TRIAD Arbetsrapport K 12

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

AD/Cycle I Information Model

Processer och informationsflöden mellan processer

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 informationsflöden mellan processer

Stig Berild
SISU & Sveriges Tekniska Attachéer

Spridningsförbehåll:
Denna rapport får endast spridas och användas inom de organisationer som deltar som parter i TRIAD-projektet.

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.

SISU

Läsarrapport för TRIAD rapport K 12: AD/Cycle I
Information Model – Processer och och informationsflöden mellan processer

Datum:

Namn:

Företag:

returneras till:

SISU
Lars Bergman
Box 1250
164 28 Kista

Lämna gärna kommentar till enkäten på baksidan. ☐
Kryssa om du kommenterar.

SVARET önskas när du läst!

01 Jag har lämnat den vidare till följande person/er:

02 Jag har cirkulerat den till följande personer:

03 Jag har läst rapporten (ange ungefärlig tid om du kan)		
04 Värdet för mig i mitt jobb nu (Stort=3, ganska stort=2, något=1, inget=0)		
05 Värdet för mig i mitt jobb framöver (Stort=3, ganska stort=2, något=1, inget=0)		
06 Värdet för min allmänna kompetensutveckling (Stort=3, ganska stort=2, något=1, inget=0)		
07 Detaljeringsgraden var (För djup=3, lagom=2, för ytlig=1, ingen åsikt=0)		
08 Dispositionen av innehållet var (Bra=2, Inte bra=1, vet ej=0)		
09 Innehållets tillgänglighet (Lättillgängligt=3, krävde viss ansträngning=2, krävde stor ansträngning=1, vet ej=0)		
10 Språket (klart=3, något oklart=2, mycket oklart=1, vet ej =0)		
11 Illustrationerna som lässtöd (gav gott stöd vid läsningen=3, gav visst stöd vid läsningen=2, gav dåligt stöd vid läsningen=1, vet ej=0)		
12 Användningen av illustrationer (För få=3, Lagom=2, För många=1, Vet ej=0).		

Returneras till
SISU, Lars Bergman
Box 1250, 164 28 Kista

Läsarrapport för TRIAD rapport K 12: AD/Cycle I Information Model – Processer och och informationsflöden mellan processer

Jag har lämnat dessa uppgifter tidigare. ☐

Din befattning i korta ord

Din enhet i korta ord

Ditt personliga intresse i sammanhanget

Din erfarenhet i korta drag

Din utbildning

Din kommentar till rapporten i övrigt:

AD/Cycle Information Model:

Processer och informationsflöden mellan processer

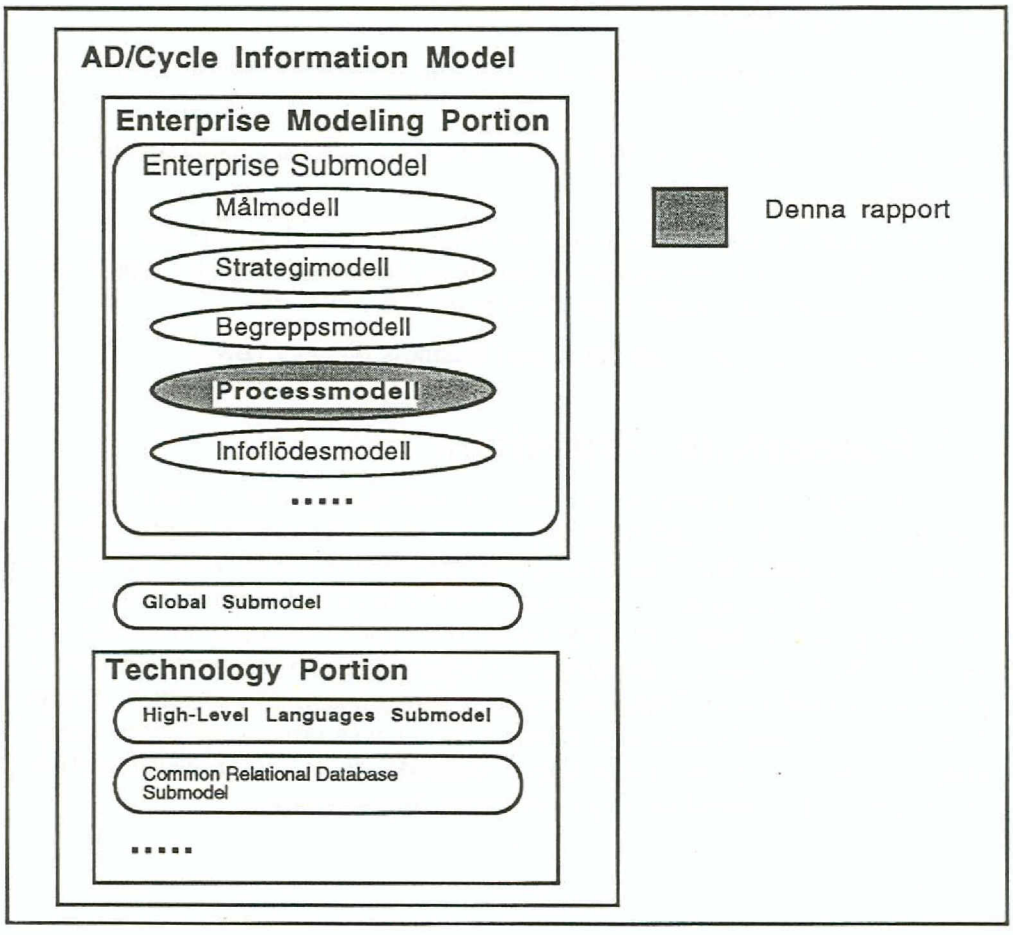
Innehåll

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 tar upp den modell som stödjer beskrivning av processer och informationsflöden mellan processer. Modellen återfinns inom Enterprise Submodel. Där finns bla även begreppsmodellen, vars huvudsakliga omfattning beskrivits i rapporterna TRIAD K6, K7 och K15.

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



FIGUR 0

Processmodellen är till för att stödja de metodansatser som brukar gå under beteckningen Structured Systems Analysis. Inom denna domän återfinns Yourdon, Gane and Sarson, DeMarco, mfl. De har mycket stora likheter med varandra varför det bedömts vara möjligt att finna en generell modell inom IM

för att klara samtliga varianter (mer eller mindre). Till vissa delar ligger skillnaderna på den grafiska notationen, något som ju inte påverkar modellens uppbyggnad.

1.2 Använd notation

Denna rapport använder en något enklare notation än tidigare rapporter. 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.

Exempel på verksamhetsmodellnivå anges som tidigare kursiverat. Ofta börjar de nu med stor bokstav.

Exempeltexten från rapport TRIAD K6, avsnitt 2, blir nu, något modifierad, som följer:

Exempel på förekomst av entity type är Process. Exempel på förekomst av entity type Process är *Fakturautskick*, medan en förekomst av entity type Entity Type kan vara *Person*. Entity type *Kund* är ett annat exempel.

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

1.3 Rapportens disposition

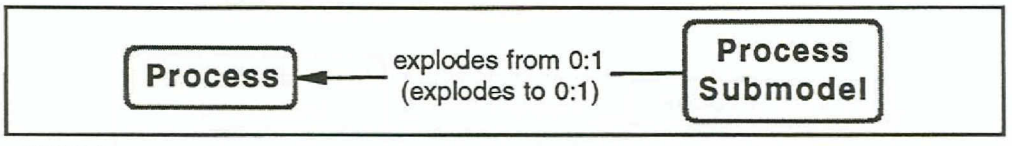
De entity types, som behövs för att beskriva uppbyggnaden av en enkel process, introduceras i avsnitt 2. Bekymret uppstår när vi behöver beskriva processer på olika detaljeringsnivåer. Beskrivningar på de olika nivåerna berör ju samma sak, bara med olika grad av precision i beskrivningen. De behöver beskrivningsmässigt hänga ihop. De entity types som behövs för detta introduceras i avsnitt 4. Dessförinnan har ett flernivå exempelunderlag presenterats i avsnitt 3. Exemplet, formulerat med hjälp av avsnitt 4-begreppen, visas i avsnitt 5. Avsnitt 6 innehåller ett mer komplicerat exempel för att visa processmodellens generalitet. Avsnitt 7 tar kortfattat upp resterande delar av processmodellen. Avsnitten 8 och 9 innehåller några avslutande kommentarer.

2. Processmodellens grundläggande entity types

2.1 Process, Process Submodel

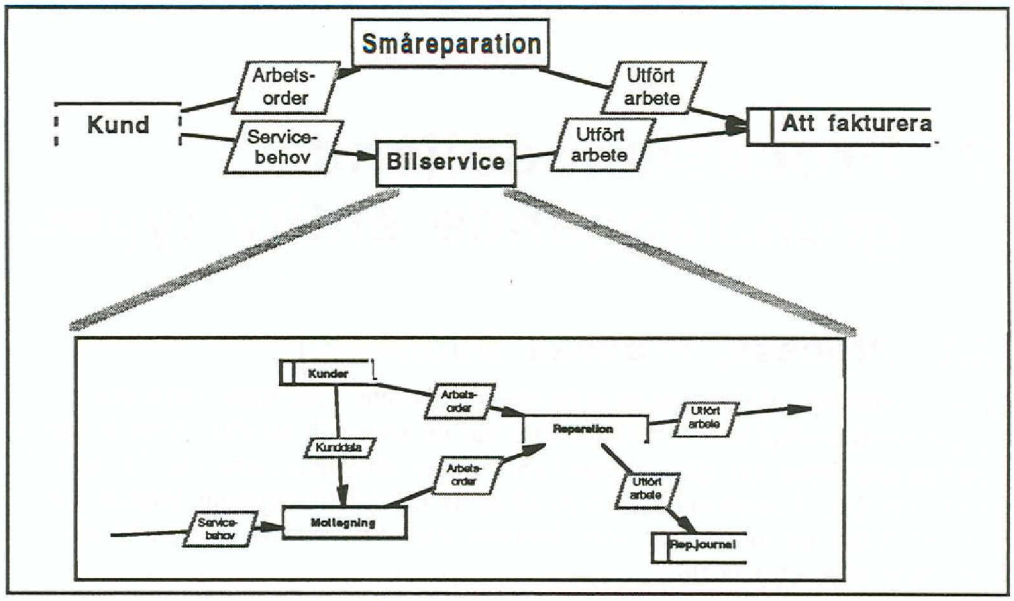
I en verksamhet pågår normalt olika slags aktiviteter. Där finns aktiviteter som pågår kontinuerligt och aktiviteter som har en start och ett slut. Även kontinuerliga aktiviteter kan ofta indelas i en följd av aktiviteter, som var och en har start och slut. I samband med utveckling av informationssystem fokuseras intresset kring de aktiviteter som bedöms vara intressanta informationskonsumenter eller -producenter.

En verksamhetsorienterad aktivitet med en definierad början och slut kallas i IM för en **Process**. Innebörden av mer komplexa eller sammansatta processer beskrivs i form av ett data flow diagram. Komponenter i ett sådant diagram kan vara processer, som i sin tur beskrivs i form av data flow diagrams. Denna successiva precisering kan behöva utföras över ett antal nivåer. En Process beskrivs alltså dels genom ett antal attribut, dels vid behov av ett data flow diagram. Ett data flow diagram kallas i IM för **Process Submodel**. Vi får en första del av processmodellen enligt figur 1.



