NTNU
Norges teknisk-naturvitenskapelige universitet

Fakultet for fysikk, informatikk og matematikk

Institutt for datateknikk og informasjonsvitenskap

BOKMÅL

Sensurfrist: 27. juni, 2006

Eksamen i fag
TDT4140 Systemutvikling

6. juni, 2006
kl 0900 - 1300

Hjelpemidler A1:
Kalkulator tillatt
Alle trykte og håndskrevne hjelpemidler tillatt:

Faglig kontakt under eksamen:
Professor Tor Stålhane, tlf. 94484

Poengene viser hvor mange poeng det er mulig å få på hver oppgave. Innen en oppgave teller deloppgaver likt, med mindre annet er angitt.

Lykke til!

.

Innledning

Overalt i oppgaven der vi bruker ordet ”systemet” mener vi systemet JAPP som er beskrevet i vedlegg A.

Dersom du trenger informasjon som ikke står i oppgaveteksten må du:

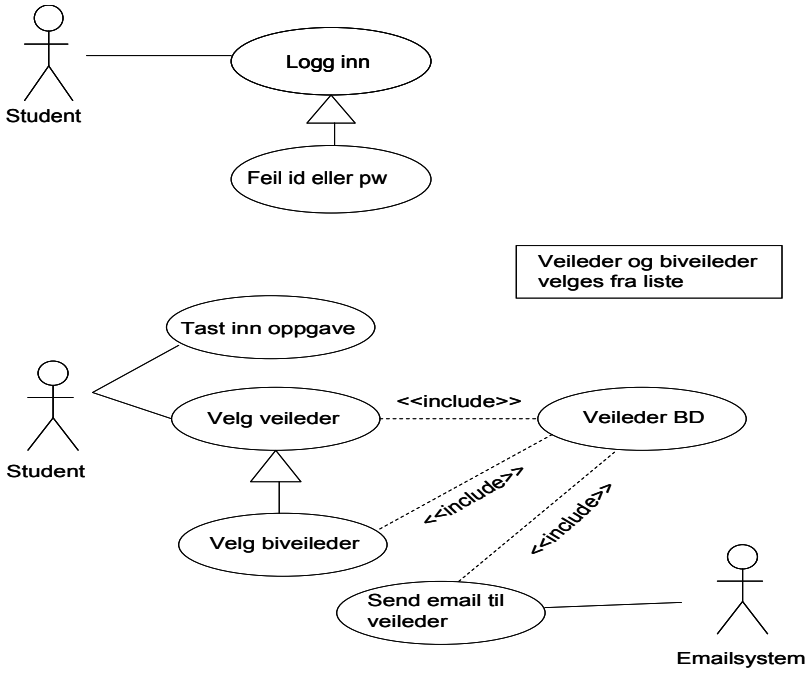
- Forklare kort hvorfor du trenger denne informasjonen
- Gjøre de nødvendige antagelsene. Disse antagelsene må beskrives i besvarelsen.

OBS: I oppgaveteksten har jeg brukt de to begrepene ”velg masteroppgave” og ”registrer masteroppgave” om hverandre. Dette har skapt noe forvirring hos en del av studentene. Derfor: både de som har behandlet dette som samme funksjonen og de som har regnet dette som to funksjoner skal ha full skår viss de har gjort det på en riktig og konsistent måte ellers.

Oppgave 1 – Use case, 25 poeng

1. Lag use case diagram for følgende funksjoner:
   - En student logger seg inn på JAPP.
   - En student velger en masteroppgave. Use case må inneholde både personaktører og eventuelle systemaktører.

     Siden et use case er et scenario kan man godt dele opp use case for registrering av oppgave i to deler – en med og en uten biveileder.



