PORAZDELJENI OBJEKTI IN CORBA V SODOBNEM POSLOVNEM SVETU

Marko Juvančič

Povzetek

Način poslovanja se z novimi tehnologijami spreminja z neverjetno hitrostjo. Da bi računalništvo kot podpora poslovanju zmoglo slediti temu tempu, so se razvile nove tehnologije in koncepti. Članek nam kot odgovor na nove zahteve predstavi porazdeljeno računalništvo. Opozori nas na pravilno načrtovanje arhitekture sistema, ki bo deloval kot porazdeljen. Na koncu je opisana tehnologija CORBA in njena primerjava z COM+.

Abstract

With new technologies the way of making business is changing with unbelievable speed. In order for the computer science as a supporting pillar of business operations to follow it, new technologies and concepts have been developed. The article presents distributed object computing as a response to the new requirements. It emphasizes the correct planning system architecture, which will act as distributed. At the end of the article technology CORBA and its comparison with COM+ is described.

■■■

Poslovne aplikacije prihodnosti bodo tekle na različni, večinoma specializirani strojni opremi, ki bo sodelovala tako z drugo strojno opremo, kot tudi z obstoječimi računalniškimi sistemi, ki jih uporabljamo danes. Če želimo, da bodo sistemi, ki jih gradimo danes, ustregli zahtevam sodobnega poslovanja, morajo biti zgrajeni na porazdeljenih objektnih tehnologijah.

Razlog, zaradi katerega se mora spremeniti način gradnje sistemov, je predvsem spreminjanje poslovnega sveta, katerega podpirajo. Ne samo, da je potrebno med seboj povezati informacijske otoke znotraj podjetja, ampak se mora podjetje povezati tudi s strankami, dobavitelji, partnerji in celo konkurenco. Na ta način se ustvarjajo tako imenovane navidezne korporacije.

Edini način, ki zagotavlja uspešno delovanje navidezne korporacije, je informacijska tehnologija.

Porazdeljeno računalništvo

S pojmom *porazdeljeno računalništvo* (distributed object computing) označujemo združitev objektno orientiranih tehnologij in arhitekture odjemalec-strežnik. Vendar je porazdeljeno računalništvo več kot zgolj 'vsota' obeh. V porazdeljenem računalništvu so združene najboljše lastnosti obeh svetov: zmožnost predstavitve resničnega sveta objektov in način dela iz arhitekture odjemalec-strežnik.

Z razvojem porazdeljenega računalništva se v temeljih spreminja način delovanja poslovnih informacijskih sistemov. Še več: način, na katerega se razvija programsko opremo za poslovne namene, je spremenjen za vedno.

Porazdeljeno računalništvo je način dela, ki dopušča, da so objekti razpršeni po različnih računalniških omrežjih in jim omogoča, da skupaj delujejo kot celota. Za tovrstne aplikacije lahko uporabimo Sunov slogan - *omrežje je računalnik* (the network is the computer).

Pri takem načinu dela lahko odjemalci postanejo strežniki in obratno: strežniki postanejo odjemalci. To je povsem nepomembno, ker govorimo o objektih, ki sodelujejo: odjemalec zahteva storitev drugega objekta in strežnik zahtevi ustreže. Fizično se lahko oba nahajata na istem računalniku ali vsak na svojem koncu sveta in sta napisana vsak v svojem objektno orientiranem jeziku. Razlike ni, ker: *omrežje je računalnik.*

Kaj pa obstoječi računalniški sistemi, ki niso bili napisani za sodelovanje prek različnih mrež ali pa niso napisani v objektno orientiranem jeziku? Take sisteme lahko zapakiramo (wrap) in videti bodo kot povsem navaden objekt. *Pakiranje* imenujemo tehniko, ko naredimo objekten vmesnik za dostop do funkcionalnih delov obstoječih aplikacij. Ko je aplikacija zapakirana (wrapped), jo lahko enakovredno uporabljamo v okolju razpršenih objektov.

