OOPSLA '96

Object-Oriented Programming Systems, Languages & Applications

Stig Berild

Innehållsförteckning

1. Konferensen, allmänt

1.1 Konferensfakta

Konferensen ”Object-Oriented Programming Systems, Languages & Applications”, förkortad OOPSLA är en årlig, internationell konferens sponsrad av ACM. Årets konferens var den 11:e i ordningen, därtill strategiskt lokaliserad till San Jose, Kalifornien, mitt i Silicon Valley. Information finns på Internet under www.acm.org/sigplan/oopsla/.

Mellan 2- och 2500 deltagare mötte upp under tre konferensdagar. Av dessa återfanns många på de två aktivitetsdagar som föregick konferensen.

De två inledande dagarna bjöd på:

* Tutorials. En rikhaltig uppsättning en- eller halvdagskurser för såväl noviser som experter.

* Workshops. Syftet var att diskutera och utbyta erfarenheter under specifika teman för speciellt inbjudna (sådana som i förväg skickat in ett bidrag).

Under andra dagen tillkom

* Educator’s symposium, ett forum för utbyte av idéer, innehåll, erfarenheter kring utbildning inom Computer Science, företrädesvis inom Object Technology.

* Doctorial Symposium där doktorander diskuterade sina forskningsprojekt med varandra och med en expertpanel.

De tre konferensdagarna omfattade:

* Konferensen bestående av
  - Anföranden av prominenta personer
  - Utvalda bidrag av karaktären forsknings-, idé- och erfarenhetsrapporter. Ca 170 papper inskickade varav ca 40 valts ut. Ingen vidare imponerande siffra inom ett så stort och uppmärksammat område.

* Demonstrationer, ca 45 minuter vardera, av objektorienterade tillämpningar, verktyg, prototyper, mm.

* Posters, dvs presentationer på tavlor av forskningsresultat, eller pågående arbete som någon enskild person eller grupp vill föra fram och diskutera med konferensdeltagarna.

Med andra ord en imponerande uppsättning aktiviteter! Antalet konferensdeltagare var trots detta lägre än tidigare. Även påtagligt få utställare, dessutom de flesta med enbart konsultertjänster och utbildning i produktsortimentet. Förutom ett antal bok- och tidskriftsförlag. De senares montrar var de enda ordentligt välbesökta. Att den konkurrerande konferensen

ObjectWorld genomfördes samtidigt i Frankfurt, kan ha påverkat. Sedan finns numer ytterligare ett antal intressanta årliga konferenser med snarlikt tema att välja bland, exv Object Expo (www.sigs.com/conferences/) och TOOLS (www.eiffel.com/tools/).

Kanske är det så att Objektorientering (OO) blivit så brett att många väljer att delta på och exponera sina produkter på mer specialinriktade konferenser som Java-xxx, GUI-xxx, Reuse-xxx, ...?

<årets konferenssymbol>

Figur 1 Årets konferenssymbol

1.2 Konferenstema

Som konferenstiteln antyder har historiskt gällt en programmeringsinriktad infallsvinkel på OO. Samma tyngdpunkt gäller fortfarande. Eftersom OO och OO-modeller numer innefattar alla faser av ett systems livscykel samt kommer till användning inom allt fler tillämpningsområden har OOPSLA även inkluderat dessa perspektiv i intressesfären. I ingressen talades sålunda om ”perspectives on objects that range from the latest technical advances in the field to the human issues of software development cultures”.

Inte minst det mänskliga perspektivet är intressant. OO-folket, OO-kulturen har länge varit en ”closed community”, som i grunden trivts mycket bra med stämpeln av att ligga på teknikens framkant och som knappast funnit någon anledning, än mindre upplevt det som angeläget att släppa in företrädare för andra teknikområden, att behöva blanda sig med. Inom gruppen kan man skratta åt gamla COBOL-programmerare, datamodellerare, vattenfallsmetod-utvecklare, m fl kategorier. Varför störas av synpunkter, motargument från omgivningen, speciellt som det knappast kan leda till någonting fruktbart. Man har ju definitionsmässigt rätt och andra fel.

Presumtiva användare av tekniken har därutöver skrämts bort av en svårtillgänglig jargong och begrepp som var och ett framförs som ett under av uppenbar och kristallklar innebörd men som för en utomstående framstår i lika kristallklart sken som lika mångtydigt som otydligt. Det enkla, förnuftsbaserade har gjorts komplicerat och otillgängligt.

OO måste demystifieras, inte mystifieras. OO binder i långa loppet ris åt sin egen rygg. Ett flertal av de mer framstående OO-potentaterna valde också att konstatera att det nu är dags för en attitydförändring om OO ska kunna överleva och utvecklas. Ignorans för andra inriktningar, andras åsikter måste ersättas med öppenhet, respekt, ödmjukhet. Inställningar som ”glöm det du lärt dig och följ oss istället” måste ersättas med en positiv vilja att både ge och ta, en lyhördhet för alternativa ansatser och för användarbehov. Annars kommer aldrig OO att nå det förtroende och den spridning alla OO-företrädare så länge förutspått.

Som ett led i denna strävan hade två prominenta företrädare för ”människoperspektivet” bjudits in, Christopher Alexander och Larry Constantine. Mer om deras budskap nedan.

Anföranden och paneldiskussioner kring de ”mjuka sidorna” av Object Technology blev konferensens stora behållning. Att temat upplevs som angeläget bevisades av den stora uppslutning åhörare som lockade till dessa sessioner.

För övrigt kändes konferensen något urvattnad. Den entusiasm som präglat föregångarna verkade något avsvalnad. Kanske var det en tillfällighet?

1.3 Konferenspublik

Här möts OO-diggarna, OO-ekvilibristerna, de djupa tänkarna, förutom alla vi andra vanliga, som bara vill lyssna och veta mera. De allra flesta har en OO-språksorientering. Karaktären är påtagligt akademisk. Klädseln utgörs av T-shirt, jeans, ofta kortbyxor, sandaler. Skägg är vanligt. Många ungdomar. Hyfsad könsfördelning.

