TRIAD Arbetsrapport K 15

Krav på IA
Nästa Generation Modellering
Avancerad utbildning för handledare
Katalogprinciper
Verktyg
Informationspridning

AD/Cycle I Information Model

Härledningsspecifikationer i begreppsmodellen

Rapport K nr 1: IRDS
Rapport K nr 2: IRDS Modeller och modellnivåer
Rapport K nr 3: Kopping begreppsmodell - relationsmodell
Rapport K nr 4: IBM:s Repository Manager- en Introduktion
Rapport K nr 5: IBM:s Repository Manager: Datamodelleringsbegreppen
Rapport K nr 6: IBM:s Repository Manager: Begreppsmodellering i Information Model
Rapport K nr 7: IBM Repository Manager: Attribut- och värdemodellering i Enterprise Submodel
Rapport K nr 8: Navigering i Repository
Rapport K nr 9: TRIAD Newsletter – IRDS inom ISO. Dagsläget
Rapport K nr 10: TRIAD Newsletter –ISO/IRDS. Händelseutvecklingen 91/92
Rapport K nr 11: Samverkan mellan resurskataloger – visioner eller behov
Rapport K nr 12: AD/Cycle I Information Model – Processer och och informationsflöden mellan processer
Rapport K nr 13: AD/Cycle I Information Model – Info Flows inom Processmodellen
Rapport K nr 14: AD/Cycle I Information Model – Relationsdatabasmodellering
Rapport K nr 15: AD/Cycle I Information Model – Härledningsspecifikationer i begreppsmodellen

Stig Berild
SISU & Sveriges Tekniska Attachéer

Rapporten är skriven i och för TRIAD delprojekt Katalogprinciper.

TRIAD-projektet drivs gemensamt av Televerket, Posten, Statskontoret, Ericsson Data Services och SISU, Svenska Institutet för Systemutveckling.
För vidare information kontakta SISU Informationscentrum, tel 08-752 16 00, fax 08-752 68 00, Box 1250, 164 28 Kista.

AD/Cycle Information Model:

Härledningsspecifikationer i begreppsmodellen

Innehåll

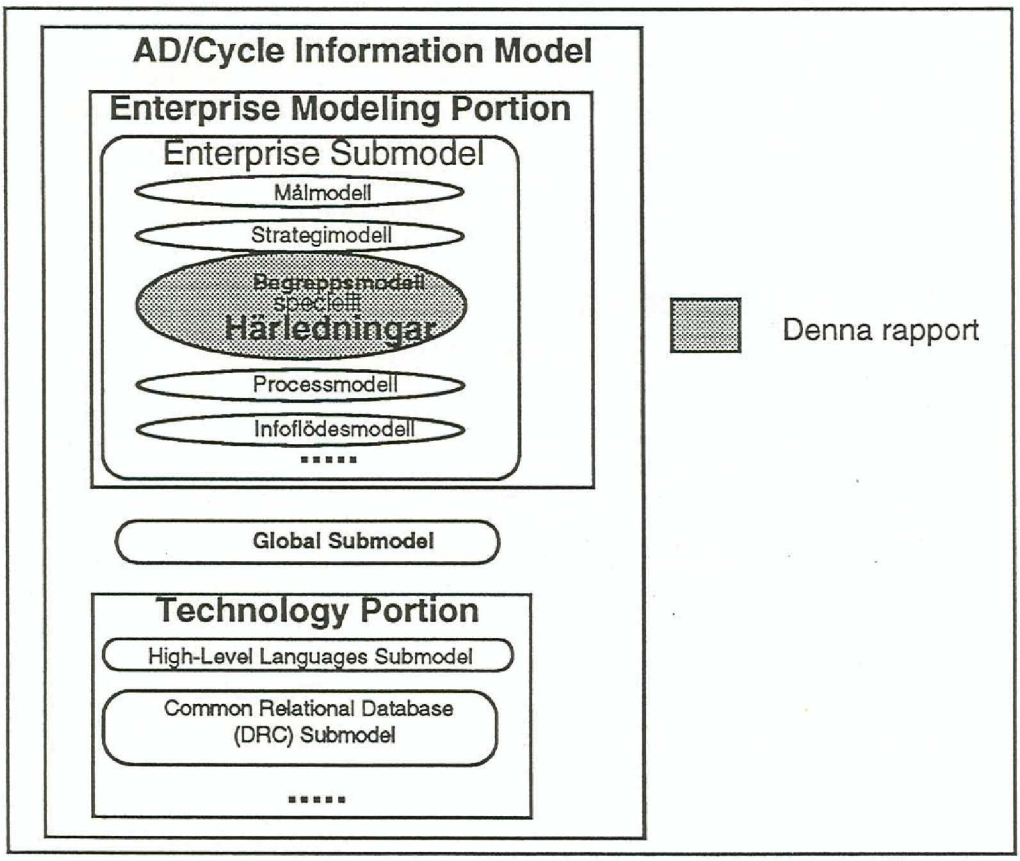
1. Inledning

1.1 Rapportens omfattning

IBMs AD/Cycle Information Model (IM) består av ett antal submodeller, grupperade enligt figur 0. Figuren visar endast några av submodellerna. Det finns fler. Nya tillförs med jämna mellanrum.

Denna rapport beskriver hur härledningar specifieras i den submodell inom "Enterprise Modeling Portion", som går under benämningen "begreppsmodellen". Den grundläggande uppbyggnaden av begreppsmodellen har beskrivits i TRIAD-rapporterna K6 och K7.

Innehållet i rapporten svarar mot IM version 1, release 2, modification 2.



FIGUR 0

1.2 Använd notation

Denna rapport använder samma notation som TRIAD K12. Komponenterna i begreppsmodellen skrivs i fet stil när de först introduceras. Entity types börjar med stor bokstav medan övriga bokstäver är gemena. Relationship types och attribute types skrivs alltid med små bokstäver och omgärdas med citationstecken. Notationen skiljer inte längre på typ och förekomst. Vad som gäller framgår förhoppningsvis av sammanhanget.

Med verksamhetsmodell menar vi generellt förekomstnivå av Enterprise Submodel. En verksamhetsmodell kan alltså bestå av förekomster av processmodellen, infoflödesmodellen, begreppsmodellen eller en kombination av dessa. Exempel på verksamhetsmodellnivå anges kursiverat i texten. Ofta börjar de med stor bokstav.

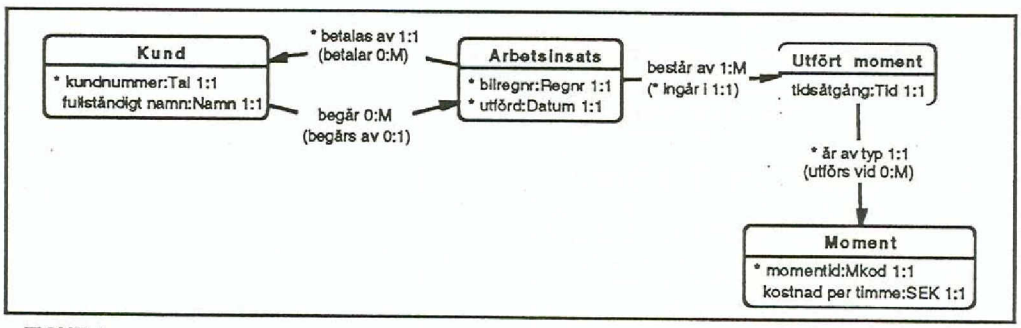
Graferna är som tidigare framställda med Business Modeler (BM).

1.3 Rapportens disposition

Vi börjar med att introducera några centrala begrepp i avsnitt 2. Det finns ett antal olika typer av möjliga härledningsspecifikationer. De delar samtliga vissa gemensamma drag. Dessa redovisas i avsnitt 3. Avsnitten 4 till 7 beskriver och exemplifierar var och en av de olika härledningstyperna, en per avsnitt. Den slutliga begreppsmodellen (de delar som är av intresse i härledningssammanhang) visas i avsnitt 8 tillsammans med en sammanfattande kommentar.

1.4 Exempelunderlag

Vi använder samma modell som i rapporterna TRIAD K13 och K14, något ändrad och utökad.



FIGUR 1

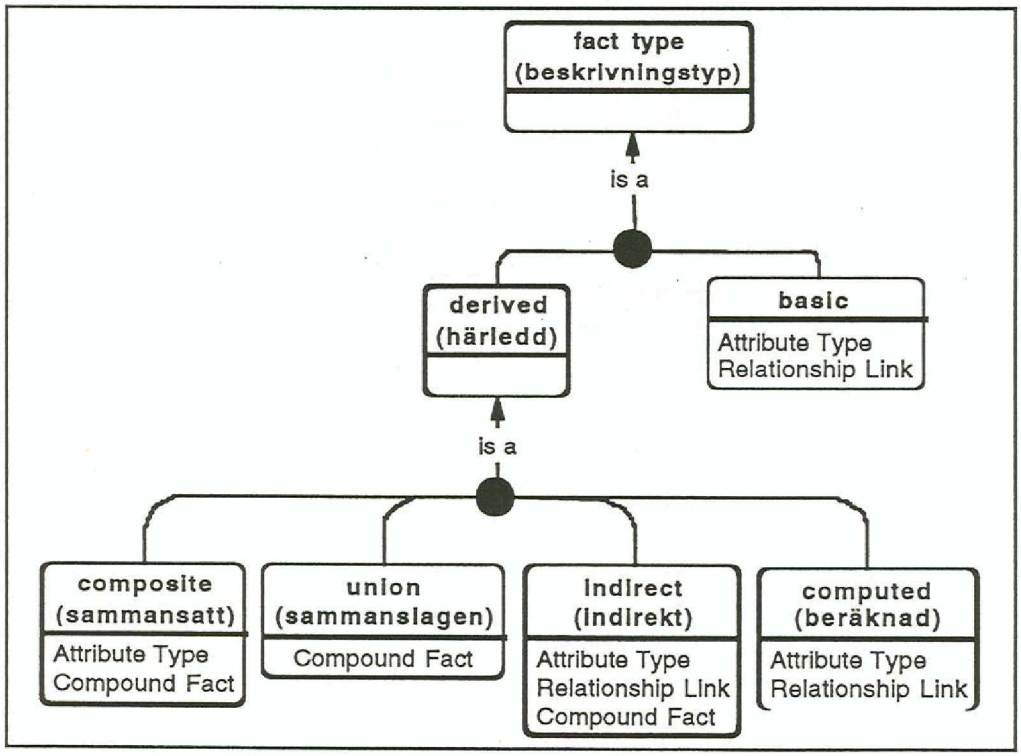
2. Några begrepp

De engelska begreppen redovisas tillsammans med motsvarande svenska. Förhoppningsvis bidrar detta till minskad risk för missförstånd. Den som vid tillfälle läser den engelska manualen kan också lättare känna igen sig bland de engelska begreppen.

