UKE 11 - UML modellering del 2

IN1030 - Gruppe 2

Hva skal vi i dag?

- **Designpatterns**
- **Klassediagram**
- **Aktivitetsdiagram**

Menti

Brukerhistorier og krav - hvordan henger det sammen?

- Datainnsamling
  - Intervju
  - Observasjon
  - Spørreundersøkelse
- Analyse av innsamlet data
  - Hvem?
  - Hva?
  - Hvorfor?
- Brukerhistorier
  - Som....ønsker jeg....slik at...
- Krav
  - Hvordan skal vi implementere brukerhistoriene?
    - Funksjonelle
    - Ikke-funksjonelle
- Kravspesifikasjon

Eksempel - IN2000

Brukerhistorier:

1. Som bruker ønsker jeg å kunne se en oversikt over badeplasser i Oslo i form av liste og kart, slik at jeg får vite mine valgmuligheter til badeplass
2. Som bruker ønsker jeg å kunne trykke på en badeplass, og få en enkel oversikt over værforhold og badetemperatur, slik at jeg kan vurdere passende badeplass.

Funksjonelle krav:

1. Systemet skal kunne gi en oversikt over badeplasser i Oslo — Brukerkrav
   a. Oversikt over badeplasser i form av liste.
   b. Oversikt over badeplasser i form av kart. } Systemkrav
2. Systemet skal kunne gi bruker mutlighet til å trykke på en badeplass, og få opp basisinfo
   a. Basisinfo som: Navn på badestedet, solstatus, temperatur i vannet, sannsynlighet for regn, temperatur i luften.

Ikke-funksjonelt krav:

1. Systemet skal kunne gi bruker flere alternativer til navigering. — Brukerkrav
   1.1. Gjennom en hovedmeny
   1.2. Gjennom fragment i fragment } Systemkrav
2. Systemet skal benytte seg av API av Meteorologisk institutt

Ukesoppgave forrige uke: Spørsmål 2

Spørsmål: Hva skiller aktører fra interessenter?

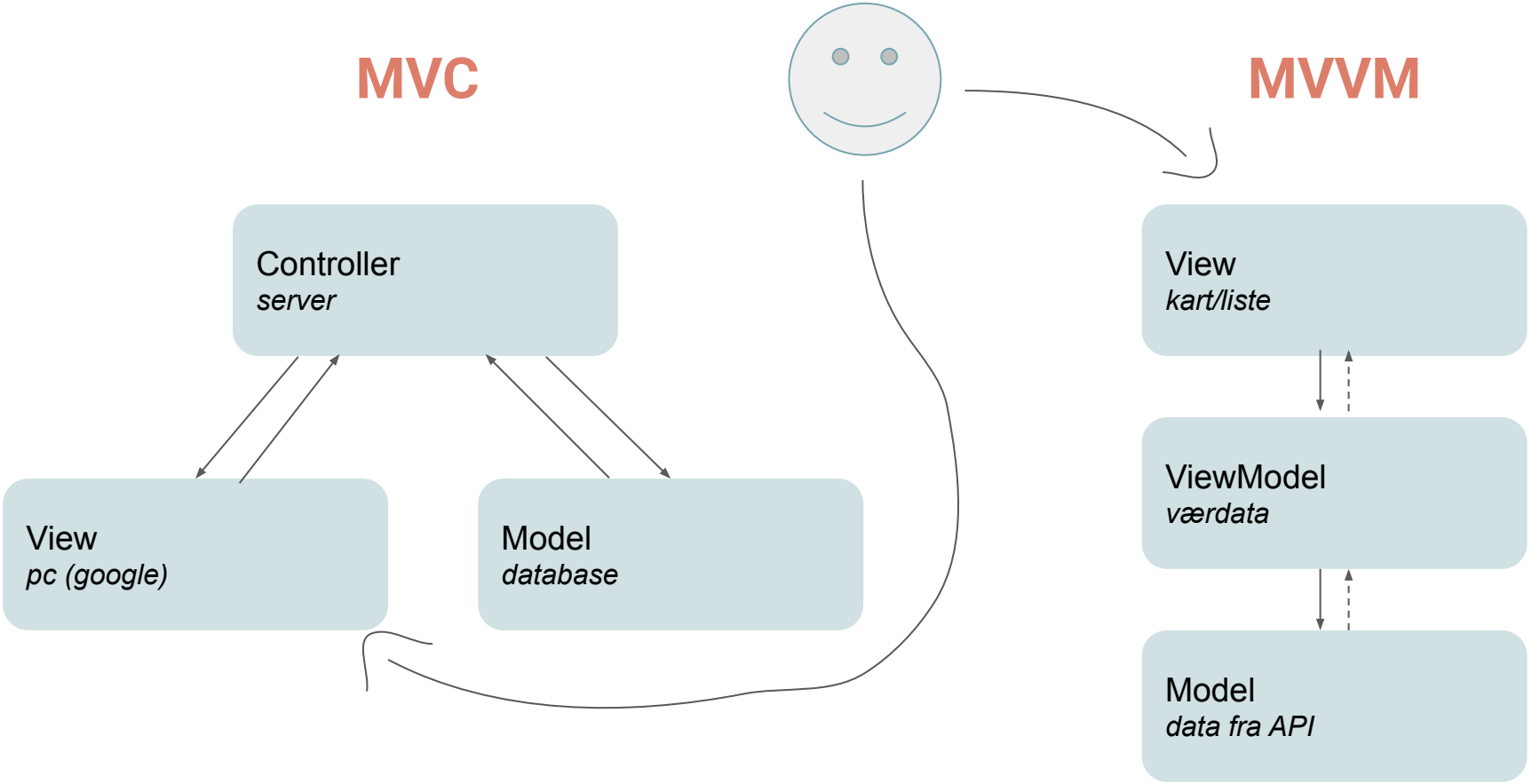
Svar: Skillet mellom aktør og interessent omhandler **rolle/tilknytning** til bruken og utviklingen av systemet.