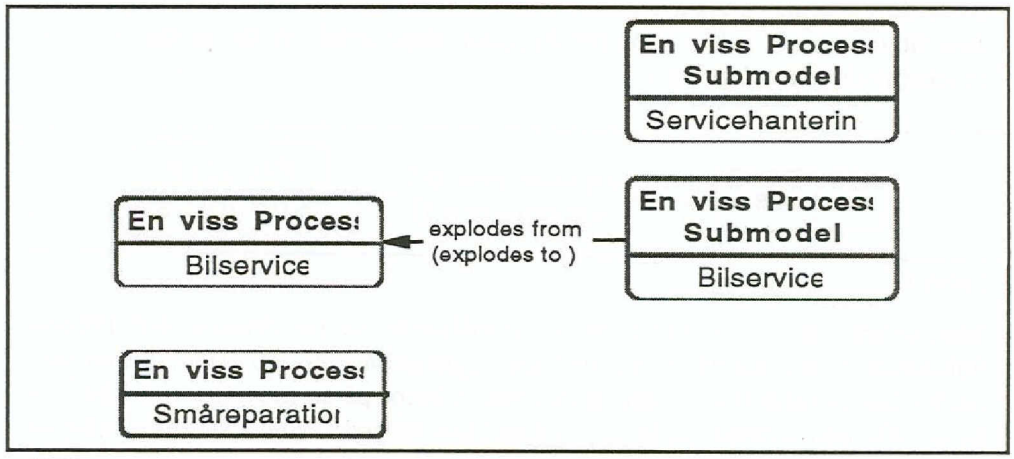
FIGUR 1

Som synes i modellen råder ett 1:1-förhållande mellan Process och Process Submodel. Processer på lägsta nivå har ingen submodel-definition. Att en submodel inte alltid behöver tillhöra en Process beror på att modellen vill tillåta en ansats där den översta nivån är ett data flow diagram, som inte nödvändigtvis går under en gemensam Process-beteckning. Så är fallet med exemplet i figur 2. Figuren visar en Process Submodel, som vi kan benämna *Serviceadministration*. En av submodellens komponenter är Processen *Bilservice*, som i sin tur definieras i form av en visad Process Submodel. Respektive submodels övriga beståndsdelar tas upp i kommande avsnitt.



FIGUR 2

I enlighet med begreppsmodellen i figur 1 blir förekomstnivån som följer (figur 3), om vi väljer att ge respektive Process Submodel samma namn som dess Processnamn.



FIGUR 3

Nå, hur kommer övriga typer av komponenter i en Process Submodel in i bilden? Hur ser modellen till att de hänger ihop? Över till avsnitten 2.2 och 2.3.

2.2 Info Flow, Info Store, External Agent

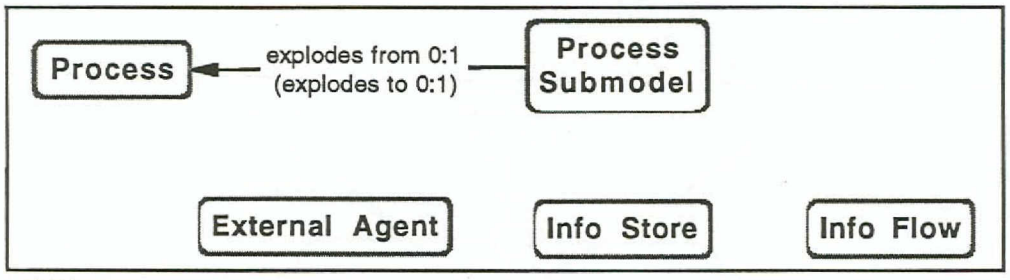
En vanlig komponent i data flow diagrams är, som vi redan sett, Processer. Dessa Processer hanterar bla data. För att kunna utföra sin uppgift behöver vissa Processer tillförsel i form av data. Andra Processer genererar data som ett resultat av sitt arbete medan ytterligare andra både tar emot och levererar data. Data till och från Processer kallas **Info Flow**.

Det som händer i ett data flow diagram är ingen isolerad företeelse, utan en del i ett större sammanhang, inom vilket det aktuella delproblemet avgränsats. För att ge en mer heltäckande bild innefattar normalt data flow diagrams också sin

närmaste kontaktyta med omgivningen. Kontaktytan består av samtliga omgivande företeelser (människor, maskiner, processer, ...) som tar emot eller levererar Info Flow. Dessa kallas i IM för **External Agents** och visas grafiskt i exemplen i form av gråtonade rektanglar. *Kund* i figur 2 ovan, är exempel på en External Agent.

Data kan också lämna en Process genom att det lagras undan för senare bruk utanför Processens kontroll. En sådan lagringsplats går under beteckningen **Info Store**. För grafisk symbol och exempel, se *Att fakturera* i figur 2. Det är inte heller säkert att utdata från en Process omedelbart går över i form av indata till nästa Process. Data behöver mellanlagras på obestämd tid. Även för detta ändamål används Info Store.

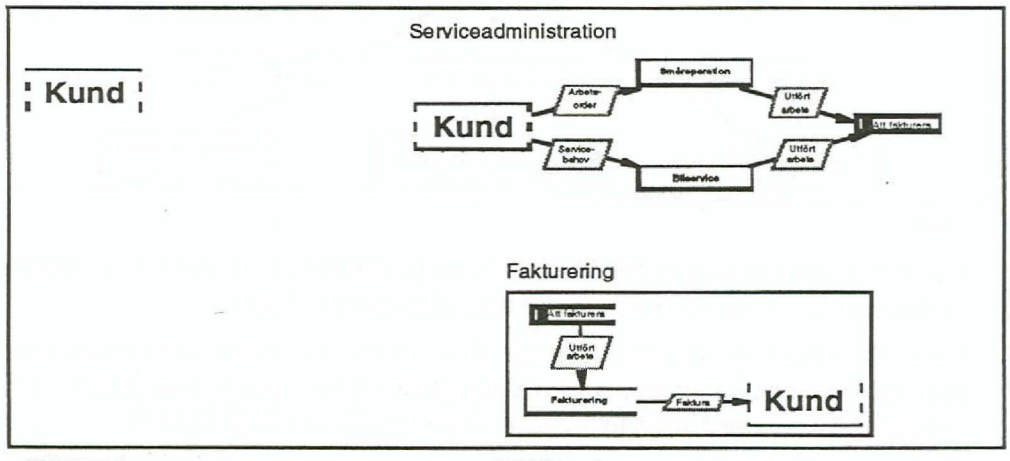
Vi kan nu komplettera processmodellen med de nya begreppen. Än så länge saknas de flesta relationship types. Nästa avsnitt diskuterar denna kompletteringing.



FIGUR 4

2.3 Process Context, Info Flow Vector

Möjliga komponenter i en Process Submodel är enligt ovan Process, External Agent, Info Store och Info Flow. Observera, att en förekomst av någon av dessa typer mycket väl kan ingå som komponent i flera olika Process Submodels. Exv kan *Kund* vara External Agent även i en eventuell Process Submodel *Fakturering*. Figur 5 a visar en External Agent *Kund*. Figur 5 b visar två exempel på External Agent *Kund* som komponent inom viss Process Submodel.

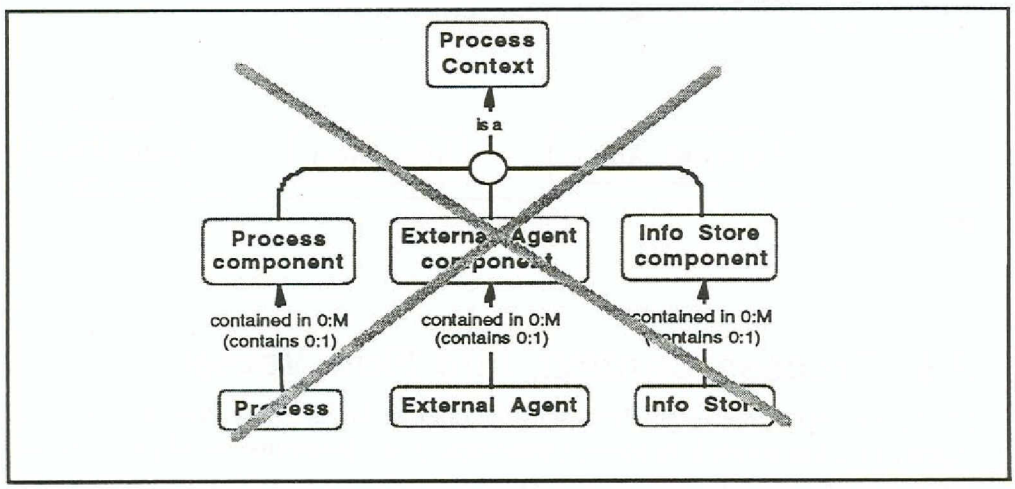


FIGUR 5 A

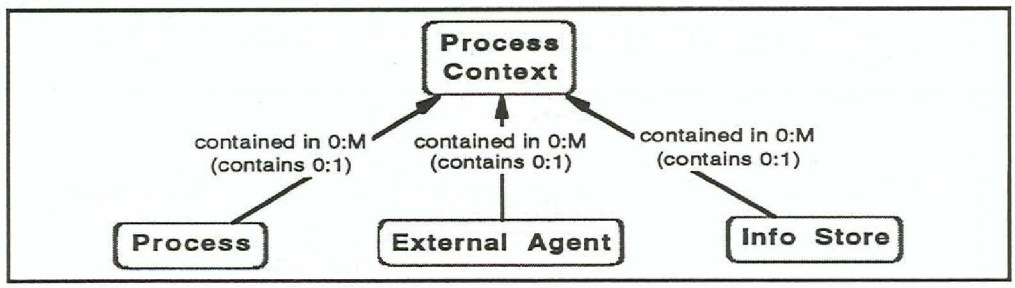
FIGUR 5 B

På samma sätt kan *Att fakturera* vara Info Store i exv både *Serviceadministration*, *Fakturering* och *Ekonomisk statistik*. Det är heller inget som hindrar att en viss förekomst uppträder som flera komponenter under samma Process Submodel. Se exv Info Flow *Utfört arbete* i figur 2, ovan.

Både den fristående aspekten och komponentaspekten måste kunna hanteras av processmodellen. Vi får entity types Process och Processkomponent (inom viss Process Submodel), External Agent och External Agentkomponent (inom viss Process Submodel), osv. Lyckligtvis visar det sig att tre av de fyra typerna av komponenter har en gemensam beskrivning. Komponentrollen för Process, External Agent och Info Store har, förutom gemensamma attribut, det gemensamt att var och en kan ta emot respektive lämna Info Flows. Detta manifesteras genom att de tre komponentrollerna samlas under en gemensam entity type **Process Context**. Med generaliseringssamband skulle exv BMs notation enligt figur 6 kunnat användas. Nu visar det sig att ingen av subtyperna representerar någonting unikt. Subtyperna behövs inte. Delmodellen kan därför förenklas till figur 7.