2. Lag tekstlige use case for følgende funksjon:
   - En student logger seg inn på JAPP.

Logg på JAPP
Prebetingelse: studenten er ikke logget på JAPP.
1. Tast inn brukernavn ***2. Tast inn passord***

3. Systemet viser JAPP startside for riktig personkategori
1.1 Gi melding om ukjent brukernavn ***1.2 Inkrementer teller*** ***1.3 Viss teller < maks, gå til 1. Ellers gi feilmelding og terminer JAPP.*** ***2.1 Gi melding om feil passord*** ***2.2 Inkrementer teller*** ***2.3 Viss teller < maks, gå til 1. Ellers gi feilmelding og terminer JAPP***

- En student velger en masteroppgave

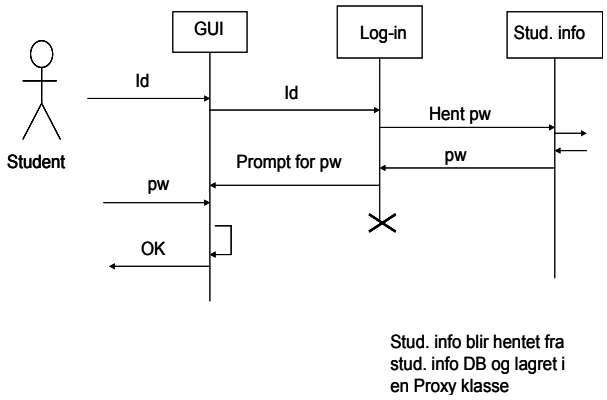
Student velger masteroppgave
Prebetingelse: studenten er logget inn på JAPP og har valgt funksjonen "Velg masteroppgave"
1 Tast inn oppgave tittel – maks n tegn. ***2 Velg hovedveileder fra liste av tilgjengelige veiledere*** ***3 Viss studenten ikke skal ha en biveileder, gå til 5*** ***4 Velg biveileder fra liste av tilgjengelige veiledere*** ***5 Systemet genererer email til hovedveileder*** ***6 Velg funksjonen "Avslutt registrering"***
1.1 Gi melding om for lang tittel. ***1.2 Gå til 1*** ***2.1 Studenten velger funksjonen "Avslutt registrering" uten å ha registrert veileder. Gi melding om oppgaven må ha veileder*** ***2.2 Gå til 2***

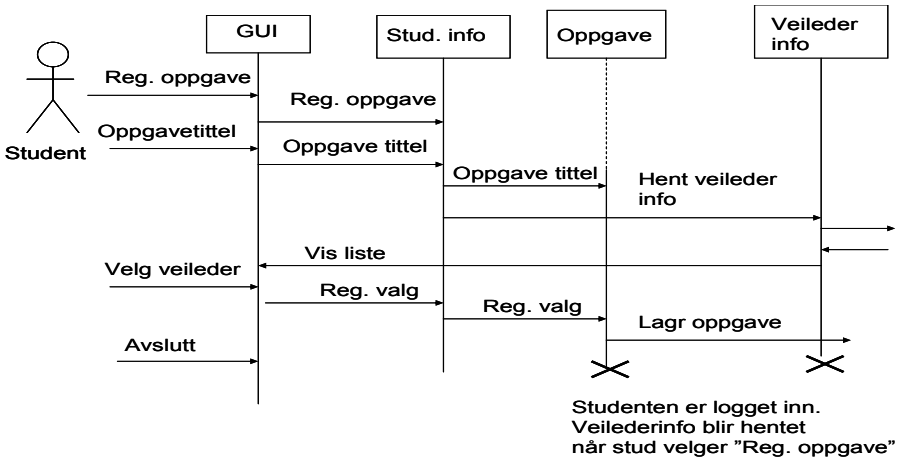
- Et medlem av fagstaben setter karakter på en masteroppgave.