- En aktør er en samlebetegnelse for å angi alle grupper av personer som **anvender** systemet, eller **øvrige systemer** som blir **anvendt** av systemet
  - Aktører må være enten:
    - **brukere** av systemet
    - **andre systemer** som brukes av/bruker systemet
- En interessent er en betegnelse for alle personer, grupper eller organer, som **påvirkes** av eller **påvirker** systemets utvikling/bruk. Både direkte og indirekte.
  - Interessenter kan være:
    - **brukere** av systemet
    - inkluderer **alle** som påvirker/påvirkes av systemets kravspesifikasjon og utvikling

Designpattern?

Les kapittel 6 i boken og på forelesningsfoilene.

Modeller brukes (blant annet) for å skape felles forståelse for arkitekturen, slik at utviklere kan jobbe på hver sin enhet etter samme mønster - for så å "merge" enhetene sammen til et stort system


MVC
MVVM
Controller
server
View
pc (google)
Model
database
View
kart/liste
ViewModel
værdata
Model
data fra API

Modularisering?

Arkitektur-prinsipp !

Høy kohesjon:
Et objekt skal kun ha ansvar for relaterte ting til det objektet
- **Lokalisere feil**
- **Holde orden**

Lav kobling:
Et objekt skal samarbeide/være avhengig med et begrenset antall andre objekter
- **videreutvikling**

Spørsmål?

Hvorfor lager vi så mange diagrammer?

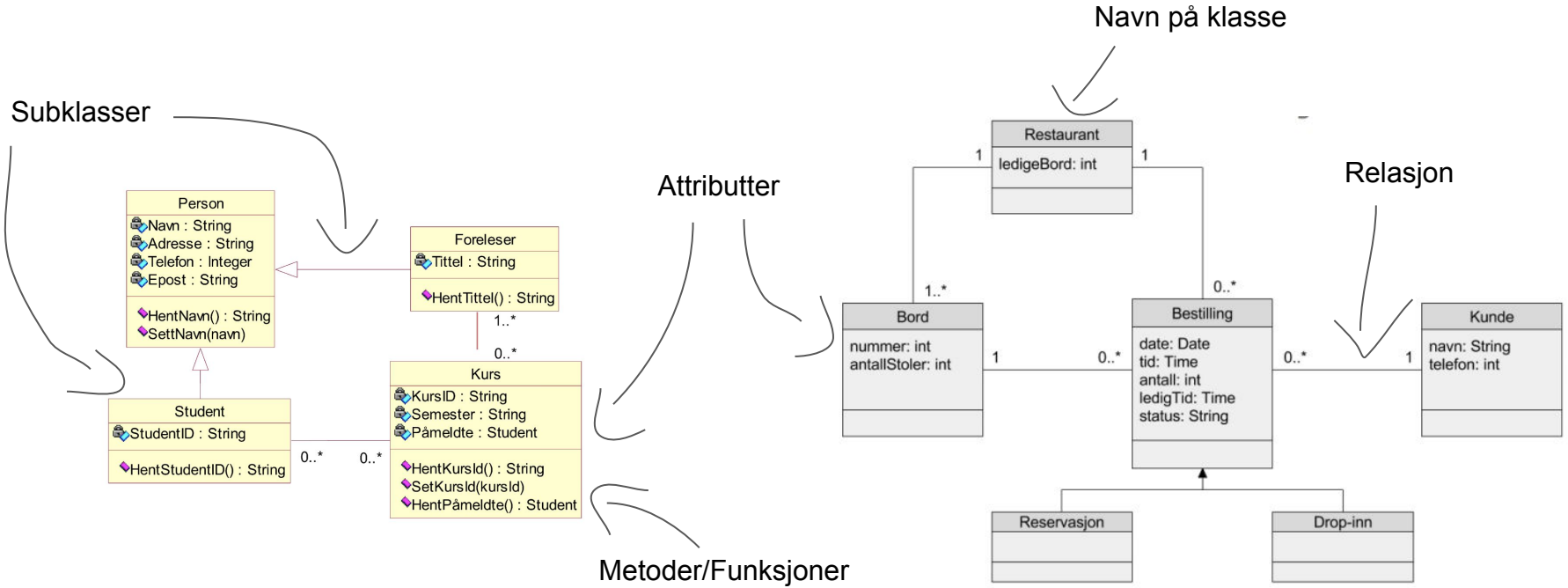
Vi ønsker å vite hva skal lage før vi lager det.
Vi må kommunisere med kunden hva vi skal lage.
Vi ønsker å lage en felles forståelse.
Vi ønsker å finne et designpattern som alle utviklerne forstår, fordi vi samarbeider jo om koden.
Vi ønsker at andre skal forstå systemet vi har laget, blant annet så de kan opprettholde det.

