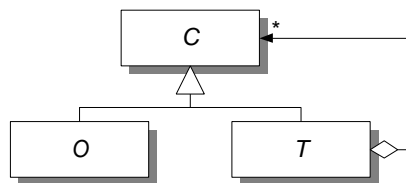


***OBJEKTORIENTERET ANALYSE OG
DESIGN AF UDVALGTE DELE AF
STADS
COT/4-03-V2.1***



Center for Objekt Teknologi

*Center for
Objekt Teknologi*

Revisionshistorie:	V1.0	13-02-98	Første version.
	V1.1	16-02.98	Første endelige version.
	V2.0	15-04-98	Anden endelige version med ny dokumentstandard
	V2.1	13-05-99	Endelig version efter COT-review

Forfatter(e): Johnny Olsson, WM-data
Kirsten Hjerrild Nielsen, WM-data
Kasper Østerbye, Aalborg Universitet
Allan R. Lassen, Rambøll

Status: Endeligt

Offentliggørelse: Offentligt

Sammendrag:

Dette dokumentet indeholder en objektorienteret analyse og design af udvalgte dele af det studieadministrative system STADS. Analysen omfatter de delsystemer, som i det eksisterende STADS kaldes Rammer og Uddannelsernes Struktur. Designet omhandler primært regelcheckereren.

© Copyright 1998 WM-data

Center for Objekt Teknologi (COT) er et projekt til forskning i, samt modning og indførelse af objektteknologi i danske virksomheder. Projektet er støttet af Center for IT-forskning og Erhvervsfremmestyrelsen.

Deltagere i projektet er:
Maersk Line, Maersk Training Center, Bang & Olufsen, WM-data, Rambøll, Danfoss, Systematic Software Engineering, Odense Stålskibsværft, A.P. Møller, Aarhus Universitet, Odense Universitet, Københavns Universitet, Dansk Teknologisk Institut og Dansk Maritimt Institut

Indholdsfortegnelse

1. INDLEDNING	4
2. ANALYSE AF UDVALGTE DELE AF STADS	5
2.1 Det nuværende STADS	5
2.2 Systemdefinition	7
2.3 Analysemetode	7
2.4 Problemområdet	9
2.4.1 Klynger	9
2.4.2 Struktur	9
2.4.3 Eksempel på en studieordning	14
2.4.4 Klasser	16
2.4.5 Hændelser	22
2.5 Anvendelsesområde	22
2.5.1 Funktioner	23
2.6 Grænseflade	23
3. DESIGN AF UDVALGTE DELE AF STADS	25
3.1 Regler	25
3.2 Dispensationer	25
3.3 Funktioner og check af regler	26
3.4 Funktioner	26
3.5 Oo-Regelcheckeren	27
3.5.1 Det effektive regelsæt	29
3.5.2 Kontrolleringsrelationen uden nedarvede regler	29
3.5.3 Kontrolleringsrelationen med nedarvede regler	29
4. KONKLUSION	32
4.1 Erfaringer	32
4.2 Videre arbejde	33
5. LITTERATURLISTE	34

1. Indledning

Denne rapport indeholder analyse og design af dele af et studieadministrativt edb-system for videregående uddannelsesinstitutioner. Analysen er inspireret af et eksisterende edb-system, STADS, som er i drift på en række af landets uddannelsessteder i en Oracle-version.

Rapporten er skrevet som et del af COT case 4 projektet. COT case 4 omhandler integration med ikke-objektorienterede systemer. Denne rapport er resultatet af et empirisk projekt foretaget hos WM-data med projektdeltagere fra Aalborg Universitet, Rambøll og WM-data.

Projektet har haft som formål:

- at opnå erfaring med objektorienteret analyse og design - herunder brug af UML,
- at identificere væsentlige problememner for det fortsatte COT case 4 arbejde,
- at etablere en case for et efterfølgende Oracle 8 projekt,
- at dokumentere erfaringerne i nærværende rapport.

Rapporten er opdelt i to dele. Første del indeholder en analyse af dele af problemområdet for det eksisterende STADS. De udvalgte dele svarer til delsystemerne Uddannelsernes Struktur (BE), Rammer (RA) og Regelchecker (RC) med fokus på kontrol og tilrettelæggelse af studierne struktur og de personlige studieføløb.

Anden del af rapporten indeholder et design af en del af det analyserede system. Hovedvægten er lagt på design af systemets funktioner og regelchecker. Designet er ikke komplet i metodemæssig forstand. Det primære i designet har været at etablere det nødvendige case-materiale for et efterfølgende studie af Oracle 8.

Den anvendte analyse er foretaget med den objektorienterede analysemetode beskrevet i (Mathiassen et al., 1997). Denne metode er anvendt, fordi dens notation anvender UML, som vi har ønsket at arbejde med, og fordi flere af projektets deltagere har haft kendskab til metoden og udviklerne af metoden.

Rapportens målgruppe er personer med kendskab enten til objektorienteret analysemetoder og/eller det studieadministrative system STADS. For personer med kendskab til objektorienterede analysemetoder giver rapporten vurderinger på anvendelse af objektorienteret metode. Folk med kendskab til STADS får ved at læse rapporten indblik i den begrebsverden objektorientering bygger på.

2. Analyse af udvalgte dele af STADS

Dette kapitel indeholder en objektorienteret analyse af dele af det studieadministrative system STADS. Analysen vedrører delsystemerne Uddannelsernes Struktur (BE), Rammer (RA) og Regelcheckeren (RC) med fokus på kontrol og tilrettelæggelse af studierne struktur og de personlige studieforløb. Analysen vedrører ikke alle detaljer af de involverede delsystemer. Analysen baserer sig dels på analysen for det eksisterende STADS (WM-data, 1997), dels på en ny analyse af det studieadministrative problemområde.

2.1 Det nuværende STADS

STADS-projektet blev startet i 1989. Formålet med projektet var og er at lave et administrativt system, der både kan bruges til at administrere studerende og uddannelser på landets videregående uddannelsesinstitutioner og til at indberette studenterårsværk til undervisningsministeriet. Da institutionerne er meget forskellige mht. administration og struktur af uddannelser, er systemet meget generelt. Det består bl.a. af en kerne af generelle regler for, hvordan studerende kan gennemgå et uddannelsesforløb. Reglerne kan tilpasses af den enkelte institution.

Systemet har været under udvikling siden 1993. De første dele af STADS gik i drift i begyndelsen af 1996.

STADS er udviklet på en Oracle-database (primo 1998 version 7.1). Skærbilleder er udviklet i Oracle Forms, udskrifter i Oracle Reports.

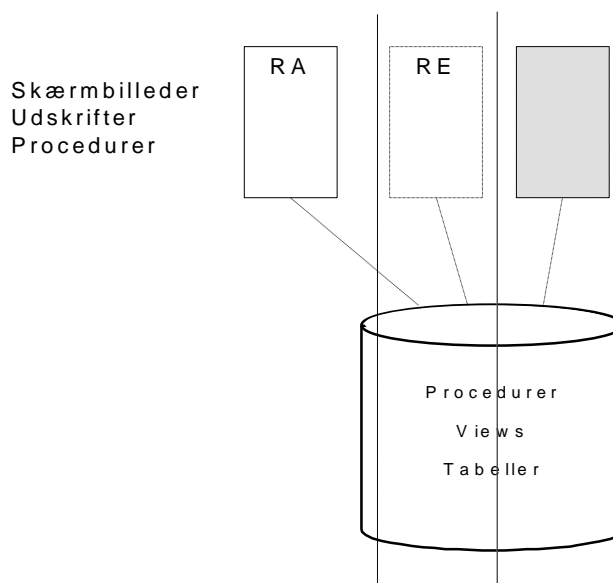
STADS er opdelt i en række delsystemer, der beskrives kort i det følgende.

- BE: Uddannelsernes Struktur (af brugerne kaldet US). Delsystemet rummer beskrivelser af bekendtgørelser, uddannelser, studieordninger og aktiviteter med regler for beståelse og gennemførelse af studieforløb.
- RA: Studerendes Rammer. Delsystemet rummer personoplysninger og oplysninger om studerendes tilknytning og status i forhold til uddannelserne. Delsystemet rummer desuden regelcheckeren, som kan kontrollere de studerende i forhold til reglerne på uddannelserne.
- TI: Tilmelding. Delsystemet rummer udbud af aktiviteter i de forskellige undervisningsperioder og eksamensterminer, samt de studerendes tilmeldinger til de udbudte aktiviteter.
- PL: Planlægning. Delsystemet rummer fordeling af tilmeldte studerende på undervisnings- og eksamenshold, samt planen for gennemførelse af undervisning og eksamen mht. brug af lærere, eksaminatorer, censorer og lokaler.
- RE: Resultater. Delsystemet rummer de resultater (evt. karakterer) som de studerende opnår ved eksamen (og undervisningsdeltagelse).
- AU: Åben Uddannelse. Delsystemet rummer oplysninger om udbud under åben uddannelse, samt oplysninger om åben uddannelsesstuderendes betalingsforhold.

Ligeledes foretager delsystemet integrationen med de videregående uddannelsers økonomistyringssystemers (ØSS) debitor modul vedr. studenterdebitorer.

- OP: Optagelse. Delsystemet rummer ansøgninger om studiepladser ved institutionen og varetager overførslen af disse til KOT (den koordinerede tilmelding - eksternt system) hvor udvalgte ansøgere tildeles studieplads ved institutionen. De optagne sendes over i delsystem Rammer, hvor de indskrives på en uddannelse.
- ST: Statistik. Delsystemet rummer udtræk af statistikoplysninger fra de øvrige delsystemer. Herfra kan institutionen generere statistikker dels til eget brug dels til indberetning til ministeriet. Desuden opsamles institutionens optjente STÅ (studenterårsværk), som udgør grundlaget for ministeriets tildeling af ressourcer til institutionen.
- AS: Autorisation. Delsystemet rummer definition af nødvendige roller for at kunne benytte systemet, samt systemets brugere og disses tilknytning til rollerne. Endvidere mulighed for at skræddersy menustrukturer til at afspejle logiske arbejdsområder.
- VY: Fælles værktøjskasse delsystem. Delsystemet rummer centrale begreber som perioder (semestre og terminer), hierarkisk ordnede administrative enheder på institutionen, fejlttekster, trimmeparametre og jobdefinitioner. Skærbilleder til bestilling og overvågning af batchjobs og udskrifter ligger i dette delsystem ligesom batchafvikleren, den proces, der afvikler alle jobs i systemet.

Desuden er der udviklet et system til håndtering af uddannelsesstøtte, kaldet US/STADS. Dette er udviklet efter samme koncept som de øvrige delsystemer i STADS, men kan installeres uafhængigt af STADS.



