Neumann János Számítógép-tudományi Társaság

IV. ORSZÁGOS OBJEKTUMORIENTÁLT KONFERENCIA

Nyílt szoftverek, komponensek, alkalmazások az új évezredben

OO Technológia, Java, Linux, e-business, e-commerce

Budapest, 2000. március 21-24.

Fő média szponzor:

COMPUTERWORLD SZÁMÍTÁSTECHNIKA

Támogatók

IBM infOpen IQSII JAVA Microsoft ORACLE Sun microsystems We're the dot in .com SYSDATA TRIAD

Megnyitó, Plenáris

BME Központi épület
Díszterem
Bp., XI., Műegyetem rkp. 3.

Szekciók, Tutoriál

ELTE-BME Informatika épület
Termek: B.017, B.018, B.019
Bp.,XI., Pázmány Péter sétány 1/D.

Tartalomjegyzés

A Konferencia Program Bizottsága

Elnök: *Juhász István*

Tagok:
Dömölki Bálint
Gál László
Klotz Tamás
Kovács András
Papp Sándor
Péter László
Porkoláb Zoltán
Sugár Péter
Zsemlye Tamás

Köszöntő

Túl az Y2K-n, túl 2000. február 29-én, az informatikai világvégét elkerülve a magyar OO társadalom egy újabb (reményeim szerint) jelentős eseményhez érkezett. Immár negyedszer kerül megrendezésre ez az országos konferencia, amelynek alcíme olyan területeket nevez meg, amelyek meghatározzák napjaink és a közeljövő informatikájának a fejlődési irányait.

Az előadások témaköreit és az előadók által képviselt cégeket, intézményeket áttekintve úgy gondolom, hogy a konferencia egy jó keresztmetszetét adja a hazai OO (és azon túli!) világnak. A spektrum meglehetősen széles, az egyetemi kutatóhelyek eredményeitől az infokommunikáció területén meglevő alkalmazásokon át, az e-business, e-commerce megoldásokon keresztül, a szabványok, nyelvek, elvek, módszertanok, technológiák sokszínű világának napi helyzetével ismerkedhetünk meg.

Csak győzzünk eljutni minden bennünket érdeklő előadásra!

Gondolom a szünetekben lesz alkalom a szakmai és nem szakmai tapasztalatcserére, és jut idő majd a baráti beszélgetésekre is. Ez utóbbira különösen jó alkalmat kínál a szerda esti fogadás.

Remélem minden résztvevő talál érdeklődésének megfelelő, esetleg a napi munkájában felvetődő kérdésekre választ adó előadást vagy tutoriált. Az előadók pedig remélhetőleg azzal az érzéssel távoznak, hogy értő közönség előtt beszélhettek izgalmas és modern dolgokról, és esetleg új ismereteket közvetítettek.

Vágjunk hát bele!

Juhász István

Tervezői Fejlesztői (1) elsődleges prioritás, (2) másodlagos

A Neumann János Számítógép-tudományi Társaság

IV. ORSZÁGOS OBJEKTUMORIENTÁLT KONFERENCIÁJA

A KONFERENCIA PROGRAMJA

2000. március 21.

Tutoriálok

8.00	**Regisztráció**
9.00-13.00	**Párhuzamos tutoriálok**
	1. UML-alapú objektumorientált elemzés és tervezés BZ (2) Boros Péter, Quatrosoft
	2. Elosztott objektum technológia és komponens architektúrák (CORBA, Java, DCOM, COM, CCM, EJB) RL (2) Horváth Attila, IQSOFT
	3. XML Németh László, IQSOFT Markovits Péter, Oracle Hungary
13.00-14.00	**Ebéd**
15.00-19.00	**Párhuzamos tutoriálok**
	6. Elemzési tervezési minták BZ (1) Steingart Ferenc, ELTE
	7. D/COM, COM+ RL (1) Kovács Ferenc, BME
	8. Szoftverfejlesztés korszerű módszertan szerint fejlett életciklus támogató eszközök alkalmazásával Nick János, IQSOFT Makai Zsuzsanna, IQSOFT Boros Péter, Quatrosoft

2000. március 22.

8.00-9.00	**Regisztráció**		
9.00-9.30	**Megnyitó** Juhász István a Program Bizottság elnöke, DE		
9.30-11.00	**Java, XML and the future of e-business** Simon Phipps Chief e-business Evangelist, IBM Corporation		
11.00-11.30	**Szünet**		
11.30-13.00	**Building the Business Internet with Windows DNA 2000** Nic Merriman Windows DNA szoftvertervező mérnök, Microsoft		
	Ebéd		
	I. szekció	II. szekció	III. szekció
	OO fejlesztési technikák és tapasztalatok Szekcióvezető: Langer Tamás	**Java** Szekcióvezető: Zsemlye Tamás	**Infokommunikáció és OO technológia** Szekcióvezető: Wiener József
15.00-15.30	**Korszerű módszertant a korszerű eszközökhöz** *(szoftverfejlesztés, bjektumorientált, komponens alapú módszertan szerint)* Boros Péter, Quatrosoft BZ (1)	**Java az elektronikus kereskedelmeben** Tresch Ádám Profitrade90 Kft	**Az integrált TMN és CORBA, a fejlett hálózatmenedzselés eszközei** Papp Sándor MATÁV Rt.
15.30-16.00	**Egységesen, hatékonyan, egyszerűen!** *(fejlett életciklus támogató eszközök használata)* Boros Péter, Quatrosoft BZ (1)	**ORB, JDBC kapcsolatok Jbuilderrel** Vér György Borland Magyarország	**Példa az objektum- modellezés megvalósítására a MATÁV távközlő-hálózatában alkalmazott hálózatüzemeltető rendszerben** Kozábel Zsuzsa, MATÁV Rt.
16.00-16.30	**Objektumorientált elemzés és tervezés** *(egy nagy méretű projekt tapasztalatai)* Petrovics Péter, Unisys BZ (1)	**Jstuct** Gelle István Sysdata Kft.	**A TeMIP implementálása a MATÁV távközlő hálózatában** Vékássy Bence Compaq Magyarország Kft.
16.30-17.00	**Szünet**	**Szünet**	**Szünet**
17.00-17.30	**Koncepcionális tervezést támogató Keretrendszer** *Zsuffa Zsolt*, Zs. Zs. Bt.	**Valósidejű információmegosztás** Szatmáry Viktor	**Hálózat management a Westel 900-nál** Maradi István, Westel900
17.30-18.00	**A szoftver minősége kritikus az e-üzletben** *(webes alkalmazások – HTML, Java - funkcionális és teljesítmény tesztelése)* Szász Csaba, AlviCom Kft Nick János, IQSOFT	**CORBA és EBJ technológiák alkalmazása** Csizmadia Balázs	**WAP, a wireless Internet Technológia** Szűcs László NOKIA Hungary Kft.
18.00-18.30	**Qt objektumok Többplatformos felhasználói felülettel rendelkező alkalmazások írása C++ - ban** Németh Miklós, IQSOFT	**OO alkalmazások az innovációs korszakban – Az IBP Keretrendszerben** Nguyen Nhan Bao Integra Informatika Bt.	**AXE adatbázis kezelése Weben keresztül** Váradi István Ericsson Magyarország Kft.
19.00 -	**Fogadás**	**Fogadás**	**Fogadás**

2000. március 23.

9.00-9.30	**Regisztráció**		
9.30-11.00	**Programozási paradigmák és programozási nyelvek** Porkoláb Zoltán, ELTE BZ (1)		
11.00-11.30			
11.30-13.00	**Broadening the scope of UML** Olivier Roubine, Rational Software Corporation		
	I. szekció	**II. szekció**	**III. szekció**
	Java és a beágyazott rendszerek Szekcióvezető: Zsemlye Tamás	**E-business, e-commerce** Szekcióvezető: Sugár Péter	**Új programozási paradigmák** Szekcióvezető: Porkoláb Zoltán
15.00-15.30	**JavaCard megoldások** Magyari Péter, Schlumberger Magyarország	**e-business rendszerek integrálása és az OO technológiák** dr. Sugár Péter, IBM Magyarországi Kft	**Programozási paradigmák egymás mellett - Perl esettanulmány** Frohner Ákos, ELTE
15.30-16.00	**Jini technológia és alkalmazása metaszámítási rendszerekben** Juhász Zoltán, Veszprémi Egyetem Zsemlye Tamás,SUN Microsytems	**e-commerce, vízió és valóság** Olti Ferenc DTI Consulting	**Tervezési minták használata** Boros Péter QuattroSoft Kft BZ (1)
16.00-16.30	**A JavaSpaces felhasználása párhuzamos és elosztott objektumorientált programozásban** Juhász Zoltán, Veszprémi Egyetem	**e-üzleti alkalmazás fejlesztése Java Servlet és JSP technológiával** Nick János, Kelemen Sándor IQSOFT Rt.	**Mobil programozási rendszerek gyakorlati összehasonlítása** Gulyás László BZ (1) MTA SZTAKI BZ (1)
16.30-17.00	**Szünet**	**Szünet**	**Szünet**
17.00-17.30	**Beágyazott rendszerek UML modell alapú automatizált implementálása** Szász Csaba, AlviCom Kft	**Az IBM e-business megoldásai és a biztonság** Geist Éva IBM Magyarországi Kft.	**Objektumrelációs technológiák használata az Oracle-ben** Kis László, BZ (1) Allround Kft.
17.30-18.00	**ORB interoperabilitás és skálázhatóság vizsgálata** Werner Zsolt BME Ericsson Hungary ORB Interoperability Lab	**e-development: hatékony életciklus támogatás az e-business fejlesztésekben** Kovács András Makai Zsuzsa,IQSOFT Rt	**JAVA komponensek Oracle adatbázisban** Markovits Péter Oracle Hungary Kft
18.00-18.30	**Java Collections Framework benchmark-tesztelése – mikor káros a szinkronizáció** Werner Zsolt, BME	**XML és EDI – XML/EDI** BZ (1) Gergály Péter SZÁMALK	**Business Components for JAVA az e-business Oracle objektumai,** Molnár Balázs,Oracle Hungary Kft
18.30-19.00	**WAP technológiát alkalmazó e-commerce megoldások** Ráth Péter, Anemo Kft.	**BizTalk: XML alapú elektronikus kereskedelmi környezet** Gál László, Microsoft Magyarország Kft	**Konkurrens objektumrendszerek specifikacioja** Blum László Veszprémi Egyetem
19.00-19.30	**Szünet**	**Szünet**	**Szünet**
19.00-	**Panel** Moderátor: Hutter Ottó		

2000. március 24.

