEMA

AIDA/GBS uporablja tehnologijo, ki združuje dve najbolj napredni smeri v današnjem računalništvu:

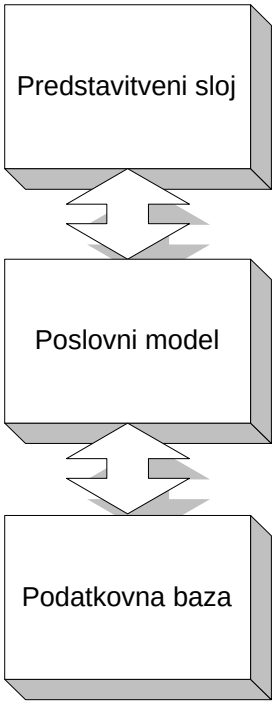
- Internet\Intranet
- Objektna tehnologija

V naslednih podtočkah bomo poskušali arhitekturo podrobneje opisati.

## 3.1. Troslojna arhitektura

Prva generacija sistemov odjemalec-strežnik (FoxPro, Clipper, Gupta SQLWindows ipd.) je izvedena na tako imenovani dvoslojni arhitekturi, in sicer je v prvem sloju združena funkcionalnost sistema ter predstavitvena koda, ki se izvaja na odjemalcih - delovnih postajah. Ker je slednje bistveno več (do 80%), se funkcionalnost porazgubi po kodi, zato je tovrstne aplikacije zelo težko vzdrževati in dograjevati. Na drugem sloju se nahaja podatkovna baza na strežniku. Zaradi omenjenih težav se v zadnjem času vse bolj uveljavlja troslojna arhitektura, pri kateri predstavitveni sloj ločimo od funkcionalnega. Naša troslojna arhitektura je sestavljena na naslednji način:

*1.sloj*: Predstavitveni (uporabniški vmesnik) na osnovi Intranet tehnologije: na odjemalcih samo spletni pregledovalnik (npr. Netscape),
*2.sloj*: Poslovni model, ki predstavlja in izvaja poslovne funkcije, v našem primeru obračun plina, teče na aplikacijskem strežniku. Poslovni model je implementiran v programskem jeziku in okolju Smalltal,
*3.sloj*: Podatkovna baza, ki skrbi za varno shranjevanje ter integriteto podatkov, ki jih uporablja poslovni model, teče na podatkovnem strežniku. Podatkovna baza je objektna, podjetja Versant in je tesno povezana z 2.slojem.



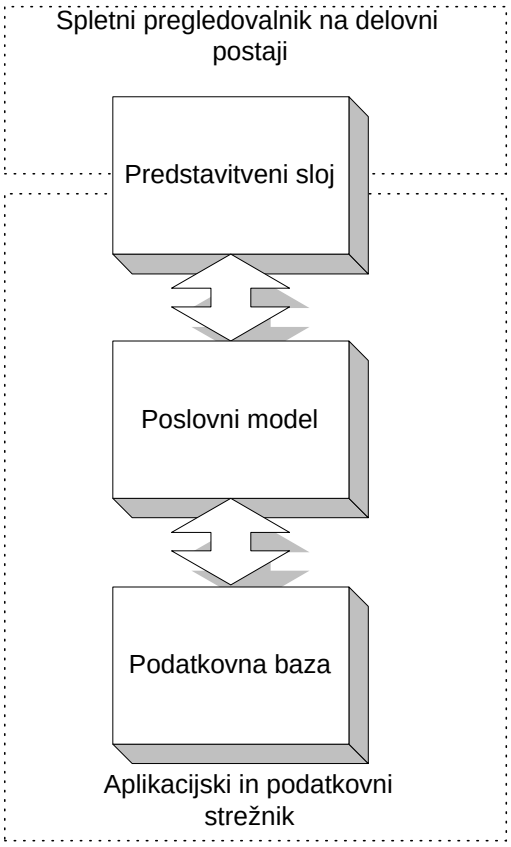

Slika 3. Troslojna arhitektura

Prednosti: troslojna arhitektura na osnovi Intranet pristopa močno poenostavi vzdrževanje in izpopolnjevanje sistema, ker je za podprtje sprememb v poslovanju potrebno spreminjanje samo na enem mestu: na aplikacijskem strežniku. Z uporabljeno tehnologijo je možno uvesti spremembe v poslovnem modelu v nekaj minutah, brez izklopa sistema. Sprememba je takoj v uporabi na vseh odjemalcih.