Skillnaden mot den andra stora OO-konferensen ObjectWorld är stor. Där träffas personer med industrianknytning, ofta något äldre, ännu oftare klädda i kostym. Skägg hittar man möjligtvis hos föreläsarna. Den manliga dominansen är stor. ”State-of-the-art” är ledstjärnan.

Förresten; kanske börjar OOPSLA tappa den ungdomliga trenden. Jag fick två gånger i hissen höra av högst allvarliga personer med lokal anknytning och med genomträngande blick att vitmelerade skägg tydligen började bli inne igen, att gamlingarna var på väg att ta över. Inklusive råsopen att vi gamlingar tydligen tagit kål på ungdomarna genom att driva denna intellektuella elit till utmattningens och slaveriets gräns. I ett försök undan total tillintetgörelse valde jag att tyda påhoppen som ett allmänt utslag för den hektiska, dynamiska och otroligt krävande arbetsmiljö som präglar Silicon Valley. Där finns fascinationen, spänningen, utmaningarna, möjligheterna men också de förbrukade, utslitna, desillutionerade. Kanske är det så att OO-programmerare m.fl. oftare än andra deltar i riskutsatta, tidspressade projekt. Kanske bör det tolkas som ett utslag av besvikelse för att OO generellt sett ännu inte fått det givna genombrott och den totala dominans alla dess företrädare tagit för givet. Kanske hade jag bara oturen att möta två desillutionerade. Eller - i bästa fall - lustigkurrar.

1.4 Inne i år

Roll-begreppet; i den skepnad det lanserats av Trygve Reenskaug i den nyligen utgivna boken ”Working With Objects”, Manning Publications, ISBN 0-13-452930-8. Boken behandlar OOram-metoden. Reenskaug har en mycket lång erfarenhet i branschen men har först nu hamnat i det mer intensiva rampljuset. Refererades ofta och gärna.

Sedan är det ingen nackdel om vi under samtal med andra blandar in begrepp som ”metafor”, ”pattern”, ”artifact”, ”ontology” och varför inte ”use case, you know”, lite här och där.

1.5 Ute i år

C++; liksom övriga år. Dock nu än mer efter inträdet av Java på arenan som kompanjon (konkurrent?) till Smalltalk.

2. Intressanta teman

2.1 Objektorienterad Analys och design (OOAD)

2.1.1 Inledning

Området har länge varit ett getingbo av olika ansatser, heta debatter, mycket tyckande. Metoderna har präglats av en ständig vidareutveckling, av ett lånande av andra, anpassning efter ökade erfarenheter. Tala om rörligt mål. Att vara ansvarig införare och underhållare av en OOAD-metod i en organisation måste vara en utmaning av rang. I realiteten börjar numer det mesta i de olika ansatserna att bli ganska överensstämmande. Tiden är mogen för harmonisering, ensning, sammanjämkning. Det är dags att byta ut de galjonsfigurer (så kallade ”gurus”) som hittills banat vägen och givit namn åt metodansatser och istället överföra ansvaret till olika typer av marknadsgrupperingar, industrikonsortia, standardiseringsorgan. Om så inte sker löper området OOAD att hamna i vanrykte, att inte tas på allvar. Object Management Group (OMG) har nyligen tagit ett hedervärt initiativ i rätt riktning. Se nästa avsnitt.

2.1.2 Standardisering

2.1.2.1 OMG

Object Management Group är en organisation med över 700 företag som medlemmar (däribland SISU). Deras syfte är att med hjälp av ett objektorienterat angreppssätt åstadkomma samverkan mellan fristående system och systemkomponenter, oavsett nätbasering.

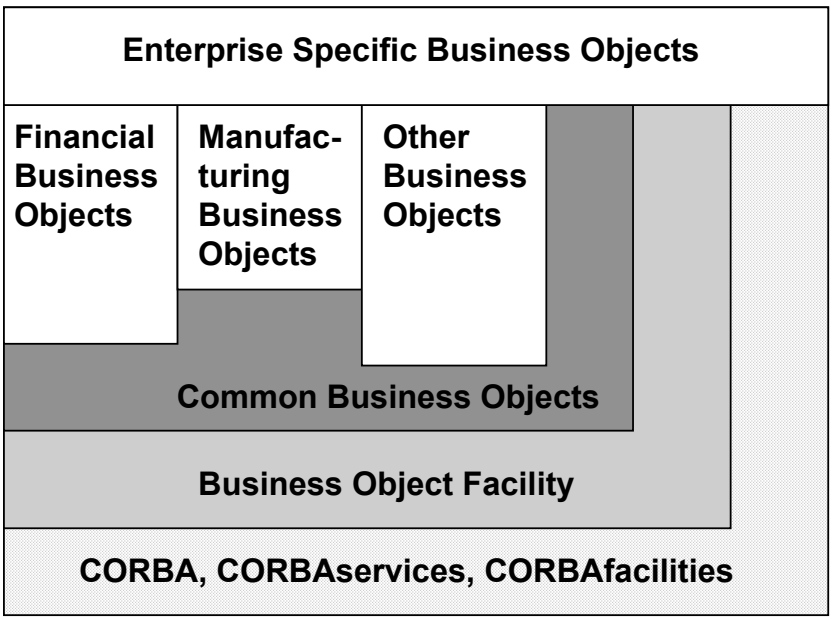
Tidigare har man helt arbetat med infrastrukturproblematik och samverkansmekanismer. Referensmodellen OMA (Object Management Architecture) och specifikationerna CORBA, CORBAservices och CORBAfacilities är välkända. Tillämpningssidan lämnades helt åt sitt öde. Orsaken kan hämtas ur OMGs

upprinnelse, med de stora teknologileverantörerna (hård/mjukvara) som dominerande aktörer. Användarna/kunderna fick ta vad som bjöds. Dessa har efterhand riktat allt starkare kritik mot teknikfokuseringen. Kunderna önskar visserligen fungerande basteknik men också betydligt mer av tillämpningsorienterat stöd. Med god lyhördhet för medlemmarnas synpunkter omorganiserades OMG för ett år sedan med syftet att bära fram de mer tillämpningsorienterade (domänspecifika, vertikala) aspekterna.

