UNIVERSITETET I OSLO

Det matematisk-naturvitenskapelige fakultet

Eksamen i: INF1050
Eksamensdag: 3. Juni, 2008
Tid for eksamen: 09:00-12:00
Oppgavesettet er på 3 **sider**
Vedlegg: Ingen
Tillatte hjelpemidler: Alle trykte og skrevne

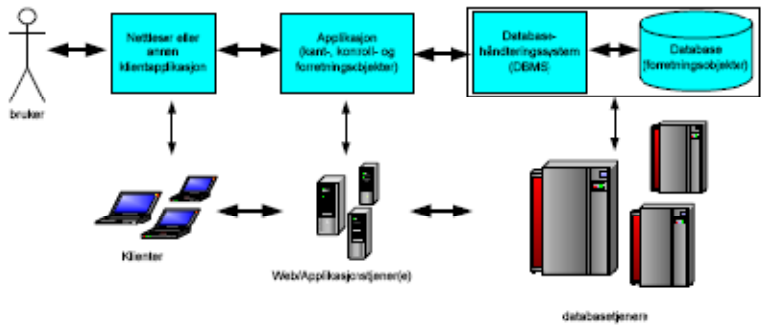
Kontroller at oppgavesettet er komplett
før du begynner å besvare spørsmålene.

Innledning
Forsikringsselskapet *Forsikring2* har en rekke spesialpakker med forsikringsløsninger som er tilpasset forskjellige befolkningsgrupper. Blant annet har de *NHO Worries* for NHO-medlemmer, *Distre Pluss* for akademikere og *Komplett Total* for de som har alt. Den nyeste satsningen til selskapet er pakken *Trang Hybel* for studenter. Ditt IT-selskap *Try-IT* har blitt forespurt om å delta i en anbudskonkurranse for mulig levering av en IT-løsning for det nye produktet. Produktet skal utelukkende tilbys og selges via internett.

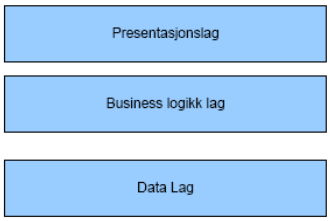
Oppgave 1 Tidlig fase (20%)

a) Skisser hvordan en moderne logisk og fysisk lagdelt arkitektur kan se ut.

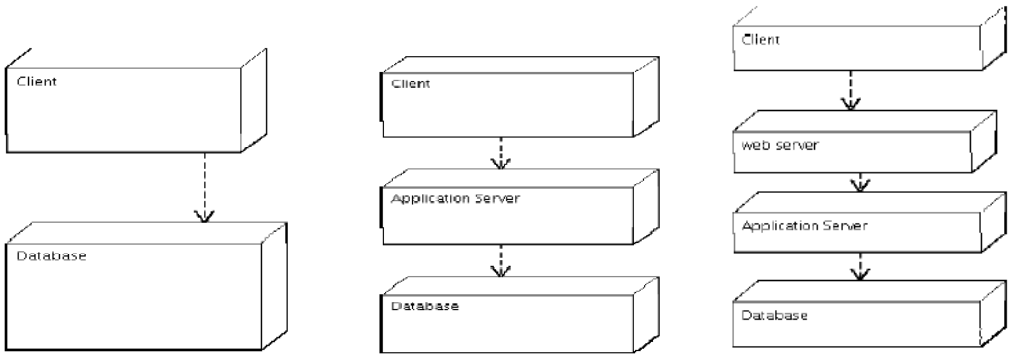
Det tiltenkte (og optimale) svaret her kan være følgende eksempel på trelagsarkitektur. Inne i applikasjonslaget, er det også angitt en generisk logisk arkitektur som er et resultat av ansvarsdrevet design. Figuren er gjennomgått på forelesning.



Andre tilstrekkelige skisser er for eksempel å gi logisk arkitektur



samt å forklare hvordan den logiske arkitekturen fordeler seg i en valgt fysisk arkitektur. Fysiske arkitekturer kan være en av følgende:



Den siste fysiske arkitekturen (web-arkitekturen) bør her være å foretrekke.

b) Man er svært opptatt av å lykkes med det nye produktet. For å fastsette kravene til IT-systemet til *Trang Hybel*, må synspunkter fra de rette interessentene (”stakeholders”) tas hensyn til. Hvilke interessenter bør være representert, og hvilken kravinnhentingsmetode vil du bruke for hver interessent? (Nevn minst 3 interessenter). Begrunn kort.