Obstoječe rešitve na temelju arhitekture odjemalec-strežnik imajo vso funkcionalnost ter predstavitveni del na odjemalcih, medtem ko je na strežniku podatkovna baza. V primeru sprememb je potrebno spremeniti aplikacijo na vseh odjemalcih, kar otežuje in podraži vzdrževanje sistema.

### 3.2. Intranet tehnologija

Izraz Intranet pomeni Internet za interno uporabo, znotraj podjetja. Podjetja, ki so začela s priključevanjem na Internet, so kmalu začutila, da se prednosti, kot so cenenost ter lahek dostop do raznoraznih informacij ter bogate možnosti organiziranja le-teh (na osnovi hipertekstnih povezav med dokumenti) dajo s pridom uporabiti za dostop do internih informacij - od tu naziv Intranet. Veliko študij je pokazalo zelo velik učinek naložbe v Intranet (300-600%).




slika 2. Troslojna Intranet tehnologija

Če so bile v začetku informacije na Intranetu bolj statične narave (razni poslovniki, pravilniki, poročila, telefonski imenik), pa se zadnje leto vse bolj selijo tudi klasične poslovne aplikacije na Intranet tehnologijo. Kot smo že omenili, je glavni razlog za to enostavnost tehnologije in s tem nizki stroški postavitve in vzdrževanja.

Pri izdelavi sistema AIDA/GBS smo uporabili pri nas razvit spletni strežnik in ogrodje za izdelavo dinamičnih spletnih rešitev AIDA/Web.

### 3.3. Objektna tehnologija

Osnovna značilnost objektnega pristopa je, da poskuša informatizirati realni svet čimbolj naravno, brez nepotrebnih transformacij. Zato modelira objektni pristop svet s pomočjo objektov, ki neposredno odražajo realne objekte. Klasični pristop (npr. strukturna analiza ter entitetno modeliranje v relacijski tehnologiji - Oracle, Informix, Gupta), poskuša realni svet razdeliti na podatke ter nato definirati funkcije nad podatki, kar sčasoma zamegli realni problem.

Prednosti objektne tehnologije v primerjavi z relacijsko so:

Objektni pristop je bližji naravnemu razumevanju problemov pri uporabnikih, zato je bolj razumljiv ljudem, kot so npr. organizatorji poslovanja. Pri izdelavi objektnih sistemov ni

transformacij od faze analize do izvedbe, tako kot pri relacijskih sistemih. Rezultat faze analize je poslovni model, ki se 'pili' naprej do končne izvedbe brez dodatnih transformacij.

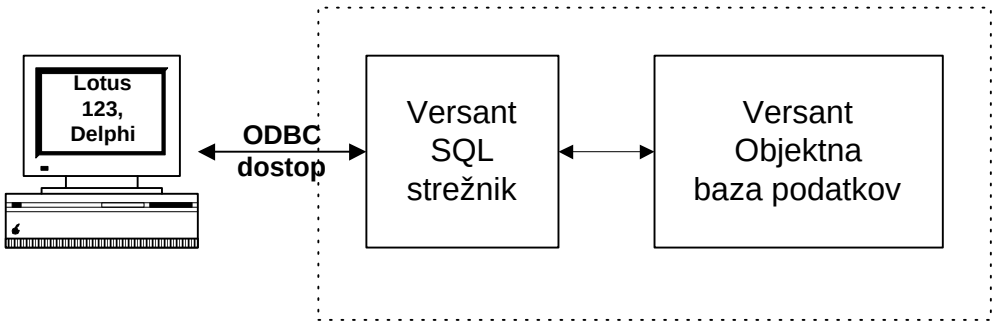
Z objektno tehnologijo je možno izvesti poljubno kompleksne sisteme. Pri relacijski smo omejeni na dvodimenzionalni svet relacijskih tabel. Kompleksnejše modeliranje zahteva uporabo več tabel, kar močno poslabša performance sistema.

