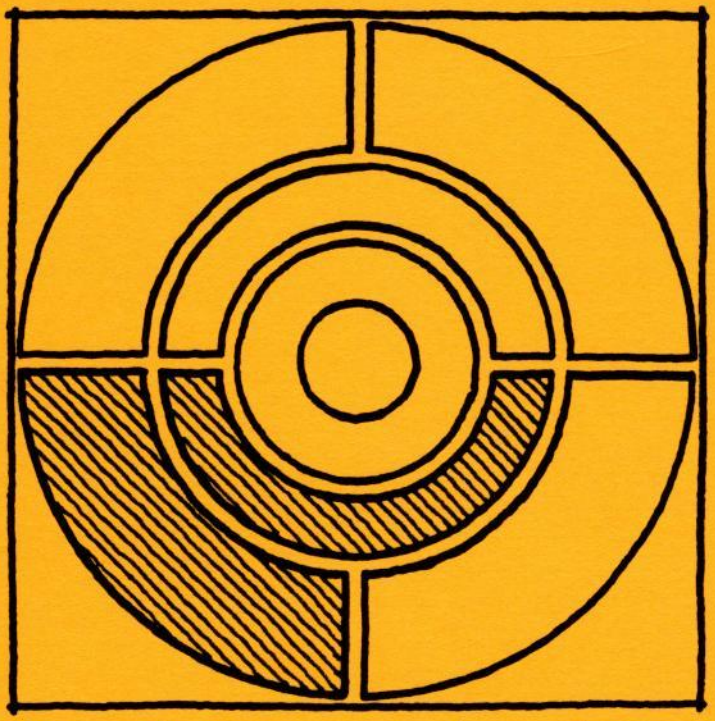
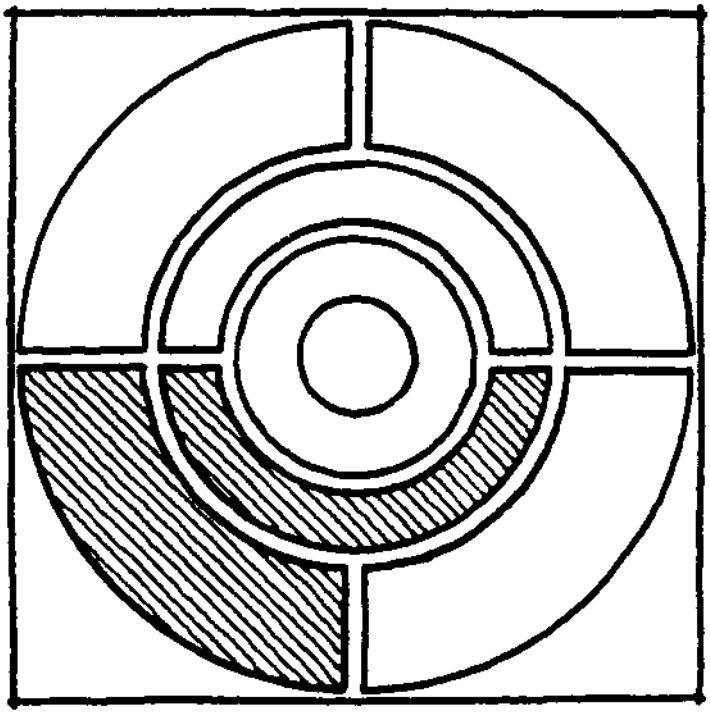


Struktureret Analyse



Struktureret Analyse

Struktureret Analyse

Version 1, december 1991

Ansvarlig:
Systemudviklingsservice

Fordelingsnøgle: **HSA1**

Indholdsfortegnelse

Indledning og formål

Denne håndbog er en vejledning i anvendelse af Struktureret Analyse på Datacentralen.

Formålet med håndbogen er at beskrive metodens indhold og anvendelsesområde, og derigennem at kunne fungere som opslagsbog og huskeliste.

Håndbogen kan derimod ikke erstatte uddannelse og praktisk træning i, hvordan man gennemfører analysen og det overordnede design. Den nødvendige uddannelse kan fx erhverves på Datacentralens praktikkursus i Struktureret Analyse.

Håndbogen er justeret efter de krav, som indgår i Datacentralens kvalitetsstyringssystemer i forbindelse med gennemførelse af analyse og design.

Grundlæggende begreber og definitioner

Dette afsnit redegør for idegrundlaget bag Struktureret Analyse. Metoden placeres i forhold til det samlede udviklingsforløb og til andre systemudviklingsmetoder. Endelig defineres en række af de ord og begreber, som anvendes i Struktureret Analyse.

Idegrundlag

Grundtanken i Struktureret Analyse (SA) er, at man arbejder med fysiske og logiske *modeller* af systemet med det formål, at opnå et velstruktureret, rettelsesvenligt design med mindst mulig redundans (gentagelse) og indbyrdes afhængighed mellem systemets funktioner.

SA benytter datastrømsdiagrammer - en kombination af grafik og tekst - til at anskueliggøre de overordnede niveauer af beskrivelsen. Datastrømsdiagrammerne udarbejdes i niveauer, hvor antallet af niveauer afhænger af systemets omfang og kompleksitet.

Dette, at læseren kan vælge et udsnit af systemet med den detaljeringsgrad, han selv ønsker, er med til at give et bedre overblik og en lettere indføring i systemet i forhold til traditionel verbal fremstilling.

Hertil kommer en række fordele, som viser sig i selve udviklingsprocessen. Som eksempel kan nævnes diagrammernes fleksibilitet, rettelsesvenlighed og enkle syntaks, som understøtter en naturlig, iterativ arbejdsproces med stort brugerengagement.

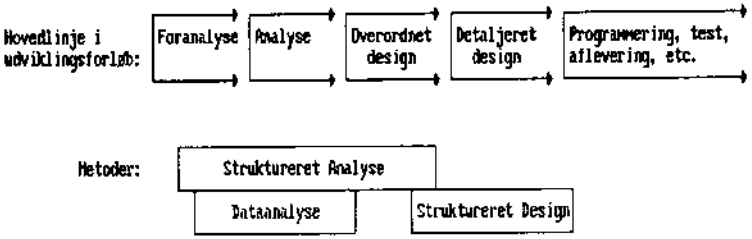
Placering i udviklingsforløbet

Metoden Struktureret Analyse benyttes i foranalysen som beskrivelsesværktøj til funktionsanalyse af et eksisterende system eller forretningsgang.

SA bruges endvidere ved detailanalyse og overordnet design af funktionerne i det fremtidige system.

Samspil med andre metoder

Resultatet af det overordnede design (=den ny fysiske model) danner grundlag for det detaljerede design, som benytter metoden Struktureret Design. En ny fysisk model, som er struktureret efter de retningslinjer, som angives i denne håndbog, er et velegnet udgangspunkt for det detaljerede design.



Dataanalyse kan udføres før, eller parallelt med, SA. Det seneste tidspunkt for afslutningen af dataanalysen fremgår af arbejdsmetodikken for den nuværende logiske model, idet resultatet af dataanalysen indgår i denne.

Referencer

Denne håndbog fungerer i samspil med følgende andre brochurer som beskriver tilgrænsede metoder:

Dataanalyse	(HDA)
Struktureret Analyse	(HSA)
Dialognotation	(HDN)

Ordforklaringer

I dette afsnit defineres de begreber, som indgår i metoden

SA. Ordforklaringerne er ordnet systematisk, således at beslægtede begreber så vidt muligt defineres i sammenhæng. Visse begreber, fx entitet og relation, er ikke defineret her, da de behandles under Dataanalyse. Der henvises til brochuren herom.

Overordnede begreber, der ikke er knyttet til en bestemt metode, er defineret i "Ordbog for systemudvikling".

Analysemodeller og analysearbejde

Model

Grafisk, verbal og/eller konkret fremstilling af virkelige eller tænkte fænomener, systemer eller genstande. Afhængig af modellens formål vil den fremhæve visse aspekter af forlægget, og undertrykke eller ignorere resten.

Fysisk model

Model af den fysiske implementering af et eksisterende eller tænkt system.

Logisk model

Model af de basale eller essentielle dele af systemet. Modellen beskriver den rene transformation af inddata til uddata uden tekniske begrænsninger, redundans, tidsforløb, datamedier, eller andre henvisninger til en bestemt fysisk implementering.

Funktion

En arbejdsrutine eller forretningsgang i det system, der beskrives. Svarer til en proces i analysemodellerne.

Implementering

Virkeliggørelse af en logisk model. Omfatter alle aktiviteter fra overordnet design frem til ibrugtagningen af systemet. NB: I andre systemudviklingsmetoder kan der forekomme mere snævre definitioner af implementering.

Redundans

Forekomst af identiske funktioner, registre eller data flere steder i et system eller en model. Redundans kan være symptom på dårligt eller ugennemtænkt design, men kan også indføres bevidst som led i den fysiske implementering og optimering af systemet.

CASE-værktøj

Integreret - oftest pc-baseret - programmel, som understøtter systemudviklingen. Den del, som er rettet mod SA, indeholder typisk tegnefunktioner, teksteditor, rapportgenerator, samt validerings- og balanceringsfunktioner. (eks.: DesignAid)

Contextdiagrammet

Contextdiagram

Datastrømsdiagram, der viser systemets afgrænsning og dets interaktion med omverdenen. Hele systemet vises som en enkelt proces, og samtlige ind- og udgående datastrømme forbindes til deres leverandører/modtagere.

Leverandør/modtager

Person, institution, system, database eller lignende, som leverer data til eller modtager data fra systemet.

