UNIVERSITETET I OSLO

Det matematisk-naturvitenskapelige fakultet

Eksamen i :	IN 219 — Store programsystemer
Eksamensdag :	Lørdag 13. desember 1997
Tid for eksamen :	09.00 - 15.00
Oppgavesettet er på :	3 sider
Vedlegg :	Ingen
Tillatte hjelpemidler :	Alle trykte og skrevne

Kontroller at oppgavesettet er komplett før du begynner å besvare det.
Les gjerne i gjennom hele settet før du begynner med besvarelsen.

Oppgave 1 (30 %)

Vi modellerer et firma ved bruk av følgende to basismengder:

Ansatt – informasjon om hver ansatt
Prosjekt – informasjon om hvert prosjekt som utføres av firmaet

På et vilkårlig tidspunkt trenger ikke alle ansatte nødvendigvis arbeide på et prosjekt og alle prosjekter trenger ikke nødvendigvis å være aktive (dvs. at noen ansatte arbeider på prosjektet til enhver tid). I tillegg finnes det en global konstant:

maksStørrelse : N

Nedenfor vises et utkast til et skjema som definerer firmaet:

Firma
ledere: F Ansatt
leder_for : Ansatt 7 ledere
arbeider_på : Ansatt 7 Prosjekt
∀m : ledere' m f *dom* leder_for
∀p : Prosjekt '#(arbeider_på 5 {p}) Â maksStørrelse

Oppg. 1A Forklar sammenhengene mellom ansatte, ledere og prosjekter som er modellert ved 'leder_for' og 'arbeider_på' (*tips:* et diagram vil kunne hjelpe!)

Følgende skjema har til hensikt å sette en ansatt på et prosjekt:

SettPåProsjekt

ΞFirma
e? : Ansatt
p? : Prosjekt

e? f arbeider_på 5 {p?}
#(arbeider_på 5 {p?}) < maksStørrelse
ledere' = ledere
leder_for' = leder_for
arbeider_på' = arbeider_på I {e? 2 p?}

Oppg. 1B Utvid skjemaet SettPåProsjekt til å gi passende feilmeldinger hvis forhåndsbetingelsene ikke er oppfylt.

Oppg. 1C Beskriv effekten av operasjonen SettPåProsjekt hvis den første forhåndsbetingelsen (e? f ...) fjernes fra predikatene.

Følgende skjema har til hensikt å definere forfremmelse av en ansatt til å bli leder. Ingen forhåndsbetingelser er imidlertid definert.

Forfremmelse

ΞFirma
e? : Ansatt

. . . . .
ledere' = ledere B {e?}
leder_for' = leder_for
arbeider_på' = arbeider_på

Oppg. 1D Definer passende forhåndsbetingelse(r) for skjemaet ovenfor og begrunn valget.

Gitt at ledere og andre ansatte utgjør en struktur med to nivåer, omskriver vi det opprinnelige skjemaet slik at vi får:

Firma

ledere: F Ansatt
arbeidere : F Ansatt
leder_for : arbeidere 7 ledere
arbeider_på : arbeidere 7 Prosjekt

ledere H arbeidere = G
∀p : Prosjekt '#(arbeider_på 5 {p}) Â maksStørrelse

Oppg. 1E Er dette nye skjemaet bare en omskriving av det opprinnelige skjemaet, eller har basismodellen blitt endret?

Oppg. 1F Skisser effekten, hvis noen i det hele tatt, av det modifiserte skjemaet Firma på skjemaene SettPåProsjekt og Forfremmelse hvor du bruker din egen versjon av Forfremmelse med inkluderte forhåndsbetingelser (oppg. 1D).

Oppgave 2 (20 %)

Oppgave 1 beskrev kravspesifikasjoner i Z. Med dette som utgangspunkt ønsker vi nå å lage et objekt-orientert design.

Oppg. 2A Lag et overordnet objekt-orientert design utfra informasjonen gitt i de to første skjemaene (Firma og SettPåProsjekt) beskrevet i oppgave 1. Bruk de teknikkene du finner hensiktmessig (diagrammer, CRC-kort, pseudokode etc.).

Oppg. 2B Gi en vurdering av gjennomføringen av oppgave 2A der du vektlegger hva som var vanskelig og hva som gikk greit. Gi til slutt en kortfattet generell beskrivelse av bruk av objekt-orientert design i sammenheng med Z-spesifikasjoner.