	I. szekció	II. szekció	III. szekció
	Debreceni műhely Szekcióvezető: Juhász István	**BME műhely** Szekcióvezető: Kondorosi Károly	 Szekcióvezető: Kelen András
9.30-10.00	**Chronos, minőség fogalom az Interneten** Noszály Csaba Debreceni Egyetem	**Mobil Agent prototípus CORBA alapon** Goldschmidt Balázs, BME	**Specifikáció (model) alapú tesztelés** Bányai Attila, TRIAD
10.00-10.30	**Szóra bírjuk Alice-t** Bátfai Norbert Debreceni Egyetem	**Kétirány CORBA/COM híd megvalósíthatósága** Kádár András, BME	**Szoftver életciklus követés ISO/MSZ 12207 szabvány szerint** Bertalan Gábor,TRIAD
10.30-11.00	**Alkalmazásfejlesztés dokumentumai** Veres Károly, Vég Csaba Logos Bt.	**Tervezési minták alkalmazása – vasúti szimuláció** dr.Frigó József, BME	**Bűnök és parancsolatok, avagy az OO technológia buktatói** Kelen András, TRIAD
11.00-11.30	**Szünet**	**Szünet**	**Szünet**
11.30-12.00	**Generikus programszerkezetek** Nagy István - Vég Csaba Debreceni Egyetem	**A CORBA oktatásának tapasztalatai** László Zoltán, BME	**A követelmény elemzés és követés** Szádeczky-Kardoss Szabolcs, TRIAD BZ (1)
12.00-12.30	**Cserélhető ("plug & play") adatkapcsolat és adattárolás** Espák Miklós, Nagy István, Vég Csaba Debreceni Egyetem	**A SYSTEM ARCHITECT 2001 integrált vállalati vizuális modellező eszköz** Szendrői Etelka BZ (1) Pécsi Tudományegyetem Pollack Mihály Műszaki Főiskolai Kar	**Forward engineering: Architektura központú fejlesztés, minta alapú kódgenerálás** Tóth Bálint, TRIAD
12.30-13.00	**A haszon nyílt forrásai - A "szabad szoftverek"** Magosányi Árpád, Matáv Rt		
13.00-	**Konferencia zárás**		

TÁMOGATÓK

Fő média szponzor:

IDG SZÁMÍTÁSTECHNIKA

IBM

INFOPEN

IQSOFT

JAVA SZÖVETSÉG

MICROSOFT

ORACLE

SUN MICROSYSTEMS

SYSDATA

TRIAD

A konferencia helyszíne

A konferencia programjaira az alábbi helyszíneken kerül sor:

Megnyitó, Plenáris ülések
BME Központi épület (K)
Díszterem
Bp., Műegyetem rkp. 3.

Szekciók, Tutoriálok
ELTE-BME Informatika épület (I)
Termek: B.017, B.018, B.019
Bp., Pázmány Péter sétány 1/D.

2000. március 21.	*Tutoriálok*

Időpont	*Program*	*Tutoriál előadó*	*Helyszín*
8:00 – 19:00	Regisztráció		I - főportánál
9:00 – 13:00	1. Tutoriál	Boros Péter	I- B.018
	2. Tutoriál	Horváth Attila	I- B.017
	3. Tutoriál	Németh László Markovits Péter	I- B.019

13:00 - 14:00	Ebéd		I- Aula

15:00 – 19:00	6. Tutoriál	Steingart Ferenc	I- B.017
	7. Tutoriál	Kovács Ferenc	I- B.019
	8. Tutoriál	Nick János Makai Zsuzsa Boros Péter	I-B.018

2000. március 22.	*Konferencia: Megnyitó, Plenáris előadások, Szekciók*

Időpont	*Program*		*Helyszín*
8:00 – 13:00	Regisztráció		K Aula
9:00 – 13:00	Megnyitó Plenáris előadások		K Díszterem
13:00 – 14:00	Ebéd		K épület
14:00 – 19:00	Regisztráció		I – főportánál
15:00 – 18:30	*Szekciók*	*Szekcióvezető*	
	I. Szekció	Langer Tamás	I – B.018
	II. Szekció	Zsemlye Tamás	I – B.017
	III. Szekció	Wiener József	I – B.019
20:00	Fogadás		K épület

2000. március 23.	*Konferencia: Plenáris előadások, Szekciók*

Időpont	*Program*		*Helyszín*
9:00 – 13:00	Regisztráció		K Aula
9:30 – 13:00	Plenáris előadások		K Díszterem
13:00 – 14:00	Ebéd		K épület
14:00 – 19:00	Regisztráció		I – főportánál
15:00 – 19:00	*Szekciók*	***Szekcióvezető***	
	I. Szekció	Zsemlye Tamás	I – B.018
	II. Szekció	Sugár Péter	I – B.017
	III. Szekció	Porkoláb Zoltán	I – B.019
19:30	Panel	Moderátor: Hutter Ottó	I- Java klub (ELTE oldal, V. emelet, hátsó lépcső)

2000. március 24.	*Konferencia: Szekciók*

Időpont	*Program*		*Helyszín*
8:00 – 13:00	Regisztráció		I – főportánál
9:30 – 13:00	*Szekciók*	***Szekcióvezető***	
	I. Szekció	Juhász István	I – B.018
	II. Szekció	Kondorosi Károly	I – B.017
	III. Szekció	Kelen Andás	I – B.019

ELŐADÁSVÁZLATOK

Ebben a fejezetben a szerzők által beküldött előadásvázlatokat eredeti tartalommal, változtatás nélkül tesszük közzé. A vázlatok közreadásával a konferencián elhangzó előadások szelekcióját kívánjuk elősegíteni.

Bányai Attlia: Specifikáció (model) alapú tesztelés

Az elõadás meghatározza a tesztelés szerepét az életciklusban és néhány ötlettel szolgál a tesztelésre forditandó idõ csökkentésére és a tesztelés minõségének javítására. Ezután a specifikáció alapú teszt eset generálás megvalósítási lehetõségeirõl és ezek elõnyeirõl lesz szó.
Az elõadásban zárasaként az automatikus teszt végrehajtás lehetõségeit fogjuk felvázolni.

Bátfai Norbert (Antal Péter, Horváth Balázs, Jeszenszky Péter, Nagy László Zsolt, Noszály Csaba): Szóra bírjuk Alice-t

Előadásunkban bemutatjuk a 2000 évi Loebner díjas ALICE chat robotot és a hozzá fejlesztett audió interfészt.
A témát esettanulmányként dolgozzuk fel, a tapasztalatokat a HUMAN INTERFACE projektünkben kívánjuk felhasználni.
Rendszerünk alapja Richard Wallace ez évben Loebner díjat nyert. AIML-t feldolgozó. JAVA-ban implementált Alice nevű programja. Az 1990-ben alapított díjat az elnyeréséért évente versenybe szálló számítógépes programok közül a Turing teszten legjobban teljesítő rendszer kapja.
Alice GNU GPL engedéllyel terjesztett forrását fejlesztjük az IBM ViaVoice technológiáján alapuló, beszédfelismerést és szintetizálást Speech for Java eszközeit felhasználva.
Alice szóra bírását videóanyagon mutatjuk be.

Bao, Nguyen Nhan: OO alkalmazások az innovációs korszakban – Az IBP Keretrendszer

- Célok
  - Korszerű és célszerű "fejlesztési technológia" kialakítása
  - Integrált Banki Termék előállítása
- Módszertan és eszközök
  - Objektum Orientált módszertan, OMT/UML
  - Software through Picture (StP)
  - Forté Application Developer
  - SQL (Oracle 8)
- *Választott eszközök I.* : Forté
  - Fejlesztés, telepítés, adminisztrálás
  - Transactional Object Oriented Language (TOOL): 4GL nyelv
  - Elosztott alkalmazások, drag-and-drop akár futási időben
  - Terhelés kiegyenlítés (load-balancing), hibatűrés (failover), manuálisan vagy automatikusan
  - Gartner Group értékelése

•*Választott eszközök I* : StP
  - Vizuális CASE eszköz
  - OO és Relációs modellezést támogatja
  - Széleskörű átjárhatóság más eszközökbe: fejlesztés, dokumentálás
  - Rugalmasan testreszabható: StP-QRL script, StP-API
  - F-16 tervezése

•SQL alapú DBMS
  - Oracle 8: piacvezető adatbáziskezelő
  - Szabványos SQL: gyártótól való függetlenség

Háromrétegű, elosztott rendszermodell

•Logikai és fizikai rétegek
•Nagyobb teljesítmény érhető el
•Terhelés kiegyenlítés, hibatűrés...

Végfelhasználói elvárások kielégítése (I.) Végfelhasználói elvárások kielégítése (II.)

Rendszermodell-implementáció és perzisztencia

•Igény
  - Vizuális tervből lehessen információt átvinni implementációba

Megoldásunk

Alternatíva I.: "Kétlépcsős" koncepció

•Előny
  - Egyszerű gondolat

•Hátrány
  - Külön IBP repository-t kell létrehozni
  - OO és Relációs modellezés eszközben és időben távol kerül egymástól

Alternatíva II. : "Háromszöges" koncepció

•Előny
  - Nem kell külön IBP repository
  - OO és Relációs tervezés StP-OMT és StP-IM modulokban

•Hátrány
  - Bonyolultabb StP scriptek

Fejlesztés: UML és IM közötti átjárás

•Átjárás problémái
  - Információ veszteség
  - Adattípus
  - Automatizálás

•Implementálás
  - Testreszabás
  - GBO_Domains
  - StP scriptek

Átjárás Forté-be

•Osztály-diagram leképzendő elemei
  - Adattartalom: osztályok, attribútumok
  - Viselkedés: metódusok, paraméterek
  - Kapcsolatok: ismeretség, tartalmazás, öröklődés
  - Kapcsolatokból adódó implicit viselkedés

•Nehézségek
  - Nincsenek helyből felkínált szolgáltatások, mint adatbáziskezelőben: keresés, szűrés...
  - TOOL és SQL kommunikációja: objektum és adatrekord közös életciklusa

Fejlesztés: Forté és IM közötti átjárás

•Konzisztencia probléma

•Megoldás

- Konvenció alkalmazása (StP konvenció, Forté konvenció, név konvenció)
- GBO_Domains felhasználása

Összefoglalás

•Fejlesztési folyamat fölépései bemutatása

•Átjárás alternatív koncepciói (Kétlépcsős, Háromszöges)

•Átjárás munkafolyamata

•OO-Relációs leképzéshez testreszabások StP-ben

•OO-Relációs leképzést végző StP script

•OO modell-Forté TOOL leképzés szabályrendszere

•OO modell-Forté TOOL leképzést végző script

Bertalan Gábor: **Szoftver életciklus követés ISO/MSZ12207 szabvány szerint**

Az előadás az ISO/MSZ12207 szabványról, alkalmazásáról és implementciójáról szól.A szabvány rövid bemutatása után megvizsgáljuk a szabvány alapú fejlesztési folyamat egyes lépéseit, a szabvány által támasztott követelményeket és a megvalósítás lehetőségeit.Végül összefoglajuk a szabvány szerinti fejlesztés elönyeit.