Øvrige datastrømsdiagrammer

Datastrømsdiagram *(forkortes DFD: Data Flow Diagram)*

Diagram, der på et givet niveau viser systemets processer og de datastrømme og registre, processerne benytter.

Figur 0

Diagram, der viser det øverste niveau i analyseområdet. Processerne på dette DFD, som viser systemets overordnede opdeling i funktioner eller delsystemer, er kendetegnet ved et 1-cifret referencenummer. Hvis analyseområdet er en del af et større systemkompleks, behøver der ikke at findes en figur 0 for hele komplekset. I stedet kan man udarbejde en oversigt (liste) over delsystemerne.

Niveaudeling

Af hensyn til overskueligheden kan DFD tegnes i flere niveauer. Hver overordnet proces nedbrydes i underordnede DFD'er indtil man når en detaljeringsgrad, hvor behandlingen mest hensigtsmæssigt kan beskrives i en procesbeskrivelse.

Datastrøm

Viser en overførsel (transport) af data til og fra processer og registre. Pilespidsen angiver datastrømmens retning. En datastrøm kan bestå af et enkelt dataelement, af andre datastrømme, eller en kombination heraf.

Register

Opbevaringssted for data, som skal kunne bruges flere gange eller på et senere tidspunkt.

Logisk register

Samling af data om et bestemt begreb, fx kunder, varer, fakturaer, etc. I en logisk model kan alle data i det logiske register læses umiddelbart, uden hensyn til sekvens, accesstid, søgekriterium, etc.

Fysisk register

En bestemt implementering af et logisk register, fx tabel, diskette, samlemappe, sekventiel fil, osv.

Processer

Proces

Fremgangsmåde for behandling af data, foretaget af et manuelt eller maskinelt system. I SA opdeler man processer efter deres formål, i fysiske og logiske processer.

Fysisk proces

Proces, som udelukkende udfører en eller flere af følgende former for behandling:

- Kontrol af validitet eller fuldstændighed.
- Konvertering fra en repræsentationsform, et format eller medium til et andet.
- Transport af data, også når der indgår valg af transportvej på grundlag af datastrømmens indhold eller status.
- Styring eller igangsætning af andre processer, fx operativsystem, tele- og databasestyring, kørselsafvikling, mv. Kaldes også administrativ proces.

Logisk proces

Proces, som skaber data, der ikke fandtes i forvejen, på grundlag af beregninger, beslutninger eller komplicerede søgninger. Man kan opdele de logiske processer i fundamentale processer og vedligeholdelsesprocesser.

Fundamental proces

Proces, der udfører fundamentale aktiviteter, dvs. aktiviteter, der medvirker til opfyldelse af systemets overordnede formål.

Vedligeholdelsesproces

Proces, der vedligeholder (opretter, ændrer og sletter) lagrede data i systemet.

Administrativ proces

Proces, der varetager administrationen af systemet, dvs. funktioner omkring sikkerhed, back-up, kørselsafvikling, styring af terminaler og databaseadgang, etc. Forekommer kun i fysiske modeller.

Essentiel proces

Synonym for logisk proces. Begrebet er introduceret af McMenamin og Palmer, og benyttes i flere lærebøger. Tilsvarende kan man tale om essentielle aktiviteter, data og registre.

Procesbeskrivelse

Verbal og/eller grafisk beskrivelse af en proces og de regler, der styrer dens forløb.

Dataordbogen

Dataordbog

Oversigt, der definerer systemets dataelementer, datastrømme og registre.

Dataelement

Den mindste enhed af data, som er beskrevet i dataordbogen. Indgår i en eller flere datastrømme og/eller registre.

Hændelsesmodeller

Hændelse

En ændring af tilstand eller status, som systemet er indrettet til at reagere på med en respons.

Ekstern hændelse

En hændelse i systemets omverden som bevirker, at der sendes en datastrøm til systemet. Kaldes derfor også datahændelse.

Intern hændelse

En hændelse som opstår i systemet, hvor det medfører en behandling af lagrede data. Hændelsen er ofte et forudbestemt tidspunkt der indtræffer, hvorfor interne hændelser også kal-

des tidshændelser. Men det kan også være et resultat af et antal eksterne hændelser, fx udskrivning af et kontoudtog efter et vist antal bevægelser på kontoen.

Respons

Systemets planlagte reaktion på en hændelse. Kan bestå i en eller flere datastrømme som forlader systemet, opdatering af et eller flere registre, eller begge dele.

Ad hoc-respons

Systemets reaktion på en ikke-planlagt impuls fra omverdenen, fx en strømafbrydelse, eller en fejl som opstår i systemet, fx læse/skrivefejl på et lagringsmedie.

Denne type fejl er i sagens natur irrelevant i de logiske modeller. Først ved design af det ny fysiske system behøver man derfor at tage højde for disse situationer.

Notation og symboler

En model i SA består af følgende elementer:

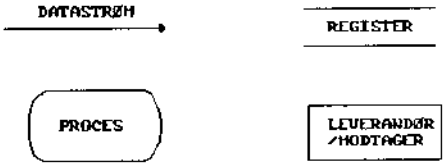
- datastrømsdiagrammer
- dataordbog
- procesbeskrivelser.

I dette afsnit beskrives notation og regler for hvert af disse elementer. Sammenhængen og samspillet mellem elementerne beskrives i afsnittet om regler og konventioner.

Datastrømsdiagram (DFD)

Symboler

Et DFD må kun indeholde de viste symboler, samt navne og andre referencer til det system, man beskriver.



Symbolet for leverandør/modtager må normalt kun fore-

komme på contextdiagrammet, men det kan i analysearbejdets indledende fase være nyttigt at tegne det med på de øvrige diagramskitser for at støtte hukommelsen.

Navngivning

Alle processer, registre og datastrømme skal være forsynet med et navn, som klart angiver formål og indhold.

Proces:

For processer på nederste niveau i modellen skal navnet beskrive den funktion, processen udfører. Beskrivelsen skal være kort, og så vidt muligt formuleret som en aktiv handling. For nogle processer falder det naturligt at give dem navn efter det uddata, de danner (eks.: Udsend rykker), i andre tilfælde vil man fokusere på den behandling, som processen udfører på sit inddata (eks.: Kontroller ordre).

Navnene på højere niveauer skal så vidt mulig være dækkende for de funktioner, de underliggende processer udfører. Dette er med til at sikre en hensigtsmæssig gruppering af processer, men det kan ofte være vanskeligt at finde præcise navne.

Datastrøm:

Datastrømme skal navngives præcist, entydigt og forståeligt. Til systemets ind- og uddata bør man bevare de navne, som

benyttes af brugerne, såfremt navnene opfylder ovenstående krav. Hvis dette ikke er muligt, skal der være henvisninger i dataordbogen mellem brugernes betegnelse og den nye.

Datastrømme, som går til eller fra registre, behøver ikke at navngives, men omfatter da alle data i en forekomst på registret. Hvis en proces benytter ganske bestemte data fra et register, bør man som hovedregel vise dette ved at navngive datastrømmen.

For at bevare overblikket på DFD'erne, er det ofte nødvendigt at arbejde med flere niveauer (detaljeringsgrader) af datastrømme. Hvert niveau skal defineres i dataordbogen under et sigende navn. Data, som vanskeligt kan samles under et fælles navn, bør holdes adskilt.

Register:

Registre navngives efter det begreb, registerets indhold er knyttet til. I de fysiske modeller kan navnet afspejle den eksisterende eller valgte implementering af systemet ved at angive det fysiske medie, fx kartotek, mappe, bunke, diskette, etc.

Leverandør/modtager:

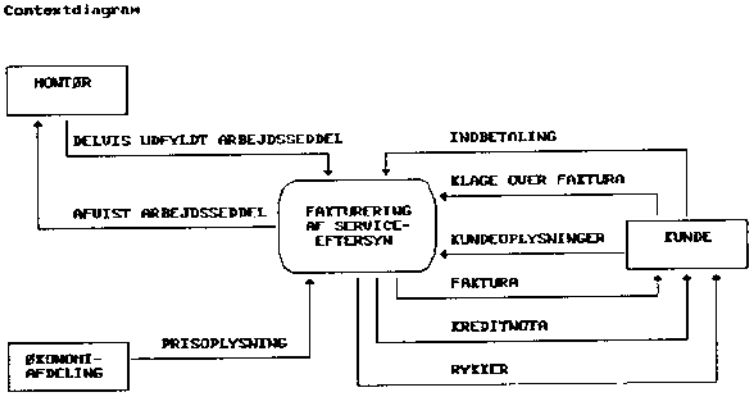
Leverandør/modtager-symbolet navngives efter den pågældende interessent til systemet, fx et andet system, en person, en myndighed, osv.

Identifikation og reference

For at holde orden på de forskellige modeller, skal alle systemelementer være forsynet med en reference.

Contextdiagram:

Har intet nummer, og navngives "Contextdiagram". Processen har systemets navn, men intet nummer.



DFD:

Hvert diagram skal være forsynet med hoved- eller bundlinje, der som minimum skal indeholde systemets navn, diagrammets navn, samt diagrammets nummer. Indenfor et konkret projekt kan der være yderligere krav af hensyn til konfigurationsstyring.

Alle diagrammer har samme nummer og navn som den proces, de beskriver. Figur 0 er en beskrivelse af contextdiagrammets proces, og har derfor samme navn som systemet.

