IN 1030

29 mars 2022

Modellering av krav

UNIVERSITETET I OSLO

Yngve Lindsjørn

ynglin@ifi.uio.no

Temaer

- Modellering av krav
- UML diagrammer (UML står for **U**nified **M**odeling **L**anguage)
  - **Use Case (Bruksmønster)**
  - Domenemodell
  - Sekvensdiagram
  - Aktivitetsdiagram

Først – Anbefalt Video

Smidig arbeidsform

https://ntnu.cloud.panopto.eu/Panopto/Pages/Viewer.aspx?id=6ce3502e-d594-4de3-8fc9-ae4500dba5b3&start=0

Funksjonelle og ikke-funksjonelle krav

- Funksjonelle krav
  - Hva systemet skal gjøre
    - Hvilke tjenester (funksjoner) skal systemet tilby?
    - Hvordan skal systemet reagere på ulike typer *input* ?
- Ikke funksjonelle krav
  - Hvordan skal systemet implementere de funksjonelle kravene?

Eksempler fra oblig4

a) Skriv minst seks funksjonelle krav som brukerhistorier for billettsystemet til Virtuoso Kino. Brukerhistoriene skal være fra *minst* tre forskjellige aktører. Sett brukerhistoriene opp i en prioritert liste basert på hva dere anser som viktigst for sluttproduktets funksjonalitet.

a) Sett opp en liste over *minst* fire ikke-funksjonelle krav som dere ønsker å stille til billettsystemet. Del opp kravene i produktkrav, organisatoriske krav og eksterne krav, og få med minst ett krav av hver type.

Gode beskrivelser av krav

er viktig for

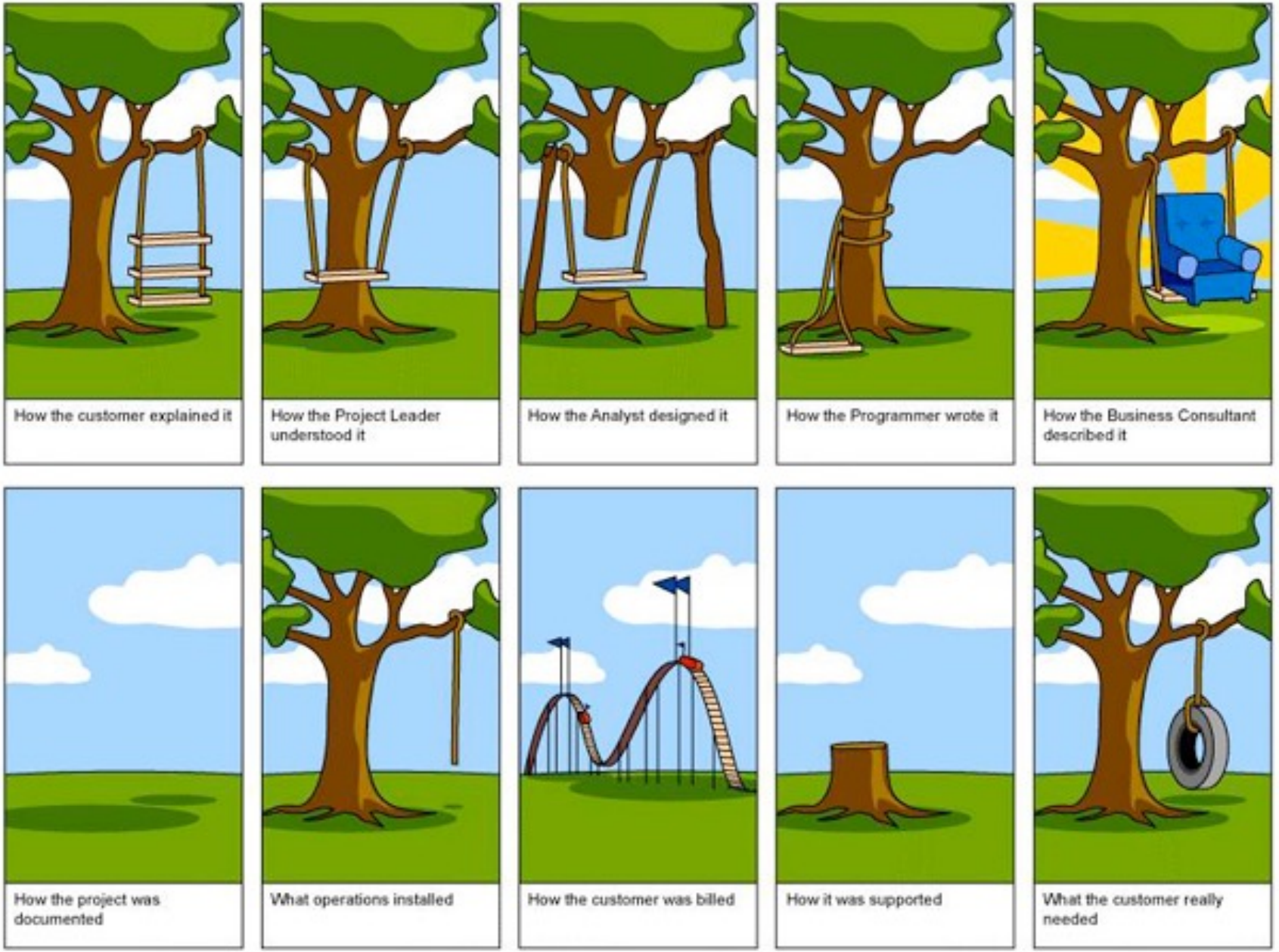
- kontrakt oppdragsgiver – leverandør
- planlegging og oppfølging
- arkitektur, design og test
- å støtte videreutvikling og vedlikehold

Kravene bør være

- forståelige (alle interessenter/stakeholders må kunne forstå kravspesifikasjonen),
- testbare (vi må kunne avgjøre om det ferdige systemet gjør det det skal), og
- sporbare (vi må vite hvilken del av koden som skal endres når det kommer nye krav).