Stakeholderlisten bør minst inneholde: Kunden (*Forsikring2*), leverandøren av IT-løsningen for *Trang Hybel* (altså den leverandøren som vinner anbudsrunden), samt sluttbrukergruppen (som her er kjøperne av produktet *Trang Hybel,* nemlig studenter).

c) Hva synes du er de tre viktigste ikke-funksjonelle kravene til IT-systemet for *Trang Hybel*. Begrunn. Skisser kort hvordan du kan evaluere at disse kravene oppfylles.

Det er flere mulige alternativer her, men de mest opplagte er: Brukervennlighet for sluttbrukerne er svært viktig, siden dette er hele presentasjonen og salget av produktet *Trang Hybel*. Oppetid er også svært viktig, siden ingen salg blir gjort all den tid systemet er nede. Datasikkerhet er også viktig, siden systemet er web-basert, noe som innebærer at den delen av systemet som registrerer opplysninger som kommer under personvernloven, er desentralisert og således under mindre kontroll.

Brukervennlighet kan testes ved brukergrensesnittprototyping (i for eksempel Genova) mot relevante stakeholders. Oppetid kan stipuleres i forkant av installasjon ved å velge robuste løsninger basert på historikk, og kan sjekkes under drift for eksempel ved antall feilmeldinger pr. døgn. Datasikkerhet må ivaretas ved eksisterende teknologi og må garanteres på forhånd. Det er ikke sikkert studentene resonnerer akkurat som over, siden akkurat hvordan ikke-funksjonelle krav kan oppfylles og testes ikke er

gjennomgått i detalj, men det viktigste er at studenten forsøker å argumentere.

d) Du (som leverandør av IT-systemet) og kunden diskuterer hvilken kontraktstandard dere skal bruke. Hvilke tre kontraktstandarder kjenner du til, og hva er den viktigste forskjellen mellom dem?

Det er gjennomgått tre kontraktstandarder: IKT Norge sin standardavtale, Statens standardavtale (ved DIFI—Direktoratet for forvaltning og IKT), og PS2000 (ved Dataforeningen). Førstnevnte er laget for å ivareta leverandørers interesser, annennevnte er laget for å ivareta kunders interesser, mens PS2000 er en avtale som er ment å representere både kunde og leverandør. PS2000 er således ment å være konfliktløsende, siden et problem med ensidige kontrakter er at de kan fremme konflikt.

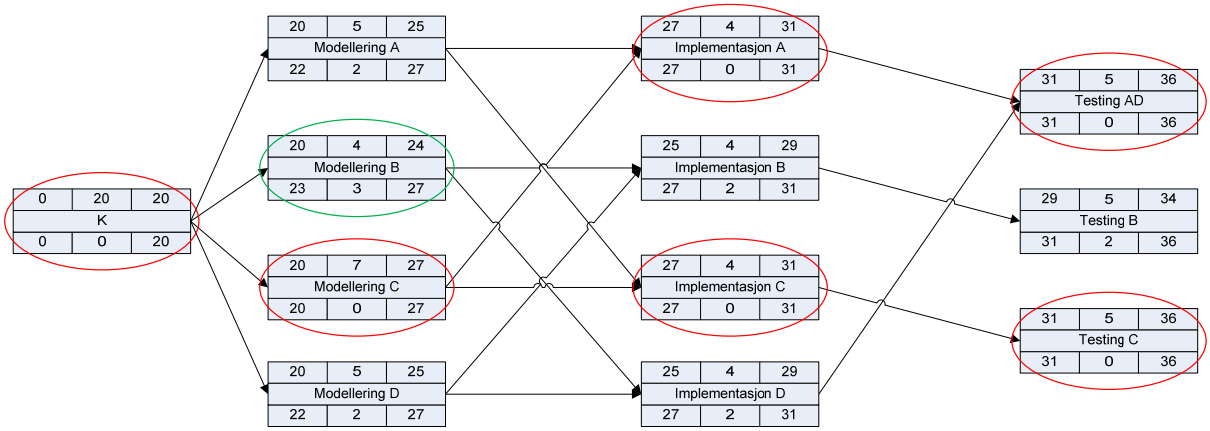
Oppgave 2 Prosjektstyring (20%)