Normalt tilføjer man et ekstra ciffer for hvert niveau, man går nedad i systemet. Hvis systemet har mange niveauer eller indgår i et eksisterende nummersystem, jfr. afsnit om systemforvaltning, kan man vælge kun at skrive sidste ciffer i procesboblen, som så har diagrammets nummer som (underforstået) præfiks.

Processer:

Hver proces har et hierarkisk nummer, som nævnt ovenfor. Den tilhørende procesbeskrivelse identificeres ved processens fuldstændige nummer og navn. Sidste ciffer i processens nummer kan vælges frit. Der kan således ikke på et givet diagram aflæses en behandlingssekvens ud fra numrene.

Registre og datastrømme:

Defineres i dataordbogen, og refererer hertil ved hjælp af et entydigt navn. Bemærk: En datastrøm må gerne optræde flere steder i modellen med det samme navn, endda på samme diagram, når bare dens sammensætning og status , fx godkendt eller ajourført, er identisk.

Dataordbog

Symboler og indhold

Dataordbogen er den del af modellen, hvor systemets datastrømme, dataelementer og registre beskrives. Med de viste symboler kan man vise gentagelse af elementer (eller grupper af elementer), samt valg mellem alternativer, som ikke kan forekomme samtidig.

Bemærk, at tegnet '+' ikke angiver en bestemt rækkefølge.

```
   =       'Består af'
   +       'Og sammen med'
 (   )     'Gentagelse'
[  |  ]    'Valg'
 (   )     'Ikke altid til stede'
```

Anvendelse af kommentarer

Hvis man analyserer et eksisterende system, har man mulighed for at registrere mange andre oplysninger om de fysiske data, såsom format, værdimængde, formelle kontroller, osv. Disse oplysninger kan, hvis det er vigtigt at fastholde dem, tilføjes som kommentar (*markeres sådan*), eller man kan bruge en specialblanket til beskrivelse af hvert dataelement for sig.

Først i den nye fysiske model eller senere, er det nødvendigt at fastlægge format og værdimængde for de data, som skal med i det nye system.

Hvis man har behov for at beskrive 'livsforløbet' for et dataelement eller register, kan man anvende et tilstandsdiagram, som beskrives nærmere i håndbogen om Dataanalyse.

'Tomme' data

En datastrøm må ikke defineres som en 'individstruktur', hvor der er afsat tomme pladser til data, som (endnu) ikke er til stede. Dette er kun tilladt ved definition af et register. Datastrømme, som indgår i en proces, må kun indeholde de data, denne bestemte proces har brug for. I fysiske modeller kan denne regel fraviges, men man bør angive i dataordbogen hvilke data, der er relevante.

Detaljeringsgrad

Der er ingen grund til at detaljere definitionen af dataelementer mere end nødvendigt for forståelsen. Et CPR-nummer kan fx defineres ned til de enkelte cifre, men det vil kun være relevant hvis man selv skal designe en checkciffer-kontrol, eller lignende.

Formelle kontroller

Formelle kontroller af dataelementer kan samles i en procesbeskrivelse, men det kan være mere hensigtsmæssigt at be-

skrive værdimængden som en kommentar i dataordbogen. Dette gælder især hvis det pgl. dataelement kontrolleres flere steder i systemet.

Dataordbogens struktur

Hvis man vælger at lade dataordbogen indeholde både dataelementregister- og datastrøms-beskrivelser, og den skal fungere som fælles reference for et projekt, bør den kunne læses i alfabetisk rækkefølge eller have en alfabetisk søgemulighed.

Eksempel på dataordbog:

```
Faktura =       Fakturanummer
                + [ Kundenummer | Kundedata ]
                  * til faste kunder udskrives alene kundenr. *
                + 1{ Fakturalinje }8
                + total excl. moms
                + (momsbeløb)              * kun for danske kunder *
                + totalbeløb

Fakturalinje =  Varenummer
                + Varebetegnelse   * kan både være reservedele og
                                     kørsel, arbejdstimer, etc.  *
                + Enhedspris
                + Vareantal
                + Varebeløb

Kundedata =     Navn
                + Adresse
                + Postnummer
                + Postdistrikt
```

Man kan også vælge at opdele ordbogen i flere dele, da register- og dataelement-beskrivelserne skal bruges i hele systemet, samt til dataanalyse og detaljeret design.

Man kan således vælge at knytte datastrømsbeskrivelserne til det enkelte DFD, hvor datastrømmen optræder.

Hvis man anvender et CASE-værktøj, har man mange muligheder for at få overblik over systemet, både på skærm og papir. Herved bliver den fysiske organisation af dataordbogen mindre afgørende.

Procesbeskrivelse

En procesbeskrivelse (PB) beskriver den fremgangsmåde der skal følges for at danne processens uddata, herunder beregninger, transformation og evt. logiske kontroller.

Hvis det er nødvendigt for forståelsen, kan PB'en indledes med en kortfattet beskrivelse af processens formål. Dette må ikke gøres til generel standard for projektets PB'er, da formålet normalt vil fremgå tilstrækkeligt tydeligt af processens navn.

Procesbeskrivelsen må ikke indeholde information som er overflødig, eller som fremgår af andre dokumentationselementer. For eksempel:

- liste over ind- og uddata
- liste over registre
- henvisning til hvilken proces, der går forud og hvilken, der følger efter
- tidspunkt for igangsætning
- omtale af data som er læst, men som ikke bruges.

Hvornår skrives procesbeskrivelse

Der skrives PB for alle processer på nederste niveau, dvs. når det ikke er nødvendigt eller hensigtsmæssigt at detaljere DFD'et yderligere. Det nævnes ofte som rettesnor, at en PB ikke bør fylde mere end een side tekst. Dette krav kan ikke altid opfyldes, fx kan en redigering med mange felter og beregninger meget vanskeligt beskrives udtømmende på een side.

I store systemer kan der undertiden opstå behov for en overordnet beskrivelse af formålet med systemet. Sådanne beskrivelser bør ikke skrives som procesbeskrivelser for de overordnede processer, for ikke at skabe redundans og vedligeholdelsesproblemer. Formålsbeskrivelsen hører til i et indledende afsnit i systembeskrivelsen.

Valg af beskrivelsesform

Metoden SA stiller ingen krav til procesbeskrivelsens form. De mest gængse beskrivelsesformer er følgende:

- begrænset sprog
- struktureret sprog
- beslutningstabeller

Når man vælger blandt disse er det vigtigt at holde sig for øje, at PBen skal kunne læses af personer med forskellige forudsætninger, og at den skal kunne vedligeholdes og mangfoldiggøres på en hensigtsmæssig måde.

Ovennævnte hensyn fører som regel til, at man vælger en verbal fremstillingsform til den endelige beskrivelse. Der er dog intet i vejen for, at man i selve funktionsanalysen bruger andre værktøjer i situationer, hvor det er hensigtsmæssigt. I det følgende gives en kort karakteristik af de enkelte beskrivelsesformer:

Begrænset sprog:

Beskriv behandlingen i 'telegramstil' i aktiv form (bydemåde). Brug kun de navne på data, som er defineret i dataordbogen. Gør præcis rede for betingede handlinger

(*'hvis dato er udfyldt, og kode-x = 2, så...'*),

og husk den modsatte betingelse!

(*'..., ellers...'*).

Beskriv betingelsen for iterationer

(*'for alle...', 'så længe...', 'indtil...'*).

Pas på, ikke at overdrive stilen. Brug hellere lidt flere ord, hvis det får meningen klart frem.

Eksempel:

```
Proces 2.3.1
Udfyld Kundeoplysninger på faktura

Hvis Kundenummer findes på Arbejdsseddel,
og kundenummer findes på Kunderegister,
så skriv Kundenummer.

Hvis Kundenummer findes på Arbejdsseddel
men ikke på Kunderegister, og hvis Kunde-
data findes, så dannes Fejlmeddelelse 1
(inkonsistente oplysninger).

Hvis Kundenummer ikke findes på Arbejds-
seddel, men Kundedata findes, så skriv
Kundedata.

I alle andre situationer afvises Arbejds-
seddel med Fejlmeddelelse 2 (oplysninger
mangler).
```

Struktureret sprog:

En yderligere formalisering af begrænset sprog, som nærmer sig begrebet pseudokode.

Iterationer og betingelser, evt. i flere niveauer ('nested'), angives med indrykning af teksten. Hver indrykning afsluttes med en end-markering.

Metoden er til de fleste formål unødigt formel. Den er vanskelig at skrive og læse for andre end programmører, og kan derfor ikke anbefales generelt. Til specielle formål, fx et kompliceret regelsæt med flere betingede handlinger, kan man dog udnytte dens præcision i kombination med en mere læsevenlig beskrivelse.

Eksempel:

```
Proces 2.3.1
Udfyld Kundeoplysninger på faktura


Hvis Kundenummer på Arbejdsseddel
   Hvis Kundenummer på Kunderegister
      Udfyld Kundenummer
   Ellers
      Hvis Kundedata på Arbejdsseddel
         Skriv Fejlmeddelelse 1
      Ellers
         Skriv Fejlmeddelelse 2
      Sluthvis
   Sluthvis
Ellers
   Hvis Kundedata på Arbejdsseddel
      Skriv Kundedata
   Ellers
      Skriv Fejlmeddelelse 2
   Sluthvis
Sluthvis
```

Beslutningstabeller:

Se illustration. Beslutningstabeller kan for visse typer processer være et nyttigt analyseredskab som sikrer, at man tager stilling til samtlige mulige situationer.