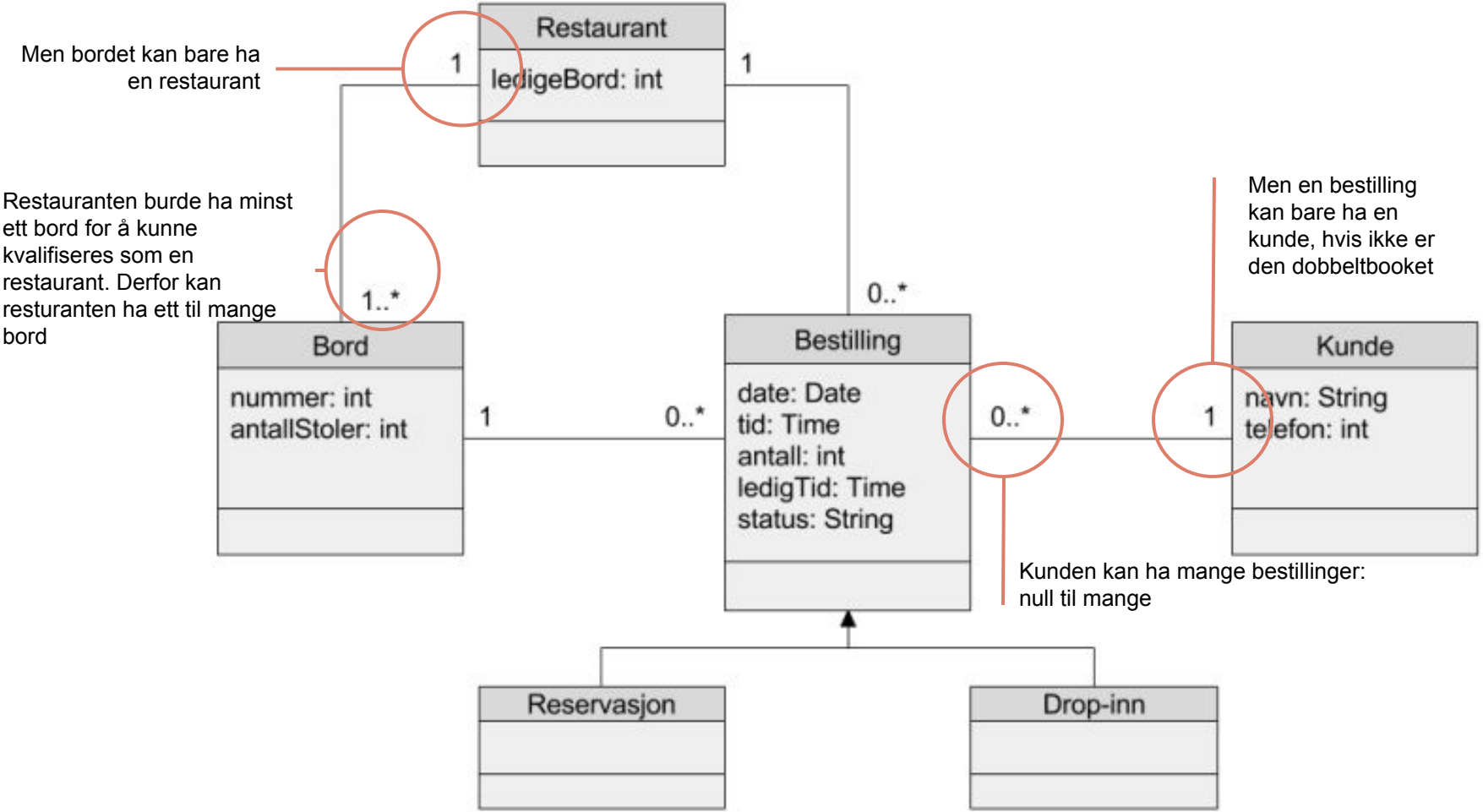
Strukturelle modeller

Klassediagram

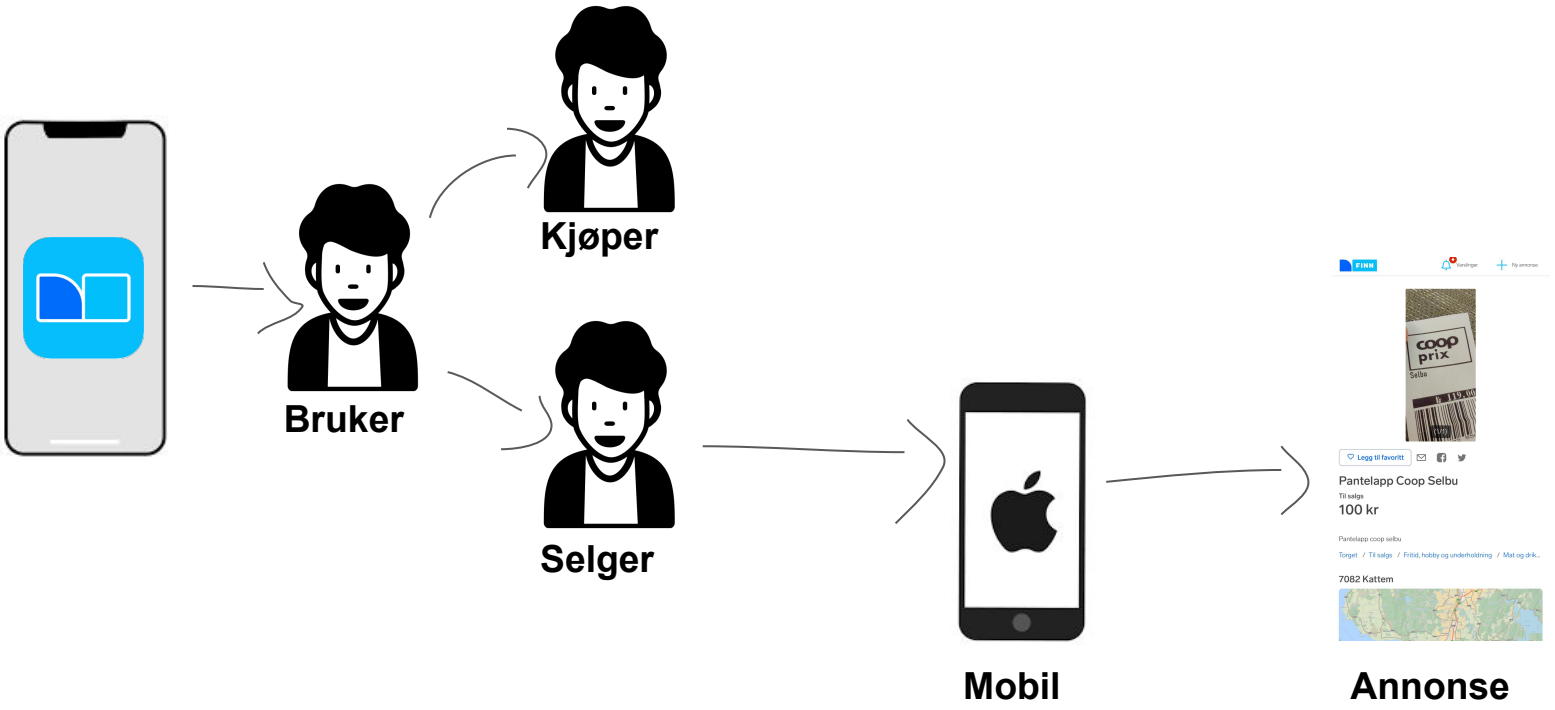
- Strukturmodell som viser strukturen i et system
- Klassediagram → viser objektklassene i systemet og assosiasjonene mellom disse klassene
- Assosiasjon: en link mellom klasser som indikerer at det er en relasjon mellom dem
  - 1 nøyaktig én
  - 0 .. 1 null eller én
  - * eller 0 .. * null eller mer
  - 1 ..* én eller mer