Entity Types beskrivs normalt genom en uppsättning Attribute Types och Relationship Links. Repetera gärna TRIAD rapport K6. Samtliga visade Attribute Types och Relationship Links i figur 1 är, i IMs terminologi, **basic**. Innebörden framgår av respektive namn och förklarande text.

Av primärt intresse för denna rapport är **fact types (beskrivningstyper)** som definieras med referens till andra, redan definierade fact types. Dessa kallas **derived** eller **härledda**. Derived fact types kan i sin tur delas in i fyra undertyper **composite (sammansatt)**, **indirect (indirekt)**, **union (sammanslagna)** och **computed (beräknade)**.

Det visar sig att både Attribute Types och Relationship Types, förutom att vara basic, också kan uppträda som derived i olika former. Som derived återfinns också en ny entity type **Compound Fact**. Figur 2 visar vad som kan vara vad, med hjälp av en generaliseringsmodell. (Obs, syftet är bara att förklara hur begreppen hänger ihop. Modellen är inte en del av begreppsmodellen.)



FIGUR 2

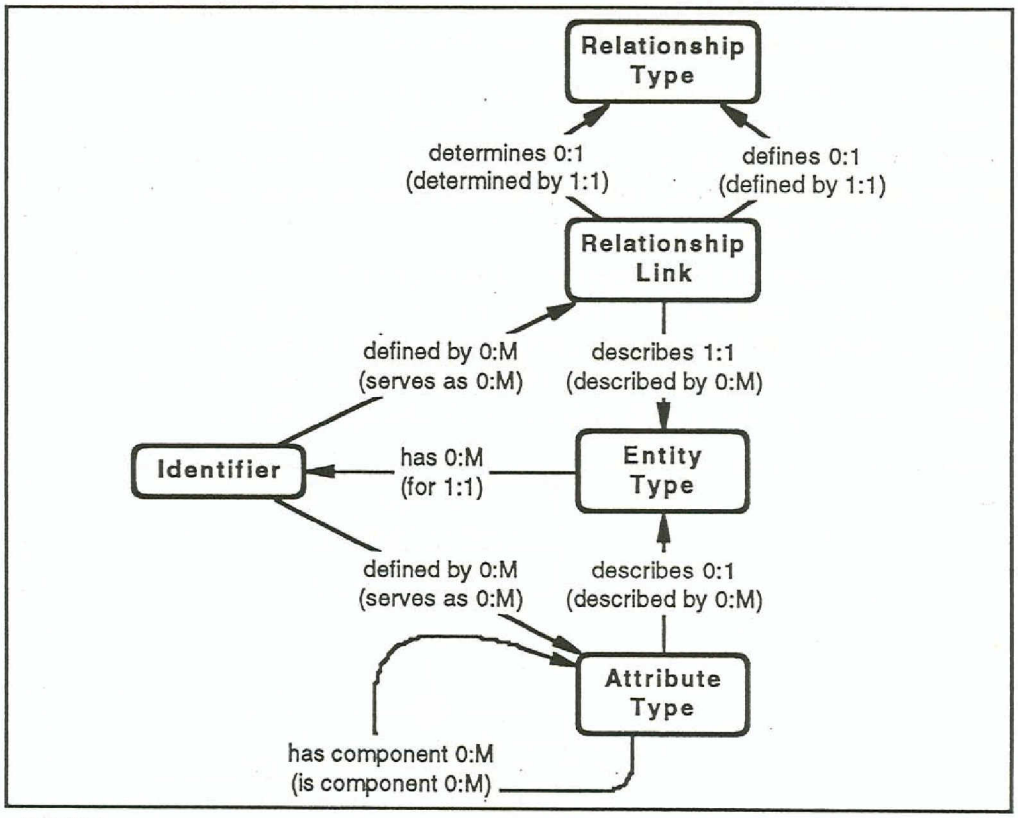
Både Attribute Type, Relationship Link och Compound Fact har en attribute type benämnd "derivation type". Den anger den roll en viss förekomst företräder i ett givet sammanhang.

Inom Attribute Type är möjliga värden: **BA**sic, **CO**mposite, **CU** (computed), **IN**direct.

Inom Relationship Type är möjliga värden: **BA**sic, **CU** (computed), **IN**direct.

Inom Compound Fact är möjliga värden: **CO**mposite, **UN**ion, **IN**direct.

Innan vi går vidare, återknyter vi bekantskapen med de delar av begreppsmodellen från TRIAD K6 och K7, som är av intresse för denna rapport. Se figur 3.



FIGUR 3

3. Generell modell av derived fact types

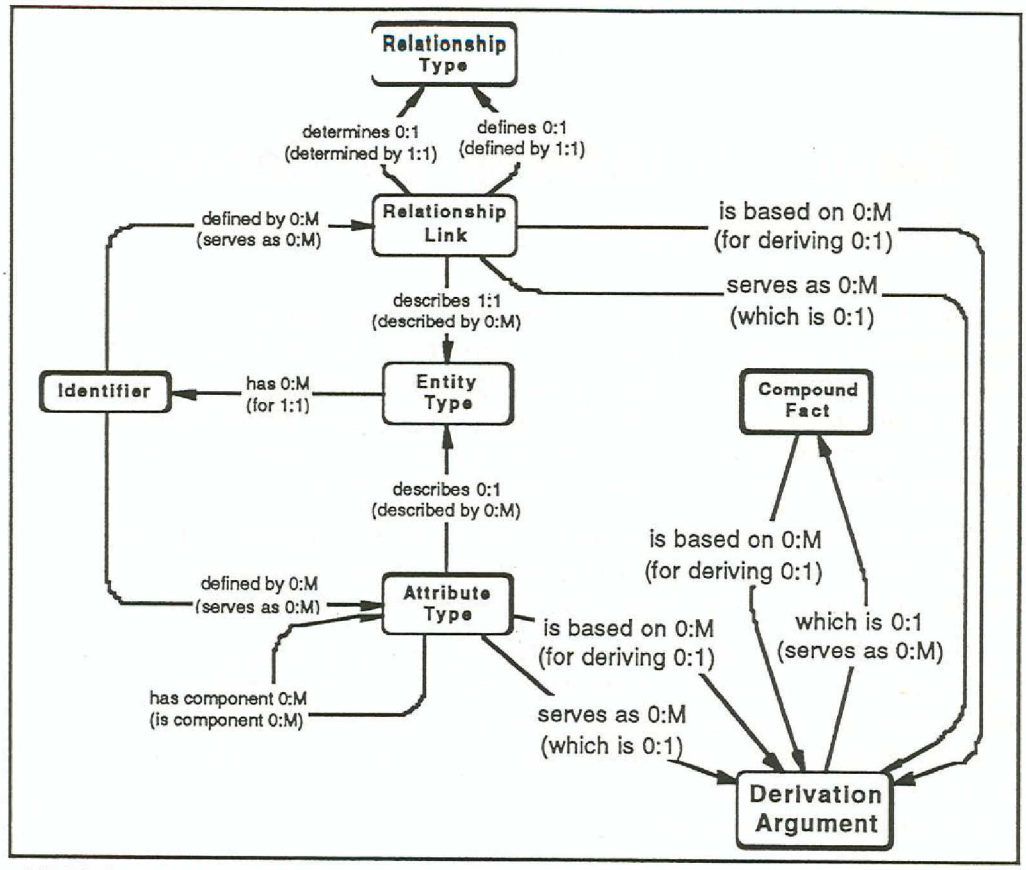
Som synes av figur 2 är det Attribute Type, Relationship Link och en ny företeelse (beskriven i avsnitt 4.2) benämnd Compound Fact, som var och en kan förekomma i olika varianter av derived fact types. Gemensamt för varje derived fact type är att den härleds ur andra fact types. Dessa senare kallas **argument** fact types. Eftersom uppbyggnaden av varje variant definieras på i princip samma sätt, väljer vi att redovisa den generella delen av modellen först.

En derived fact type kan vara definierad över ett antal argument. Varje argument modelleras under entity type **Derivation Argument.** Dessa refereras i alla tre fallen med hjälp av relationship type "is based on/for deriving". Se figur 4.

Ett argument är alltid endera av Attribute Type, Relationship Link eller Compound Fact. Ett visst Derivation Argument refererar mao antingen till en Attribute Type, Relationship Link eller Compound Fact. Denna referens realiseras genom relationship type "serves as/which is". Av någon anledning har man valt att ändra referensriktning mot Compound Fact till "which is/serves as", men innebörden är densamma.

Ett viktigt villkor att hålla i minnet är att en derived fact type och samtliga dess Derivation Arguments måste beskriva en och samma Entity Type.

Därmed är det mesta av vad som modellmässigt behöver tillföras figur 3, för att klara av härledningar, avklarat.



