TRIAD

Rapport K 25

Introduktion till

GDMO-STANDARDERNA

Stig Berild

TRIAD-PROJEKTET DRIVS GEMENSAMT AV TELIA, POSTEN, STATSKONTORET, ERICSSON OCH SISU, SVENSKA INSTITUTET FÖR SYSTEMUTVECKLING. FÖR VIDARE INFORMATION KONTAKTA SISU, ELECTRUM 212, 164 40 KISTA, TEL 08-752 16 00, FAX 08-752 68 00.

Rapporterna beställs frå
SISU • Electrum 212 • 164 40 Kista • Fax 08-752 68 0
Rapporterna är endast tillgängliga för TRIAD-parterna och är avgiftsfri

Rapportdisposition

ISO/IEC-standarderna 10165-1 och 10165-4 handlar i ett generellt perspektiv om att etablera förutsättningar för utbyte av data och direktiv mellan fristående system. Detta är en bred och angelägen frågeställning och av den anledningen har rapporten delats in i en del A som diskuterar det generella perspektivet och en del B som beskriver de aktuella standardernas sätt att angripa problemet.

Del A: Datautbyte; förutsättningar och principer

Del A innehåller en allmän introduktion till datautbyte. Exempel har medvetet tagits från alldagliga situationer, i ett försök att beskriva en allmän problemställning. Att direkt gå in på Managed Objects, Agents, m m innebär en risk att alltför snabbt hamna i en teknikorienterad detaljdiskussion utan att först ha etablerat en grundläggande referensram. Detta vill jag undvika. Risken att "tala förbi varandra" är ändå uppenbar inom detta område där olika synsätt, modeller och traditioner (datamodeller, objektorienterade modeller, modeller för öppna system, modeller för datautbyte mm) till vissa delar behöver samordnas och inordnas för att tillgodose gemensamma eller åtminstone överlappande behov.

Del B: Introduktion till GDMO-standarderna

Del B ger en introduktion till några av de standarder inom OSI som återfinns under den gemensamma rubriken "Structure of Management Information", i dagligt tal kallat standarder för "Managed Objects" eller "GDMO-standarden". GDMO är en förkortning av Guidelines for Design of Managed Objects. Dessa guidelines utgör i själva verket bara en av för närvarande sex standard-dokument inom området. De har det gemensamma ISO/IEC-numret 10165 med under-numrering från 1 till 7 (nr 3 saknas). Del B inriktar sig framförallt mot delarna 1 och 4, eftersom dessa beskriver "the Management Information Model", d v s den modell som reglerar hur den giltiga semantiken kan formuleras i samband med datautbyte .

Standarderna har tagits fram under lång tid och av olika grupper av personer. Följden har blivit en hel del motstridande budskap och formuleringar. Utrymme finns dessutom för olika tolkningar, något som tydligt kan märkas i artiklar och böcker inom området. Även tolkningar och synpunkter i denna rapport är subjektiva. Desutom är de baserade på en generell modelleringsbakgrund snarare än på kunskap inom tillämpningsområdet. Introduktion till tillämpnings-området och synpunkter på en tidigare version av rapporten har författaren, på ett alldeles utmärkt och tålmodigt sätt, fått av Erik Jonsson, Björn Norén och Oscar Bravo, samtliga från Telia.

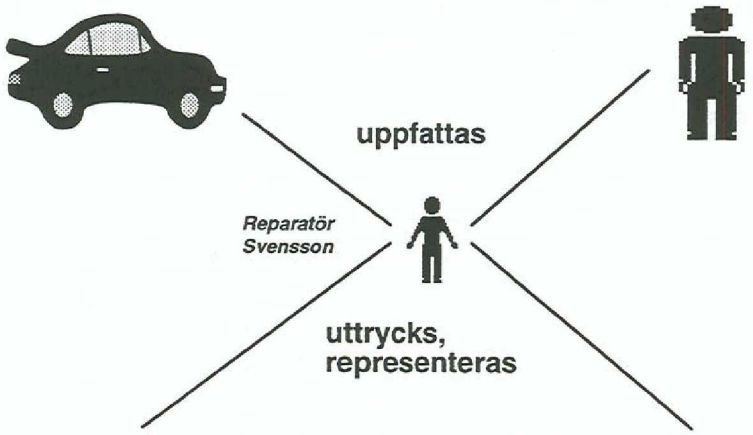
A. Datautbyte; förutsättningar och principer

1. Modeller

1.1 Verklighetsmodell; en verklighetsuppfattning

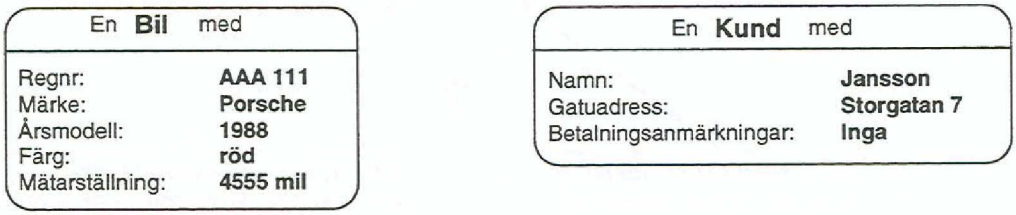
Vi bildar oss alla uppfattningar om världen eller verkligheten omkring oss. Vi ägnar oss också åt att drömma och fantisera, åt visioner och förhoppningar. I datamodelleringssammanhang brukar man kalla dessa iakttagna, upplevda, tänkta abstrakta eller konkreta företeelser för "verkligheten", eller i engelsk terminologi för Universe of Discourse (UoD). Ibland håller vi dessa uppfattningar för oss själva, ibland har vi behov av att förmedla uppfattningarna till andra, d v s utbyta information. Detta utbyte kan föras fram på olika sätt (tal, skriven text, bilder, ...) och enligt vissa regler (svenska språket, grafiska symbolsammansättningar, ...). Syftet med utbytet är att vi vill *förmedla* våra uppfattningar – vi vill att mottagaren ska förstå det vi förmedlar. Där börjar problemen. Vilka förutsättningar måste då vara uppfyllda för att ett vettigt utbyte av information ska kunna komma till stånd?

Figur 1 visar reparatör Svensson i färd med att uppfatta en viss verklighet. Observera, att uppfattningar alltid är subjektiva och alltid styrs av ett visst syfte. Svensson väljer att uttrycka sin aktuella uppfattning av denna verklighet med hjälp av textsymboler ("AAA 111", "Porsche", "röd", ...) och i en kombination som svarar mot reglerna i den svenska språkläran. Kanske skriver han ner det på papper, kanske säger han det muntligt till sin kollega.



Figur 1

Kanske väljer han, istället för svenska språkläran, att uttrycka sig i halvgrafisk formalism enligt figur 2.



Figur 2

1.2. Verksamhetsmodell; en verklighetsabstraktion

Oavsett vilken uttrycksform som används har antagligen inte mottagaren några svårigheter att förstå "budskapet". Varför då? Jo, mottagaren är förmodligen beredd på i stort sett vad det är Svensson brukar prata om, d v s känner till och förstår det aktuella ämnesområdet (context). Dessutom har Svensson inte bara formulerat symboler för respektive verklighetsföreteelse. Han har varit vänlig nog att inkludera "hjälptexter" som "Bil", "Märke", "Färg" m.fl. Dessa hjälptexter uttrycker en abstraktion, d v s en allmän, generaliserad uppfattning om den aktuella verkligheten. Att abstrahera är en helt naturlig, ofta omedveten mental process. Uppfattat likartade företeelser grupperas in under en viss gemensam typ av företeelse. En viss typ uttrycks i allmänhet genom en textsymbol, ett begrepp, t ex "Bil" eller "Färg".

Visst är det större chans att förstå

"Bilen AAA111 är av märket Porsche av årsmodell 1988 och med mätarställningen 4555 mil. Färgen är röd och gul."

än

"AAA111; Porsche, 1988. Röd, gul.4555 mil.".

Inte ens om vi lägger till stödord med sina givna roller enligt språkläran kan vi vara säkra på att entydigt ha förstått budskapet:

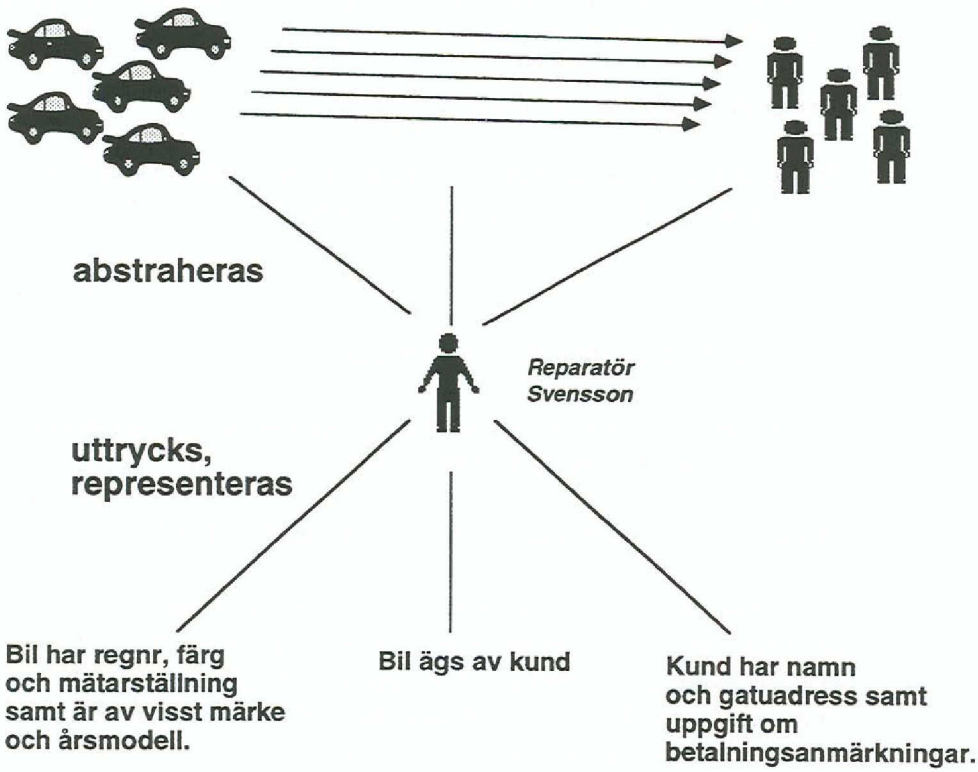
"AAA111 är en röd och gul Porsche från 1988 med 4555 mil."

Att vi ändå kanske anar innebörden i de två senare utsagorna beror antagligen dels på att vissa specifika symboler ger associationer till vad det är fråga om dels på att symbolernas inplacering som satsdelar vägleder. De flesta vet till exempel att Porsche är ett bilmärke. "Gul" förekommer som satsdelen attribut varigenom man kan förstå att det beskriver något. O s v.

Antagligen är inte utgångsläget i figur 3 lika lätt.

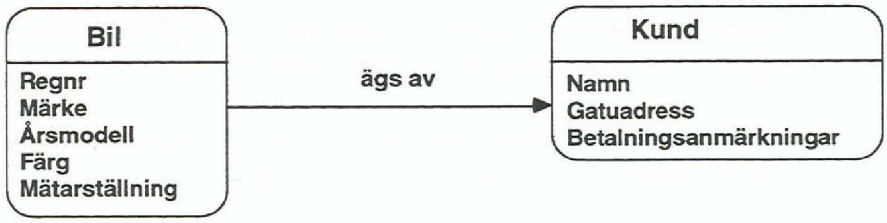
Figur 3

Tillbaka till Svensson. Låt oss anta att Svensson väljer att formulera sina abstraktioner av den aktuella verkligheten med hjälp av textsymboler enligt figur 4. Kanske vill han tala om sin allmänna uppfattning för någon för att denna person bättre ska kunna förstå när han i fortsättningen pratar om något inom samma område.



Figur 4

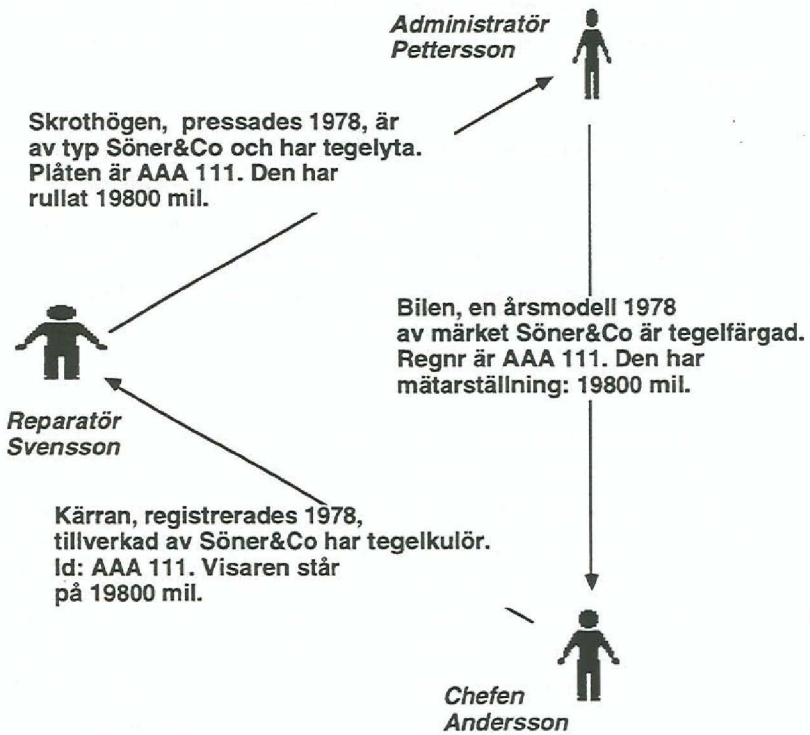
Svenska språket ersatt med grafiska symboler kan se ut som i figur 5. Figuren är exempel på en mycket enkel verksamhetsmodell.