Pravilno izbrana arhitektura - ključ do porazdeljenega računalništva

Danes razvijalci, ob začetku dela na določenem projektu, navadno nimajo celovite sistemske arhitekture, s katero bi začeli graditi. To dejstvo postane največji problem takrat, ko želimo poslovno računalništvo povzdigniti iz nivoja posameznih aplikacij na nivo celostnih poslovnih rešitev.

Arhitektura je visoko nivojski opis organizacije sistema. Vedeti je treba, da arhitektura ni opis rešitve nekega določenega problema ali pa navodilo za uspešno oblikovanje sistema. Arhitektura mora nuditi informacijo o splošni zgradbi sistema. V tem smislu določa arhitektura odnose med komponentami sistema, vendar ne opisuje, kako so te komponente realizirane.

Uspešen razvoj aplikacij zahteva določitev arhitekture - načrt infrastrukture in tehnologij. Čeprav v računalništvu ne obstaja ena sama, vseobsežna arhitektura, je potrebno pri razvoju aplikacij nove generacije razgraditi dve arhitekturi: tehnično in informacijsko.

Tehnična arhitektura je načrt za sestavljanje tehnoloških rešitev. Določa, katera orodja in katere tehnike bodo uporabljene.

Informacijska arhitektura pa opiše vsebino, obnašanje in sodelovanje poslovnih objektov. Predpiše elemente, iz katerih bo sestavljena aplikacija.

Prednosti porazdeljenih objektov

Objektno orientiran razvoj zelo poenostavlja in pospešuje razvoj aplikacij. Porazdeljeni objektni modeli in orodja pa še razširijo in nadgradijo objektni pristop k razvoju. Objekti lahko živijo na različnih računalnikih na omrežju in vendar se zdi, kot da so del lokalne aplikacije. Porazdeljeni objekti prinašajo kar nekaj tehničnih prednosti:

1. Obstoječe rešitve lahko nadgradimo in jih kasneje ponovno uporabimo. Tehniko pakiranja (objektni vmesniki za obstoječe programe) lahko uporabimo na različnih delih računalniškega sistema in poenostavimo delo z njim. Na ta način ne samo da naredimo en program javno dostopen drugim objektom, ampak je lahko več programov skupaj videti kot en sam objekt. Pakiranje nam omogoča, da sredstva, vložena v razvoj nekega sistema, še zdaleč niso izgubljena, ampak lahko obstoječi sistem kasneje ponovno uporabimo na višjem nivoju.
2. Ker vsi objekti (lokalni in oddaljeni) med seboj komunicirajo na enak način, lahko programerji določijo, da se objekti izvajajo na računalnikih, ki najbolj ustrezajo zadani nalogi. Tako bi na primer objekt, ki mora izvajati zapletene izračune, tekel na močnejšem računalniku kot objekt, ki zahteva samo vnos nekih vrednosti. S tem pripomoremo tudi k boljši izkoriščenosti strojne opreme.
3. Ker odjemalec vidi vse objekte kot lokalne, odjemalec ne potrebuje informacije o vrsti računalnika. S tem je preseljevanje objektov med računalniki močno poenostavljeno.
4. Povezave znotraj sistema so veliko močnejše. Programsko in strojno opremo lahko na različnih mestih povežemo v enoten sistem - aplikacijo.

Cilj in namen porazdeljenega računalništva je jasen: izboljšati klasično tehnologijo odjemalec-strežnik, tako da bo učinkovita in fleksibilna ter manj zapletena.

Porazdeljeno računalništvo potrebuje standarde - CORBA

Brez standardov je skupno delo različnih sistemov praktično nemogoče. Object Management Group (OMG) je konzorcij, katerega sestavlja več kot 700 podjetij in organizacij (razvijalci in uporabniki), ustanovljen pa je bil leta 1989 z namenom, da določi standarde, potrebne za delovanje porazdeljenih objektov v heterogenih okoljih. Leta 1992 je OMG objavil standard CORBA (Common Object Request Broker Archtecture), ki določa usluge, ki jih mora nuditi ORB (Object Request Broker). Z objavo tega standarda so se odprla vrata v svet porazdeljenega računalništva.