Beslutningstabel:

Proces 3.2.1 Udfyld Kundeoplysninger på faktura

BETINGELSER:	1	2	3	4	5	6	7	8
Kundenummer på Arb.seddel	J	J	J	J	N	N	N	N
Kundenummer på Kundereg.	J	J	N	N	J	J	N	N
Kundedata på Arb.seddel	J	N	J	N	J	N	J	N
HANDLINGER:								
Skriv Kundenummer	X	X						
Skriv Kundedata					X		X	
Fejlmeddelelse 1			X					
Fejlmeddelelse 2				X		X		X

Reduceret beslutningstabel:

Proces 3.2.1 Udfyld Kundeoplysninger på Faktura

BETINGELSER:	1	2	3	E
Kundenummer på Arb.seddel	J	J	N	
Kundenummer på Kundereg.	J	N	-	
Kundedata på Arb.seddel	-	J	J	
HANDLINGER:				
Skriv Kundenummer	X			
Skriv Kundedata			X	
Fejlmeddelelse 1		X		
Fejlmeddelelse 2				X

Når tabellen er udarbejdet og reduceret til de relevante situationer, er det nemt at omskrive resultatet til verbal form.

Beslutningstabeller er vanskelige at læse og vedligeholde hvis de indeholder mere end 4-5 betingelser. Iøvrigt henvises til speciallitteratur om udarbejdelse og anvendelse af beslutningstabeller.

Regler og konventioner

Generelle regler

Context-diagrammet

Leverandør/modtagerne på context er systemets interessenter. Hvis analyseområdet er en del af et større systemkompleks, kan et register eller en database optræde som leverandør/modtager. Hvis det pgl. register er direkte tilgængeligt for vort system, vises det med registersymbolet.

I den endelige dokumentation må leverandør/modtager - symbolet kun forekomme på context-diagrammet.

Datastrømsdiagrammer (DFD)

Et DFD bør af hensyn til overskueligheden ikke indeholde for mange processer, dvs. højst 7 - 9. Antallet kan reguleres ved at ændre opdelingen i niveauer. For få processer pr. DFD - og dermed flere niveauer - fremmer dog heller ikke overblikket.

Det vigtigste hensyn, man skal have for øje ved vurderingen, er en forståelig og funktionsorienteret opdeling af processerne. Dette gælder ikke mindst for figur 0 i de logiske

modeller, hvor man ønsker et overblik over samtlige funktioner (hændelser) på samme diagram.

Navngivningen er tit den bedste rettesnor. Hvis niveaudelingen er funktionel, vil et præcist og dækkende navn til processen ofte give sig selv, hvorimod en tilfældig eller uhensigtsmæssig opdeling viser sig ved, at det er umuligt at finde et godt navn. I det nuværende fysiske system kan den uhensigtsmæssige opdeling dog være en selvstændig pointe, som først skal ændres ved overgangen til den nuværende logiske model.

Modellen afbilder behandlingen af *en enkelt forekomst* af en hændelse, men skal naturligvis tage højde for alle de varianter, der kan forekomme. En af konsekvenserne heraf er, at alle navne på datastrømme og processer skrives i ental. Man kan også sige, at modellen er et snapshot eller øjebliksbillede. Den viser intet om systemets igangsætning, overgivelse af kontrol eller behandlingssekvens, bortset fra den sekvens der kan udledes af datastrømmens bevægelsesretning.

Et DFD må derfor ikke indeholde "datastrømme", som udelukkende tjener styringsformål, dvs. kontrolstrømme, kørselsparametre, end-of-file-markeringer og lignende.
Undtagelse: Ved design af online-delen i den ny fysiske model kan man vælge at beskrive dialogen ved hjælp af tilstandsdiagrammer, kontrolstrømme og kontrolprocesser, som beskrives med en særlig notation (se brochuren om Dialognotation).

For at undgå for lange, snørklede og krydsende datastrømme, er det tilladt at tegne det samme register flere (højst 2) steder

på diagrammet. Dette markeres på begge forekomster med '*' i eller ved siden af registersymbolet. I mange tilfælde kan problemet dog løses ved en ændret placering af registret i forhold til de processer, der benytter det.

Krydsende datastrømme er tilladt, men kan minimeres ved et gennemtænkt layout af diagrammet.

Processer

En proces er altid aktiv; den skal ikke startes eller stoppes. Den kan foretage sin behandling når de nødvendige input-datastrømme er til stede.

Hvis en proces modtager identiske datastrømme flere steder fra, kan den kun behandle dem på samme måde. Hvis processen har brug for at skelne mellem de to datastrømme, må de derfor navngives forskelligt.

En proces kan ikke "huske" data. Hvis det er nødvendigt at gemme data fra tidligere gennemløb af processen, eller at kende antallet af foretagne gennemløb, må man derfor selv sørge for at gemme oplysningerne i et register.

Ved gennemgang af processernes navne kontrollerer man, at hvert overordnet procesnavn er dækkende for de funktioner, der udføres af de underliggende processer. Hvis det er svært at finde på et dækkende navn, tyder det på en ikke-funktionel opdeling af systemet.

Datastrømme

En datastrøm indeholder en enkelt forekomst (den aktuelle) af det pågældende data. Datastrømmen kan også være tom, hvis betingelserne for at danne den ikke er til stede.

Når en proces skriver data på et register, tegnes en datastrøm *til* registeret. Læsning på et register tegnes som en datastrøm *fra* registeret. Det er med andre ord en bevægelsesretning af data, der vises, selv om man rent fysisk er nødt til at læse en record på et edb-register før man kan rette i den.

NB: Hvis man skal addere et tal til et eksisterende (fx ved bevægelser på en bankkonto, hvor man beregner og gemmer den aktuelle saldo), eller hvis det gamle data skal bruges til en kontrol før der kan opdateres, så skal diagrammet vise, at processen benytter registerdata til sin behandling. Der skal derfor være datastrømme både til og fra registeret.

Registre

Et register, der kun benyttes i en del af systemet, vises ikke før det DFD-niveau, hvor det har forbindelse med to eller flere processer. Hvis registeret kun benyttes af en enkelt proces på laveste niveau, skal det dog tegnes med her.

Registersymbolet behøver ikke tegnes med på de underliggende diagrammer, efter at det er vist første gang. Datastrømme med registerdata kan da tegnes som "løse" pile fra diagrammets omverden, på samme måde som datastrømme fra andre hovedprocesser.

Man kan dog vælge at tegne registrene med på de underliggende diagrammer, da det ofte giver et bedre overblik over sammenhængen i systemet.

Supplerende regler for logiske modeller

En proces må ikke modtage datastrømme, som den ikke benytter (ændrer eller bruger til kontrol af andre data), eller som den sender uændret videre ("vagabonderende data"). Hvis processen benytter et, eller ganske få, data fra en sammensat datastrøm, bør man i stedet tegne særskilte datastrømme med de benyttede dataelementer.

På figur 0 går alle datastrømme mellem processerne via logiske registre. Dette følger nødvendigvis af opdelingen af funktioner efter hændelser.

Hvis et DFD indeholder processer, som ikke kommunikerer med andre processer eller registre på samme DFD, er det tegn på, at nedbrydningen ikke er hensigtsmæssig. Hvis den "ensomme" proces ikke hører mere naturligt hjemme på et andet underordnet DFD, hører den til på figur 0.

Ethvert registerdata kan umiddelbart læses. I en logisk model skal man ikke tænke på - eller beskrive - accessveje, returkoder, pointere, osv.

Et logisk register må ikke indeholde data som ikke benyttes af systemet, eller af andre systemer som registeret betjener.

Online-systemer

Det er en udbredt opfattelse blandt systemudviklere, at online-systemer er en særlig kategori af systemer, som er vanskelig at beskrive tilfredsstillende ved hjælp af Struktureret Analyse.

Denne opfattelse er kun delvis korrekt. En stor del af vanskelighederne kan føres tilbage til fejl ved udarbejdelsen af de logiske modeller, idet man ofte fristes til at nedbryde sit system i "skærmbilleder" i stedet for i hændelser, hvis der på forhånd er stillet krav om, at systemet skal designes som et online-system.

Et skærmbillede er ikke en proces, men derimod en (fysisk) datastrøm. I de logiske modeller defineres dens dataindhold i dataordbogen på normal måde, evt. sammen med kriterier for de enkelte dataelementers validering.

Det er vigtigt under arbejdet med de logiske modeller, at man betragter hændelser og respons mere overordnet, i lyset af systemets formål og fundamentale aktiviteter. Først ved design af den nye fysiske model afgrænser man online-delen og designer dialoger, skærmbilleder, osv.

Som led i designarbejdet kan man udvide funktionsmodellen med tilstandsdiagrammer, kontrolstrømme, osv., hvilket kræver andre notationsformer end SA. Hvis man anvender prototype-værktøjer, kodegeneratorer mv., er der ofte indbygget forskellige muligheder for automatisk generering af programdokumentation.

Validering og balancering

Når man har gjort en model færdig, skal den kontrolleres for korrekthed og konsistens.

1. Man kontrollerer den saglige korrekthed - om modellen beskriver systemet fyldestgørende og i overensstemmelse med virkeligheden - ved at sammenholde modellens processer, data og eksterne grænseflader med den foregående model. Den nuværende fysiske model kan direkte kontrolleres mod den virkelige verden, evt. i form af kundens godkendelse af beskrivelsen.

2. Den formelle korrekthed (validering) handler om overholdelse af regler og konventioner for diagrammer og dataordbog. Man kontrollerer fx om alle datastrømme er defineret i dataordbogen. Hvis en datastrøm er defineret i flere niveauer, skal hvert niveau være defineret, ned til det enkelte dataelement.