Figur 1. STADS's arkitektur

STADS-systemets arkitektur er vist på Figur 1. De stiplede linier viser den arkitektoniske opdeling i delsystemer. Den grå kasse dækker over alle de ikke viste delsystemer. Delsystemerne deler database, men tabeller, views og procedurer har tilhørsforhold til ét

delsystem. Kommunikation mellem delsystemerne foregår via snitviews og snitprocedurer. Views, som er sammensatte af data fra flere delsystemer, baserer sig på snitviews.

2.2 Systemdefinition

Systemet, som i dette dokument analyseres, er en del af det eksisterende STADS. Følgende systemdefinition baseret på BATOFF-kriteriet (Mathiassen et al., 1997) beskriver systemet.

Ved systemet forstås udvalgte dele af delsystemerne Uddannelsernes Struktur (BE), Rammer (RA) og Regelcheckeren (RC) i det eksisterende STADS med fokus på kontrol og tilrettelæggelse af studierne struktur og de personlige studieforløb. Systemet skal håndtere bekendtgørelser, studieordninger, uddannelser, regler, dispensationer, studieaktiviteter og studieforløb.

Brugere er eksisterende STADS-brugere og nye institutioner, som ikke skal have hele STADS-pakken.

Den teknologiske platform er en Oracle 8 database.

Analysen er et studieprojekt med det formål:

- at lære objektorienteret modellering,
- at bibringe en fælles forståelse af det studieadministrative problemområde,
- at foretage et design af den centrale del af systemet, regelcheckeren,
- at udgøre et forarbejde til eksperimentelle undersøgelser af Oracle 8 og integration af objektorienterede med ikke-objektorienterede systemer/migrering.

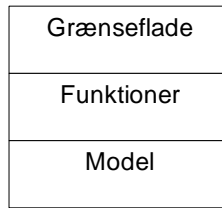
2.3 Analysemetode

Analysen af STADS anvender en objektorienteret analysemetode beskrevet i (Mathiassen et al., 1997). Ifølge metoden er formålet med analyse at:

- opnå en forståelse af et kommende edb-systems virkefelt,
- opstille krav til edb-systemet,
- at designe et edb-system.

Metoden er åben, og vi har i analysen af udvalgte dele af STADS valgt at frit fortolke metoden, hvor det har vist sig hensigtsmæssigt for at opnå forståelse for problemområdet og været relevant for det efterfølgende design. Metoden anvender notationsformen Unified Modeling Language, UML, som er en forening af de mest udbredte notationer indenfor objektorienteret modellering.

Idet analyse ses i forhold til et edb-system, er det nødvendigt at have en forståelse af, hvordan et edb-system arkitektur ser ud. Metodens forståelse af et systems arkitektur er vist på Figur 2.

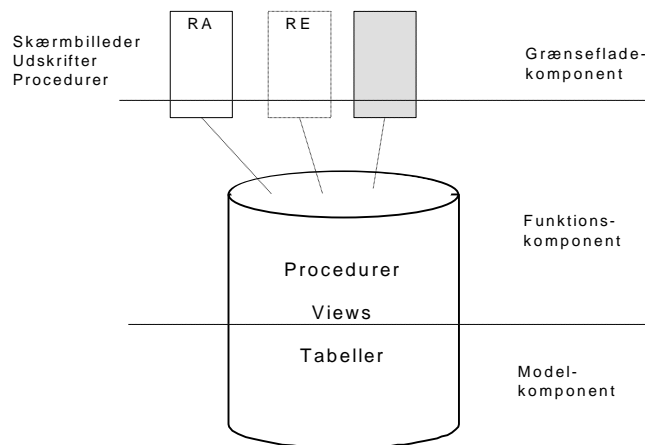


Figur 2. Arkitektur af edb-system opdelt i komponenter

Modelkomponenten indeholder en model af edb-systemets problemområde, dvs. en beskrivelse af klasser, struktur og dynamik for det område systemet skal administrere. Funktionskomponenten indeholder de faciliteter, som brugere anvender på systemet, og som opdaterer modelkomponenten. Grænsefladekomponenten binder systemet sammen med brugere i form af billeder, udskrifter og funktionalitet, der sætter brugeren i stand til at opdatere modelkomponenten. Komponenterne indeholder desuden en grænseflade til andre systemer.

Strukturen af et systems modelkomponent er meget stabil. Ændringer af systemet bør så vidt muligt ikke berøre modelkomponenten. Funktionskomponenten er mindre stabil, da der over tid kan ske ændringer i den funktionalitet, som brugere skal have stillet til rådighed. Grænsefladen er den mindst stabile komponent bl.a. fordi brugergrænsefladeteknologien er foranderlig, og fordi systemet skal kunne håndtere, at kommunikationen med andre systemer ændrer sig.

Opdelingen i model-, funktions- og grænsefladekomponent er anvendt i denne rapport's analyse og design af STADS. Denne opdeling går på tværs af den eksisterende opdeling af STADS i delsystemer. På Figur 3 er vist, hvordan denne opdeling ser ud for det eksisterende STADS. Det præcise snit mellem komponenterne afgøres i forbindelse med design.



Figur 3. Opdeling af STADS i model-, funktions- og grænsefladekomponenter

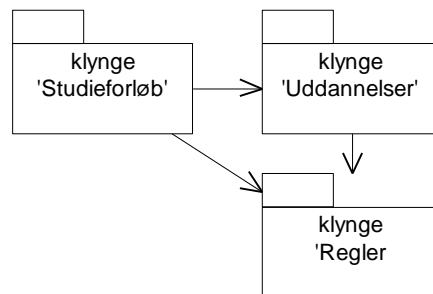
Den følgende analyse er beskrevet med det samme systemfokus som vist på Figur 3. Der foretages først en analyse af problemområdet, dvs. edb-systemets modelkomponent. Dernæst af anvendelsesområdet, dvs. funktionskomponenten. Endelig analyseres grænsefladen. Notationen i analysen er forsøgt tilnærmet UML (Rational, 1997).

2.4 Problemområdet

I det følgende beskrives systemets problemområde. Da systemet skal være generisk og med mange tilpasningsmuligheder for den enkelte institution, er problemområdet meget generelt.

2.4.1 Klynger

Overordnet består systemet af beskrivelser af uddannelser, studerendes studieforløb og regler for studieforløb og uddannelser. Systemet er inddelt i tre klynger som vist på Figur 4. Strukturen af de tre klynger og relationerne mellem dem er beskrevet i det følgende. De stiplede linier viser afhængigheder mellem klyngerne. Klyngen 'Studieforløb' har et afhængighedsforhold til 'Uddannelser'.



Figur 4. Klynger i systemet

2.4.2 Struktur

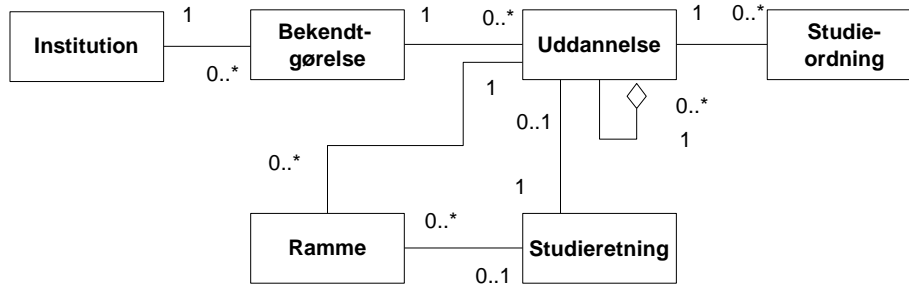
Klyngen 'Uddannelse' indeholder beskrivelser af en institutions uddannelser. En uddannelse er defineret ved en bekendtgørelse. Et eksempel er 'Bekendtgørelse om de humanistiske uddannelser på universiteterne'. En uddannelse kan bestå af underuddannelser, hvilket er modelleret ved at en uddannelse aggregerer sig selv. Et eksempel på en uddannelsesstruktur er cand.mag. uddannelsen, som består af en grunduddannelse og en overbygningsuddannelse. Institution er den institution systemet fungerer på.

En uddannelse er udbudt på et givet tidspunkt ved at være associeret med Ramme og Studieretning. For cand.mag. uddannelsen er 'dansk' og 'fransk' mulige studieretninger. En ramme udtrykker et udbud af en uddannelse på et givet tidspunkt med en angivet studieretning.

Forholdet mellem Uddannelse og Ramme kan beskrives som en mønster bestående af et Item og en Context¹, som vist på Figur 5.

Bemærk at der ikke er angivet navne på associationerne på figuren og de efterfølgende figurer. Det skyldes, at det begrebsapparat, der findes for studieadministration ikke lægger op til en naturlig navngivning af relationerne.

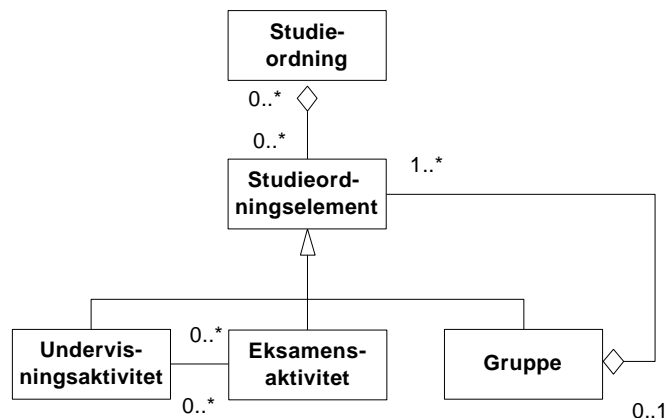
¹ 'Item Context' mønstret kan sammenlignes med 'Item Description' (Coad, 1992). Description beskriver et element, Context beskriver i hvilken kontekst, Item har sit virke. Eksemplet illustrerer, at en uddannelse er noget konstant, hvor ramme udtrykker tidspunktet for, hvornår uddannelsen kan følges af studerende. Mønstret er identificeret gentagne gange i forbindelse med STADS-analysen.



Figur 5. En uddannelses relationer

En uddannelse er konkretiseret i et antal studieordninger. En studieordning består af grupper af undervisnings- og eksamensaktiviteter. En studieordnings grupper kan bestå af andre grupper. Det rekursive struktur af grupper og aktiviteter er modelleret ved introduktion af den abstrakte klasse Studieordningselement og ved at grupper er aggregerede objekter fra subklasser til Studieordningselement, som vist på Figur 6. Strukturen svarer til mønstret Composite (Gamma et al., 1994). Undervisnings-, eksamensaktiviteter og grupper kan indgå i flere studieordninger. Strukturen er ikke en træstruktur, men en orienteret acyklisk graf.