Vzdrževanje in izpopolnjevanje objektnih sistemov je bistveno lažje. Spremembe v poslovnem modelu delujejo takoj, medtem ko moramo pri relacijski najprej popraviti shemo baze podatkov, nato pa še vse aplikacije, ki to bazo uporabljajo.

Hitrost objektnih sistemov je 10 - 100 krat višja od relacijskih. Osnovni razlog je v načinu dostopa do podatkov. Pri objektnih sistemih dostopamo s pomočjo navigacije (sledenja referenc med objekti), medtem ko se v relacijskih pogosto uporabljajo performančno zelo zahtevni stiki med več tabelami. Višja hitrost pomeni tudi, da niso potrebni zelo močni strežniki za doseganje ustreznih performanc sistema.

V tretjem sloju naše arhitekture je uporabljena objektna podatkovna baza Versant. Objektna podatkovna baza je tesno povezana z okoljem Smalltalk. To pomeni, da se pri izvedbi in vzdrževanju ni potrebno posebej ukvarjati s podatkovno shemo v bazi podatkov, ker se le-ta samodejno prilagaja spremembam kode v okolju Smalltalk. Naše iskušnje kažejo, da je potrebno le 1% kode nameniti delu s podatkovno bazo v nasprotju z 40-60% pri relacijski bazi podatkov. Z uporabo objektne podatkovne baze ne izgubimo možnosti modeliranja in izvedbe kompleksnih sistemov. Objektna podatkovna baza ima vse lastnosti relacijske baze, kot so zagotavljanje integritete podatkov, arhiviranje za zaščito podatkov, možnost poizvedb s SQL dostopom ipd.

Orodja, kot so Lotus 123, Approach, Delphi ipd. lahko dostopajo do podatkov v objektni bazi preko standardnega ODBC vmesnika. To je v vašem primeru še posebno zanimivo za pripravo analiz v preglednicah, kot je Lotus 123.




slika 3. SQL dostop do podatkov v objektni podatkovni bazi

### 3.4. Tiskanje

Uporabnik lahko tiska neposredno iz svojega spletnega pregledovalnika vse kar na pregledovalniku vidi. Tak način tiskanja je najpogostejši, ker je hiter in enostaven. Ni pa primeren za tiskanje v večjih količinah. Fakture se trenutno tiskajo neposredno iz

pregledovalnika. Možno je tudi tiskanje preko aplikacijskega strežnika (2.sloj) na poljuben tiskalnik (ki pa mora biti ustrezno konfiguriran na strežniku). Tiskanje preko strežnika je predvsem namenjeno tiskanju v večjih količinah, kot npr. predtiskanje obrazcev, računov ipd.

### 3.5. Neodvisnost od platforme

Ena od največjih prednosti naše tehnologije je neodvisnost od platforme - operacijskega sistema, na katerem sistem teče. Sistem je možno prenašati brez sprememb iz ene na drugo izmed naslednjih platform: Windows NT, OS/2, Sun Solaris, HP HP-UX, IBM AIX. Možna je uporaba več platform hkrati, kar lahko v opisanem sistemu pride prav v naslednjih primerih: za povečanje odpornosti delovanja: glavni strežnik teče npr. na Windows NT, medtem ko teče rezervni na OS/2, za povezavo s SCADA sistemom: glavni sistem na OS/2 ali WinNT, dodatni proces na Sun računalnikih v SCADA sistemu.