Blum László - Kozma László: **Konkurens objektumrendszerek specifikációja**

Előadásunkban egy rövid bevezetőt szeretnénk adni a konkurens objektum-orientált rendszerek temporális logika segítségével történő specifikációjáról. Az objektum-elvű paradigmát alkalmazva konkurens világra megbízhatóbb és összetettebb alkalmazások megírására nyílik módunk. Azonban a konkurencia bevezetésével nem várt eredményeket kaphatunk már a specifikációs szinten is. Ezeket összefoglalóan öröklődési anomáliáknak nevezzük. Előadásomban két eljárás mutatok be a fentebb említett rendszerek specifikációjára. Az egyik Lamport TLA+ specifikációs nyelve a másik az általunk kifejlesztett szinkronizáció-specifikációs módszer: a szinkronizációs halmazok. Specifikációs eljárásunk támogatja a nyílt rendszereket és a széles körben elterjedt reflektív aktor modelt használja.

Boros Péter: **Egységesen, Hatékonyan, Egyszerűen Fejlett életciklus támogató eszközök használata**

Témáink

- I. Eszközök használata
- II. e-Business, új kihívások
- III. Teljes életciklus támogatás
- IV. Igénykezelés, Requisite Pro *
- V. Elemzés - tervezés, Rational Rose *
- VI. Dokumentáció, SoDA *
- VII. Változás kezelés + konfiguráció kezelés
- VIII. Ötletek, konkrét felhasználások

I. **Eszközök használata**
Jó elmélet
Kiváló technikák
Jó szakemberek
„Rossz" eszköz
Nehézkes projekt menet,
alkalmanként sikertelenség.

- Eszköz használat előnyei
  - Eszközhöz köthető technológiák
  - Közös „nyelv" a fejlesztők között
  - Az egyes célterületre készült eszközök együtt működése!
  - Ha jó a felülete, akkor célprogramokkal lehet bővíteni.

Kellő iparági támogatottság, de legalább nyílt legyen.

II. e-Business, új kihívások

- e-Businessben rejlő lehetőségek
  - rövidesen minden megoldás alapja lehet
  - hosszú időre szóló megoldások
  - informatikai megoldások melletti elkötelezettség
- Elvárások
  - gyakorlati felkészülés (technológiák stabilizálása)
  - szoftver rendszerek architektúrális felkészítése (3tier, distributed)
  - case eszközök felkészültsége (elosztott fejlesztés)

Ehhez a szoftver készítésnek is fejlődnie kell.

Csizmazia Balázs: **A CORBA és az EJB technológiák alkalmazása ügyviteli szempontból kritikus alkalmazások Internetre vitelére**

Az utóbbi években az Internet és a rá épülő elektronikus (többnyire kereskedelmi célú) alkalmazások térhódítása még a területet jól ismerő piaci elemzőket is meglepte. A hagyományos kereskedelemtől az ilyen területekre átpártolt piaci résztvevők arányát a legoptimistább piaci elemzők is rendre alábecsülik.
A vezető számítástechnikai cégek a kétezres évszámváltás körül kialakult problémák megoldódásával az ilyen céllal lérehozott, és az évszámváltás bekövetkezte óta felszabadult szellemi kapacitásaikat és tárgyi eszközeiket más célokra kezdik használni: alkalmazásaik Internetre telepítésére. Az áttérést - lehetőleg lépésenként - a már meglevő alkalmazások minél nagyobb mértékű újrafelhasználásával kell megoldani. Ezen kívül szükséges (és hasznos) lehet már bevált, de a jövő kihívásait várhatóan kielégíteni már nem képes - a cég ügyvitele szempontjából azonban kritikus - alkalmazások újakra cserélése. Míg ez utóbbi megoldás sok alkalmazásnál valószínűleg elkerülhetetlen lesz, addig ennek az áttérési módnak a tényleges költségei rendkívül nagyok, és a legtöbb cég nehezen szánja rá magát a már évek, esetleg évtizedek óta működő olyan alkalmazások lecserélésére, amelyek korábbi sikereikhez nagymértékben hozzájárultak. Egy másik egyre inkább jelentkező igény az alkalmazások grafikus kezelői felülettel történő ellátása. Ezek az igények sok területen túlzóak, és feleslegesek, viszont sokan a piacon maradás egyik szükséges feltételének

tekintik.
A Java 1.0 technológia megjelenésekor sokan próbálkoztak több-kevesebb sikerrel alkalmazások Internetre vitelével ezen eszköz segítségével, de a tapasztalatok szerint a technológia a megjelenésekor komolyabb vevői igények kielégítésére még alkalmatlan volt - részben egy elosztott infrastruktúra (és adatbáziskapcsolat), részben pedig egy, a webböngészőknél megszokott grafikus felületeknél komolyabb grafikus eszköz hiánya miatt. Erre az áttérési problémára egy kiváló megoldásként ígérkezik az OMG CORBA technológiája mint elosztott Internet-alapú infratruktúra és a Sun Java 1.2 technológiája mint az ezt kiegészítő grafikus és alkalmazás-logikai elemeket is biztosító hordozható alkalmazás-architektúra. Ezek a szabványos technológiák lehetővé teszik például már működő COBOL nyelven készült alkalmazások Internetre vitelét viszonylag kis befektetéssel. Ez az eszközkészlet lehetővé teszi az IDL/Java és az IDL/COBOL leképezési szabályok alkalmazásával és a meglevő JDBC-kapcsolattal a lépésenkénti áttérés megszervezését, ahol az első lépés általában az alkalmazás grafikus felhasználói felületének a webre történő átvitele, majd ezt követi az alkalmazás tényleges üzleti logikáját megvalósító kódrészek átírása. A folyamatos működést pedig az IDL interfészek változatlansága segíti. Mivel a legtöbb alkalmazás nem egyetlen komponensből áll, ezért ezeknek a biztonságos együttműködését is meg kell szervezni. Erre használható az Enterprise Java Beans (EJB) technológia. Ezzel megvalósíthatók meglevő alkalmazások biztonságos, tranzakció-alapú működtetése több hasznos objektumszolgáltatással együtt. Ez az eszköz is felhasználható lenne a CORBA komponensmodellel való kompatibilitása miatt COBOL programok Internetre vitelére, de a CORBA oldalán még a tényleges implementációk hiánya gondot okozhat.

***Espák Miklós - Nagy István - Vég Csaba :* Cserélhető („plug & play") adatkapcsolat és adattárolás**

A szoftverfejlesztés menete három egymástól jól elkülöníthető lépésből áll. A fejlesztést a feladat *elemzés*ével kezdjük, mely célja az alkalmazásnak kizárólag a szakterület fogalmaival történő pontos leírása. A *tervezési lépés* célja az alkalmazás informatikai fogalmakkal történő leírása. Ezen lépés során hozzuk meg azokat a döntéseket, melyek a későbbi megvalósítás menetét határozzák meg, ideértve a programozási nyelv kiválasztásától a szoftver szerkezetének megtervezését is. A harmadik, *megvalósítási lépés*ben a tervezés folyamán meghatározott technológiák segítségével implementáljuk az alkalmazást.
Az elemzési lépés megvalósítása során egy szakterületi modellt hozunk létre, míg a tervezési lépés eredményeként tervezési elemekkel bővítjük ki a szakterületi modellünket. Az ideális az, ha a szakterületi modell teljesen elkülönül a tervezési elemektől. Ez számos előnnyel jár. Az alkalmazás szerkezete áttekinthetőbbé válik, így az elemzést, illetve tervezést végző szakember a saját területét érintő osztályokkal külön tud foglalkozni. A tervezési döntések során meghatározott implementációs elemek a szakterületi modelltől függetlenül lesznek megvalósítva, így maga az implementáció cserélhetővé válik. Ezt többféleképpen is hasznosíthatjuk a fejlesztés során.
Egy ilyen módon megvalósított, már működő rendszerben az adatok tárolását megvalósító részt egyszerűen le lehet cserélni, hiszen az független az alkalmazás többi részétől. Amennyiben több

adatforrást is implementálunk, a rendszer indításakor megadhatjuk a számunkra szükségeset, sőt az egyes adatelemek akár különböző adatforrásból is származhatnak.
Egy konkrét példán keresztül olyan osztályrendszert alakítottunk ki, amellyel a szakterületi modellben használt kapcsolódási viszonyokat (mint osztályokat) a szakterületi osztályoktól elkülönülten tudjuk kezelni.
A szakterületi objektumok elérése és azokon a strukturális műveletek végrehajtása interfészeken keresztül történik. Egyetlen szakterületi objektum elérése a DATA interfészen keresztül történik. Az elérés többféleképpen is implementálható, attól függően, hogy milyen adattárolási, illetve adatelérési módot szeretnénk megvalósítani. Ilyen módon valósítható meg az, hogy a szakterületi objektumokat egymástól teljesen különböző tárolás esetén is transzparens módon tudjuk elérni. Az alkalmazás egyes helyein szükségünk lehet adott szakterületi osztályhoz tartózó több objektumra is. Ezt úgynevezett LIST-ek segítségével lehet kezelni. Az egy 1:N asszociációban szereplő objektumok az 1 számosságú szerep esetén a DATA, az N számosságú szerep esetén pedig LINK interfészen keresztül érhető el, mely utóbbi a LIST kiterjesztettje.
Az interfészeken keresztül történő adatkezeléssel a konkrét adatkapcsolat és a tárolási mód cserélhetővé válik. Mivel a fent említett leképezések csupán absztrakt leírásai a kapcsolódási viszonyoknak, ezért azoknak a (tervezés folyamán kiválasztott) konkrét implementációs technikánál történő megvalósulását is meg kell adni. Ezekhez a leképezésekhez többfajta implementációs technikát is kidolgoztunk: az adatelérés megvalósítható RMI vagy CORBA segítségével, míg a végső adattárolás egyaránt történhet JDBC-vel, vagy szerializációval. A cserélhető architektúra lévén egyaránt kínálkozik lehetőség arra, hogy az alkalmazásból közvetlenül a JDBC-s megvalósítást használjuk, és arra, hogy pl. egy CORBA implementációt használjunk, amely pedig akár JDBC-s, akár szerializációs implementációt használ. Ezek a kiszolgálók így tetszőleges módon összefűzhetők, csupán az szükséges, hogy a lánc végén egy a tárolást közvetlenül megvalósítani képes kiszolgáló álljon.