a) IT-ansvarlig i *Forsikring2*, som tidligere har vært med på en del mindre men svært vellykkete IT-prosjekter, sier at han tror det ville ta deg og ditt team omlag 9000 timesverk å lage første versjon av systemet for *Trang Hybel*. Han begrunner dette med at de prosjektene han var med på, var omtrent en tredjedel av størrelsen, og at de tok rundt 3000 timesverk. Du har bedt om å få se kravspesifikasjonene til de prosjektene han snakker om. Noen av spesifikasjonene er svært saklige og bruker ikke flere ord enn nødvendig, mens andre har lange utlegninger, bl.a. om de ønskete positive følgene av det ferdige systemet. Nevn noen faktorer ditt team må være oppmerksomme på, slik at dere estimerer tidsbruk så realistisk som mulig i dette prosjektet.

De to viktigste faktorene her er faren ved kompleskistetseksplosjon og bias framprovosert ved irrelevant informasjon. Man bør derfor ha en sunn skepsis til anslagene til IT-ansvarlig i *Forsikring2*, fordi det ligger en fare i at man overfører erfaringer fra mindre og mellomstore prosjekter til store prosjekter, siden de vesentlige forskjellene (bla mhp produktivitet og risiko-eksplosjon) ikke tas nok hensyn til. Her er dette tydelig, siden han gjør en lineær ekstrapolasjon av tidsbruk fra små prosjekter til stort prosjekt. Flere av spesifikasjonene fra hans erfaringsgrunnlag kan dessuten ha bidratt til at han har feilaktig klassifisert størrelsen til prosjektene, siden noen av spesifikasjonene har irrelevant informasjon og noen ikke.

b) Anta følgende: Du vet at kravinnhenting er det første som må gjøres. Siden må kravene modelleres i use cases. Men så snart et use case er ferdig beskrevet, kan en mer formell modellering gjøres av use caset i form av sekvens- og klassediagrammer, uavhengig av om de andre use casene er ferdige. Ved hjelp av planning poker, har ditt team estimert at kravinnhentingen K vil ta 20 enheter tid. Dere antar på dette stadiet at dere kommer til å få 4 overordnete funksjonelle krav A, B, C, D. Dere estimerer at det vil ta 5 enheter tid å lage et use case samt modeller for A, at det vil ta 4 enheter for B, 7 enheter for C og 5 enheter for D. Når use case og modellene for A og C er ferdige, kan kode, databaseskjemaer og brukergrensesnitt genereres for disse. Dette vil ta totalt 4 enheter. Når modellene for B og D er ferdige, kan kode, databaseskjemaer og brukergrensesnitt genereres for disse. Dette vil også ta totalt 4 enheter. Testing skal gjøres sammen for A og D (totalt 5 enheter), men skal gjøres enkeltvis for B (5 enheter) og C (5 enheter). Tegn et nettverks-diagram, og angi de aktivitetene som ligger på kritisk vei, samt den aktiviteten med størst slakk.

PERT-diagrammet under er et optimalt svar. Kritiske aktiviteter er sirklet i rødt og den aktiviteten med størst slakk er sirklet i grønt.

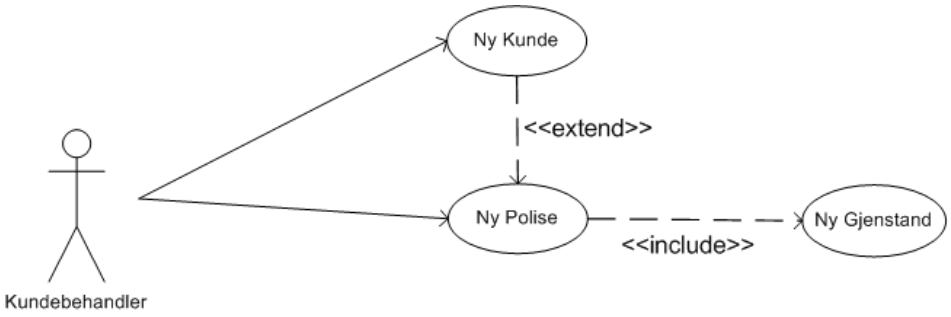


Oppgave 3 Modellering (40%)
Du skal nå modellere den delen av systemet som registrerer et salg av en polise under *Trang Hybel*. Systemet skal holde oversikt over kunder, poliser og tilhørende gjenstander (bolig, mobiltelefon, sykkel, og lignende) som er forsikret i *Forsikring2*.