Utfordringer i kravhåndtering

- Kommunikasjon
- Felles forståelse
  - bl.a. av når et krav er realisert?
- Sammenhengen fag-IT
  - Hvis fagsiden dominerer, kan funksjonalitet bli besluttet uten tilstrekkelig innsikt i hva som kan realiseres i systemet
  - Hvis IT dominerer, kan den tekniske sjargongen bli dominerende og dermed blir det vanskelig å få kravene riktige
- Sporbarhet fra krav til arkitektur, design og kode er viktig


How the customer explained it
How the Project Leader understood it
How the Analyst designed it
How the Programmer wrote it
How the Business Consultant described it
How the project was documented
What operations installed
How the customer was billed
How it was supported
What the customer really needed

Funksjonelle krav

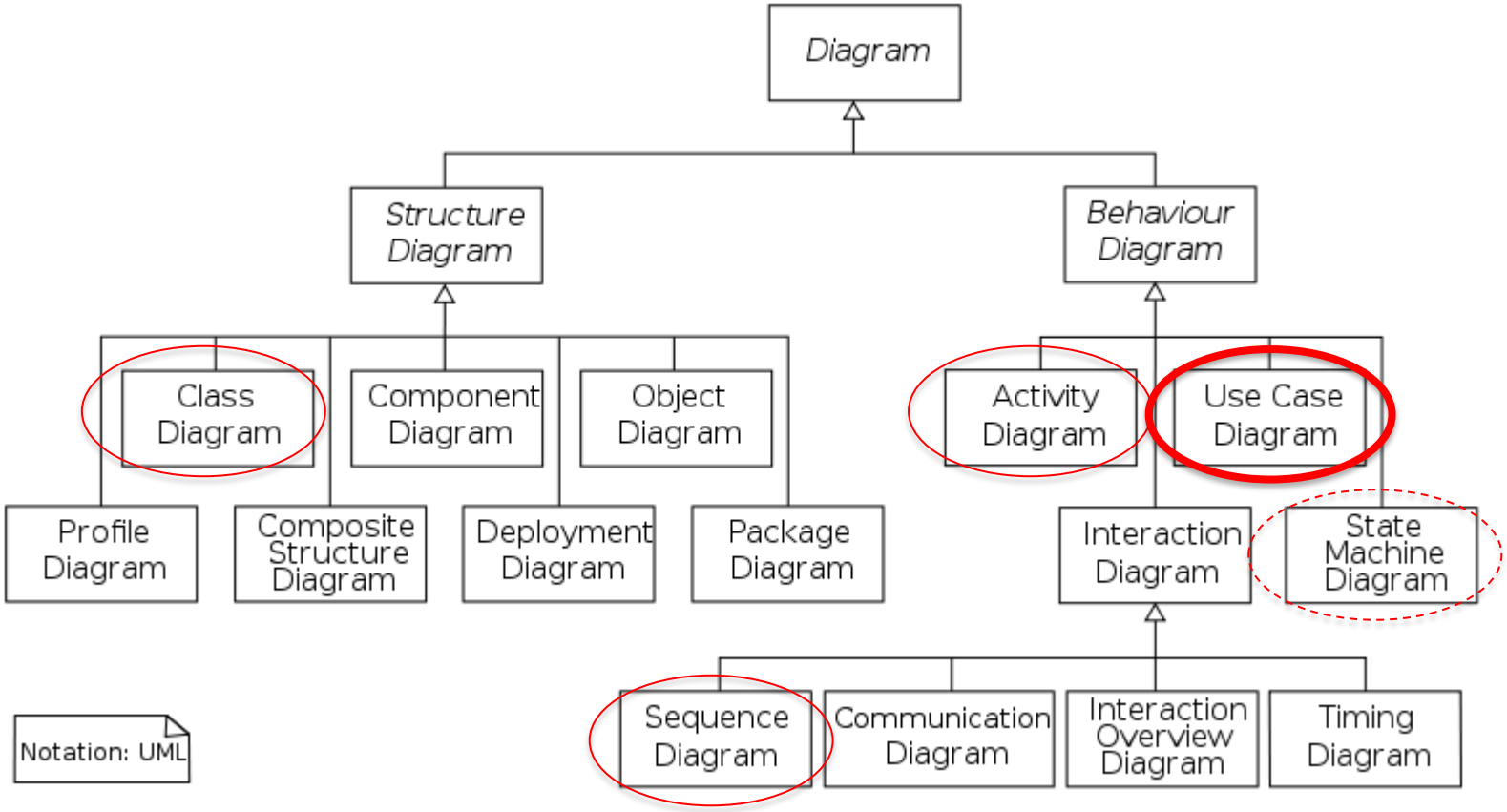
- Hva (ikke hvordan)
- Forretningskrav
  - Hva organisasjonen ønsker
- Brukerkrav
  - Hva brukerne ønsker å kunne gjøre
- Systemkrav
  - Mer detaljert definisjon av hva som skal implementeres sett fra brukernes perspektiv

Beskrive krav

- Tekst
- Strukturert tekst
  - User story (brukerhistorie)
  - Use case (brukstilfelle)
- Modeller
  - UML (Unified Modeling Language)
  - BPMN (Business Process Model and Notation)

UML – diagrammer

(de med ring rundt er de dere skal kunne)



Kilde: http://en.wikipedia.org/wiki/Unified_Modeling_Language#Structure_diagrams

Diagrammer i UML (Unified Modeling Language)

- Use case diagrammer viser systemets funksjonalitet og samspillet mellom systemet og omgivelsene (brukere, andre systemer, komponenter)
- Sekvensdiagrammer viser samspill mellom system og omgivelser og mellom de forskjellige delene av systemet (mer detaljer for hvert Use Case med objekter og metodekall)
- Klassediagrammer viser klassene i systemet og assosiasjonene mellom disse klassene
- Aktivitetsdiagrammer viser forretningsprosesser og -arbeidsprosesser

Det er disse fire diagramtypene dere skal lære i IN1030

Interessenter (Stakeholders) og aktører

Interessenter

- Oppdragsgivere (kunder)
- Brukere
- Utviklere og vedlikeholdere
- Systemeiere og forvaltere
- Andre interessenter (fagforeninger, lovgivere, andre systemer)

Se også hva er en interessent definert av Difi (direktoratet for forvaltning og IKT).

- En aktør for et system representerer en rolle som et menneske eller et annet system som har et mål med systemet.

- Det er ofte flere interessenter enn aktører, og en aktør er som oftest også en interessent.

Use case modellering
Identifiser aktører

- En aktør representerer en rolle som et menneske eller et annet system når det kommuniserer med dette systemet
- En aktør kommuniserer med systemet via ett eller flere use case
- En aktør er ofte også en interessent (stakeholder), men det finnes en del interessenter som ikke er aktører

Interessenter

Oppgave:

Finn interessenter for et system for registrering og behandling av lånesøknader

Use case modellering
Identifiser brukere → aktører

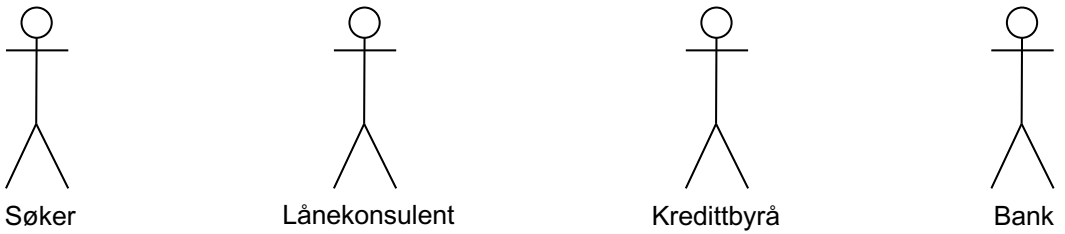
Oppgave:

Finn aktører for et system for registrering og behandling av lånesøknader, både **primære** aktører (har et eget mål), og **sekundære** aktører (trengs for å oppfylle de primære aktørenes mål)