Oppgave 3 (20 %)

Oppg. 3A Program-metrikker kan anvendes på et programsystem for å indikere kvaliteten på eksterne attributter. Beskriv tre slike metrikker og hvilke eksterne attributter de (sannsynligvis) sier noe om.

Oppg. 3B I den obligatoriske oppgaven lagde dere en kvalitetsplan og en prosjektplan. I hvilken grad fulgte dere disse planene i prosjektet? Hvis dere skulle ha gjennomført prosjektet om igjen, hva ville dere ha forbedret? Begrunn svarene. (Du kan evt. bruke erfaringene fra et annet prosjektarbeid du har deltatt i, men da må du gi noen ekstra opplysninger om dette arbeidet.)

Oppg. 3C Beskriv kort en standard modell for evaluering av modenheten til en organisasjon/bedrift med hensyn til dens programvare-prosesser. Hva er hensikten med å gjennomføre en slik evaluering?

Oppgave 4 (30 %)

Anta at du er ansatt i en programutviklingsbedrift og har vært leder av flere prosjekter. Du har erfart problemer med å holde oversikten over hvilke versjoner som finnes av ulike dokumenter, kodefiler etc., og hvilke versjoner av slike komponenter som samlet utgjør en konfigurasjon av et system. Dere har hittil ikke hatt noe verktøystøtte for hjelp til slike problemstillinger.

Oppg. 4 Formuler et brev til edb-sjefen der du beskriver problemer med å identifisere ulike versjoner av ulike komponenter, og sikre at endringer utføres og nye versjoner og "releaser" lages på en kontrollert måte. Beskriv fordeler og ulemper ved ulike alternativer til løsning. Til slutt argumenter for ett av disse alternativene. Gjør selv de antakelsene du finner nødvendig. Signer brevet anonymt, f.eks. G. Hansen (slik at sensor ikke ser hvem du er!).

Skisse til lønsning, Sensurfrist: 6. januar

Formal Methods - Tutorial 2 Solution

Oppg. 1A The following constraints are imposed by the schema given.

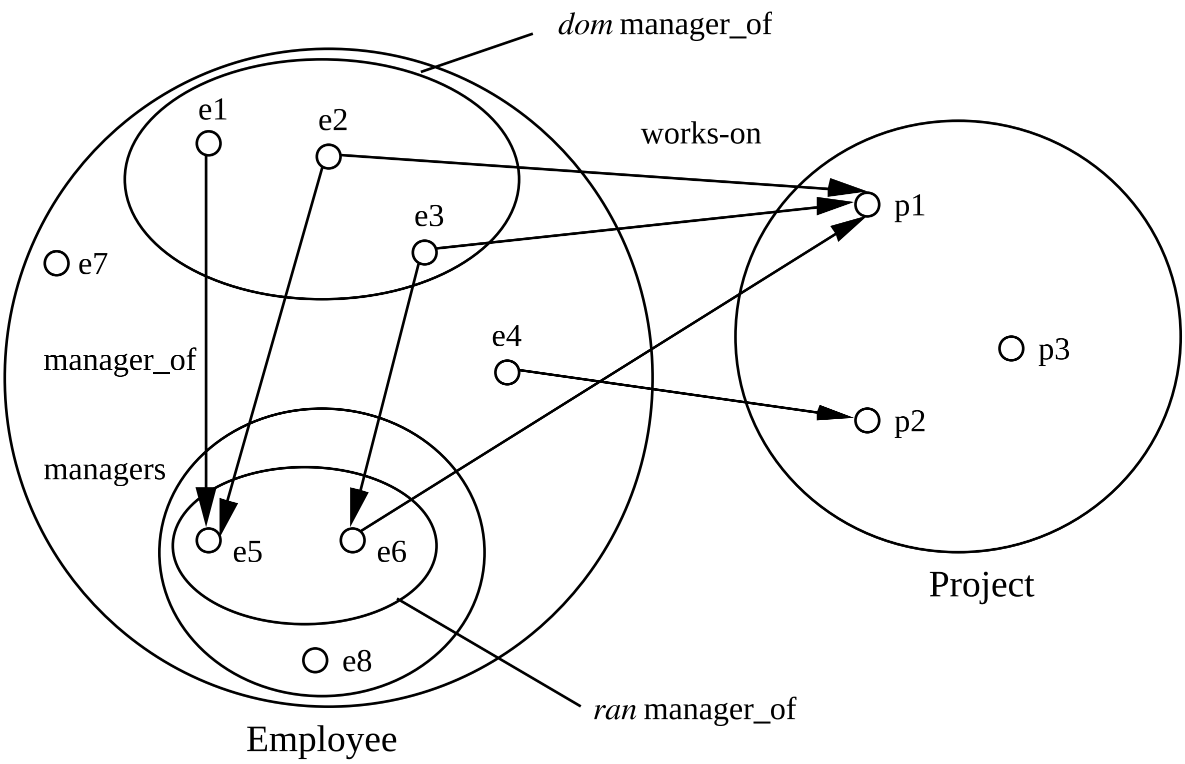
managers is a finite set of Employee, managers and other employees are drawn from the same set;