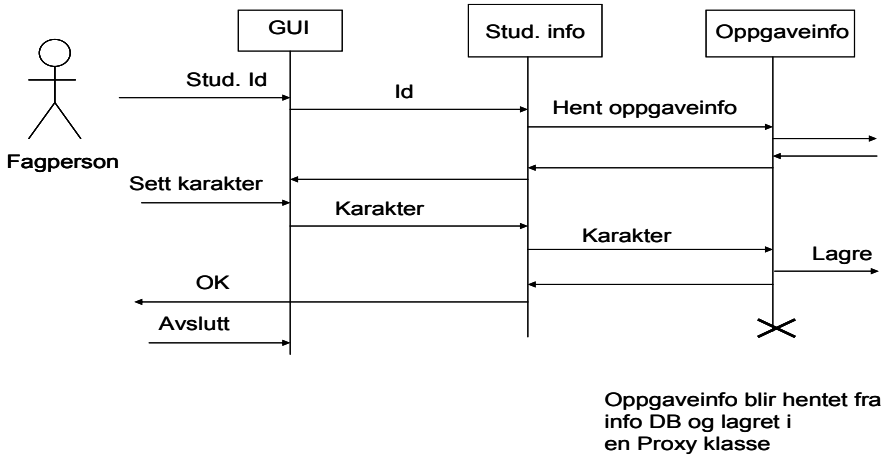
Sett karakter på masteroppgave
Prebetingelse: personen har logget inn på JAPP og har valgt funksjonen "Sett karakter på masteroppgave"
1 Personen finner studenten via studentnummer ***2 Personene taster inn karakter*** ***3 Velg funksjonen "Avslutt karaktersetting"***
1.1 Studenten finnes ikke i registeret. Gi feilmelding ***1.2 Gå til 1*** ***2.1 Ikke en av bokstavene A til F. Gi feilmelding*** ***2.2 Gå til 2***

3 Lag sekvensdiagrammer for de situasjonene som er beskrevet i oppgave 1, spørsmål 2.

Det er fullt mulig for eksempel å la GUI kommunisere med stud. info. Log-in klassen ble lagt inn for å få en enkel GUI. Tilsvarende forhold gjelder for de andre sekvensdiagrammene. Det er imidlertid viktig at det er sammenheng mellom sekvensdiagrammene og klassediagrammene.







4 Sammenlikn use case diagram og tekstlig use case for valg av masteroppgave. Gi minst to eksempler på viktige beslutninger vi kan vise i et tekstlig use case som det er vanskelig eller umulig å vise i et use case diagram.

Viktige beslutninger som ikke kan vises i et use case diagram.
Generelt er det vanskelig å vise sekvensering og feilsituasjoner i et use case diagram.
Få å vise dette må man bruke kommentarer.
Maks lengde på inntasta tekst.

Hva som skal skje viss ikke det er valgt en hovedveileder.
At valg av biveileder er valgfritt.

Oppgave 2 – Planlegging, 30 poeng

NTNU krever at systemet skal leveres, testes og godkjennes i tre inkrementer.

1. Hvorfor er det fordelaktig både for utviklerne og brukerne at systemet blir levert i tre inkrementer?

Fordeler for utviklerne:
Ved å se på ett inkrement av gangen kan de fokusere på en del av systemet og behøver ikke å ta inn over seg hele systemet med all funksjonalitet. I tillegg kan de få feedback fra kundene på hvert inkrement og har dermed en større sjanse til å levere noe som kundene er fornøyd med.

Fordeler for kunden:
Kunden kan få noe av systemet tidligere og kan dermed begynne å bruke systemet, få erfaringer og få støtte til deler av arbeidsprosessen. I tillegg kan brukerne gi feedback på funksjonalitet og GUI og dermed få et system de vil bli mer fornøyde med.

2. Del opp systemkravene i tre inkrementer – A, B og C - som kan implementeres i sekvens. Lag en liste som viser hvilke krav som skal implementeres i hvilket av de tre inkrementene A, B og C. I tillegg må du forklare hvorfor du har valgt denne oppdelinga.

   - ***Inkrement A***
     Krav 1, krav 2a og 2b, 4a og 4b. Når dette er implementert kan studenter og fagstab logge seg inn. Studentene kan velge en masteroppgave og faglærer kan tilordne en sensor til oppgaven. Dette er et minimum av funksjonalitet som gjør at de involverte kan få hjelp med sine oppgaver.
   - ***Inkrement B***
     Krav 2c, 3a, 3b, 3c, 3d, 4e og 4e. Når dette er implementert kan også administrasjonen få gjort sine oppgaver. I tillegg kan vi nå få laget sensurskjema slik at oppgaven kan få en karakter uten å måtte ty til de gale rutinene.
   - ***Inkrement C***
     Kravene 5 og 4c. Krav 5 inneholder de automatiske prosessene. Disse kommer til slutt siden de lett kan gjøres manuelt – for eksempel ved å bruke email.