Figur 5

1.3 Begreppsmodell; en abstraktion av abstraktionen

Är då alla problem undanröjda i och med att vi givit namn åt våra abstraktioner eller typer? Titta på figur 6. Svensson, Pettersson och Andersson pratar visserligen med varann men budskapen går inte fram. Varför? Därför att de antagligen har abstraherat verkligheten olika. Åtminstone använder de inte samma begrepp på sina abstraktioner (typer). Vem vet om det som Svensson karakteriserar som 'skrothögar' är samma företeelser som det Andersson kallar 'kärror', om det Svensson kallar 'typ' svarar mot det Pettersson kallar 'märke', o s v?



Figur 6

Enda sättet att undvika missförstånd är att tala om vilka begrepp man använder sig av och vad de står för, d v s tala om hur ens verklighetsabstraktion ser ut. Hittills har vi haft behov av att prata om individuella företeelser i verkligheten. Nu har vi dessutom behov av att prata om abstraktioner. I grunden gäller samma resonemang som tidigare men med den skillnaden att "verkligheten" nu är våra abstraktioner. Om Pettersson påstår att hans abstraktion är

"Regnr, Färg, Årsmodell, Märke, Mätarställning, Bil, Rullande med fyra hjul, Kund, Namn, Gatuadress, Betalningsanmärkningar, ägs av"

ger det sannolikt lika lite utbyte som att formulera "AAA111; Porsche. Röd, gul. Sladd." när man egentligen menar "En bil med regnr AAA111 är av märket Porsche, har färgerna röd och gul och ägs av en kund vars namn är Sladd". Det behövs återigen "hjälptexter" för att förklara vad de olika symbolerna står för och hur det hela hänger ihop.

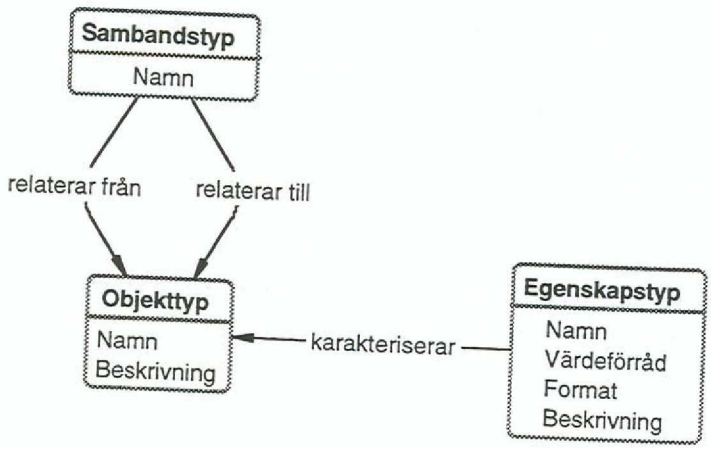
Visserligen är redan sättet att formulera abstraktionerna i figur 4 och 5 en hjälp för att förstå att Bil är någon annan typ av företeelse än t ex Färg ("Bil har ..." i texten resp Bil överst i figuren) men det framgår inte vad Svensson tycker att ordet Bil representerar.

Regnr, Färg, Årsmodell, Märke, Mätarställning är var och en ***namn*** *på* ***egenskaps-typer*** *som kan användas för att* ***karakterisera*** *en* ***objekttyp*** *med* ***namnet*** *Bil som dessutom kan* ***beskrivas*** *enligt "Rullande med fyra hjul". Egenskapstyperna med namnen Namn, Gatuadress och Betalningsanmärkningar karakteriserar objekttypen Kund. Objekttypen Bil är* ***relaterad till*** *objekttypen Kund genom* ***sambandstypen*** *med* ***namnet*** *"ägs av"*

är sannolikt mer klargörande. Visst kan man kanske tycka att det är lite fånigt övertydligt, men det tycker vi nog bara så länge vi väl känner till den värld som abstraheras samt när de begrepp som används är allmänt välkända och entydiga. Så fort vi börjar prata om mer en komplex verklighet med abstraktioner som omfattar många, kanske delvis nya begrepp och dessutom uttryckta av någon vi inte känner, än mindre hans eller hennes syften, blir den stabila marken snart kvicksand.

I figur 5 fick verklighetsabstraktioner namn och de relaterades till varandra genom en grafisk notation. Samma grafiska notation kan användas för att uttrycka den klargörande texten ovan som är en abstraktion av abstraktioner. Det finns t ex likartade företeelser som vi grupperar under begreppet objekttyp (Bil och Kund), andra som hänförs till begreppet egenskapstyp (t ex Märke, Färg, Gatuadress) och ytterligare andra som vi kallar sambandstyp (ägs av). På samma sätt som Bil beskrivs med bl a Märke på första abstraktionsnivån, beskrivs Egenskapstyp med bl a Värdeförråd samt relateras till den Objekttyp som beskrivs, på nästa abstraktionsnivå. Hur beskrivningsbegreppen hänger ihop enligt Svenssons uppfattning åskådliggörs i figur 7. Figur 7 är en **begrepps-modell** som uttrycks med hjälp av grafisk notation. Samma begreppsmodell skulle också kunna formuleras i i svenska språket eller någon annan lämplig

syntax om så önskas. Den grafiska notationen ger i allmänhet en bra överblick i ett kompakt format.



Figur 7

Om Svensson, Pettersson och Andersson kunde tala om för varann hur deras respektive abstraktioner ser ut med hjälp av samma begreppsuppsättning (t ex den som redovisas figur 7), ger de varandra en bättre chans att korrekt tolka varandras utsagor. Eftersom de då har ett enhetligt språk (som alla kan) för att uttrycka abstraktioner, är det bara att gå till avsändarens abstraktion och se vad en "hjälptext" på första abstraktionsnivån betyder.

Exempel:

> *Svensson kan gå in i Anderssons abstraktion och där se att Anderssons 'Kärran' är namnet på en objekttyp som beskrivs som "En gammal bil, som inte längre är värd att vagnskadeförsäkra." Där framgår också att kärra bla karakteriseras med hjälp av egenskapstypen 'Id', beskriven som "En unik identifikation som erhålls i samband med registrering." med syntaxen "<Bokstav> <Bokstav> <Bokstav> <Siffra> <Siffra> <Siffra>". Det är nu ganska lätt för Svensson att konstatera att 'Kärra' är detsamma som det egna 'Skrothög'. Alternativt kanske Svensson mer exakt konstaterar att "en Kärra från 1980 eller tidigare och som rullat minst 18000 mil" motsvarar det egna 'Skrothög'. Den egna egenskapstypen 'Plåten' verkar vara exakt samma sak som Anderssons 'Id'. Osv.*

Denna möjlighet att tolka en utsaga betyder givetvis inte att Svensson måste göra avkall på sin egen uppfattning, bara att han nu fått reella förutsättningar att tolka inkommande utsagor, så länge han vet varifrån de kommer.

Det finns dock en kvarvarande risk. De kanske inte förstår samma sak med begreppsmodellens begrepp. Det Andersson ser som objekttyp kanske Svensson kallar entitetstyp, eller värre, egenskapstyp? Är det olika namn på samma sak eller betyder de helt olika saker? Bara en nyansskillnad?

Antalet begrepp för att uttrycka en verksamhetsmodell är bara en bråkdel av verksamhetsmodellens begrepp. I realiteten behövs normalt bara ett fåtal begrepp. Nog borde väl Svensson, Pettersson och Andersson kunna komma överens om dessa så att de har chansen att på ett entydigt sätt utbyta information om sina respektive verksamhetsmodeller? Kanske kunde begreppen i figur 7 vara en bra start? Ett exempel på en synnerligen väletablerad och standardiserad begreppsmodell är den i databassammanhang vanliga relationsmodellen (begreppsdelen av SQL). SQL-begreppen tycks dock mer tilltala system-konstruktörer och programmerare än "gemene man".

De aktuella standarddokumenten 10165-1 och 10165-4 har som huvudsyfte att definiera en begreppsmodell för formulering av Management Information inom Open Systems Interconnection. Begreppsmodellen kallas för Management Information Model.

1.4 Modellnivåer enligt ISO-standarden ISO/JTC1 10027

Modellnivåer enligt resonemanget ovan finns bla beskrivna och definierade i en ISO-standard inom området resurskataloger med beteckning ISO/JTC1 10027 "Information Resource Dictionary System (IRDS) Framework", 1990. Synsättet i denna standard är direkt tillämpbart inom Systems Management-området. Jag har valt att använda svenska namn på de olika modelltyperna. Standardens motsvarande begrepp ges inom parentes. Tyvärr är både bristen på distinktion och språkförbistringen stor inom detta område.

I ISO/JTCI är data om en betraktad verklighet (Universe of Discourse – UoD) formulerad i en **verklighetsmodell** (application database, d v s på application level). En företeelse i verkligheten blir en dataavbild i verklighetsmodellen. Exempel:*bilen med registreringsnumret AAA 111, färgen röd, registreringsnumret AAA 111.*

Betraktas verkligheten från ett generellt perspektiv, kan detta formuleras som en abstraktion av verkligheten i en **verksamhetsmodell** (Information Resource Dictionary Model eller på IRD level). Hit hör t ex *objekttypen bil* och *egenskaps-typen färg.*

Betraktar man nu istället abstraktionen d v s verksamhetsmodellen från ett generellt perspektiv, kan detta formuleras i form av en abstraktion av abstrak-tionen i en **begreppsmodell** (IRD Definition model eller på IRD Definition level). Hit hör exempelvis *begreppet objekttyp* och *begreppet egenskapstyp.*

Ibland finns behov av att betrakta även begreppsmodellen från ett abstrakt perspektiv. På denna nivå finns den så kallade **fundamentala modellen** (IRD Definition Schema model). Här förklaras t ex själva ordet *begrepp.*

2. Datautbyte

2.1 Överenskommelser

Om reparatör Svensson och administratör Pettersson ska kunna utbyta information måste de först komma överens om ett antal saker:

a. En uppsättning begrepp (begreppsmodell) med vars hjälp de kan diskutera sina respektive enskilda verksamhetsmodeller och som också kan användas för att upprätta den gemensamma verksamhetsmodell (se b, nedan) som ska gälla i samband med överföringar.

b. Den verksamhetsmodell som ska användas för utbytet, d v s de begrepp som behövs för att entydigt förklara de data som skickas, tillsammans med principer och villkor för hur dessa begrepp "hänger ihop". (T ex kanske en bil alltid måste beskrivas med hjälp av regnr medan färg är valfritt. En viss färg å andra sidan skickas aldrig utan att den kan hänföras till en viss bil.) Antingen medföljer begreppen varje datautbyte i form av "hjälptexter" eller fördefinieras olika typer av datautbyten till sin uppbyggnad med hjälp av begreppen.

c. De syntaktiska konstruktioner som ska utnyttjas så att respektive komponent entydigt kan packeteras och lika entydigt kan packas upp och återfinnas.

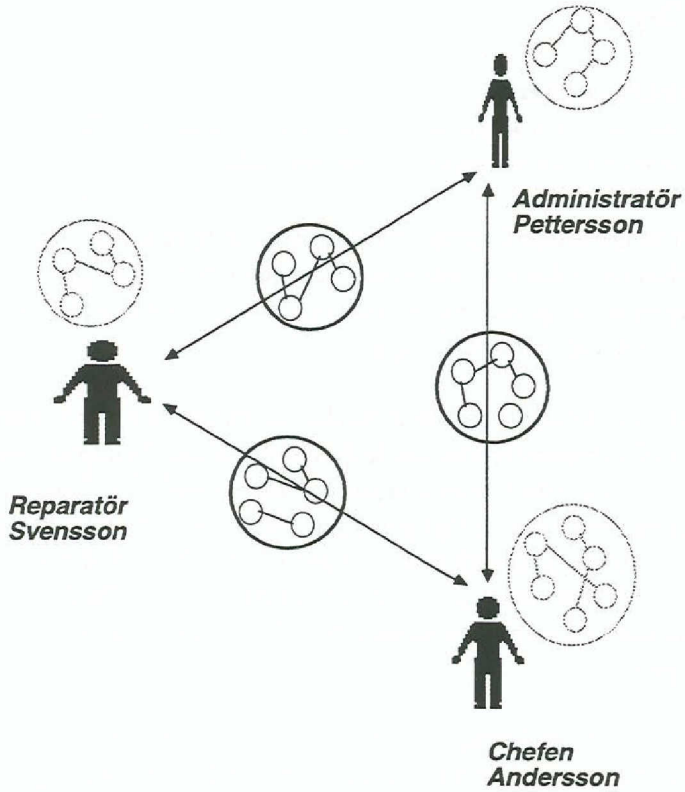
d. Kodningsprinciper för de symboler som används för att uttrycka verklighetsmodellen. T ex. kanske färg uttrycks genom sifferkoder där 1 betyder grön, 2 röd o s v.

e. Principer för hur ett datautbyte rent utförandemässigt ska gå till – umgängesformer, exempelvis hur man tar kontakt, hur man avslutar och hur man hanterar störningar.

C, d och e brukar gå under det gemensamma namnet protokoll. Ett protokoll är m.a.o. en överenskommelse eller uppsättning regler som två eller flera kommunicerande enheter använder för att strukturera och genomföra sin konversation.

Bilaterala överenskommelser

Samma typ av förhandling och överenskommelse måste Svensson ha med Andersson och Andersson med Pettersson. Se figur 2.1. Tillsammans behövs tre bilaterala överenskommelser.



Figur 8

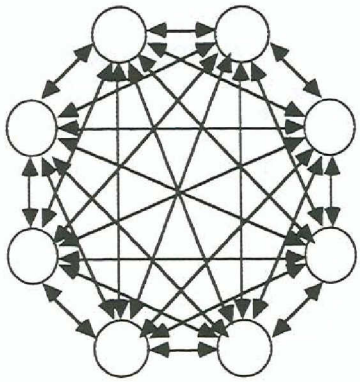
Vi måste hela tiden komma ihåg att det som förhandlas är principer i samband med datautbytet. Hur Svensson själv väljer att se och definiera sina abstraktioner på sitt eget kontor är Svenssons privata ensak. Däremot måste givetvis Svensson kunna förstå både sin egen och den begreppmodell som överenskommits ska gälla för överföringen med den andra parten. Annars har han ju inga möjligheter att översätta sin egen verksamhetsmodell till/från den verksamhetsmodell som gäller för utbytet och därmed indirekt inte möjlighet att definiera vad det är för typer av data som skickas/emottas.