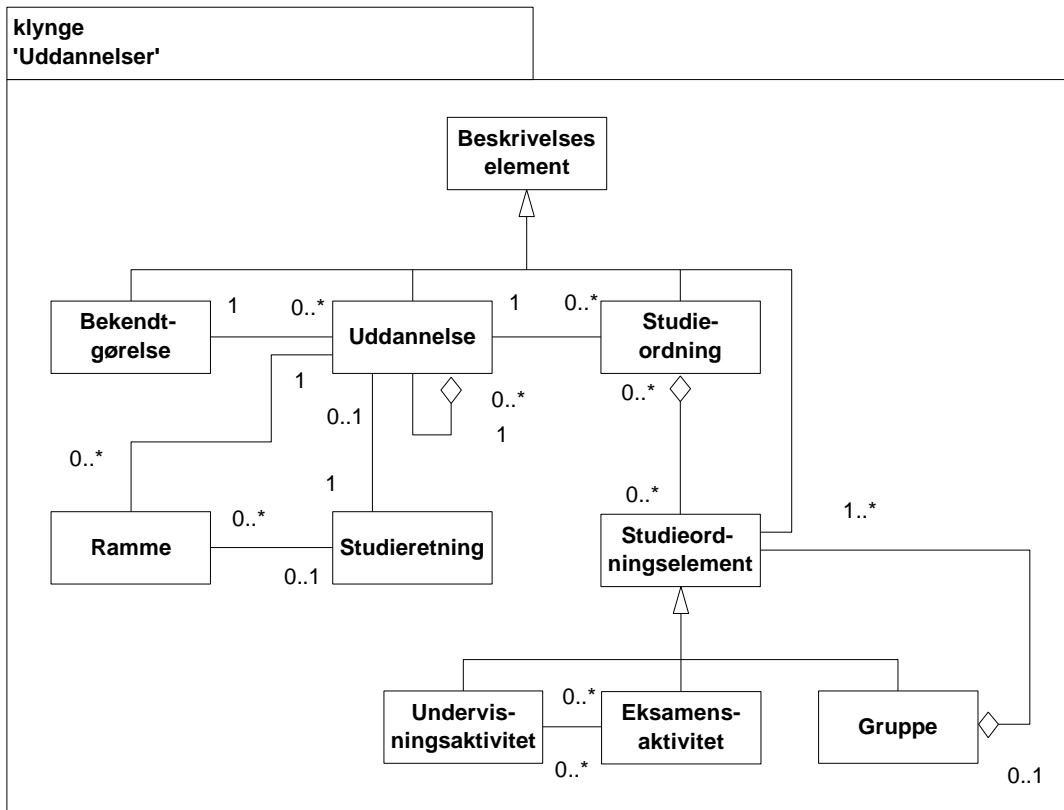
Et eksempel på en studieordning er Studieordning for bachelor uddannelsen i datalogi. Studieordningens grupper indeholder de enkelte semestres undervisnings- eksamensaktiviteter.



Figur 6. Struktur af en studieordning

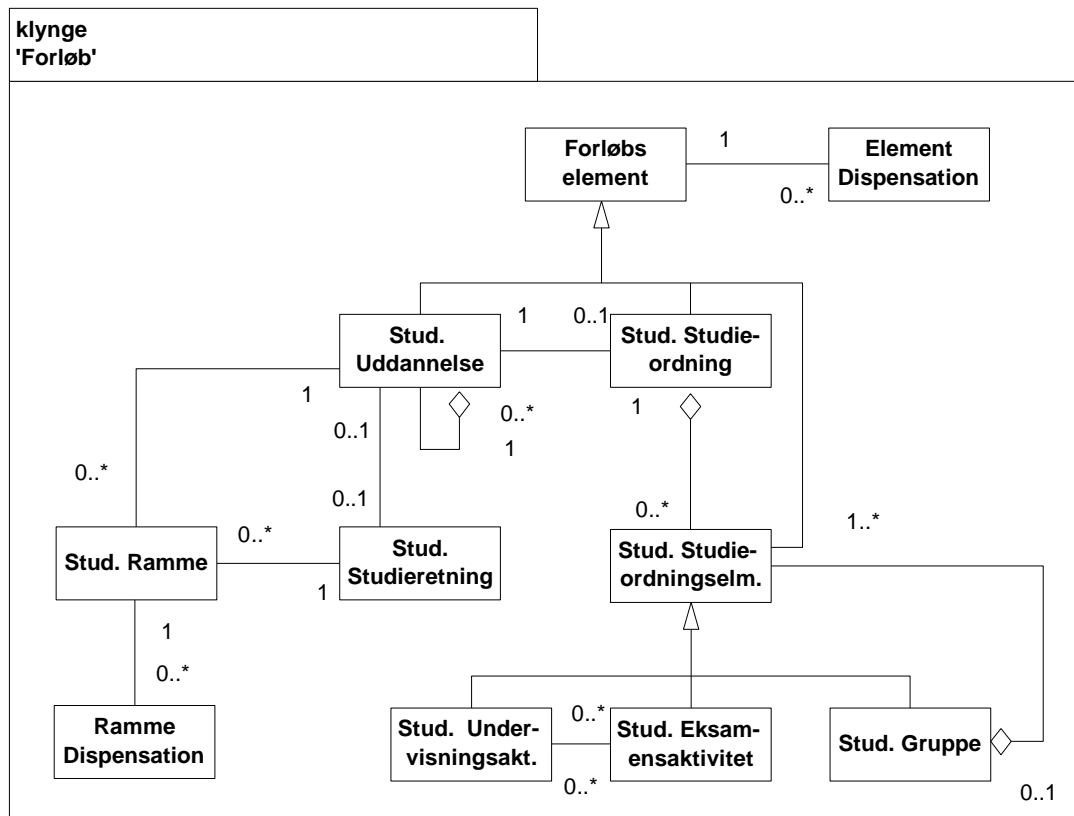
Idet en undervisningsaktivitet kan lede frem til en given eksamensaktivitet, er undervisnings- og eksamensaktiviteter associerede. En undervisningsaktivitet indgår ikke nødvendigvis i en studieordning, men kan være relateret ved at føre frem til en eksamensaktivitet.

På Figur 7 er vist den samlede struktur af klyngen 'Uddannelser'. Alle uddannelseselementer er generaliserede i den abstrakte klasse Beskrivelselement.



Figur 7. Struktur af klyngen 'Uddannelser'

Klyngen 'Studieforløb' er vist på Figur 8. Strukturen for klyngen 'Forløb'. Strukturen er parallel med strukturen i 'Uddannelser'. Hver klasse under beskrivelseselement - undtaget bekendtgørelse og institution - findes som element i et studieforløb. Der kan dispenseres for et uddannelseselement i et normalt studieforløb. Ramme Dispensation er klassen af dispensationer, der kan gives for en studerendes ramme. Endvidere er der en forskel i kardinaliteten på relationerne, således at et Forløbselement kun kan forekomme én gang i et uddannelsesforløb. Dvs. at strukturen er en træstruktur, hvor strukturen i klyngen 'Uddannelser' er en graf. Element Dispensation er dispensationer, der gives enkelte elementer i et studieforløb.

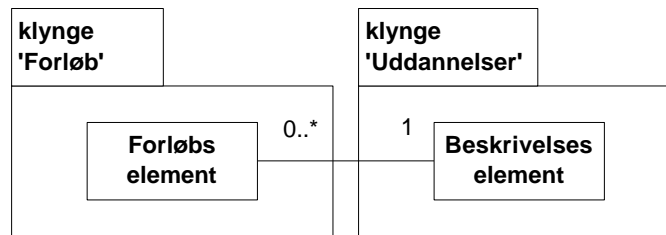


Figur 8. Strukturen for klyngen 'Forløb'

Klyngerne 'Uddannelser' og 'Studieforløb' er forbundne ved at klasserne Forløbselement og Beskrivelseselement er associerede. Associationen mellem Uddannelseselement og Studerendes Forløbselement er i (Coad, 1992) beskrevet som et 'Item Description' mønster. Beskrivelseselementet er en beskrivelse af den studerendes uddannelseselement. Klassen Studerendes Ramme er ligeledes associeret med Ramme.

Klasser under henholdsvis Forløbselement og Uddannelseselement er logisk set parvist associerede.

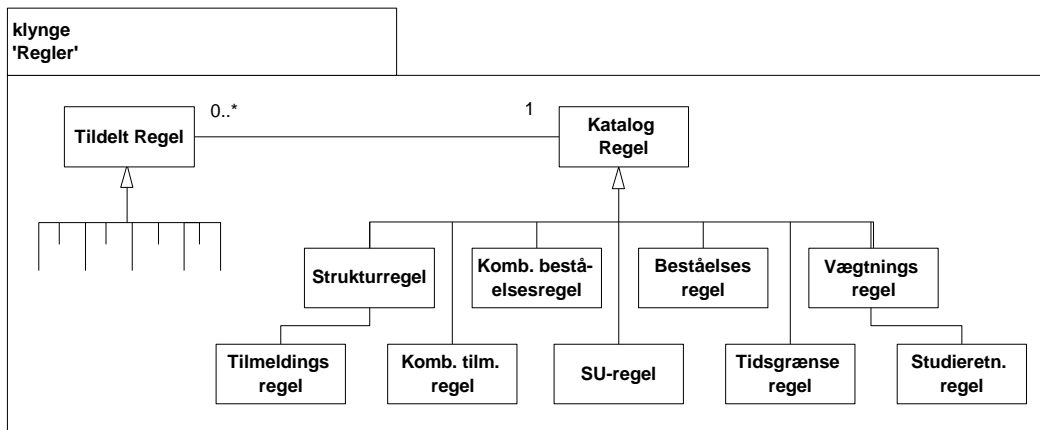
Studerendes Studieordning er associeret med Studieordning m.v. Af hensyn til overskuelighed viser Figur 9 kun associationen mellem superklasserne.



Figur 9. Forholdet mellem 'Studieforløb' og 'Uddannelser'

Klyngen 'Regler' indeholder klasser af katalogregler og klasser af tildelte regler, som vist på Figur 10. En katalogregel er en generisk beskrivelse af forhold for en studerendes studieforløb. Det kan for eksempel være krav om, at beståelse af en aktivitet kræver en given karakter. En tildelt regel er forekomsten af en regel gældende for en specifik aktivitet. Det kan eksempelvis være, at 'Matematik 1' skal bestås med mindst 7 i karakter.