Notasjon


Navn på klasse
Subklasser
Attributter
Relasjon
Metoder/Funksjoner
Person
Navn : String
Adresse : String
Telefon : Integer
Epost : String
HentNavn() : String
SettNavn(navn)
Foreleser
Tittel : String
HentTittel() : String
Student
StudentID : String
HentStudentID() : String
Kurs
KursID : String
Semester : String
Påmeldte : Student
HentKursId() : String
SetKursId(kursId)
HentPåmeldte() : Student
1..*
0..*
0..*
0..*
Restaurant
ledigeBord: int
1
1
Bord
nummer: int
antallStoler: int
Bestilling
date: Date
tid: Time
antall: int
ledigTid: Time
status: String
Kunde
navn: String
telefon: int
Reservasjon
Drop-inn
1..*
0..*
1
0..*
0..*
1


Men bordet kan bare ha en restaurant
Restaurant
ledigeBord: int
1
1
Restauranten burde ha minst ett bord for å kunne kvalifiseres som en restaurant. Derfor kan resturanten ha ett til mange bord
1..*
0..*
Men en bestilling kan bare ha en kunde, hvis ikke er den dobbeltbooket
Bord
nummer: int
antallStoler: int
1
0..*
Bestilling
date: Date
tid: Time
antall: int
ledigTid: Time
status: String
0..*
1
Kunde
navn: String
telefon: int
Kunden kan ha mange bestillinger: null til mange
Reservasjon
Drop-inn