For enkelhets skyld skal du anta følgende:

(1) Kunder er identifisert ved *fødselsnummer*.
(2) Alle mulige gjenstander som kan forsikres kan identifiseres ved hjelp av en identifikator som heter *gjenstandID*. Du trenger ikke spesifisere hvordan denne identifikatoren lages.
(3) Alle gjenstander som forsikres kan ha kun en polise som er assosiert med kun en kunde.
(4) En polise kan forsikre kun en gjenstand.

a) Vi tenker oss at denne delen av funksjonaliteten til *Trang Hybel* kan modelleres som tre Use Cases (bruksmønstre): ”Ny Polise”, ”Ny Kunde” (som aktiveres dersom kunden som ønsker en polise ikke eksisterer i systemet fra før) og ”Ny Gjenstand”. Primær aktør er *Kundebehandler*. Lag et UML Use Case diagram med disse tre bruksmønstrene. Spesifiser evt. avhengigheter mellom dem ved hjelp av *include* og/eller *extend*. Gjør rede for antakelser.



Siden "Ny Gjenstand" er en del av hovedflyt er det naturlig en "include" i "Ny Polise".
Siden "Ny Kunde" er et unntakstilfelle (vi antar at i hovedflyt så eksisterer kunden)
så kan "Ny Kunde" anses som en "extend" på "Ny Polise".

Hvorvidt "Ny Gjenstand" og/eller "Ny Kunde" også kan initieres fra aktøren (og da skal de evt. ha egne piler fra aktøren) avhenger av om man tenker seg at dette er egen, selvstendig funksjonalitet som aktøren kan bruke eller ikke. Begge deler er OK løsninger. Men siden vi her tenker oss at "Ny Kunde" er unntakstilfelle kan en kanskje tenke seg at det kan initieres fra Kundebehandler før man oppretter poliser, mens det er ikke like naturlig for "Ny Gjenstand". Derfor har jeg valgt å vise direkte kommunikasjon mellom aktør til "Ny Kunde" men ikke til "Ny Gjenstand". En "A"-oppgave her forventer jeg at har med litt diskusjon av denne typen.