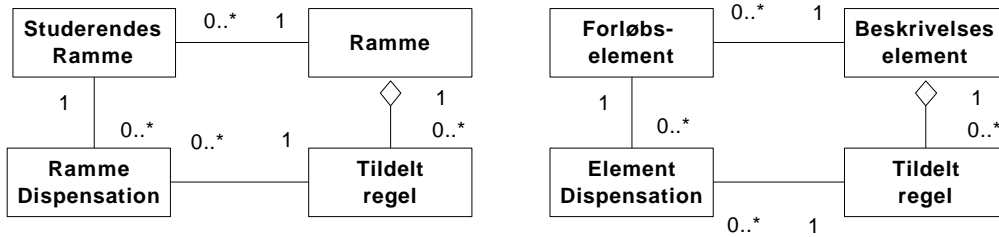
Under klassen Tildelt Regel findes klasser med tildelte regler af samme typer som under klassen Katalog Regel. Disse er parvist associerede, hvilket ikke er vist på figuren.



Figur 10. Struktur af 'Regler'

Associationen mellem Tildelt regel og Katalogregel er at forstå som et 'Item Description' mønster.

Forholdet mellem klyngen 'Regler' og klyngerne 'Uddannelser' og 'Studieforløb' består i at en række regler kan tildeles et uddannelseselement eller en ramme, hvilket er modelleret i en aggregeringsstruktur og ved at en dispensation i et studieforløb altid vedrører dispensation fra en tildelt regel. Relationen er vist på Figur 11.

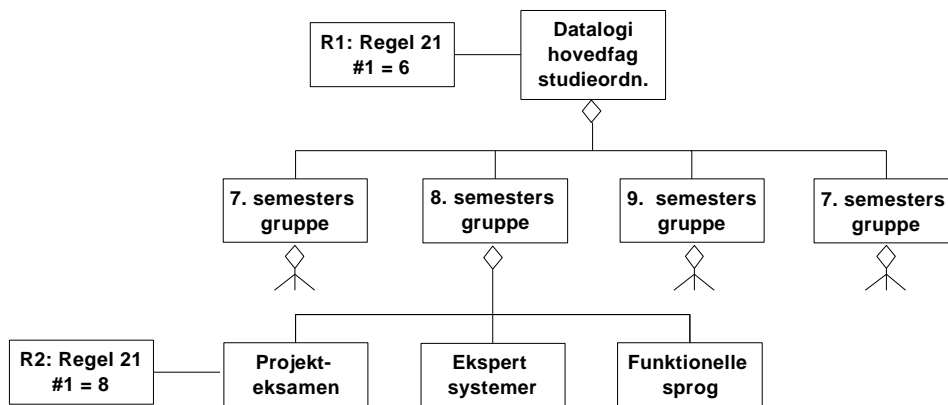


Figur 11. Forholdet mellem klyngerne 'Uddannelser ' og 'Regler' og 'Forløb'

2.4.3 Eksempel på en studieordning

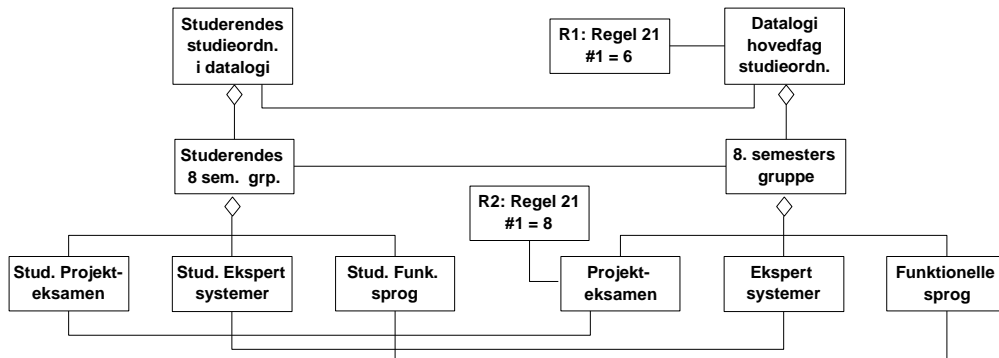
I dette afsnit gives et eksempel på modellering af en studieordning og en studerendes forløb af denne studieordning.

I eksemplet ses kun på den del af beskrivelsen, som hører under selve studieordningen. I et virkeligt eksempel skal også inddrages uddannelse, ramme, studieretning, bekendtgørelse og institution, men for at give et indblik i strukturen er det nok kun at koncentrere sig om studieordningen, som ifølge afgrænsningen ikke er påsat alle regler. Strukturen af en beskrivelse følger den viste på Figur 6. Kun aktiviteter for 8. semesters gruppe er vist.



Figur 12. Studieordningen for hovedfag i datalogi.

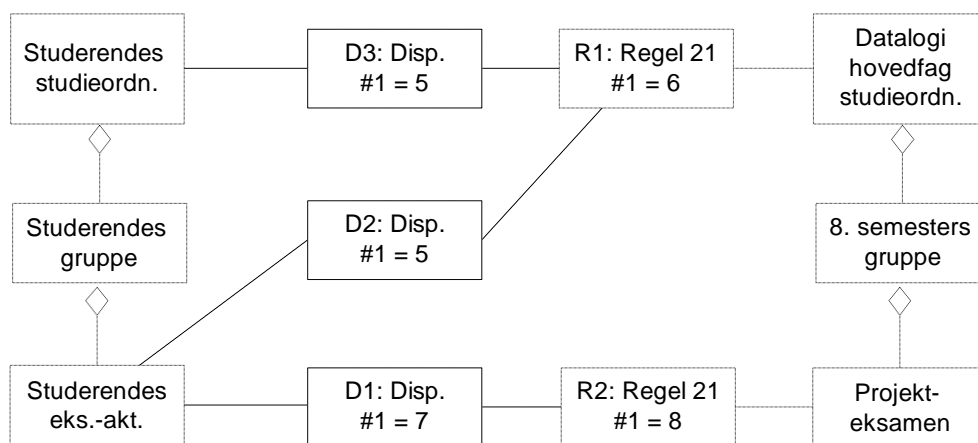
På Figur 12 er vist et udsnit af strukturen af studieordningen for hovedfag i datalogi. Under gruppen '8. semester' er 3 eksamensaktiviteter: 'Ekspertsystemer', 'Funktionelle Sprog' og projektexamen. På studieordningsniveau angiver regel 21, at for at bestå en forløbs-element, skal den studerende opnå et karakterresultat på mindst 6. For projektet er tilknyttet en regel 21, der udtrykker, at til projektexamen skal opnås et resultat på mindst 8 for at bestå prøven. Denne regel overskygger reglen på studieordningen. Studieordningens øvrige aktiviteter ses bort fra i det følgende.



Figur 13. En studerendes forløbselementer associeres med beskrivelseselementer.

På Figur 13 er vist strukturen af en studerendes forløb af studieordningen for hovedfag i datalogi. Ethvert forløbselement associerer sig til et beskrivelseselement. Forløbselementer vist til venstre, mens beskrivelseselementer er til højre. For eksaminerne i ekspertssystemer og funktionelle sprog gælder reglen R1. Dette udtrykkes også ved, at R1 nedarves til studieordningens elementer. For projektet gælder reglen R2.

Gives der dispensationer fra en regel opstår der forskellige situationer, som vist på Figur 14. De stiplede kasser viser 3 forskellige situationer med dispensationer. Eksamensaktiviteterne Ekspertsystemer og Funktionelle Sprog er ikke indtegnet på figuren.



Figur 14. Dispensation fra regel 21 i 3 forskellige situationer

Situationen med dispensation D1 betyder, at den studerende gives dispensation fra kravet om at projektet skal bestås med mindst 8. Karakteren 7 er nok for at bestå projekteksamen.

Dispensation D2 betyder, at der gives dispensation fra reglen om beståelse af med mindst 6 i karakter. For projekteksamen betyder det imidlertid intet, da R1 overskygges af R2.

For den studerende kræves således stadig 8 i karakter for beståelse af projekteksamen.

Situationen med dispensation D3 betyder, at alle elementer under studieordningen kan bestås med karakteren 5. Undtaget er projekteksamen, da den regel, der dispenseres for (R1), ikke gælder ved den studerendes projekteksamen, hvor R2 stadig er gældende. For

Ekspertsystemer og funktionelle sprog, der ikke er indtegnet på figuren, betyder dispensationen D3, at for beståelse af disse kræves mindst karakteren 5.

Princippet om at regler tilknyttet et beskrivelseselement på et højere niveau, og ditto med dispensationer på forløbselementer, har virkning på elementer længere nede i strukturen kaldes nedarvning². Betydningen af virkefeltet for nedarvede regler og dispensationer er klar, men kompleks. I dette afsnit vil princippet kun blive illustreret ved ovenstående eksempel. I designkapitlet gennemgås en metode til at finde hvilke regler og dispensationer, der for et forløbselement, er de relevante af på et givent tidspunkt.

2.4.4 Klasser

Dette afsnit beskriver de enkelte klasser i systemet. Beskrivelserne af superklasserne Beskrivelseselement og Forløbselement indeholder tilstandsdiagrammer, der også gælder for underklasserne.

2.4.4.1 Katalogregel

Klassen sammenfatter mængden af (abstrakte) regler, som kan anvendes på de forskellige beskrivelseselementer, som en uddannelse består af. Den abstrakte del af en katalogregel er infoværdier, som er værdidomæner, der specificeres ved tildeling af regler.

Der findes en række forskellige typer regler, som virker på forskellig måde. Der er følgende overordnede regeltyper:

- Strukturregler.
- Beståelses regler kombineret med struktur regler.
- Beståelses regler.
- Vægtningsregler (er et aspekt af beståelse).
- Tilmeldingsregler.
- Tilmeldingsregler kombineret med struktur regler.
- SU-regler.
- Tidsgrænseregler.
- Studieretningsvalgs regler.
- Reglernes semantik, som er vidt forskellig fra regel til regel, er ikke beskrevet i dette dokument. Et eksempel på en regel er 'Karakteren skal være mindst #1'. Symbolet #1 betyder, at for en given anvendelse af reglen er #1 den karakter, som skal opnås for beståelse.
- Attributter: Nummer, tekst og virkefelt (lokal eller nedarvet: en regel er lokal, hvis den kun gælder for det forløbselement, som den er direkte tilknyttet. En nedarvet regel gælder også for de forløbselementer, som hører under det forløbselement, hvor reglen er tilknyttet).