Frigó József: Tervezési minták alkalmazása - vasúti szimuláció

A RailCAD vasúti szimulációs projekt öt évnyi tapasztalatszerzési lehetőséget biztosított a tervezési mintákkal kapcsolatban, az alapmű megjelenésétől kezdve napjainkig. A projekt során értékes tapasztalatok gyűltek össze számos tervezési minta használatával kapcsolatban, beleértve az elkészült rendszer karbantartása során jelentkező előnyöket is.

A szimuláció központi része a vasúti biztosítóberendezések működésének oktatási célú szimulációja. A különböző berendezéstípusok működésének megadásához egy speciális leíró nyelvet definiáltunk, amelyet a Builder tervezési minta alkalmazásával olvashatunk be és az Interpereter minta szerint működik. A program részleteinek tervezését és programozását az Iterator, Proxy, Template Method és az Observer minták alkalmazása tette kényelmessé. A rendszer bővítése során a State és "Chain of Responsibility" tervezési mintákat a vonatszimulátorban alkalmaztuk sikerrel.

Végül egy érdekes tapasztalat a tervezési minták hiányával kapcsolatban: a Singleton minta használatát elvetettük - és sajnos tapasztalnunk kellett mindazokat a jelenségeket, amelyek elkerülésére ezt a tervezési mintát kitalálták.

Feladat: oktatás

Ki használja?

- forgalmi szolgálattevő
- mozdonyvezető
- oktató
- tanfolyam-tervező

Környezet

- Linux

- TCP/IP
- tcl/t

Felépítés
- adatbázis-szerver
- állomás-tervező
- állomás-megjelenítő
- vágányút-tervező

Látvány
Vasúti biztosítóberendezés
Pálya elemeihez tartozó logika
- váltó
- jelző
- szigetelt szakasz, behatási pont
- sorompó

Vágányutakhoz tartozó logika
- vonat-vágányút
- tolató vágányút
- térközök
- összetett vágányutak

Változáskezelés és megfigyelők
Logikai specifikáció
- objektumok és változóik
- táblázatok
- értékadás-sorozat
- késleltetés

Működés
- objektumok önállóan
- változásokra reagálnak
- változáskezelés Változáskezelés – tömörítve

probléma
- a megfigyelési hálózat bonyolult
- állomás méretével hatványozottan nő

megoldás
- ismétlődő részek keresése megfigyelési gráf indirekt tárolása
- bizber-szimulátor
- bizber-hibakereső
- menetrendi tervező
- menetrendi vezérlő
- vonatszimulátor
- ETCS-MMI-szimulátor

Gyűjtemények
- STL
- saját gyűjtemények

Iterátor
Az ottfelejtett iterátor esete
Mire jó a bizbernyelv?
- bizberszimuláció
- vágányút-keresés
- menetrend
- vonat-állapot

Hogyan vezessük a mozdonyt?
Mozdonyvezető állapotai
- áll
- tolató
- normál vonat

Infrastruktúra - remeték
- Mi az a remete?
- Miért nem használtuk?
- Miért kellett volna mégis?

Hogyan lehetne most bevezetni?
Infrastruktúra – eseménykezelés
Eseménykezelő keretrendszer
- események küldése
- események figyelése időfüggő feldolgozá

Gergály Péter: **XML és EDI – XML/EDI**

Egyre több helyen, egyre többet lehet hallani, olvasni a Web új nyelvéről, az XML-ről. Az XML nem csak egy új nyelve a Web-nek, hanem ennél lényegesebb széles kihatása van. Várhatóan az elektronikus kereskedelmi rendszerek alapvető technológiájává válik az XML. Ez a változás az EDI rendszereket is érinteni fogja.
Előadásomban a miért kell az EDI rendszereknek változni, hogyan fog hatni az XML, milyenek az XML/EDI rendszerek kérdésekre szeretnék választ adni. Nem utolsósorban kitérek arra is, hogy hogyan jelenik meg az OO szemlélet az EDI rendszerek területén.

Goldschmidt Balázs - Kondorosi Károly - László Zoltán: **Mobil Agent prototípus CORBA alapon**

1) A mobile agentek megjelenése
   A mobile agent-ek iránti igény már a kód migrálása kapcsán felmerült, különös tekintettel a dinamikus terheléselosztásra. Az objektum orientált szemlélet, ezen belül is a Java technológia és a middleware fejlődése teremtette meg a biztonságos hátteret a továbblépéshez, az autonóm ügynökök megvalósításához.
2) A jelenlegi mobile agent rendszerek áttekintése
   Ez a rész röviden összefoglal néhány megvalósított és elérhető mobil agent rendszert (Voyager, Ajanta, Tacoma, Aglets, Grasshoppers) valamint kitér a MASIF-ra.
3) A prototípus célja
   A megvalósított rendszereken végzett kisérletek eredményei alapján célszerűnek tűnt egy agent prototípus elkészítése, elsődlegesen abból a célból, hogy reálisan értékelhetők legyenek a fent említett rendszerek. Ez a rész specifikálja a prototípust.
4) A felhasznált módszerek, eszközök
   A demonstrációs célú modell csak a JDK 1.2.-re és a CORBA-ra épül. A kód és az állapot mozgatása szerializálás révén történik. A néhány osztályból álló alaprendszert mindössze 300 Java forrással valósítottuk meg.
5) Eredmények
   Az általunk definiált Agent osztályból leszármaztatott objektum símán migrál egyik futtatási környezetből a másikba több gép között is. Az ügynök szolgáltatása, hogy tetszőleges (szerializálható) objektumot képes magával vinni.
6) Továbbfejlesztési lehetőségek
   Vizsgálni kívánjuk az ügynök elérhetőségét és annak különböző modelljeit, az átviteli közeg biztonságát és az ügynök és ügynökségek jogosultságainak kezelését.

Gulyás László - Láng László - Nagy Ferenc - Nagy Péter - Tejfel Máté - Tenczer Róbert: **Mobil programozási rendszerek gyakorlati összehasonlítása**

Az elosztott rendszerek egyik új implementációs technikája a mobil számítás (mobil kód). E megközelítés szerint az alkalmazás komponensei és az azokat végrehajtó hardver közötti kötés dinamikus, azaz a kód a végrehajtás során a végrehajtó gépek között vándorolhat. Bár a mobil számítás "eredendő szükségszerűségét" demonstráló alkalmazás, vagy az azt mindenképpen igénylő feladat (*killer application*) valószínűleg nem létezik, a koncepció hasznos eszközt adhat az elosztott alkalmazásokat fejlesztők kezébe. (Egyiket a sok közül.)B

E felismerést támasztja alá az elérhető mobil fejlesztőeszközök, programozási környezetek nagy száma is. Az elmúlt fél évben az ELTE TTK Általános Számítástudományi Tanszékének szemináriumán ezek közül a legfontosabbnak, legérdekesebbnek tűnő, ingyenesen elérhető rendszereket vettük górcső alá. Vizsgálatunk nem csupán az elvi megoldások, illetve kényelmi szolgáltatások összehasonlítására terjedt ki, hanem egy leegyszerűsített prototípus-alkalmazás megvalósítására is minden rendszerben.
Az összehasonlítás eszközeként használt alkalmazás egy minimalizált "elektronikus vásárlórendszer" (*electronic shopping*) volt, melyben a különböző "eladók kínálatából", a felhasználó kívánságára egy mobil ágens válogatott. Mivel az "eladók szervereit" ugyancsak mobil ágensekkel implementáltuk, a prototípus alkalmat adott a mobil számítás kapcsán gyakran emlegetett *dinamikus szoftverfrissítés / bővíthető szerver* koncepciójának szemléltetésére is.
Előadásunkban e szemináriumi vizsgálódás eredményeit tekintjük át, a munka jellegéből következően elsősorban programozói szemszögből, a technikai részletekre koncentrálva.

***Kádár András - Kondorosi Károly :* Kétirányú CORBA/COM híd megvalósíthatósága**

Az előadás egy kétirányú CORBA-COM híd megvalósíthatóságának vizsgálatáról, az ezzel kapcsolatos kísérletekről, valamint az eredményekre alapozott fejlesztés célkitűzéseiről és kezdeti eredményeiről számol be.
A CORBA és a COM

Rövid bemutatás az átjárhatóság szempontjából kritikus különbségek hangsúlyozásával.

Az átjáró tervezésének legfontosabb problémái

- univerzalitás
- egyszerű és összetett adattípusok megfeleltetése
- különböző metódushívási módozatok megfeleltetése
- kivételkezelés eltéréseinek (hiányának) kezelése
- többszörös öröklés kezelése
- interfészkönyvtárak
- névkezelés
- perzisztencia
- a híd elhelyezése az architektúrában
- az implementációs platform és nyelv megválasztása

Kísérleti eredmények

COM szerver elérése CORBA kliens által Windows NT platformon implementált hídon keresztül.

Fejlesztés a vizsgálati eredmények alapján

Célkitűzés: híd megvalósítása egy - az irányítástechnikában elterjedt - COM interfészhez, az OPC-hez (OLE for Process Control).

A létrehozandó híd jellemzői: CORBA objektumként jeleníti meg az OPC szervereket, kezeli azok életciklusát, továbbítja a metódushívásokat, és lehetővé teszi a visszahívásos (callback) mechanizmus használatát a változásfigyelés megvalósításához.

A fejlesztés első eredményeinek bemutatása:

Egyszerű híd, amely egyelőre az OPC interfészeknek csak egy részhalmazát implementálja és egy CORBA kliens, amely egy kész OPC szerver szolgáltatásait használja és adatait éri el a hídon keresztül, ám a híd ismerete nélkül.

Kelen András: **Bűnök és parancsolatok, avagy az OO technológia buktatói**

Az előadás igyekszik több projektben résztvevő személy tapasztalait összefoglalni azok számára, akik azt hiszik, hogy az OO minden gondjukat megoldja, ill. azt gondolják, hogy ha valami nem SSADM alapu akkor azzal nemszabad foglalkozni, de leginkább azok számára, akik szeretnének tanulni a mások bajaiból.