Danes je dostopnih precej implementacij za več kot 30 osnov (platform). Nudi jih veliko proizvajalcev. Omenimo samo IBM, Visigenic, BEA in Iona. Vse implementacije pa znajo sodelovati med seboj .

CORBA določa tudi precej storitev (danes 24), dostopnih vsem objektom CORBA, znanih pod skupnim imenom *CORBAservices*. Te storitve skrbijo za ustvarjanje in uničevanje objektov, dostop do objektov preko njihovega imena, dinamično ustvarjanje povezav med njimi, objavljanje njihovih lastnosti, shranjevanje, koordinacijo transakcij, varno poslovanje ...

Lastnosti CORBA

CORBA je uporabna in pomembna predvsem zaradi dejstva, da določa vmesnik, ki lahko poveže vse obstoječe oblike vmesnikov odjemalec-strežnik. Povedano z drugimi besedami: CORBA definira objekte, ki med seboj povežejo že obstoječe aplikacije.

Hkrati nudi CORBA tudi dobro osnovo za prihodnost, ko se bodo aplikacije razvijale s povezovanjem obstoječih komponent. Za komponente pa lahko imamo tudi celotne programe, ki se jih s CORBA povezuje v večje sisteme. CORBA je bila načrtovana z namenom, da omogoči inteligentnim komponentam, da odkrivajo ena drugo in med seboj sodelujejo preko objektnega vodila (ORB). ORB je vmesnik, ki vzpostavi povezavo odjemalec-strežnik med različnimi objekti. Z uporabo ORB-a lahko odjemalec pokliče metodo na strežniku, kot da je njegova. Zato sploh ni pomembno, kje se

strežnik nahaja. Lahko je na istem računalniku ali pa na drugem koncu sveta. ORB prestreže klic in poišče objekt, ki lahko ustreže zahtevi. Posreduje mu parametre, pokliče pravo metodo in vrne rezultat. Tako odjemalcu ni potrebno vedeti, kje se objekt, ki je zahtevi ustregel, nahaja. Prav tako ga ne zanima programski jezik, v katerem je bil objekt napisan, niti operacijski sistem, niti strojna oprema, na kateri objekt živi.Vendar nudi CORBA več kot zgolj medsebojno povezovanje komponent. V njej je določeno veliko število storitev za ustvarjanje in uničevanje objektov, dostop do objektov preko njihovega imena, dinamično ustvarjanje povezav med njimi, objavljanje njihovih lastnosti, shranjevanje, koordinacijo transakcij...

CORBA se bo tudi v prihodnosti razvijala zelo hitro. V začetku leta 1998 bo objavljena CORBA 3.0 (sedanja verzija je 2.0). Na programskem nivoju se bo povečalo število že pripravljenih ogrodij za aplikacije (application framework). To so programi, v katerih je povsem realizirana poslovna logika, vizualni del (in morebitne razširitve) pa lahko vsako podjetje oblikuje po svojih potrebah. Razvoj programa je tako veliko hitrejši. Obstajajo že ogrodja za trgovino, bančništvo, elektronsko trgovanje, računovodstvo, transport, zdravstvo... Število področij je praktično neomejeno.

CORBA ali COM+?

Konzorcij OMG sestavljajo vsi vodilni proizvajalci programske opreme. Pomembna izjema je Microsoft. Formalno je sicer član OMG, vendar je njegova vloga zgolj opazovalna in se je odločil za svojo pot. Ker se Microsoft zaveda pomembnosti porazdeljenega računalništva, je razvil povsem svojo tehnologijo, znano pod imenom COM+ (prej imenovano DCOM).

Ker ima COM+ podoben namen kot CORBA, se veliko razvijalcev sprašuje, katero tehnologijo uporabiti.

Med njima obstaja nekaj razlik.