FIGUR 4

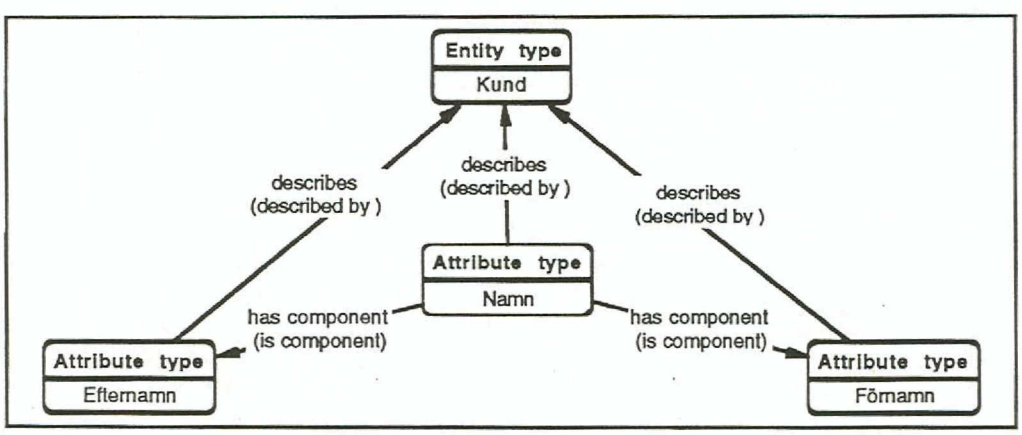
Kvar att beskriva är villkor och konsekvenser för varje variant.

4. Composite fact types

Ordet "composite" står för något som är sammansatt av flera komponenter (arguments).

4.1 Composite Attribute type

Hur ett composite eller sammansatt Attribute Type definieras finns beskrivet redan i TRIAD K6. En kort repetition kan vara på plats. Komponenter för ett composite Attribute Type är andra basic eller composite Attribute Types. De sammanbinds över en 'Ordered' relationship type "has component/is component". 'Ordered' innebär att varje komponent har en viss given position inom den sammansatta konstruktionen. (Detta visas inte i modellen.) Vidare gäller att varje komponent, oavsett nivå i komponenthierarkin, beskriver samma Entity Type. Antag att Attribute Type *fullständigt namn* i själva verket är sammansatt av två basic Attribute Types *förnamn* och *efternamn*. Förekomstmodellen blir enligt figur 5.



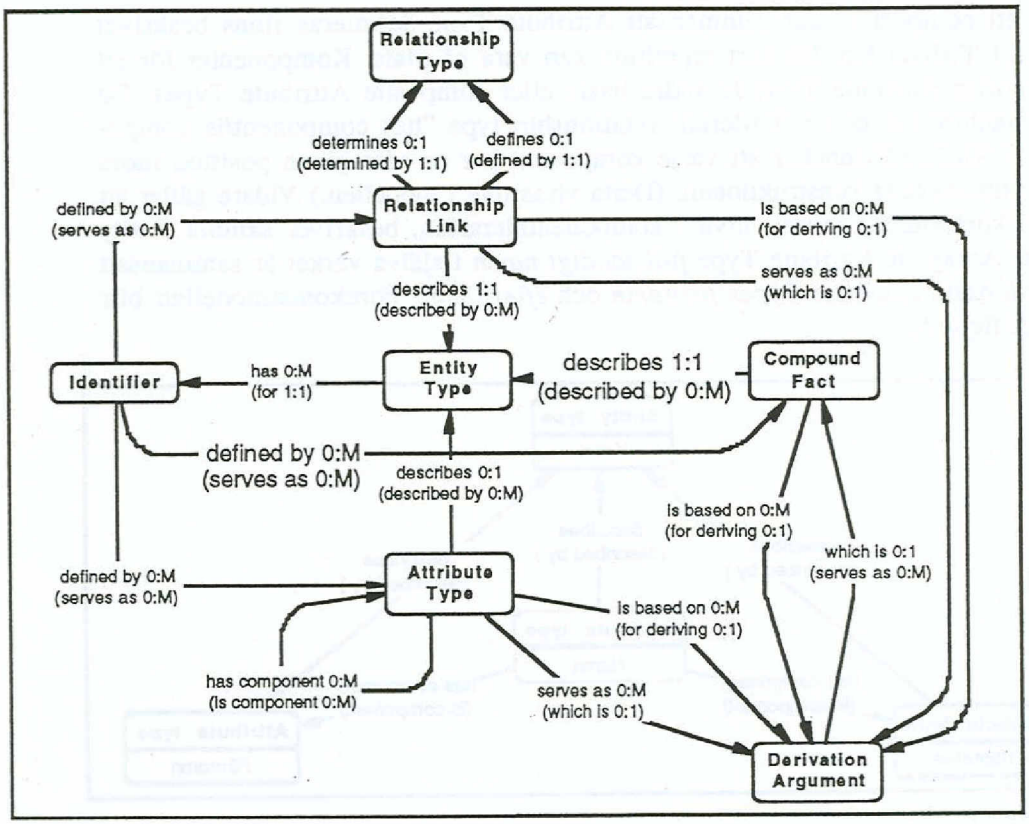
FIGUR 5

4.2 Compound Fact, speciellt composite Compound Fact

Ett composite **Compound Fact** (borde i konsekvensens namn ha hetat Compound Fact Type) är att se som ett komplement till composite Attribute Types. Medan de senare endast kan byggas upp med hjälp av andra Attribute Types, tillåter Compound Fact en blandning av Attribute Types och Relationship Links. Ett composite Attribute Type är fortfarande ett Attribute Type med alla de egenskaper ett sådant kan beskrivas med, exv ett väldefinierat värdeområde i form av referens till en (sammansatt) Info Type.

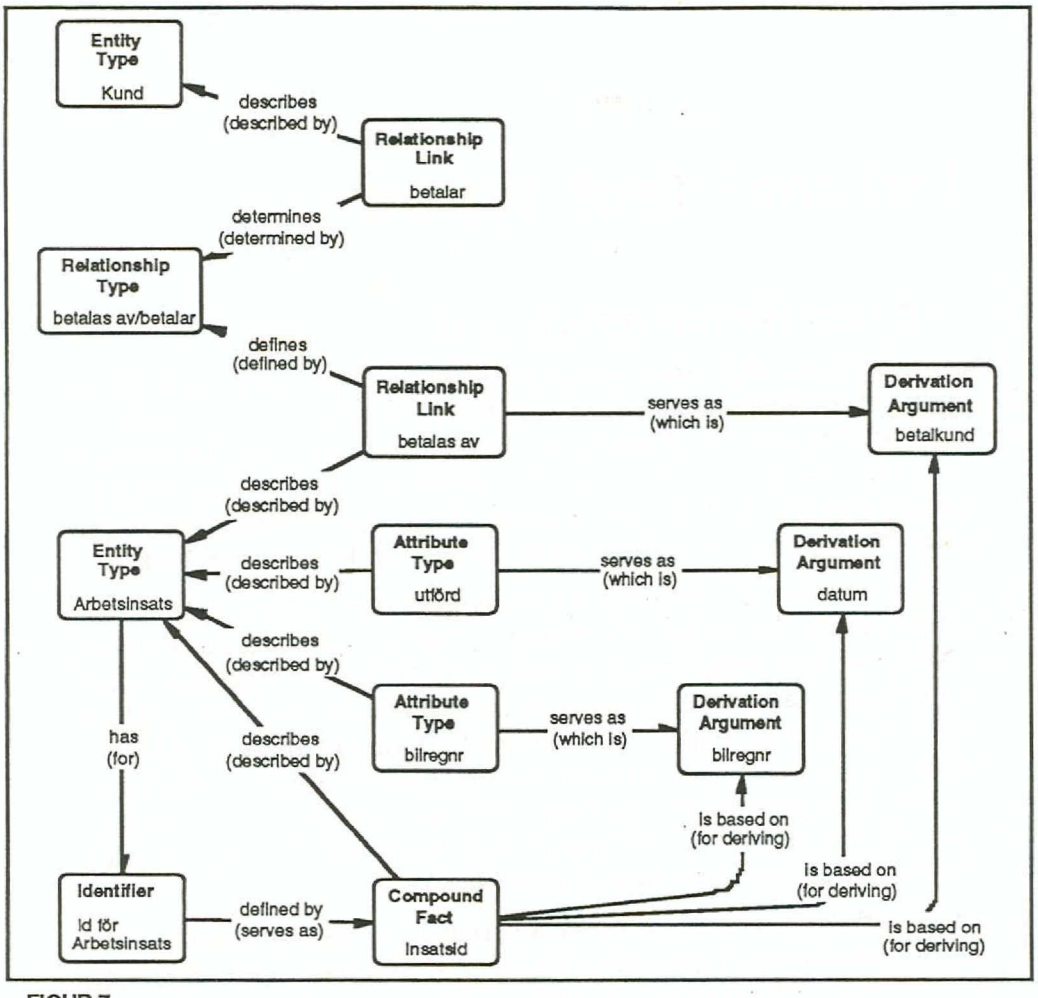
Konsekvensen av en blandning blir ett odefinierbart värde. Man kan fråga sig var behovet av ett sådant kan finnas. Argumentet är att det kan finnas begrepp i en verksamhet som återspeglar en gruppering av mer basala fakta. Dessa begrepp måste kunna fångas upp och beskrivas. Genom att begreppet kan hanteras som en enhet kan dess specifika egenskaper beskrivas. Dit hör existensvillkor, antalsuppgifter mm.

Som synes av figur 6 beskriver ett Compound Fact (genom "describes/described by") en viss Entity Type. Där framgår också att det kan vara komponent i en Identifier på samma sätt som Attribute Types och Relationship Links. Denna referens till Compound Fact ersätter i själva verket motsvarande referenser till dess komponenter.



FIGUR 6

Vi anknyter till det just sagda genom att i exemplet definiera identifieraren till *Arbetsinsats* som en Compound Fact. Identifieraren består av två Attribute Types (*bilregnr* och *utförd*) och en Relationship Link (*betalas av*).