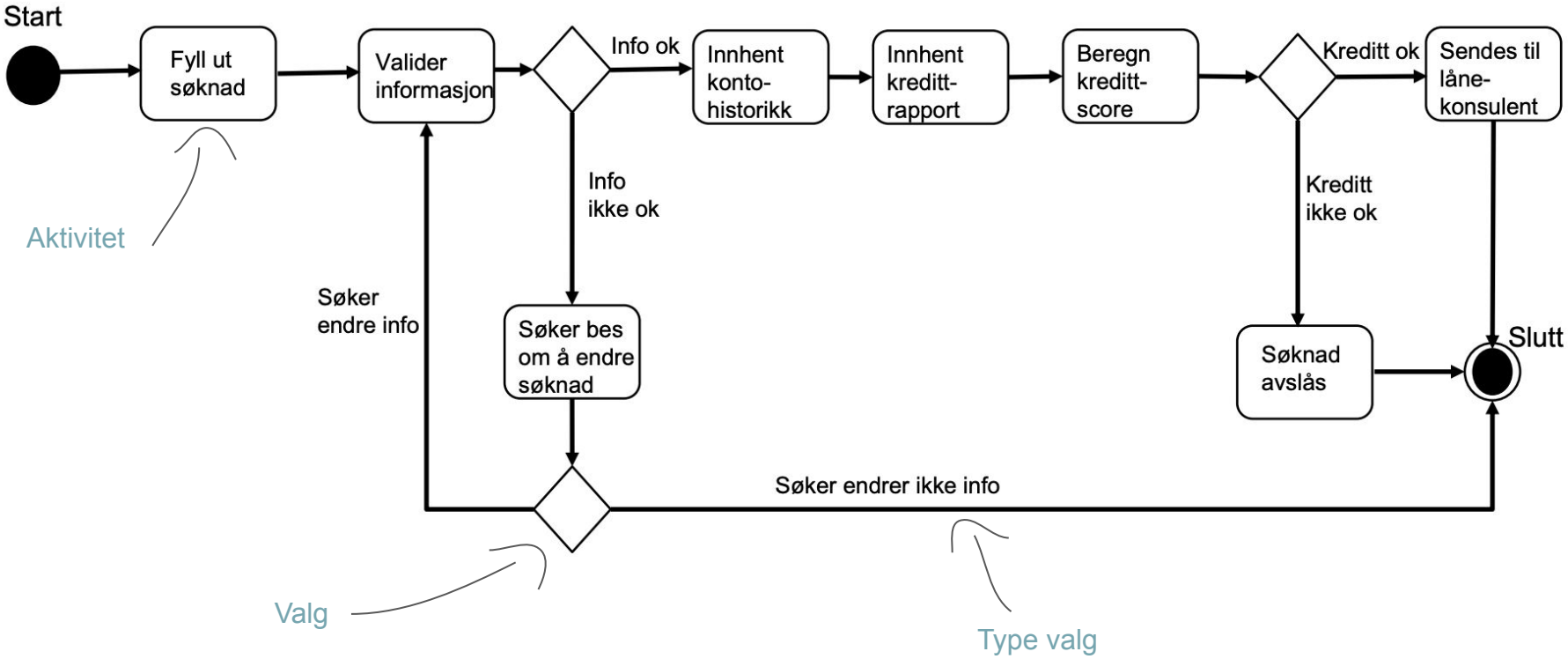
Liten historiefortelling



Spørsmål?

Atferdsmodeller

Aktivitetsdiagram

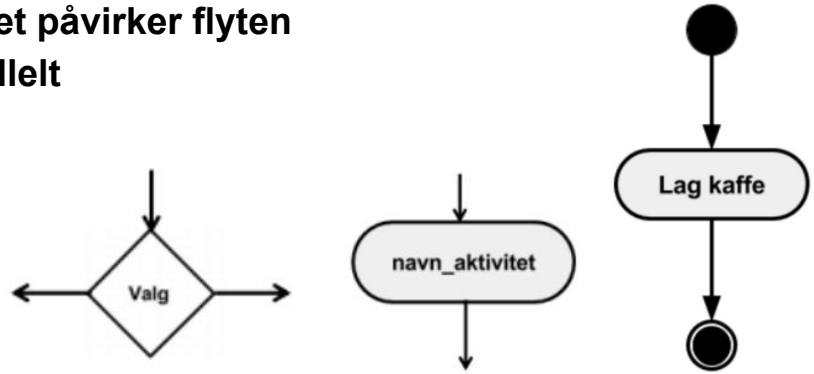


Oppgave 2A

Oppgave 2.a:
Hva er et aktivitetsdiagram?

Svar del 1:
Aktivitetsdiagram (= **flytskjema** (flowchart)) som viser:

- Grafisk representasjon av **arbeidsflyt**
- **Aktiviteter** og tilhørende **handlinger** (actions)
- Viser overordnet **kontrollflyt**
  - Beskriver hvordan mulige **utfall** av en **aktivitet påvirker flyten**
  - Viser hvilke **aktiviteter** som kan utføres **parallelt**



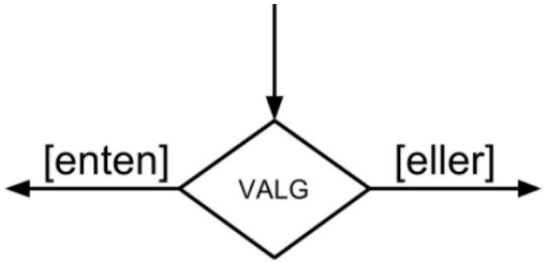
Oppgave 2A

Svar del 2: Grunnkomponenter i et aktivitetsdiagram:

- Start: angir hvor flyten starter
- Slutt: angir hvor flyten ender
- Aktiviteter: angir ulike aktiviteter som inngår i arbeidsflyten
    - Representeres med navngitte, avrundede rektangler
    - Kan være fysisk (“godkjenn søknad”) eller elektronisk (“vis kjøpshistorikk”)
- Valg: angir at man står ovenfor et valg (decision)
    - Eksempel: IF, IF-ELSE, CASE, osv.

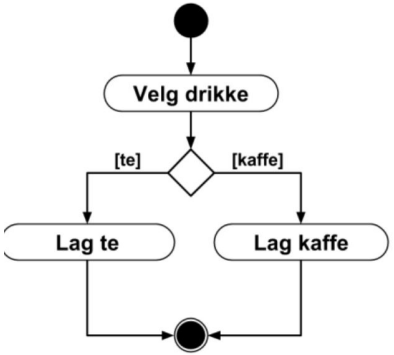
<u>Alternative løp:</u> valgdiamanter

- Beskriver hvordan handlinger (valg) påvirker flyten i systemet
- Viser de mulige utfallene av en aktivitet
- Kan ha mange utfall, men kun ett av utfallene velges



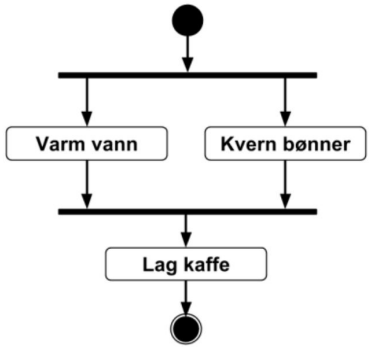
Oppgave 2A

Svar del 3: Eksempel1: Bruk av valgdiamanter:
- Hva forteller dette diagrammet oss? → Valg av drikke
- Hvilke aktiviteter inngår i arbeidsflyten? → Velg drikke, Lag té eller Lag kaffe
- Hva er de ulike valgene? → Té eller kaffe



Blokkeringer (bar)
- Representerer start (split) og slutt (join) for parallelle prosesser
- Viser hvilke prosesser man må vente på, før man kan gå videre