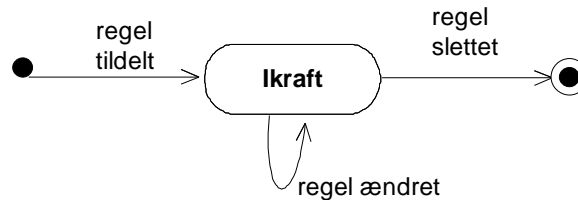
2.4.4.2 Tildelt regel

Klassen modellerer de regler, som tildeles beskrivelseselementer og rammer. En tildelt regel eksemplificerer med udfyldelse af infoværdier en katalogregel. Eksempelvis kan reglen fra

² Nedarvning i denne forbindelse må ikke forveksles med nedarvning kendt som specialisering i objektorienterede sprog.

sidste afsnit blive tildelt med værdien 6, således at reglen bliver 'Karakteren skal være mindst 6'.

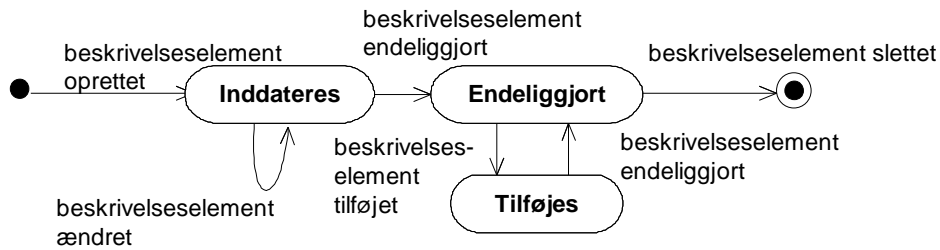
Attributter: Infoværdier.



Figur 15. Tilstandsdiagram for Tildelt Regel

2.4.4.3 Beskrivelselement

Klassen modellerer en generel beskrivelse af et uddannelseselement. Et element kan være sammensat af og associeret med andre uddannelseselementer. En sådan struktur af uddannelseselementer benævnes en uddannelsesstruktur.



Figur 16. Tilstandsdiagram for Beskrivelselement

På Figur 16 er vist tilstandsdiagram for Beskrivelselement. Specialiseringer af denne klasse har samme diagram. Tilstanden inddateres er den initiale tilstand, som et element er i indtil det er endeliggjort. Efter endeliggørelse kan et element ændres (tilføjes) og det vil forblive i den tilstand indtil det igen endeliggøres. Termerne inddateres, endeliggjort og tilføjes blev oprindeligt indført i STADS.

De er nu en del af brugernes forståelse, så de bruges fortsat, trods de sproglige misvisninger ordene har.

Attributter: Kode, navn, belastning (antal points/årsværk elementet vægter).

2.4.4.4 Institution

Denne klasse har til formål at give mulighed for at tilknytte regler, der gælder for hele institutionen.

Attributter: Institutionens navn.

2.4.4.5 Bekendtgørelse

Klassen dækker over bekendtgørelser, som definerer sammenhængen mellem uddannelsesdelene og de overordnede beståelsesregler for hele bekendtgørelsen eller uddannelsesdelene.

Attributter: Ikrafttrædelsesdato, bekendtgørelsesnummer, titel, dato og år.

2.4.4.6 Uddannelse

Klassen beskriver mængden af uddannelser. En uddannelse kan eksempelvis være kandidatuddannelsen i datalogi. For en uddannelse findes en eller flere studieordninger.

Attributter: Ikrafttrædelsesdato, indgangshjemmel.

2.4.4.7 Studieordning

En studieordning er en institutions måde at konkretisere ministeriets bekendtgørelse om en bestemt uddannelsesdel med en bestemt studieretning. Studieordningen er udformet som en struktur af grupper og aktiviteter.

Attributter: Ikrafttrædelsesdato, indgangshjemmel.

2.4.4.8 Studieordningselement

Klassen er en generalisering af Gruppe, Uddannelses- og Eksamensaktivitet. Klassen er introduceret for at modellere den rekursive struktur af grupper med aktiviteter.

Attributter: ingen særlige.

2.4.4.9 Gruppe

En gruppe er en samling af aktiviteter (undervisnings- og/eller eksamensaktiviteter) og evt. andre grupper. En gruppe benyttes til at samle nogle aktiviteter i en studieordning, så der kan formuleres regler om beståelse eller studieførløb knyttet til gruppen, eller som et redskab i studieordningen til at opnå logiske samlinger af eksamens og undervisningsaktiviteter. En logisk samling kan for eksempel være et semester.

Attributter: ingen særlige.

2.4.4.10 Undervisningsaktivitet

En undervisningsaktivitet er en enhed til planlægning af institutionens undervisning. En undervisningsaktivitet kan indgå i en studieordning, hvis der er tale om obligatorisk undervisning, men som regel er undervisningsaktiviteterne udenfor studieordningerne.

Undervisningsaktiviteter kan være knyttet til eksamensaktiviteter, som ligger i studieordningerne, forstået på den måde, at studerende, der følger undervisningsaktiviteten som regel efterfølgende skal til eksamen i eksamensaktiviteten (eller rettere i nogle prøver knyttet til eksamensaktiviteten).

Typisk ligger undervisningsaktiviteter udenfor studieordningen med tilknyttet eksamensaktivitet.

En undervisningsaktivitet kan være en del af en uddannelse ved at indgå i en gruppe eller ved at indgå direkte i en studieordning.

Attributter: teori/ praktik.

2.4.4.11 Eksamensaktivitet

En eksamensaktivitet er den enhed, hvortil der knyttes en karakter for en studerende efter eksamen. Eksamensaktiviteten har en belastning, som definerer, hvor mange årsværk, beståelse af aktiviteten kan udløses pr. studerende. Eksamensaktiviteter indgår typisk i studieordninger og er i føromtalte graf forsynet med regler for, hvornår aktiviteten skal betragtes som bestået.

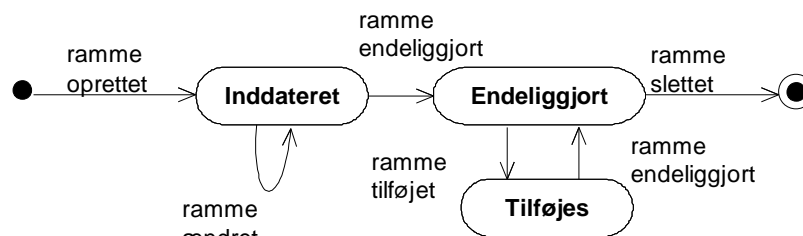
Attributter: teori/praktik.

2.4.4.12 Ramme

En ramme er et perspektiv på konteksten for en uddannelse, dvs. et udbud af en uddannelse på et givent tidspunkt. Rammen angiver studieretning og tidspunkt for, hvornår uddannelsen kan følges. Et eksempel på en ramme er uddannelsen i datalogi, som den er udbudt pr. 1. september 1997.

En ramme er et sæt retningslinier for, hvilke studieordninger, der kan tildeles studerende, der følger en bestemt uddannelse med en bestemt studieretning. Retningslinierne er formuleret som regler for valg af studieretninger på de enkelte dele af uddannelsen.

Tilstandsdiagram for klassen Ramme er vist på Figur 17.



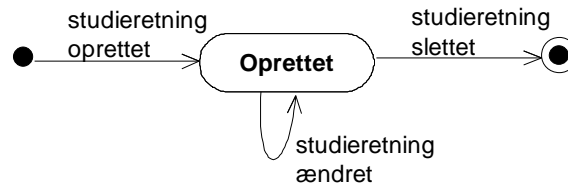
Figur 17. Tilstandsdiagram for Ramme

Attributter: Ikrafttrædelsesdato, navn, kode.

2.4.4.13 Studieretning

En studieretning er et valg af faglig retning, der gør en uddannelsesdel mere specifik. F.eks. 'dansk' eller 'fransk' som studieretning på en cand.mag. uddannelse.

Attributter: Navn, kode.



Figur 18. Tilstandsdiagram for Studieretning

2.4.4.14 Studerendes ramme

Strukturen af en ramme er givet ved beskrivelsen af den studerendes ramme i klassen Ramme. En studerendes ramme udtrykker vedkommendes studieforløb.

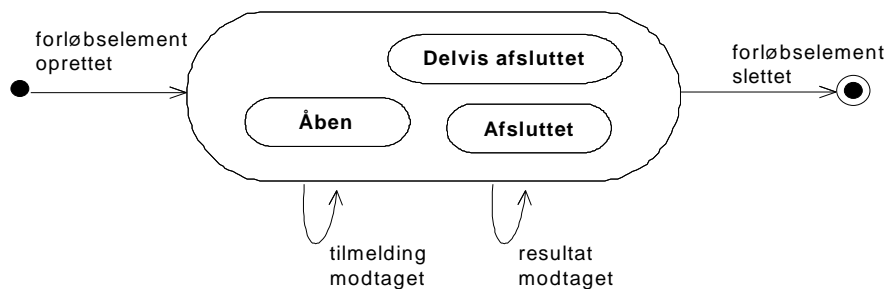
Attributter: oprettelsesdato, indskrivningsmåde.

2.4.4.15 Studerendes forløbselement

Klassen er en generalisering af de forskellige typer elementer, en studerendes uddannelse består af. Hver type af studerendes uddannelseselementer er en pendant til en type af uddannelseselementer. Studerendes uddannelseselementer har alle samme tilstandsdiagram. Et element kan være aggregeret eller associeret af en række andre elementer, som beskrevet for klasserne vedrørende uddannelsernes struktur.

Tilstandsdiagram for Forløbselement er vist på Figur 19. Et forløbselement kan skifte mellem tre tilstande forårsaget af de to hændelser tilmelding modtaget og resultat modtaget. Afgørelsen af hvilken tilstand et forløbselement er i, foretages i forbindelse med check af de tilknyttede regler. Det er således en meget kompleks beregning, som ligger bag ved et forløbselements tilstand. Beregningen foretages ved check af regler.

Det er ligeledes i forbindelse med check af regler, at der oprettes og nedlægges forløbselementer afhængig af, hvad den studerende tilmelder sig og består.



Figur 19. Tilstandsdiagram for Forløbselement

Attributter: Fortolket resultat, karakter, optjent STÅ (STudenter Årsværk: enhed til afregning af tilskud pr. student mellem institutioner og undervisningsministerium).

2.4.4.16 Studerendes uddannelse

En studerende kan følge en uddannelse, som beskrevet i klassen Uddannelse.

2.4.4.17 Studerendes studieordning

Klassen omfattende mængden af studerendes studieordninger. En studerende kan have flere studieordninger tilknyttet sig.

Attributter: se studerendes gruppe.

2.4.4.18 Studerendes studieordningselement

En generalisering af studerendes gruppe og aktiviteter.