Use case modellering
Identifiser brukere → aktører

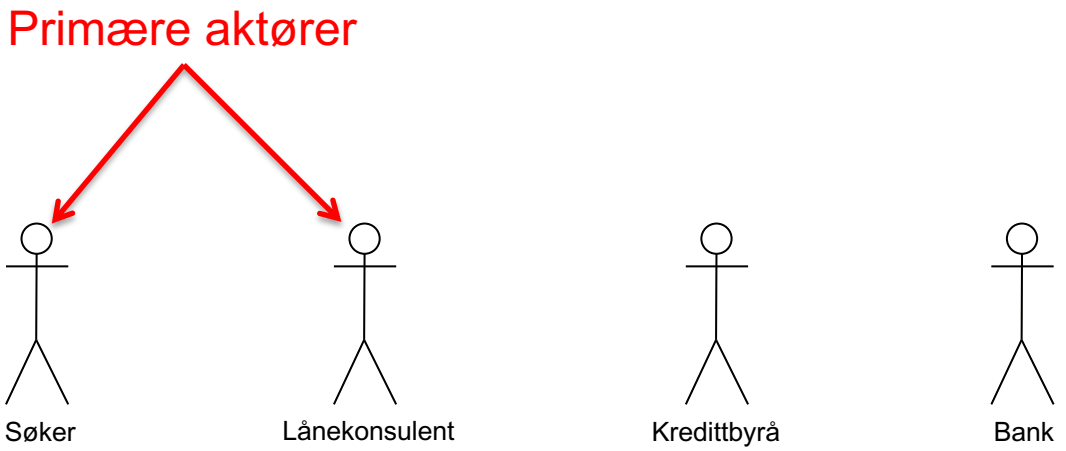
Eksempel:

Aktører for et system for registrering og behandling av lånesøknader



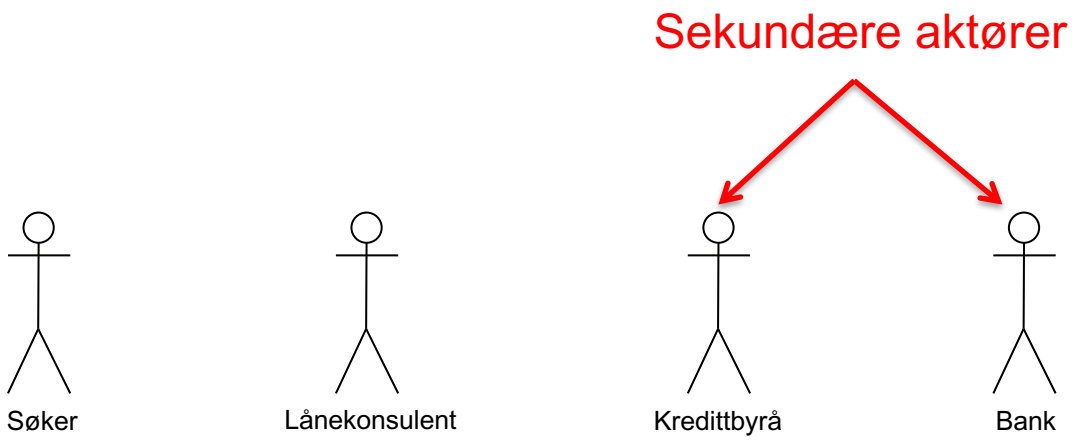
Use case modellering
Identifiser primære aktører

- Primære aktører har egne mål, dvs. de initierer use case (en eller flere) som oppfyller deres mål.



Use case modellering
Identifiser sekundære aktører

- Sekundære aktører har ikke egne mål, men er nødvendige for å realisere målene til de primære aktørene



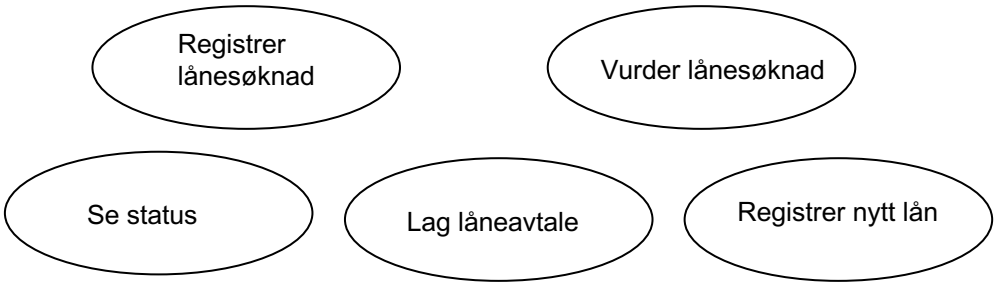
Finne aktører

- I workshop
  - Brainstorming
- Fra tekstlige krav
- Blant prosjektets interessenter
- Spørsmål
  - Hvem skal bruke systemet?
  - Hvem skal administrere systemet?
  - Hvem
    - tilbyr informasjon til
    - bruker informasjon fra, eller
    - fjerner informasjon fra systemet?
  - Hvilke eksterne ressurser skal systemet bruke?
  - Hvilke andre systemer skal kommunisere med dette systemet?

Use case modellering: Identifiser aktørenes mål → Use case

- Et use case beskriver hvordan systemet oppnår et mål av verdi for en aktør
  - En historie
  - Et komplett use case består av flere ulike hendelsesforløp (flyt)
- Et use case beskriver en komplett funksjonell enhet
  - One person – one place – one time
- Et use case er testbart