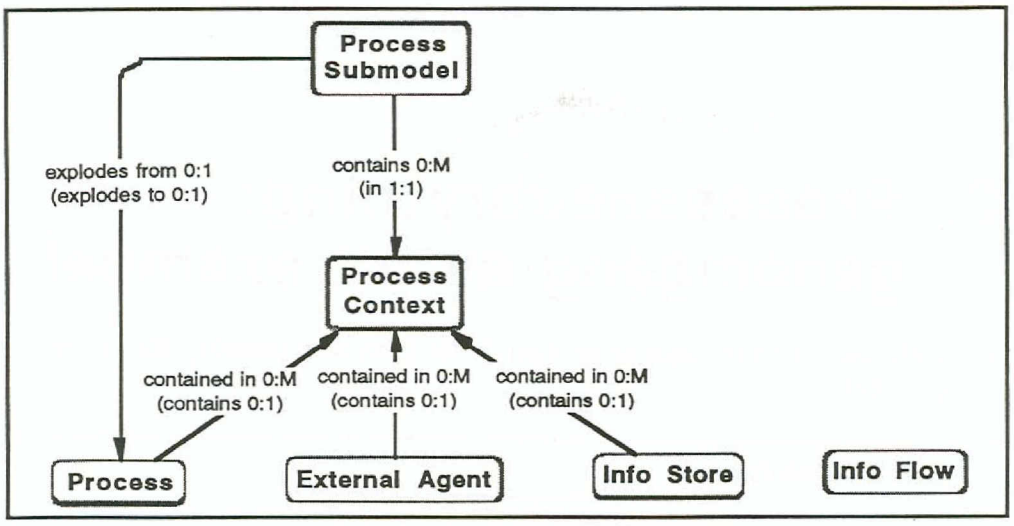
FIGUR 6



FIGUR 7

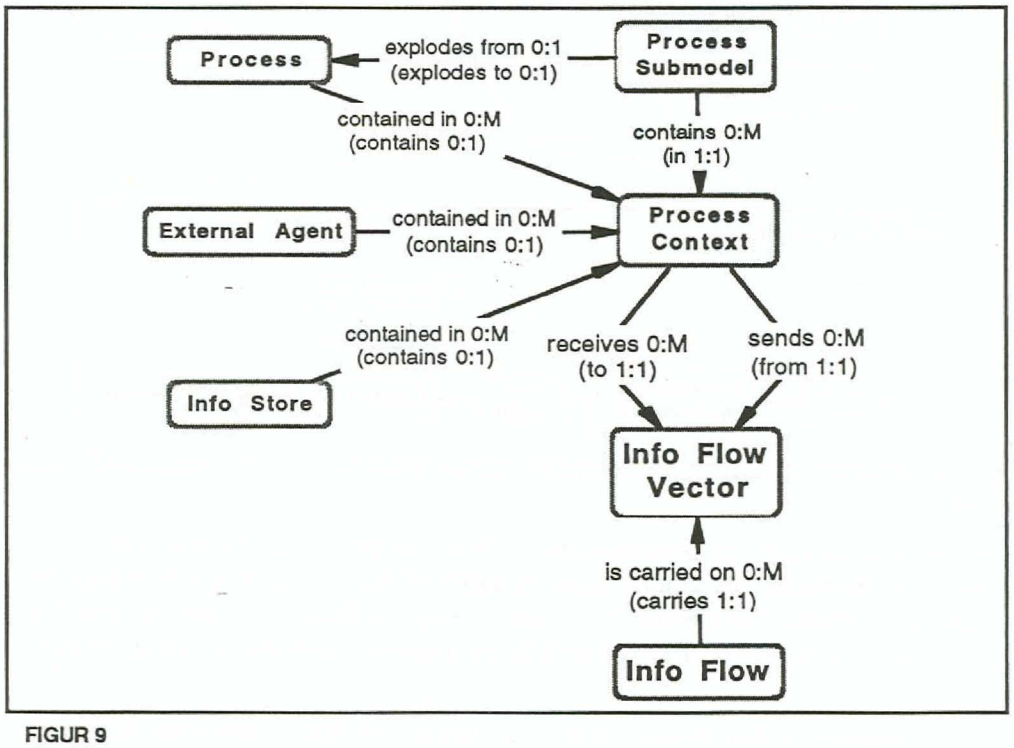
Figuren uttrycker (något förenklat) att varje förekomst av Process Context antingen är en Process, en External Agent eller en Info Store.

Varje förekomst av Process Context är i sin tur en komponent under viss Process Submodel. Kompletterar vi delmodellen från figur 4 med denna relationship type och med modellen i figur 7 blir resultatet enligt figur 8.



FIGUR 8

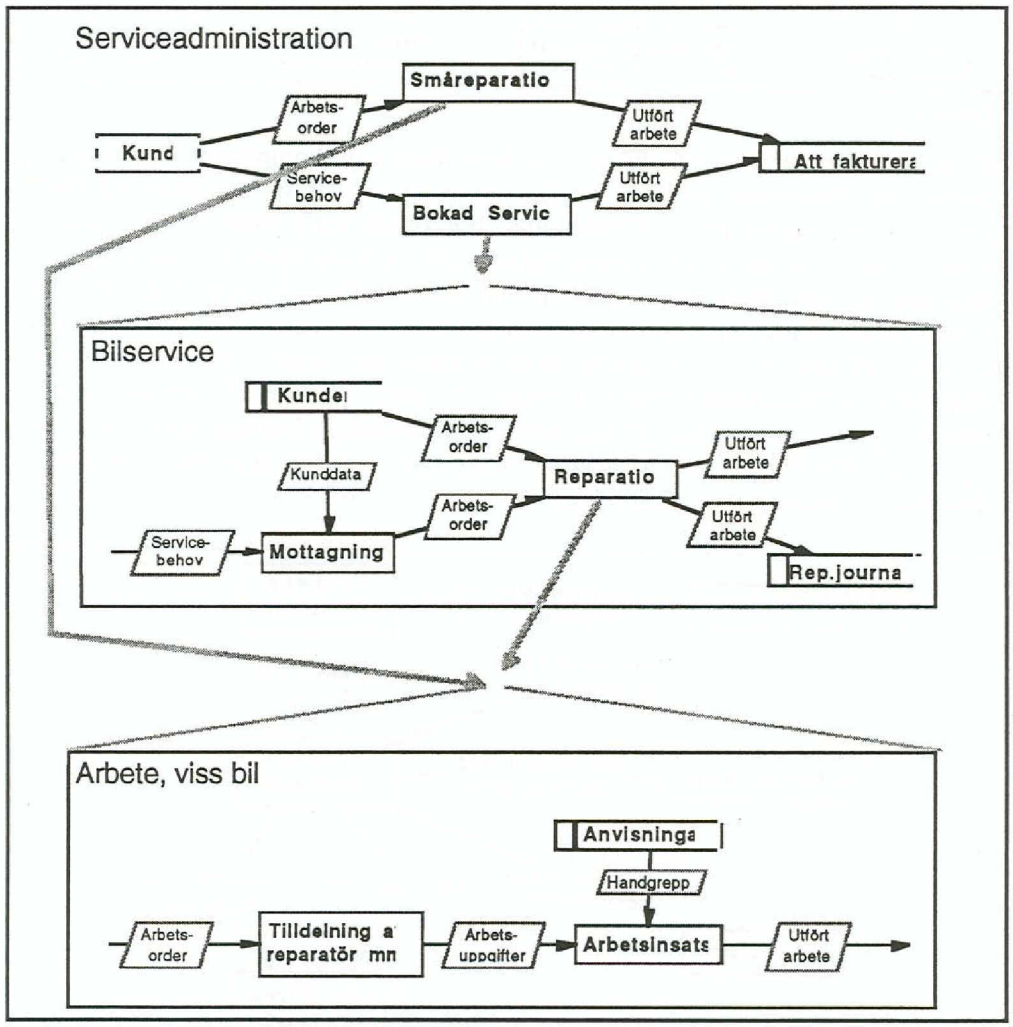
Kvar att införa är dels Info Flow, dels Info Flow som komponent i en Process Submodel. Observera återigen att om ett visst Info Flow återfinns på flera platser inom samma Process Submodel, blir också det i form av lika många komponenter. En sådan komponent kallas i IM för **Info Flow Vector**. I figur 2 återfinns tex Info Flow *Utfört arbete* som två Info Flow Vectors. En viss Info Flow Vector svarar mao mot ett visst Info Flow, "flödande" från viss Process Context till viss Process Context. Som vi snart ska se är detta en ungefärlig definition. Tillkommande behov kommer att ställa nya krav. Se av den anledningen figur 9 som ett mellansteg mot den fullständiga begreppsmodellen. Av layoutskäl har vi valt att flytta vissa entity types och att presentera de nytillkommande delarna med större text.



FIGUR 9

3. Processnedbrytning; genomgång av ett exempel

Figur 10, som är en utvidgning av figur 2, får tjäna som underlag för ett flernivåexempel.

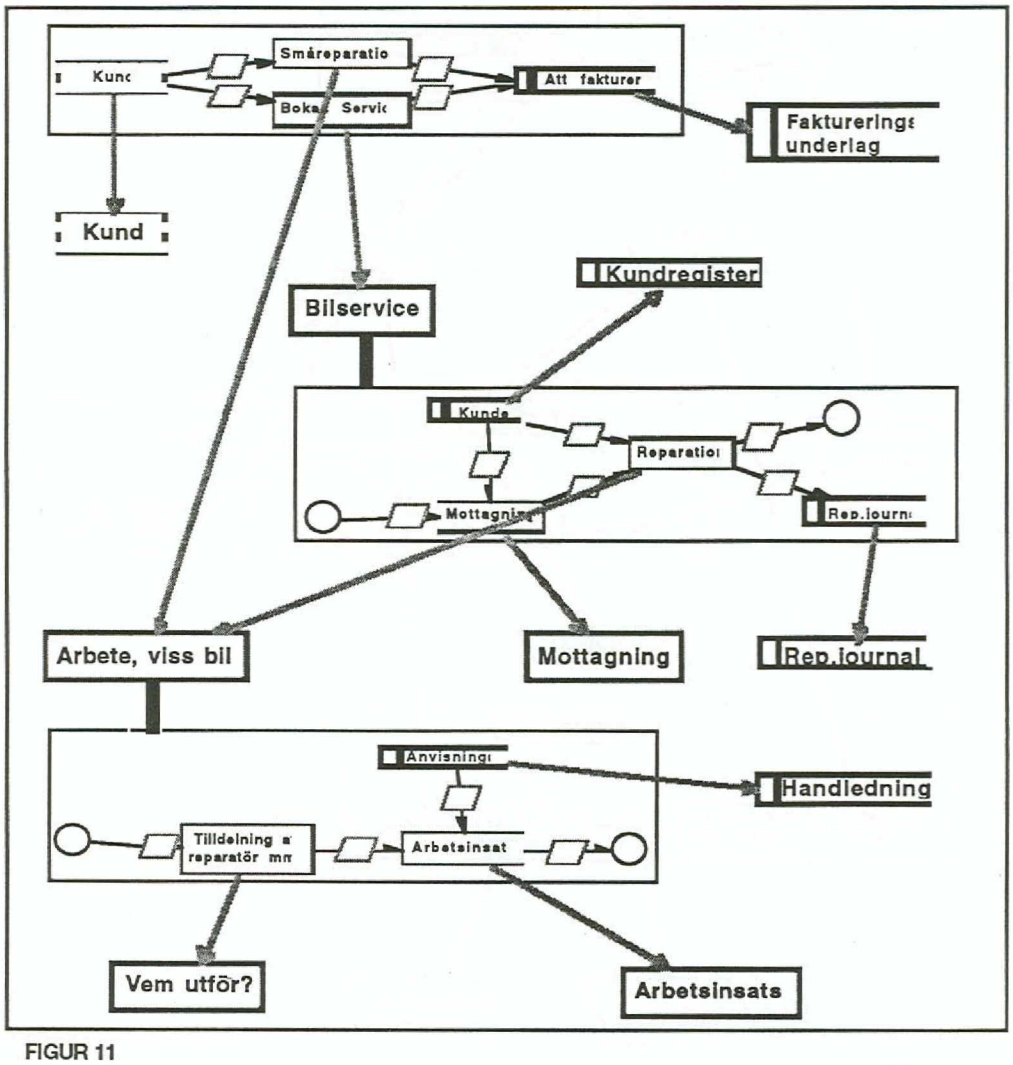


FIGUR 10

Observera återigen, att processmodellen är uppbyggd för att kunna tillåta olika namn att användas för samma saker på olika nivåer, dvs som komponenter under olika Process Submodels. Exv visar det sig att *Småreparation* på nivå 1 är samma sak som *Reparation* på nivå 2. Processens "neutrala" namn är *Arbete, viss bil.*