2.4.4.19 Studerendes gruppe

En gruppe af aktiviteter, som en studerende kan/må følge.

Attributter: gennemsnitskarakter, obligatorisk/valgfri at bestå gruppens aktiviteter.

2.4.4.20 Studerende undervisningsaktivitet

Klasse over mængden af studerendes undervisningsaktiviteter.

Attributter: undervisningsform.

2.4.4.21 Studerendes eksamensaktivitet

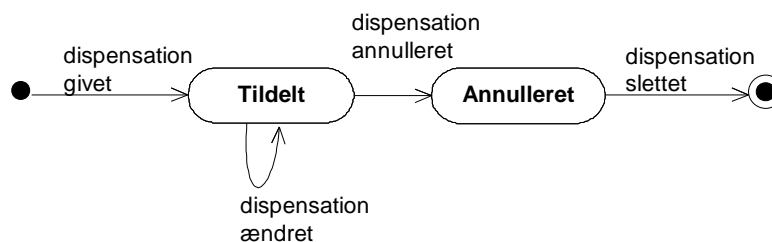
Klasse dækkende over mængden af studerendes eksamensaktiviteter.

Attributter: ingen særlige.

2.4.4.22 Element Dispensation

Klassen sammenfatter mængden af dispensationer fra regler, en studerende kan have på et forløbselement. En dispensation gælder for én regel for en studerendes forløbselement. Når en dispensation gives, associeres den med et forløbselement og den regel i uddannelsesbeskrivelsen, som der dispenseres fra. Tilstandsdiagram for Element Dispensation er vist på Figur 20. Der kan gives flere dispensationer for en regel, men kun én dispensation ad gangen er i virkning, eksempelvis begrænset ved en tidsperiode.

Attributter: periode for ikrafttrædelse.



Figur 20. Tilstandsdiagram for Element Dispensation

2.4.4.23 Ramme Dispensation

Klassen sammenfatter mængden af dispensationer fra regler, en studerende kan have på en ramme. Bortset fra at dispensationer vedrører rammer, er klassen meget lig med klassen Element Dispensation.

Attributter: periode for ikrafttrædelse.

2.4.5 Hændelser

Følgende liste indeholder mulige hændelser i problemområdet. Hændelser, der skaber relationer mellem uddannelseselementer og forløbselementer, er ikke medtaget i listen.

Hændelserne er relevante for en komplet analyse af STADS' problemområde, men for formålet med denne rapport har det ikke været nødvendigt at beskrive disse hændelser.

Hændelser vedrørende beskrivelser af uddannelser:

- Tildelt regel tildelt.
- Tildelt regel ændret.
- Tildelt regel slettet.
- Beskrivelselement oprettet.
- Beskrivelselement tilføjet.
- Beskrivelselement endeliggjort.
- Beskrivelselement slettet.
- Studieretning oprettet.
- Studieretning ændret.
- Studieretning slettet.
- Ramme oprettet.
- Ramme ændret.
- Ramme endeliggjort.
- Ramme slettet.

Hændelser vedrørende studerendes uddannelsesforløb:

- Tilmelding modtaget.
- Resultat modtaget.
- Forløbselement oprettet.
- Forløbselement slettet.
- Ramme/Element Dispensation givet.
- Ramme/Element Dispensation ændret.
- Ramme/Element Dispensation annulleret.
- Ramme/Element Dispensation slettet.

2.5 Anvendelsesområde

I det følgende beskrives dele af funktionsområdet. Vægten lægges på beskrivelse af systemets funktioner med henblik på at disse skal uddybes i design kapitlet. Flere af funktionerne foretager - udover det beskrevne - oprettelse, omstrukturering og nedlæggelse af forløbselementer. I dette dokument vil denne del af funktionaliteten ikke blive nærmere beskrevet.

2.5.1 Funktioner

Systemet skal understøtte de funktioner, som fremgår af følgende liste:

Check Konsistens: Check af overensstemmelse mellem regler i en studieordning.

Check Undervisningstilmelding: Check af en eller flere undervisningstilmeldinger for en studerende.

Registrér Eksamenstilmelding: Registrerer tilmeldinger til en eller flere eksamensaktiviteter for en studerende.

Registrér Resultat: Registrerer resultater for en eller flere eksamensaktiviteter for en studerende.

Registrér Resultat Før Dato: Som Registrér Resultat, men ser kun efter resultater med bedømmelsesdato før Før Dato. Det har betydning i forbindelse med konvertering og ajourføring.

Slet Tilmelding til Aktivitet: Fjerner de valgfri aktiviteter fra den studerende ramme, der har tilmeldinger i en given periode hhv. termin/eksamenstype. Elementerne fjernes kun, hvis der ikke er tilmeldinger til dem i en anden periode/termin, og hvis de ikke har et resultat eller er meritoverført. Ved fjernelse af undervisningsaktiviteter fjernes også - med tilsvarende forbehold - eventuelle eksamensaktiviteter, som undervisningsaktiviteten fører frem til.

Funktionerne kendetegnes ved at de gør brug af de regler, som er tilknyttet forløbselementerne. Det er værd at bemærke, at funktionaliteten i høj grad bygger på de semantiske betydninger af regler, som ikke er beskrevet i denne analyse.

2.6 Grænseflade

For denne analyse er det ikke relevant at forholde sig til brugergrænsefladen.

Grænsefladen til andre delsystemer i STADS er defineret ved (relationelle) snitviews og snitprocedurer til en Oracle 7/8 database. Systemdefinitionen dækker ikke over det samlede Rammer og Uddannelsernes Struktur i det eksisterende STADS. Persondata (studerendes adresser, adgangsgrundlag, meritter m.v.) antages tilgængelige ligesom data fra andre delsystemer, dvs. en relationel systemgrænseflade.

Edb-systemet skal tilbyde:

- Snitviews til uddannelsesrammer og studieordninger til indskrivning.
- Snitviews og referencetabeller til endelige undervisningsaktiviteter og eksamensaktiviteter inkl. karakterer.
- Snitprocedure til at gennemføre check af regler ved tilmeldinger og melde tilbage om evt. regelbrud.
- Snitprocedure til at registrere resultater i studerendes rammer m.h.p. at vurdere beståelse i forhold til regelgrundlaget.

- Snitviews og referencetabeller til endelige eksamensaktiviteter og undervisningsformer.

Edb-systemet skal tilbydes følgende views og procedurer:

- Snitviews til studerendes persondata - herunder bl.a. adresse og meritter.
- Snitviews og referencetabeller til semesterperioder, administrative enheder for institutionen.
- Snitview over gymnasiale niveauer og adgangseksaminer.
- Snitviews til resultater benyttes ved check af regler. Der refereres desuden via referencetabel til det gældende resultat fra forløbselementet.
- Snitviews over personoplysninger i AU, samt snitprocedurer til vedligehold af og oplysning om betalingsforhold for åben uddannelsesstuderende
- Snitprocedure til overførsel af en STÅ-transaktion. F.eks. ved beståelse eller efter annullering af resultat, der tidligere har været afleveret til ST som bestået.

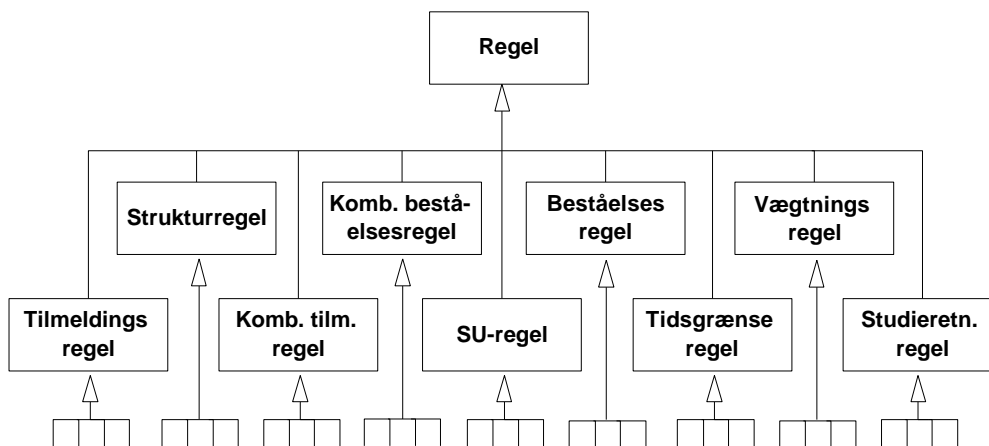
3. Design af udvalgte dele af STADS

På baggrund af analysen af udvalgte dele af STADS gennemgås i dette kapitel nogle designovervejelser. Designet er på ingen måde fuldstændigt. Der ses udelukkende på nogle enkeltdele vedrørende systemets funktioner og check af regler.

3.1 Regler

Forholdet mellem klasserne Regel og Tildelt Regel er i analysen beskrevet som en association. Mængden af forskellige regler er imidlertid konstant (i det nuværende STADS er antallet godt 100), og hver regel findes kun én gang ('one-of-a-kind'). Det bemærkes endvidere, at en katalogregels tilstand er konstant. I analysen er beskrevet, at hver klasse katalogregel har sin associerede tildelte regel klasse, dvs. der også findes godt 100 klasser for tildelte regler. Da enhver tildelt regel altid er associeret med det samme (statiske) katalogregel-objekt over hele sin livscyklus vælges, at denne association mellem katalogregler og tildelte regler designes som et klasse-instans forhold. Dvs. at en tildelt regel gøres til en instans af en katalogregel-klasse.

På Figur 21 ses klassen Regel, dens subclasses og de ikke tegnede klasser er de konkrete klasser af katalogregler. I det følgende omtales katalogregler som hidtil, men teknisk set er en katalogregel ikke fysisk til stede i systemet. En katalogregel udtrykker et klassetilørsforhold.

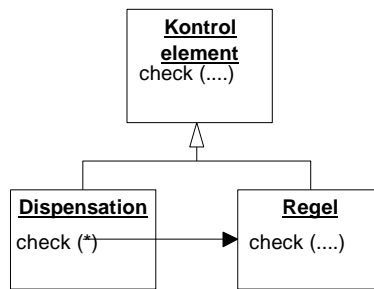


Figur 21. En katalogregel er en klassebeskrivelse for et tildelt regel-objekt

3.2 Dispensationer

Der kan dispenseres fra en regel for en studerendes studieforløb. Som beskrevet i analysen er klassen Dispensation associeret med klassen Forløbselement. For et element kan gives flere dispensationer.

Da der endvidere er brug for at kunne referere regler og dispensationer under ét, indføres en superklasse kaldet Kontrollement. Strukturen er vist på Figur 22. En dispensation overskygger en regel for en aktivitet og giver den nogle særlige egenskaber.