Plötsligt fanns begreppet ”Business Objects” på allas läppar. Ett allt starkare intresse växte fram för CORBA-teknologins reella användning inom olika tillämpningsområden såsom finans, hälsovård, telekommunikation, elektronisk handel och transport. Tillämpningsspecifika komponenter i form av Business objects (BO) ska enligt den nya visionen samverka med hjälp av en generell Business Object Facility (BOF), en slags högnivå Business Object Request Broker. Över denna BOF ”sammanträffar”

* tillämpningsspecifika BOs.
* tillämpningsområdesspecifika, förhoppningsvis av OMG standardiserade, BOs.
* tillämpningsneutrala BOs - Common Business Objects, ävenledes förhoppningsvis OMG-standardiserade.

Se vidare figur 2.



Figur 2

I samma tillämpningstillvända anda riktades även blickarna mot OOA&D. Tiden var mogen för ett helhetsgrepp vad gäller

* de modelleringsbegrepp som ska användas under olika skeden av OOAD (metamodell) och som representerar de modeller som anses behöva konstrueras. De flesta tycks ense om att OOAD åtminstone omfattar modellerna

- Static model
- Dynamic models; behaviour, state-transition, interaction models
- Usage model
- Architecture model; system composition model

* det begreppsmodelleringsspråk som ska användas för att formellt och entydigt kunna beskriva metamodellen (meta-metamodell). En standardiserad grafisk notation för dessa begrepp underlättar förståelse samt samverkan mellan människor med olika bakgrund.

I OMGs anda är syftet att lägga grunden för samverkan, interoperabilitet mellan producenter och konsumenter av olika OOAD-modeller, dvs ofta någon form av CASE-verktyg, planeringsverktyg o dyl. Snarlikt standardiseringsarbete har pågått i ett antal år inom EIA/CDIF. Se vidare nedan under 2.1.2.3 COMMA.

Eftersom OMGs syfte är interoperabilitet ansåg man det klokt nog varken vara deras uppgift eller ens möjligt, lämpligt att standardisera de analys- och designmetoder som tänkes utnyttja begreppen. Metodval beror på många faktorer, som tradition, kompetens, typ av tillämpning, ambitionen med aktuell analys och design, storlek på projekt, osv. Vad ska kunna förstås och kommuniceras utan koppling till hur det togs fram.

I enlighet med OMGs normala standardiseringsrutiner har ett Request for Proposal (RFP) skickats ut. Detta RFP anger syfte, villkor, avgränsningar för den OOAD-facilitet man önskar standardiserad, samt inbjuder medlemmar att lämna preciserade förslag. Senaste inlämningsdatum är 17 januari 1997. De som avser lämna in något alster ska indikera detta med ett Letter of Intent (LOI) senast 15 oktober 1996. Ett LOI innebär ingen förpliktelse att faktiskt komma med ett förslag.

Förhoppningen är att de inlämnade förslagen ska kunna kompromissas fram till en enda resulterande standardspecifikation. Kompromissarbetet brukar ges 3-5 månader. Tanken är att en standard ska kunna presenteras någon gång vid halvårsskiftet 1997.

Misslyckas man blir det ett kännbart och allvarligt bakslag för OOAD och dess företrädare, eftersom detta ”rally” följs med mycket stor uppmärksamhet och ivrigt intresse av de allra flesta OO-intresserade. Lyckas man, blir det en lysande triumf för det förslag som utgjort huvudingrediensen i standarden.

Ett antal företag har skickat in ett LOI, men redan pågår i skymundan samarbetsdiskussioner mellan de flesta av dem. Att kunna lämna in ett förslag som ett antal tunga företag står bakom, skapar en styrkeposition inför kompromissarbetet.

För närvarande (oktober) tycks Unified Modeling Language (UML) och Common Object Methodology Meta-model Architecture (COMMA) vara givna förslag. Mer om dem nedan.

Att denna standardiseringsaktivitet inom OMG är spännande och röner stor uppmärksamhet visade inte minst den våldamma tillströmningen av åhörare till

paneldiskussionen ”Standardizing OOAD: What does it mean?”. Frågor som ”Är tiden mogen?, Vad ska standarden innehålla? Är det OMGs bord? Vad är syftet? Vad blir effekterna? Vem tjänar på en standard? Låser man inte fast något som hela tiden måste kunna utvecklas? Ger det ökad öppenhet eller låsning? Går det verkligen att nå en enda standard från så (förmodligen) olika utgångsförslag? Eftersom, handen på hjärtat, väldigt få tillämpar OOAD-metoder idag - kommer en standard att få någon reell betydelse? Kanske just därför? Kommer inte visionen om globalt samverkande, distribuerade systemkomponenter/objekt att i grunden ändra förutsättningarna för innehållet i OOAD?”

På den sista frågan uttryckte Mary Loomis, Hewlett-Packard och ordförande i OOAD Task Force, en stark förhoppning om att kunna nå fram till ett enda, samkompromissat förslag. Kunskapen om effekterna av ett misslyckande verkar enande. Dessutom är inte begrepps-skillnaderna så stora (även om metoderna möjligtvis är det). I slutändan av OMG-aktiviteten står framförallt användarna som vinnare. Säkerligen även CASE-leverantörer som slipper svara upp mot en lika vildvuxeb flora ansatser. Allt under förutsättning att man kan ensas förstås.

Några mycket spännande månader är att vänta.