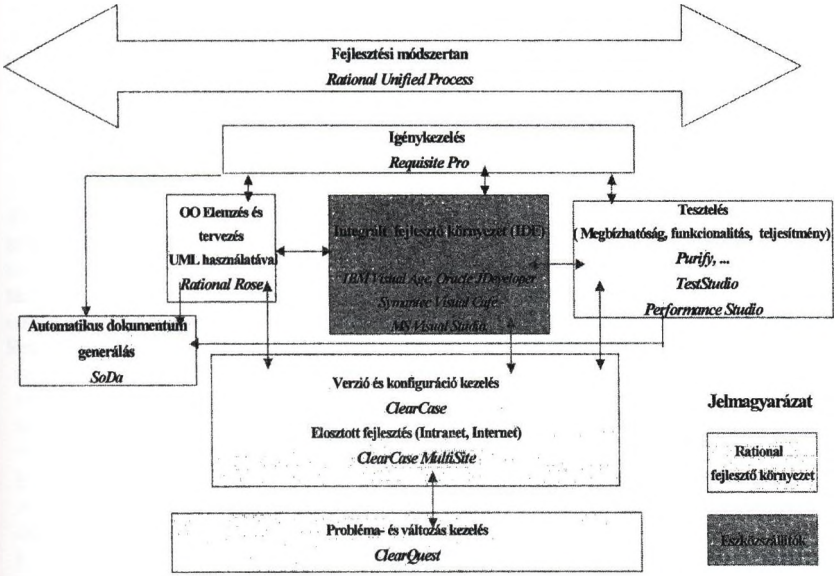
Kovács András – Makai Zsuzsanna: **e-development: hatékony életciklus támogatás az e-business fejlesztésekben**

Az Internet szinte minden iparágban gyors változások hajtóereje. Az Internet alapú gazdaság erősebben függ a szotvertől mint eddig a gazdaság bármikor: hatékony és megbízható szoftver nélkül az e-business nem működik, a szoftver már nemcsak segítője, hanem integráns része az üzletnek. A vállalatoknak azzal a paradoxonnal kell szembenézniük, hogy egyre rövidebb idő alatt kell új fejlesztéseiket illetve az alkalmazásaik folyamatos továbbfejlesztését megoldani miközben egyre jobb minőségű szoftvereket kell előállítani.
A szoftver ipar válasza a paradoxon megoldására két egymással szoros összefonódó fejlődési vonulat: jobb fejlesztő és futtató technológiák (i) szorosan integrálva egyre hatékonyabb életciklus támogató eszközökkel (ii). Az e-business alkalmazások ilyen módon történő hatékony fejlesztése az e-devopment, vagy e-fejlesztés.

Az e-fejlesztési technológiának támogatnia kell

- A gyors alkalmazás-és tartalom fejlesztést/módosítást
- Az alkalmazás-és tartalom fejlesztést és publikálást földrajzilag elosztott környezetben
- Biztosítania kell a fejlesztési folyamatok minél magasabb fokú atomatizálását, automatikus dokumentálását és magasszintű minőségbiztosítását, melynek része a minőség mindhárom dimenziójának (megbízhatóság, funlcionalitás és terhelhetőség) a tesztelése.
-

Az előadás a korszerű e-fejlesztést az életciklus támogató eszközök (ii) oldaláról mutatja be. Ismertetünk egy komplex teljes megoldást, ami kielégíti fenti elvárásokat. Széles körben elterjedt és igazolt, a szoftver iparág vezető cégei használják. Nyitott, gyakorlatilag mindegyik népszerű fejlesztő környezettel használható.
Fontosabb komponensei a következő ábrán láthatóak



Az életciklus támogató eszközök lényegesen gyorsabbá, hatékonyabbá teszik az e-business fejlesztéseket és magasabb minőségi mutatók elérését teszik lehetővé. A szoftver fejlesztés csapatmunka. Az ismertetésre kerülő eszközök egységesítik a fejlesztő csapatok munkáját, szabályozzák és javítják a szereplők közötti információcserét.

László Zoltán - Goldschmidt Balázs: CORBA oktatásának tapasztalatai

Jelen cikk célja annak bemutatása, hogy a BME Villamosmérnöki és Informatikai karán a Szoftverspecifikáció és tervezés béta szakirányon a 7. félévben futó Objektum orientált rendszerek című tárgyban miként oktatták a szerzők az elosztott objektumok kezelését, ezen belül a CORBA alkalmazását.
Az előadás a tantárgy részletes tematikájára, ezen belül is kiemelten a gyakorlati feladatokra koncentrál. A feladatok megoldásához a Rational Rose Enterprice-t is felhasználtuk.
Feltételeztük, hogy a hallgatóság rendelkezik általános operációs rendszer és hálózatos alapismeretekkel, valamint közepes Java programozási gyakorlattal.
A laborfoglalkozások célja a CORBA alkalmazásának elsajátítása volt. Az oktatás WinNT és Linux környezetben JDK 1.2-re épült. Megköveteltük a valóságos párhuzamosságot és a platformok közötti átjárhatóságot (NT-s szerver, Linux-os kliens és fordítva).
Az előadás röviden ismerteti a fejlesztő környezetet és módszert, a Rose szerepét majd részletesen bemutatja a mintafeladatokat.
A feladatok felölelték az implementált CORBA által nyújtott lehetőségek csaknem egészét.

- Abstract factory a szerver oldalon
- Callback mechanizmus
- Aszinkron hívás és várakozás a szerverre

- Paraméterátadás (inout és out, holder classok használata, kivételek dobása)
- Speciális paraméterek átadása (felhasználó által definiált objektumok, öröklés az IDL-ben)
- Dinamikus hívás

Az előadás végezetül összefoglalja az első kurzus tapasztalatait és a továbblépés irányait tárgyalja.

Maradi István: **Hálózat management a Westel 900-nál**

A mobil hálózatok világszerte nagy sebességgel nőnek, ami nem csak a kiszolgált előfizetők számában mutatkozik meg, hanem a felügyelendő berendezések és rendszerek dinamikus bővülésében is. Ennek fényében a hálózatmanagement nagy jelentőséggel bír, hiszen ezen keresztül érvényesülhet a szolgáltatók közötti minőség különbség. A hálózat management nagybonyolultságú szoftver rendszereken nyugszik, melyek fejlesztése mindig is kiemelt feladata volt lesz az üzemeltetőknek. Az itt felmerülő feladatokról, nehézségekről és lehetőségekről szól az előadás.

Molnár Bálint: **Struktúrák szimulálása Java-ban a java.lang.reflect package, és belső osztályok segítségével**

1. A felmerült probléma kifejtése, kommunikáció szerver-kliens architektúrában Java kliens és C-szerver között struktúrákkal, a korábbi C-kliens funkcionalitásának megtartásával.

2. Lehetőségek a megvalósításra, Hashtable, Properties, reflect, és ezek összehasonlítása.

3. A java.lang és java.lang.reflect csomagnak a probléma megoldása során felhasznált lehetőségei :

   - java.lang.reflect.Field osztály
   - java.lang.reflect.Array osztály
   - java.lang.Class osztály
   - java.lang.Object osztály

   Illetve az említett osztályok felhasznált függvényei.

4. Az adatkonverzió részletes leírása, a belső osztályok mezőinek konvertálása a kommunikációhoz szükséges byte-folyammá (kérés), illetve adott osztály és mezőinek feltöltése az adatfolyamból (válasz), mezők (fields) byte-folyammá való alakítása.

5. Összetett struktúrák kezelése, illetve több dimenzió melységű belső osztályok kezelése (belső osztályok belső osztályai), rekurzió.

6. A mi alkalmazásunkban megvalósított funkcionalitás, teljesítmény mérések, alkalmas-e a módszer nagymennyiségű adathalmaz továbbítására.

7. A megoldás jelentősége, mennyire volt hasznos az osztály implementálása a projectben.

- Abstract factory a szerver oldalon

- Callback mechanizmus
- Aszinkron hívás és várakozás a szerverre
- Paraméterátadás (inout és out, holder classok használata, kivételek dobása)
- Speciális paraméterek átadása (felhasználó által definiált objektumok, öröklés az IDL-ben)
- Dinamikus hívás

Az előadás végezetül összefoglalja az első kurzus tapasztalatait és a továbblépés irányait tárgyalja.

***Nagy István -Vég Csaba:* Generikus programszerkezetek**

A komponens paradigma esetén a hangsúly az objektumok belső tartalmáról azok szolgáltatásaira, azaz funkcióira helyeződik át. Az egyedeket „funkcionális csomópontoknak" lehet tekinteni, ahol a kívülről látható funkciók a lehetséges kapcsolatokat jelölik, azaz az egyedek a függvényeiken keresztül manipulálhatók. Így egy komponens szerkezete és megvalósítása csak annyiban lényeges, amennyiben az kívülről is látható. A komponens tekinthető egyfajta „szaktudáskonzervnek" is, hiszen egy szaktudás eredményeként jön létre, ugyanakkor annak csak a felhasználása lehetséges, módosítása nem.
Komponens elvű alkalmazás fejlesztése során a program írása adott (nem pedig tetszőleges) módon történik. Gyakorlatilag nem attribútumokat, hanem tulajdonságokat (*property*) definiálunk, vagyis az attribútumok író és olvasó metódussal rendelkeznek. A komponens funkciót is egy adott sorrend szerint rendezzük el. Ha egy program ilyen módon készül el, akkor az egy szabványos formával fog rendelkezni, amely könnyen olvasható, és a struktúrája is könnyebben azonosítható. Ugyanakkor az így kapott forrásban nagy a redundancia, ezért a tulajdonságokat egy automatikus mechanizmussal lenne célszerű generálni.
A generikus szerkezetek segítségével egy ilyen automatikus forrásgenerálás valósítható meg. Ez egy makró eszköz: adott szintaktikájú forrásból a generikus definíciók alkalmazásával egy eredmény szöveget generálunk, mely lehet egy program forrása vagy egy szöveges dokumentum. Ugyancsak lehetséges, hogy egy program forrásszövegében speciális megjegyzésként helyezünk el a generikus szerkezetre történő hivatkozást. Az eredmény ekkor ugyanabban a forrásban is megjelenhet, az eredeti programszöveg átírásával. Ennek a módszernek nagy előnye az, hogy az esetleges hibák közel esnek a problémaleíráshoz használt generikus szerkezetekhez, így azok könnyen megtalálhatók és javíthatók. A generálás során az egyes elemekhez szükséges információk gyűjtése is megtörténik, így gyakorlatilag több elem segítségével építünk meg további elemeket.
A generikus definíciók alkalmazása (hívása) tulajdonságlista segítségével történik. A tulajdonságokat alapértelmezésben egy általános szintaktika alapján adhatjuk meg, lehetőség van azonban arra is, hogy az adott generikus definíciókhoz külön elemzőt definiáljunk, mely a szövegnek a hagyományos szintaktikájától eltérő feldolgozással építi fel a tulajdonság listát, ezzel tovább rövidíthető illetve speciális körülményekhez igazítható a generikus szerkezetre való hivatkozás.

***Nick János – Kelemen Sándor:* e-üzleti alkalmazás fejlesztése Java Servlet és JSP technológiával**

Az előadás egy konkrét Webes alkalmazás fejlesztésének tapasztalatai alapján mutatja be a Java Servlet és JSP technológiák alkalmazását.