Distinktionen mellan Process, External Agent och Info Store som fristående företeelse och som komponent i en Process Submodel (som en Process Context) framgår av figur 11, nedan. Varje submodellbeskrivning inramas av

en rektangel. Dess komponenter pekar var och en på den motsvarande fristående företeelsen. Om denna är en Process kan det tänkas att den i sin tur specificeras i form av en submodell. Så är fallet med *Bilservice* och *Arbete, viss bil*. Den tjocka linjen står för kopplingen mellan en process och dess motsvarande Process Submodel. Notera att den översta submodellen inte har någon Process-koppling, vilket den heller inte behöver ha, enligt tidigare diskuterade förutsättningar.



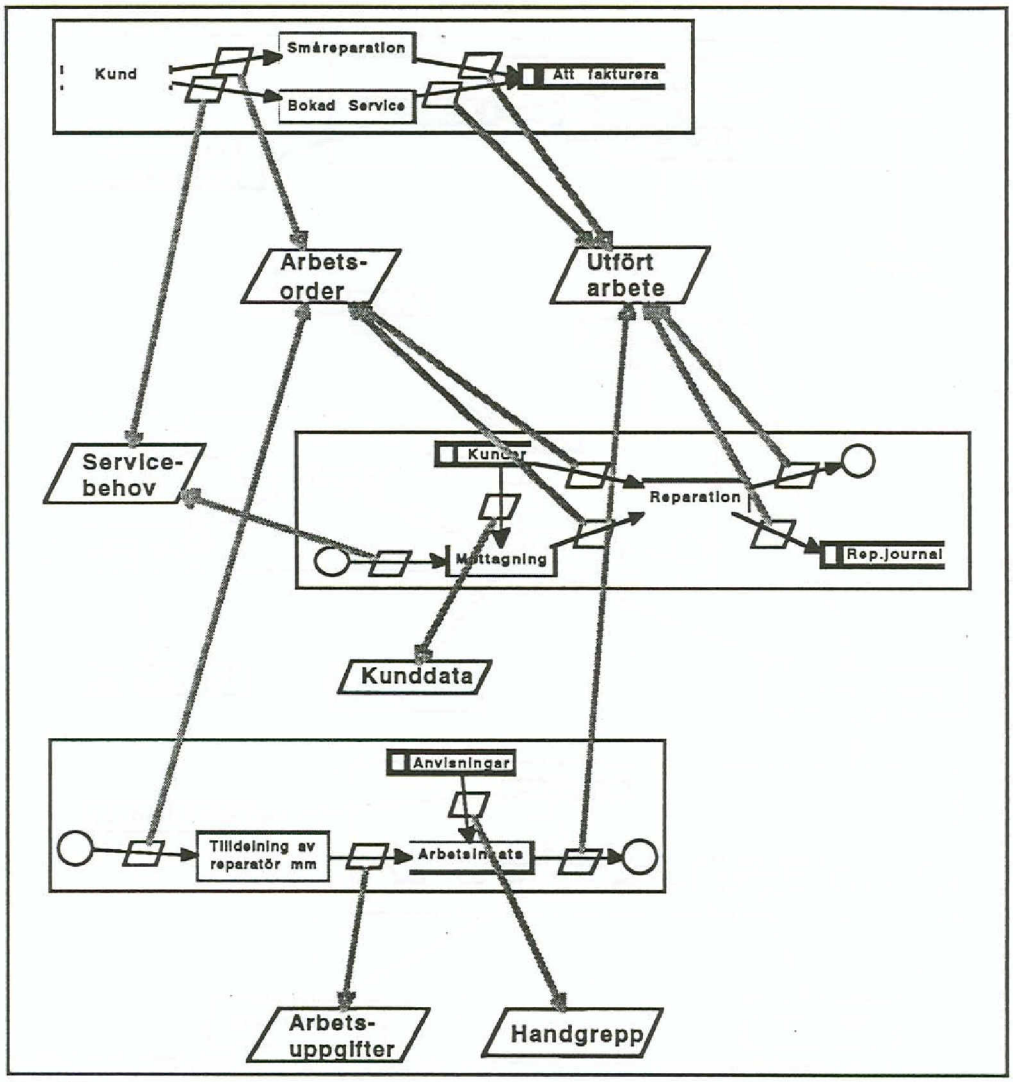
FIGUR 11

Flera Info Flow Vectors kan, som redan nämnts, referera till samma Info Flow. Av någon anledning tillåts dock inte lokala namn på en Info Flow Vector. Det går alltså inte att benämna två Info Flow Vectors för V1 och V2 och låta de båda peka på samma Info Flow I1. I en graf kommer dessa två Info Flow Vectors att på respektive pil ha namnet I1. En pil representerar på så vis en Info Flow Vector (var flödet sker) medan dess namn står för underliggande Info Flow (vad som flödar).

Det vore ju annars värdefullt att under en modellerings gång kunna använda submodell-lokala namn på Info Flows. Exv kunde input (Info Flow vector) till *Småreparation* vara *Begärt arbete*, motsvarande för *Reparation* vara *Arbetsorder* medan det i den lokala beskrivningen under *Arbete, viss bil* kunde

heta *Arbetsuppgifter*. I ett senare skede kanske konstateras att de står för samma Info Flow, som man i det läget väljer att benämna *Arbetsdata*. Ett utvidgat exempel i avsnitt 4 belyser problematiken.

Figur 12 nedan visar referenserna från samtliga Info Flow Vectors till respektive underliggande Info Flow.



FIGUR 12

4. Fler entity types

4.1 Process Port

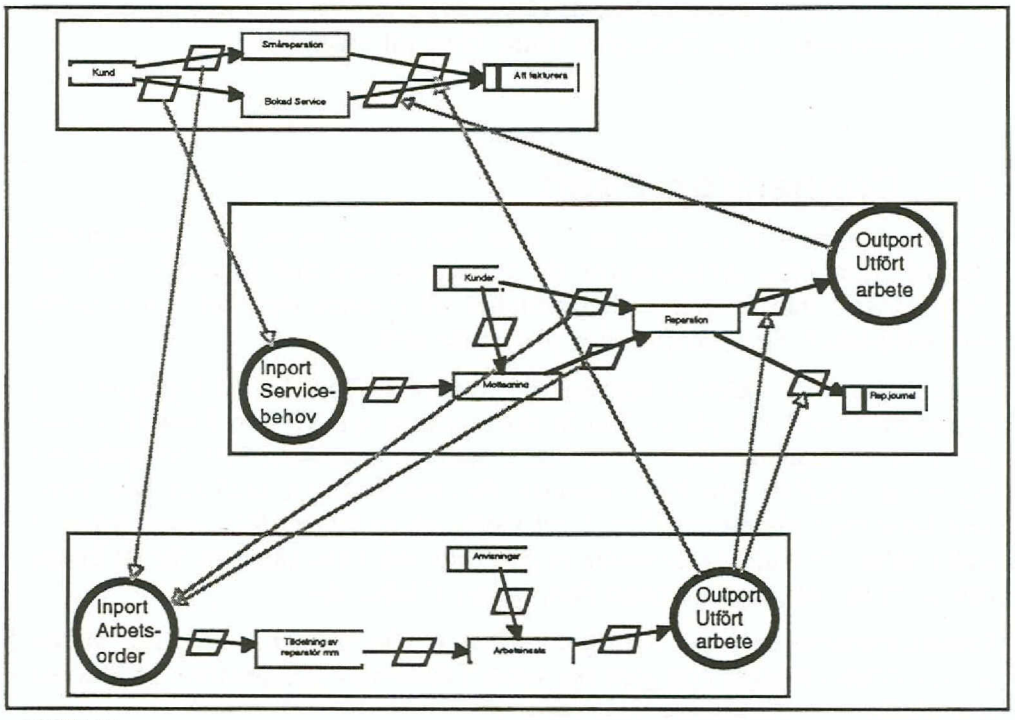
Låt oss nu fokusera intresset på förhållandet mellan Process Submodels på olika nivåer. Successiv precisering är i sig en naturlig aktivitet under en systemutveckling. Bekymret ligger i att hålla konsistens mellan beskrivningarna på de olika nivåerna. Exv måste ett Info Flow in till en process på nivå x också finnas representerat på processens mer detaljerade beskrivning på nivå x+1, osv.

Som komplicerande faktor i detta perspektiv vet vi redan att en och samma Process kan förekomma som komponent i flera separata Process Submodels. Kanske används olika namn för dem på respektive plats. Osv.

Det måste, med givna förutsättningar, gå att hålla reda på vilka delar på nivå x+1 som hör till vilka delar på nivå x. Rittekniskt kan olika tekniker tillämpas. Processmodellen måste på motsvarande sätt kunna tillhandahålla tillräcklig uppsättning beskrivningselement för en entydig definition.

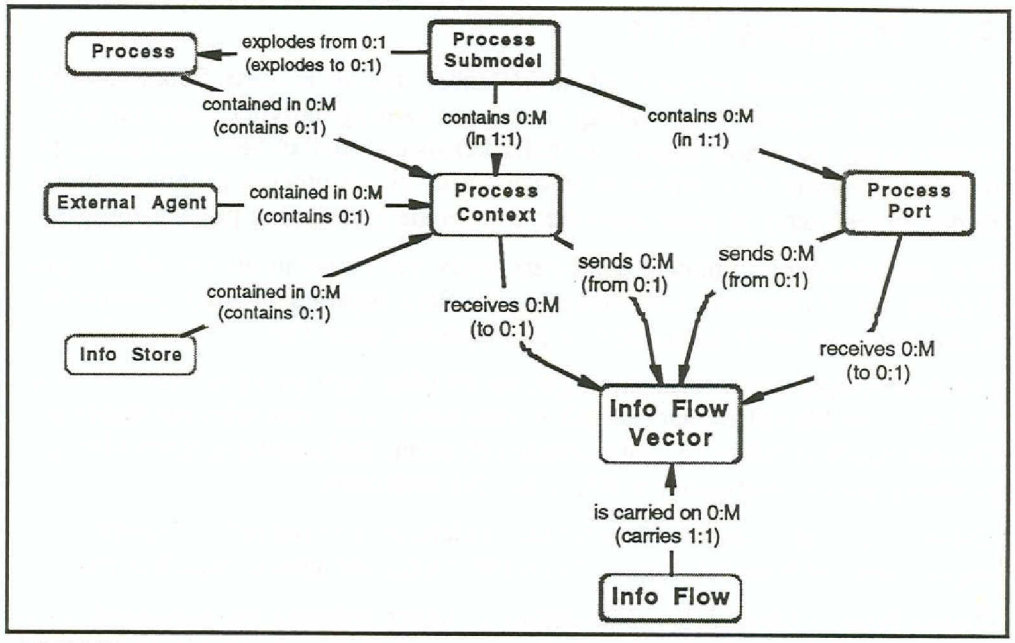
Gränsen mot omvärlden för Process Submodels, som beskriver en översta nivå, anges i form av External Agents. Även Info Stores för allmänt bruk hör dit.