Jämför med översättning av data strukturerade enligt relationsmodellen till/från data strukturerade enligt en Entity-Relationshipmodell. En flervärdig attributtyp i en ER-modell kanske behöver uttryckas som en egen relation i relationsmodellen, o s v.

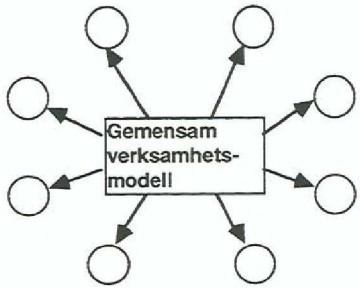
Gemensam "plattform"

Bilaterala avtal i all ära, men börjar det bli många som ska utbyta data ökar det totala antalet erforderliga överenskommelser snabbt. Kunde alla istället komma överens om en gemensam begreppsmodell och tillika en gemensam verksamhetsmodell uttryckt i termer av den gemensamma begreppsmodellen, blev det genast mer hanterligt.

Antag att Svensson är en av ringarna i figur 9 a. Han har där att förstå och översätta till/från sju andra verksamhetsmodeller, antagligen var och en uttryckt med hjälp av sin variant av begreppsmodell. Om det istället hade funnits en given gemensam verksamhetsmodell, uttryckt i enlighet med en viss överenskommen begreppsmodell, kunde Svensson i princip utbyta data med tusen andra utan ansträngning. Allt han behöver kunna är att översätta till/från den lokala till den gemensamma verksamhetsmodellen. Se figur 9 b.

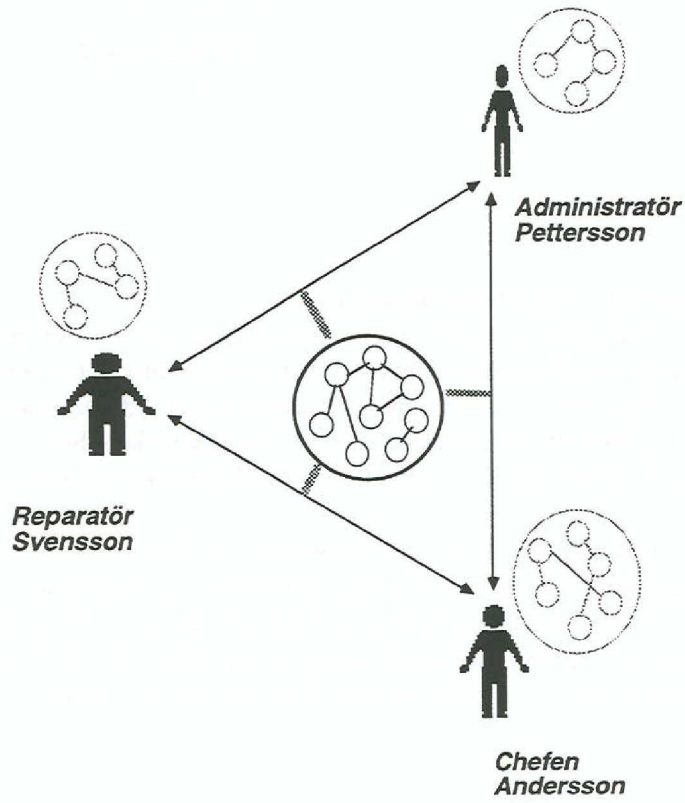


Figur 9 a



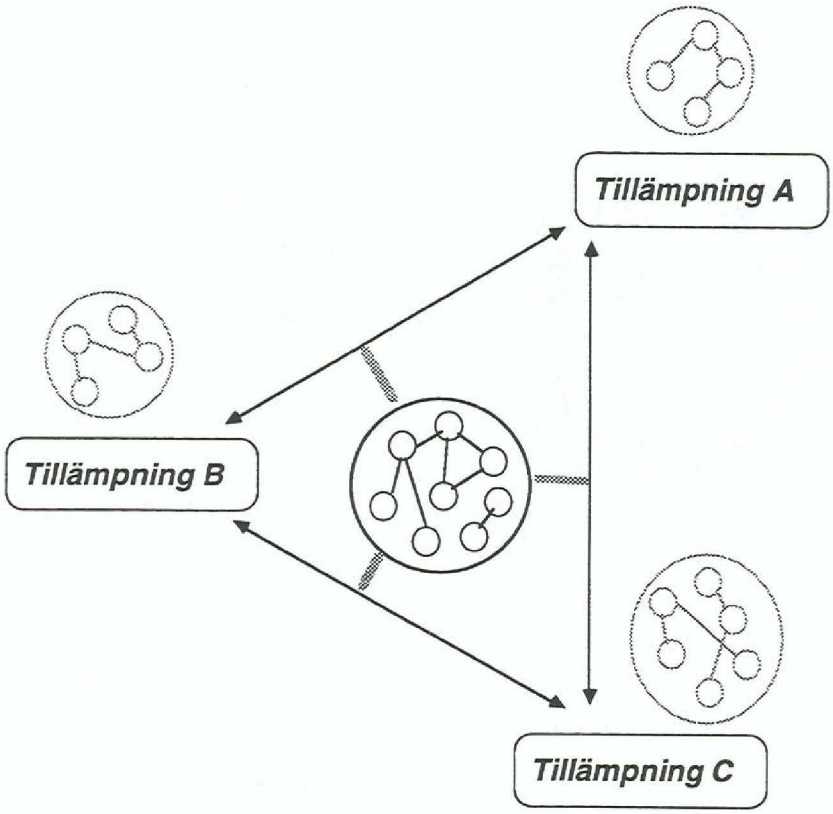
Figur 9 b

Alltså eftersträvar vi om möjligt en situation enligt figur 10, istället för figur 8.



Figur 10

I en modern informationsmiljö är det oftast tillämpningar av olika slag som direkt utbyter data. Samma förutsättningar och villkor gäller även för dem. Se figur 11.



Figur 11

2.4 Datautbyte – ett generellt problem

I en alltmer informationsintensiv och integrerad värld ökar behovet på olika typer av datautbyten. Dels finns behov av utbyte mellan olika mer eller mindre fristående tillämpningar inom en och samma verksamhet. I det fallet är det en intern angelägenhet att definiera en eller flera lämpliga verksamhetsmodeller. Dels finns behovet mellan olika mer eller mindre fristående verksamheter.

Företag och organisationer lever inte sina isolerade liv. De samverkar på olika sätt. En ingrediens i denna samverkan är datautbyte. Så länge man skickar pappersdokument, som fakturor o dyl, är det ganska lätt att tolka innehållet eftersom man av lång hävd valt att gruppera olika uppgifter på dokumentet på ett enhetligt sätt och med givna platser. Dessutom finns i allmänhet goda ledtexter.

I ökande utsträckning utbyts numer data på elektronisk väg. Med ökade volymer och utbyten, utan det mänskliga ögat vakande över det hela, ökar kravet på enhetlighet och kvalitet i processen. Misstag måste med alla medel undvikas. Kravet på standarder får genomslag. Exempel därvidlag kan hämtas från många olika tillämpningsområden, bla:

- Handelsdata (EDIFACT)
- System- och systemutvecklingsdata (CDIF)
- Produktdata (PDES/STEP)
- Geografiska data (arbete inom STANLI och CEN)
- Systems Management (GDMO)

I grunden är samma angreppssätt tillämpbart oavsett tillämpningsområde. Det behövs överenskomna verksamhetsmodeller uttryckta i termer som finns definierade i en lika överenskommen begreppsmodell för att förklara de verklighetsmodelldata som utbyts. Verklighetsmodellerna är specifika för respektive avsändare. Verksamhetsmodell är specifik per tillämpningsområde eller del av tillämpningsområde. Däremot finns egentligen ingen anledning att ha olika begreppsmodeller eftersom ju en sådan, oavsett tillämpningsområde, är en abstraktion av verksamhetsmodeller. Om verksamhetsmodellen innehåller verklighetsabstraktionen Bil eller Modem eller Leverantör spelar i det perspektivet ingen roll. De är alla tre exempel på objekttyper. (Jämför med SQL, som ju kan användas för att hantera data i databaser från många olika tillämpningsområden.)

Dock kan det finnas skillnader i syftet med datautbytet.

(A) I det enkla fallet är det fråga om "Varsågod, här kommer några intressanta uppgifter".

(B) I ett något mer komplext utbyte kan det vara fråga om en begäran om uppgifter där begäran är knuten till vissa villkor, kanske formulerade med hjälp av utdrag ur verklighets- och verksamhetsmodellen. "Ge mig regnr och märke på de bilar som kunden med namnet Sladd äger."

(C) Ibland kan det vara fråga om ett direktiv eller en begärd åtgärd: "Kunden med namnet Sladd har inte betalat. Skicka betalningsanmodan. Sätt inkassobyrån på honom efter tre veckor, om det behövs.". 'Skicka betalningsanmodan' och 'aktivera inkassobyrån' riktar sig båda till någon handläggare, som sedan ser till att kund och inkassobyrå blir vidtalade. Begäran är indirekt. Om och i så fall hur kund och inkassobyrå aktiveras är utanför avsändarens kontroll.

(D) Till sist kan det vara fråga om "raka rör", d v s en begäran som skickas direkt till den som det berör, för reaktion. Mottagaren, liksom alla övriga inblandade, har deklarerat vilket umgänge i form av erbjudna data eller beteenden denne accepterar och tar ansvar för. Samspelet kan därför ses som reglerat och under kontrollerbara former. Mellanhänder saknas. Var och en företräder sig själv. Detta är situationen i system byggda enligt så kallad objektorienterad teknik.

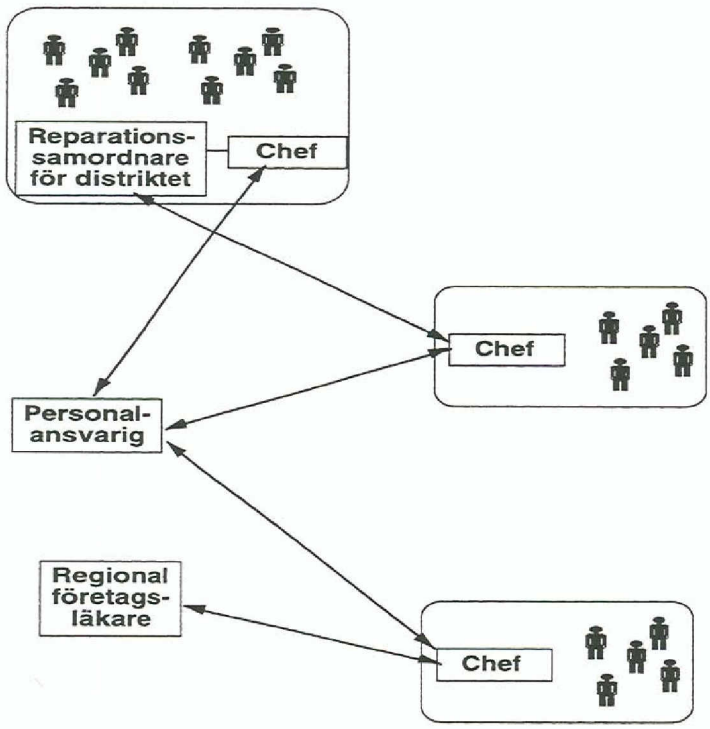
Utgångspunkten för GDMO-standarderna har varit förutsättning C ovan. Denna förutsättning finns beskriven i standarden ISO/IEC 10040 "Systems Management Overview".

B. Introduktion till GDMO-standarderna

1. Återblick på bilverkstaden

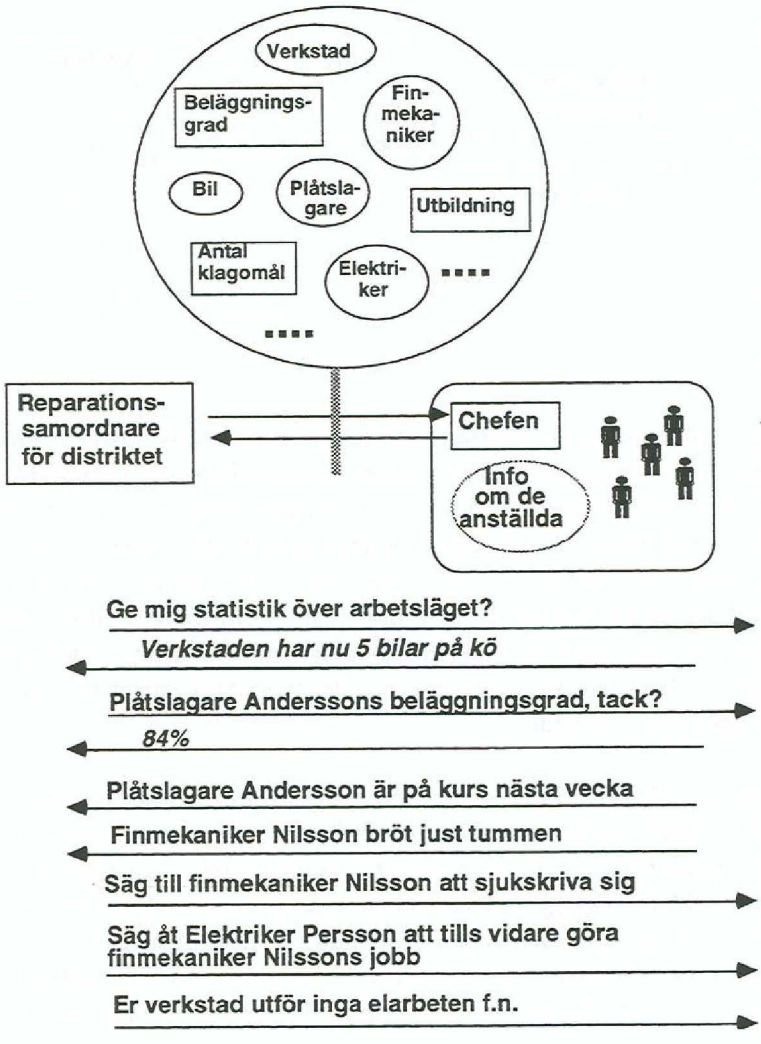
Inledningsvis introduceras här översiktligt ett exempel på ett öppet system taget från ett alldagligt tillämpningsområde. Syftet är att introducera den grundfilosofi som systems management-standarderna baseras på, utan att direkt riskera hamna i områdesspecifika detaljer. I den mån den alldagliga situationen överensstämmer med, eller åtminstone är tillräckligt snarlik, ett open system ur ett systems management-perspektiv, kan den ju alltid användas som ett referensexempel när man känner behov av "kliva ett steg tillbaka" och hitta exempel, som inte är så belastade av historia, fackområde, omständigheter och tekniska förutsättningar.