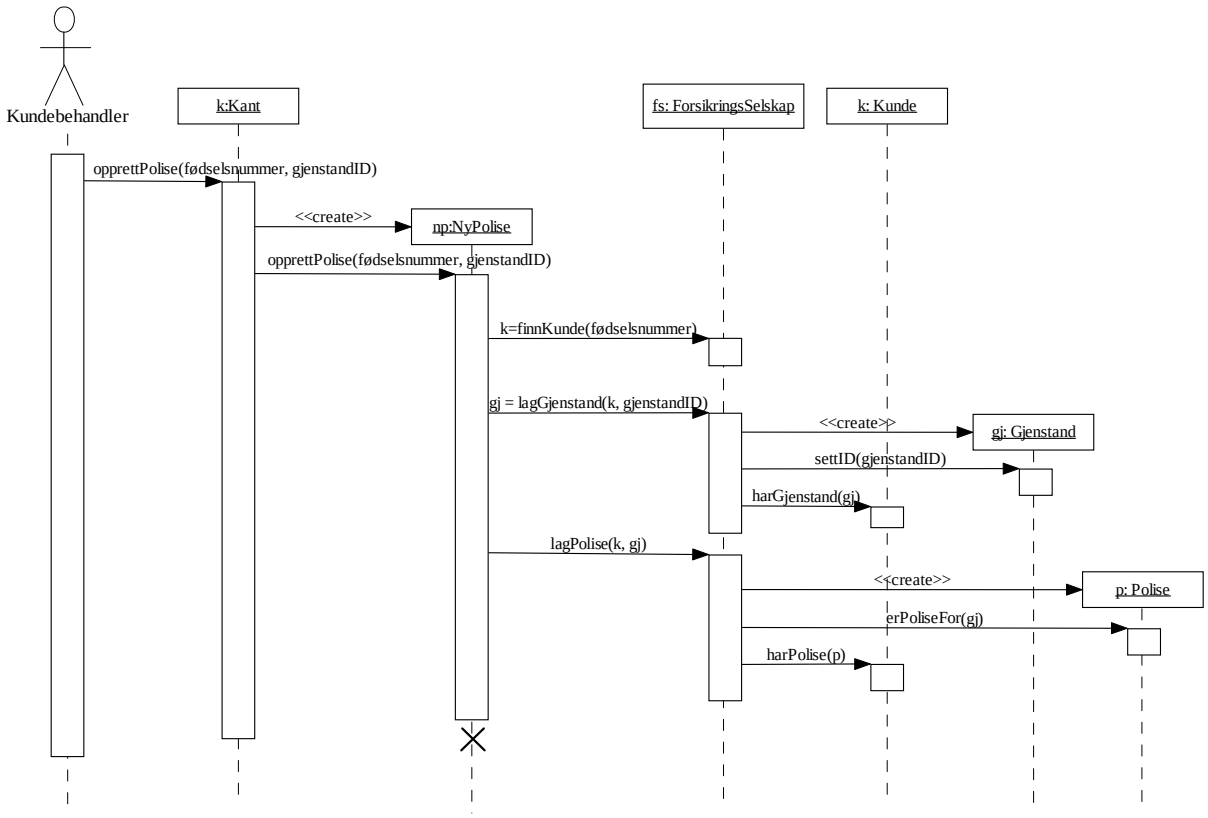
b) Lag en tekstlig spesifikasjon av bruksmønsteret ”Ny Polise” med hovedflyt og alternative flyt, prebetingelser og postbetingelser. ”Ny polise” skal registrere en ny polise på en spesifisert gjenstand for en spesifisert kunde. Du skal anta at i normaltilfellet så er kunden allerede registrert men at gjenstanden *ikke* er registrert i systemet. Dersom kunden ikke eksisterer må ny kunde opprettes. Dersom gjenstanden allerede er registrert må man sjekke at det ikke allerede eksisterer en polise på gjenstanden. Gjør rede for evt. andre antakelser du gjør.

```
Aktør: Kundebehandler
Prebetingelser: Ingen
Hovedflyt:
1. Kundebehandler oppgir fødselsnummer og gjenstandID
2. Systemet finner kunden
3. Systemet oppretter gjenstanden (use caset ”Ny gjenstand” inkluderes)
4. Systemet oppretter en ny polise for kunden for gjenstanden

Alternativ 1, steg 2: Kunden er ikke registrert
A1.1 Use caset ””Ny Kunde” utføres
   Use caset fortsetter fra steg 3.
Alternativ 2, steg 3: Gjenstanden eksisterer fra før
A2.1: Systemet sjekker om det eksisterer polise på gjenstanden fra før
      A.2.1.1     Det eksisterer polise fra før:
                  Systemet gir feilmelding.
                  Use case avslutter
      A.2.1.2     Det eksisterer ikke polise fra før:
                  Use case fortsetter fra steg 4.

Postbetingelser:
1: Hvis kunde ikke eksisterte fra før: Ny kunde er opprettet
2: Hvis ikke gjenstand eksisterte fra før: Ny gjenstand er opprettet
3:  Hvis gjenstand eksisterte og hadde polise: Aktør fikk feilmelding.
Ingen nye registrering
4: Hvis polise ikke eksisterte fra før: Ny polise er opprettet
```

Under ser du et sekvensdiagram for hovedflyt for ”Ny Polise”.