FIGUR 7

5. Indirect fact types

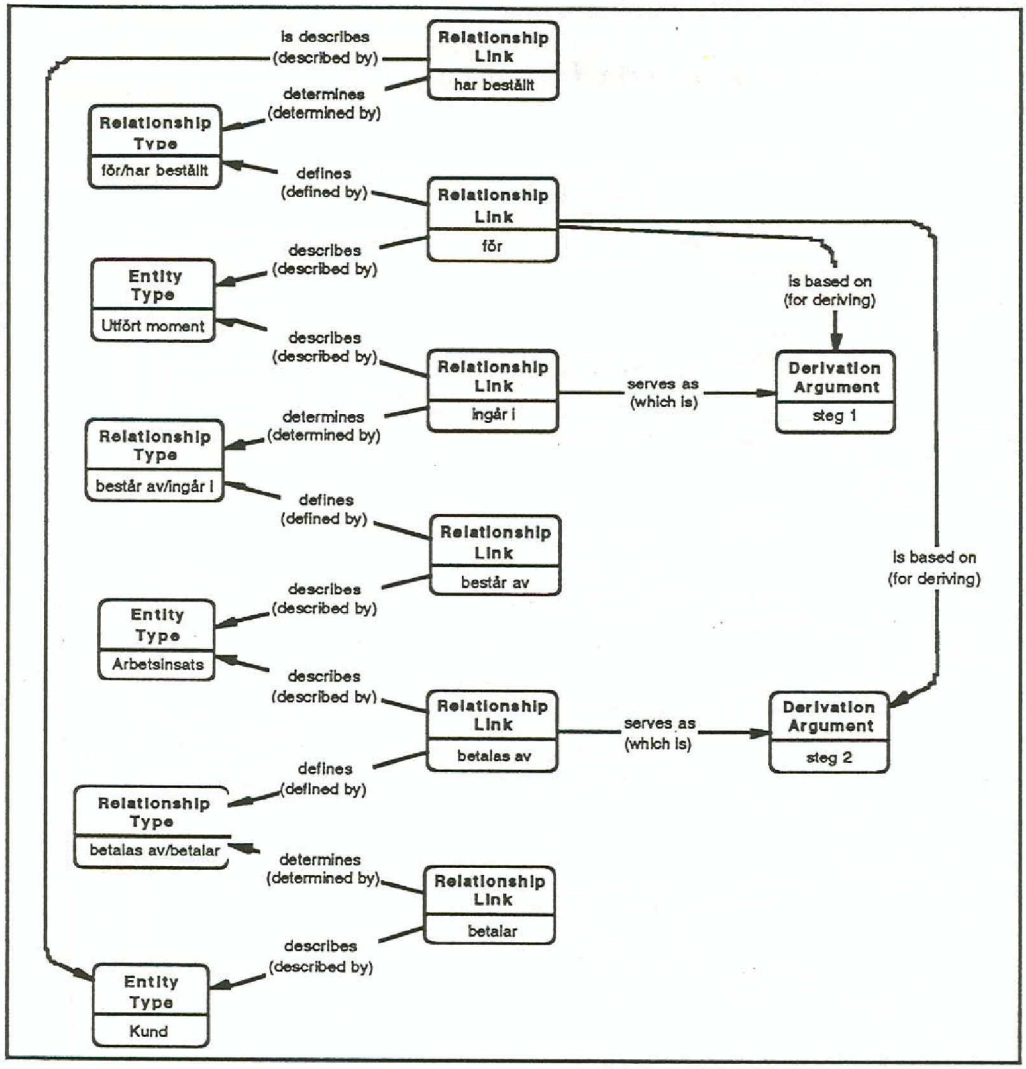
5.1 Indirect Relationship Link

En Relationship Link utgår från en beskriven Entity Type och pekar på en beskrivande Entity Type. Ibland kan det finnas behov av att relatera en beskriven Entity Type med en Entity Type, vilken återfinns först efter indirekt navigering över flera Relationship Links i begreppsmodellen. Denna kedja av Relationship Links vill man kanske namnge så att den ur beskriven Entity Types perspektiv upplevs som en helt vanlig, namngiven Relationship Link.

Enligt samma mönster som tidigare definieras en indirect Relationship Link med hjälp av referens till ett antal Derivation Arguments över "is based on/for deriving". Entydig navigering kräver att respektive Derivation Argument placeras i rätt ordning, något som hanteras automatiskt inom IM i och med att denna relationship type anges som Ordered (framgår inte av vår begreppsmodell). Genom en indirect Relationship Link kan två Entity Types bindas ihop över en godtyckligt lång kedja. Det enda som krävs är att kedjan hänger ihop, dvs beskrivande Entity Type över en Relationship Link måste vara beskriven Entity Type över nästa Relationship Link, osv.

Observera att definitionen binds till en Relationship Link men att även en neutral (indirekt) Relationship Type måste skapas. Vad gör man i detta läge med inversen? Måste den definieras som en indirect Relationship Link med samma kedja definierad "åt andra hållet"? Enligt IM behöver inversen definieras som en Relationship link med ett namn men utan den indirekta definitionen. Inversen anses definierad som huvudriktningens Relationship Link men tagen bakifrån. Värdet IN(direct) hos attribute type "derivation type" för Relationship Type ger information om att den ska hanteras som indirect.

Antag att vi önskar etablera en Relationship Link för *Utfört Moment* med namnet *för*, refererande till den *Kund* momentet utförs för. Inversen kallar vi *har beställt.*

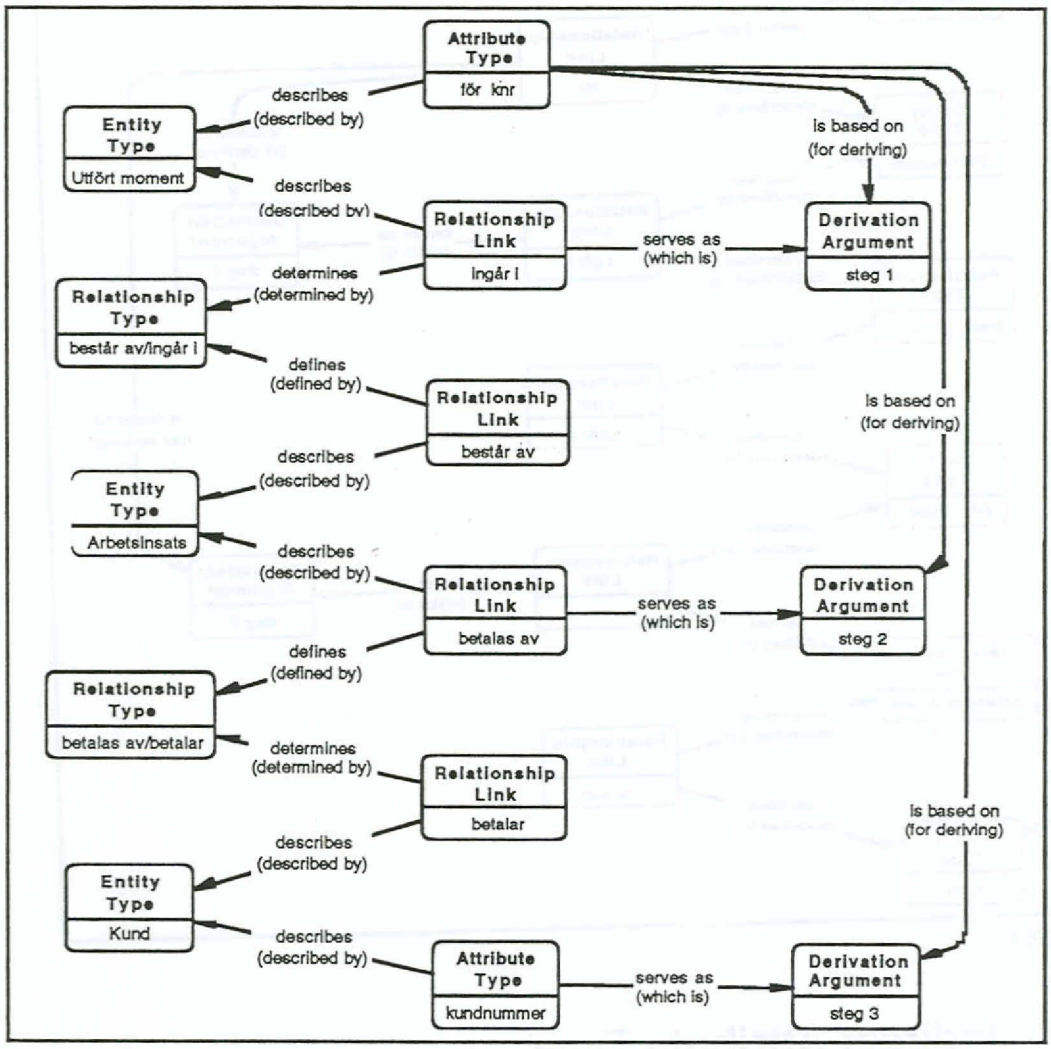


FIGUR 8

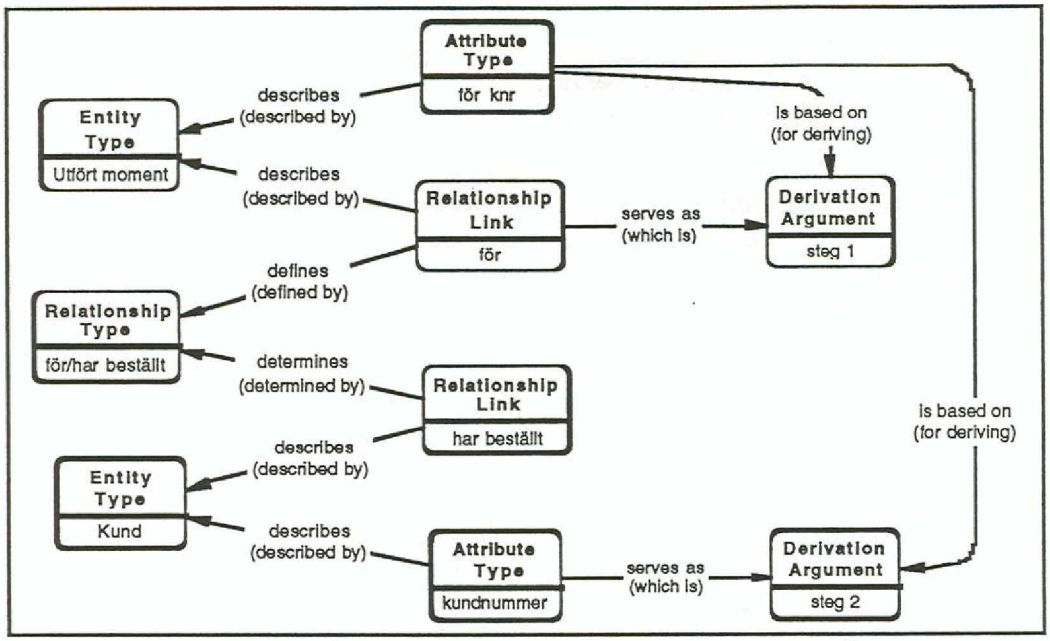
5.2 Indirect Attribute Type

Indirect Relationship Links är uppbyggda av ett antal andra Relationship Links. Indirect Attribute Types har i princip samma uppbyggnad. Skillnaden ligger i att den avslutande länken i kedjan måste vara en Attribute Type.