Övriga Process submodels måste därutöver kunna redovisa sin kontaktyta mot de Process Submodels där den ingår som en komponent. Denna kontaktyta modelleras under begreppet **Process Port**. Exemplets Process Ports har vi redan smugit in i figur 12 i form av små cirklar. Figur 13 visar dem i förstorat skick.



FIGUR 13

En Process Port är en komponent i en Process Submodel på samma sätt som Process Contexts. Den kan både sända (exv *Inport Arbetsorder*) och ta emot (exv *Outport Utfört arbete*) Info Flows via Info Flow Vectors. Dess i övrigt något specifika roll har ansetts motivera den som en egen entity type i Processmodellen. Se kompletteringen i figur 14. Observera, att det nu är motiverat att ändra från 1:1 till 0:1 för relationship link "from" respektive "to".



FIGUR 14

Process Ports är en första förutsättning för att kunna länka ihop de olika nivåerna. Ännu klarar dock inte processmodellen av att beskriva de gråtonade pilarna, dvs kopplingarna mellan de olika nivåerna. För detta och för ett annat syfte behövs entity type Connection Point. Över till nästa avsnitt.

4.2 Connection Point

Processen *Arbete, viss bil* har en Process Port in och en ut. Den inkommande "för in" Info Flow *Arbetsorder* till Processen och den utgående "för ut" *Utfört arbete* från Processen. Det stämmer väl överens med hur den fungerar som komponent. Se *Småreparation* på översta nivå.

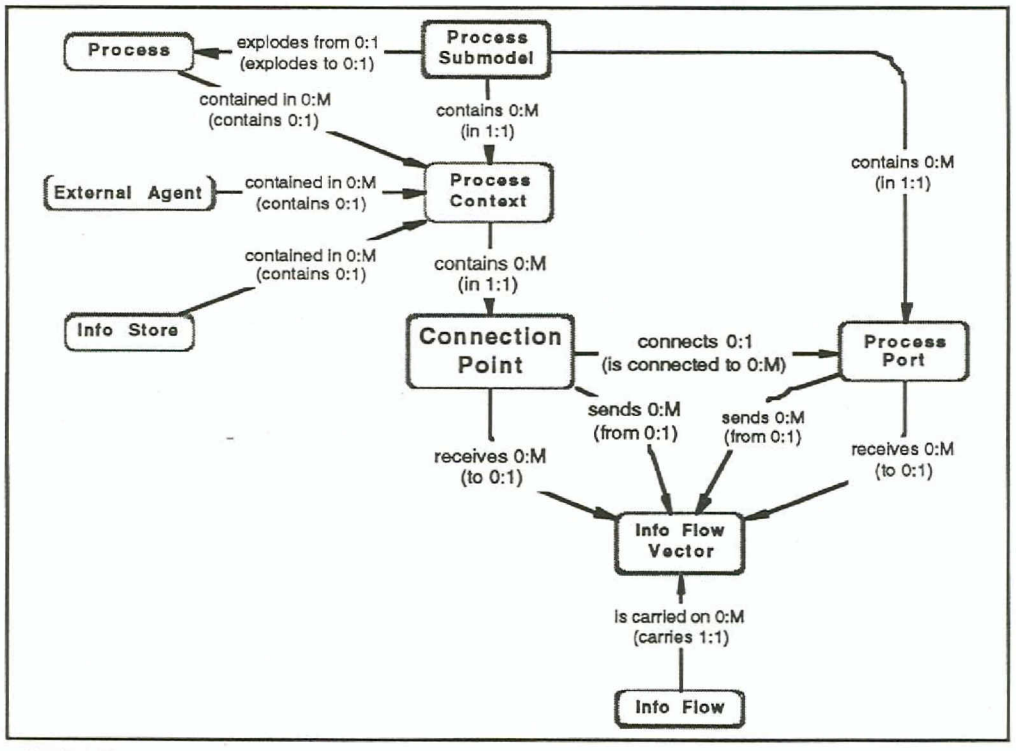
Inom *Bilservice* däremot kommer det två Info Flow in till Processen (se *Reparation*) och två går ut. Det är dock samma typ av Info Flow in i båda fallen in och samma Info Flow ut i båda fallen. För *Arbete, viss bil* spelar det ingen roll varifrån Info Flow kommer. Processen arbetar med det Info Flow som inkommer någonstans ifrån. I det vidare perspektivet (*Bilservice*) måste, å andra sidan, de båda *Arbetsorder* in på något sätt föras samman för att kunna länkas till en och samma Process Port, *Inport Arbetsorder*.

Samma resonemang gäller för *Utfört arbete* ut. Det lämnar *Arbete, viss bil* via en och samma Process Port men ska sedan i *Bilservice*-perspektivet spridas via två Info Flow Vectors.

För att avbilda förutsättningarna på en underliggande nivå kompletteras varje Process Context med en "kontaktpunkt" dit varje Info Flow Vector kopplas, istället för direkt till en Process Context. "Kontaktpunkten" blir i processmodellen en entity type **Connection Point**. Därmed skapas en möjlighet att sammanföra mer än en Info Flow Vector mot en och samma in- eller utgång. En förutsättning är givetvis att samtliga dessa Info Flow Vectors refererar till samma Info Flow, dvs till det Info Flow, som på den underliggande nivån redovisas till/från motsvarande Process Port.

I det generella fallet kan ju en Process ha flera Process Ports såväl in som ut. Det är därför viktigt att kunna definiera vilken Connection Point i en Process Submodel som svarar mot vilken Process Port, för att åstadkomma en fullständig beskrivning över hur de olika nivåbeskrivningarna hänger ihop. Observera att en Process Port står för en del av den "neutrala" processbeskrivningen (del av dess Process Submodel) medan en Connection Point svarar mot processens inlänkning i något sammanhang (som komponent i någon annan Process Submodel). Behovet tillgodoses genom relationship type "connects" mellan Connection Point och Process Port. Underligt nog måste inte Connection Points attribute type "name" anges. Anges det, finns ingen kontroll på att namnet är unikt inom given Process Context. Alltså kan inte namnet användas för att referera till viss Connection Point inom viss Process Context. Förväntas referensen alltid ske via en utpekad Info Flow Vector, eller?

Processmodellen byggs ut enligt figur 15.



FIGUR 15

5. Exemplets verksamhetsmodell

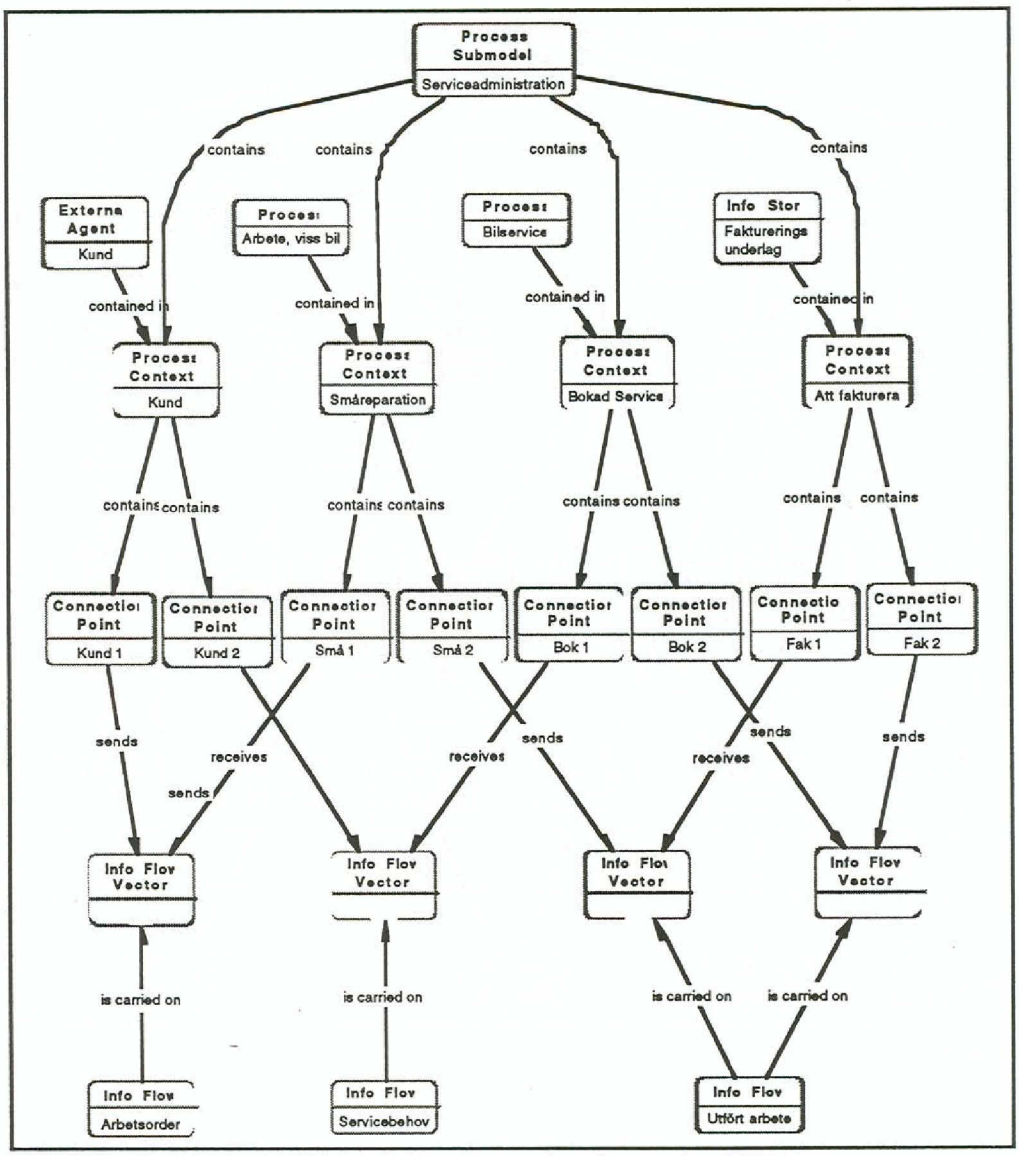
5.1 Inledning

Verksamhetsmodellen går inte att presentera i sin helhet på en A4-sida. Vi delar upp den i ett antal delmodeller enligt följande:

* Process submodel: *Serviceadministration*, figur 16.
* Process submodel: *Bilservice*, figur 17.
* Process submodel: *Arbete, viss bil*, figur 18.
* Processsammankopplingen, figur 19.
* Sammankopplingen mellan flöden på olika nivåer, figur 21.