2.1.2.2 UML (Unified Modeling Language)

Läge

Tre av de mer (mest?) kända OOAD-företrädarna är Grady Booch, James Rumbaugh och Ivar Jacobson. Länge representerade de var sin metodansats. För ett par år sedan ”införlivades” Rumbaugh till företaget Rational, det företag Booch sedan länge varit verksam i. Första steget mot mer ensning och stabilitet inom OOAD metoder hade tagits. Vid föregående OOPSLA, oktober 1995 presenterade de under stor uppmärksamhet ett gemensamt förslag till metod under beteckningen ”Unified Method, version 0.8”. Debattens vågor gick heta från ”Äntligen, här har vi den slutliga standarden!” till ”Tror två gubbar att de förstår bäst och kan diktera innehållet i OOAD-metoder?”. Ungefär samtidigt förstärkte Rational kraftigt sitt momentum genom att införliva Objectory och Ivar Jacobson. Plötsligt fanns tre av totalt en handfull framträdande metod-gurus i företaget.

Man valde att i första hand koncentrera sig på att ta fram en gemensam begreppsplattform för analys och design. Med given semantik hos begreppen och deras samband kan OO-tillämpningar byggas på ett entydigt sätt. Dessutom kan modeller lika entydigt kommuniceras mellan dem som förstår och accepterar begreppen. Ju mer accepterad begreppsplattform desto större samverkansmöjligheter. Frikopplingen mellan begrepp och metod bidrar samtidigt till flexibilitet och frihet i metodval. Metodpreferens är ju, som tidigare nämnts anvhängigt ett antal varierande faktorer. Följdriktigt omdöptes specifikationen till ”Unified Modeling Language” (UML). Jacobsons influens resulterade (augusti 1996) i en komplettering av Version 0.8 med ett Version

0.9 Addendum. Den närmast liggande målsättningen är att nå fram till en Version 1.0 för vidare leverans till OMG i januari 1997.

Väl medvetna om OMG-processens villkor har man skickligt lyckats knyta till sig mycket starka bundsförvanter i och för den avslutande finjusteringen och för att markera kraft. Bland bundsförvanterna finns Texas Instruments, Hewlett-Packard, Oracle och Microsoft. Vem står emot denna armada?

Birds of a feather

En av Rational spontant ordnad sammankomst (går i konferenssammanhang normalt under epitetet ”Birds of a feather”) kring UML gav dem ett angenämnt belägg för att intresset kring UML är stort. En överfull lokal rymmande ca 300 sittande lyssnade andäktigt till vad de tre herrarna Booch, Rumbaugh och Jacobson hade att säga och hur de valde att svara på intrikata frågor. En gång guru, alltid guru tycks gälla. Det ska dock sägas att det knappast längre är dessa tre personer som agerar för detta, utan vi alla övriga. Ett svaghetstecken att hela branschen sitter som väldisciplinerade skolbarn och okritiskt bejakar vad tre, i och för sig mycket kvalificerade, personer ”klämmer fram”. Guru-fixeringen måste upphöra. Utvecklingen bör fortsättningsvis ske inom starka branschorganisationer eller standardiseringsorgan. OMG-initiativet är i det sammanhanget befriande - antagligen även för triumviratet.

Ingenting påtagligt nytt presenterades annat än budskapet att oavsett om man får UML accepterat av OMG eller ej kommer man att fortsätta driva sin inriktning. Ett resonemang som tillstår den som bedömer sig ha makt över marknaden. Hoppas att kompromissviljan finns när det kommer till kritan.

De tre herrarna tycktes trivas alldeles utmärkt i strålkastarljuset. De tycktes därutöver trivas förträffligt med varandra. Jacobson formulerade som avslutning en varm, finstämt skämtsam och personlig hedersbetygelse åt sina två nyvunna kompanjoner.

2.1.2.3 COMMA (Common Object Methodology Meta-model Architecture)

COMMA är beteckningen på en generell metamodell, inkluderande de allra flesta kända OOAD-metoder, inklusive dem som Rational står för. Bakom COMMA står OPEN Object Alliance, en sammanslutning av ett antal företag och kända företrädare inom metodområdet.

Bland företagen återfinns i-Logix, Intellicorp, Intersolv, Mitre, ObjectFocus, Object International, ParcPlace-Digitalk, PLATINUM, Ptech, Swissbank.

Bland kända företrädare återfinns Colin Atkinson, Peter Coad, Donald Firesmith, Ian Graham, Brian Henderson-Sellers och Rebecca Wirfs-Brock.

OPEN (den misstänksamme frågar sig säkert vilket som kom försSammanslutningen har till syfte att föra ut och vidareutveckla en flexibel

metodik för hantering av systems fulla livscykel, inte bara utvecklingsfasen. Vilket namn kan vara lämpligare för detta än OPEN? OPEN står, förmodligen efter viss kreativ möda, för Object-oriented Process, Environment, and Notation. Till skillnad från UML är OPEN alltså huvudsakligen en metodik. COMMA står för den bakomliggande metamodellen. En anpassning av COMMA för att svara mot OMG-kraven är under utarbetande. Basen är till stora delar det arbete som under flera år bedrivits inom EIA (Electronic Industries Association) under beteckningen CASE Data Interchange Format (CDIF). Se vidare SISU/TRIAD-rapport 21, 1994. CDIF är både namnet på den arbetsgrupp som bedriver arbetet och de standarder som tas fram. Syftet är att ta fram standarder för utbyte av information mellan modelleringsverktyg inom mjukvaruområdet. För att åstadkomma ett standardiserat utbyte behöver åtminstone följande ingredienser specificeras:

a. Ett överföringsformat inklusive syntax och inkodningsregler.

b. En uppsättning modelleringsbegrepp (metamodell), exv ”Objekttyp, Flödestyp, Aktörstyp, Händelsetyp”. Med hjälp av begreppen kan innehållet i de samverkande verktygens databaser beskrivas. Sändare och mottagare använder sig av en gemensamt förstådd överföringsmodell.

   I exv ett CASE-verktygs databas kan finnas ett antal datamodeller, flödesmodeller, objektmodeller, processmodeller, tillståndsmodeller, användningsfall, mm representerande beskrivning av en eller flera tillämpningar. Exv kan det i databasen finnas uppgift om att händelsetypen ”Telefonsignal” aktiverar aktörstypen ”Ordermottagare” som har att registrera en viss förekomst av objekttypen ”Kund”, mm och därefter skicka informationstypen ”Order” till aktörstypen ”Leverans”.