   Der må ikke forekomme processer, som kun modtager datastrømme og ikke afgiver nogen, eller omvendt.

3. Kontrol af modellens konsistens (balancering) sker ved at sammenligne dens forskellige bestanddele.

   Hvert DFD sammenholdes med det overliggende (figur 0 med context-diagrammet). Antal, retning og navn på samtlige datastrømme, som har forbindelse med processer eller registre på andre diagrammer, skal kunne genfindes på det overliggende diagram.

   Navne og hierarkiske numre på DFD'er og processer skal være konsistente.

   Hvis man arbejder i et CASE-værktøj, kan kontrollerne under punkt 2 og 3 udføres maskinelt.

4. Hver procesbeskrivelse sammenholdes med det overliggende DFD. Alle datanavne som indgår i behandlingen skal være repræsenteret i en datastrøm *til* processen, og alle data som produceres skal kunne findes i en datastrøm *fra* processen.

   Alle udgående datastrømme fra processen skal kunne dannes af de indgående data, ud fra regler og konstanter i procesbeskrivelsen.

5. Alle navne på data og registre skal svare til dataordbogens definitioner.

Valg af fremgangsmåde

Siden Struktureret Analyse først blev beskrevet samlet af Tom DeMarco i 1978, har andre forfattere videreudviklet metoden, og vurderet dens anvendelighed i forskellige situationer.

Det mest vidtrækkende valg, systemudvikleren skal tage stilling til, er hvorvidt man skal analysere det eksisterende system, eller gå direkte til en logisk model af det nye system. Dette afsnit indeholder retningslinjer for valg af fremgangsmåde.

Den klassiske metode ("de fire faser")

Struktureret Analyse, som beskrevet af DeMarco, tager udgangspunkt i et eksisterende administrativt system eller en forretningsgang, som kan være enten manuel eller automatiseret. Modellen af det nuværende fysiske system reduceres derefter til en nuværende logisk model for at sikre, at de nye funktioner kommer til at arbejde optimalt sammen med de dele af det gamle system, som skal videreføres. Fremgangsmåden beskrives mere detaljeret i næste afsnit.

Fordelen ved den klassiske metode er, at den giver en sikker reference til den virkelige verden, idet beskrivelsen direkte kan genkendes og godkendes af brugerne. Herved sikrer man sig mod, at systemudviklerne designer systemet efter deres egen opfattelse af kundens arbejdsrutiner og problemer, som ikke nødvendigvis svarer til virkeligheden.

Der vil altid være en risiko for at hænge fast i beskrivelsen af det eksisterende systems mange detaljer. Såvel den enkelte systemudvikler som projektets ledelse skal hele tiden holde sig for øje, at beskrivelsen af den nuværende system i de fleste tilfælde kun skal tjene som grundlag for en ændring af funktioner og forretningsgange.

Blitzing

Flere forfattere har forsøgt at beskrive en hurtigere udviklingsproces, ved at modificere DeMarcos arbejdsmetodik. McMenamin/Palmer kalder deres udgave for "blitzing", og denne betegnelse vil blive brugt i det følgende.

Ideen i blitzing er, at man starter med at skitsere en logisk model af det nye system, hvor man kun inddrager den nuværende fysiske implementering på de punkter i detailanalysen, hvor der er tvivl om funktionernes indhold.

Hvad skal man vælge?

Nedenstående betingelser kan betragtes som en rettesnor for, hvornår det er forsvarligt at gå direkte til den nye logiske model efter den metode, som beskrives i afsnittet Blitzing:

- der er udført foranalyse
- mange af det nuværende systems funktioner skal ændres eller nedlægges i det nye system
- der findes en rimelig dokumentation af det nuværende system
- kravene til det nye system er veldefinerede
- systemet størrelse og kompleksitet er ikke større end, at det vil være muligt at skitsere en logisk model med en begrænset indsats af tid og ressourcer.

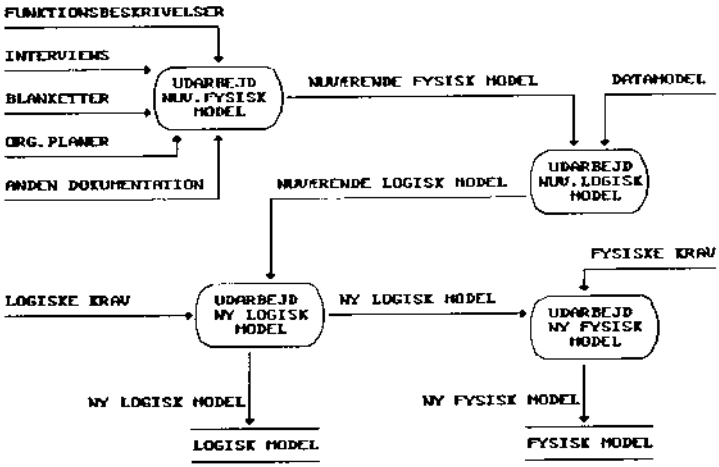
Hvis flere af disse betingelser ikke er opfyldt, bør man følge den klassiske fremgangsmåde, og starte med at udarbejde en nuværende fysisk model.

Arbejdsmetodik for de fire faser

Den "klassiske" model for analysen er opdelt i fire faser:

- nuværende fysisk model
- nuværende logisk model
- ny logisk model
- ny fysisk model

som vist på nedenstående figur. I dette afsnit beskrives udgangspunkt og fremgangsmåde for de enkelte faser.



Nuværende fysisk model

Formål:

Systemudviklerne og kunden skal opnå en fælles forståelse af det nuværende systems formål og virkemåde. Modellen viser *hvordan* systemet virker, og kan derfor kontrolleres ved direkte at sammenholde den med virkeligheden.

Udgangspunkt:

Systemet analyseres på grundlag af eksisterende dokumentation (beskrivelse af forretningsgange, organisations- og bemandingsplaner, brugsvejledninger, blanketter, etc.), interviews af nøglepersoner i virksomheden samt de medarbejdere, der arbejder med systemet til daglig. Hvis der tidligere er udført foranalyse, kan en større eller mindre del af analysen være indeholdt i foranalyserapporten.

Fremgangsmåde:

Analysen udføres og dokumenteres ved hjælp af de symboler, regler og konventioner, der er beskrevet i de foregående afsnit. Der kan ikke angives en præcis fremgangsmåde, som er ideel for alle projekter. De følgende retningslinjer vil kunne anvendes i de fleste tilfælde, men kan fraviges efter behov:

1. Begynd med at afgrænse analyseområdet, og skaf overblik over systemets hovedfunktioner. Studer eksisterende skriftlig dokumentation, og tal med personer, der har et bredt

kendskab til virksomheden. Lav udkast til contextdiagram og figur 0. Start dataordbogen, og definer alle datastrømme (blanketter, dokumenter, osv.) som optræder på diagrammerne.

Arbejdet bør udføres i tæt samarbejde med brugerne. På denne måde opnår man den størst mulige sikkerhed for, at beskrivelsen stemmer overens med virkeligheden, så der ikke opstår misforståelser. Hvis det ikke er muligt at inddrage brugere direkte, skal kunden verificere de overordnede udkast før man går videre.

2. Undersøg hovedfunktionerne og beskriv dem i detaljer. Planlæg detailanalysen på forhånd. Overvej, om dele af systemet kan analyseres parallelt af forskellige personer, og hvilken rækkefølge, der er mest hensigtsmæssig. Aftal tid til interviews med de brugere, som varetager de pågældende funktioner.

   Interviews fastholdes med DFD-skitser, suppleret med notater og evt. båndoptagelse. De foreløbige skitser skal gøres tilgængelige for alle på projektet, således at nye sammenhænge eller modstridende oplysninger kan blive opdaget og fulgt op så hurtigt som muligt.

   Man arbejder parallelt med de overordnede niveauer (context og figur 0), detaljerede DFD'er for de enkelte funktioner, samt dataordbog. Vær forberedt på at skifte mellem forskellige dele af beskrivelsen, og om nødvendigt gå tilbage og rette og supplere, når der dukker nye detaljer op. Derfor bør finpudsning og rentegning af diagrammerne udsættes så længe som muligt.

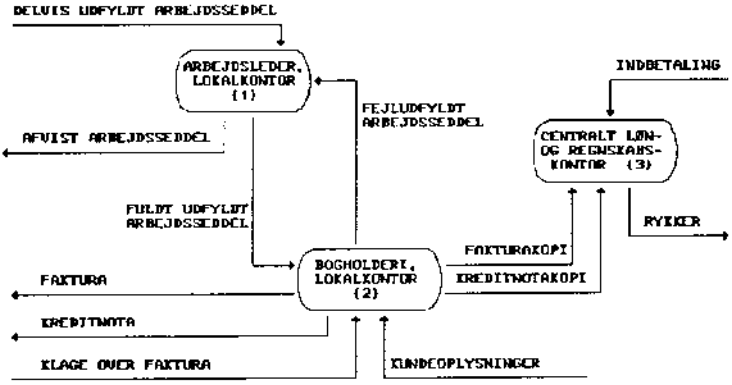
Først når modellen er ved at være færdig, fastlægges den endelige opdeling i hovedfunktioner og niveauer, og de detaljerede behandlingsregler omskrives til procesbeskrivelser.

3. Modellen gøres færdig og rentegnes/skrives, evt. ved hjælp af et CASE-værktøj. Løse ender i modellen identificeres og afklares, evt. i samråd med brugerne.