Figur 22. Kontrol element er superklasse for Regel og Dispensation. For klassen Kontrol element er check en abstrakt metode. Dispensation specialiserer check til at foretage delegering til den regel, dispensationen gælder for. Check-metoden på klassen Regel er den egentlige semantik for reglen.

Gives der dispensation for eksempelvis en bestået regel på en undervisningsaktivitet overskygger dispensationen undervisningsaktivitetens regel, og referencer til aktivitetens regel går nu via dispensationen ved delegering i metoden check, som delegerer kald videre til aktivitetens regel med de dispenserede parametre, som vist på Figur 22³. Når en regel er checket, returneres til dispensationen, som eventuelt afviger fra checkets resultat. På den måde bliver dispensationen et transparent led ved bearbejdning af en studerendes studieforløb.

Der kan gives flere dispensationer fra en regel, men på et givent tidspunkt kan kun én være i virkning.

3.3 Funktioner og check af regler

Hver af systemets funktioner stiller krav om check af en mængde regler for de berørte forløbselementer. Fremfindning af relevante regler for et forløbselement og check af regler er en generel operation for systemet. Denne operation er endvidere af en så stor kompleksitet, at den er nyttig at isolere i en selvstændig komponent. Komponenten kaldes oo-regelchecker (objektorienteret regelchecker). Oo-regelchecker er ikke identisk med regelchecker i det eksisterende STADS. I denne rapport er oo-regelchecker begrænset til kun at foretage check på et enkelt forløbselement, dvs. oo-regelchecker understøtter ikke, at et check på eksempelvis en eksamensaktivitet propagerer sig til et check på hele den studerendes uddannelse. Dette var vi valgt af afgrænse os fra i denne rapport. Oo-regelchecker varetager ikke oprettelse og nedlæggelse af forløbselementer. Det har vi også valgt af afgrænse os fra.

STADS-regelchecker har andre funktioner, som heller ikke er beskrevet i denne rapport. I det følgende beskrives et design af oo-regelchecker og de funktioner, som den varetager.

3.4 Funktioner

Strukturen af funktioner og deres forhold til regelchecker beskrives i det følgende med udgangspunkt i funktionen til registrering af studerendes resultater.

En funktion fastsætter hvilke regler, der er relevante at checke for den givne situation. Før regelchecket foretages overgives resultatet til forløbselementet. På baggrund af en mængde af

³ Strukturen er en variant af mønstret Decorator (Gamma et al., 1994).

regelklasser, i det følgende omtalt som katalogregelsæt, som er relevante for funktionen foretager regelchecket undersøgelser af hvilke regler, der gælder for aktiviteten, herunder hvilke regler der er givet dispensation for. Regelchecket returnerer om checket var ok eller ej. På baggrund af regelchecket bliver resultatet endeligt registreret i det relevante forløbselement eller afvist. I tilfælde af fejlet regelcheck returneres hvilken regel, der er fejlet, så det er muligt at reagere på baggrund af fejlen.

På abstrakt form ser funktionen Registrér Resultat ud som følgende:

```
Funktion Registrer Resultat(Forløbselement felem, Resultat res)
Katalogregelsæt krs := (6,19,21,...funktionens krævede regler...)
felem.tildelResultat(res)
regelCheckResultat := felem.checkRegler(krs)
if regelCheckResultat succes
then felem.anerkendTildeltResultat
else felem.afvisResultat
```

felem er elementet, aktiviteten m.v., som er modtaget et resultat, res, for.

Funktionen anvender en række operationer på forløbselementet. Operationen tildelResultat forhåndsregistrerer resultater før check af dette. Operationen anerkendTildeltResultat er den egentlige registrering af et resultat og afvisResultat: dropper det forhåndsreserverede resultat på forløbselementet i tilfældet, hvor regelchecket fejler.

Operationen CheckRegler på Forløbselement er den interessante i forhold til regelcheckereren og er beskrevet i næste afsnit.

Systemets øvrige funktioner er ikke beskrevet dette dokument. Funktionerne gør alle brug af regelcheckereren ved at kalde den med et katalogregelsæt. Endvidere gør funktionerne brug af en række operationer, der er specifikke for funktionen.

Systemets funktioner hører til i hver deres respektive klasse. Mængden af funktionsklasser udgøre systemets funktionskomponent. Hver funktionsklasse har - udover en metode med funktionen - en tilstand, der angiver mængden af relevante katalogregler for funktionen.

3.5 Oo-Regelcheckeakeren

Udgangspunktet for et regelcheck er, at en funktion ønsker at sikre, at nogle givne betingelser er opfyldt. Nogle funktioner checker om en given studerende opfylder beståelses kravene, andre om betingelserne for en tilmelding er opfyldt og opdaterer forløbselementer. Der er flere forhold som vi tager udgangspunkt i designet af regelcheckereren.

1. Regelcheck initieres altid med udgangspunkt i et forløb for en studerende, ikke i en beskrivelse.
2. Vi ønsker at den samme regelchecker kan benyttes af flere forskellige funktioner og på alle slags regler, og ønsker derfor at parametrisere regelcheckereren på passende vis.
3. Et forløb vil altid være beskrevet ved en træstruktur. Dette er ikke nødvendigvis tilfældet med beskrivelses strukturen.

En konsekvens af (1) er at konsistenscheck, som er et check der tager sit udgangspunkt i beskrivelserne, ikke skal understøttes af regler og regelcheckeren. Der er altså ikke regler der udtaler sig om regler.

Det skal være muligt at initiere et regelcheck på ethvert forløbselement. For at tilgodese en række forskellige funktioner vil regelcheck metoden tage en mængde af katalog regler som parameter. Kun tildelte regler og dispensationer, der er beskrevet ved disse katalog regler, vil blive checket. Det er selvfølgelig interessant at få at vide om checket gik godt eller fejlede, men hvis der viste sig at være regelbrud er det mere interessant at få fat på hvilke regler der fejlede.

Regelcheck for et forløbselement vil derfor bestå i:

1. At udvælge alle de regler og dispensationer der er relateret gennem kontrollering og som er med i katalogregelsættet (en dispensation siges at være med i katalogregelsættet, hvis den regel den dispenserer fra er med), og
2. At checke de kontrolelementer der derved er udvalgt.

På abstrakt, algoritmisk form ser klassen Forløbselement's operation CheckRegler ud som følger:

```
Forløbselement CheckRegler (Katalogregelsæt krs)
  EffektivRegelsæt := udvælgRegler(krs)
  For hver regel r i effektivRegelsæt: r.check(this)
```

Den anvendte syntaks for beskrivelsen af operationen betyder, at klassen Forløbselement har metoden CheckRegler.

Operationen udvælgRegler finder mængden af tildelte regler og dispensationer, som er relevante for det aktuelle forløbselement. Denne operation er beskrevet i næste afsnit. For hver regel og dispensation i det effektive regelsæt udføres et regelcheck på forløbselementet. Metoden check på en regel er et kald af reglens semantik, som vist i det følgende.

```
Kontrollement check (Regel reg)
  reg.check(felem, this)
```

Årsagen til at objektet selv er en parameter på metoden, er at dispensationer skal kunne håndteres transparent.

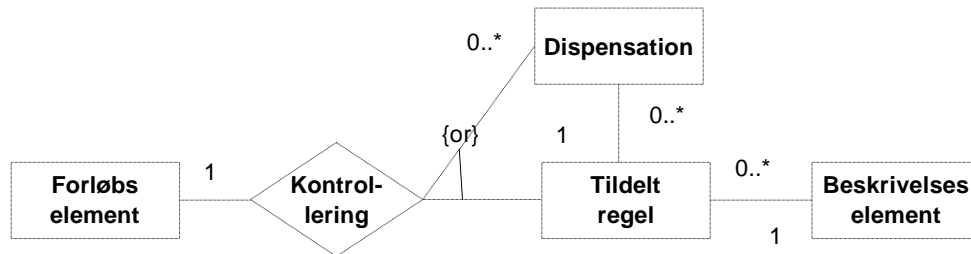
Et regels operation check indeholder semantikken af reglen og er derfor vidt forskellig fra regel til regel. Dette dokument har ikke til formål at beskrive semantikken af reglerne. En enkelt regel, Regel 21: 'Karakteren skal være mindst #1', skitseres som eksempel på indhold af en katalogregel.

```
Regel check (Forløbselement felem, tildeltRegel treg)
  if felem.modtagetResultat >= treg.#1
  then returner RegelOverholdt
  else returner RegelBrudt
```

3.5.1 Det effektive regelsæt

Som nævnt består et regelcheck af (1) at udvælge alle de effektive regler og dispensationer, dvs. kontrolelementer, for forløbselementet, og (2) at checke de regler og dispensationer der derved er udvalgt. I dette afsnit beskrives det første af disse trin.

Et forløbselement kan både relatere sig til tildelte regler og dispensationer for sådanne. Denne relation vil vi kalde for kontrollering, som vist på Figur 23.



Figur 23. Forholdet mellem forløbselementer og kontrolelementer

Problemet består således i at definere kontrolleringsrelationen på basis af de relationer der er givet fra modellen. Kontrolleringsrelationen defineres i det følgende. Først ignoreres nedarvede regler. Dernæst ses relationen, når der også medtages nedarvede regler.

3.5.2 Kontrolleringsrelationen uden nedarvede regler

Regelcheckerens tager udgangspunkt i et givent forløbselement. Mens man ikke har direkte adgang til regler, så kan de nås via det associerede beskrivelseselement, som igen har adgang til reglerne. Beskrivelseselementet udstyres med en metode, der udvælger det sæt tildelte regler, der ligger inden for et givent katalogregelsæt.

På basis af forløbselementet har man direkte adgang til eventuelle dispensationer, der måtte være tilknyttet elementet. Dog er man kun interesseret i dispensationer for de regler, der blev udvalgt på basis af katalogregelsættet.