Vi återknyter bekantskapen med bilverkstaden, men i ett nu betydligt utökat skick. Där finns flera samarbetande verkstäder, var och en med en ansvarig chef. Där finns också olika gemensamma ansvarsområden tilldelade olika personer vid olika verkstäder (ex v reparationssamordnare) eller på annan plats (ex v företagsläkare).



Figur 1

För varje verkstad gäller regeln att det är chefen som bestämmer. Chefen håller ett vakande öga på vad som händer och sker i den egna verkstaden, styr det interna arbetet samt håller kontakten med omgivningen. En chef för en verkstad kan samtidigt ha en mer övergripande roll i verksamheten, exv som Reparationssamordnare för ett visst distrikt. För att kunna utföra en övergripande roll behövs uppgifter av olika slag från alla eller vissa av verkstäderna (verklighetsmodell). Uppgifterna avser ju någon del av verksamheten och uttrycks därför lämpligen med hjälp av begreppen i en överenskommen verksamhetsmodell. Några av begreppen och exempel på utbyte av data och direktiv visas i figur 2. Där indikeras också att chefen inte alltid måste gå ut i verkstaden och hämta in uppgifter till ett begärt svar. Ofta kan han basera sina svar på redan tidigare insamlad information ("Info om de anställda"). Den kan ligga i hans huvud, i en anteckningsbok, i en databas.

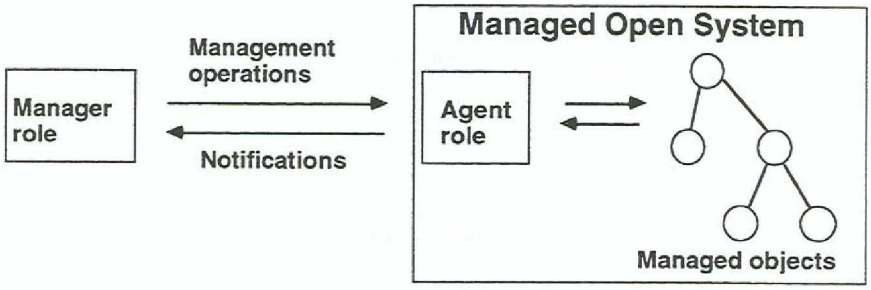


Figur 2

2. Principer för systems management

Bilverkstadsexemplet har stora likheter med systems management. Antag att vi kallar chefen för en **agent** (använder engelsk terminologi) med ansvar för och övervakning av en uppsättning resurser i ett nät (open system), istället för resurser i en verkstad. Antag också att vi kallar en övergripande, styrande roll för **manager**.

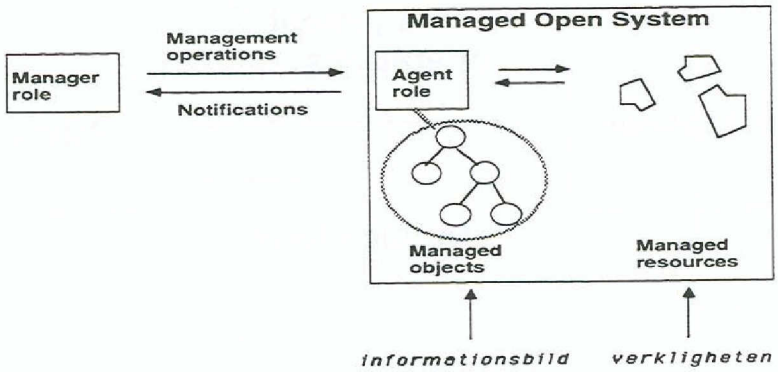
Samverkan sker i form av informationsutbyte och direktiv. En agent informerar en eller flera managers om tillstånd, inträffade händelser och omständigheter inom "sitt" öppna system. En sådan levererad uppgift kallas för en **notification.** Antingen kan det ske på eget initiativ eller som reaktion på en begäran. En manager, å sin sida, begär uppgifter från en eller flera agenter och kan även begära att en agent vidtar vissa åtgärder inom "sitt" öppna system. En sådan begäran kallas för en **management operation.** I den översiktligt beskrivande standarden ISO 10040 uttrycks detta förhållande i en bild, visad i förenklad form i figur 3.



Figur 3

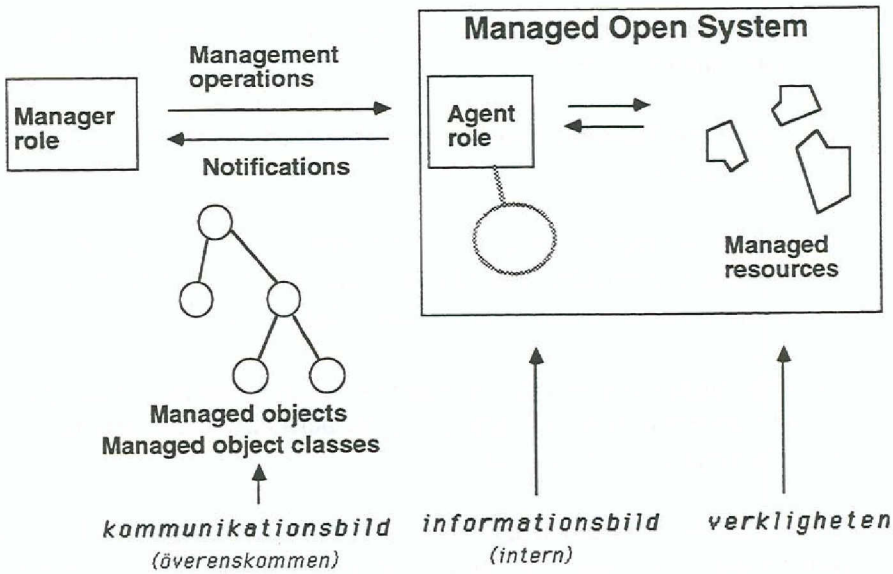
Bilden är något missvisande eftersom man inte gör någon åtskillnad mellan en resurs (resource) och databilden av en resurs **(managed object).** I standardens text framgår dock klart denna distinktion ("a managed object is the abstraction of a resource that represents its properties as seen by management"). Agenten ansvarar alltså internt inom sitt open system för att resurserna övervakas och styrs, samt ser till att han har lämplig information om dessa resurser, deras tillstånd, egenskaper osv i form av en verklighetsmodell. Var denna modell finns, vad den innehåller, hur den är uppbyggd, m m ligger, som tidigare nämnts, inom agentens eget ansvarsområde.

I standarden ISO/IEC 7498-4 definieras begreppet Management Information Base (MIB) som "the set of management objects within an open system" och "the conceptual repository of management information within an open system". Med denna definition ligger det nära till hands att jämställa en MIB med en verklighetsmodell eller informationsavbild av ett öppet systems resurser. Managed Objects hamnar konsekvensmässigt inom agents ansvarsområde och kontroll. Se figur 4. Med tidigare resonemang i minnet kan vi konstatera att detta synsätt är olyckligt och inte överensstämmande med intentionerna för GDMO. Begreppet MIB är otydligt och inte av avgörande betydelse för förståelse av principerna för systems management.



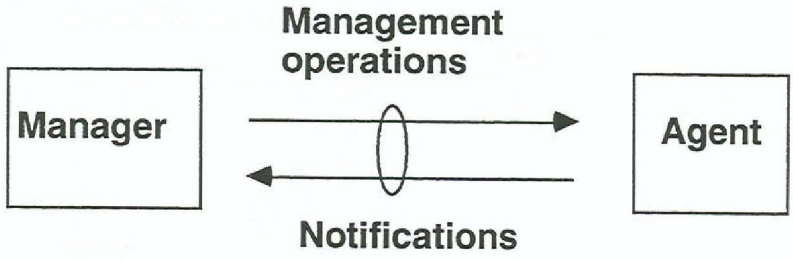
Figur 4

Med andra ord, verklighetsmodellen och dess managed objects tillsammans med en abstraktion i form av en verksamhetsmodell (**managed object classes**) finns i ett GDMO-perspektiv endast till för att entydigt kunna formulera informationsutbyten och direktiv. Hur man internt inom respektive open system ser till att uppfylla sin angivna roll är ett rent lokalt ansvar. Se figur 5.



Figur 5

För det fortsatta resonemanget förenklar vi bilden ytterligare till figur 6.



Figur 6

Av primärt intresse är att se till så att utbytet genom ringen kan formuleras och förstås. Väsentligt för Manager och Agent att komma överens om härvidlag är bla följande:

a. vilken umgängesform som ska gälla, d v s hur förbindelse upprättas, avslutas, hur fel hanteras, m m

b. vilka typer av samverkan som kan förekomma, d v s vad en manager kan begära eller delge en agent (typer av management operations) och vad en agent kan begära och delge en manager (notifications) samt syntax för respektive typ

c. vilka typer av uppgifter som ska kunna refereras och förstås av parterna (verksamhetsmodell)

d. hur de specifika uppgifterna (ur verklighetsmodellen) ska vara kodade (exv enligt ASN.1)

e. vilken begreppsmodell, d v s beskrivningsspråk för att formulera verksamhetsmodeller, som ska användas för entydig definition av den överenskomna verksamhetsmodellen enligt punkt c

GDMO-standarderna specificerar punkterna e och c samt ger anvisningar om hur verksamhetsmodeller tas fram och hur formulär eller mallar (templates) för ifyllande lämpligen bör utformas.

Begreppsmodellen kallas i de aktuella standarderna för **Management Information Model.** Nästa kapitel ger en översikt över modellens centrala begrepp. Kapitel 4 och 5 går igenom de olika typer av operationer som manager respektive agent kan initiera och de begrepp som är intimt kopplade till dessa operationer. Kapitel 6 beskriver ytterligare ett par begrepp. Varje begrepp förklaras och exemplifieras. Begreppen beskrivs i en kompakt grafisk notation för bättre överskådlighet. För introduktion till den grafiska notationen, se lämplig introduktion till Telmod . Grafen kompletteras allteftersom nya begrepp introduceras.

Observera att graferna endast ger en översikt. Syftet är att ta upp de viktigaste begreppen och hur de är relaterade för att visa Management Information Models huvudsakliga uttryckskraft. För en fullständig beskrivning hänvisas till standard-dokumenten.

Avsnitt 7 diskuteras ett begrepp som, enligt min bedömning, har fått en omotiverad tyngd i standarden.

Managed objects måste kunna pekas ut eller refereras. Principer för hur sådana referenser formuleras diskuteras översiktligt i kapitel 8. Kapitel 9 beskriver idén med användning av mallar för att dokumentera komponenter i verksamhets-modellen. Kapitel 10, till sist, innehåller några avslutande reflexioner, bl a om GDMOs framtoning som en objektmodell enligt den så kallade objekt-orienterade ansatsen.

3. Grundläggande begrepp

Till de grundläggande begreppen i Management Information Model (begrepps-modellen) hör "Managed Object Class" och "Attribute Type". Här introduceras de genom att dels beskriva vilka krav som ställs för att något ska anses vara en förekomst under de olika begreppen, dels genom att ge exempel på förekomster.

Observera att följande relatering till tidigare modellresonemang gäller: "Managed Object Class" är ett begrepp i en begreppsmodell. En viss given managed object class tillhör en verksamhetsmodell. En viss given företeelse av en viss given managed object class tillhör en verklighetsmodell.

Begreppen "Superclass" och "Subclass" är två viktiga roller för en Managed Object Class. Rollerna definieras och exemplifieras.

3.1 Managed Object Class

Innan vi går in på klassdefinitionen kan det vara lämpligt att se vad som kännetecknar de företeelser som inordnas i klasser, d v s vad som kännetecknar ett managed object.

Managed Object

Ett managed object är "the abstraction of a resource that represents its properties as seen by management". Därmed menas den uppfattning någon har för något behov, av en företeelse i verkligheten. Om flera ska kunna kommunicera om denna företeelse måste den formuleras i form av överenskomna symboler. Man så att säga utbyter databilder av företeelserna.

Managed object är detsamma som inledningskapitlets "objekt". Tillägget "managed" har antagligen valts för att ge en vinkling mot avsett tillämpnings-område.

Exempel på managed objects kan vara "reparatör Svensson" och "modem X32-H5".

Managed Object Class

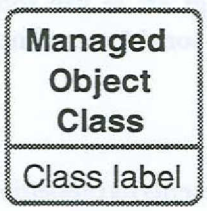
Att standardisera varje enskilt objekt i varje tänkbart open system är varke möjligt eller praktiskt. Att däremot se till att samma begrepp används för samm typ av företeelse är både praktiskt (man skapar en gemensam förståelse för va det är för typ av företeelse man pratar om) och möjligt (antal typer är en bråkd av antalet enskilda objekt).