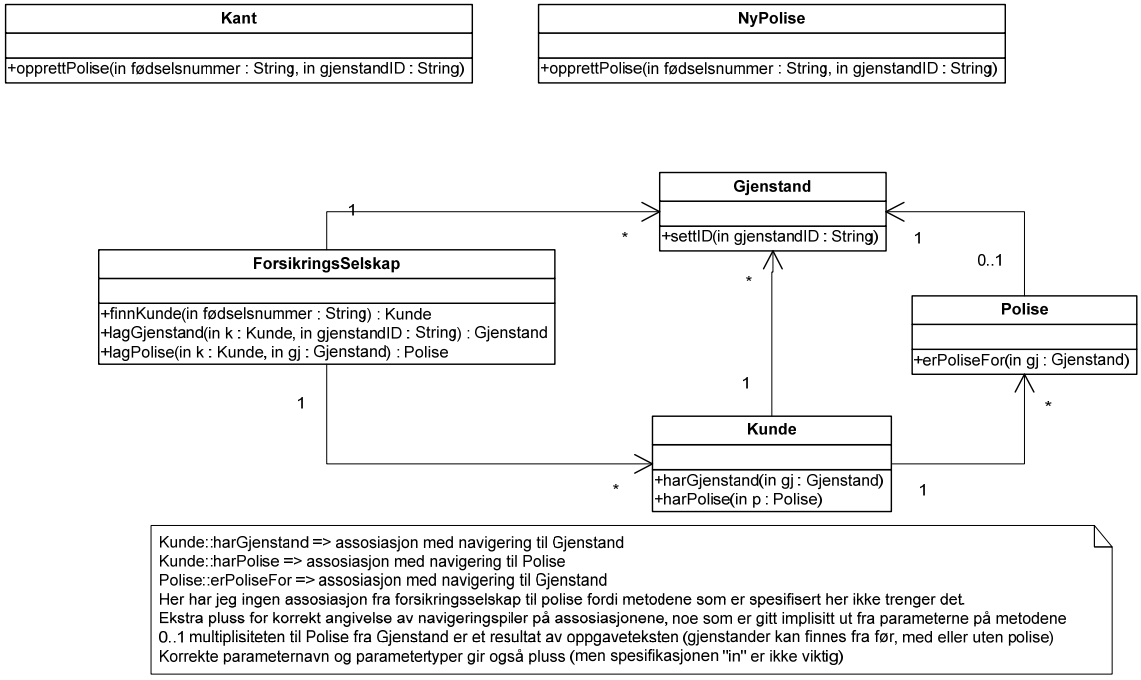
c) Hvilke forretningsobjekter finnes i sekvensdiagrammet?

```
fs: ForsikringsSelskap
k: Kunde
gj: Gjenstand
p: Polise
```

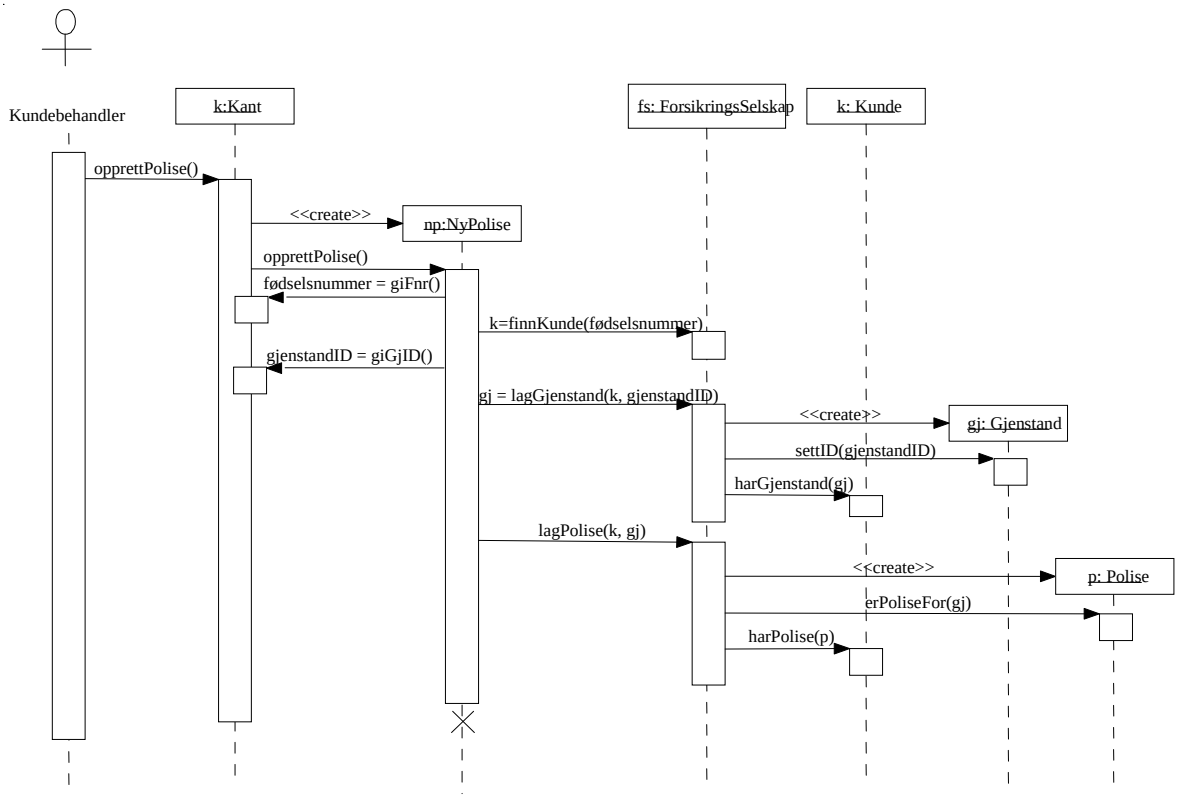
d) Gi en vurdering av i hvilken grad sekvensdiagrammet har en sentralisert eller delegert kontrollstil.

```
Her har fs: ForsikringsSelskap en del av ansvaret for å lage gjenstander
og poliser, som kontrollobjektet ikke vet detaljene i. Derfor er ikke
sekvensdiagrammet helt sentralisert, for i så fall ville ingen meldinger
utgått fra forretningsobjektene. Men man kunne selvsagt tenke seg enda mer
delegerte løsninger, hvor for eksempel også Kunde, Gjenstand eller Polise
hadde mer ansvar enn nå.
```