Antag samma situation som ovan, men att det inte är en *Kund* i sig utan *Kund*ens *kundnummer* som efterfrågas. Indirect Attribute Type för Entity Type *Utfört moment* får heta *för knr*. I det första alternativet (figur 9) antas vi inte känna till nyligen definierad indirect Relationship Link, men väl i det andra (figur 10).

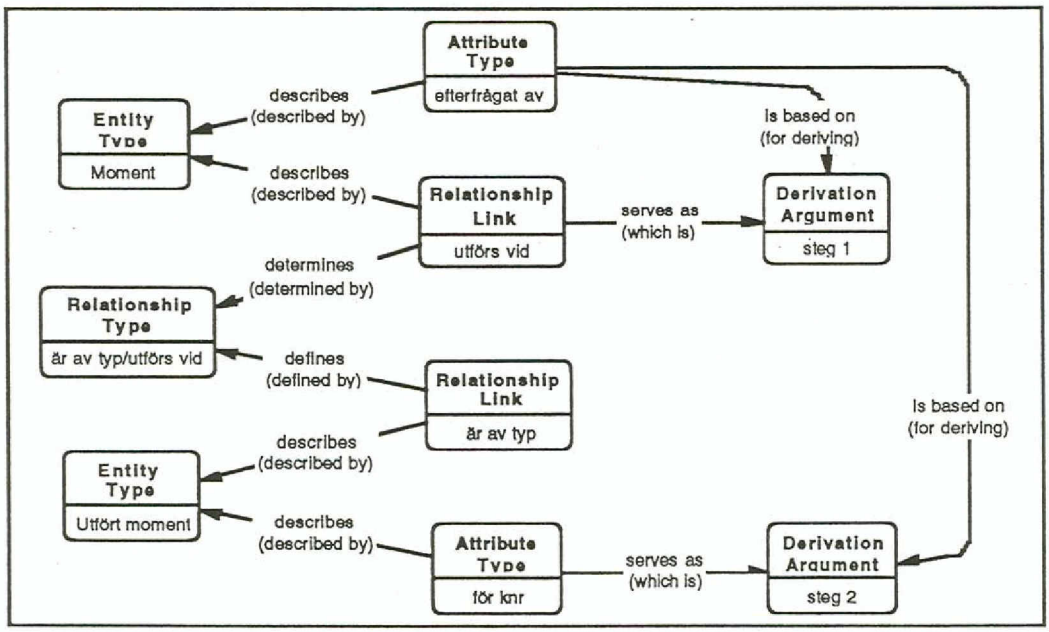


FIGUR 9



FIGUR 10

Observera, att avslutande Attribute Type mycket väl kan vara en indirect Attribute Type. Antag att vi för Entity Type *Moment* vill etablera en indirect Attribute Type *efterfrågat av*, bestående av alla *kundnummer* för alla de som någon gång betalat för ett visst *Moment*. Lyckligtvis finns redan indirect Attribute Type *för knr* under *Utfört moment*. Den indirekta definitionen behöver därför bara omfatta två steg: *utförs vid* och *för knr*. Se resulterande förekomstmodell i figur 11. Definitionen av *för knr* visas inte, den antas vara utförd enligt figur 9 eller 10.

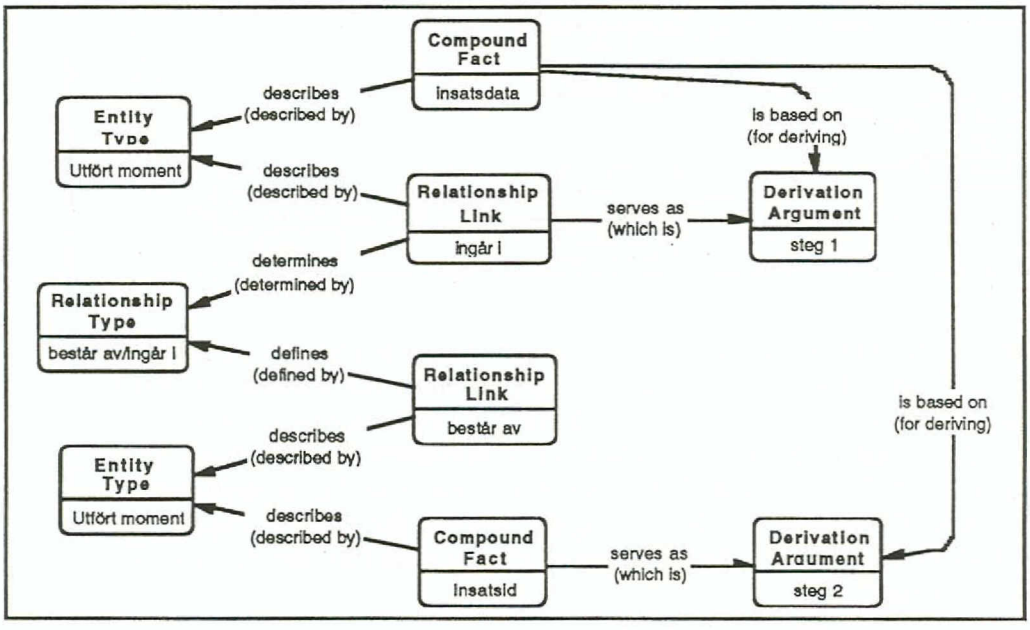


FIGUR 11

5.3 Indirect Compound Fact

Indirect Attribute Types består av en eller flera Relationship Links avslutade med en Attribute Type. Samma sak gäller indirect Compound Facts, med den skillnaden att de avslutas med ett Compound Fact.

Vi kompletterar *Utfört moment* med ett indirect Compound Fact *insatsdata*, uppbyggd genom Relationship Link *ingår i* och tidigare definierat Compound Fact *Insatsid* under *Arbetsinsats*. Se figur 12. *Insatsdata* avses ge en sammansatt information om den *Arbetsinsats* ett visst *Utfört moment* tillhör. För definition av *Insatsid*, se figur 7.



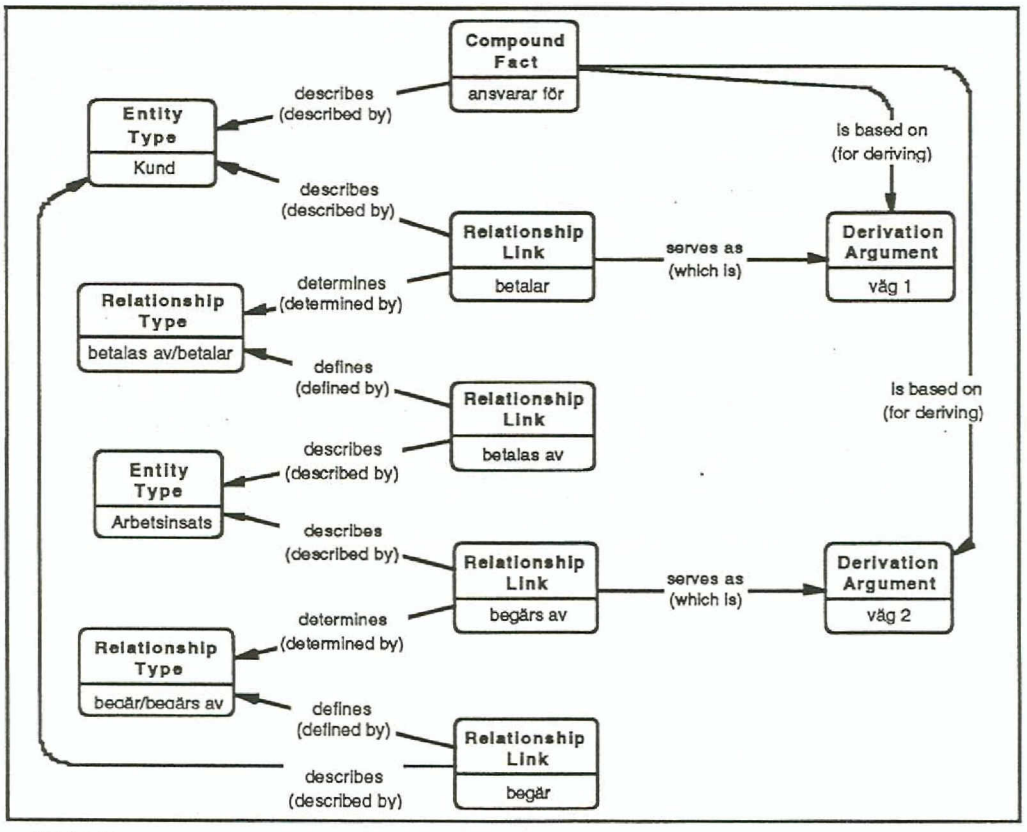
FIGUR 12

6. Union fact types

Composite fact types används när man vill åstadkomma värden, där vart och ett är uppbyggt som en kombination av ingående delar. Indirect fact types används när man vill referera till något över en definierad navigering i modellen. Union fact types kännetecknas av att varje Derivation Argument står för sin egen referens eller fact type. Respektive resultat sammanförs med mängdoperatorn UNION till en enda fact type.