En managed object class är en abstraktion av en befintlig eller tänkt uppsättnin managed objects som alla upplevs vara tillräckligt lika, givet något visst syfte, fö att klassas under samma generella typtillhörighet. En sådan typ ges i allmänhe ett unikt namn eller en unik beteckning i samband med informationsutbyte. Fö att de olika parterna i ett utbyte ska kunna förstå samma sak med begreppe måste det vara väldefinierat och allmänt överenskommet (standardiserat), någo som just GDMO-standarderna syftar till att underlätta.

ISO/IEC 10040 definierar managed object class enligt följande: "a named se of managed objects sharing the same set of attributes, notifications and mana gement operations". Definitionen refererar till begrepp som förklaras senare denna rapport.

> *Distinktionen mellan type och class är inte entydig. Olika modelleringsansatser väljer olika tolkningar. Inom området objektorientering ser man numer i allmänhet type som den tänkta abstraktionen uttryckt i en verksamhetsmodell medan realiseringen av densamma kallas class. Inom GDMO har man valt en mängdorienterad tolkning av begreppet class.*

Figur 7 visar den första, synnerligen enkla versionen av Management Informa tion Model i grafisk notation. Vi har begreppet "Managed Object Class" me karakteristiken "Class label". I realiteten kan en managed object class beskriv med ett antal olika uppgifter.



Figur 7

Exempel på managed object classes är "reparatör" och "modem" eller korrektar "en managed object class med class label 'reparatör' " respektive "en manage object class med class label 'modem' ".

3.2 Attribute Type

Innan vi går in på typdefinitionen kan det vara lämpligt att se vad som kännetecknar de företeelser som grupperas under samma typ, d v s vad som kännetecknar ett attribute.

Attribute

Attribute = "a property of a managed object"

Därmed menas en uppfattad egenskap eller karaktäristik, som är baserad på något behov. Ett attribute uttrycks vid utbyte i form av någon överenskommen symbol eller databild, ofta som text.

Attribute är detsamma som egenskap. Här har man valt att inte komplettera med prefixet "managed".

Exempel på attributes kan vara "reparatör Svenssons beläggningsgrad 85%" och "modemet X32-H5s baudrate 9600". Som framgår under type-diskussionen nedan kan det även vara enbart "beläggningsgrad 85%" och "baudrate 9600".

Attribute Type

En viss attribute type är en abstraktion av en uppsättning attribut, som uttrycker samma typ av uppgifter, formulerade i form av symboler enligt någon given syntax.

I vissa modelleringsansatser anses tillhörighet till samma object class vara en viktig förutsättning för klassning under samma typ. I andra ansatser har man en betydligt generellare tolkning i och med att det räcker att attributen allmänt upplevs uttrycka samma slags uppgift för att inordnas under samma typ. I det senare perspektivet är t ex "ålder" en attribute type som kan användas för att beskriva både bils ålder, persons ålder eller ålder på arkeologiska fynd. I det förra perspektivet anser man det vara tre olika attribute types eftersom de beskriver tre olika object types och av den anledningen i strikt mening har något olika karakteristik. Exv kanske bils ålder visar sig behöva uttryckas genom värde-alternativen "ny", "begagnad" och "veteranbil", persons ålder genom ett tal 1-120 och arkeologifynden genom ett betydligt större talområde, kompletterat med e Kr eller f Kr.

ISO/IEC 10165-1 definierar attribute type genom "a named definition of a specific kind of attribute, including definitions of its syntax (type) and semantics. An attribute is an instance of an attribute type."

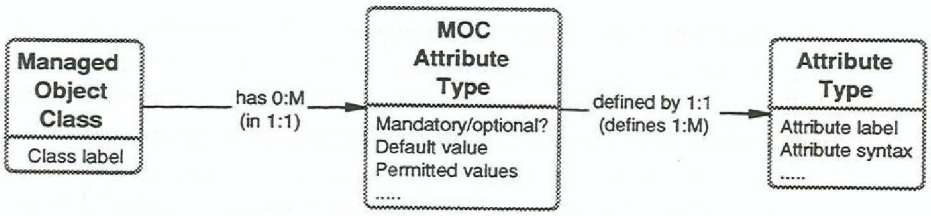
De aktuella standarderna ger olika och ibland motstridiga budskap om attributes. Å ena sidan sägs att det beskriver ett managed object, å andra sidan att samma attribute type kan användas för många managed object types, ibland kompletterat med villkoret att det måste vara använt, d v s minst beskriver en managed object type.

I denna rapport tolkas fortsättningsvis begreppet attribute type som fristående från managed object class. Exempel på attribute types blir då "beläggningsgrad" och "baudrate". En motsvarande konsekvens blir som sagt att ovanstående exempel på attributes omformuleras till "beläggningsgrad 85%" och "baudrate 9600". Se figur 8.

Figur 8

Med denna definition blir attribute type snarlikt det som i andra ansatser går under benämningen domän eller info type.

Utöver att vara fristående definitioner kan attribute types relateras till managed object classes som beskrivningselement. Varje sådan relatering har sin karakteristik vid sidan av den generella under det fristående attribute type. Modellen utökas enligt figur 9.



Figur 9

3.3 Subclass och Superclass

I olika modelleringssammanhang brukar det vara intressant att uttrycka vissa specifika samband mellan klasser. Dels formulerar dessa samband viktiga existerande förhållanden i den bakomliggande verkligheten, dels kan de användas för att underlätta formulering av olika operationer. Till dessa hör hör så kallade sub- och superclass-samband (is-a).

En viss managed object class B är subclass till en annan managed object class A om det i verkligheten är så att man upplever att var och en av de företeelser som man klassificerat in under subklassen B också hör hemma i superklassen A. Resonemanget bygger på att man upplever någon slags neutral företeelse som visar sig spela båda "rollerna". Sedan kan det kanske finnas företeelser som bara tillhör A. Däremot får det inte finnas företeelser som bara tillhör B, för att definitionen ska hålla. Antag att alla som jobbar i bilverkstaden är anställda. Varje sådan "företeelse" tillhör klassen Anställd. Av dessa är några reparatörer,

d v s tillhör klassen Reparatör, andra Plåtslagare, osv. Eftersom varje reparatör också är en anställd gäller att object class Reparatör är subclass till object class Anställd. Sammalunda gäller för Plåtslagare.

Tittar vi från andra hållet gäller att Anställd är en superclass till såväl Reparatör som Plåtslagare.

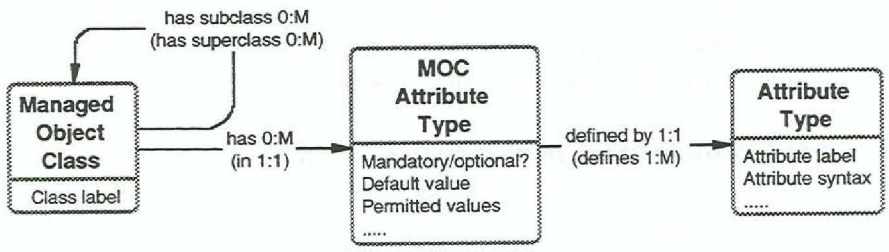
Finessen med det hela kommer i informationsutbytessammanhang. Om vi vet att varje Anställd beskrivs med bla attribute type Namn och de som är reparatörer med bla attribute type Beläggningsgrad kan vi snabbt konstatera att den som är Reparatör både har Namn och Beläggningsgrad medan det kan finnas anställda som inte är reparatörer och alltså har Namn men då inte Beläggningsgrad. Namn finns under Anställd men hör också till Reparatör som en följd av subklass-förhållandet.

Man säger att förekomster i en subclass vid behov ärver attributes från sin motsvarighet i relaterad superclass.

Vår tidigare exemplifierade företeelse har namnet "Svensson" som anställd och beläggningsgraden "85%" som reparatör. Som reparatör ger arvsmekanismen med automatik tillgång även till namnet.

En subclass-struktur sträcker sig ofta över flera nivåer. Exempelvis kan det finnas subklasser Motorreparatör och Växellådereparatör till Reparatör. En Motorreparatör har då, förutom sina egna lokala attribut, tillgång till både namn från Anställd och beläggningsgrad från Reparatör.

I vissa modelleringsansatser tillåts enbart s k trädstrukturer vilket innebär att det endast finns en superclass på varje övernivå att ärva ifrån. Andra ansatser tillåter arv från flera superclasses på varje övernivå (så kallade multipla arv). Detta ger större flexibilitet men kan samtidigt ge upphov till olika typer av namnkonflikter. Inom GDMO har man valt att tillåta multipla arv. Se figur 10 för den kompletterade modellen.



Figur 10

4. Management operations

Några begrepp i GDMO-standarderna är nära kopplade till de olika typer av operationer som en manager kan begära att en agent ska utföra. Därför redogör jag här helt kort för de operationstyper som definieras i ISO/IEC 10165-1.

En operation begärs alltid av en manager och riktas alltid till en eller flera agenter. Däremot är föremålet för operationen antingen ett visst managed object eller ett visst attribute som tillhör ett visst managed object.

4.1 Operationer som berör managed objects

De operationer som berör managed objects omfattar operationstyperna *Create*, *Delete* och *Action*.

Create

Create innebär att agenten ombeds skapa en ny förekomst (managed object) ur en viss angiven Managed Object Class. Samtidigt skapas vissa (i allmänhet obligatoriska) attributvärden enligt regler i klassdefinitionen eller genom parametrar till Create-operationen. Man får utgå ifrån att den just skapade förekomsten i verklighetsmodellen har en upplevd abstrakt eller konkret motsvarighet i verkligheten.

Delete

Delete är en begäran om att agenten tar bort ett visst existerande managed object från verklighetsmodellen. En följdeffekt blir att alla dess attribut försvinner. En annan följdeffekt kan vara att ett antal andra, relaterade managed objects också behöver raderas. Dit hör t ex objects som är subobjects enligt en definierad subclass-struktur och objects som definierats som underordnade i namngivningshänseende (se kapitel 8, sid xx). Om och hur raderingen ska genomföras vid dessa situationer för att upprätthålla enhetlighet i verklighetsmodellen, definieras inom verksamhetsmodellen.

Action

Action används av en manager när något annat än Create eller Delete ska utföras mot ett managed object. Med andra ord avser operationen att en handling eller action, definierad som en del av en Managed Object Class, utförs mot ett visst managed object tillhörigt klassen. Parametrar av olika slag kan behöva bifogas som kompletterande direktiv och för att leverera resultat.

4.2 Operationer som berör attributes till managed objects

Attributoperationerna består av följande typer:
Get attribute value
Replace attribute value
Replace-with-default value
Add member
Remove member

De två första typerna är självförklarande.

När man vill ersätta ett aktuellt attributvärde med ett standardvärde som är fördefinierat under klassdefinitionen, används "Replace-with-default-value".

För de attributes som kan vara flervärdiga måste finnas möjlighet att lägga till och ta bort specifika värden. "Add" och "Remove" används för detta ändamål.

4.3 Diskussion

Som synes är likheten mellan management operations och operationstyper hos enklare former av databashanterare påtaglig. Den överenskomna verksamhetsmodellen motsvarar schemat, agenten motsvarar databashanteraren och managern motsvarar en anropande process eller person. Skillnaden ligger i operationstypen "Action".

Beskrivingen i standarden är något otydlig beträffande innebörden av den action eller åtgärd som refereras genom operationstypen "Action". Klart är dock att åtgärden appliceras på eller gäller för visst managed object.

En användning av "Action" tycks vara att referera till en komplex och sammansatt operation som inte lämpligen kan uttryckas genom en eller flera separata operationer av övriga typer (beroende på villkorskonstruktioner, transaktionsorientering, e.dyl.). En annan troligare användning är för att begära någon typ av följdeffekt eller aktivitet ute hos den bakomliggande resursen.

Med den första infallsvinkeln borde det kunna finnas önskemål om operationer som berör ett antal managed objects som tillhör samma eller olika managed object classes (jmf en Select-sats i SQL). Gäller det senare, verkar det rimligt att orientera en viss action mot endast ett managed object. Vi antar i fortsättningen att det senare synsättet gäller.

Att tala om att object class Reparatör kan ha 'beläggningsgrad' och kan 'byta topplock' är ju inte så väsensskilt. En action kan alltså ses som en egenskap eller karakteristik på samma sätt som ett attribut. I verksamhetsmodellen har man valt att placera action-egenskaperna under en egen rubrik. Detta diskuteras nedan. Egenskaperna måste finnas med för att kontrollera att det är accepterbara operationer som begärs (mot ett visst managed object som tillhör en viss class) och att rätt uppsättning parametrar finns med.

4.4 Begrepp i Management Information Model för att reglera tillåtna management operations

Operationstyperna Create och Delete har sina klara, fördefinierade syften. Vilken handling som avses i samband med en Action-operation bifogas istället som en parameter. Tillåtna handlingar måste kunna definieras. Management Information Model tillhandahåller ett specifikt begrepp för ändamålet.

Action

En Action är en beskrivning med placering under viss managed object class. Action beskriver den typ av beteende som får utföras på objekt tillhöriga klassen.

För varje action anges vilka förutsättningar, parametrar m m som ska vara uppfyllda för att hanteringen ska vara korrekt.

I ISO/IEC 10165-1 definieras action som "an operation on a managed object, the semantics of which are defined as part of the managed object class definition."

Till skillnad från attribute types, som instantieras per managed object, är det alltid en viss action under viss managed object class som refereras i samband med operationen Action. Denna action begärs dock med referens till visst managed object och med undermeningen att ett motsvarande beteende ska äga rum hos den bakomliggande resursen. Ta följande exempel:

För managed object class Reparatör gäller bl a en viss erforderlig skolbakgrund och vissa definierade typer av arbetsuppgifter som reparatören är auktoriserad att utföra. För Reparatör Svensson gäller den generella beskrivningen och kompetensen. Ingen instansiering av dessa uppgifter är aktuell. Jämför skillnaden mot attribute type "beläggningsgrad" där Svensson har sitt värde, Pettersson sitt, o s v. Instansiering skulle också bli aktuell för actions om vi hade att realisera just Svenssons sätt att utföra respektive arbetsuppgift.

Tillägget i grafen visas i figur 11, nedan.