   Metamodellen kan liknas vid schemat för en vanlig databas.

c. En uppsättning modelleringsbegrepp (meta-meta modell) för att beskriva metamodellen. Jämför begreppen ”relation”, ”attribut”, ”domän”, mm i relationsmodellen för en vanlig relationsdatabas. CDIF använder sig av en enkel form av Entity-Relationshipmodell som meta-metamodell. Med hjälp av meta-metamodellen kan metamodeller diskuteras samt kan metamodell-komponenter kommuniceras.

Delarna a och c är standardiserade. Under b finns ett antal modeller för beskrivning av olika faser av systemutvecklingscykeln och enligt olika metodansatser definierade. Fler tillförs efterhand. Aktuellt i samband med inlämningen till OMG är arbete på en integrerad metamodell för OOAD. Sannolikt kommer man samtidigt att rekommendera CDIFs meta-metamodell som standard för meta-metamodellering.

Medan UML i första hand integrerar de tre huvudpersonernas metodansatser har COMMA som målsättning att svara upp mot i stort sett alla förekommande OOAD-ansatser i sin metamodell. Dessutom skiljer sig de båda meta-metamodellerna åt. CDIFs är betydligt enklare samt en konventionell datamodell medan UMLs har betydligt fler drag av renodlad objektmodell.

2.1.3 Användningsfall

Varför denna uppståndelse kring användningsfall (use cases)? Även i år. Där finns egentligen ingenting fantastiskt nytt eller komplext, utan något ganska naturligt att formulera och som har funnits i olika nyanser i ”alla år” inom den så kallade administrativa tillämpningssfären. Dock verkar det vara en nyupptäckt för programmerarskråret. Är det så illa som Constantine m.fl. (se nedan) skämtar om att OO-folk genuint ogillar användare (actors) och av den anledningen inte finner någon mening med modeller där det perspektivet finns företrätt? Eller är det kanske bara så att OO och OO-språk historiskt främst kommit till användning inom realtidssystem, tekniska tillämpningar där ofta användare inte spelar en lika påtagligt central roll?

Härmed ingen kritik mot användningsfallsmodeller. Det är säkerligen ett mycket effektivt konstruktionsinstrument. Annars skulle knappast de andra metoderna inkluderat det i sin repertoar. Heller ingen kritik mot upphovsmännen Ivar Jacobsson och hans gäng. Tvärtom. De har funnit en ”lucka” inom OO och skickligt fyllt ut den till det internationella samfundets vördnad och respekt. På köpet ger det Sverige renommé genom referens till den ”skandinaviska skolan”, mm. I bara farten placerar för övrigt en hel del personer Simulas uppkomst alldeles oförtjänt till Sverige. På direkt förfrågan mumlar vi svenskar något vagt om att Simula definitivt hade och har sina främsta anhängare i Sverige.

På tal om Skandinavien. Under en paneldebatt försökte man till och med analysera skandinaviernas erkända förmåga att hantera abstrakta modeller av olika slag, förmåga till problemanalys, mm. Slutsatsen tycktes bli att det är så mörkt och kallt där uppe i Skandinavien att vi inte har mycket annat att göra än sitta inne, analysera och fundera. Något vi gärna instämmer i för att hålla denna exotiska tolkning vid liv. Och för det goda rykte det för med sig.

2.1.4 Övrigt

Några uppsnappade synpunkter från paneldebatten ”Standardizing OOAD; What does it mean?”:

* Notation är mycket viktigt eftersom det är kommunikationsinstrumentet med och mellan utvecklare, användare m.fl. parter. Borde därför standardiseras.

* En standard får inte vara rigid. Den måste erbjuda möjligheter till välordnad vidareutveckling och anpassning.

* Både metod och i viss mån begrepp är avhängigt den tilltänkta tekniska infrastrukturen för det tillverkade systemet. Varierar exv beroende på om Smalltalk eller C++ ska vara målspråk.

* Användningsfall är i sig knappast med nödvändighet kopplat till OO. De behövs och kan konstrueras oavsett vilka övriga beskrivningsmodeller man avser ta fram.

* En processbeskrivning, en metod är nödvändig för att placera in modellerna i ett sammanhang och för att relatera dem till varandra. Alltså behöver även processen standardiseras. OMGs RFP är därför otillräcklig. Resultatet blir i bästa fall att antal modellbegrepp på vilka ett antal olika metoder kommer att appliceras. Dagens splittrade situation kommer att fortsätta, bara på en högre nivå.

* En process-standard skulle kunna etableras på en ganska generell nivå ur vilken sedan specialiseringar kan skapas för olika syften. Syftet är att lägga en grov ram för enklare gemensam förståelse. Kanske är det dock för tidigt med process-standard. Kanske måste den just nu läggas på en så generell nivå, för att klara alla upptänkliga behov, att den blir urvattnad och ”orkeslös”. Kanske måste begreppen ytterligare stabiliseras först.

* Vi får trots allt inte glömma att OOAD knappast befinner sig i systemutvecklinguniversums medelpunkt, även om OO-konferenser och dess OO-förespråkare gärna vill ge sken av detta. De flesta system utvecklas fortfarande konventionellt.

* Begreppsstandardisering är en överloppsgärning. Det behövs olika begreppsuppsättningar (metamodeller) för olika syften. Däremot bör respektive namngiven metamodell (UML, COMMA, ...) standardiseras så att man exakt vet vad respektive står för. I alla händelser bör vi avvakta användning av COMMA och UML och de erfarenheter dessa ger. Båda språken är ju alldeles nya och oprövade (”pappersdrakar”), även om de baseras på en massa erfarenheter från användning av deras föregångare och inspriationskällor.

Som synes är debattläget intensivt, åsikterna många och olika. Trots det stabila läge alla tycks instämma i uppnåtts.