Fact type för ett Derivation Argument kan, i detta fall, vara av vilken typ som helst. Följdaktligen kan resulterande, sammanslagen fact type ha en inte entydigt tolkbar blandning av värden. Därav följer att endast Compound Fact kan ingå i rollen som union fact type. Det enda kravet som i princip ställs är att varje Derivation Argument ska avse fact types från samma Entity Type.

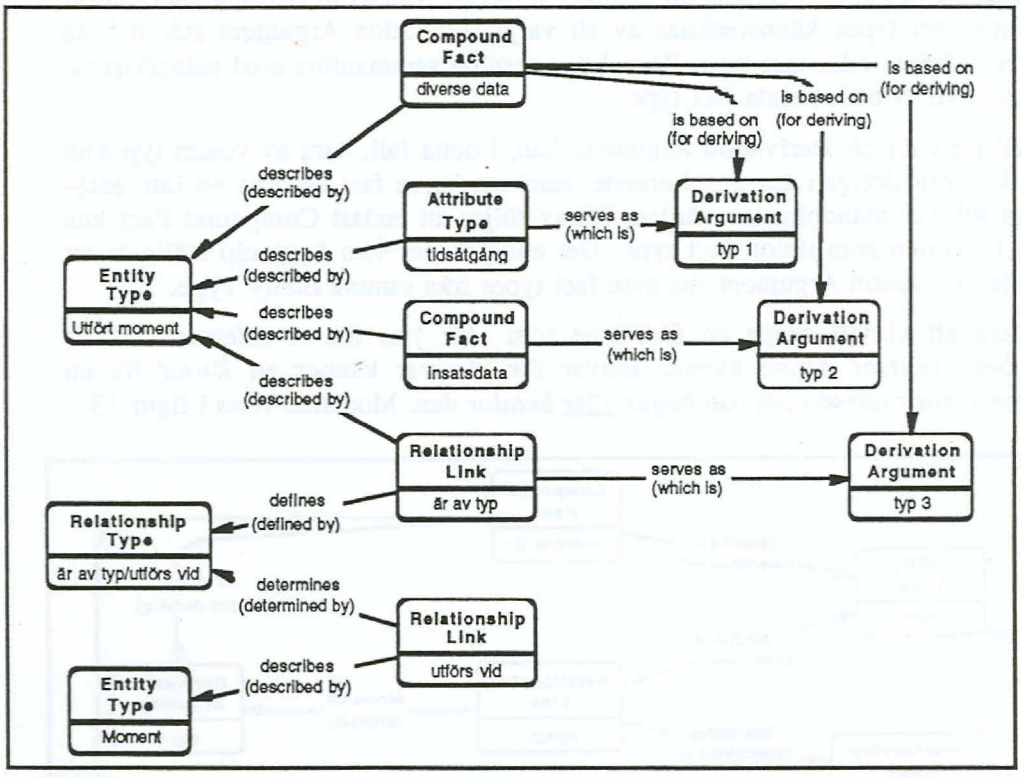
Antag att vi vill skapa en fact type som för viss *Kund* refererar till de *Arbetsinsats*er denne känner ansvar för. Ansvar känner en *Kund* för en *Arbetsinsats* oavsett om han *begär* eller *betalar* den. Modellen visas i figur 13.



FIGUR 13

Samtliga resulterande värden är här identifierare för *Arbetsinsats*er. Ett annat sätt att formulera samma fact type visas i avsnitt 7.3.

Vill vi av någon outgrundlig anledning för *Utfört moment* definiera en fact type *diverse data* bestående av den samlade mängden värden över *insatsdata, tidsåtgång* och *är av typ*, går det bra. Resulterande fact type blir en blandning av värden från ingående fact types Attribute Type, Relationship Link och Compound Fact. Det är omöjligt att från resulterande värden avgöra varifrån de kommit. Se figur 14. Att *diverse data* är av kategorin union framgår av att dess "derivation type" satts till värdet UN.



FIGUR 14

Samma definition, men "derivation type"=CO skulle resultera i ett antal värden, där varje värde är sammansatt av tre delar, den första från *typ1*, den andra från *typ2*, den tredje från *typ 3*. Det skulle bli lika många sammansatta värden som antalet möjliga kombinationer av de tre delarna.

"Derivation type" = IN skulle resultera i ett felmeddelande.

7. Computed fact types

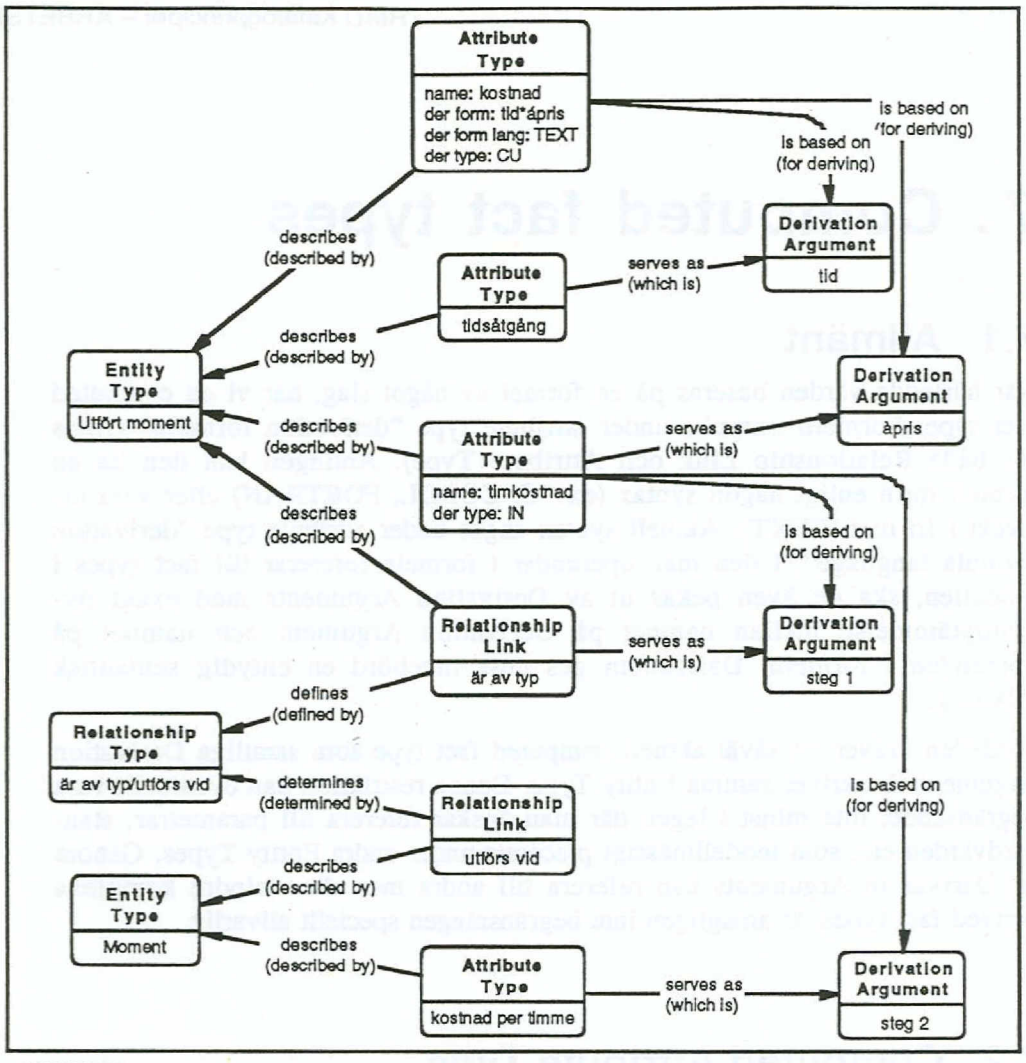
7.1 Allmänt

När härledda värden baseras på en formel av något slag, har vi en computed fact type. Formeln uttrycks under attribute type "derivation formula" (finns för både Relationship Link och Attribute Type). Antingen kan den ha en formell form enligt någon syntax (exv C, COBOL, FORTRAN) eller vara uttryckt i fri text (TEXT). Aktuell syntax anges under attribute type "derivation formula language". I den mån operander i formeln refererar till fact types i modellen, ska de även pekas ut av Derivation Arguments med exakt överensstämmelse mellan namnet på Derivation Argument och namnet på operanden i formeln. Därigenom ges dess innebörd en entydig semantisk tolkning.

Modellen kräver att såväl aktuell computed fact type som samtliga Derivation Arguments beskriver samma Entity Type. Denna restriktion kan eventuellt vara begränsande, inte minst i lägen där man önskar referera till parametrar, standardvärden etc, som modellmässigt placerats under andra Entity Types. Genom att Derivation Arguments kan referera till andra mer eller mindre komplexa derived fact types, är antagligen inte begränsningen speciellt allvarlig.