5. Notifications

5.1 Notification som operation

En notification sänds av en agent till en eller flera managers i form av event-reports. Den beskriver någon inträffad händelse eller omständighet i anslutning till ett visst managed object.

En notification kan antagligen avse även information om ett uppkommet tillstånd, där den sista händelsen bara är sista länken i ett antal förutsättningar för att tillståndet ska anses gälla.

Man har valt att begränsa en händelse (och dess ursprung) till endast ett managed object. Anledningen till detta är oklart. I ett generellt perspektiv borde det kunna inträffa någon händelse i ett open system som berör ett antal objekt av samma eller olika klasser.

Eftersom det är agenten som skickar en notification, kan man ana att det också är agenten som avgör när det är dags att skicka densamma. Sättet att kommunicera med de bakomliggande resurserna samt att besluta om när en notification av fördefinierad typ ska skickas är agentens ensak och utanför det externa systemets kontroll så länge aktuell notification överensstämmer med överenskomna villkor (semantik). Utsänd notification ska referera till visst managed object och vara definierad under den managed object class som aktuellt object tillhör. Parametrar kan behövas som kompletterande förklaring.

På samma sätt som action beskriver en möjlig typ av beteende hos en resurs, beskriver en notification en möjlig typ av händelse eller uppkommet tillstånd hos en resurs.

Med samma exempel som under avsnitt 4.3 kan ett nytt tillstånd eller en ny händelse vara att en önskad action blivit utförd eller att något onormalt inträffat. Beläggningsgrad för Reparatör kan t ex stå under agentens uppsikt och vid överskridande av 85% innebära att notification 'överbelastning' skickas tillsammans med bl a parametern beläggningsgrad.

'Sjukskrivning' kan vara en annan tillåten och i gränssnittet accepterad typ av notification för class Reparatör. Sannolikt uppstår den genom att en viss reparatör på något lämpligt, inom det aktuella öppna systemet överenskommet, sätt meddelar agenten om läget. Kanske uppstår en intern sammansatt dialog enligt

reparatör:	"Jag halkade just på en oljefläck"
agent:	"Hur allvarligt är det?"
reparatör:	"Jag bröt antagligen tummen"
agent:	"Ta kontakt med vårdcentralen och gå hem"
agent:	"Jag sjukskriver dig för i morron tills vidare så får vi se vad doktorn säger därefter"

avslutad med notification 'sjukskrivning' till managers Försäkringskassa och Reparationssamordnare.

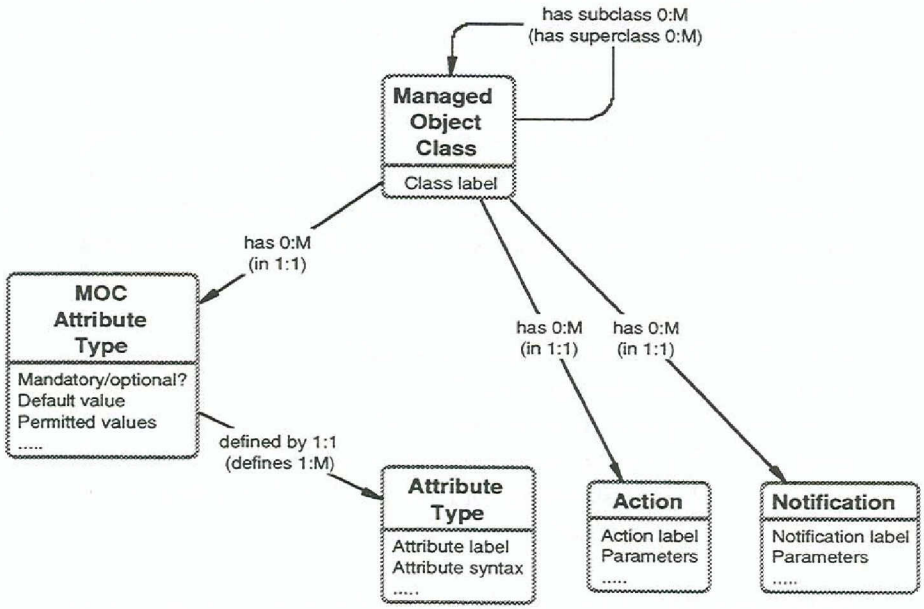
Alternativt kan det tänkas att chefen i rollen av manager får 'intagen på sjukhus' som en notification från ett annat öppet system Sjukhus och som en följd av detta omformulerar uppgiften till en notification av typ 'sjukskrivning'. Osv.

5.2 Notification som begrepp i Management Information Model

En notification är en beskrivning under en managed object class av en möjlig, för omgivningen intressant typ av händelse som kan inträffa hos den modellerade resurstypen. Jmf det tidigare exemplet när chefen meddelar läkaren att reparatör Svensson råkat ut för en personskada (bröt tummen). Antagligen svarar notification 'personskada' under managed object class Reparatör, kompletterad med bla parametrar som beskriver skadan mer i detalj, när det inträffade, ..., som en möjlig "bärare" av uppgiften till omvärlden. På motsvarande sätt kan det finnas anledning att definiera notification 'powerfailure' för klassen 'modem'.

I gränssnittet mellan agenten och omgivande managers definieras endast att en viss typ av uppgift kan komma att levereras avseende object tillhörigt viss managed object class. Hur och när händelsen uppfattas av agenten, osv ligger helt utanför GDMOs möjlighet att påverka eller ha synpunkter på. Däremot kan olika stipulerade krav, förutsättningar m m mycket väl ingå i beskrivningen av en viss typ av notification.

Beskrivningen placeras under en egen rubrik under resurstypens modellavbild, d v s dess managed object class. Se figur 11 (något ändrad i layouten jämfört med tidigare).



Figur 11

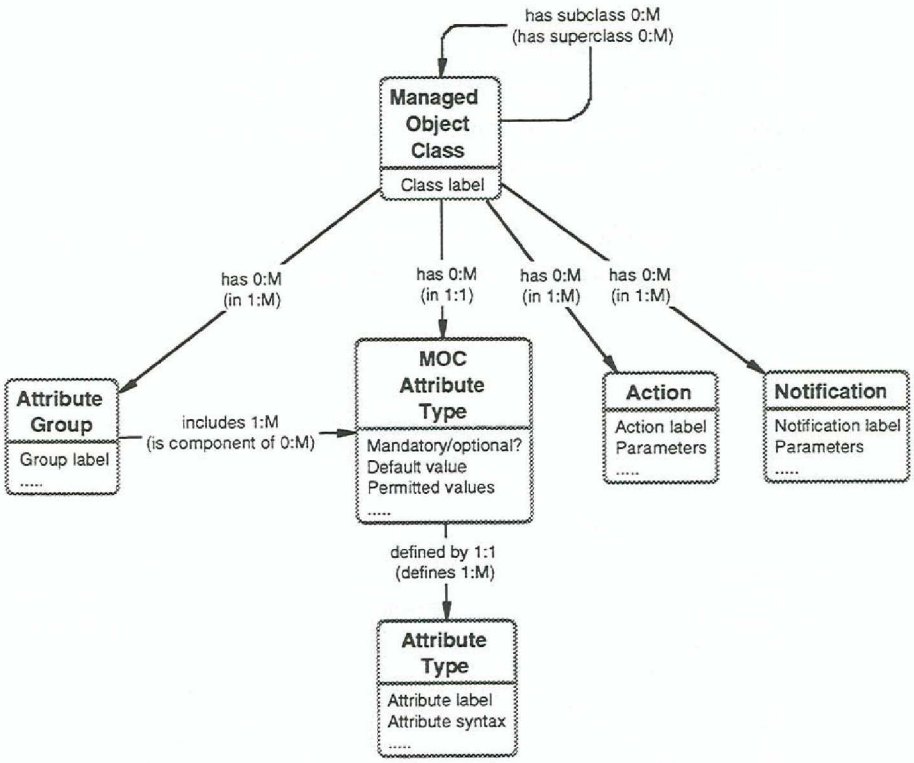
Inte heller notification behöver instansieras per förekomst.

6. Grupperingsbegrepp

6.1 Attribute group

Om man kan bedöma att flera attribute types alltid samtidigt refereras och hanteras lika i vissa situationer, finns möjligheten att samla dem i en attribute group under ett eget namn. På så vis kan man rent tekniskt ersätta en operation av samma typ på vardera av ett antal attributes (tillhöriga olika attribute types) med en enda operation på en grupp attributes. Operationen utförs även i detta fall på varje attribute i gruppen. Konstruktionen har bara relevans som en effektiviseringsåtgärd i samband med formulering av management operations över gränssnittet. Grupperingen formulerar alltså inte någon semantisk valör. Dess naturliga hemvist vore snarare under management operations som ett slags sammansatt operation. Se figur 12.

Som en parentes kan nämnas att attribute groups under vissa förutsättningar även kan användas för att utöka redan definierade attribute groups i eventuella superclasses.



Figur 12

6.2 Package

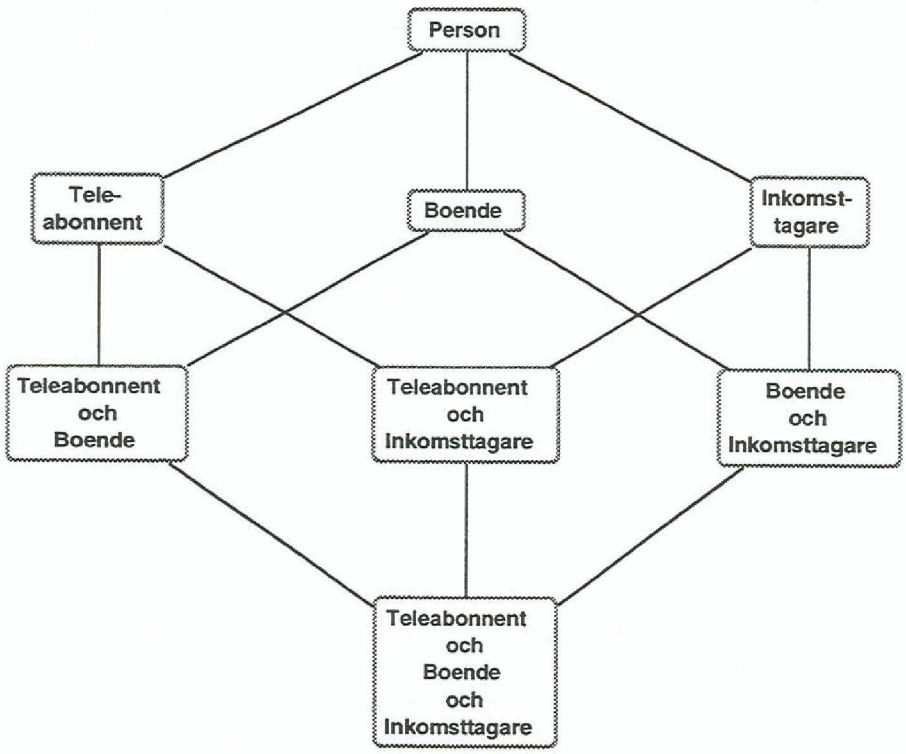
Ett package är en gruppering av ett antal beskrivningselement för en viss Managed Object Class. Ett beskrivningselement kan vara av typen attribute type, attribute group, action, notification och behaviour (se avsnitt 7). Package är ett något ovanligt begrepp i modelleringssammanhang. Syftet är att gruppera beskrivningeelement som av någon anledning är underställda samma existensvillkor, kanske beroende på någon specifik roll, version, konfiguration eller användning av en Managed Object Class.

Ett sätt att göra detta är att skapa en ny klass och etablera ett subklassförhållande. I vissa situationer kan sådana subklasser dock upplevas som artificiella eller bli oöverskådligt många. Ta t ex klassen Person som alltid har namn och personnummer, ibland telefonnummer (om teleabonnemang finns), ibland gatuadress, postnummer och ort (om fast bostadsadress finns), ibland årsinkomst, slutlig och inbetalad skatt (om inkomsttagare).

Vi skulle kunna formulera subklasserna Teleabonnent, Boende och Inkomsttagare. Därutöver har vi ju olika kombinationer av dessa:

- Telefonabonnent och Boende men inte Inkomsttagare
- Telefonabonnent och Inkomsttagare men inte Boende
- Boende och Inkomsttagare men inte Telefonabonnent
- Boende, Inkomsttagare och Telefonabonnent

Det blir många subklasser i olika beroendeförhållanden om alla aspekter ska representeras i en generaliseringsstruktur. Se figur 13.



Figur 13

Kanske vill man helt enkelt bara ha objektklassen Person och istället till viss del kompensera genom att anlägga olika villkor för när attribut ska, respektive inte ska, existera. Exempelvis vet vi att antingen finns både gatuadress, postnummer och ort eller ingen av dem. Samma gäller årsinkomst och skatt. I det generella fallet kan existensvillkor mellan attribute types vara godtyckligt komplexa. Inom GDMO har man valt att formulera villkor genom grupperingar enligt "alla eller ingen" i form av packages (t ex package 'Adress' innehållande attribute types Gatuadress, Postnummer och Ort).

Ett package kan vara ovillkorligt, d v s måste alltid finnas för samtliga objects av klassen eller villkorligt, d v s kan som helhet utelämnas. Bivillkoret är dock att om ett package finns ska samtliga ingående element finnas representerade. Finns Adress ska alltså samtliga tre attribute types (Gatuadress, Postnummer, Ort) finnas med.