Napjainkban nem csak az Interneten fellelhető oldalak száma szaporodik rohamosan, hanem a már meglévő és új oldalak minősége is egyre emelkedik. A web-helyek üzemeltetői oldalaik folyamatos megújításával igyekeznek minél vonzóbbá tenni lapjaikat az Interneten barangoló potenciális ügyfelek számára. Ennek alapfeltétele, hogy el tudják végezni a naponta felmerülő változtatási igényeket (pl. újabb hirdetések, hírek, üzenetek közzététele). Mivel a web-helyek üzemeltetőitől programozási ismeretek nem várhatók el, a JSP technológia HTML sablonokkal (template-ekkel) oldja meg az információk igény szerinti megjelenítését. Így ugyanazon eredményhalmaz megjelenítésének megváltoztatása csak a sablon megváltoztatását igényli, az üzleti logikát tartalmazó alkalmazást nem kell javítanunk vagy újrafordítanunk. A dinamikus HTML oldalak előállításához szükséges információkat adatkezelő komponensek (Java beans) gyűjtik össze az adatbázisból, és az üzleti logikának megfelelő eredményhalmazok a JSP sablonokban definiált módon jelennek meg a kliensek számára.

A kliensek a szokott módon a számukra megjelenő HTML oldalakon található linkek segítségével navigálnak a rendszerben. Eközben minden linkre kattintás során böngésző alkalmazásuk a web szervertől újabb HTML lapok letöltését kezdeményezi. Statikus oldalak igénylése esetén a web szerver azonnal elküldi az eredményt a kliens számára, míg a dinamikus oldalakra intézett kérelmeket a Servleteknek továbbítja. A Java nyelven megírt Servlet alkalmazás tartalmazza az üzleti logikát. Elvégzi a kérés paramétereinek feldolgozását, kigyűjti a szükséges adatokat az adatbázisból, majd JavaBean formájában összeállítja az eredményhalmazt. Ezt adja át a JSP futtató környezetnek, amely a JSP sablonban definiált módon megformázza az eredményhalmazt, előállítja és elküldi a kérés eredményét jelentő HTML lapot.

Noszály Csaba: **Chronos, minőség fogalom az Interneten**

Előadásunkban bemutatjuk a Chronos szoftver tervezésének matematikai hátterét és a szoftver JAVAban implementált prototípusát.
A Chronos szortvert a Web szerverek elé telepítjük, gyakorlatilag minden Webes kommunikáció a Chronoson keresztül folyik. Itt fontos momentum, hogy a program lokális telepítésű, ez biztosítja, hogy minden, az adott Web szerver szolgáltatta laphoz, erőforráshoz képes egy vektort rendelni.
Rendszerünk alapfeltétele, hogy benne a böngésző userek egyértelműen azonosítva legyenek, így hozzájuk is rendelünk egy vektort. (Hossú távon érdekes lehet, hogy böngészőhöz, böngészetthez ugyanolyan típusú jellemzőt társítunk.)
Maga a böngészés, tehát amikor egy felhasználó lekér egy kiszolgálótól egy Webes erőforrást, transzparens módon tartalmazza a hozzájuk rendelt vektorok kölcsönhatását, ezt a kölcsönhatást vezényeli le a Chronos szoftver. Hogyan válasszuk meg ezt a kölcsönhatást? (E kérdés viszgálata az előadás egyik fő aspektusa.) A vektor koordinátáit mint kategóriákat, a koordináták numerikus értékeit mint az adott kategóriában való erőt képzeljük. Ez a dinamikai rendszer definiálná az Interneten a minőség fogalmát. Erre a rendszerre épülhetnének majd rá a legkülönfélébb elemző, prediktáló szoftverek.

Olti Ferenc: **E-Commerce, vízió és valóság**
Vízió

- USA-ból vezérelt önmagát megvalósító jóslatköteg
- minden e
- B2B (business to business) megoldások
- B2C (business to customer) megoldások

- számítógép mindenütt
- kommunikáció minden irányban
- trillió dolláros üzlet

Valóság

- Erősödő verseny az igazán nagyok között, élet-halálharc a kicsiknek
- piacszerzés új lehetőségei és nehézségei
- tőkeszerzés szempontjából polarizálódás az iparágak között
- permanens fenyegetettség az ajtó előtt

régiók versenye

Valóság ....

- Teljesen átalakuló iparágak
  - bank, biztosítás
  - informatika
  - telco
  - retail
  - hirdetés
- Humán kommunikáció átalakulása
- Kihívások Magyarország számára
  - előnyök, lehetőségek
  - hátrányok, fenyegetettség

***Papp Sándor:* Az integrált TMN és CORBA, a fejlett hálózatmenedzselés eszközei**

1. téma: A TMN alapelvei (összefoglalás)
2. téma: Szétosztott komponens technika: az Object Request Broker
3. téma: A CORBA és a TMN közötti viszony
4. téma: CORBA és TMN: az integrált hálózatmenedzselés eszközei
5. téma: CORBA „Plug and Play" környezet
6. téma: A több gyártótól származó CORBA termékek együttműködési képességei

A prezentációban érintett kérdések

- A CORBA és a TMN közötti viszony: azok milyen mértékben kiegészítői és mennyiben ellenfelei egymásnak?
- Milyen mértékben felel meg a CORBA alapú TMN az ITU TMN szabványoknak és ez milyen mértékben határozza meg a teljes rendszer-struktúrát?
  - Valószínűleg milyen mértékben fogja a CORBA befolyásolni a TMN szabványok fejlődését?
- Érvek a CORBA és TMN együttes alkalmazása mellett és ellen
  - A legfőbb előnyök, amelyek a CORBA és TMN együttes alkalmazása mellett szólnak egy integrált hálózatmenedzsmentben.
- A hálózat-üzemeltetők és szolgáltatók lehetőségei a kategóriájukban legjobb szoftverek hagyományos OSS rendszereikbe való rugalmasabb integrálására a CORBA alkalmazásával
  - Hogyan tud a CORBA hozzájárulni egy valódi „plug and play" környezethez?
- A különböző gyártók CORBA termékei közötti együttműködési képesség értékelése

***Ráth Péter:* WAP technológiát alkalmazó e-commerce megoldások**

1. Bemutatkozás
2. Az Anemo Kft. rövid bemutatása
3. A WAP technológia lehetőségei az informatikai rendszerekben, e-commerce, e-business
4. WAP technológia áttekintése
5. WAP-ot alkalmazó architektúrák
6. Dinamikus WAP tartalom előállíthatósága, technológiák (Java, PHP, CGI, Oracle timeport)
7. AnemoSky demo alkalmazások
8. A WAP technológia lehetőségei az informatikai rendszerekben, e-commerce, e-business. A mobil alkalmazások előretörése.
9. A WAP technológia áttekintése, összehasonlítás meglévő technológiákkal, a vezeték nélküli környezet kihívásai
10. WAP architektúrák
11. Dinamikus WAP tartalom előállítása. Szempontok, lehetőségek és technológiák áttekintése

***Szádeczky-Kardoss Szabolcs:* A követelmény-elemzés és -követés**

Az elõadás a követelmény és a követelmény-elemzés fogalmának tisztázása után
bemutatja a követelmény-elemzés fontosságát és hozzájárulását a fejlesztési projektek sikeréhez. Röviden jellemezzük a követelmények minõségének értékelési szempontjait, majd a számítógépes támogatás lehetõségeit vizsgáljuk
meg. Az elõadás második felében egy kiemelkedõ funkcionalitással rendelkezõ,
objektum-orientált követelmény-elemzõ és -követõ szoftverrel, az RTM Workshop-pal illusztráljuk az elveket..
A program terminológiájának tisztázása után az architektúrájának egyes részeit tekinthetjük meg röviden. Az elõadás végén
röviden összefoglaljuk az elhangzottakat.

***Szász Csaba – Nick János:* A szoftver minősége kritikus az e-üzletben**
Webes alkalmazások (HTML, Java) funkcionális és teljesítmény tesztelése

Az Internet robbanásszerű terjedésének köszönhetően használata fokozatosan a mindennapi életünk részét képezi. Az üzleti alkalmazások egyre nagyobb számban jelennek meg az Interneten, amelyek sokféle és összetett szolgáltatásokat kínálnak. A felhasználók száma egyik napról a másikra ugrásszerűen megnövekedhet, valamint a gyorsan változó követelményekre és új igényekre az alkalmazás fejlesztőinek rövid időn belül reagálni kell, ami nagy kihívás a számítástechnikai cégek számára.
Az üzleti alkalmazások sikerének egyik kritikus tényezője az alkalmazás megfelelő minősége. A funkcionálisan helyes működés, a megbízhatóság és a megfelelő teljesítmény vizsgálata a fejlesztési projektek igen fontos részét kell, hogy képezzék. A rövidülő fejlesztési idő és az

iteratív fejlesztési módszerek alkalmazása miatt szükségessé válik automatikus tesztelés végrehajtására képes eszközök alkalmazása.
Az igények gyors változásainak megfelelően módosított alkalmazás esetében nemcsak a változásokat hanem a teljes alkalmazást kell vizsgálni. Ez természetesen azt is jelenti, hogy ugyanazt a funkciót minden változás után újra és újra meg kell vizsgálni, le kell tesztelni. Ettől a fáradságos munkától szabadít meg az automatikus tesztelést végző eszköz. Az automatizált tesztelés az első szakaszban ugyan költségekkel jár, de a befektetett munka bőségesen megtérül a második, harmadik tesztelési körben. Az előadás bemutatja az alkalmazás funkcionális tesztelésének automatizálási lehetőségeit, a tesztelés eredményeinek elemzését, további kezelését.
Az alkalmazás telepítése után kiderülő teljesítményproblémákért nagyon magas árat fizetünk, akár az elmaradt hasznot, akár a csökkent termelékenységet, akár a felhasználói panaszokat nézzük. A teljesítmény tesztelés segítségével alkalmazásaink teljesítménye mérhető és előre megjósolható. Az automatikus teljesítmény tesztelő eszköze pontosan úgy teszteli a teljesítményt, ahogyan majd a végfelhasználó fogja azt látni, vagyis áthelyezi a hangsúlyt a „szerver bombázásáról" a valós felhasználói terhelésre a létező vagy a hipotetikus használati forgatókönyvek modellezésével. Ez az intelligens megközelítés a szükséges információknak a megfelelő minőségben történő biztosításával segíti az erőforrás-tervezési és telepítési döntések megszületését. Az előadás bemutatja nagyszámú virtuális felhasználó egyidejű vezérlését, és a teszt végén keletkező eredményhalmaz kiértékelését.