5.2 Process Submodel: *Serviceadministration*

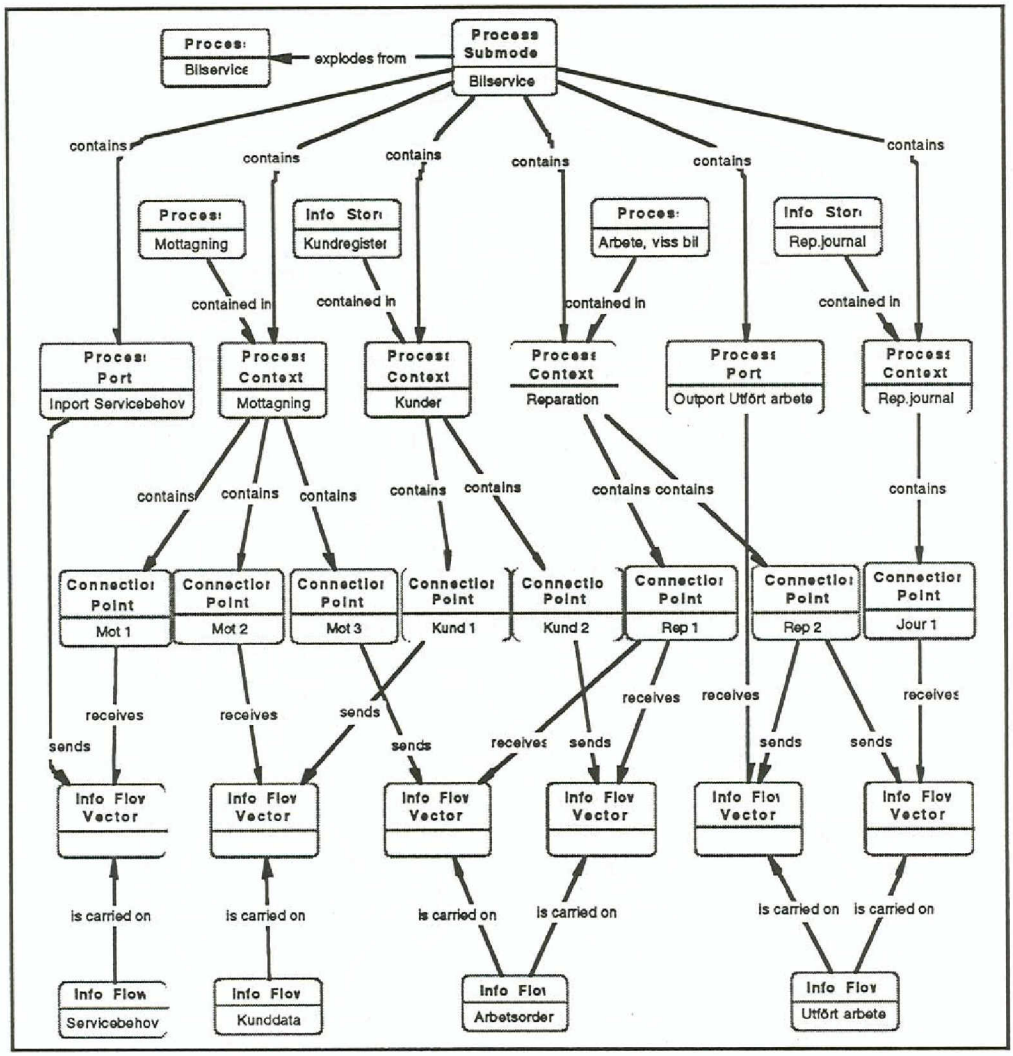
Den del av verksamhetsmodellen, som berör *Serviceadministration* "lokalt", visas i figur 16.



FIGUR 16

5.3 Process Submodel: *Bilservice*

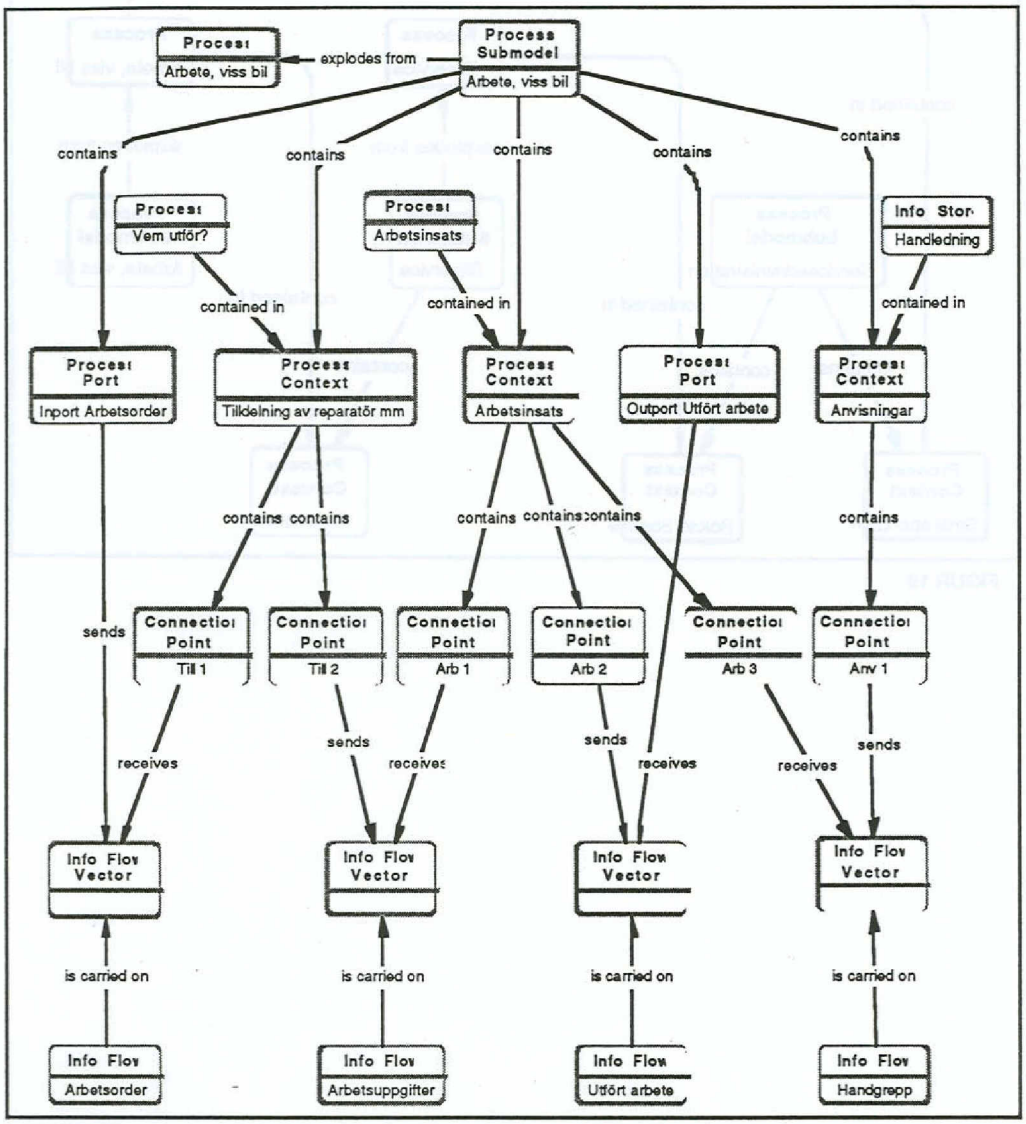
Den del av verksamhetsmodellen, som berör *Bilservice* "lokalt", visas i figur 17.



FIGUR 17

5.4 Process Submodel: *Arbete, viss bil*

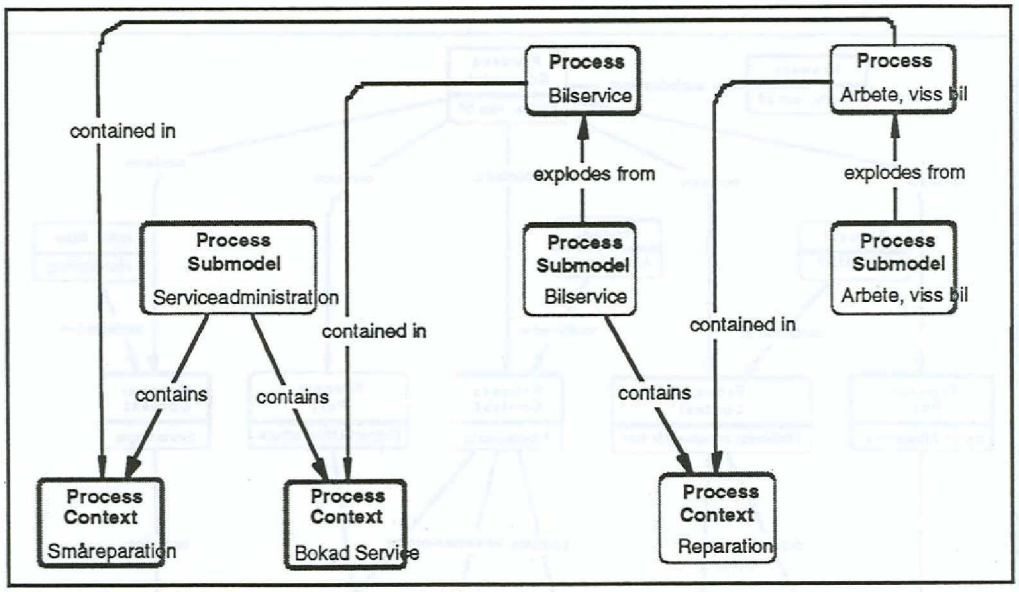
Den del av verksamhetsmodellen, som berör *Arbete, viss bil* "lokalt", visas i figur 18.



FIGUR 18

5.5 Processsammankopplingen

Processernas referenser till varandra finns med i figurerna 16 - 18. Figur 19 visar dessa kopplingar i en och samma graf.