### 3.6. Varnost (zaščita pred nepooblaščenim dostopom)

Ker delujejo Intranet sistemi na osnovi Internet tehnologije in ker je na Internetu trenutno nekaj 100 miljonov uporabnikov, se možnost vdora v Intranet sisteme precej poveča. Zato smo posebno pozornost posvetili zaščiti naših sistemov pred nepooblaščenim dostopom in sicer:
- uporabnik se mora sistemu predstaviti s svojim uporabniškim imenom in geslom,
- uporabnikom lahko določimo dostopne pravice (kdo, kaj, na katerem objektu lahko dela),
- seja (Session) z uporabnikom se samodejno prekine po dveh urah neaktivnosti,
- vsa aktivnost vseh uporabnikov se beleži (Audit log).

Možna je tudi dodatna zaščita z enkripcijo komunikacije med spletnim pregledovalnikom (to je standardna funkcija pregledovalnikov, kot npr. Netscape) in aplikacijskim strežnikom.

### 3.7. Zanesljivost delovanja

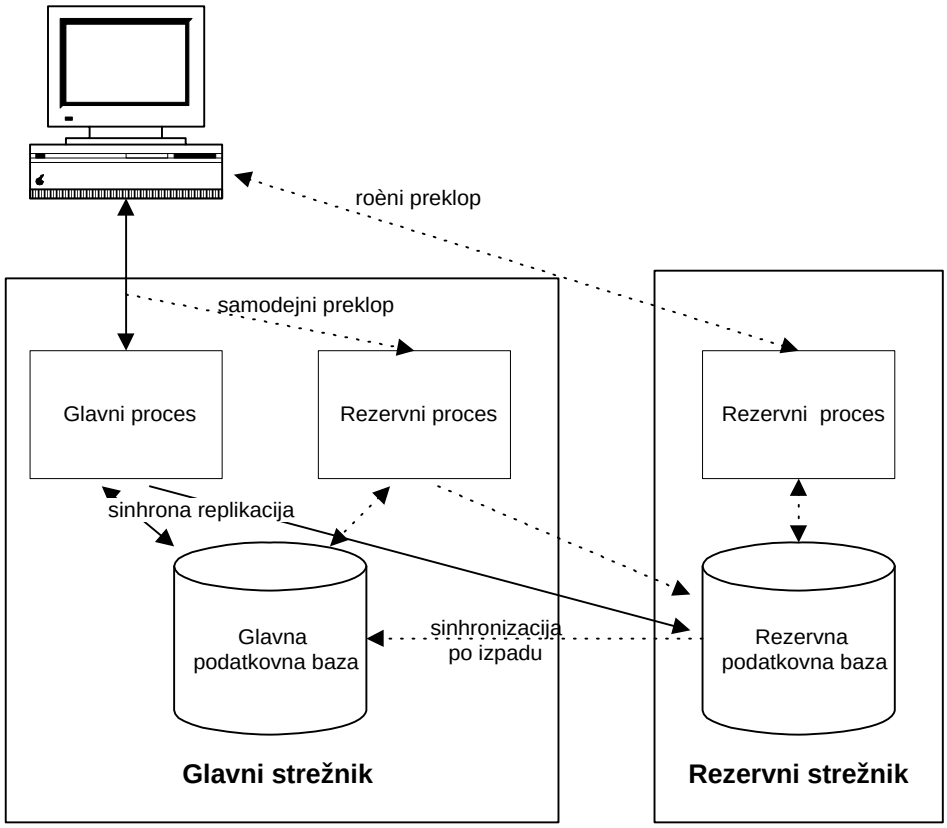
Pri troslojni arhitekturi je zelo pomembno zagotoviti zanesljivo delovanje drugega in tretjega sloja (poslovnega modela ter podatkovne baze), ker izpad enega izmed teh dveh slojev povzroči izpad celotnega sistema. Izveden sistem ima vgrajeno odpornost proti izpadom (Fault Tolerant operation) in sicer na 2. in 3. sloju:

Na drugem sloju je zaščita tronivojska:

Zaščita proti izpadom v primeru napak v izvajanju programa (Runtime errors), s pomočjo paralelnega izvajanja zahtev s strani uporabnikovih spletnih pregledovalnikov. Če pride do take napake, uporabnik samo osveži spletno stran in nadaljuje z delom. Razlog napake ostane zapisan in ga lahko kasneje diagnosticiramo ter odpravimo.