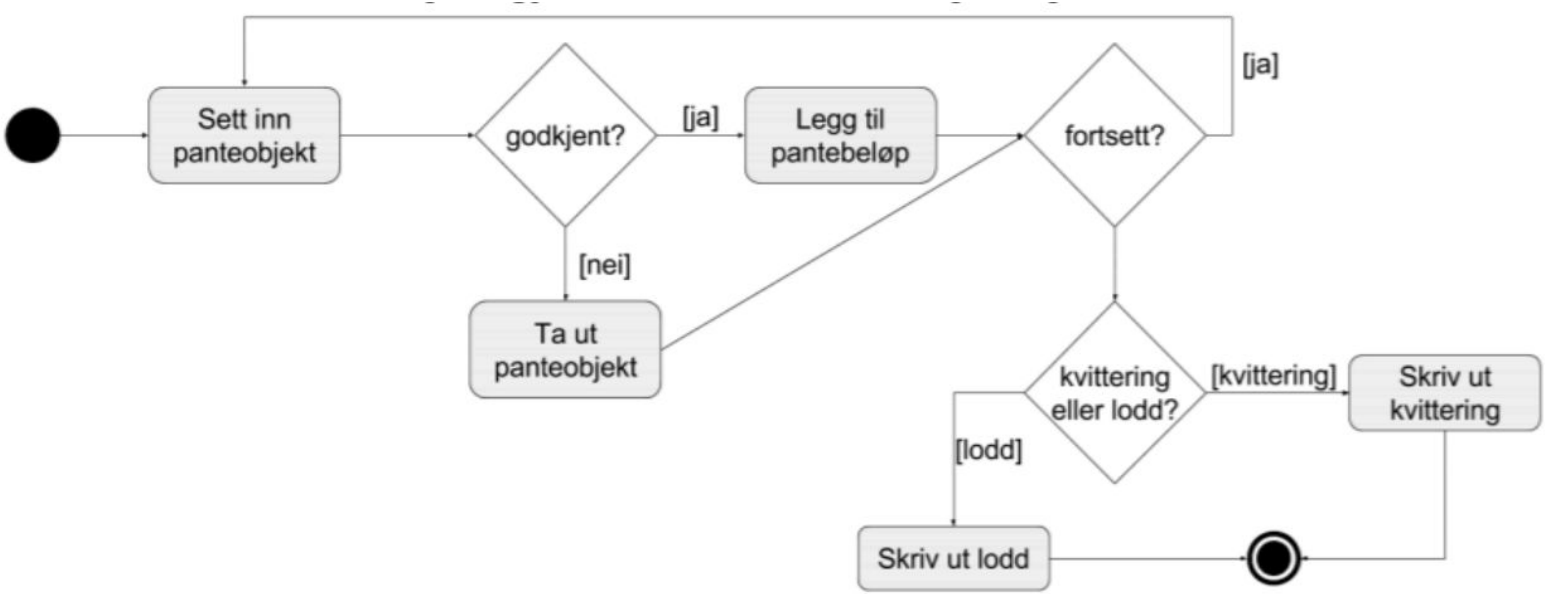
Eksempel 2: Bruk av blokkeringer:
- Hva forteller dette diagrammet oss? → Viser hvordan kaffe lages
- Hvilke aktiviteter inngår i arbeidsflyten? → Lag kaffe, Varm vann/kvern kaffebønner
- Hvilke aktiviteter kan utføres parallelt? → Varm vann og Kvern bønner



Oppgave 4C

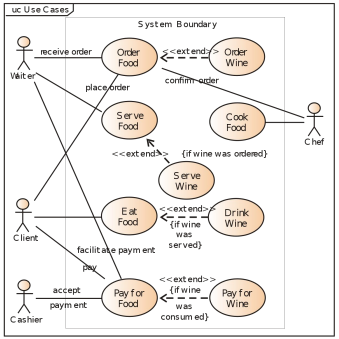
Svar del 2:

Sett sammen aktiviteter og avgjørelser til et helhetlig diagram

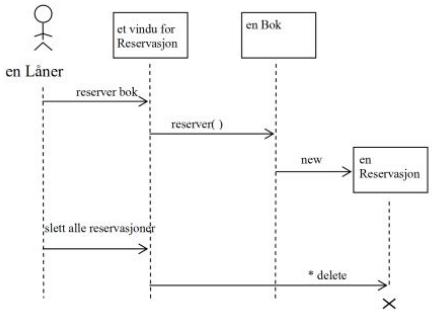


Spørsmål?

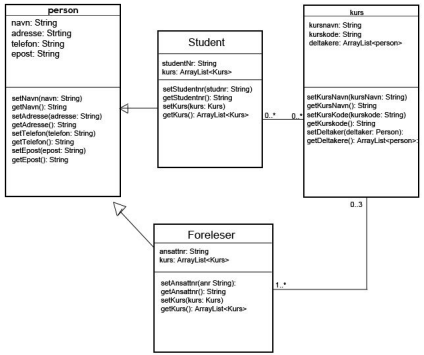
Use Case-diagram:



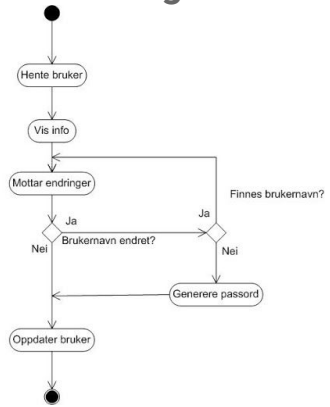
Sekvensdiagram:



Klassediagram:



Aktivitetsdiagram:



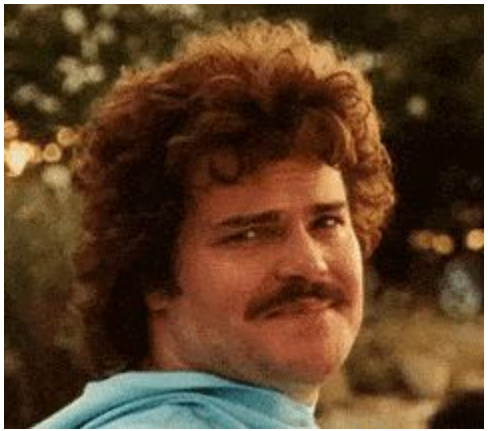
Interaksjonsmodeller

Strukturelle modeller

Atferdsmodeller

Ukesoppgaver

Talk but write



UKESOPPGAVER

Fasit

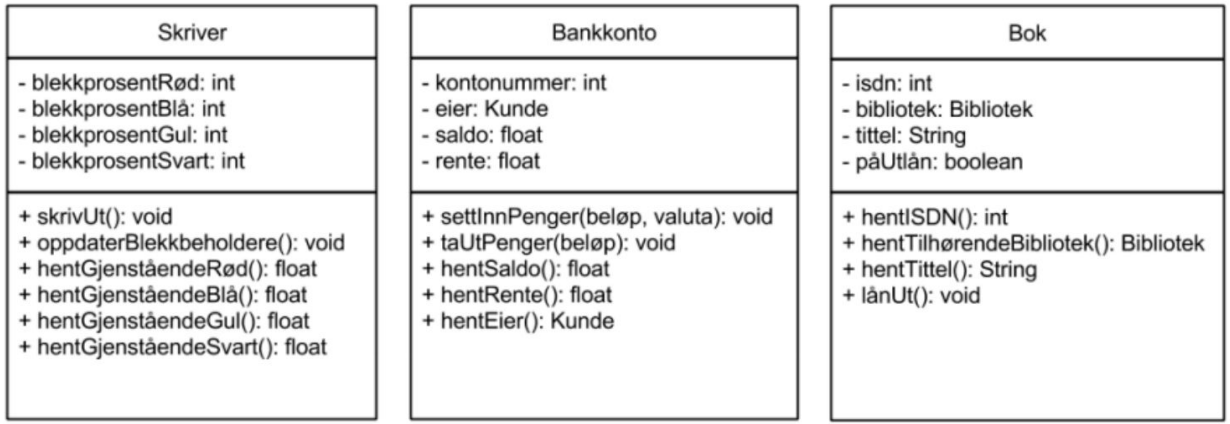
Oppgave 1A

Oppgave 1.a:
Ta utgangspunkt i følgende klasser:

- En skriver til en PC. Du kan anta at det er en blekkskriver, og at den ikke står i nettverk
- En bankkonto
- En bok som tilhører et bibliotek

Definer attributter og operasjoner (metoder) for disse klassene.

Svar:

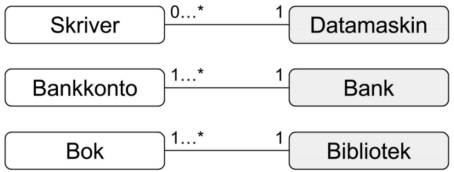


Oppgave 1B

Oppgave 1.b:
Finn andre klasser som disse klassene kan assosieres med.

Svar:
Hvilke andre klasser er naturlige å modellere?
- Bruk domenekunnskap for å identifisere øvrige klasser
- Hva er informasjonsbehovet? Hva skal de ulike klassene gjøre?
- Hvilke assosiasjoner ønsker vi mellom klassene?
- Hva forteller forholdene oss?

Skriver	Datamaskin
Bankkonto	Bank
Bok	Bibliotek

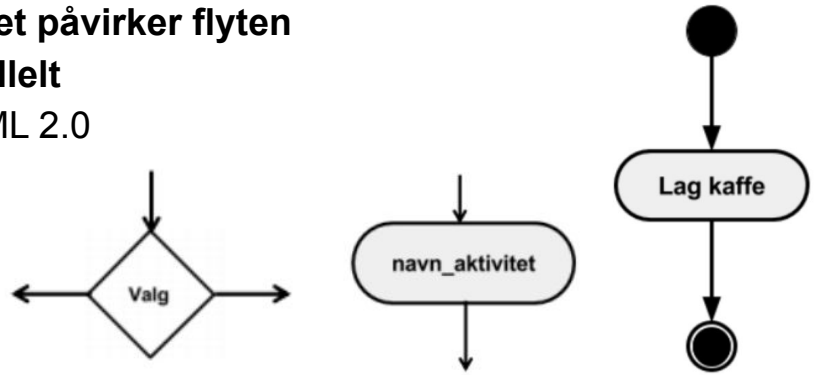


Oppgave 2A

Oppgave 2.a:
Hva er et aktivitetsdiagram?