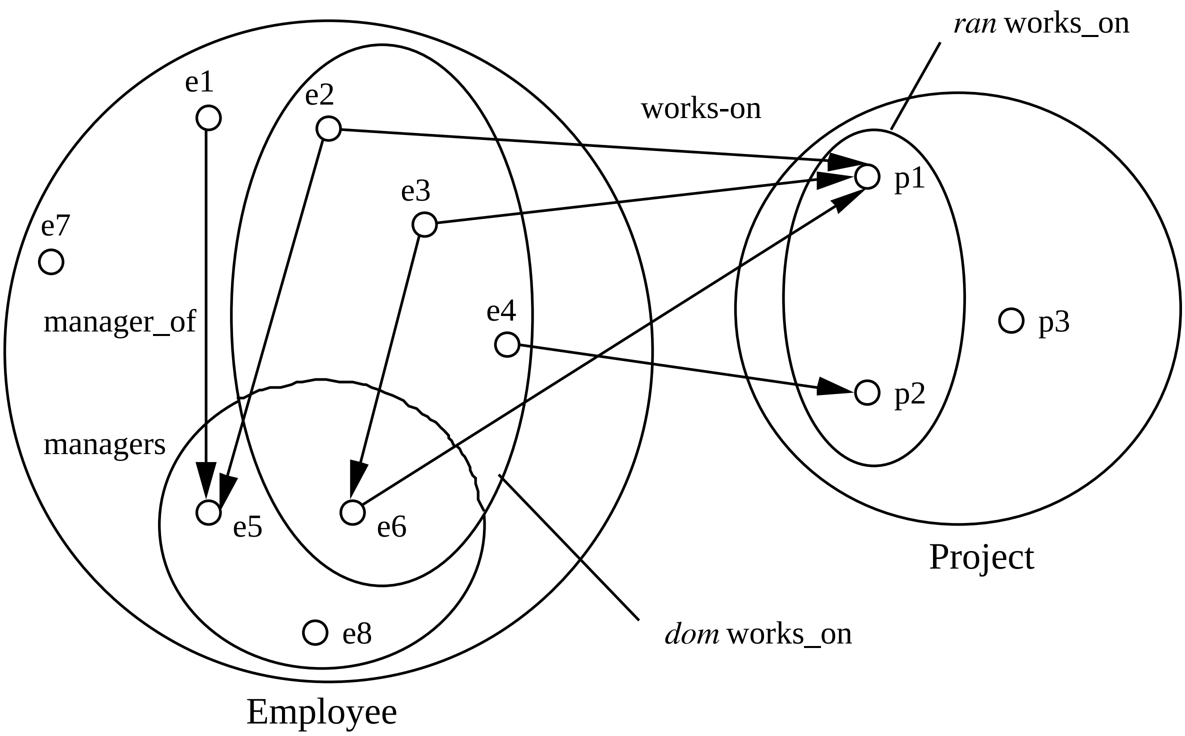
manager_of is a partial mapping, an employee has a unique manager but one manager may manage many employees; an employee may not have a manager and a manager may not manage any employees.

works_on is a similar partial mapping, an employee may work on only one project at any given time but a project may have many employees assigned to it; projects can exist without employees and employees can exist without being currently assigned to any project.

The first predicate states that a manager cannot be in the domain of the manager_of mapping, therefore we have only one level of management! This also prevents a manager managing himself.

The second predicate states that for all projects, the size of the works_on set must be less than or equal to the constant maxSize.