2.2 Objektorienterade språk

Smalltalk har länge varit det givna OO-språket att använda och resonera kring, en renlärighetens klara bäck. C++ är i sammanhanget att betrakta som ett förorenat vattendrag. Ett alldeles nytt störande inslag i mönstret är givetvis uppkomlingen Java. Man vet inte riktigt vad man ska göra med det, vad man ska tycka. Är det något att ta till sitt hjärta, eller bara en rå ihopblandning av C++ och Smalltalk, en vårflod som snart ebbar ut? Att intresset är stort framgick tydligt av deltagarantalet vid James Goslings - Javas upphovsman - presentation som inbjuden talare under temat ”The Feel of Java”. Java är med andra ord i högsta grad "inne”.

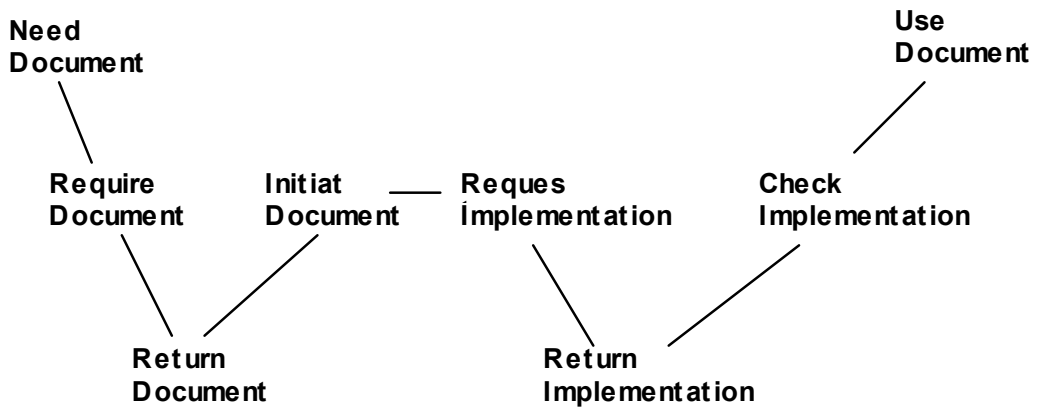
Gosling noterade bl a:

* Java har utvecklats för konkreta behov, inte ur akademisk teori. Genomgående målsättning under framtagningen har varit ”Keep it simple”. Inga ”Wouldn’t it be nice if we included .....”.

* Bland egenskaper att speciellt framhålla:

- Avancerad kompileringskontroll
- Generering av byte-kod
- Bra ”exception handling”
- Strongly typed
- Dynamic linking - late binding
- Portabilitet
- Runtimesystemet i sig portabelt.
- Omfattande bifogat klassbibliotek.

* Till skillnad från General Magic’s motsvarande lösning (Telescript) för agenter skiljer Java klart på beteende och data. Kanske inte den snyggaste OO-lösningen, men oftast mycket praktisk för att vinna transportprestanda. Programmet förs över en gång varefter det kan arbeta på många objekts data. Knappast ens en OO-lösning! Se figur 3.



Figur 3

* Java är för övrigt det hittills närmaste OO-området kommit en massmarknad. ”Java is for the masses”.

2.3 Återanvändning

Återanvändning är ett stående tema på varje OO-konferens. Man kan skönja både uppgivenhet och förhoppning. Alldeles uppenbart finns många infallsvinklar, barriärer, ansatser. Vi väntar fortfarande på det stora flödet goda erfarenheter, inte bara några enstaka oaser i öknen. Återanvändning satt i system. Återanvändning är ju trots allt ett av de tyngsta argumenten för OO.

Kanske kommer OMGs unga men mycket starka intresse för Business Objects (BO) och för ämnesområdesspecifik BO-standardisering att bli en katalysator som kan påskynda intresse, kanalisera krav, vara en seriös plattform för såväl process som resultat. Se figur 2, ovan.

En panel under rubriken ”Perspectives on Reuse” diskuterade olika infallsvinklar på återanvändning. Nedan följer några uppsnappade synpunkter:

* Återanvändning måste ses i ett större perspektiv, inklusive organisation, samverkansmiljöer, systemlivscykelansats och metodmognad, inte som en tekniknära facilitet. För att fungera måste många befattningsprofiler engageras, förstå, acceptera och verka för återanvändning. Det gäller att övertyga

  - Sponsor

    ”Är det värt det? En investeringskostnad med osäkra, för avlägsna intäkter.”

  - Project manager

    ”Osäker kvalitet. Vilka garantiåtaganden? Hur länge? Man köper grisen i säcken.”

  - Application developer

    ”Not invented here-attityd. Inte lika roligt längre. För mycket styrning och beroende. Tappar kompetens, blir bara en robot.”

  - Component crafter

    ”Mina komponenter är bättre, tycker jag. Jag har mer känsla för behoven.”

  - Library Manager

    ”Alla måste fråga mig. Efterfrågestyrd service är enklare än aktiv marknadsföring. Det är så jobbigt att underhålla ett bibliotek. ”

* I grunden måste återanvändning baseras på service och marknadsföring, inte på tvång.

* Design Patterns är ett finurligt sätt att få med utvecklare och implementerare på återanvändning samtidigt som det skalar bort lite av ”egot” hos dem.

* Frameworks skapar stadga och enhetlighet. Utvecklare känner igen sig. Måste dock vara synnerligen välgenomtänkt. En balansakt mellan oönskade bindningar och för mycket genereralitet, för lite profil.

* På sikt kommer sannolikt web-en att bli ”the ultimite reuse library”. Dock krävs ordnade former för service, ansvar, ägarskap, mm.

* Återanvänd service, dvs packetera en viss typ av service för något avgränsat behov som ett objekt, snarare än att se objektet som en verklighetsavbildning i sig. Man kan då tänka sig att detta serviceobject inom sig är uppbyggt av ett antal samverkande interna objekt som tillsammans åstadkommer utlovad service. Någon såg här en naturlig roll för användningsfall. Andra ansåg att serviceprofilen var just avsikten med Business Objects (BO), medan ytterligare någon ansåg BO mer vara representanter för tingen i verksamheten. Alldeles klart är dock att definitionen på BO är mycket oklar. Ett buzzword på tyckarens bord.