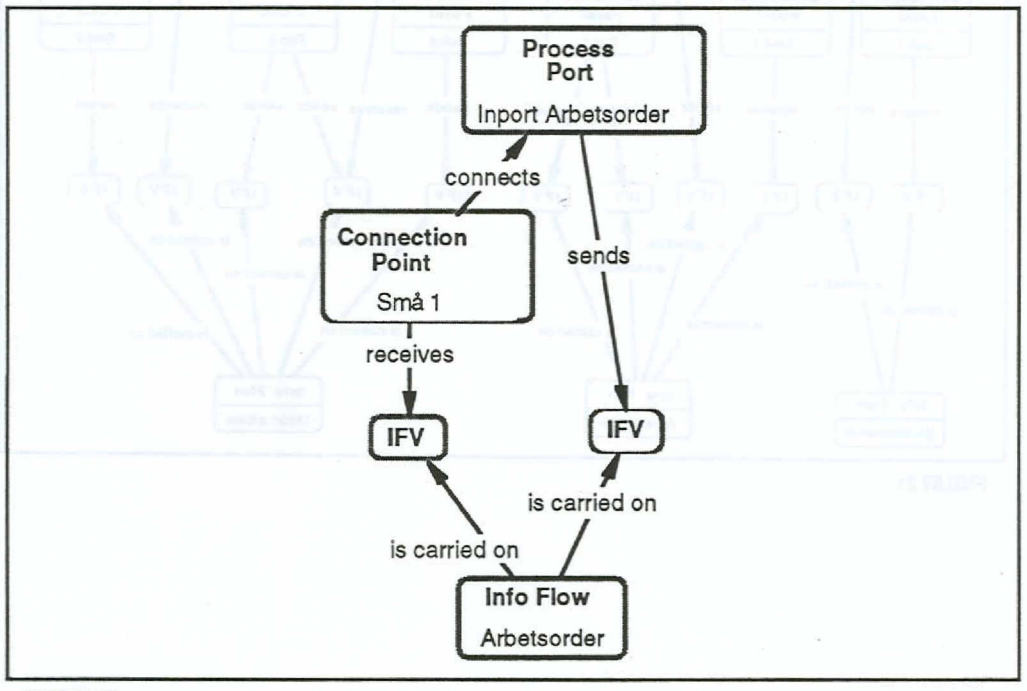


FIGUR 19

5.6 Sammankopplingen mellan flöden på olika nivåer

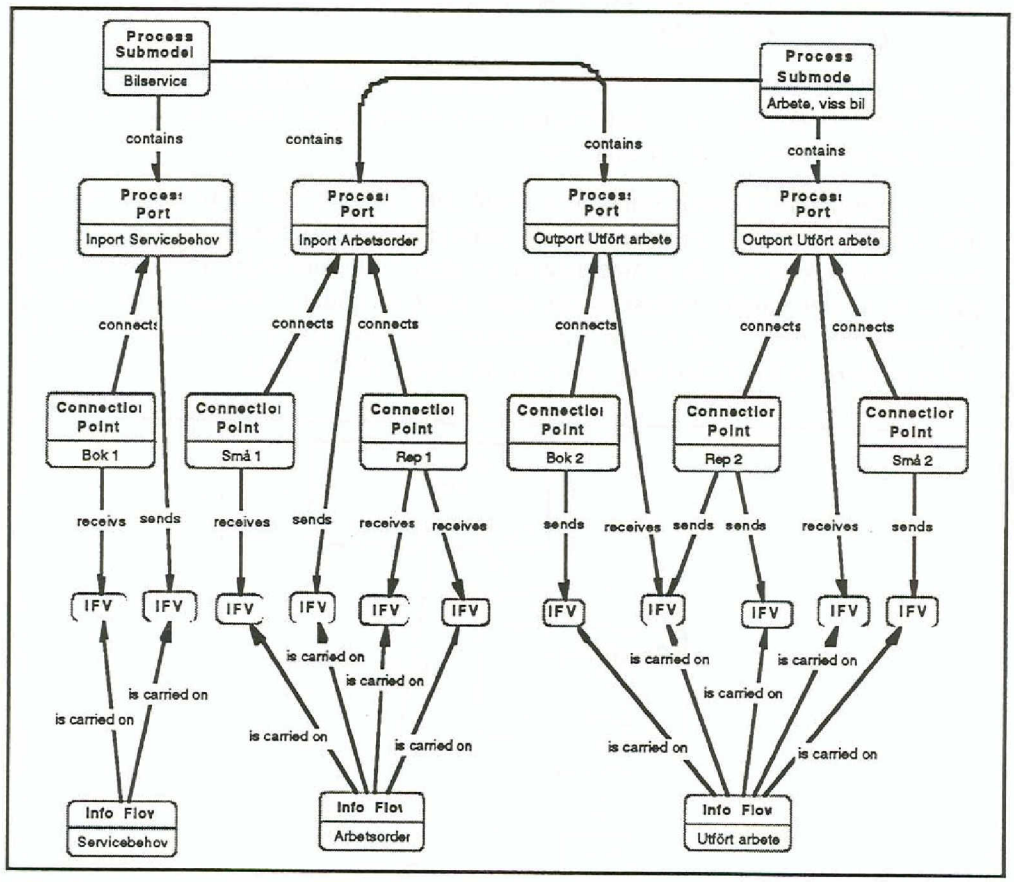
De Info Flows som går in mot eller ut från en Process, som har en egen Process Submodel, beskrivs med den Connection Point, som refererar till Process Submodels motsvarande Process Port.

Ta exv *Småreparations* Connection Point *Små 1*, som "tar emot" *Arbetsorder* från *Kund*. Eftersom *Småreparation* refererar till Processen *Arbete, viss bil* måste *Små 1* kopplas ihop med någon av dess "inkommande" Process Ports. I vårt fall är det enkelt, eftersom det bara finns en sådan - Process Port *Inport Arbetsorder*. Den, i sin tur, är "avsändare" för den Info Flow Vector som för *Arbetsordern* in till *Tilldelning av reparatör* mm. Se figur 20.



FIGUR 20

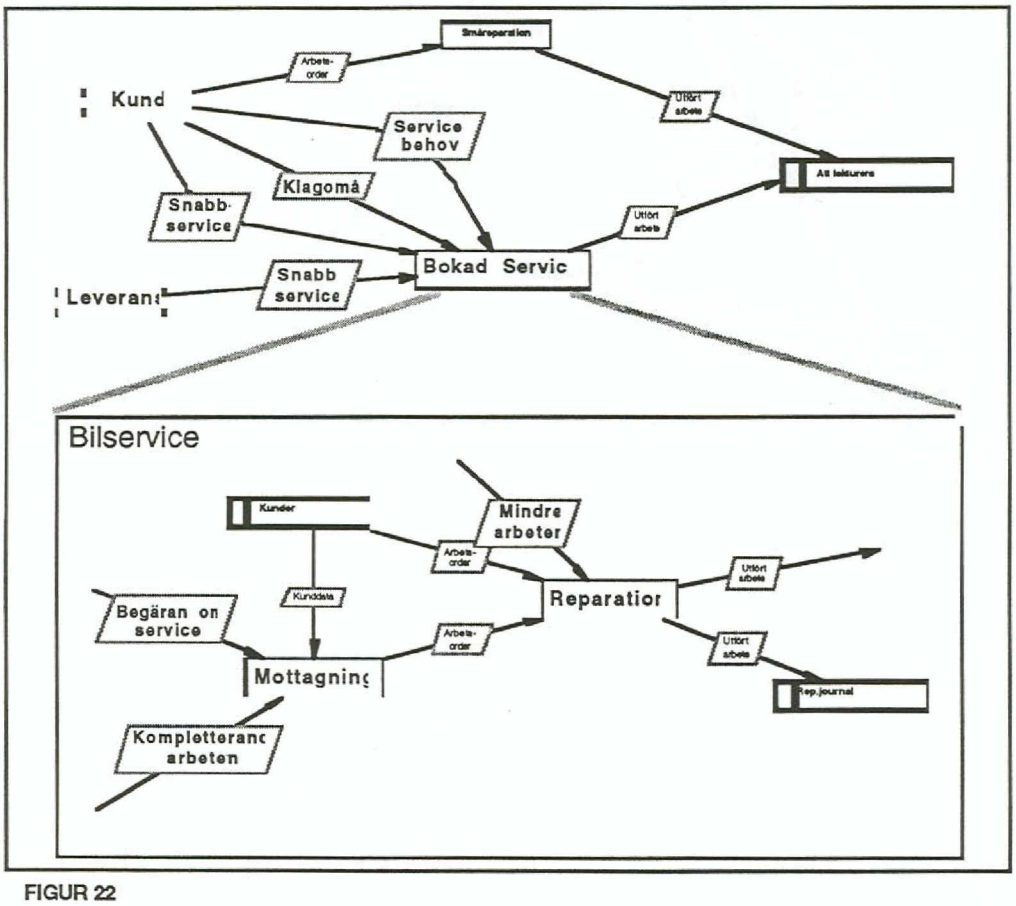
Den fulla kopplingsbilden visas i figur 21. Av utrymmesskäl har Info Flow Vector förkortats IFV.



FIGUR 21

6. Process Ports och Connection Points igen

Behovet av Process Ports och Connection Points belyses bättre genom ett avslutande, något utvidgat exempel baserat på de två översta processnivåerna Serviceadministration och Bilservice. Vi är endast intresserade av att beskriva infoflödena in till Bokad Service och deras motsvarighet på nästa nivån Bilservice.

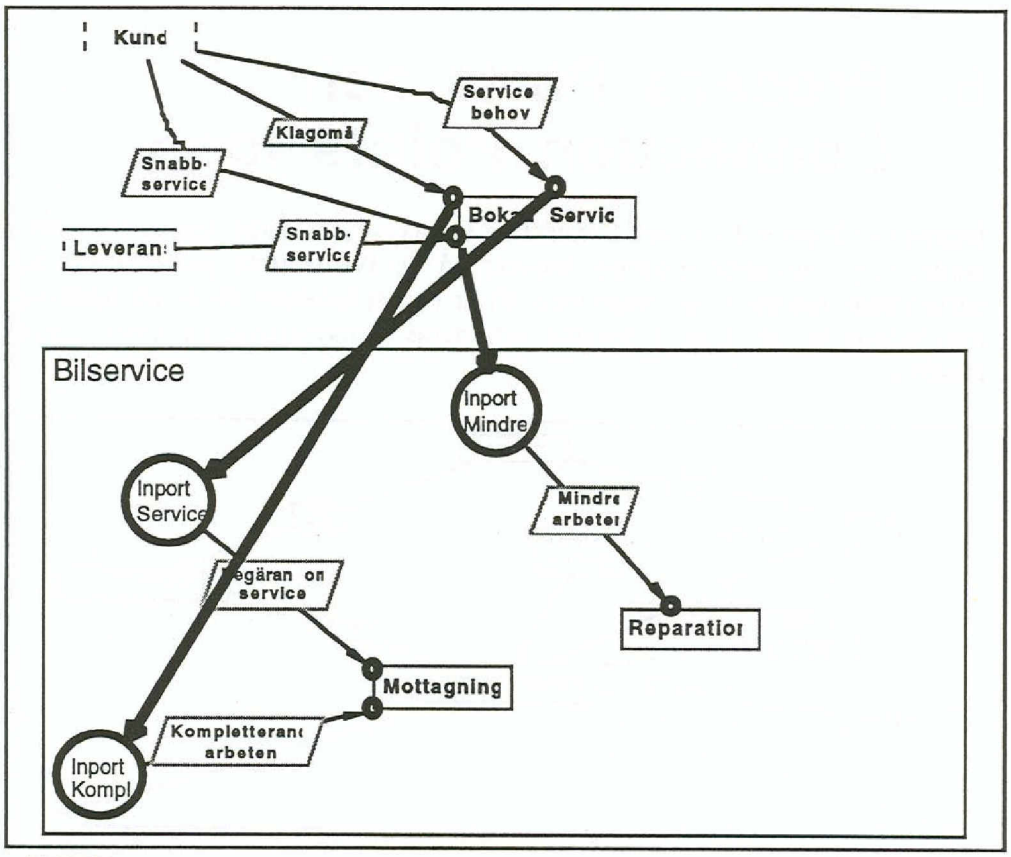


FIGUR 22

Bilden är inte tillräckligt exakt i sin formulering av vad på *Serviceadministrations*nivå som svarar mot vad på *Bilservice*nivå. Svarar *Klagomål* mot *Mindre arbeten* eller mot *Kompletterande arbeten* eller kanske (mer semantiskt långsökt) mot *Begäran om service*?