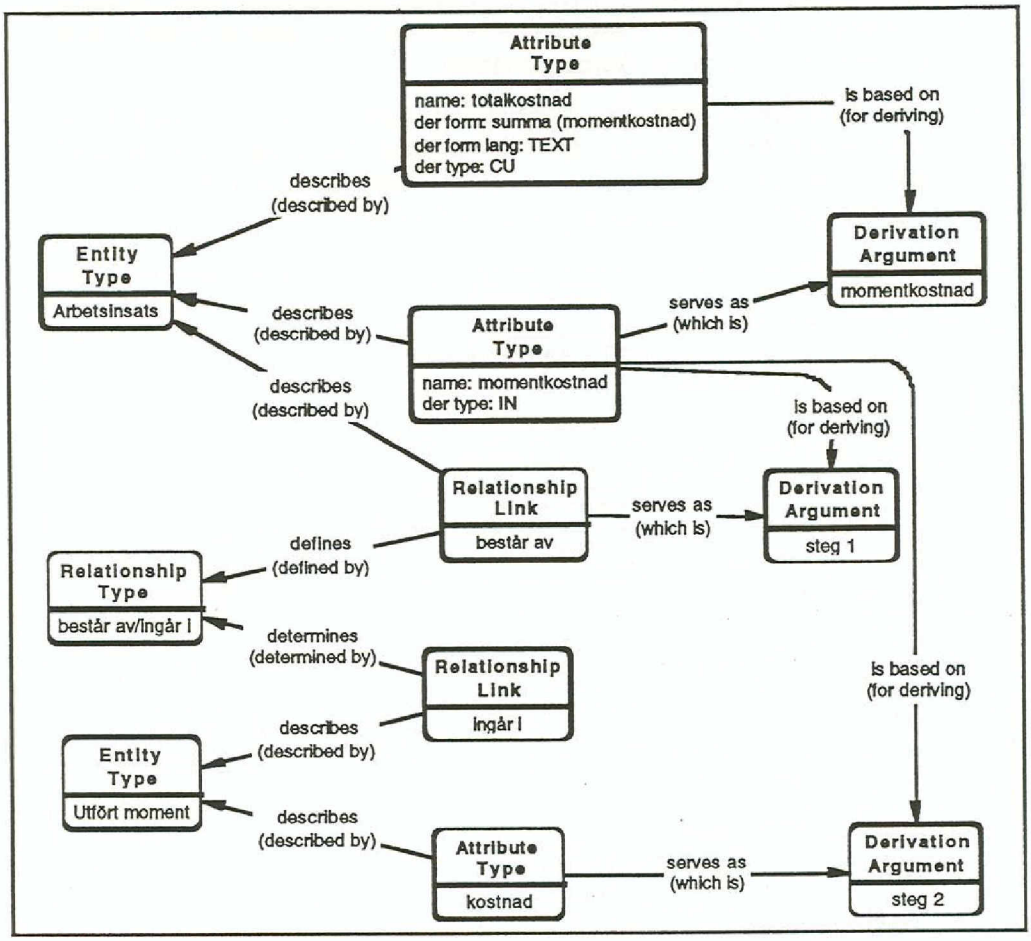
7.2 Computed Attribute Type

Åter till *Utfört moment* igen. Vi vill nu komplettera med en Attribute Type *kostnad*, som motsvarar (*tidsåtgång* * *kostnad per timme*) för refererat *Moment*. Först behöver vi skapa en indirekt Attribute Type *timkostnad* för att komma åt *kostnad per timme*. Se resulterande förekomstmodell i figur 15, där de två derived Attribute Types är beskrivna något utförligare än tidigare.



FIGUR 15

Komplettering av *Arbetsinsats* med Attribute Type *totalkostnad* ger samma principiella uppbyggnad som ovanstående. Se figur 16.



FIGUR 16

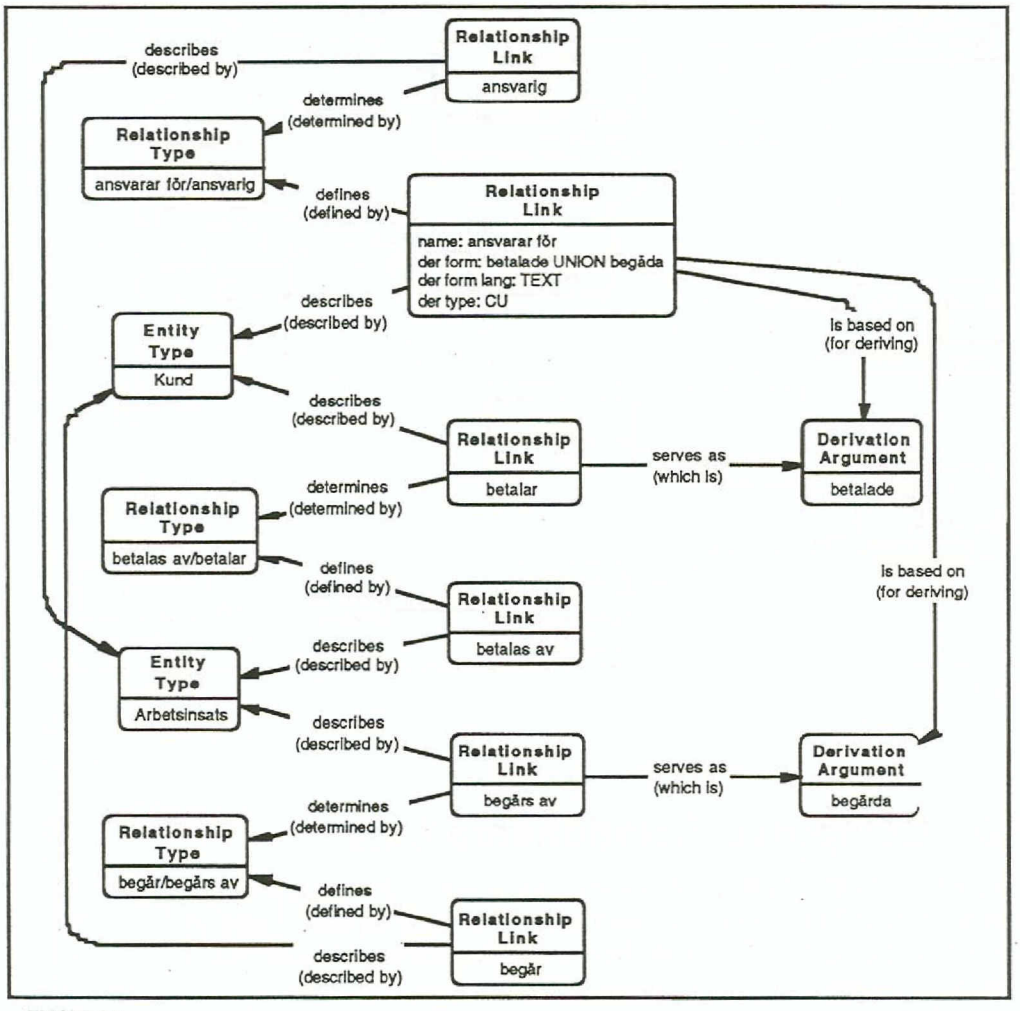
7.3 Computed Relationship Link

Parallella navigeringar, som startar vid samma Entity Type och slutar vid samma Entity Type, kan sammanföras under en computed Relationship Link. Varje navigering svarar mot ett Derivation Argument och resulterar i en mängd värden svarande mot identifierare för förekomster av refererad Entity Type. Computed Relationship Links "derivation formula" anger vilka operationer som ska utföras på mängderna för att avsedd slutmängd ska erhållas. Eftersom respektive delmängd (en per Derivation Argument) är förekomster av samma Entity Type, blir även den resulterande mängden entydigt förekomster av samma Entity Type.

Det är inget som hindrar att en computed Relationship Link ingår som steg vid definition av andra indirekta Relationship Links. Med hjälp av en kombination av indirect och computed Relationship Links kan mycket komplicerade samband upprättas. Sedan återstår det för Case-verktyg att kunna erbjuda ett överskådligt gränssnitt för hanteringen.

Ett bekymmer i detta sammanhang: Hur definierar man inversen över en computed Relationship Link? Vid indirect underförstås baklängesnavigering, men hur tar man exv inversen av en skärningsmängd?

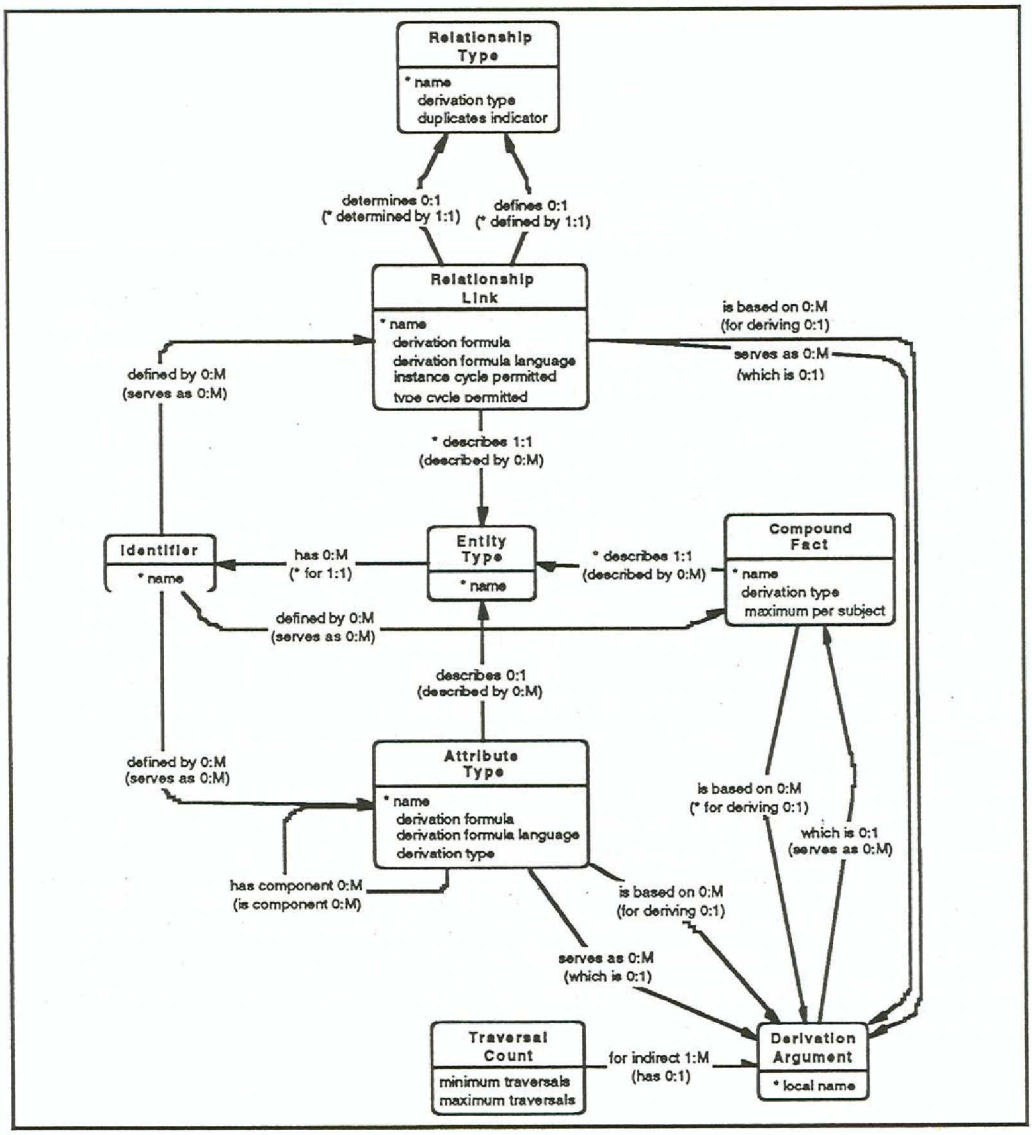
Ta exemplet från avsnitt 6 med en generell *ansvarar för* Relationship Link mellan *Kund* och *Arbetsinsats*. Den formuleras här som en computed Relationship Link. Den som så önskar kan sedan använda *ansvarar för* som en länk i en kedja, exv för att sammanbinda *Kund* med *Moment*.