* Ad hoc reuse är ibland mycket effektivt. I en socialt välfungerande miljö får man tips om komponenter (både egenutvecklade, externa, inköpta) hos kollegan ”next door”, över mail, vid lunchbordet, ... Ett återanvändningsnätverk utan byråkrati.

* Trots allt gäller det i slutändan att övertyga de människor som deltar i ett utvecklingsarbete om nyttan med återanvändning bortsett från alla regler, tekniska hjälpmedel, mm. Återanvändning måste bygga på sin egen lönsamhet, tolkat av den som ska återanvända.

* Att återanvända är en fråga om förtroende. Har lika myckat att göra med psykologi och ansvar som med organisation och regler. Välfungerande service måste dock kunna tillhandahållas. Dit hör en välfungerande organisation och metoder för såväl förstagångsutvecklingen som för återanvändningsprocessen.

* Påfyllnad, underhåll av objektbibliotek är ett problem i sig. Hur motivera bidrag? Behövs belöningssystem? Hur rensa, reorganisera, underlätta användning, marknadsföra...?

* Ska återanvändning ske genom kopiering eller referens? I det senare fallet krävs en mycket genomtänkt hantering.

* Någon riktade ett tröstens ord till de församlade. Han hade ett mångårigt förflutet vid en stor kopieringsmaskinstillverkare, dvs en verksamhet präglad av denna industriella, ingenjörsmässiga utveckling och drift vi drömmer om att ta efter inom mjukvaruområdet. Han dödade krasst vår illusion genom att konstatera att de olika kopiatormodellerna utvecklades i stor isolering med mycket lite återanvändning, trots att både vissa sammansatta mekanismer, och småkomponenter, som skruvar och muttrar, mycket väl skulle kunna återanvändas. Ändå var det en av de mer framgångsrika kopiatortillverkarna! Frågan är om det var så trösterikt mitt i mjukvarukrisens alla dystra profetsior.

2.4 De mjuka aspekterna

2.4.1 Christopher Alexander

Arkitekt, professor University of California, Berkeley. Har skrivit ett antal uppmärksammade böcker genom åren. Kanske mest känd inom IT genom sin bok ”A Pattern Language” (1977), något av en kultbok för anhängare av patterns inom OO. Det intressanta är att han egentligen applicerar patterns på fysiska strukturer som bostäder, allmänna platser, i sin egenskap av arkitekt. Tänkandet kan dock appliceras i sin generaella form även på datasystem. Det är lika viktigt att det arkitekten konstruerar åt oss fungerar, som att datasystemen svarar upp mot våra behov. Förr var det enklare eftersom de som bodde på en plats själva utformade sin omgivning. Det moderna samhället har tagit in andra krafter med endast begränsad kunskap om det de sätts att

förverkliga. Samma gäller konstruktörer av datasystemen. Det behövs patterns i båda fallen för att se helheter, mönster, för utvärdering. Alexander talar om det moderna samhällets bristande moral och ansvarstagande. Gäller även OO-folket genom sin ofta påtagliga teknikfokusering. Var finns värdering av upplevelseaspekter, relationer, sinnesstämningar, rytmer? (Här gjorde Alexander en mycket talande referens till den stora, sällsynt opersonliga konferenssalen.)

Vi diskuterar mycket sällan hur mjukvara ska hjälpa oss att bevara och utveckla en bra värld. ”Where is the responibility to create a lasting living structure?”. Vi löser endast de små lokala problemen enligt principen "Guns for hire": "Tell us what to do and we do it.”

I avslutningen gav Alexander en välriktad salva mot teknikavsnävning och för helhetssyn och ansvar. Det är de som befinner sig i teknikens framkant som kan påverka - om de vill. Gör man inte det fritar man sig svekfullt från ansvar, blir robotar under andra krafters kontroll. Ett fritagande som vi senare kan komma att ställas till svars för.

2.4.2 Larry Constantine

Larry Constantine är välkänd för alla inom Software Engineering och inte minst för dem som en gång i tiden lärde sig principerna bakom ”Structured Design”. Constantine är professor, debattör och internationell föreläsare inom såväl computer science som human science. Han för med kraft fram behoven av de mänskliga hänsynen kring informationssystemutveckling och -användning. Som inbjuden talare valde Constantine att tala under temat ”Objects as if People Mattered”. Med hjälp av ett sällsynt elegant presentationsmaterial inkluderande både bild, ljud, video, delgav han lyssnarna många tänkvärdheter. Bland dessa fanns:

* Människorna har kommit bort i OO-världen. En genomgång av 15 böcker om OO Analys och Design, utgivna 1992-1995 visade att endast 160 sidor diskuterade användbarhet och användare.

* Det har varit för mycket onödigt metodkrig genom åren. Endast roligt för deltagare på konferensers paneldebatter, men knappast klargörande och fruktbart för användare. Även om markant fokus nu riktas mot UML får vi inte glömma bort andra ansatser. De kanske har lika mycket substans även om de inte lyckats marknadsföra sig lika effektivt.

* Alla mjukvarusystem är verktyg med syftet att hjälpa människor att fullgöra sina uppgifter snabbare, mer korrekt, trevligare, ... Alltså måste vi, oavsett typ av system, veta vad användarna behöver för att konstruera det väl. Tyvärr tycks utvecklare inte (inte heller OO-utvecklare) gilla användare. Människor är ju så ”unpredictable”. ”We advertised for workers but got people”.

  Därför pratar vi gärna om system och gränssnitt som ”user friendly” med betydelsen att det är vi utvecklare som tänkt till lite grann, varit lite bussiga mot användarna. Nästa ambitionsnivå är att vara ”user centered”. Här får

användarna vara med på ett hörn och tycka till. ”User centric” är ett annat fekvent använt honnörsord som lägger ytterligare tyngdpunkt på att lyssna på användarna.