- Neodvisnost od sistema: COM+ je dosegljiv samo na Windows, medtem ko je CORBA dostopna na več kot tridesetih osnovah.
- Jezikovna neodvisnost: COM+ je zelo vezan na C++. Aplikacije si med seboj pošiljajo kazalce C-jevske oblike, ampak kazalcev ne podpirajo vsi programski jeziki (Java, COBOL), kar zna razvijalcu povzročiti precej preglavic. CORBA teh težav nima.
- Zrelost specifikacij: COM+ je v eni ali drugi obliki (od OLE do COM+) dosegljiv že dolgo, vendar se specifikacije neprestano spreminjajo, tako da jim je težko slediti. Na drugi strani nudi CORBA natančno določene in nespremenljive lastnosti.
- Uporabnost: COM+ je dosegljiv le uporabnikom namiznih računalnikov, medtem ko je CORBA dostopna tudi na poslovnih sistemih.
- Podpora Interneta: COM+ je vezan na okolje Windows in kot tak neprimeren za delo v heterogenem okolju Interneta.
- Varnost odjemalca: COM+ ne nudi nobenih zagotovil o lepem obnašanju objektov. COM+ vam zelo enostavno pobriše disk ali kaj podobnega. CORBA tega ne dopušča.
- Varnost povezav: COM+ nima nobenega orodja za ugotavljanje avtentičnosti medobjektnih povezav. CORBA jih nudi precej.

CORBA v praksi

Podjetje Ixtlan Consulting iz Ljubljane je na Internetu pripravilo demonstracijo arhitekture CORBA. Razvili so Internet plačilno banko. Njen namen je prikazati možnost uporabe arhitekture CORBA in porazdeljenih objektov v poslovnem svetu. Primer Internet plačilna banka poenostavlja delovne postopke plačil. Predstavljena rešitev prikazuje uporabo objektov CORBA pri reševanju komunikacije med različnimi informacijskimi sistemi. Najbolje, da si način delovanja Internet plačilne banke ogledamo na primeru.

Vzemimo, da mora podjetje Vijak d.o.o. podjetju Matica d.o.o. plačati račun za dobavo surovin.

Poglejmo, katere postopke je potrebno izvesti pri tem procesu.

- Računovodja podjetja Vijak d.o.o. vnese nakazilo v njihov računalniški program in izpiše plačilni nalog,
- kurir podjetja Vijak d.o.o. odnese plačilni nalog v plačilno banko,
- referent plačilne banke vnese plačilni nalog in nakaže denar podjetju Matica d.o.o.,
- kurir podjetja Matica d.o.o. se oglasi v plačilni banki, kjer mu posredujejo seznam nakazil na račun podjetja Matica d.o.o., med
- katerimi se nahaja tudi nakazilo podjetja Vijak d.o.o.,
- saldakontist podjetja Matica d.o.o. vnese nakazilo podjetja Vijak d.o.o. v svoj računovodski program in s tem je transakcija zaključena.

Kot vidimo, je potrebno podatke o istem nakazilu trikrat ročno vnesti na treh različnih mestih. Ne le, da lahko od začetka do konca transakcije preteče več dni, tudi nevarnost napake je velika.

Očitno je, da je možno celoten postopek izpeljati veliko bolje. Vprašanje je - kako. Odgovor se imenuje Internet plačilna banka. Podjetjem, ki uporabljajo storitve Internet plačilne banke, zadostuje, da vsako nakazilo vnesejo le v svoj informacijski sistem (v našem primeru bi bilo to v podjetju Vijak d.o.o.). Za vse nadaljne korake poskrbijo računalniški programi.