Oppg. 1 b Possible schemas to handle errors:

EmployeeAssigned
ÛCompany
e? : Employee
p? : Project
m! : message
e? a works_on 5 {p?}
m! = EmployeeAlreadyAssignedToProject

TeamFull
ÛCompany
p? : Project
m! : message
#(works_on 5 {p?}) Ê maxSize
m! = FullTeamAlreadyAssigned

Assignment
AssignToProject m EmployeeAssigned m TeamFull

In TeamFull, #(works_on 5 {p?}) = maxSize is also acceptable.

If you include the error reports within the main schema then you must specify the no-change post-conditions in all cases, for example:

AssignToProject
□Company
e? : Employee
p? : Project
m! : message
e? a works_on 5 {p?}
m! = EmployeeAlreadyAssignedToProject
managers' = managers
manager_of' = manager_of
works_on' = works_on
m
#(works_on 5 {p?}) Ê maxSize
m! = FullTeamAlreadyAssigned
managers' = managers
manager_of' = manager_of
works_on' = works_on
m
e? f works_on 5 {p?}
#(works_on 5 {p?}) < maxSize
managers' = managers
manager_of' = manager_of
works_on' = works_on I {e? 2 p?}
m! = SuccessfulAssigment

Oppg. 1c It would no longer be possible to recognise the error that an employee is already assigned to the project. However, the function overriding (works_on I {e? 2 p?}) would still work, overwriting the existing mapping with the same one again! The size of the *dom* works_on will not be increased.

Oppg. 1d The straightforward answer is to have two preconditions:

e? f managers	e? is not already a manager
e? f *dom* manager_of	e? is not managed by anybody

It can be argued that is the second precondition could be replaced by removing e? from *dom* manager_of as an action.

Oppg. 1e The basic model has changed because managers can no longer be assigned to work on projects. In the original schema it was allowable for managers to work on projects as they are a subset of Employees. The new schema restricts workers to work on projects (works_on : workers 7 Project).

Oppg. 1f AssignToProject should have e? : Employee replaced by e? : workers (or some equivalent change); also workers' = workers should be added to show no change in the workers set.

Promotion should be changed in a similar way by defining e? : workers and also removing the employee from the workers set by workers' = workers \ {e?}.

Here's a possible alternative schema for Promotion which changes w? from a worker to a manager and ensures that there are no inconsistencies in the schema.

Promotion
Company
w? : workers
managers' = managers B {w?}
manager_of' = manager_of 4 {w?}
works_on' = works_on 4 {w?}
workers' = workers \ {w?}

Final thought - when answering this type of question your answer must be based on the specification given even if it seems rather odd! For example, the schema given says nothing about how (or whether) managers manage projects; manager_of and works_on are separate mappings.

Oppgave 3 (20 %)

Oppg. 3A Interne program-metrikker (interne attributter) kan måles direket, f.eks. linjer kode. Eksterne attributter er størrelser vi ønsker å si noe om, men som bare indirekte kan måles via interne attributter, f.eks. vedlikeholdbarhet, se f.eks. figure s. 625 i boka.

Oppg. 3C CMM eller SPICE. Hensikt, å vurdere dagens status (etablere base-line), identifisere områder for forbedring - utgangspunkt for forbedringsplan.

Oppgave 4 (30 %)

Fra foiler:

→ SCCS , RCS – håndterer multiple versjoner av tekstfiler (vanligvis kildekode-filer) i henhold til en “sjekk-ut/sjekk-in”-protokoll

→ Av hensyn til diskplass lagres bare forskjellene (delta) mellom versjoner. SCCS anvender forover-deltaer (lagrer original); RCS bakover-deltaer (lagrer siste)

→ RCS tillatter uavhengig utvikling av forskjellige releaser

→ RCS (og CVS) har stort sett overtatt etter SCCS

→ Finnes mange mer moderne og avanserte verktøy: ClearCase (forløper: DSEE), Vesta, ...

→ Identifisering av gyldige konfigurasjoner må fortsatt gjøres manuelt

NAME
cvs - Concurrent Versions System

DESCRIPTION
cvs is a front end to the rcs(1) revision control system which extends the notion of revision control from a **collection of files** in a single directory to a hierarchical **collection of directories** consisting of revision controlled files. These directories and files can be combined together to form a software release. cvs provides the functions necessary to manage these software releases and to control the **concurrent editing** of source files among **multiple software developers.**