Eksempel: Use cases for lånesystemet

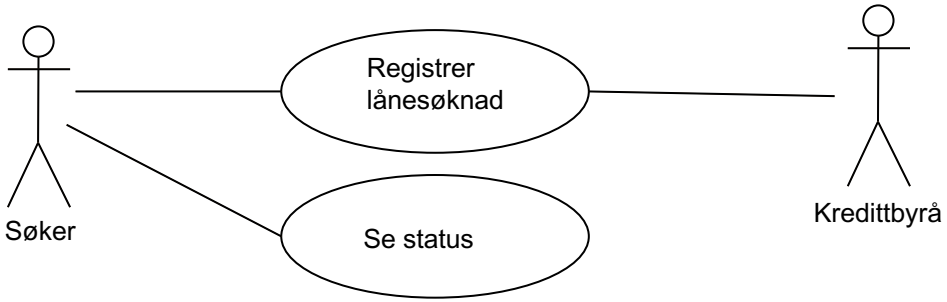


Finne use case

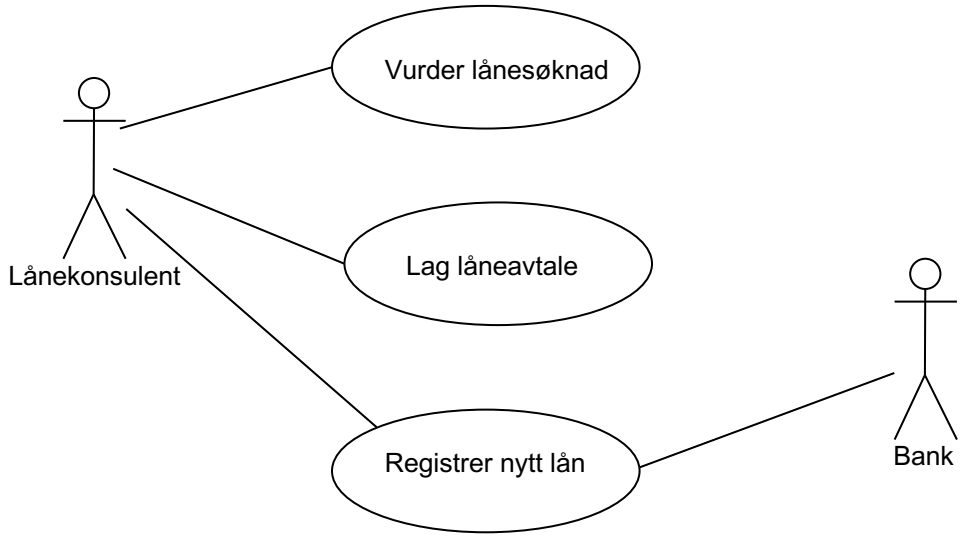
- Kan bruke de samme metodene som brukes for å finne aktører
  - F.eks. gjennomføre workshops, gjerne i flere runder
- Fra prosessmodeller over forretnings- og arbeidsprosesser
- Spørsmål

  For hver aktør:
  - Hvilke mål ønsker aktøren å oppnå med bruk av systemet?
  - Hvilke resultater vil aktøren oppnå med bruk av systemet?
  - Hva er de viktigste oppgavene som aktøren ønsker at systemet skal kunne utføre?
  - Vil aktøren skape, lagre, endre, lese eller slette data i systemet?
  - Vil aktøren ha behov for å informere systemet om eksterne endringer?
  - Har aktøren behov for å bli informert om hendelser i systemet?

Use case modellering:
Tegn use case diagram – aktør søker



Tegn use case diagram – aktør Lånekonsulent



Eksempel – eksamensoppgave INF1050 - 2012

Oppgave 2 – Modellering av en nettbank (40 %)

Du skal lage en modell for et program som skal implementeres for en nettbank. I bankens system skal en kunde først logge seg inn i nettbanken med brukernavn og passord. Følgende tabell beskriver funksjoner som skal være tilgjengelige etter vellykket innlogging:

#	Funksjonelle krav
1	Systemet må kunne gi en oversikt over alle kontoene kunden har i nettbanken. I oversikten skal det gis saldo for hver konto som er tilgjengelig i nettbanken.
2	Ved å trykke på et kontonavn eller kontonummer skal det gis en oversikt over alle transaksjonene siste måned for denne kontoen. Detaljene i en transaksjon viser dato for transaksjonen, en forklarende tekst, beløp og saldo på kontoen etter transaksjonen.
3	Systemet må kunne gi en oversikt over alle transaksjonene for en gitt konto for et gitt tidsintervall (for eksempel siste år).
4	Systemet må kunne betale en regning fra en gitt konto ved bruk av KID-nummer.
5	Systemet må kunne legge inn en ny betalingsmottaker som fast mottaker for en gitt konto.
6	Systemet må kunne gi en oversikt over alle faste betalingsmottakere for en gitt konto, samt endre eller slette informasjon om en betalingsmottaker.

a) Lag et bruksmønster-diagram (use-case diagram) der du inkluderer alle bruksmønstrene som er nødvendige for å implementere kravene i tabellen over.

Detaljert beskrivelse av hovedflyt - Registrer lånesøknad

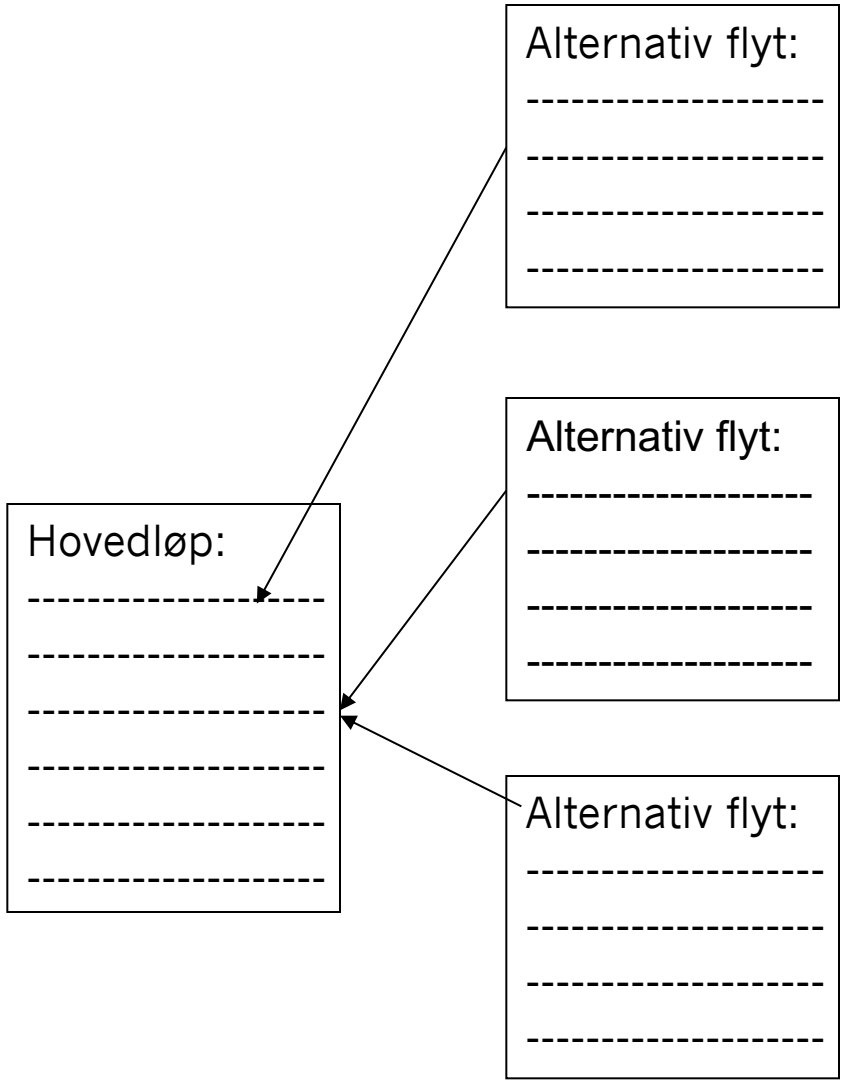
1. Søkeren fyller ut en lånesøknad
2. Systemet validerer informasjonen i lånesøknaden
3. Systemet henter søkerens kontohistorie med banken
4. Systemet innhenter kredittrapport for søkeren fra et kredittbyrå
5. Systemet beregner søkerens kreditt score basert på kredittrapport og kontohistorie
6. Systemet informerer søkeren om at søknaden er mottatt og blir vurdert
7. Systemet setter status på lånesøknaden til ”Initiell kredittsjekk ferdig”
8. Systemet legger lånesøknaden i oppgavelisten til en lånekonsulent