***Szathmáry Viktor:* Valósidejű információmegosztás**

- mire ad megoldást a JWorx keretrendszer?
    kapcsolódó technológiák (pl. JavaSpaces)
    gyakorlati problémák internetes alkalmazásoknál:
        futtatókörnyezetek sokfélesége
            (browserek: java 1.1, nincs RMI, firewall nehézségek)
    előnyök:
        egyszerűség
        következetesség
        gyors implementálhatóság
        skálázhatóság
- alkalmazási lehetőségek
    chat
    "shared workspaces"
    bármilyen subscribe-push információ (pl. tőzsde, hírek, stb)
- a framework felépítése (a wobok világa)
    World
        Wob-ok nyilvántartása, lookup név alapján
        [addWob, removeWob, getWob]
    Wob (world object)
        "subscription" elv
            SubscriptionManager - jogosultságok ellenőrzése
            WobListener - a Wob eseményeiben érdekelt objektum
                [események: subscribed, unsubscribed, subscriptionDenied]
        Registry (Wob-ból örökítve)
            tömbelemek (Item) nyilvántartására

itemsInit, itemAdded, itemUpdated, itemRemoved
Channel (Wob-ból örökítve)
egyszeri üzenetek szórása
message(Channel chan, Object msg);
- példa a JWorx keretrendzser alkalmazása
TCP/IP kapcsolat rétege, Item szerializálás
aszinkron eseménykezelés
sessionkezelés
konkrét wobok
- implementációs kérdések
object pooling
benchmark eredmények
- további lehetőségek
párhuzamosan több protokoll kezelése
több szerver összekapcsolása

***Szendrői Etelka:* A SYSTEM ARCHITECT 2001 integrált vállalati vízuális modellező eszköz**

A System Architect 2001 az 1986-ban alapított, New Yorki központú Popkin Software & Systems cég terméke. Magyarországi disztribútora a Magic (ONYX) Hungary, Budapesten.
A System Architect 2001 olyan modellező eszköz, amely egyetlen termékben kínálja az üzleti, a folyamat, a komponens, az objektum és az adat modellező technikákat, módszereket.
A System Architect támogatja többek között a Catalyst és IDEF szabványokat az üzleti és folyamat modellezésben, az UML-t a komponens és objektum modellezésben. Teljes támogatást nyújt a hagyományos strukturált elemző és tervező technikákhoz.

Üzleti folyamatok (újra)tervezése (BPR)

Az üzleti folyamatok modellezéséhez gazdag diagram készlet áll rendelkezésre, melynek segítségével elkészíthető a vállalat üzleti folyamatainak modellje, szervezeti modellek, elhelyezkedési modellek, alkalmazás modellek, adat és technológiai modellek.

Mátrix szerkesztő

A Mátrix szerkesztő támogatja a különböző szabványok szerinti (IDEF0, IDEF3,CSC Catalyst) mátrixok készítését, valamint lehetővé teszi, hogy saját felhasználói mátrixok készüljenek.

Adatmodell készítés

A System Architect 2001 számos adatbázis kezelőhöz –többek között Oracle, Sybase, DB2, SQL Server – képes sémát generálni és visszafejteni, valamint Magic, Visual Basic vagy Power Builder alkalmazásokat is képes generálni.

Tervezés

A System Architect támogatja a hagyományos strukturált módszertanok alapján történő tervezést, többek között az SSADM-et.

Többfajta UML alapú objektum orientált technikát támogat. A System Architect az alkalmazás fejlesztőkhöz való kapcsolaton kívül kódgenerálást és visszafejtést is lehetővé tesz a C++, Java és Visual Basic programozási nyelvek felé.

Jelentés, dokumentáció készítés

Mind a WORD mind a HTML formájú jelentések készítését támogatja. Számos Word sablon áll rendelkezésre a dokumentációk és jelentések elkészítéséhez.

Oktatás

Intézményünkben az Információs rendszerek tervezése c. tantárgy keretében találkoznak a hallgatók a System Architect CASE eszközzel és kiegészülve a Magic alkalmazás-generátor

alkalmazásával, a rendszerfejlesztés teljes életciklusát átfogó eszközrendszer áll rendelkezésünkre az oktatásban.

***Szűcs László:* WAP, a wireless Internet technológia**

Abstract
Az elmult néhány év a mobil technológiák olyan mértékű fejlődését hozta, amely megteremtette a nagytömegű információk elérésének igényét a mobilokról. A vezeték nélküli hálózati technológia azonban nem alkalmas (még) a közvetlen Internet elérésre, tekintettel a szűk sávszélességre, a kiszámithatatlan késleltetésekre, a csomagvesztésre,.....Ezért néhány nagy mobil cég a Wireless Application Protocol létrehozására szövetkezett, alapvetően az Internet tartalom és értéknövelt szolgáltatások vezeték nélküli elérésére. Az ipari szabványositásra lértehozták a WAP Forum-ot, amelyhez ma már több mint 500 cég csatlakozott. Ennek a protokol architektúrának az ismertetése az előadás anyaga, kiegészitve objektum orientált alkalmazási példával.

***Tóth Bálint:* "Forward engineering": Architektúra központú fejlesztés, minta alapú kódgenerálás**

Az előadás a szoftver fejlesztés új paradigmájával, az Architektúra Központú Fejlesztéssel foglalkozik. Bemutatjuk a paradigma elveit, összevetjük a hagyományos megoldásokkal. Vázoljuk a fejlesztési folyamatot és megvalósítását.
Megmutatjuk, hogy milyen hatással van ez a technológia fejlesztés sebességére, minőségére és karbantarthatóságára. Elemezzük a technológiának a szoftver életciklusra való kihatását.

***Vég Csaba:* Információ technológia — 2000 után**

Az elkövetkezendő évek informatikájának az egyik legnagyobb problémája valószínűleg az ipari méretekben történő alkalmazásfejlesztés, a „nagyüzemi szoftvergyártás" megjelenése lesz. A címben az angol nyelvű ismeretterjesztő sorozatra történő utalás is hangsúlyozza, hogy az előadás az informatika néhány *lehetséges* jövőbeni fejlődési irányát kísérli meg körvonalazni, elsősorban az ipari méretű szoftverkészítés szempontjából.
Informatika — 2001.
A Komponens-elv.
A közeljövő alkalmazásfejlesztésében a főszerepet valószínűleg a szoftver-komponensek fogják játszani. Az OO szemléletre épülő modern komponens-elv a hagyományos OO paradigmán is túlmutat, mivel lehetőséget ad arra, hogy az alkalmazást lazán összekapcsolt elemekből illesszük össze, illetve tágabb lehetőséget ad az elemek újrafelhasználására.
A komponens-elv esetén egyrészt megjelenik egy technikai szempont, amely a komponensek összekapcsolásának lehetséges módjait írja le. Beszélhetünk azonban egy komponens-szemléletről is, amely adott előírások szerinti — „szabványos" — programkészítést jelent.
A szoftverkészítés egyik legnagyobb problémája a szaktudás, amely jelenleg alapvetően csak a gyakorlatban végrehajtott fejlesztéssel, a „jó utak" megkeresésével és bejárásával szerezhető

meg. A szoftver-komponensek ebből a szempontból tekinthetők „szaktudás-konzerveknek", mivel adott feladatra készen alkalmazható megoldást nyújtanak. A komponens-szemlélet már „bejárt" és bevált utakat foglal össze, így alkalmazása esetén olyan, mintha azonnal többéves szaktudást szereznénk meg, amely különösen jelentős az ipari méretű fejlesztések esetén.

Generikus programozás

A szoftverfejlesztés és különösen a „működő" szoftverek előállítása, így a komponens-elv alkalmazása során az elkészített programok szövegében általában nagyon nagy a redundancia. A helyzet bizonyos szempontból hasonló a gépi-kód és az OO nyelvek viszonyára. Részben a jelenlegi nyelvek is kiterjeszthetők a komponens elvnek megfelelően. Általánosabb megoldást kapunk, ha lehetőségünk van arra, hogy generikus szerkezetek fogalmazzunk meg, és azokat adott forrásszövegek esetén alkalmazzuk.

A generikus programozás jelentősége a kész komponensekkel szemben, hogy ebben az esetben nem csak a szaktudás eredménye, hanem közvetlenül a rögzített szaktudás is felhasználható. Másik előny, hogy más definíciók alkalmazásával más konvencióra, ill. platformra képezhetjük le a forrásszöveget.

Informatika — 2010.

Közjáték — a Java+.

A Java+ egy olyan kísérleti rendszer, amely képes arra, hogy beolvasson Java forrásprogramokat és annak elemeiről — a függetlenül megadható külső ügynökök segítségével — információkat gyűjtsön, majd az információk birtokában a program bizonyos átalakításait javasolja, illetve kiegészítő információk megadásával ellenőrzéseket (pl. elő- és utófeltételek) hajtson végre.

A Java+-hoz hasonló, forrásprogram-újraíró rendszer segíti a helyes programozási stílus alkalmazását (kódolási konvenciók, final argumentumok, stb.), illetve különösen hasznos a fejlesztés ún. „refactoring" fázisaiban, amikor a kódon olyan átalakítást hajtunk végre, amelynek eredményeképpen az áttekinthetőbb lesz, ill. lehetőséget ad a további bővítésekre.

Az „Expert" technológia.

A forrásprogram-újraíró technológia tanulsága, hogy az alkalmas egy gyenge minőségben megadott forráskód átrendezésére az adott módon rögzített háttér-szaktudás felhasználásával. A módszer általánosításával egy olyan technológiához jutunk, amely adott forrást képes a rögzített szabályoknak megfelelően átalakítani, illetve ahhoz hozzáilleszteni és így kiegészíteni a megvalósításhoz szükséges szaktudással. A módszer segítségével függetlenül adható meg a rendszer szakterületi modellje és a megvalósításhoz szükséges tervezői megoldások. Az utóbbiak egy folytonosan bővíthető és újrafelhasználható szakismereti bázist alkothatnak.

A módszer lehetővé tenné, hogy a követelmények dokumentumaiból (a „szakterületi elemzésből") a rendelkezésre álló tervezési szakismeretek bázisa alapján közvetlenül előálljon a kész rendszer.

A beillesztés és újrarendezés segítségével mind a szakterületi modell, mind a szakismeretek definíciói egyenként is megadhatók, melyek egy szakértői rendszerbe kerülnek, azaz az ismeretek folyamatosan bővíthetők, a rendszer „tanítható".

Informatika — 2020.