3. Systemet må være ferdig 15. august. 1. september er siste frist for å registrere en masteroppgave og systemet må være operativt i god tid før denne dagen slik at alle studentene får registrert sine oppgaver. Dette betyr at vi må ha tett oppfølging av framdriften i prosjektet.
   - Hvordan vil du planlegge for å kunne ha tett oppfølging og hva slags info trenger du for dette formålet?

     Noen viktige momenter:
     Identifiser viktige risiki når det gjelder levering i tide. Bruk både proaktive og reaktive tiltak.
     Del prosjektet opp i små deler – ett til to dagsverk og følg opp kontinuerlig.
     Fjern mindre viktige aktiviteter fra planen – i det minste midlertidig.
     Ostehøvelprinsippet fungerer ikke.

- Hvordan vil du prioritere viss du oppdager at prosjektet ikke kan levere alt til avtalt tid?

  Det viktigste er at studentene får registrert oppgavene sine før fristen går ut. Ut fra dette bør man prioritere å implementere kravene 1, 2a og 2b. Når disse er ferdig bør man implementere kravene 4a og 4b. Resten av kravene er ikke kritiske før det nærmer seg tid for innlevering av masteroppgaver.

Oppgave 3 – Klassediagram, 30 poeng

1. Om pattern:
   - Hva er et pattern og hva brukes de til?

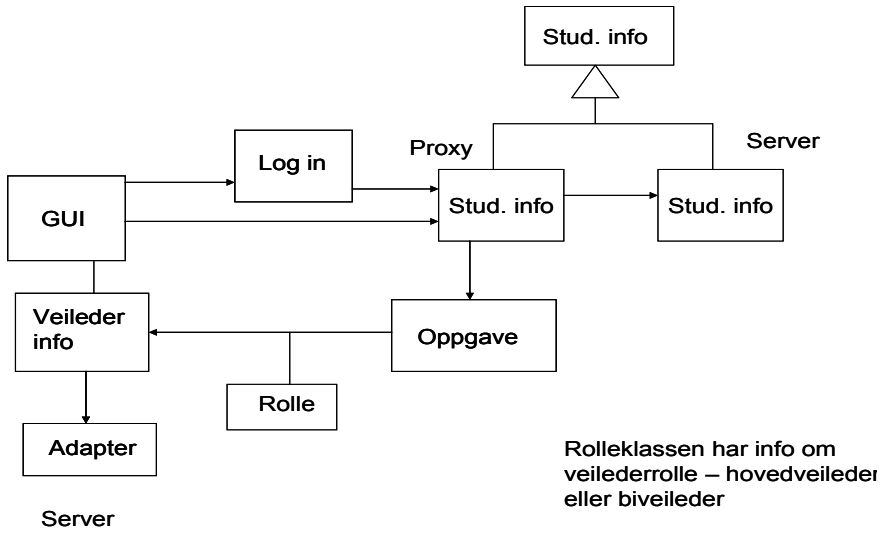
     Et pattern er en standardløsning på et standardproblem. Løsningen er på et overordna nivå og må tilpasses hvert enkelt konkret problem.

   - Hvorfor er det viktig å bruke pattern i systemutvikling?

     Bruk av pattern gjør at vi får en fornuftig løsning – det har fungert mange ganger før – og vi får en løsning som er gjennomtenkt. I tillegg vil det at det er en standardløsning gjøre det lettere for andre å forstå og eventuelt endre løsningen senere viss det skulle bli nødvendig.