Tilleggsinformasjon

- Input
  - Lånesøker må oppgi: Navn, adresse, telefon, fødselsdato, arbeidsgiver, årslønn og samlet gjeld
- Forretningsregler
  - Eksempel: Lån gis bare til personer med fast jobb
- Ikke-funksjonelle krav
  - Brukervennlighet
    - Ny bruker skal kunne registrere søknad på mindre enn 10 min.
    - Mindre enn 1 av 10 brukeres skal avslutte registreringen uten å fullføre.
  - Ytelse
    - Systemet skal håndtere inntil 100 samtidige brukere

Alternativ flyt

- Use casene kan oppnå sitt mål på flere måter, og kan feile på flere måter, så detaljerte use case har også alternative flyt.
- Alternative flyt (blant annet feilsituasjoner) er viktige da det ofte er mer ”uenighet” blant prosjektets interessenter om hva som skjer i de tilfellene enn for hovedløpet.
- Ethvert steg i hovedløpet kan være utgangspunkt for et alternativ flyt.



Use case ”Registrer lånesøknad”- Alternative flyt

Hovedløp:

1. Søker fyller ut en online lånesøknad
2. Systemet validerer informasjonen i lånesøknaden
3. Systemet henter søkerens kontohistorie med banken fra kontosystemet
4. Systemet innhenter kredittrapport for søkeren fra et eksternt kredittbyrå
5. Systemet beregner søkerens kreditt score basert på kredittrapport og kontohistorie
6. Systemet informerer søkeren om at søknaden er mottatt og blir vurdert
7. Systemet setter status på lånesøknaden til ”Initiell kredittsjekk ferdig”
8. Systemet legger lånesøknaden i oppgavelisten til en lånekonsulent

Alternativ flyt 1, steg 2: Informasjonen i lånesøknaden er ikke komplett og korrekt

A1.1. Systemet returnerer lånesøknaden til søker for ytterligere utfylling.

A1.2. Systemet setter lånesøknadens status til ”Avventer”.

Alternativ flyt 2, steg 3: Det er ikke tilstrekkelig kredittinformasjon om søkeren (kredittrapport er ikke tilgjengelig eller er for dårlig)

A2.1. Systemet sender beskjed til søker om at søknaden er avslått

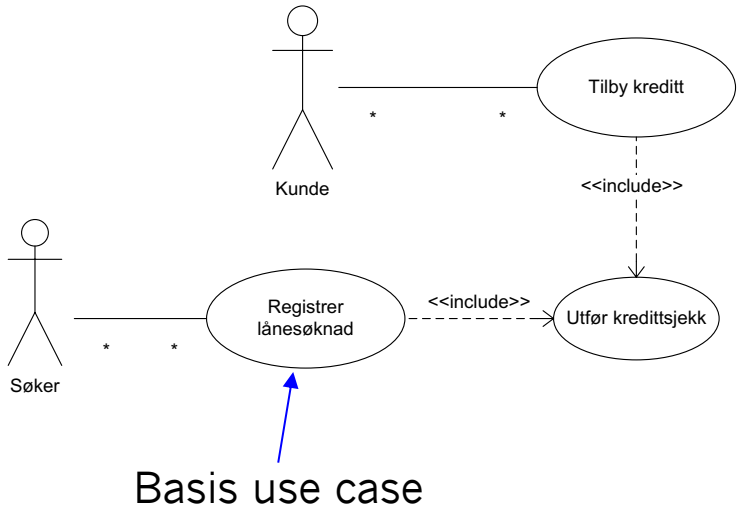
A2.2. Systemet setter lånesøknadens status til ”Avslått”.

Relasjoner mellom og use case – Extend og Include relasjonen

- Include-relasjonen: Et use case kan være en del av ett flere andre use case.
- Extend-relasjonen: Et use case som beskriver tilleggsoppførsel som utføres under gitte omstendigheter

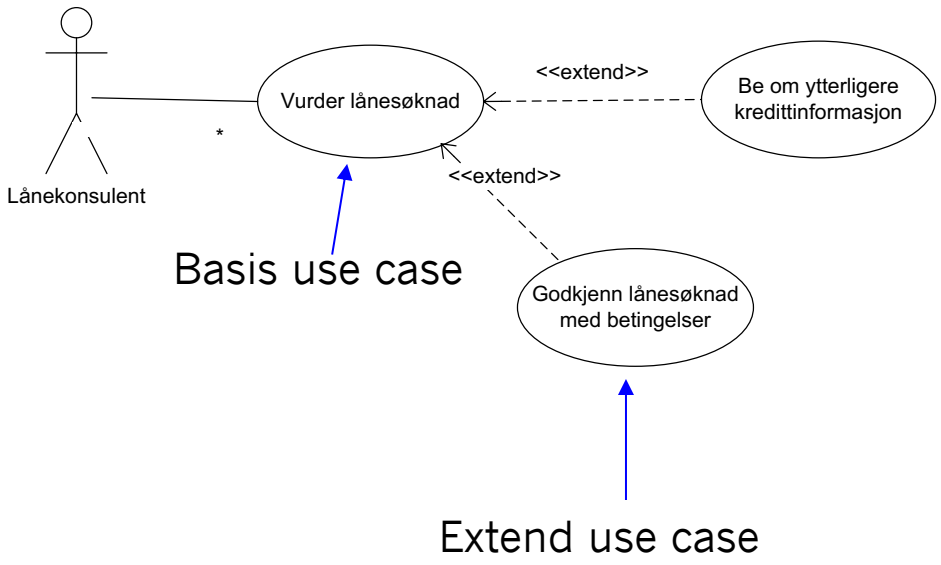
Include-relasjonen

- To eller flere use cases kan ha en felles del (noen like steg). Denne delen kan da legges ut i et eget use case som disse use casene kan inkludere.
- Basis use caset vet hvilke use case det inkluderer



Extend-relasjonen

- Alternativ oppførsel som utføres i noen tilfeller kan skrives som eget use case som utvider (extends) et annet
- Extend use case beskriver hvordan oppnå et tilleggsresultat – utvider funksjonaliteten
- Basis use caset kjenner sine extend use cases
- Bruk av alternativ flyt vs. bruk av extend use case:
  - Alternativ flyt beskriver hva som skjer ved avvik i normal flyt, mens
  - Extend use case beskriver hvordan oppnå tilleggsresultat.