   Systemudvikleren skal holde sig for øje, at beskrivelsen kun skal bruges som udgangspunkt for den logiske model. Funktioner og data, der forventes at skulle udgå, behøver derfor kun beskrives summarisk. Hvis der i forvejen findes dokumentation af data, forretningsgange og detaljerede behandlingsregler, kan man nøjes med at henvise hertil.

Eksempel:

Fig.8, nuværende fysisk model



Nuværende logisk model

Formål:

Modellen skal vise, *hvad* der sker i det nuværende system, uden hensyn til fysiske begrænsninger og fejlkilder, og beskrevet med mindst mulig redundans. Derved sikrer man, at det nye system kan designes hensigtsmæssigt ud fra de logiske krav, uden afsmitning fra den tidligere implementering.

Udgangspunkt:

Den nuværende fysisk model er grundlaget hvorfra man udleder den nuværende logiske model.

Fremgangsmåde:

1. Identificer og beskriv systemets eksterne hændelser.Undersøg alle indgående datastrømme på contextdiagrammet for at afklare, hvilke der er udtryk for en selvstændig hændelse. Lav evt. en hændelsesliste, som udfyldes med samtlige hændelser og deres forventede respons.

   Tegn for hver hændelse et DFD, som viser alle de logiske processer, som udføres indtil systemet har givet fuldstændig respons på hændelsen.

2. Identificer og beskriv systemets interne hændelser. Undersøg alle udgående datastrømme på contextdiagrammet, som ikke hører til en af de eksterne hændelser. Beskriv de

logiske processer i hændelsesmodeller, som under pkt. 1.

Undersøg om der er læsning eller skrivning på registre, som ikke er beskrevet tidligere. I så fald kan det være en intern hændelse, som ikke kommunikerer med omverdenen.

3. Analyser hændelserne hver for sig. Undersøg hvilke data, der er nødvendige for at systemet kan danne respons, og tegn de nødvendige datastrømme til processerne på alle niveauer i beskrivelsen.

   Giv alle datastrømme, der ikke kommunikerer med omverdenen, et logisk navn uden binding til et fysisk datamedium.

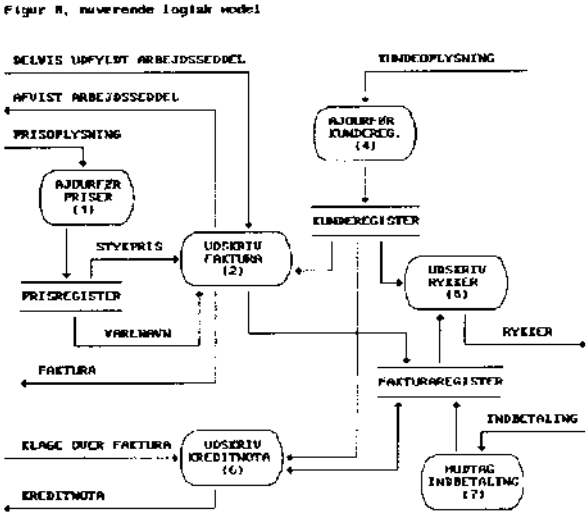
4. Kontroller, at ingen fysiske processer, dvs. processer der kun foretager sortering, konvertering, fordeling og transport af data, er medtaget. Processer, der kontrollerer data, bevares kun på grænseflader, hvor systemet læser datastrømme fra omverdenen.

5. Udfør dataanalyse, hvis det ikke er gjort tidligere. Tegn de logiske registre ind på DFD'erne. Forbind datastrømmene med registre. Hvis en datastrøm indeholder data fra flere logiske registre, skal den opdeles.

6. Tegn figur 0 på basis af samtlige hændelsesmodeller. Hver hændelse skal som udgangspunkt svare til en proces på figur 0, men det kan være nødvendigt at slå beslægtede hændelser sammen for at gøre diagrammet mere overskueligt. Giv processerne navne og numre, og tilret de øvrige diagrammer i overensstemmelse hermed.

7. Gør modellen færdig. Vurder i hvilket omfang, der er behov for at niveauopdele DFD'erne. Skriv procesbeskrivelser for alle processer på nederste niveau. Fjern fysiske elementer fra dataordbogen, og beskriv de logiske registre og relationer.

Eksempel:



Bemærkning 1:

I de logiske modeller ser man bort fra behandlingstid i systemet, ligesom tidsterminer og andre arbejdsvilkår i systemets omverden ignoreres. Hvis en hændelse afbrydes - og data dermed må gemmes i et register - skal det derfor være af rent logiske årsager, fx at systemet rekvirerer inddata fra systemets omverden.

Bemærkning 2:

Hvis den foreløbige analyse giver mistanke om, at der må være glemt en eller flere hændelser, kan det være nyttigt at udarbejde et tilstandsdiagram, som viser hele "livsforløbet" for en entitet. Det kan fx være relevant for entiteter, som både oprettes og slettes af systemet uden en direkte impuls fra omverdenen. Om brug af tilstandsdiagrammer, se brochuren om dataanalyse.

Bemærkning 3:

Hele analyseforløbet er en iterativ proces, dvs. at man skal gå tilbage og rette eller supplere et tidligere resultat hvis man senere får ny indsigt. Det kan således forekomme, at man under arbejdet med en logisk proces opdager, at den ikke er analyseret tilstrækkeligt i den forrige fase. Man bør da gå tilbage og undersøge de uklare forhold, evt. ved hjælp af yderligere interviews.

Ny logisk model

Formål:

I denne fase analyseres de logiske krav til det nye system, og løsningen indføjes i den nuværende logiske model. Den ny logiske model viser, *hvad* der sker i det nye system.

Formålet med at indarbejde kravene i en logisk model, fremfor i den eksisterende implementering af systemet, er at sikre sig mod ukontrolleret "knopskydning", hvor succesive ændringer komplicerer systemets struktur mere og mere.

Udgangspunkt:

Denne fase tager udgangspunkt i den nuværende logiske model, samt kravspecifikationen og evt. supplerende aftaler.

Fremgangsmåde:

1. Alle krav til systemet samles, og opdeles i logiske krav (krav til funktioner og data), og fysiske krav (krav til systemets fysiske udformning og virkemåde).

2. Hvert af de logiske krav analyseres. En ny funktion skitseres som et DFD, og funktionens behov for inddata og registerdata afklares. Nye data defineres i dataordbogen. En ny hændelse skal i princippet behandles som en ny proces på øverste niveau i modellen.

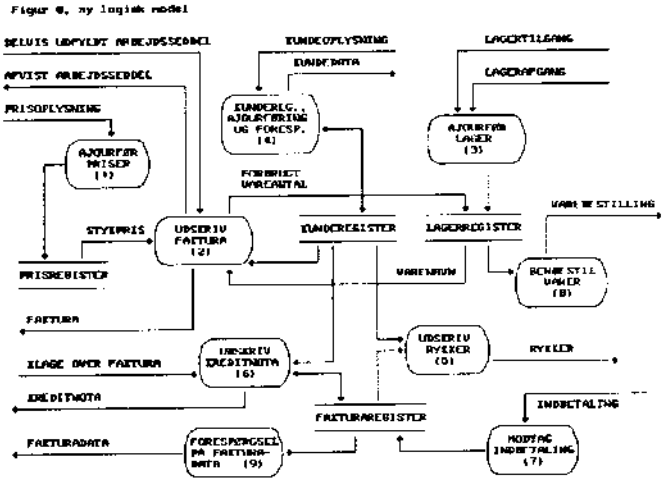
Ændringer til eksisterende funktioner og datastrømme indarbejdes i de relevante DFD'er, procesbeskrivelser og dataordbog.

3. Kontroller den reviderede model med henblik på korrekthed og overskuelighed. Hvis en procesbeskrivelse er blevet for omfangsrig, bør det overvejes at dele den op. Dette medfører et ekstra DFD-niveau.

   Hvis figur 0 som følge af nye hændelser er blevet uoverskuelig, kan det være nødvendigt at slå nogle af processerne sammen. Det kan være beslægtede hændelser, eller fx ajourføring af en række registre. Modellen skal vise de separate hændelser på næste niveau.

4. Hvis der er ændret i registrenes indhold, eller der er tilkommet nye registre, foretages dataanalysen påny.

Eksempel:



Bemærkning:

De fundamentale processer er de vigtigste for forståelsen af systemets formål og virkemåde, og de bliver derfor som regel beskrevet først og grundigst. Når man kontrollerer den samlede models fuldstændighed, må man derfor ikke glemme vedligeholdelsesprocesserne. Alle registre skal kunne ajourføres, og forældede data skal kunne slettes. Man skal tage stilling til, om systemet har behov for at kunne læse tidligere generationer af data, eller om den nyeste version er tilstrækkelig.

Ny fysisk model

Formål:

Ny fysisk model giver systemudviklerne og kunden en fælles forståelse af, *hvordan* det nye system skal udføre den behandling, der er beskrevet i ny logisk model.

Udgangspunkt:

Design af ny fysisk model tager udgangspunkt i den nye logiske model, samt kravspecifikationens forudsætninger og krav til systemets fysiske udformning.

Kunden skal have mulighed for at tage stilling til et eller flere forslag til overordnet systemdesign. Det valgte forslags opdeling i maskinel/manuel behandling, online/batch, centralt/decentralt system, etc. danner grundlag for det detaljerede design, samt for kundens tilrettelæggelse af de manuelle arbejdsgange omkring systemet.