2. Lag samtlige klassediagram for systemet på et nivå som passer til bruk tidlig i utviklingen av systemet.

Noen misforsto "samtlige klassediagram". Det skal tolkes som "klassediagram for alle klasser". Det er viktig at alle klasser som det er laget objekter til i ett eller flere sekvensdiagrammer finnes i klassediagrammet. Det er også viktig at problemer rundt veileder / bi-veileder er tatt med.
Det er ikke nødvendig at studenten har inkludert Proxy og Adapter pattern i diagrammet.



3. Hvorfor kan det være praktisk å bruke følgende pattern når vi skal lage dette systemet:

- Proxy

  Bør brukes når vi henter info fra andre databaser, for eksempel databasen for studentinfo eller sensorer.

- Adapter

  Når vi skal kommunisere med andre systemer – for eksempel Tapirs system for å trykke masteroppgaver. Adapter gjør at vi ikke trenger å endre våre klasser eller prøve å påvirke andres. I mange tilfeller kan et adapterpattern brukes i stedet for et Proxy pattern.

- Facade

  Brukes til å lage et felles grensesnitt for alle klassene i ei pakke. Facade patternet gjør at vi ikke trenger å endre på annen kode selv om vi endrer koden i en eller flere av klassene inne i pakka så lenge fasaden er uforandra.

4. Lag et detaljert klassediagram for klassen ”Masteroppgave”. Diagrammet skal også inneholde eventuelle spesialiseringer og vise relasjoner til andre klasser.
5. Hvordan ville du utvide denne klassen viss systemet også skal inneholde info om prosjektoppgaver?

   Lag en klasse som inneholder all info som er felles for masteroppgaver og prosjektoppgaver. Bruk arving til å lage to spesialiseringer – en for masteroppgaver og en for prosjektoppgaver. Eventuelt kunne man lage en mer generell løsning ved å bruke Factory pattern.

Oppgave 4 – Testing, 15 poeng

Det er vanlig å bruke use case diagrammer og sekvensdiagrammer som basis for å lage tester. Dette gjør det blant annet mulig å lage testene før man lager en design og skriver kode.

1. Forklar hvordan man kan bruke use casediagrammer og sekvensdiagrammer til å lage tester.

   Use case diagrammene viser hvilke funksjoner kunden trenger. Sekvensdiagrammene viser og tekstlige use case viser hvilken info som trengs og hva som er prebetingelser for hver enkelt funksjon. Tekstlige use case viser også feilreaksjoner og eventuelle feilmeldinger.

2. Bruk de diagrammene du laget i oppgave 1, punkt 2 og 3 til å vise hvordan du ville gjøre dette i praksis.

   Et eksempel – sett karakter: Logg inn og velg funksjonen ”Sett karakter”. Vi skal få opp et vindu med et felt for å skrive inn karakteren.
   Skriv inn en karakter – bokstav A – F. Avslutt og hent opp oppgaveinfo. Oppgaven skal nå ha få satt karakter.
   Skriv inn en feil karakter – bokstaven G. Systemet skal gi feilmelding og den ikke-tillatte karakteren blir ikke lagret.

3. Hvorfor er det fornuftig å lage testene før vi lager detaljert design og skriver kode?

Ved å lage testene før vi skriver kode vil testene være klare så snart koden er det. Dette oppmuntrer til å teste ofte – for eksempel hver gang vi endrer noe i koden. Testene kan utvikles av andre personer og vil ikke forsinke prosjektet. Det å skrive tester kna synliggjøre problemstillinger vi ikke har tenkt over da vi skrive kravspec. I tillegg unngår vi å bli for implementasjonsavhengige i testene våre.

Vedlegg A System for administrasjon av masteroppgaver.

Vårt firma – Saker og Ting eller SoT – skal lage et system for å administrere alle sider ved behandling av masteroppgaver ved NTNU. Studenter skal bruke systemet til å registrere oppgave og til å levere inn oppgaven. TAPIR skal bruke systemet til å produsere en innbundet versjon av besvarelsen. NTNU skal bruke systemet til å tilordne sensorer til oppgaven, registrere karakterer og sørge for at sensorene får utbetalt sine honorarer. Alle involverte parter skal bruke systemet til å produsere de nødvendige skjema.