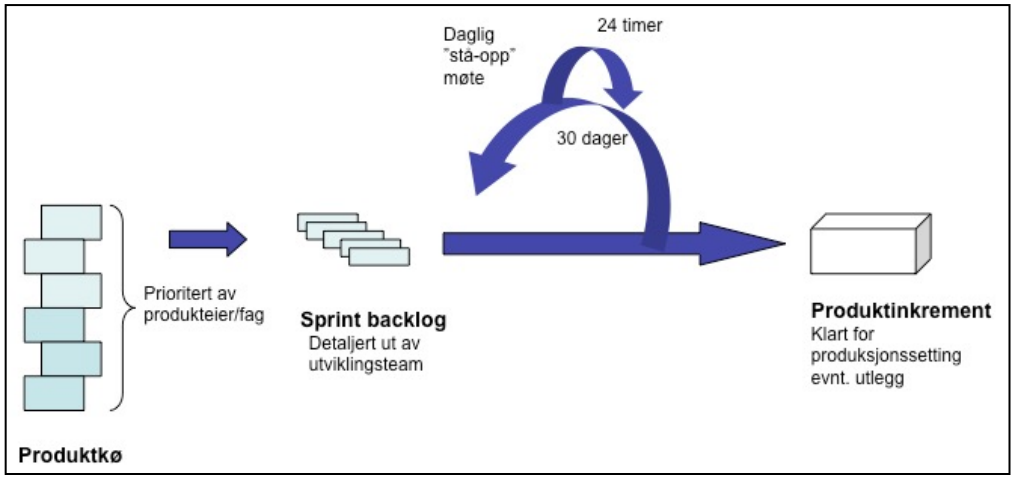


Use case vs. smidig utvikling

- I smidig utvikling jobber produkteier ofte sammen med utviklere i samme team
- Det er mindre behov for detaljerte beskrivelser av krav og ofte brukes user stories (en ”lett” versjon av use case)

  Eks: **Som** lånekonsulent
  ønsker jeg å kunne vurdere lånesøknader
  slik at jeg kan gi en riktig og rask vurdering

- Krav utvikles underveis og beskrives ”on demand”
  - ❖ Først tilstrekkelig for prioritering i produktkøen
  - ❖ Så tilstrekkelig for prioritering i sprint backloggen



Anbefalt video:
Brukerhistorier og utviklingsoppgaver

https://ntnu.cloud.panopto.eu/Panopto/Pages/Viewer.aspx?id=515fcbbb-122f-4c28-9e71-ae4a00d5a7fc&start=0

Use case vs. user stories

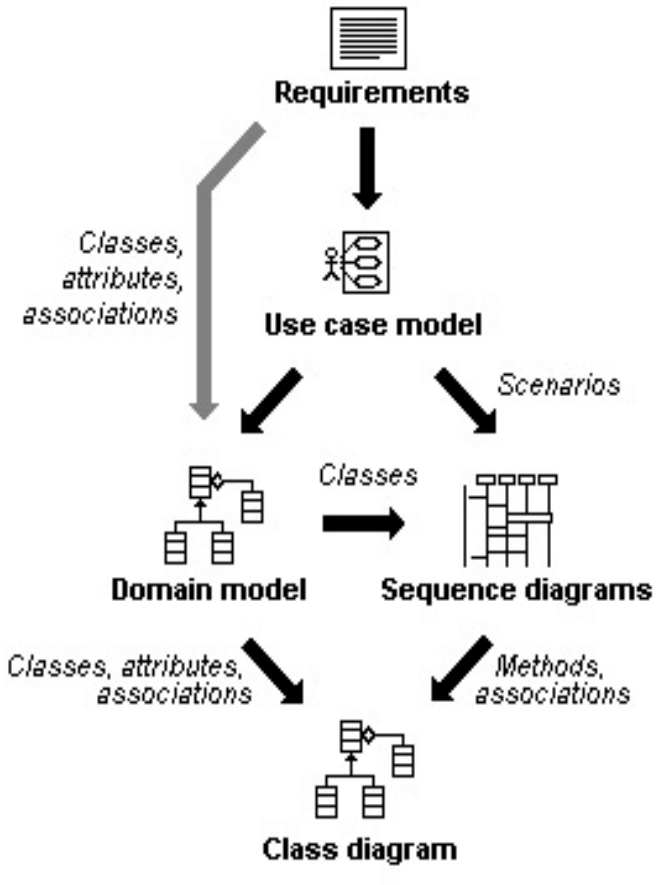
- Likheter. Begge viser
  - Hvem som skal bruke systemet
  - Hva de skal gjøre med det
  - Hvorfor de skal gjøre det
- Forskjeller
  - Omfang, kompletthet, livslengde, hensikt
  - User stories er godt egnet for å finne krav og bruke disse i smidig utvikling i samarbeid med produkteier/kunde
  - Use case er mer detaljert, har flere bruksområder videre i prosjektet og er mer egnet som dokumentasjon
- Men, det er en flytende overgang mellom dem.

Use case i prosjektplanlegging

- Planlegg hvilke use case som skal implementeres i hvilke iterasjoner av prosjektet:
  - Implementer use casene i henhold til hvor viktige de er og/eller hvor vanskelige de antas å være å implementere.
  - Hovedflyt implementeres først, deretter alternativene.
  - Estimer hvor mange use case (eller hendelsesflyt og alternative flyt) som kan implementeres i en iterasjon.

Use case i design

- Hendelsesflyt i use casene detaljeres ut i sekvensdiagram
- Domenemodellen utvides til klassediagram med systemklasser