e) Lag et klassediagram som tilsvarer sekvensdiagrammet, hvor alle klasser og metoder samt assosiasjoner mellom forretningsobjektene med multiplisitet er spesifisert. Du trenger *ikke* spesifisere attributter eller avhengighetspiler fra kant- og kontrollobjekt.



f) Med utgangspunkt i sekvensdiagrammet, lag et nytt sekvensdiagram hvor kontrollobjektet ber om *fødselsnummer* og *gjenstandID* fra kantobjektet (i stedet for at kontrollobjektet får disse som parameter i metoden *opprettPolise*). Du trenger ikke tegne hele sekvensdiagrammet på nytt, men kun de delene hvor det blir endringer.



Oppgave 4 Testing (20%)

a) Hvorfor er det typisk flere testaktiviteter (f.eks. enhets-, integrasjons-, systemtesting) i systemutvikling?

```
From slide: Test Organization and after. (45):

There are many potential causes of failures and different testing
phases/levels focus on different  types of faults. Furthermore faults
should be found as early as possible to decrease the cost of correction,
e.g., unit testing versus system testing.
```

b) Når to komponenter som skal kommunisere med hverandre har passert enhetstestene sine, hvorfor er det viktig å gjøre integrasjonstesting?

```
From slides 47-48 (Integration Testing Failures):

We need to ensure that the interfaces between these units have been
implemented correctly, e.g., consistency of parameters, file format.
Furthermore, a unit may behave correctly with a stub/driver but not
function correctly when integrated with the other communicating units.
```

c) Hva er fordelene og ulempene ved henholdsvis whitebox- og blackbox-testing? Illustrer med eksempler på teknikker innen de to kategoriene.

From slide 40 (White-box vs. Black-box testing)

Black-box: + advantage, - drawbacks

+ Check conformance with specifications
+ It scales up (different techniques at different granularity levels)
– It depends on the specification notation and degree of detail – Do not know how much of the system is being tested – What if the software performed some unspecified, undesirable task?

White-box: + advantage, - drawbacks

+ It allows you to be confident about code coverage of testing
+ It is based on control or data flow code analysis
– It does not scale up (mostly applicable at unit and integration testing levels)
– Unlike black-box techniques, it cannot reveal missing functionalities (part of the specification that is not implemented)

Example techniques:

- White-box: all-edges coverage criterion based on control flow graphs

- Black-box: Category Partition