„Hello számítógép!"[Star Trek 4]

Felmerülhet a kérdés, hogy milyen formalizmussal történjen a szakterületi modell, illetve az informatikai szaktudás megadása. Ki lehet fejleszteni külön modellező nyelvet is, variációs képessége miatt azonban a legcélszerűbbnek egy egyszerűsített természetes nyelv használata tűnik, mely minimális számú vizuális elemmel van kiegészítve.

Létezik a Java olyan generikus kiterjesztése, amely egy osztályt (pl. Vector) generikus definícióvá tud alakítani. Hasonló elv alapján megvalósítható, hogy konkrét megoldási példákból, illetve konkrét átalakítási módokból a megfelelő szakértői rendszer bizonyos szaktudás-elemeket ki tudjon emelni, „tanulni" tudjon. Ugyancsak elképzelhető, hogy

egyszerűbb esetekben adott bemenetek és az arra adandó válaszok felsorolásával a rendelkezésre álló elemekből a rendszer összeállítsa a megfelelő algoritmust.

Vékássy Bence: **A TeMIP implementálása a MATÁV távközlő hálózatban**

Bevezetés: A TeMIP-ről általában
1. A SNOMS-projekt
   - SNOMS komponensek
   - A projekt végrehajtásának fázisai
   - Bővítési irányok
3. A ki nem használt lehetőségek
4. A TeMIP jövőképe a távközlési szolgáltatónál

Veres Károly - Vég Csaba: **Alkalmazásfejlesztés dokumentumai**

Az alkalmazásfejlesztés és különösen a rendszerszervezés során a programszövegek mellett általában több szöveges dokumentumot is összeállítunk. Az „Alkalmazásfejlesztés..."* c. könyvben ismertetett rendszerfejlesztési eljárást követve ilyen dokumentum a követelményeket rögzítő előzetes felhasználói kézikönyv, a szakterületi modellt és a tervezés stratégiai döntéseit összefoglaló diagramok és szövegek, valamint a tervezési elemekkel kiegészített programozói leírások.
Ezek a dokumentumok általában nyomtatásra készülnek, azokat nyomtatott formában, papíron adjuk át a megrendelőknek, ill. a tervezőknek és a programozóknak.
Nyomtatott dokumentumok esetén több probléma is felmerül. Egyrészt ezek, különösen a nagyméretű (néhány ezer oldalas) dokumentumok általában nehezen olvashatók. Ez több félreértést is okozhat, például a megrendelő nem látja át a készítendő rendszer tervezetét. A másik probléma, hogy a különállóan elkészített dokumentumok a fejlesztés során elszakadnak a tényleges alkalmazástól, így nem követik a fejlesztés közbeni bővítéseket és módosításokat.
Az előadásban részben egy egyszerűsített, követelmény-alapú fejlesztési módszert szeretnénk bemutatni, valamint a szervezésnek egy ehhez kapcsolódó, hypertext alapú dokumentum-rendszerét.
A hypertext dokumentumok a nyomtatott szövegnél könnyebben áttekinthetők. Megfelelő technika alkalmazásával a dokumentum szimulálni képes a felületek elemeinek működését.
Megfelelő dokumentumrendszer esetén a leírás szorosabban kapcsolódik a program elemeihez, a fejlesztés során az sokkal egyszerűbben módosítható, így kisebb az esélye, hogy az elszakad a tényleges alkalmazástól.
Alkalmazásfejlesztés fázisai.
Követelményelemzésen a felhasználói kézikönyvet értjük, valamint a megtervezett felhasználói felületeket. Ez a dokumentum elsősorban a felsőszintű munkafolyamatok/üzleti-objektumok leírására helyezi a hangsúlyt, valamint tárgyalja az összes felhasználói felületet, az ahhoz kapcsolódó közép-szintű munkafolyamatokat („használati eseteket") és a felület minden egyes alkotóelemét.
Szakterületi elemzésen részben a rendszer üzleti logikájának olyan, mérnöki jellegű leírását értjük, mely mezőnként is részletezi az üzleti objektumokat, a közöttük lévő strukturális kapcsolatokat, valamint részletesen leírja az üzleti folyamatokat. A szakterületi elemzés egyben

tartalmazza az üzleti objektumoknak megfelelő csonkok (stub) programozását, azok összekapcsolását a felhasználói felületekkel és a nem szerver-oldali folyamatok (pl. ellenőrzések) megvalósítását; valamint jelenti a szerver-oldali folyamatok teljes logikai ("programozói") leírását és a logikai adatbázistervet.
Stratégiai tervezésen a platform és rendszerkövetelmények összegyűjtését, valamint a telepítési konfiguráció leírását értjük.

Hypertext dokumentumok.

A rendszer leírása három (egyre bővülő) szinten jelenik meg:

- *Felhasználói kézikönyv* (rendszer bemutatása, ill. help)
- *Logikai modell* (az előző, kiegészítve az üzleti objektumok, illetve folyamatok leírásával)
- *Program-leírás* (az előző, kiegészítve a programok leírásával)

A szervezés dokumentumain részben on-line, részben nyomtatható hypertext (html) dokumentumokat értünk.
Diagramok esetén szabványos UML alapú jelöléseket alkalmazunk.
A programok dokumentálására a programkódból automatikusan előállított html formájú hypertext dokumentációs szöveget alkalmazunk. A szerver-oldali folyamatok programozói leírása ugyancsak ezzel az eszközzel történik — ekkor azonban a megfelelő metódusok nincsenek implementálva.
A hypertext szerkezetű on-line szöveges dokumentumok elemei ún. linkekkel vannak összekapcsolva, így pl. egy hivatkozott felhasználói felület esetén azt, illetve annak a leírását azonnal meg tudjuk tekinteni. A szervezői- és programozói szintű dokumentumok esetén a hivatkozott fogalomnak, mezőnek vagy folyamatnak is hasonlóan megtekinthető a szervezői dokumentációja vagy az azt megvalósító program leírása, ill. fordítva, pl. a program leírása is tartalmaz linkeket a megfelelő fogalmakra.
A felhasználói felületek on-line dokumentációja (bizonyos szintig) szimulálja a felületek elemeinek működését (menük, stb), illetve az elemekről (ún. imagemap-jellegű technikával, azaz egy kattintással) információ kérhető.
A diagramok elemeiről a felületek leírásához hasonlóan (imagemap-jellegű technikával) információ kérhető.
Az on-line dokumentáció a felhasználói felületek működését képekkel szimulálja, így annak a megtekintéséhez elegendő az elterjedt Java 1.1 és JavaScript futtatására alkalmas web-böngésző (pl. IE4 vagy Netscape 4).

Vér György: ORB, JDBC kapcsolatok JBuilderrel

Elosztott üzleti rendszerek
- A Jó, a Rossz és a Csúf

Miért éppen az alkalmazáskiszolgálók?
- Az új szoftverkategória bemutatása

CORBA-alapok
- A CORBA interface
- A CORBA fejlesztési folyamat

RMI, IIOP és EJB
- Együtt létezés vagy együttműködés?
- Programozási modellek és protokollok
- RMI over IIOP – az RMI és a CORBA egyesítése
- Az IIOP jelenti az „E"-t az EJB-ben
- IIOP-t támogató EJB konténerek implementálása

Enterprise Java Beans
- Bevezetés
- EJB „szerződések"
- EJB-architektúra
- EJB és CORBA

EJB a gyakorlatban
- Állapottal bíró és állapot nélküli beanek létrehozása
- Entity beanek létrehozása
- Az EJB tranzakciós modell
- Az EJB biztonsági modell
- Élő példa: adatbázis-elérés alkalmazáskiszolgáló segítségével

***Werner Zsolt:* ORB interoperabilitás és skálázhatóság vizsgálata**

Napjaink elosztott technológiát alkalmazó programozásának egyik sarkalatos kérdése a különbözô elosztott protokollok interoperabilitása. Ezen belül kiemelt figyelmet érdemel a CORBA, melynek konkrét IIOP protokolljának megvalósítása eleve feltételezni nem csak a különbözô nyelven implementált objektumok által használt kommunikációs protokoll közötti interoperabilitást, hanem a különbözô gyártók által piacra dobott ORB-ok együttműködési készségét is.
A témakörben viszonylag kevés publikált eredmény van, és azok is a viszonylag régebbi, kommerciális, ill. C/C++-os GNU ORB-okat tárgyalja. Ez azt jelenti, hogy például az új, nem kommerciális Java-alapú ORB-okról, mint pl. a JavaIDL, még nem publikáltak igazán jó interoperabilitási teszteket.
Az Ericsson Conformance Lab színeiben és segítségével végzett interoperabilitási vizsgálatok a piacon található összes komoly ORB (a JavaIDL-t is) skálázhatóságát, és, ami a legfontosabb, interoperabilitását vizsgáljuk. Elôadásomban ennek elvi alapjairól, illetve a gyakorlati eredményekrôl fogok beszélni, megspékelve a legszéleskörűbb hallgatóság által is valószínűleg nagy érdeklôdéssel várt konkrét, máshol még nem publikált benchmark-eredményekkel.
Irodalom és kitekintés mások által elvégzett interoperabilitási vizsgálatokra:
http://colossus.itec.uni-klu.ac.at/~laszlo/pubrec.html - az 1999-es részben az 1. publikáció.
Ez a tesztek értékelésénél nem vette figyelembe a használt JVM skálázhatóságát, így az itt lefuttatott tesztek egy részét újra kell futtatni, immár az adott JVM teljesítôképességét is szem elôtt tartva. Ide vonatkozó referencia: Volanomark-tesztek a www.JavaWorld.com-on – a – különösen Linuxos – JVM-ek szélsôségesen különbözô skálázhatósága.
http://www.omex.ch/CorbaTB/corbatb.htm
Ez a honlap tartalmaz fordítható IDL test-suite-okat. Az alap- és az összetett típusok szerializációjának és annak CORBA-kompatibilitásának implementáltságát, valamint a (de)szerializáció/(un)marshalling, valamint a kommunikáció idôigényét vizsgálja. Nagyon extenzív körteszteket tartalmaz a legtöbb C++-os és Java-s ORB-rôl (a JavaIDL-rôl nem).

Neumann János Számítógép-tudományi Társaság
IV. ORSZÁGOS OBJEKTUMORIENTÁLT KONFERENCIA

PANEL

"A SZABAD SZOFTVER"

Moderátor: **HUTTER OTTÓ**

Meghívott előadók:

IQSOFT Rt.	NÉMETH MIKLÓS
IBM Magyarország Kft.	KISS TIBOR
Linux Szövetség	PÉTER LÁSZLÓ
SUN Microsystems	ZSEMLYE TAMÁS
Microsoft	KŐNIG TIBOR

ELTE-BME Informatika épület
Bp.,XI., Pázmány Péter sétány 1/D.
Java klub
ELTE oldal V. emelet, hátsó lépcső