FIGUR 17

8. Sammanfattning

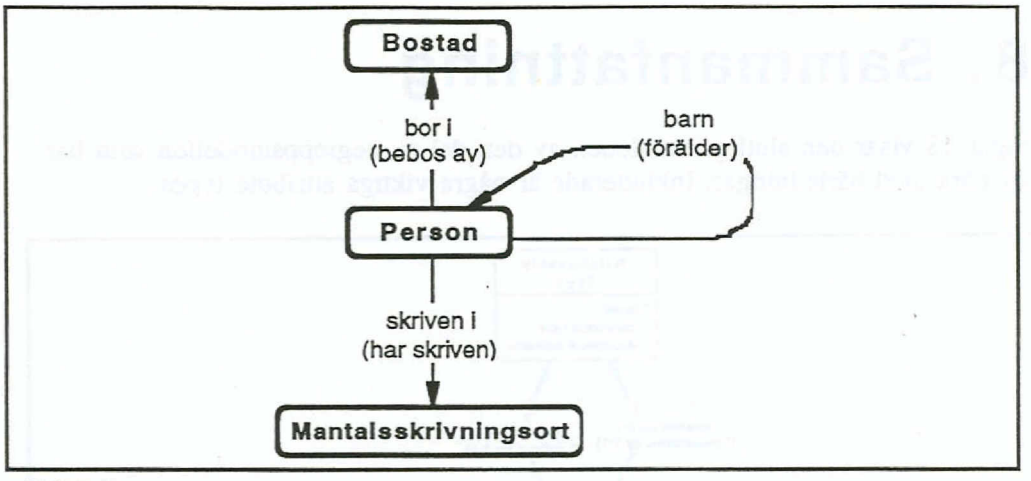
Figur 18 visar den slutliga versionen av den del av begreppsmodellen som har att göra med härledningar. Inkluderade är några viktiga attribute types.



FIGUR 18

Den skarpögde ser en ny entity type, **Traversal Count**. Dess syfte är att reglera minimum och framför allt maximum antal "varv" en rekursiv Relationship Link ska snurra som steg (Derivation Argument) i definitionen av en indirect Relationship Link. Antag modell i figur 19. Vill vi definiera en indirect Relationship Link för *Bostad* som relaterar den till dess boendes barnbarns *Mantalsskrivningsort*, blir den uppbyggd av tre steg (Derivation Arguments).

Steg 1 går över *bebos av*. Steg 2 går över *barn* med Traversal Count (maximum traversals = 2) kopplat till dess Derivation Argument. Steg 3 går över *skriven i*.



FIGUR 19

Varför "for indirect" angivits till 1:M istället för 1:1 är oklart. Varför Traversal Count behöver existera är också oklart. Med 1:1 som förutsättning kunde dess attribute types övertas av Derivation Link.

Attribute type "duplicates indicator" vid Relationship Type kan anta värdena Y(es) eller N(o). Dess syfte är att reglera hur man inom en navigering ska hantera eventuell uppkomst av värdeduplikat. Skapar vi exv en indirect Relationship Link från *Moment* till *Kund* är det sannolikt att ett visst *Moment* begärts av samma *Kund* flera gånger. Vill vi använda denna indirect Relationship Link för att (via en indirect Attribute Type) ta fram *fullständigt namn* för dessa *Kund*er, är det sannolikt önskvärt att *Kund*-duplikaten tas bort innan fullständigt namn tas fram. Vi vill ju inte få ut samma "Kalle Karlsson" 10 gånger. I andra sammanhang, exv vid vissa summeringsoperationer, är det tvärtom avgörande att duplikaten finns med för att resultatet ska bli korrekt.

Sammanfattningsvis, den redovisade modellen tillåter formulering av ett stort antal olika typer av härledningsregler. Man har valt att inte formalisera fullt ut. Vissa formler ligger som textsträngar, uttryckta efter någon angiven syntax, medan modellen endast pekar ut de inblandade komponenterna. Några garantier för konsistens mellan referenser och textinnehåll finns inte. Den som vill veta var och hur en komponent används i en härledningsregel får gå till texten. Å andra sidan, vem vill veta det, och i så fall när, under ett modelleringsarbete?

TRIAD utvecklar IA

Televerket har just tagit första steget in i sin nya IA-organisation och Posten håller på att bygga upp sin nya DA-organisation. Båda organisationerna har sett nyttan att inför 90-talet gå vidare tillsammans i TRIAD-projektet som drivs tillsammans med SISU. Statskontoret deltar också i projektet för att på sikt kunna föra ut nya synsätt och hjälpmedel inom den civila statliga sektorn.

Ericsson Data Services deltar med tyngdpunkten i den del som handlar om att utveckla kompetenta modelleringsledare, delprojektet "Avancerad utbildning för modelleringsledare".

Modelleringsmetoder är centrala i bedrivandet av verksamheten inom informationsadministrationen. Därför arbetar ett delprojekt med utvecklandet av "nästa generation modelleringsmetod" som skall sättas i händerna på informationsadministratören. Siktet är att fördjupa och bredda dagens modelleringsmetoder och där hämta in kunskap från pågående forskning och utveckling internationellt. (faktaruta om IAS91).

Som stöd för informationsadministrationen behövs verktyg. Inom TRIAD arbetar man där inom två områden, kataloger och verktyg.

Delprojektet kataloger arbetar dels med att utforma den informationsmodell som måste kunna täckas av en katalog, dels med att granska och följa utvecklingen av produkter inom området t ex IBM:s "Repository" och Digitals "CDD". Dessutom följer man standardiseringen internationellt kring IRDS. För parterna i projektet liksom för andra organisationer är detta ett tungt område både vad gäller kommande investeringar ekonomiskt och vad gäller kompetenta resurser för en kommande övergång till "repository-världen". - Det inledande skedet syftar till att bygga upp en kunskapsplattform, som sedan kommer att kunna utnyttjas för kravställande och planering och genomförande av övergång från dagens kataloghantering till morgondagens.

Den andra verktygshanterande delen inom TRIAD-projektet, delprojektet "verktyg för informationsadministration", syftar till att ta fram verktyg för uttag och dokumentering av modeller. Betoningen ligger på människa datorgränssnitt och i första skedet görs utveckling av HYBRIS-gränssnittet med prototyper för Posten och för Televerket.

För att hålla ett helhetsperspektiv på projektets delar och för att ha inpassningen av funktionen Informationsadministration i organisationens övriga verksamhet arbetar delprojektet "Krav på IA". I delprojektet arbetar man dels med att kartlägga dagens krav på dataadministration och projicera till morgondagens krav på IA. Dessutom skall man skapa en bild av IA-verksamhetens innehåll och organisation. Från detta i sin tur ställer man krav på övriga delprojekt. Vilka krav skall ställas på kompetens, metoder, hjälpmedel typ kataloger och gränssnitt?

TRIAD projektet är stort

Budgeten för TRIAD-projektet löper på 10 MSEK per år under en treårsperiod som startar vid kalenderåret 1991 års början och som alltså beräknas avslutad vid utgången av 1993.

TRIAD-projektet är ett tillämpningsprojekt

Det innebär att parterna, Televerket, Posten, Statskontoret, EDS och SISU går in med såväl persontidssatsningar som ekonomiska och att STU, Styrelsen för Teknisk Utveckling, bidrar med ett ekonomiskt tillskott som svarar mot ungefär 40 % av den insatta persontiden.

Öppet för fler deltagare

Parterna i TRIAD-projektet vill gärna öka tempot och bredda perspektivet och vill därför gärna ha fler parter in i projektet. Dessa parter får då enligt SISU:s tårtprincip "betala för en tårtbit, men ät hela tårtan", tillgång till projektets resultat med en insats som ger stor "price performance".

Nya deltagare kan gå in i hela projektet eller i det eller de delprojekt som verkar intressantast. En förutsättning är att man framförallt är beredd att satsa kompetent personal. För de flesta intressenter bord detta vara ett utmärkt sätt att driva personalutveckling för personer t ex inom DA-området, samtidigt som man bygger upp beredskapen inför 90-talets IA-verksamhet.

Kompetensutveckling viktigt resultat

En viktig effekt för parterna av deras medverkan i TRIAD är kompetensutveckling. Man satsar på att ta in personer som så småningom eller redan idag arbetar med DA och IA för att ge dem en djup och "frontlinje"-mässig kompetens. Detta skall utnyttjas när man successivt för in resultaten i den egna organisationen. Projektdeltagarna har alltså en viktig roll som kunskapsförmedlare i den egna organisationen. Dessutom ger projektarbetet deltagarna tillfälle till en egen utveckling inom det professionella området som är unik.

Informationsspridning

Det sjätte delprojektet "Informationsspridning" har till uppgift att sörja för att i första hand parterna men också SISU:s övriga intressenter successivt kan följa och tillgodogöra sig resultat från TRIADprojektet. Seminarier, rapporter och referensgruppsverksamhet är led i den verksamheten.