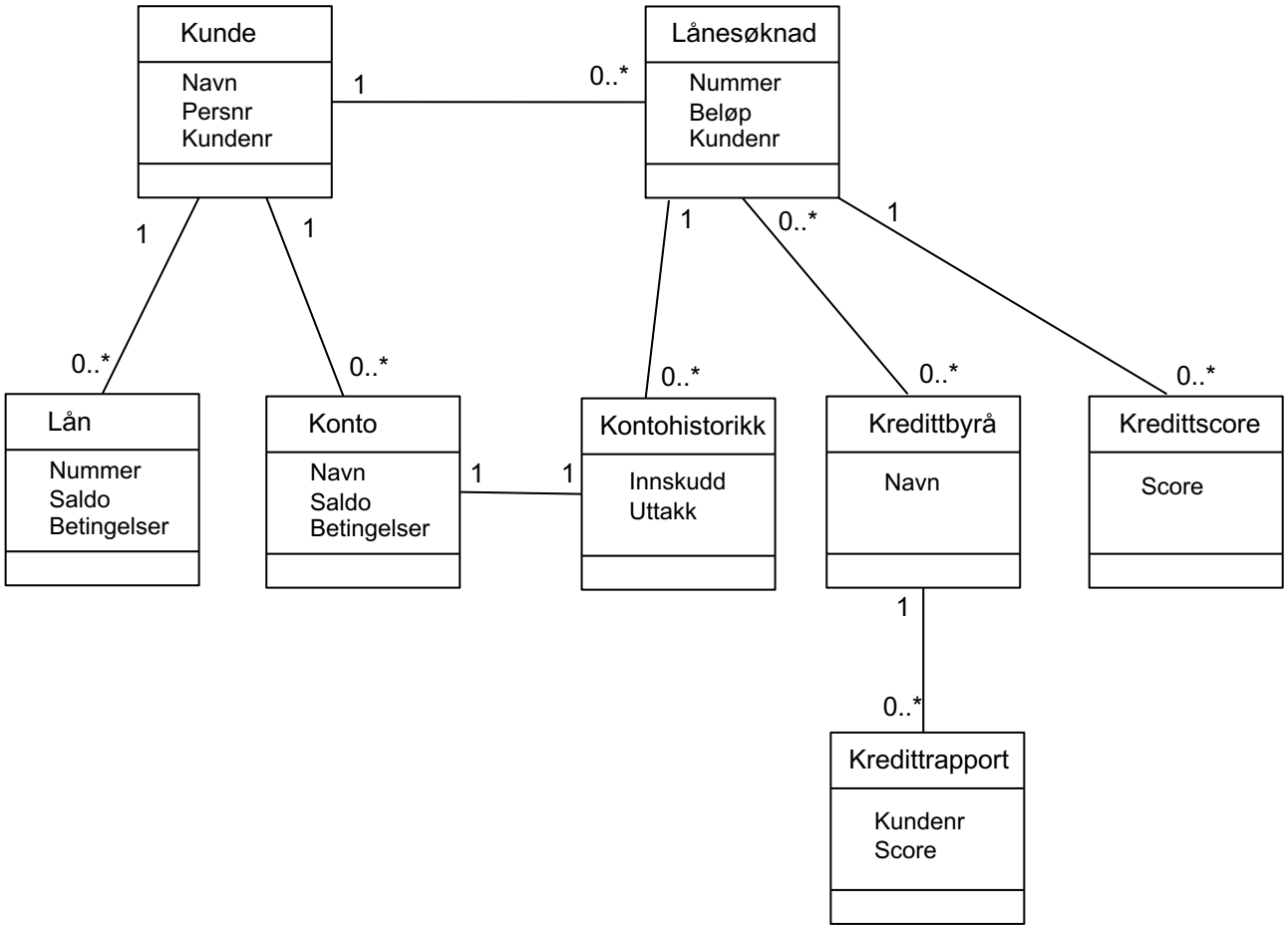


Domenemodell

- UML klassediagrammer uten metoder

- Domenemodellen viser objekter i problemdomenet.
- Hensikten med domenemodellen er å forstå objektene og få en oversikt over terminologi.
- Domenemodellen er nyttig i forbindelse med use case modellering fordi:
  - Domenemodellen viser informasjonen om objekter i use casene.
  - Den er et viktig verktøy for å sjekke at use casene er beskrevet med riktig detaljeringsnivå.