Oppnevnte sensorer har ikke adgang til JAPP. De kommuniserer med hovedveiledere, bi-veiledere og NTNUs administrasjon bare gjennom email og dokumenter som utveksles pr. post.

For å holde oppgaven på et handterbart nivå er en del spesielle funksjoner som NTNU vil trenge i et virkelig system utelatt. Dette gjelder for eksempel eventuelle industripartnere og konfidensialitetsavtaler.

Dersom du trenger å sende email skal du bruke klassen NTNU_email som har to parametere, en email adresse og en tekststreng som inneholder meldingen.

Systemet JAPP skal tilfredsstille følgende krav:

1. Alle brukere kan logge seg på systemet med sitt vanlige NTNU brukernavn og passord.
2. En student skal kunne utføre følgende funksjoner:
   a) Registrere valgt masteroppgave. Vedkommende må skrive inn oppgavetittel og navn på hovedveileder og eventuelle bi-veiledere
   b) Viss det er flere som skal utføre en oppgave sammen, må hver enkelt student registrere seg på samme oppgave. Bare den som registrerer seg først skal skrive inn oppgavetittel og navn på hovedveileder og eventuelle bi-veiledere.
   c) Legge inn den ferdige oppgaven i JAPPs database i pdf-format. Oppgaven vil bli registrert som innlevert.
3. En person i NTNUs administrasjon skal kunne utføre følgende funksjoner:
   a) Gi oppgaven en sensor. Navn og adresse hentes fra en eksisterende database for NTNU-godkjente sensorer. Sensor vil få en email om dette. Dette er bare til info og krever ikke noe svar.
   b) Registrere at sensur er utført og sette inn den gitte karakteren.
   c) Endre innleveringsdato.
   d) Endre antall timeverk som brukes som grunnlag for å beregne sensors lønn.
4. Et medlem av fagstaben skal kunne utføre følgende funksjoner:
   a) Gi oppgaven en sensor. Navn og adresse hentes fra en eksisterende database for NTNU-godkjente sensorer. Sensor vil få en email om dette. Dette er bare til info og krever ikke noe svar.
   b) Endre innleveringsdato.
   c) Laste ned en ferdig oppgave fra databasen.
   d) Laste ned et sensurskjema
   e) Gi oppgaven en karakter.
5. Systemet utfører følgende oppgaver uten brukermedvirkning:
   a) Når en oppgave er registrert vil JAPP generere en
      I. Email til hovedveileder. Dette er bare til info og krever ikke noe svar.

II. Innleveringsdato. Denne datoen er gitt av en innkodet algoritme, men kan senere overstyres av NTNUs administrasjon eller fagstab.

b) Når oppgaven er levert inn – krav 1 - vil
   I. Oppgaven bli overført til TAPIRs manuskriptdatabase.
   II. JAPP sende en email til den person hos TAPIR som er ansvarlig for trykking av masteroppgaver.
   III. JAPP sende en email til hovedveileder om at oppgaven er levert.

c) Når en oppgave er sensurert vil JAPP sende et utfylt regningsskjema til NTNUs økonomiavdeling for overføring av betaling til sensors bankkonto. Systemet bruker innlagte standardverdier for å beregne størrelsen på utbetalingen.

SoT har ingen mulighet for å endre formater eller andre egenskaper ved de eksisterende databasene som JAPP er avhengig av:

- NTNUs brukerdatabase – alle studenters navn, adresse, fakultet, email adresse, brukernavn og passord. Databaseklassen er NTNU_bruker_db
- TAPIRs manuskriptdatabase- lagring av manuskript til trykking og innbinding. Databaseklassen er TAPIR_db
- NTNUs sensordatabase – all nødvendig info om alle godkjente sensorer, som for eksempel navn, adresse, bankkontonummer og kompetanseområder. Databaseklassen er NTNU_sensor_db.

Alle databaseklassene har en get-metode og set-metode for hver parameter du eventuelt måtte trenge å gjøre noe med.