Vad som borde vara primärt är att bedöma hur användare ska nyttja verktygen/systemen för att fullgöra sina arbetsuppgifter, dvs ”usage centered design”. Ett annat namn för detta tycks vara ”purpose centric”. Användarna är knappast intressanta i sig, endast genom att de kan uttrycka sina behov, formulera sin arbetssituation, sin roll. Med detta perspektiv frågar man sig

- What kind of users?

  Här identifieras de användar-roller som är inblandade. En roll är en abstrakt uppfattning av någon som utför något baserat på intressen, förväntningar, beteende, ansvar, ...

- What are they trying to do?

  Beskrivs lämpligen i form av användningsfall. Man bör skilja på essential use cases (som beskriver syftet i en övergripande vad-form) och concrete use cases (som är mer hur-orienterad, dvs på en nivå som vanligtvis förknippas med användningsfall).

- What do they need to do it?

- How do they do it with the system?

* Vi får inte glömma att OO från början utvecklats för konstruktörer och implementerare, inte primärt för användare. Att GUI kommit att förknippas med OO beror på hur gränssnitten normalt realiseras, inte på gränssnittet i sig. Att gränssnittet innehåller symboler som står för någon typ av funktionalitet är ”common sense”, något som vi är vana vid sedan barnsben. Trycker vi på nallens mage piper den, trycker vi på hissknappen kommer hissen, vrider vi på startnyckeln startar bilen, osv. Det är inget specifikt OO i detta. Inte heller tänker vi i termer av att objekt i verkligheten skickar meddelanden mellan varandra. Detta är en teknologisk syn. För övrigt är de flesta användargränssnitt alldeles överlastade och tillämpar fåniga metaforer med långt ifrån självklar innebörd. En utvecklarens kreativa lekstuga.

2.4.3 Paneldiskussioner

Förutom Alexander och Constantine, berörde ett par paneldiskussioner de mjuka aspekterna:

* Soft Issues and Hard Problems in Software Development
* OO Anthropology: Crossing the Chasm

Många tänkvärdheter att ta till sig och fundera vidare på presenterades. Av dem har jag valt ut följande, framförd av Norman Kerth:

OO och andra nya discipliner, som hävdar så kallade paradigmskiften, skapar oro och misstänksamhet bland dem som representerar vad som stämplas som gammal, inaktuell kunskap. Dessa måste nu, kanske mer som en följd av trendvindar än sakliga skäl, ge upp sin yrkeskompetens för något diffust nytt. Kerth formulerade problemsituationen på följande tänkvärda sätt:

”These (considered old) skills are used in our livelihood and often contribute to our identity. Thus these skills are very important, and not set aside easily. They certainly are not given up without experiencing a range of emotions:

* fear of incompetence;

* longing for the old way;

* frustration over not knowing how to proceed;

* anger at the methodologists, the authors, the OOPSLA ”experts” and tool vendors who claim the transition is easier than it is;

* distrust over methodologies that change every year or so;

* mystification of objects and its vocabulary, compared to their known way of working;

* trapped into a way that makes no sense, and an observation that the ”experts” have no responsibility for the success of the project - it is the responsibility of self.”

För att nyansera bilden tillägger Kerth:

> ”There is also joy of learning, satisfaction of mastery, and the excitement of discovery.”

Ska vi lyckas föra ut OO-budskapet på bred front måste bland annat dessa mänskliga perspektiv förstås och respekteras.

2.4.4 Schack

Ett anslag av förtröstan om människans fortsatta relevans fick åhörarna från oväntat håll, nämligen från inbjudne talaren Feng-Hsiung Hsu, konstruktören av IBMs schackdator ‘Deep Blue’ - den mest avancerade datorn i sitt slag för närvarande. Hsu slog vänligt nog hål på detta med datorns överlägsna intelligens genom följande liknelse ”Om en människa tappar en vigselring på stranden letar denne efter ringen genom att följa stegen i sanden bakåt, alternativt hyr en metalldetektor. Med stor sannolikhet återfinns den. Detta är intelligens. Deep Blue skulle istället använda ett stort galler och skaka igenom hela strandens sand till ringen hittas. Om jobbet kan utföras med en våldsam hastighet, kan sökningen gå fortare än människans. I alla händelser återfinns förr eller senare ringen. Detta är råstyrka.” Som ett bevis konstaterade Hsu att

människan, i form av schackmästaren Kasparov, i somras vann en massmedialt uppmärksammad kamp mot Deep Blue. Den här gången.

Undrar om kommande generationer superintelligenta datorer någonsin kommer på idén att konstruera människor?

3. Till sist

Några uppsnappade tänkvärdheter:

* "If you have the interest, you have the capacity."

* ”Make sure there’s a good fit between the business problem you’re trying to solve and the technology you’re proposing as a solution. Just because you have a hammer in your toolbox doesn’t mean every problem is a nail.”

* ”A fool with a tool is still a fool.”

* ”The systems development 90-10 rule: If 10% of the most complex, detailed portion of a method is deferred and the focus is placed on the simpler 90%, then understanding becomes much easier. Stated otherwise, 90% of the pay-off can be achieved with 10% of the effort, and learners have a much greater chance of initial success.”

* På tal om återanvändning och mjukvaruindustri: Bard Cox’s artikel från Software Magazine 1990, ”Software technologies of the 1990’s”, är i stort sett lika tänkvärd idag som då:

  ”The denizens of the software domain from the tiniest expression to the largest application, are as intangible as any ghost. And because we invent them all from first principles, everything we encounter there is unique and unfamiliar, composed of components that have never been seen before and will never be seen again, and that obeys laws that don’t generalize to future encounters. Software is a place where deams are planted and nightmares harvested, an abstract mystical swamp where terrible demons compete with magical panaceas, a world of werewolves and silver bullets. As long as all we can know for certain is the code we ourselves wrote during last week or so, mystical belief will reign over quantifiable reason.”

---> OOPSLA 97 är planerad till 5-9 oktober i Atlanta, Georgia.