Fremgangsmåde:

Kundens fysiske krav indarbejdes i den ny logiske model. På de områder, hvor kravene er uklare eller modstridende, kan man enten foretage et valg, bede kunden om at specificere sine ønsker, eller udarbejde alternative forslag. Det kan anbefales at udføre arbejdet i følgende rækkefølge:

1. Beslut, hvilke dele af systemet, der skal automatiseres, og hvordan edb-systemet skal opdeles for at tilgodese kravene (denne opdeling kaldes også mand/maskinsnittet).

2. Edb-systemet opdeles på processorer (maskiner). Hvis behandlingen skal distribueres mellem flere maskiner, tegnes de som særskilte processer på figur 0.

   Hver af disse processer opdeles yderligere (på næste niveau) i online og batch, og disse igen i hændelser svarende til opdelingen af den ny logiske model.

3. På alle grænseflader mellem edb og omverden tilføjes konverteringsprocesser, inddatakontrol, fejlbehandling, redigering, skærmbilleddialog og menuer, osv. På grænsefladerne mellem edb-systemets logiske processer tilføjes administrative processer til at varetage kommunikation, læsning og skrivning af mellemfiler, inddatakontrol, etc.

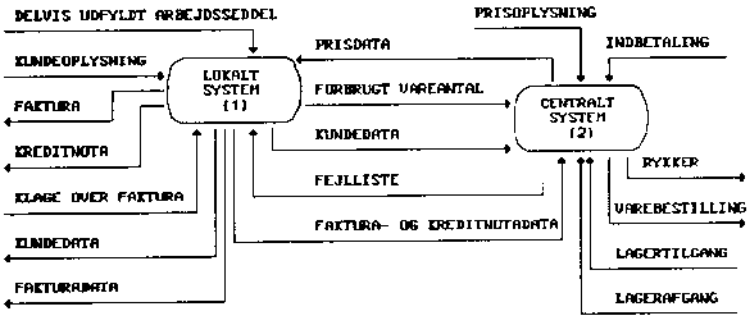
4. De fysiske databaser designes ud fra de logiske registre, under hensyn til den valgte opdeling af edb-systemet. Det vil ofte være nødvendigt at indføre redundans, dels på grund af geografisk askilte maskiner, dels for at opnå den rette balance mellem registeropslag og CPU-tid.

5. Administrationen af systemet tilrettelægges. Der skal være faciliteter til online-adgangskontrol, administration af passwords og brugerautorisationer, back-up, recovery, versions-kontrol, kørselsafvikling, reorganisering, arkivering, etc.

   Disse funktioner vil typisk indgå i et standard-styresystem, men man kan også vælge at skræddersy enkelte funktioner. For alle funktioner skal den valgte løsning være beskrevet, dog ikke nødvendigvis med DFD'er.

Eksempel:

Figur 8, ny fysisk model



Bemærkning 1:

Det vil normalt være kundens ansvar at tilrettelægge de manuelle arbejdsgange. Denne del af modellen skal derfor kun beskrives overordnet, ved hjælp af DFD. Hvis det fremmer forståelsen af systemet, kan man tegne et DFD over brugersystemet, som viser sammenhængen mellem edb-systemet og de manuelle rutiner. Hvis edb-systemet er delt mellem flere maskiner med hvert sit brugersystem, kan diagrammet også vise dette, jfr. eksemplet på næste side.

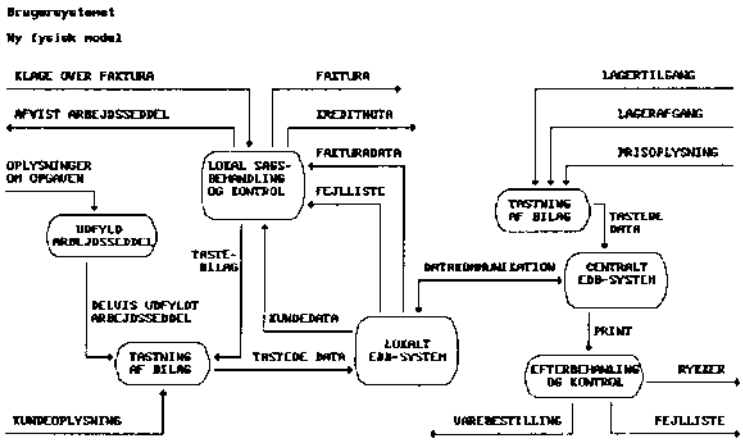
Bemærkning 2:

Hvis der fremlægges flere systemskitser for kunden, bør man ikke detaljere beskrivelsen mere end nødvendigt, indtil han har valgt den løsning, som skal implementeres.

Bemærkning 3:

For de af edb-systemets detailprocesser, som er bevaret fra den ny logiske model, kan man blot kopiere procesbeskrivelserne. Designarbejdet er i denne fase koncentreret om beskrivelsen af konverteringer og øvrige administrative processer og procedurer.

Eksempel:



Blitzing

Formål:

Formålet med blitzing er, hurtigt at nå frem til et overblik over systemets funktioner og data, som kan tjene som udgangspunkt for detailanalysen.

Nedenstående arbejdsmetodik beskriver afvigelserne fra den klassiske metode. De enkelte trin i processen udføres som beskrevet i forrige afsnit.

Udgangspunkt:

Udgangspunktet er en kravspecifikation, som beskriver både de eventuelle eksisterende funktioner og de nye krav.

Kravspecifikationen suppleres med de deltagende brugerrepræsentanters viden om virksomhedens forretningsgang, samt om øvrige krav og forventninger til det nye system.

Alle deltagere i processen skal kunne forstå og anvende begreberne i Struktureret Analyse. Desuden skal brugerrepræsentanterne tilsammen have indblik i alle væsentlige funktioner i det nye system.

Fremgangsmåde:

Systemudviklerne og et antal nøglepersoner fra kundens organisation opstiller en grov skitse til en logisk model, på grundlag af deres viden om det fremtidige systems formål og funktioner, samt evt. eksisterende dokumentation.

Det tilstræbes ikke, at gøre modellen komplet og korrekt i første omgang. Målet er at tilføre projektet en stor viden om systemet på kortest mulig tid, så det efterfølgende udviklingsarbejde kan planlægges effektivt. Det anbefales at udføre arbejdet på en række møder, hvor alle tilstedeværende kan bidrage til modellen.

Processen vil typisk indeholde følgende trin:

1. Fastlæg systemets afgrænsning og formål. Tegn udkast til context-diagram. Find systemets interessenter, og afstem deres behov med systemets formål.

2. Find systemets entiteter (jfr. DA).
   Hvilke data har systemet behov for at gemme?
   Hvordan kan disse data struktureres i logiske registre?

3. Find systemets hændelser.
   Hvordan skal inddata fra interessenterne behandles?
   Hvilke uddata skal systemet kunne levere?
   Hvordan bliver registrene vedligeholdt?

4. Find systemets omverden.
   Er context-diagrammet komplet?

Skal systemet kommunikere med andre systemer?
Skal andre kunne trække på systemets registre?

5. Udarbejd logiske modeller over hver hændelse.
   Tegn diagrammer, som viser de essentielle aktiviteter, som skal udføres for at behandle hændelsen. Beskriv dataindholdet i de tilhørende datastrømme. Vurder aktiviteternes kompleksitet, og muligheden for at skaffe oplysninger om behandlingsreglerne.

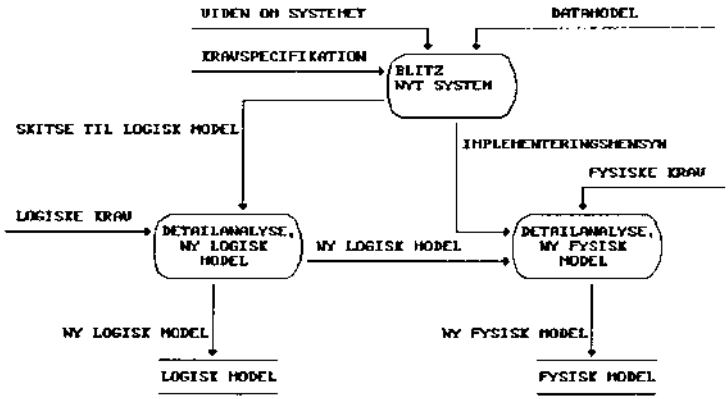
6. Punkt 1 til 5 gennemløbes iterativt, indtil der er opnået et rimeligt overblik over systemet.

7. Integrer hændelsesmodellerne til en samlet logisk model.
   Tegn figur 0, og evt. mellemliggende niveauer. Giv alle processer numre, og kontroller konsistensen af proces-, register- og datanavne.

Det videre forløb:

Den blitzede logiske model er udgangspunkt for planlægning og gennemførelse af den detaljerede analyse, samt for den ny fysiske model, jfr. nedenstående figur.

På grund af fremgangsmåden kan den blitzede logiske model være ufuldstændig. Man skal derfor, i endnu højere grad end ellers, være forberedt på, at der kan dukke detaljer op i detailanalysen, som gør det nødvendigt at revurdere det tidligere arbejde.



Systemforvaltning

De arbejdsmetodikker, som er beskrevet i de forrige afsnit, vedrører udvikling af nye systemer. I dette afsnit omtales fremgangsmåden for næste fase af udviklingen, når systemet skal vedligeholdes og videreudvikles.

Forvaltning efter nyudvikling med Struktureret Analyse

Formål

Det er vigtigt, at man i systemforvaltningsfasen fastholder metode- og dokumentationsstandarden. Det er nu, man får det fulde udbytte af det tidligere analysearbejde, og sikrer systemet mod den gradvise nedslidning, som tidligere blev anset for uundgåelig.