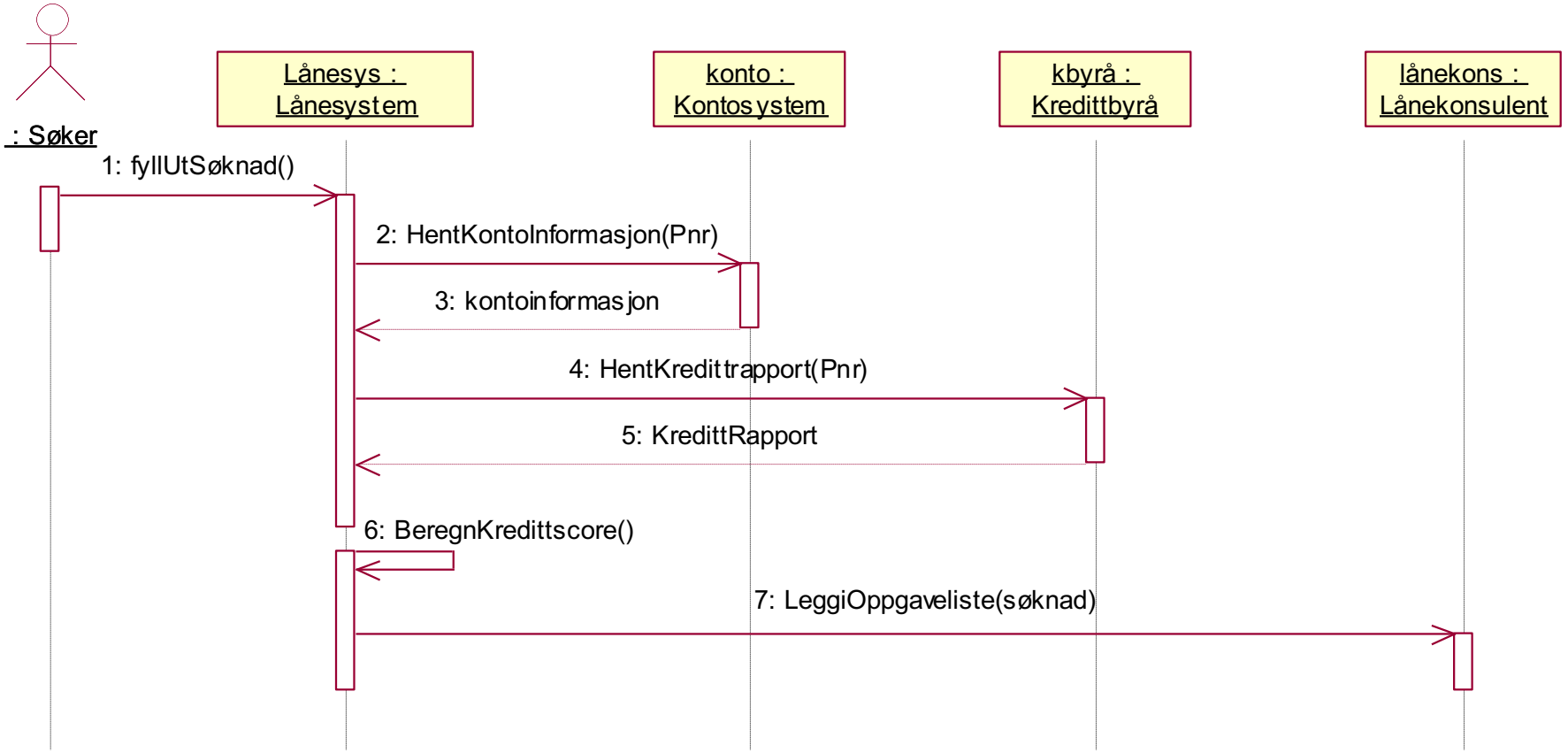
Domenemodell - eksempel



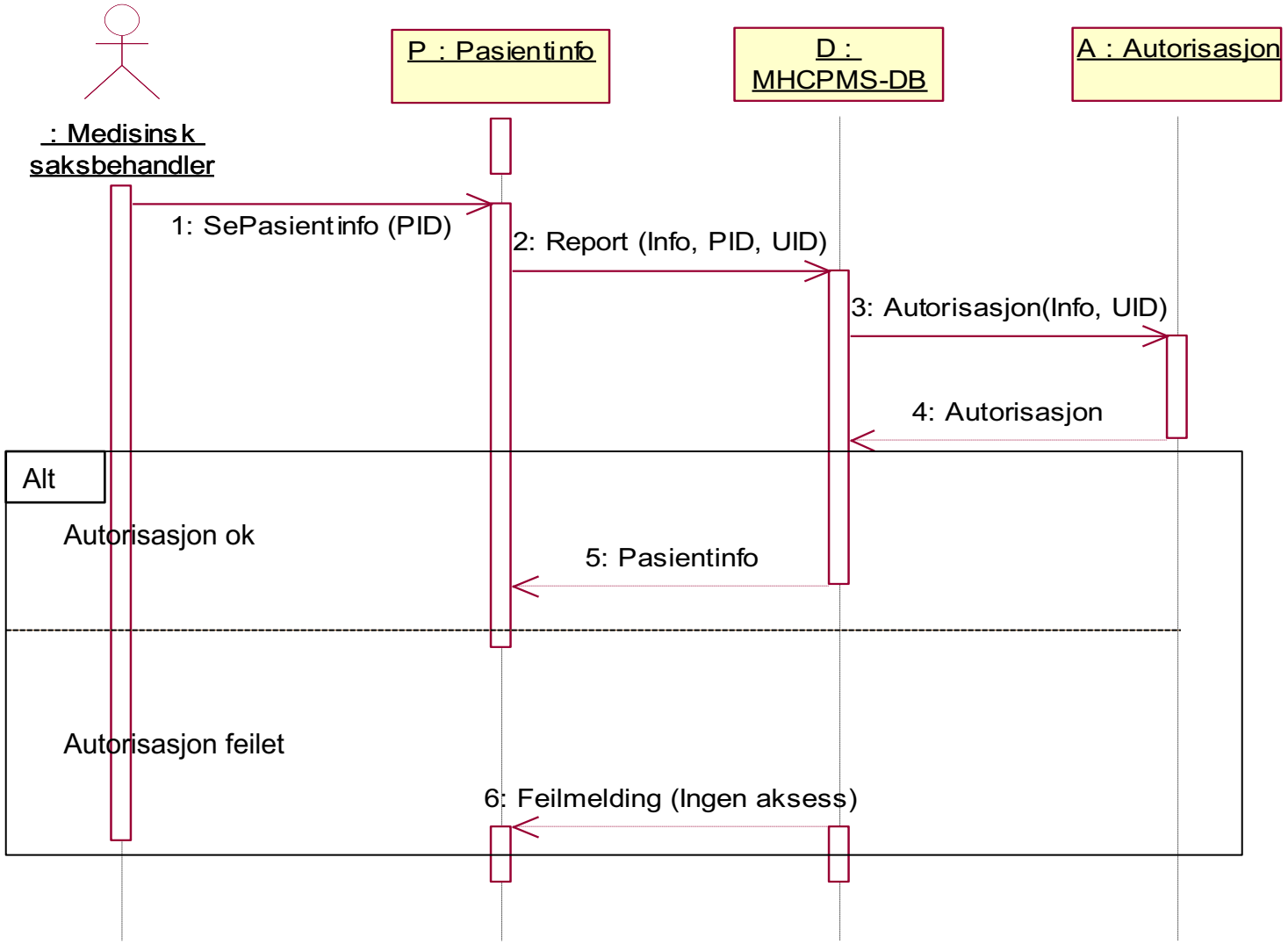
Sekvensdiagrammer

- Et flyt i et use case kan modelleres med sekvensdiagrammer.
- For hvert use case lages typisk sekvensdiagram for hovedflyt og for hyppig forekommende alternative flyt.
- Stegene (sekvensene – se tekstlig beskrivelse) i et use case vises som meldinger som sendes mellom objektene ved kall på objektenes metoder.

Sekvensdiagram “Registrer lånesøknad – hovedflyt”



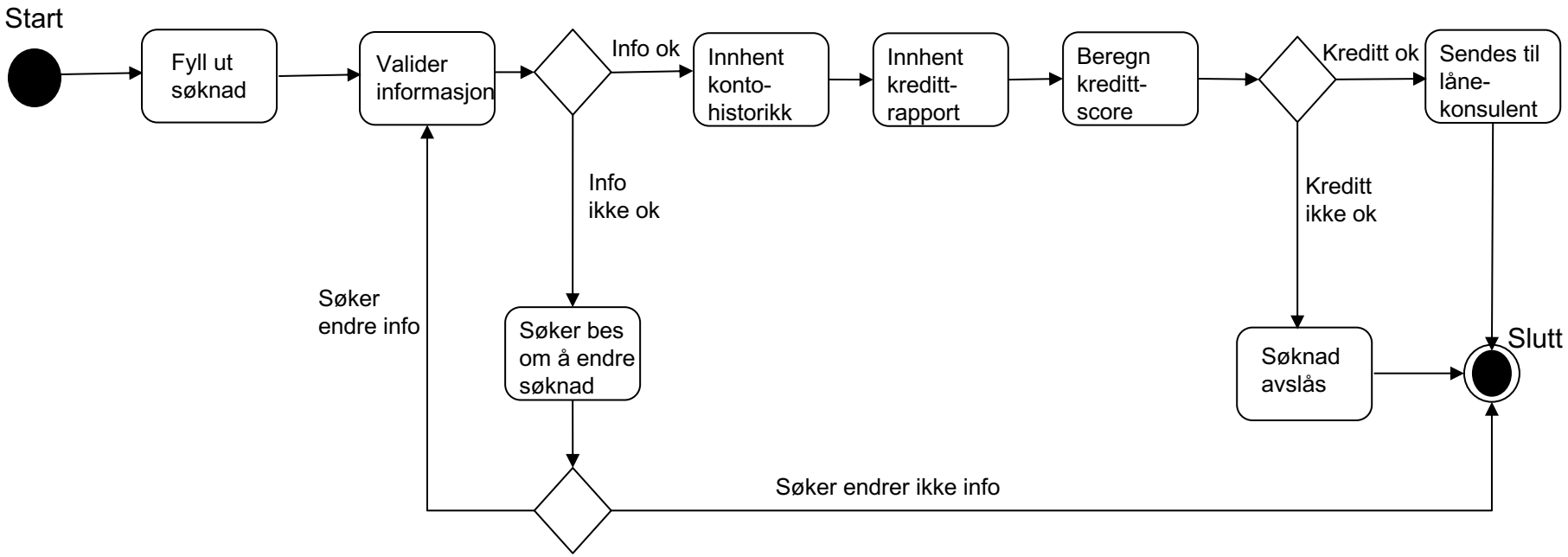
Sekvensdiagram “Se pasientinfo”



Aktivitetsdiagrammer

- Et aktivitetsdiagram kan grafisk representere hendelsesflyten i et use case.
- Stegene i use casene vises som aktiviteter  (rektangel)
- Beslutninger underveis vises som  (diamant)
- Aktivitetsdiagrammer og sekvensdiagrammer brukes noe overlappende, men sekvensdiagrammer er typisk mer kodenært mens aktivitetsdiagrammer er mer forretningsnært.

Aktivitetsdiagram “Registrer lånesøknad”



“Registrer lånesøknad”