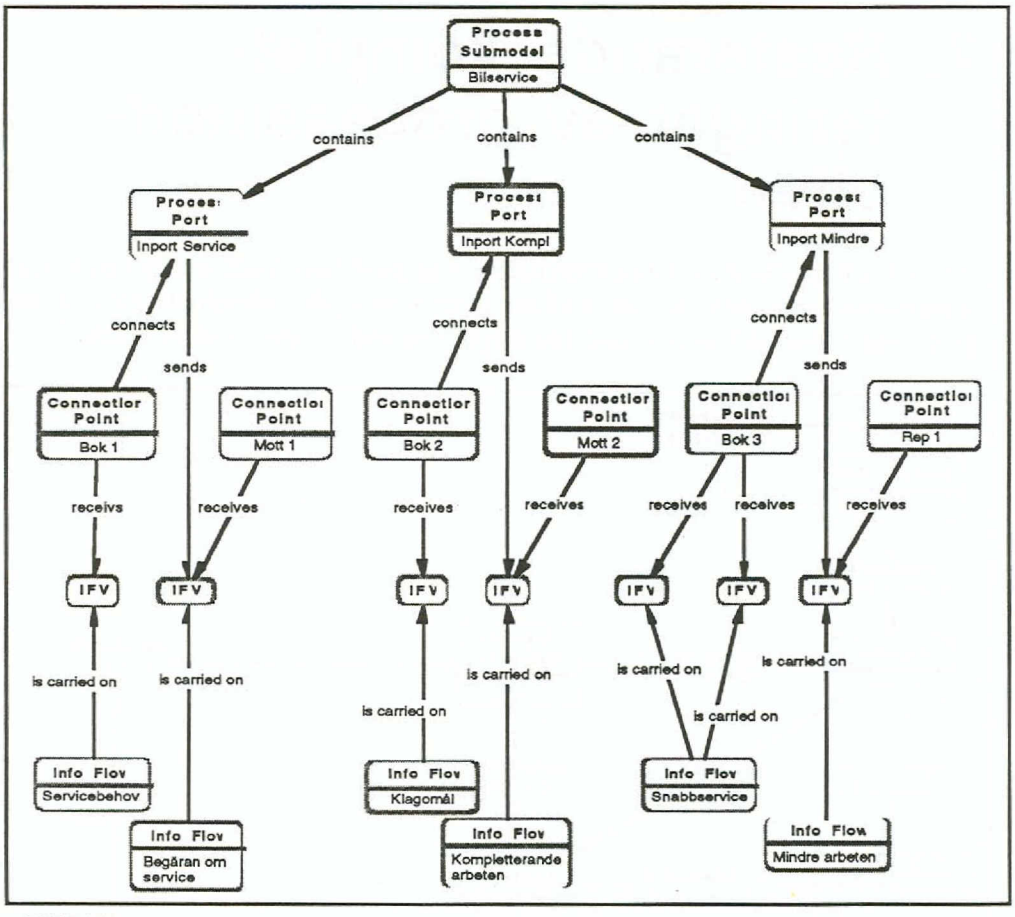
Man kunde invända att eftersom inte Info Flow Vectors har egna lokala namn, borde samma namn anges på en Info Flow Vector på övre nivån som på dess motsvarighet på nedra nivån. Några sådana krav stipulerar inte IM, antagligen för att ge frihet under arbetsprocessen. Sådana namnskillnader måste dock vara utredda innan modellen kan anses klar.

Begreppsmodellen (se figur 15) måste kunna klara av att beskriva motsvarigheten till det som visas i figur 23.



FIGUR 23

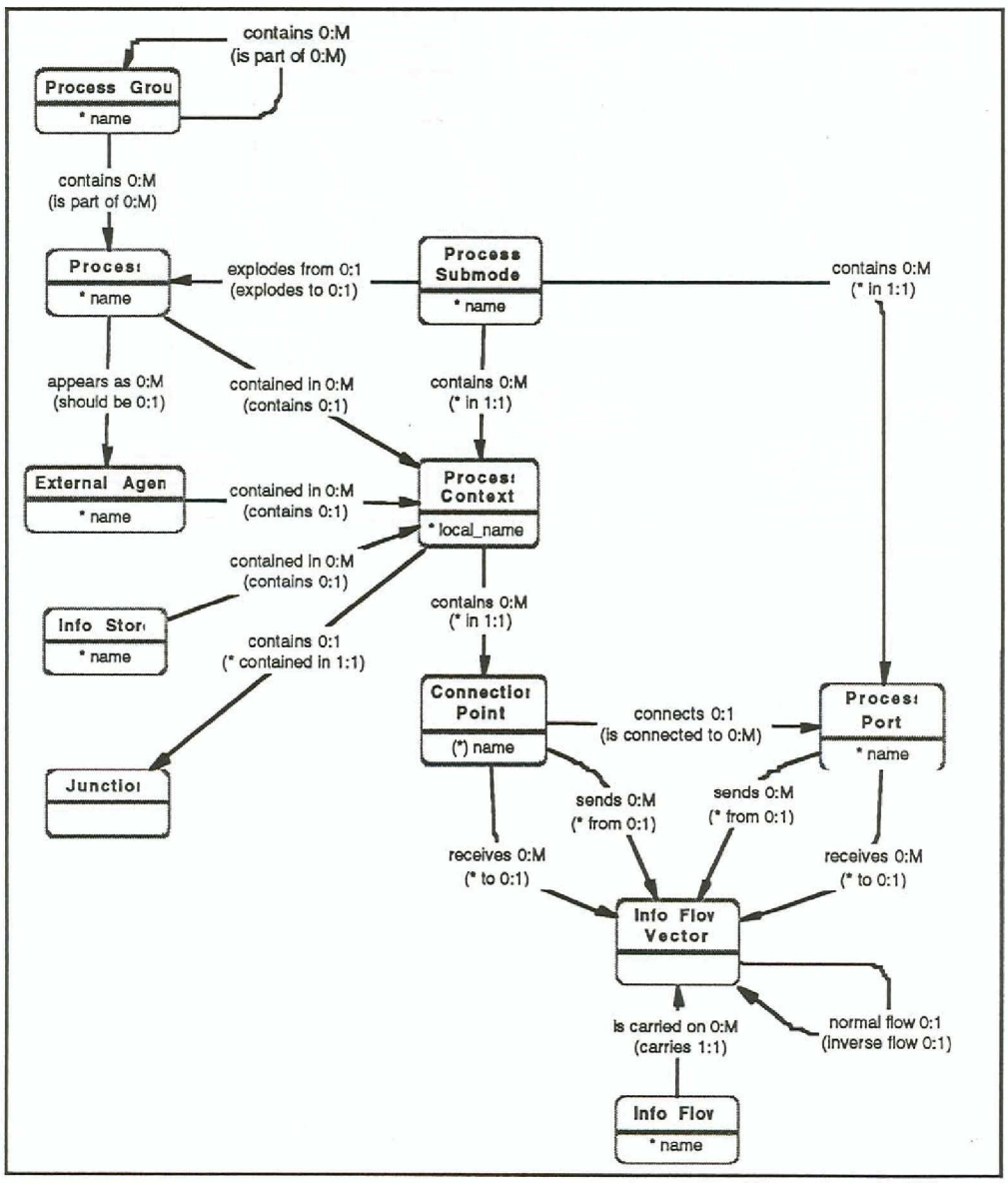
Begreppsmodellen klarar detta utmärkt. Se resulterande verksamhetsmodell i figur 24.



FIGUR 24

7. Resterande kompletteringar av processmodellen

Till sist kompletterar vi Processmodellen, dels med resterande entity types och relationship types, dels med de attribute types som ingår i identifieringar, för att ge en snabböverblick över rollen hos respektive entity type inom helheten.



FIGUR 25

Entity type **Junction** är egentligen en subtyp av Process Context (skulle kunna ha modellerats mha generaliseringssamband). Den kan ses som en enkel, speciell typ av process inom viss Process Submodel vars enda roll är att antingen föra ihop, dela upp eller omvandla Info Flows. Samma funktionalitet skulle kunna modelleras som en "vanlig" Process inplacerad som en Process Context inom aktuell Process Submodel. Antagligen är Junction en eftergift till hur vissa Case-verktyg modellerar liknande situationer, snarare än ett direkt behov.

Entity type **Process Group** har tillkommit i Version 1,Release 2 av modellen. Syftet är att fungera som en logisk sammanbindningspunkt för ett antal Processer som av någon anledning, för något behov, anses höra ihop. Process Group verkar bla vara en naturlig referenspunkt mellan processmodellen och andra submodeller inom Enterprise Submodel. (Ett antal relationship types mellan m odellerna ger belägg för detta.) Genom att Process Groups får inordnas under Process Groups kan valfritt djupa och komplexa grupperingar upprättas.

Relationship type "appears as" mellan Process och External Agent används för att notera att en Process i ett data flow diagram mycket väl kan figurera som External Agent under ett annat data flow diagram. Eftersom namngivningen inte med säkerhet överensstämmer, behövs denna relationship type för exakt referens mellan dem. Observera följande restriktion. I den mån en process uppträder flera gånger som komponent (viss Process Context) inom visst data flow diagram, sker det konsekvent antingen i rollen som Process eller som External Agent, aldrig i båda.

Behovet av relationship type "normal flow" är oklart.

8. Identifiering av entity types i processmodellen

När det gäller identifieringen i figur 25 ska följande noteras:

* IM använder konsekvent interna identifierare (Syskeys) för identifiering. Här angiven identifiering ska uppfattas mer som en specifikation över krav för unik referens till entity av viss typ.
* Vissa entity types har ett självständigt existensberättigande inom given Info Submodel. De identifieras därför genom ett entity type unikt namn. Dessa är Process, Process Submodel, External Agent, Info Store, Info Flow.
* En Process Context tillhör alltid viss Process Submodel och har därför ett namn som endast behöver vara unikt inom "sin" Process Submodel. Samma sak gäller Process Port. Underligt nog benämns inte attributtyperna för de lokala namnen på samma sätt (local_name resp. name).
* Junction behöver inget eget namn eftersom det står i ett 1:1-förhållande till Process Context.
* En Connection Point hör alltid till viss Process Context och behöver av den anledningen endast en tillkommande identifiering som gör den unik inom viss Process Context. Underligt nog har man valt att göra det lokala namnet 'optional' vilket rimligtvis borde försvåra referens till viss unik Connection Point. Obs, att det inte är några bekymmer med identifieringen i sig pga användningen av SYSKEY.
* Vi har redan tidigare noterat avsaknaden av 'lokalt' namn för Info Flow Vector. Därutöver gäller, vilket grafen inte kan uttrycka, att kontaktpunkt för Info Flow Vector antingen är Connection Point eller Process Port. Endast en "from" och en "to" behövs för identifiering. Hade Process Port modellerats som en specialisering av Process Context eller de gemensamma egenskaperna hos Connection Point och Process Port modellerats som en generaliserad entity type skulle detta problem inte behövt uppstå.

9. Till sist

Processmodellen tycks ha nått en god stabilitet. Endast smärre justeringar har gjorts mellan de senaste releaserna. Modellen klarar att uttrycka både komplicerade processbeskrivningar och obegränsat antal beskrivningsnivåer.

Varje Info Flow har en intern struktur där varje del av strukturen har en semantik som uttrycks med hjälp av referens till begreppsmodellen. Vi kallar denna delmodell av IM för Informationsflödesmodellen. Den beskrivs i en separat rapport, TRIAD K13.

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.