Zaščita proti izpadom v primeru napak v sistemski programski opremi, s pomočjo rezervnega procesa, ki se vklopi v času 2min od izpada glavnega procesa. Uporabniki opazijo izpad, ker se sistem 2min ne odziva, nato nadaljujejo z normalnim delom.

Zaščita proti izpadu strežnika, s pomočjo rezervnega strežnika, ki se vklopi v 2min od izpada osnovnega strežnika. Uporabniki morajo v tem primeru ročno preiti na rezervni sistem (navedejo naslov rezervnega strežnika in nadaljujejo z delom tam, kjer so bili prekinjeni).




slika 4. 'Fault tolerant' način delovanja

V vseh treh primerih ne pride do izgube dosedanjega dela (vnešenih podatkov, opravljenih transakcij). Uporabniki lahko vedno nadaljujejo delo tam, kjer so bili ustavljeni zaradi izpada.

Na tretjem sloju (objektna podatkovna baza) je možno doseči odpornost proti napakam z uvedbo rezervnega podatkovnega strežnika in sinhrono replikacijo sprememb v glavni in replicirani bazi podatkov. V primeru izpada glavnega podatkovnega strežnika se začne samodejno in takoj uporabljati rezervni strežnik. Po ponovni vzpostavitvi glavnega strežnika se obe bazi samodejno sinhronizirata.

Dodatno varujemo podatke z arhiviranjem, ki je pri objektni podatkovni bazi Versant možno brez ustavljanja sistema (on-line backup).

### 3.8. Vmesniki do drugih sistemov

Sistem omogoča povezavo z ostalimi sistemi na sledeče načine: preko datotek, z FTP prenosom, z Internet elektronsko pošto, preko serijskih linij, z uporabo TCP/IP protokola, HTTP protokola, do drugih podatkovnih baz s pomočjo ODBC (Open DataBase Connectivity), z drugimi objektnimi sistemi preko CORBA vodila (Common Object Request Broker), z Lotus Notes bazami ter z zunanjimi DLL knjižnicami.

## 4. ZAKLJUČEK

Kljub velikemu tveganju tako naročnika kot nas, saj smo sistem postavili na popolnoma novi in neznani tehnologiji, smo dosegli zastavljeni cilj: v zelo kratkem roku postaviti popolnoma nov sistem, ki zadovoljuje potrebe naročnika, hkrati pa nudi infrastrukturo za nadaljnji razvoj aplikacij v Intranet okolju naročnika.

## 5. LITERATURA


1. Sherman Alpert, et al: *The Design Patterns Smalltalk,* Addison-Wesley, 1998

2. Kent Beck: *Smalltalk Best Practice Patterns*, Prentice Hall, 1996

3. Peter Coad: *Object Models, Strategies, Patterns, and Applications,* Yourdon Press, Prentice Hall, 1995

4. Martin Fowler, *Analysis Patterns : Reusable Object Models,* Addison-Wesley, 1996

5. Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides: *Design Patterns, Elements of Reusable Object-Oriented Software,* Addison Wesley, 1995

6. Adele Goldberg, Kenneth S. Rubin: *Succeding with Objects, Decision Frameworks for Project Management,* Addison Wesley, 1995

7. Ivar Jacobson, Magnus Christerson, Patrik Jonsson, Gunnar Overgaard: *Object-Oriented Software Engineering: A Use Case Driven Approach,* Addison.Wesley, 1992

8. Janko Mivšek, TRIS A d.o.o.: *Smalltalk na Internetu*, Srečanje Objektna tehnologija v Sloveniji, Maribor 1996

9. David A. Taylor: *Business Engineering with Objcct Technology*, John Wiley & Sons, 1995