Det effektive sæt af kontrolelementer består af det kombinerede udvalg af regler og dispensationer. På abstrakt form er algoritmen derfor (udtrykt på metode på et forløbselement):

```
Regelsæt udvælgRegler (Forløbselement felem, Katalogregelsæt krs)
beskrivelseselement := felem.beskrivelseselement;
regler := beskrivelseselement.udvælgReglerFra( krs);
dispensationer := udvælgDispensationer (regler);
dispenseredeRegler := {r ∈ regler | ∃ d ∈ dispensationer: d.regel = r };
effektiveRegler := (regler \ dispenseredeRegler) ∪ dispensationer;
return effektiveRegler;
```

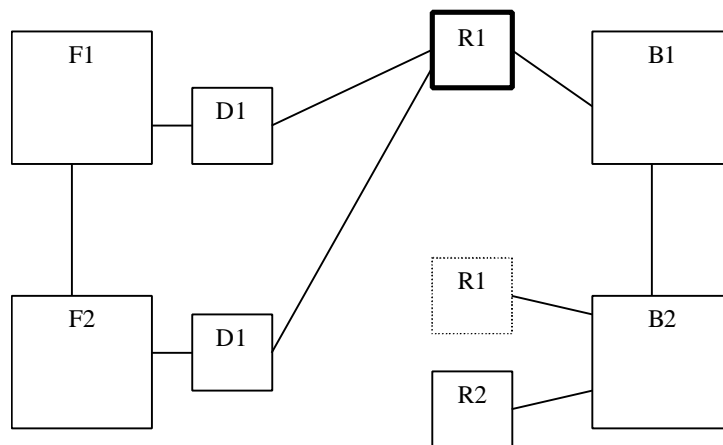
Det næste skridt er at indføje nedarvede regler.

3.5.3 Kontrolleringsrelationen med nedarvede regler

Den vigtigste observation i forbindelse med nedarvede regler er, at regler er bundet til beskrivelser, og at beskrivelseselementerne ikke udgør en træstruktur, men hvad der teknisk

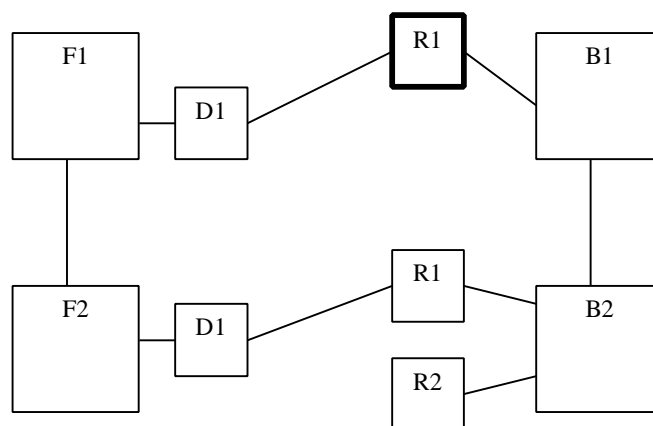
set kaldes en "directed acyclic graph", eller en retningsbestemt cykelfri graf. Det betyder, at man ikke fra et givet element kan finde en unik overordnet knude. Det har den konsekvens at man ikke kan vandre opad i beskrivelserne og finde nedarvede regler, men at nedarvede regler skal skubbes oppefra og ned, når de tilføjes. Vi må derfor gå ud fra, at ethvert beskrivelselement giver os nem adgang til alle de regler, de har nedarvet. Denne adgang til et forløbselements regler er en ikke-simpel, men mulig fremsøgning. I denne rapport har vi afgrænset os fra at beskrive den.

I forhold til ovenfor er det derfor kun interessant at se på, hvordan dispensationer nedarves. I modsætning til beskrivelserne, så er forløbselementerne organiseret i et træ, og man kan derfor fra et givet forløbselement bevæge sig til elementet længere oppe. På niveauet ovenfor tilføjes alle de dispensationer, der er knyttet til regler, der ikke allerede har en dispensation knyttet til sig, idet de mest specifikke dispensationer overskygger de mindre specifikke.



Figur 24. Nedarvning af regler. Dispensationen D1 er givet på en nedarvet regel R1 for forløbselementet F2, der er en studerendes udgave af beskrivelselementet B2.

På Figur 24 er R1 en nedarvet regel knyttet til beskrivelselement B1. Den bliver derfor nedarvet til B2, hvor den er markeret med en stiplede kant. Der er givet dispensation for regel R1 både på F1 og F2. Idet algoritmen skal starte med F2, vil F2's dispensation D1 blive fundet først. F1's D1 vil ikke blive tilføjet, idet der allerede er en dispensation for R1.



Figur 25. Nedarvning af regler og dispensationer

På Figur 25 er situationen nu at en ny R1 regel bindes til B2. Som før startes med at F2's D1 er med i mængden af dispensationer. Dernæst skal vi kikke på om F1 tilføjer yderligere dispensationer. Det gør den ikke, idet F2's D1 er associeret med B1's R1, som ikke er den R1 der er den med i de udvalgte regler for B2.

Metoden `udvælgDispensationer` følger derfor følgende abstrakte algoritme:

```
Dispensationer udvælgDispensationer (Regelsæt regler)
  mine := {d ∈ mineDispensationer | d.regel ∈ regler};
  arvede := op.udvælgDispensationer(regler);
  return mine ∪ {d ∈ arvede | d.regel ∉ {r ∈ regler | ∃ d ∈ mine:
  d.regel = r} }
end
```

Det er således muligt at finde de regler og dispensationer, som er relevante at kontrollere for et givent forløbselement. Det bemærkes, at designet afgrænser sig fra at beskrive, hvordan et check på et element propagerer sig til andre forløbselementer.

4. Konklusion

I dette kapitel resumeres kort resultaterne fra analysen og designet. Der ses kort på de erfaringer, projektet har givet, samt hvilke muligheder, der ligger i et fremtidigt arbejde med resultaterne fra denne rapport.

Analysen er foretaget over samme problemområde, som udvalgte dele af det eksisterende STADS. Fokus har været det som i dag kendes som delsystemerne Uddannelsernes Struktur og Rammer - herunder Regelchecker. Analysen indeholder en detaljeret gennemgang af problemområdets objekter, og strukturerne mellem disse. Systemets hændelser, funktioner og systemgrænsefladen er kort gennemgået.

I designkapitlet er illustreret struktur og algoritmer for en objektorienteret regelchecker, oo-regelchecker. Det er vist hvorledes funktionaliteten for et regelcheck er fordelt udover de involverede objekter. Det bemærkes, at designet ikke er komplet i forhold til analysen.

4.1 Erfaringer

Den anvendte objektorienterede analyse- og designmetode er fra (Mathiassen et al., 1997). Især analysen holde sig til metoden. Da designet primært interesserer sig for regelcheckeren, har designmetoden kun været inspirationskilde i designfasen.

Det er vores opfattelse, at metoden beskrevet i (Mathiassen, 1997) er nyttig og anvendelig for kommerciel objektorienteret analyse. Det er vores erfaring, at vi kan udtrykke det samme som i den relationelle model. Den objektorienterede analysemetode er pragmatisk i sit syn på figurer, idet den erkender, at ikke alt kan beskrives grafisk. Figurer suppleret med tekst gør objektmodellen ligeså udtryksfuld som ER-modellen, og vi har opnået at få yderligere forståelse for problemområdet ved at arbejde med det ud fra objektsynsvinklen.

Mht. til design er vores erfaringer begrænsede og vi kan derfor ikke udtale os om metoden. Dog har vi bemærket, at for et system som STADS, hvor en meget stor del af systemets dynamik er givet ved anvendelse af regler, kan klassebeskrivelser, tilstandsdiagrammer og hændelser virke 'fattige' på information i forhold til det at systemet krævede. Det kan skyldes flere ting: 1) analysen ikke indeholder en komplet beskrivelse af reglernes adfærd. 2) det komplekse ligger i den algoritmiske kompleksitet af de beskrevne operationer. 3) kompleksiteten ligger i samspillet mellem en række operationer, hvilke kunne været beskrevet med interaktionsdiagrammer. De erfaringer, vi har haft med interaktionsdiagrammer, har dog ikke eftervist dette.

Vi har erfaret, at den dynamiske beskrivelse af et system er den vanskeligste. Vi tror, det skyldes, dels at vi med vores baggrund i relationel modellering er vant til kun at beskrive et systems datamodel, dels at et systems dynamik faktisk er det sværeste at modellere. Mere erfaring med interaktionsdiagrammer kunne eventuelt afhjælpe disse vanskeligheder.

Den i rapporten anvendte notation er en tilnærmet UML. Pga. manglende grafisk værktøj har det været nødvendigt med visse layoutmæssige ændringer fra standarden. Vi har undervejs i projektføreløbet afprøvet Rational's grafiske produkt Rational Rose C++ Demo 4.0 til at tegne UML med. Dette case-værktøj kunne med fordel være anvendt fra starten, idet det direkte understøtter UML-standardens.

UML har vist sig at understøtte den objektorienterede metode fint. Vi har dog enkelte kommentarer til metoden. I situationer, hvor to parallelle hierarkier har parvise associationer mellem hierarkiernes elementer, er UML ikke i stand til udtrykke denne association på en simpel måde. Intuitivt er der ingen tvivl om hvilke elementer, der skal parres, men der er ingen præcis semantisk udtryksform for dette. I UML skal hver parvis association udtrykkes eksplicit. Ved kun at associere hierarkiernes superklasser mistes information om at elementernes association er parvis.

I arbejdet har vi gjort anvendelse af flere design mønstre (Gamma et al., 1994). Det er vores erfaring, at design mønstre er med til at give en uddybende forståelse af essensen i forskellige objektstrukturer. Når en struktur har været erkendt som et mønster, har det været let at beskrive strukturen og det har givet en fælles forståelse af strukturen.

4.2 Videre arbejde

I designkapitlet indføres en relation kaldet kontrollering. Relationen indføres som et led i strukturering af regelcheckeren. Kontrollering er en kompleks relation, idet den ikke er en del af datamodellen, men skal beregnes. Denne type relation er en interessant konstruktion i forbindelse med design, og derfor kunne det være relevant at foretage yderligere undersøgelser af brugen og indholdet af komplekse relationer.

Rapporten indeholder en analyse og et design af regelcheckeren. Designet er på et sådant niveau, at næste skridt i designfase vil afhænge af den teknologiske platform, som systemet skal udvikles på. Et spændende projekt vil derfor være at undersøges, hvorledes Oracle 8 understøtter det objektorienterede design af regelcheckeren.

5. Litteraturliste

Peter Coad, Object-Oriented Patterns, 1992, Communications of the ACM, September 1992, pp. 152-159

Erich Gamma, Richard Helm, Ralph Johnson, John Vlisside, 1994, Design Patterns - Elements of Reusable Object-Oriented Software, Addison-Wesley Publishing Company

L. Mathiassen, A. Munk-Madsen, P. A. Nielsen, J. Stage, 1997, Objektorienteret Analyse og Design, Marko, Aalborg

Rational, 1997, UML version 1.1, www.rational.com

WM-data dokumenter, 1997, 'A107.662.6 Generelt om regelcheckereren', 'A017.282. IR Tværgående STADS systemdokumentation, 'A107.3242.1 Regelkatalog STADS 4.0.a', WM-data, Århus.