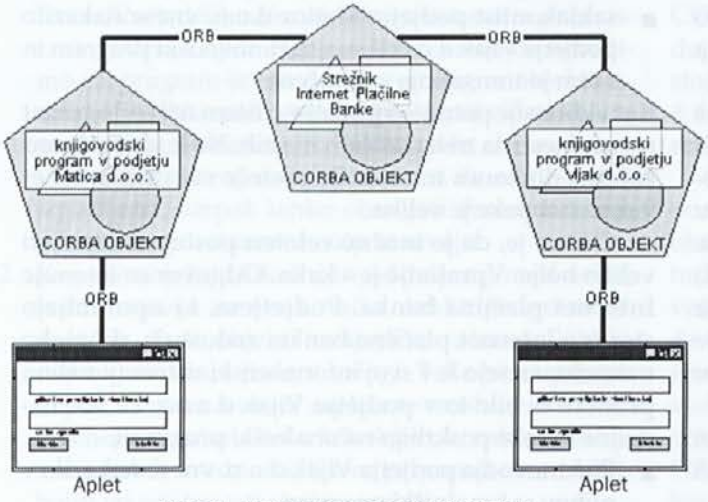
- Računovodja podjetja Vijak d.o.o. vnese nakazilo v njihov računalniški program,

- program podjetja Vijak d.o.o. zapiše transakcijo v svojo bazo in preko ORB-a obvesti Internet plačilno banko,
- bančni program v Internet plačilni banki preveri pravilnost nakazila in ga zapiše v podatkovno bazo,
- bančni program v Internet plačilni banki preko ORB-a obvesti program podjetja Matica d.o.o.. Transakcija je tako končana.

Enostavno, kajne? Tudi pridobili smo precej: čas, potreben za transakcijo, je močno skrajšan (zdaj se meri v sekundah namesto v dnevih), možnost napake pa nična.

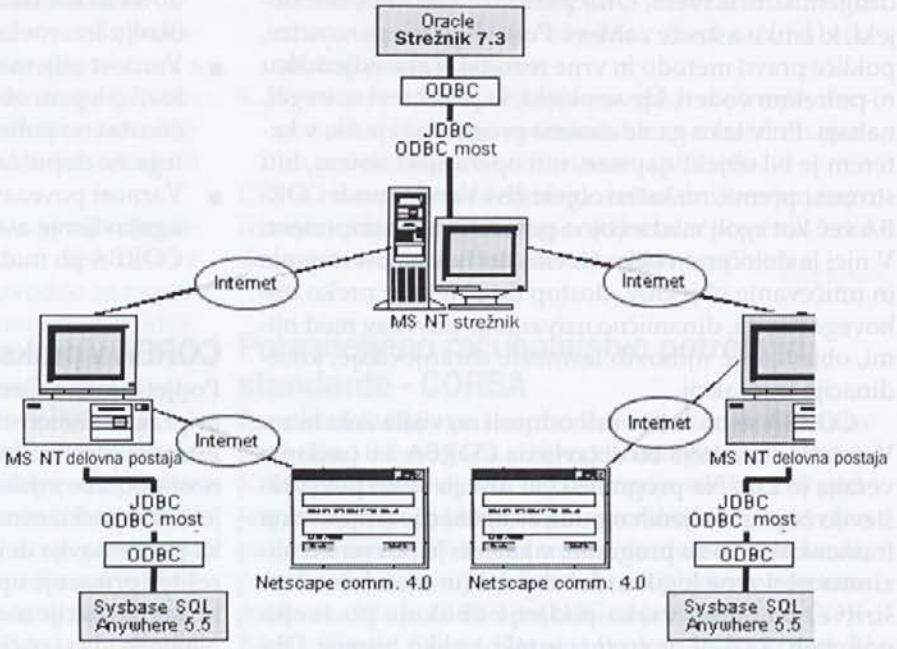
Na odjemalčevi strani (v Netscape Communicatorju) teče program, ki vpiše transakcijo v lokalno bazo in obvešča Internet plačilno banko. Ta program je obveščen tudi, kadar so na račun podjetja nakazana finančna sredstva. Seveda bi bili lahko na strani odjemalcev različni programi, napisani v različnih jezikih, ki bi tekli v različnih operacijskih sistemih in na različnih osnovah. Tudi baze v podjetjih so lahko različne. Na strežniku teče program, ki od podjetij sprejema zahteve za prenose sredstev na druge račune, jih odobrava ali zavrača (odvisno od stanja na računu), beleži prenose v svojo bazo in podjetja obvešča o nakazanih sredstvih.