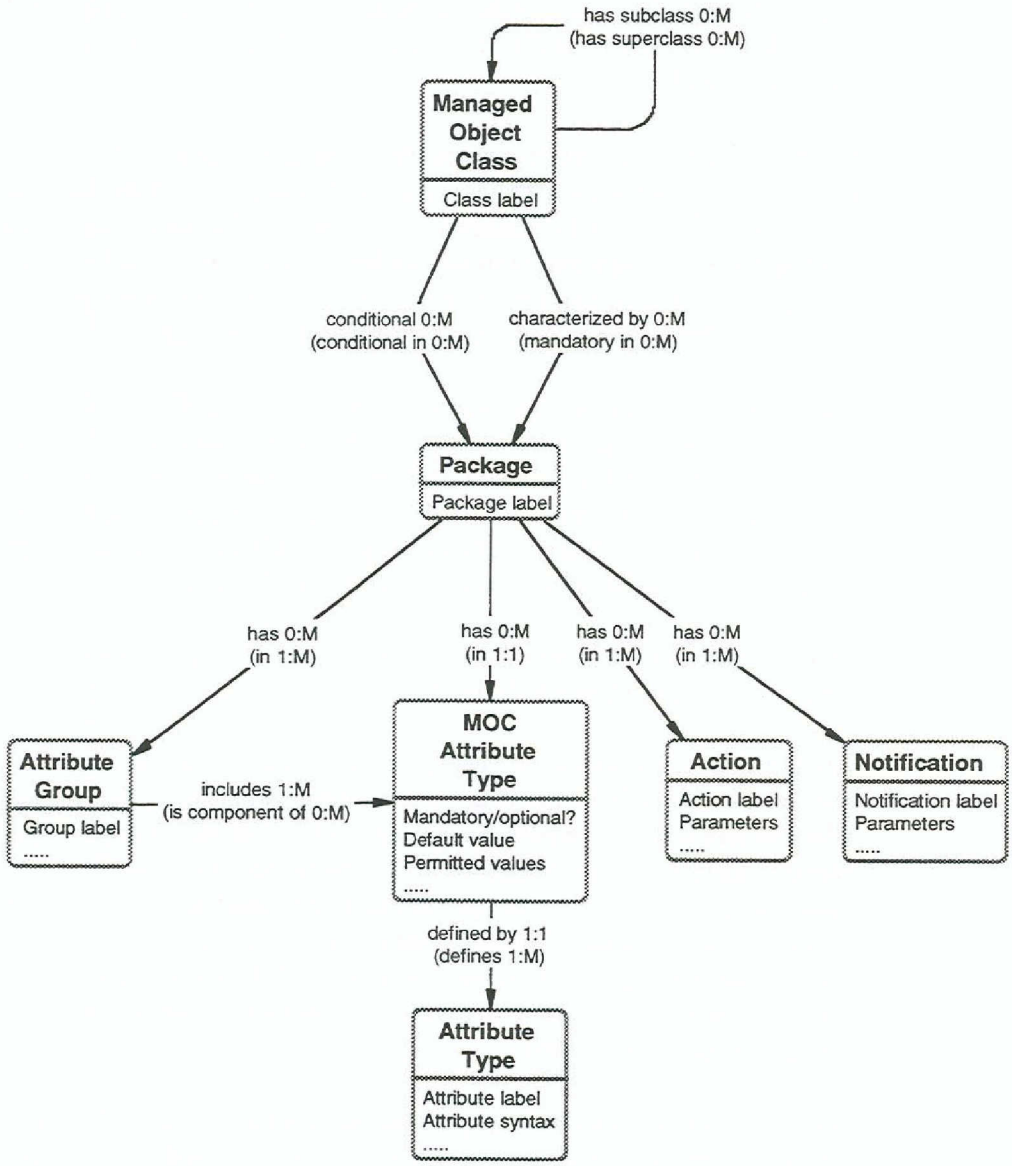
Observera att det är tillåtet att upprepa ett visst beskrivningselement under flera packages. Exempelvis kan det finnas lägen i samband med skatteåterbäring där både Adress, Slutlig och Inbetalad skatt måste finnas. Eftersom redan definierade packages inte kan återanvändas som komponent i annat package måste package 'Återbäringsuppgifter' innehålla fem beskrivningselement nämligen Gatuadress, Postnummer, Ort, Slutlig skatt och Inbetald skatt.

Package är med andra ord något mer än en underindelning eller rubriksättning av de olika beskrivningselementen under en Managed Object Class. Packages beskriver villkor och kan ha delvis överlappande beskrivningselement. Dessutom utvärderas förutsättningarna (villkoren) endast i samband med skapandet av ett visst Managed Object. Under objektets levnad får inga nya omständigheter uppstå som innebär att ett package tillkommer eller utgår.

Också värt att notera är att ett och samma package kan användas som deldefinition för flera Managed Object Classes.

Att definiera och använda packages på ett enhetligt sätt verkar inte vara en alldeles enkel uppgift. Frågan är om inte en åtskillnad mellan de beskrivningselement som beskriver en Managed Object Class och de villkor som styr deras närvaro, hade varit mer klargörande. De flesta begreppsmodeller i datamodelleringssammanhang tillämpar denna teknik med framgång. Det bör också noteras att Package i sig inte instansieras i samband med att ett objekt skapas, bara dess ingående beskrivningselement. (Det finns inga management operations eller notifications som refererar till packageförekomster, endast till enskilda beskrivningselement.)

Genom kompletteringen av begreppet package inordnas tidigare definierade begrepp under detta och managed object classes sätts istället att peka ut villkorliga (conditional) och obligatoriska (characterized by) packages. Se figur 14.


has subclass 0:M
(has superclass 0:M)
Managed Object Class
Class label
conditional 0:M
(conditional in 0:M)
characterized by 0:M
(mandatory in 0:M)
Package
Package label
has 0:M
(in 1:M)
has 0:M
(in 1:1)
has 0:M
(in 1:M)
has 0:M
(in 1:M)
Attribute Group
Group label
.....
includes 1:M
(is component of 0:M)
MOC Attribute Type
Mandatory/optional?
Default value
Permitted values
.....
Action
Action label
Parameters
.....
Notification
Notification label
Parameters
.....
defined by 1:1
(defines 1:M)
Attribute Type
Attribute label
Attribute syntax
.....

Figur 14

7. Behaviour

Utöver de begrepp som diskuterats ovan finns ett allmänt begrepp, **behaviour**, som används för beskrivande text, kommentarfält mm. Det kan användas för att beskriva samtliga övriga begrepp och är genom detta något väsensskilt dessa. Behaviour har inte innebörden av att vara någon typ av definition av objektdynamik i enlighet med ett objektorienterat synsätt. Det är helt enkelt en vanlig textsträng som kan "hakas på" vad som helst som behöver få en kompletterande, icke-formaliserad beskrivning. Behaviour är ett olyckligt namn eftersom det kan ge associationer just till en dynamisk aspekt. Visst kan texten handla om det beskrivnas eventuella beteende, men lika gärna om villkor, synpunkter, karakteristik, minnesanteckningar mm, som inte naturligt kan placeras under något annat befintligt begrepp. Alternativnamn som Text, Description e.dyl kanske skulle undanröja missförstånd?

8. Namngivningsprinciper

Numer är det vedertagligt att skilja på identifiering eller representation av objekt och referens till objekt. En objektidentifierare (OID) ses som en unik symbol enligt någon fördefinierad syntax. OID genereras av "systemet", gäller oförändrad under objektets hela livstid, saknar semantik och är ej intressant för användare. OID är enbart en unik uppbindningspunkt för olika typer av uppgifter om objektet.

Referens till objekt kan ske på många olika sätt. Ta exempelvis en kund, som vi antar kan pekas ut dels genom en kombination av förnamn, efternamn och telefon, dels genom en "artificiell" attribute type (kundnummer), dels genom något objektsamband (fakturanummer för senaste reparation), dels genom en kombination av attribut och samband med andra objektet (efternamn och regnr på ägd bil,), dels genom att succesivt lägga till alltmer avgränsande villkor o s v.

Inom GDMO görs ingen distinktion mellan identifiering och referens. Man tycks närmast se ett behov av att använda referenser vid datautbyte. En unik referens kallas inom GDMO för Distinguished Name (DN). Ett DN till ett visst managed object skapas i det ideala fallet genom ett visst värde på en för klassen definierad attribute type, t ex fakturan med fakturanummer 12345. Inom en verkstad kan detta vara helt tillfyllest.

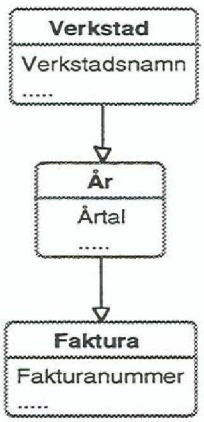
Antag däremot att Bilverkstädernas Förening vill samla in uppgifter om alla verkstäders fakturor inom ett visst år. Både verkstad A och B kan ha faktura 12345. Fakturanummer 12345 är bara unikt om även verkstaden är angiven. Fakturanummer och verkstadsnamn ger tillsammans unik referens (DN). Inom GDMO kallas under dessa förutsättningar fakturanummer för relative distinguished namn (RDN). Fakturaobjektet blir för namngivningen underordnat (**subordinate**) verkstadsobjektet. Verkstadsobjektet är överordnat (**superior**) fakturaobjektet.

Dessutom väljer föreningen att göra denna insamling varje år framöver. Då blir fakturanumret och namnet på verkstaden bara en unik referens om även året anges.

I det generella fallet kan ett godtyckligt djupt namngivningsträd skapas. Ett och samma objekt kan mycket väl nås över flera alternativa träd, när så önskas. Kraven för detta är att det attribute type som används på en viss nivå unikt refererar till ett visst objekt på denna nivå och att ett unikt superior object är angivet.

Tillåtna namngivningskonstruktioner definieras i ett naming schema som ett antal **name bindings**, var och en mellan en superior managed object class och en subordinate managed object class. Figur 15 a visar ett naming schema som omfattar två name bindings från exemplet ovan. Figur 15 b visar ett lika möjligt naming schema med hänsyn till den givna situationen.

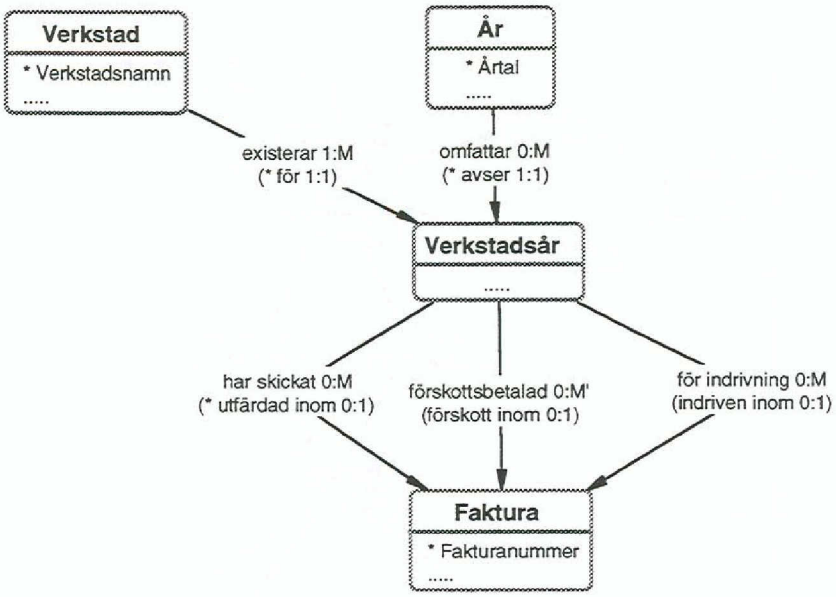
Figur 15 a



Figur 15 b

Det bör poängteras att gruppering av object classes i ett naming schema enbart är till av namngivningsskäl. Schemat ska inte förväxlas med "is-part-of" samband. Schemat är ju heller inte nödvändigtvis hierarkiskt. Eftersom sambandstyper (Relationship Types) inte fanns definierade inom Management Information Model när namngivningsprinciperna lades fast, finns ingen möjlighet att utnyttja dem och deras semantik för namngivning. En name binding är med andra ord endast ett slags diffust superior/subordinate-förhållande för namngivningsändamål.

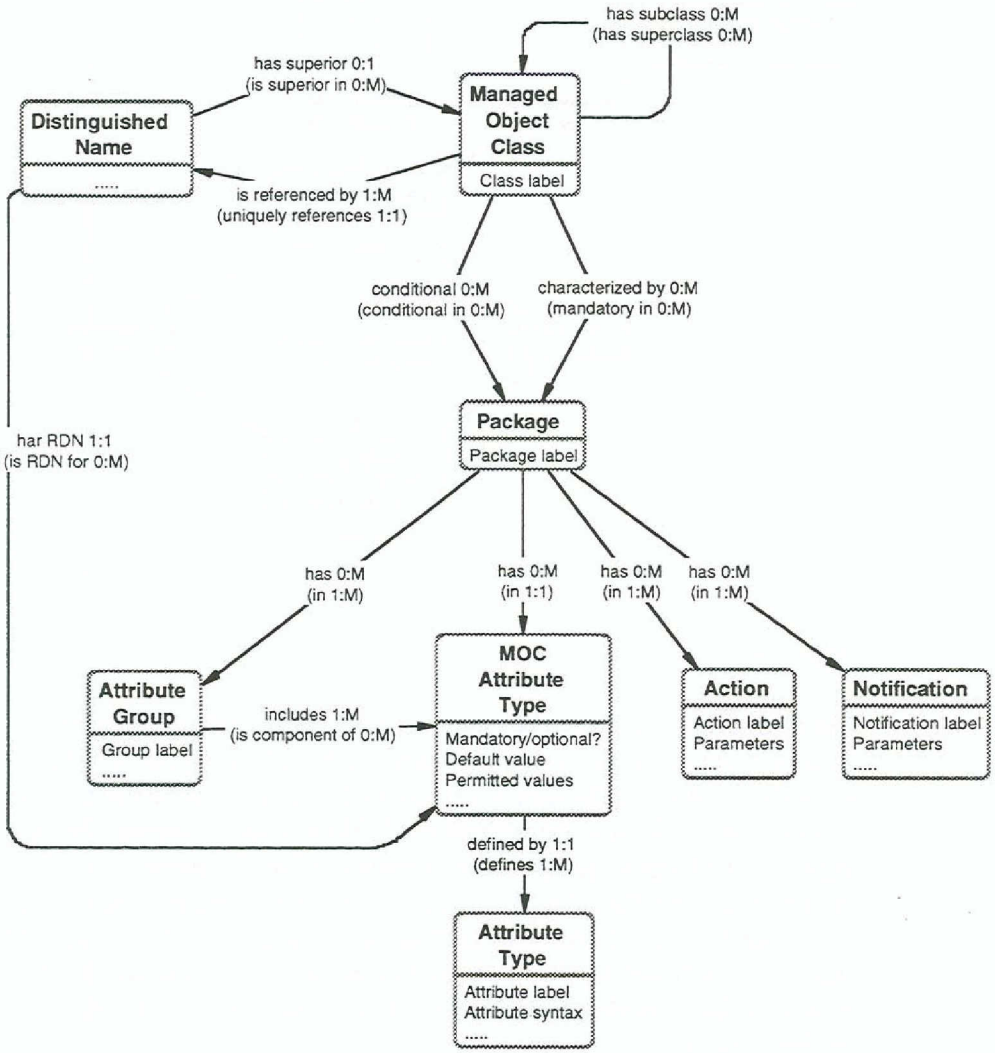
I en rikare modell skulle det kunna uttrycka olika typer av samband mellan Verkstad och Faktura varav en befinns lämplig för namngivning. I Telmod-notation skulle det t ex kunna se ut enligt figur 16.