Udgangspunkt

Beskrivelsen af det ny fysiske system er en del af den blivende dokumentation og aftalegrundlaget. Den fysiske model er udgangspunkt for analysen af de nye krav, hvor den indgår som nuværende fysisk model.

Selv om udviklingsfasens nye logiske model ikke er en del

af systemleverancen, bør den være tilgængelig i vedligeholdelsesfasen.

Fremgangsmåde:

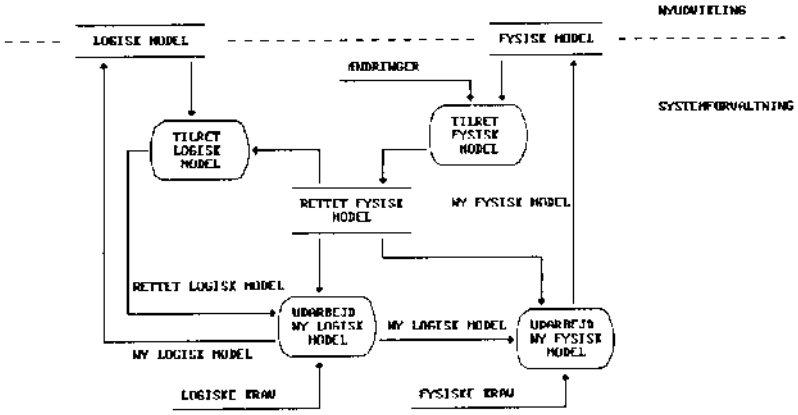
Arbejdsmetodikken består i princippet af de sædvanlige fire faser, idet man dog i vidt omfang kan genbruge resultaterne af det tidligere arbejde (se tegning næste side):

1. *Kontroller nuværende fysisk model*
   Den fysiske model kontrolleres for at sikre, at systemet stadig fungerer som beskrevet. Der kan fx være foretaget hasterettelser, som ikke er dokumenteret tilstrækkeligt.

2. *Udled nuværende logisk model*
   Den nye logiske model fra udviklingsfasen bør umiddelbart kunne genanvendes. Kun hvis pkt. 1 har afsløret udokumenterede ændringer, kan det være nødvendigt at tilføje eller rette logiske processer.

3. *Udarbejd ny logisk model*
   De nye logiske krav indarbejdes, som beskrevet i afsnittet om ny logisk model.

4. *Udarbejd ny fysisk model*
   Det meste af det overordnede systemdesign vil kunne genanvendes. Med udgangspunkt i den tidligere ny fysisk model indarbejdes de nye fysiske krav samt ændringer, som er nødvendiggjort af den ændrede logiske model. Følg arbejdsmetodikken i afsnittet om ny fysisk model.



Forvaltning af systemer med andre dokumentationsformer

Problemformulering

En af årsagerne til, at det er en årelang proces at indføre Struktureret Analyse som generel dokumentationsstandard og analysemetode, er de mange store, kørende systemer, som er udviklet og dokumenteret efter andre standarder, fx SYSKON.

En total omlægning af eksisterende dokumentation er urealistisk i de fleste tilfælde. Det er derfor nødvendigt at finde en praktisk gennemførlig strategi for gradvis omlægning af de gamle systemer.

Det er vigtigt at gøre sig klart, at analysemetode og dokumentationsstandard er to forskellige ting. Man kan derfor uden vanskelighed benytte arbejdsmetodikken i denne håndbog, selv om de endelige dokumentationselementer har andre navne. Dette er ikke mindst vigtigt af hensyn til vedligeholdelsen af medarbejdernes uddannelse.

Find et egnet projekt

Meget systemvedligeholdelse består af små justeringer af grænsefladen til omliggende systemer, teknisk optimering, fejlrettelse, mv. Et fælles træk for disse aktiviteter er, at de kun kræver minimale, eller slet ingen, rettelser af systembeskrivelsen.

Den første forudsætning for at anvende Struktureret Analyse er naturligvis, at der er tale om et analyseprojekt, fx i forbindelse med nye krav fra kunden, gennemgribende teknisk omlægning, eller lignende. Hvis ændringen er af mindre omfang, og væsentligst drejer sig om præsentation af data, dvs. redigering, design af skærmbilleder og dialog, giver SA ikke i sig selv fordele, som kan retfærdiggøre en omlægning af eksisterende dokumentation.

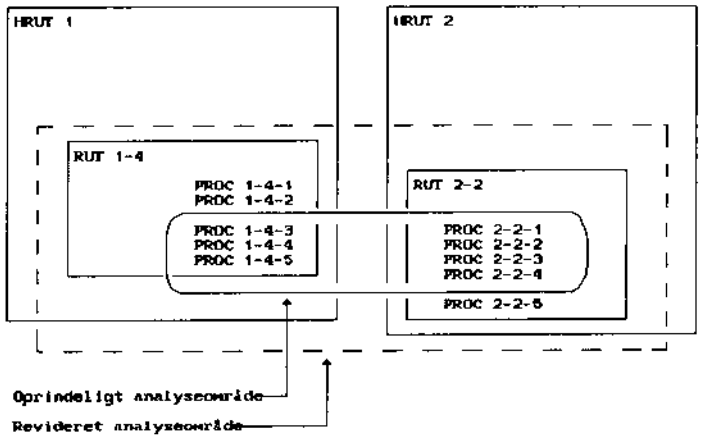
Det er ligeledes en forudsætning, at analyseområdet er nogenlunde sammenhængende. Eksempelvis vil et enkelt nyt eller ændret dataelement, som bruges mange steder i systemet, medføre enten et stort antal usammenhængende analyseområder eller en total omlægning af systembeskrivelsen.

Vælg et passende analyseområde

Hvis der ikke er tale om en større omlægning af et system eller delsystem, vil man ofte stå med et analyseområde, der går på tværs af den eksisterende opdeling i rutiner og procedurer.

Man bør da søge at udvide analyseområdet, så dets grænser falder sammen med de rutiner, eller anden overordnet inddeling af systemet, som berøres af ændringen. Man risikerer herved at skulle omskrive nogle dele af systembeskrivelsen, som ikke skal ændres, for at bevare sammenhængen. Til gengæld kan man genbruge det gamle systems interne grænseflader (mellemfiler og registre). En anden fordel er, at man uden problemer kan indføje den nye beskrivelse i det eksisterende nummersystem.

FASTLÆG ANALYSEOMRÅDET



Fremgangsmåde

Brug arbejdsmetodikken i Struktureret Analyse til enhver analyseopgave. Tegn skitser af problemområder og løsningsforslag ved hjælp af datastrømsdiagrammer. Skriv procesbeskrivelser for de detaljerede behandlingsregler.

Brug den eksisterende dokumentation af filer (datastrømme) og dataelementer. Dataordbogens notationsform er af mindre betydning for analysemetoden, og kan erstattes af en anden standard for databeskrivelser.

Programmeringsgrundlagets opbygning

På Datacentralen bruges betegnelsen programmeringsgrundlag (PG) om systembeskrivelsen for gamle systemer. Opbygningen af PG'er er i store træk følgende:

Systemet er opdelt i hovedrutiner, som hver svarer til et delsystem. I store systemer er der ofte et PG for hver hovedrutine. Hovedrutinebeskrivelsen indeholder formål, listning af ind- og uddata, oversigt over rutiner, initieringsbetingelser, mv.

Hver hovedrutine er opdelt i rutiner, som kan jævnføres med de overordnede processer på figur 0. Rutinebeskrivelsen indeholder formål, listning af ind- og uddata, oversigt over procedurer, initieringsbetingelser, mv.

Hver rutine er opdelt i procedurer, og disse i situationer eller underpunkter efter behov. Procedurebeskrivelsen indeholder

formål, listning af ind- og uddata, initieringsbetingelser, samt de detaljerede behandlingsregler.

Indpasning af SA-dokumentation

Når den nye fysiske model for analyseområdet er udarbejdet, indgår den i den eksisterende dokumentation på følgende måde:

Datastrømsdiagrammerne erstatter hovedrutine- og rutinebeskrivelserne. Nummersystemet tilrettes, således at alle diagrammers og processers numre bliver foranstillet med hovedrutinens nummer.

Procesbeskrivelserne kan udformes med blankethoved mv. efter den eksisterende PG-standard. Man bør ikke tilføje de faste indledningsafsnit, hvis oplysningerne fremgår af DFD'er mv.

Dataordbog, dataelement- og registerbeskrivelser, ER-diagrammer mv. indgår i eksisterende data- og filbeskrivelser, eller evt. i andre dokumentationselementer, som det enkelte projekt finder mere egnet.

Endelig omlægning af systemet

En trinvis omlægning, som beskrevet i dette afsnit, vil i reglen fastholde systemets overordnede opdeling. Når væsentlige dele af systembeskrivelsen er omlagt kan man, som led i

den normale vedligeholdelse, gennemføre en overordnet analyse af systemet. Man bør i den forbindelse udarbejde context-diagram for det totale system, og strukturere systemets hændelser i en logisk model.

Heller ikke i på dette tidspunkt er det absolut påkrævet at omskrive de resterende dele af PG'et. Man renummererer blot PG'et og indføjer rutine- og procedurebeskrivelserne i den nye struktur, hvor man startede med at gøre det modsatte.

Ved at følge ovennævnte strategi, burde omlægningen af selv store systemer kunne planlægges indenfor det løbende forvaltningsbudgets rammer.

DataCentralen