V demonstraciji sta zaradi dosegljivosti prek spletnih strani programa, ki simulirata dogajanje v podjetjih Vijak d.o.o. in Matica d.o.o., napisana kot Java apleta. Predstavitvena verzija omogoča zgolj nakazovanje med podjetjema.



Slika 2: **Povezave med objekti CORBA**

Slika 1: **Arhitektura sistema Internet plačilne banke**



Ko na odjemalcu (aplet-u Java) vpišemo in potrdimo novo transakcijo, se ta posreduje strežniškemu objektu, ki jo shrani v lokalno podatkovno bazo. Prav tako tudi strežniški objekt Internet plačilne banke vse transakcije shranjuje v svojo podatkovno bazo. Za komunikacijo med programi Java in podatkovno bazo se uporablja most JDBC-ODBC (JDBC-ODBC bridge).

Strežnik Internet plačilne banke uporablja podatkovno bazo Oracle Server 7.3. V podjetjih Vijak d.o.o. in Matica d.o.o. pa za potrebe shranjevanja podatkov uporabljajo Sybase SQL AnyWhere 5.5.

Ko podjetje nakaže denar, se v bazo shranijo:

- Račun, na katerega se sredstva nakazujejo,
- znesek nakazila,
- datum in čas nakazila.

Ko strežnik podjetja posreduje nakazilo naprej Internet plačilni banki, ta v svojo podatkovno bazo shrani naslednje podatke:

- Račun, s katerega naj se sredstva odvzamejo,
- račun, na katerega naj se sredstva nakažejo,
- znesek nakazila,
- datum in čas nakazila,
- odobritev nakazila.

Banka odobri transakcijo, če je na računu podjetja, ki sredstva nakazuje, dovolj denarja, drugače se transakcija zavrne. V primeru, da je bila transakcija odobrena, se v tabeli, kjer so shranjena stanja na računih podjetij, na računih obeh podjetij ažurira stanje. Šele potem se o nakazilu obvesti ciljno podjetje.

Za komunikacijo med programi je uporabljen protokol IIOP (Inter-Internet Orb Protocol), ki se

uporablja za komunikacijo med objekti CORBA v TCP/IP (internet) omrežjih. Ta protokol uporabljata tudi apleta za komunikacijo z objekti CORBA, ki so nameščeni na naših strežnikih. Visigenic-ovi razredi Java (classes) za podporo protokolu IIOP obsegajo približno 440 KB datotek, ki se morajo prenesti na računalnik, kjer se izvajajo aplet-i Java, ki komunicirajo preko ORB-a. Ker bi bil prenos teh razredov zelo zamuden, je uporabljena že vgrajena podporo za protokol IIOP, ki jo ponuja Netscape Communicator 4.0. Na ta načina je zmanjšana velikost razredov, ki se morajo prenesti na odjemalca, na samo 8 KB.

Gradnja novega sistema, ki uporablja objekte CORBA, se začne z definicijo objektov. Za vsak program določimo, katere njegove metode so vidne na ORB-u.

Program, ki teče na Internet plačilni banki, ima vidne 3 metode:

- register - metoda, ki jo pokliče program v podjetju, da se prijavi na Internet plačilno banko,
- transfer - metoda, ki prenese denar z računa podjetja na račun drugega,
- getAmount - vrne trenutno stanje na računu podjetja.

Programi, ki tečejo po podjetjih potem preko ORB-a, kličejo te metode. CORBA omogoča popolno transparentnost postopka. Programov ne zanima, na kakšnem računalniku teče strežnik Internet plačilne banke, niti v katerem jeziku je napisan. Na drugi strani pa je strežniku Internet plačilne banke vseeno, kje tečejo programi, ki uporabljajo njegove metode.

Danes je na trgu dostopnih precej implementacij ORB-ov. Konkretni primer uporablja Visigenicov VisiBroker for Java 2.5. To za pisce programov, ki sodelujejo z Internet plačilno banko ni pomembno, ker standard CORBA od verzije 2.0 naprej določa, kako naj različne implementacije ORB-ov sodelujejo med seboj.