Figur 16

Överhuvudtaget måste man vara klar över att så snart en object class befinner sig som subordinate har den ingen fristående existens, bara som underordnad sin superior. Av den anledningen är det vid modellering viktigt att göra rollerna klara genom en distinkt namngivning och lika distinkt relatering. Verkstad i figur 15 a borde egentligen ges det semantiskt mer korrekta namnet Verkstadsår så som framgår i figur 16. År i figur 15 b borde också ges samma namn (Verkstadsår). Figur 16 redovisar klart dessa nyansskillnader. En modell enligt figur 16 ställer heller inga krav på hierarkisk inordning av Verkstad och År. Det enda som sägs är att båda deras respektive referenser behövs för unik referens till ett Verkstadsår.

Standardens syn på namngivning redovisas i figur 17.


has subclass 0:M
(has superclass 0:M)
has superior 0:1
(is superior in 0:M)
Managed Object Class
Class label
Distinguished Name
.....
is referenced by 1:M
(uniquely references 1:1)
conditional 0:M
(conditional in 0:M)
characterized by 0:M
(mandatory in 0:M)
Package
Package label
har RDN 1:1
(is RDN for 0:M)
has 0:M
(in 1:M)
has 0:M
(in 1:1)
has 0:M
(in 1:M)
has 0:M
(in 1:M)
Attribute Group
Group label
.....
includes 1:M
(is component of 0:M)
MOC Attribute Type
Mandatory/optional?
Default value
Permitted values
.....
Action
Action label
Parameters
.....
Notification
Notification label
Parameters
.....
defined by 1:1
(defines 1:M)
Attribute Type
Attribute label
Attribute syntax
.....

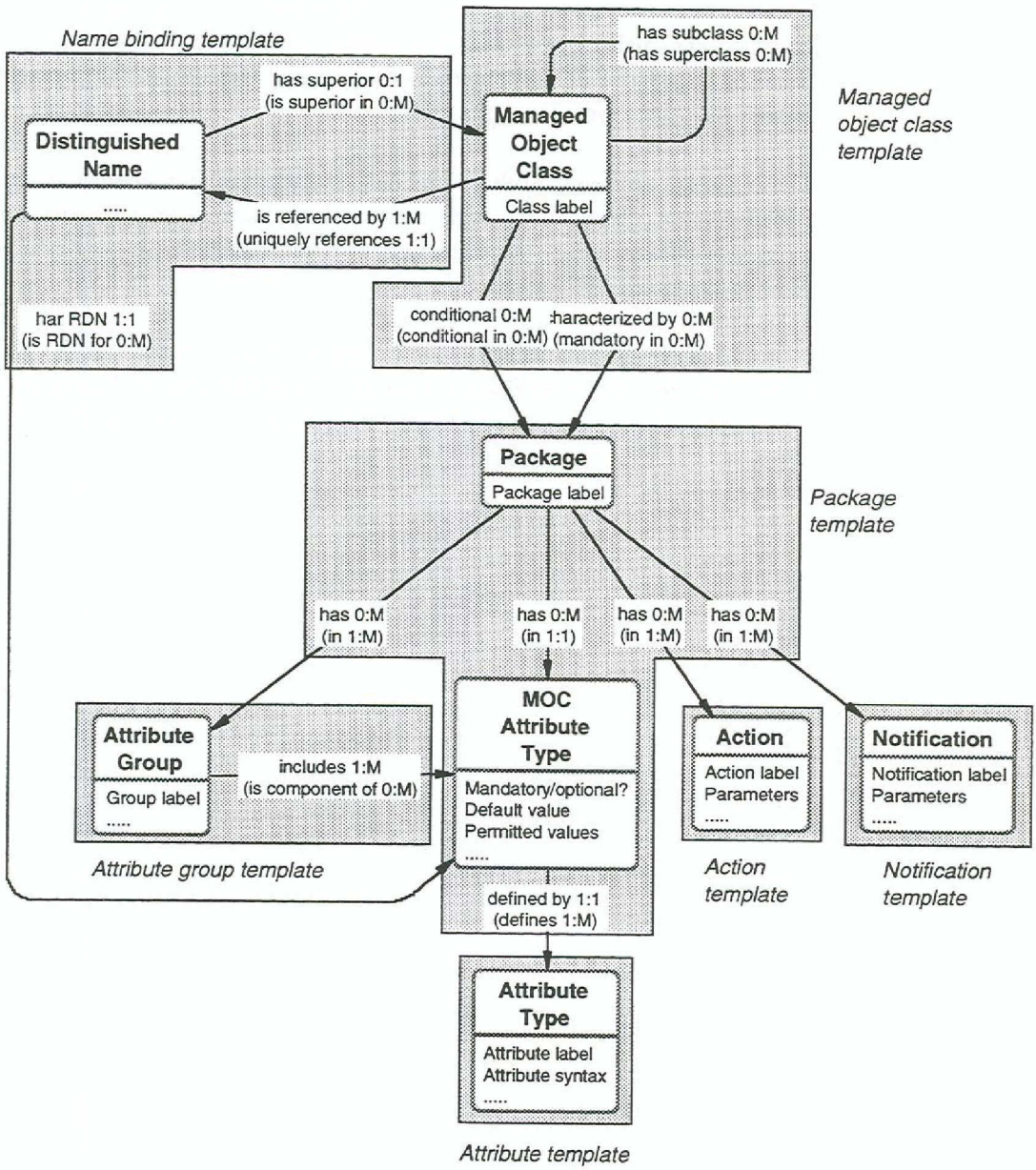
Figur 17

9. Templates

I dokumentet ISO/IEC 10165-4 ges anvisningar för hur Managed Objects lämpligen definieras i enlighet med begreppen i Management Information Model. Man har valt att gruppera de olika typerna av uppgifter i ett antal templates eller mallar. En viss template reglerar med andra ord vilka komponenter som ska anges för att beskriva en viss delmängd av Management Information Model och vilken syntax som ska gälla.

Det är givetvis inget som hindrar att ett mer användarvänligt gränssnitt används i ett datorstöd, i blankettutformning etc. så länge detta format entydigt kan översättas till standardens template-format. Även andra grupperingar av uppgifterna kan mycket väl tänkas.

I figur 18 är de uppgifter som respektive template omfattar gråtonade. Det bör noteras att det finns ytterligare ett par templates (Behaviour, Parameter) i standarden. De representerar dock uppgifter som inte redovisas i grafen.



Figur 18

Figur 19 visar, något förenklat, vad mallen för "managed object class-uppgifter" innehåller, uttryckt i enlighet med gällande syntax. I den mån en mall refererar till komponenter som beskrivs i andra mallar, finns i allmänhet flexibiliteten att använda denna andra mall som submall. Det betyder att man direkt kan infoga dessa uppgifter i ett överordnat sammanhang, istället för separat. Efter

CHARACTERIZED BY kan t ex aktuell obligatorisk package till fullo definieras, om så önskas.

```
<class-label>     MANAGED OBJECT CLASS

[DERIVED FROM
                  <class-label> [,<class-label>]* ;]
[CHARACTERIZED BY
                  <package-label> [,<package-label>]* ;]
[CONDITIONAL PACKAGES
                  <package-label> PRESENT IF condition-definition
                  [,package-label> PRESENT IF condition-definition]* ;]


 REGISTERED AS object identifier;
```

Figur 19

Givetvis kan uppgifterna i princip sammanställas i andra grupperingar. Exempelvis skulle sub/superclass-förhållanden kunna etableras genom en separat template istället för inom managed object class template (DERIVED FROM). Oavsett vilken indelning som väljs är det avgörande kriteriet att tillse att data fullt ut hanteras i enlighet med angivna villkor i Management Information Model (t ex vad som kan lämnas oifyllt, vad som kan anges med fler än ett värde, ...). Att man i standarden valt att exakt definiera templates istället för att nöja sig med definition av Management Information Model beror antagligen på att man genom enhetlig uppställning hoppas underlätta förståelse och utbyte av uppgifter, framförallt så länge dokumentationen sker manuellt. Datorstöd ger ju en helt annan flexibilitet både vid inmatning och presentation.

10. Några avslutande kommentarer

Beskrivningen av Management Information Model i denna rapport är baserad på dokumenten ISO/IEC 10165-1 och 10165-4. För närvarande diskuteras kompletteringar av modellen, framförallt i form av relationship types. Ett Committe Draft finns framme under beteckningen 10165-7 och med titeln "General Relationship Model". Med tanke på det synnerligen kompakta och från etablerade modelleringsprinciper ganska väsensskilda innehållet kommer förmodligen dokumentet att diskuteras flitigt och genomgå ett antal revideringar innan den accepteras som International Standard (IS).

I standardtexterna och i artiklar om GDMO framhävs ofta modellens objektorienterade prägel. Enligt min syn representerar standarderna snarare en begreppsapparat (Management Information Model) med vars hjälp verksamhetsmodeller kan byggas upp för att reglera framförallt semantik och i viss mån syntax för ett datautbyte baserat på ömsesidigt överenskommen förståelse. Som

vitala ingredienser finns ett antal verksamhetsbegrepp, deras betydelse och deras samband. Vissa begrepp kallas managed objects, vissa attributes, ytterligare andra actions o s v. Visst tillåter Information Model formulering av generaliseringssamband (sub/superclass-samband) men denna uttryckskraft har funnits inom datamodellering under lång tid.

I standarden finns ingenting av vad som brukar framhållas som unikt inom objektorientering, nämligen inkapsling (all samverkan med ett objekt går via definierade beteenden eller methods som helt kontrollerar ett objekts karakteristik och vilja att reagera på omgivningens "signaler"), objektsamverkan genom meddelandeutväxling samt objekts påtagliga så kallade dynamiska egenskaper. Detta är helt i sin ordning. Målsättningen är ju här bara att entydigt utbyta uppgifter av olika slag, inte att fullt ut definiera ett dynamiskt system.

Den objektorienterade anstrykningen finns i så fall i samverkan mellan de två objekten manager och agent. De utväxlar meddelanden, reagerar på dessa osv. Men det är nog inte det perspektivet som åsyftats bakom referensen till objektorientering. Skulle en begäran om en action riktas direkt till ett visst managed object (eller snarare till den bakomliggande resursen) och skulle detta object svara mot inkapslingsprincipen samt reagera på begäran, vore ju situationen ganska typiskt objektorienterad. Å andra sidan är varken förutsättningar eller målsättning i linje med detta.

Ta följande paralleller:

> *Förmodligen upplever inte en leverantör (manager), som skickar en faktura (management operation) elektroniskt till en kund (agent) enligt EDIFACT-standard (verksamhetsmodell + CMIP-syntax), att han befinner sig mitt i en objektorienterad process, även om fakturan kompletterats med en önskan om snabb betalning (Action).*
>
> *Lika lite är ett utbyte av Case-data baserat på en metamodell (verksamhetsmodell), som är utformad i enlighet med CDIFs metametamodell (Management Information Model), speciellt objektorienterat.*

Standarderna tycks ha fått ordentligt genomslag. Många managed object classes har definierats (och är under definition) i linje med standardens riktlinjer. Behovet är ju trängande i detta synnerligen internationella fält med mångskiftande aktörer. Frågan är dock om standardens kvalitet i uttryckskraft, precision, enhetlighet, m m kan tillfredsställa behoven?

Det pågår ett antal standardiseringsaktiviteter inom olika tillämpningsområden (se avsnitt 2.4, del A) med samma eller åtminstone mycket snarlika syften. Antagligen skulle resultaten från dessa ansträngningar vinna i kvalitet, för att inte tala om handläggningstid, om man etablerade aktiva kunskapsutbyten ("liasons"), något som i standardiseringssammanhang mer verkar vara en fras till protokollen än en fungerande realitet.

KORT OM TRIAD

Triad är namnet på ett treårigt samarbetsprojekt kring informations-administration och dataadministration, IA/DA, som Telia, Posten, Ericsson, Statskontoret och SISU bedriver. Syftet är att utveckla parternas synsätt, metoder och hjälpmedel inom detta område. Arbetet inom Triad är uppdelat i delprojekt som är sammanförda i tre block.

Beställarblocket vänder sig dels till dem som är verksamhetsansvariga och måste ta ställning till IA/DA-satsningar, dels till dem som har ansvaret för IA/DA inom en organisation. Delprojekten inom detta block arbetar med att formulera verksamhetens krav på IA/DA samt studerar och beskriver roller, organisation och arbetsformer för IA/DA-arbete.

Utförarblocket vänder sig till dem som arbetar med IA/DA. Delprojekten arbetar med modellering, data- och resurskataloger samt uttagssystem.

Kunskapsförmedling är det block som ser till att resultaten kommer Triad-parterna till godo. Detta sker bland annat genom kurser, seminarier samt genom att rapporter, som denna, ges ut.