Na strežniku Internet plačilne banke mora biti poleg bančnega pognan še dodaten program - ORBeline Smart Agent. Kadar se hoče odjemalec povezati z nekim objektom, agent ta objekt najde in pokliče njegovo implementacijo. Objekti se prijavijo pri agentu, tako da jih lahko odjemalci najdejo in uporabijo. Ko je nek objekt 'uničen', ga agent odstrani iz spiska dostopnih objektov.

Tudi v podjetju, ki se želi povezati z Internet plačilno banko, mora teči agent. Podoben program (gatekeeper), ki ga moramo pognati v podjetju, poskrbi, da lahko apleti, ki uporabljajo VisiBroker, komunicirajo preko omrežja, pri tem pa še vedno ne kršijo varnostnih pravil.

Programi, ki kličejo objekte CORBA, niso s programerskega stališča čisto nič posebnega. Klicanje metod v drugih objektih se ne razlikuje od klicanja lokalnih metod. Edina razlika je, da moramo, preden metodo kličemo, vzpostaviti povezavo z objektom CORBA, v katerem obstaja implementacija za to metodo. Pravzaprav povezavo naredi agent in programu zato to ni potrebno skrbeti.

Ugotovitve

Ker se način poslovanja z zmagovitim pohodom Interneta in z njim povezanih tehnologij temeljito spreminja, mora informacijska podpora poslovnega procesa slediti tem trendom. Poslovno računalništvo se bo v prihodnosti zagotovo razvijalo v smeri porazdeljenega računalništva. Zato je potrebno pri načrtovanju sistemov, ki jih gradimo danes, imeti to v mislih. Pravilno postavljena arhitektura poslovnega sistema nam bo prihranila precej dela.

Tudi obstoječe aplikacije lahko z novimi tehnologijami povežemo v celovitejše sklope. Tako aplikacije zaživijo novo življenje in uporabnikom je dosegljivih več informacij, kar je danes odločilno pri preživetju na tržišču.

Tehnologija, ki vse to omogoča, je dovolj zrela. Vedno nove aplikacije, razvite v duhu porazdeljenega računalništva, znova in znova potrjujejo to dejstvo. V svetu obstaja že ogromno tovrstnih poslovnih sistemov. Vsak razvijalec bi se moral na začetku projekta seznaniti s ustreznimi tehnologijami in jih uporabiti na svojem projektu. Pri tem ni tako pomembno, za katero tehnologijo se odloči. Vsak bo izbiral glede na svoje potrebe. Če pa ne izbere nobene, se bo zaprl v svoj vrtiček in s tem sebe in svoje podjetje obsodil na informacijski propad. Tehnologije so, vi jih samo pravilno uporabite.

Literatura

- Peter Fingar, Dennis Read, Jim Stikeleather, "Next Generation Computing: Distributed Objects for Business", SIGS Books & Multimedia, New York, 1996
- Stikeleather, Jim, "Why Distributed Object Computing is Inevitable," Object Magazine, stran 35 (marec-april 1994)
- Sutherland, Dr. Jeffrey, "Distributed Object Architecture for IS Applications, Distributed Object Computing", SIGS Publications, 1995
- Object Management Group, Inc., "OMG Business Application Architecture," White Paper Draft, 1995
- Taylor, Dr. David A., "Business Engineering with Object Technology", John Wiley & Sons, 1995
- Cox, Brad J., "Object-Oriented Programming, An Evolutionary Approach", Addison-Wesley DeGeus, Arie, "Planning as Learning," Harvard Business Review, stran 74 (marec-april 1988)

◆

Marko Juvančič dela v podjetju Ixtlan Consulting, kjer je zadolžen za razvoj informacijskih rešitev, povezanih z javanskim računalništvom. Zaključuje študij na Univerzi v Ljubljani, Fakulteti za računalništvo in informatiko. V letu od 1996 do 1997 je sodeloval pri več projektih razvoja in uvajanja informacijskih sistemov.

◆