Svar del 1:
Aktivitetsdiagram (= **flytskjema** (flowchart)) som viser:

- Grafisk representasjon av **arbeidsflyt**
- **Aktiviteter** og tilhørende **handlinger** (actions)
- Viser overordnet **kontrollflyt**
    - Beskriver hvordan mulige **utfall** av en **aktivitet påvirker flyten**
    - Viser hvilke **aktiviteter** som kan utføres **parallelt**
- Aktivitetsdiagram utgjør en del av standarden til UML 2.0



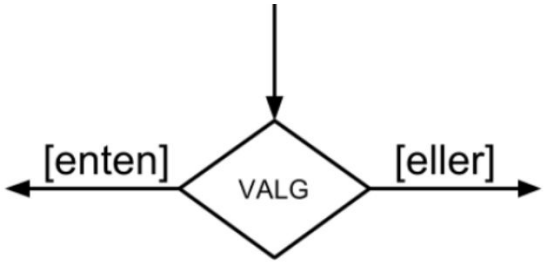
Oppgave 2A

Svar del 2: Grunnkomponenter i et aktivitetsdiagram:

- Start: angir hvor flyten starter
- Slutt: angir hvor flyten ender
- Aktiviteter: angir ulike aktiviteter som inngår i arbeidsflyten
    - Representeres med navngitte, avrundede rektangler
    - Kan være fysisk (“godkjenn søknad”) eller elektronisk (“vis kjøpshistorikk”)
- Valg: angir at man står ovenfor et valg (decision)
    - Eksempel: IF, IF-ELSE, CASE, osv.

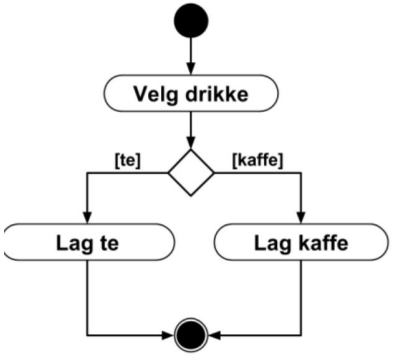
<u>Alternative løp:</u> valgdiamanter

- Beskriver hvordan handlinger (valg) påvirker flyten i systemet
- Viser de mulige utfallene av en aktivitet
- Kan ha mange utfall, men kun ett av utfallene velges



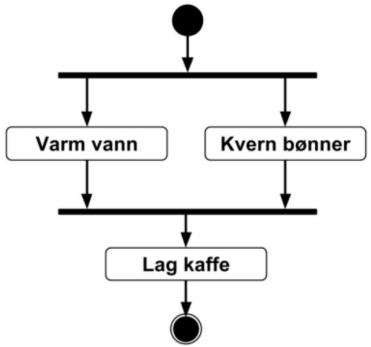
Oppgave 2A

Svar del 3: Eksempel1: Bruk av valgdiamanter:
- Hva forteller dette diagrammet oss? → Valg av drikke
- Hvilke aktiviteter inngår i arbeidsflyten? → Velg drikke, Lag té eller Lag kaffe
- Hva er de ulike valgene? → Té eller kaffe



Blokkeringer (bar)
- Representerer start (split) og slutt (join) for parallelle prosesser
- Viser hvilke prosesser man må vente på, før man kan gå videre

Eksempel 2: Bruk av blokkeringer:
- Hva forteller dette diagrammet oss? → Viser hvordan kaffe lages
- Hvilke aktiviteter inngår i arbeidsflyten? → Lag kaffe, Varm vann/kvern kaffebønner
- Hvilke aktiviteter kan utføres parallelt? → Varm vann og Kvern bønner



Oppgave 2B

Oppgave 2.b:
Hvorfor er det nyttig å lage aktivitetsdiagram?

Svar:
Aktivitetsdiagrams nytteverdi:

- Definerer **flyten** for en gitt **aktivitet** som kan **gjennomføres** i systemet
- Finner **prosesser** som kan kjøres **parallelt** og er **uavhengige** av hverandre
- Finner **“deadlocks”** i systemet
- Økt forståelse av **arbeidsrutiner**
- Aktivitetsdiagram kan modellere arbeidsflyt og organisasjonsflyt

Oppgave 3A

Regjeringen har forpliktet seg til at landet skal redusere utslippet av klimagasser. For å få alle **innbyggerne** til å bidra, ønsker **myndighetene** å registrere alt personlig forbruk som vil bidra til økt utslipp (fra bilbensin, flybensin, fyringsolje etc.). Alle personer skal betale en klimaskatt årlig avhengig av forbruk av klimagasser.

Oppgave 3.a:
Oppgi interessentene til systemet og hvilken interesse de har i systemet.

Svar:

Rolle	Interesse
Innbyggerne	Enkelt å bruke, gi god oversikt over bruk, korrekte data
Myndigheter (skatt og avgift)	Korrekt data, gi god rapportering, integrere mot relevante systemer
Datatilsynet	Overvåke, passe på at data ikke brukes til noe annet enn det er tenkt

Oppgave 3B

Oppgave 3.b:
Gi en tekstlig beskrivelse av bruksmønsteret ”Beregn mengde” som inkluderer navn, aktør, eventuelle pre-betingelser, post-betingelser, hovedflyt og minst én alternativ flyt.

“I denne oppgaven skal du modellere et bruksmønster som heter ”Beregn Mengde”. Bruksmønsteret starter ved at personen oppgir sitt fødselsnummer. Deretter viser systemet den totale klimagassmengden som personen gjennom sine kjøp har bidratt til, det vil si en totalsum summert over alle klimagasser fra årsskiftet til dags dato for den gitte personen.”

Svar del 1:
Navn - Aktør - Hovedflyt
“I denne oppgaven skal du modellere et bruksmønster som heter ”Beregn Mengde”. Bruksmønsteret starter ved at personen oppgir sitt fødselsnummer. Deretter viser systemet den totale klimagassmengden som personen gjennom sine kjøp har bidratt til, det vil si en totalsum summert over alle klimagasser fra årsskiftet til dags dato for den gitte personen.”

Oppgave 3B

Svar del 2:
Navn: Beregn mengde
Aktør: Person (eller forbruker, innbygger eller lignende)
Pre-betingelse: Ingen
Post-betingelse: Personens klimagassmengde har blitt oppdatert i databasen

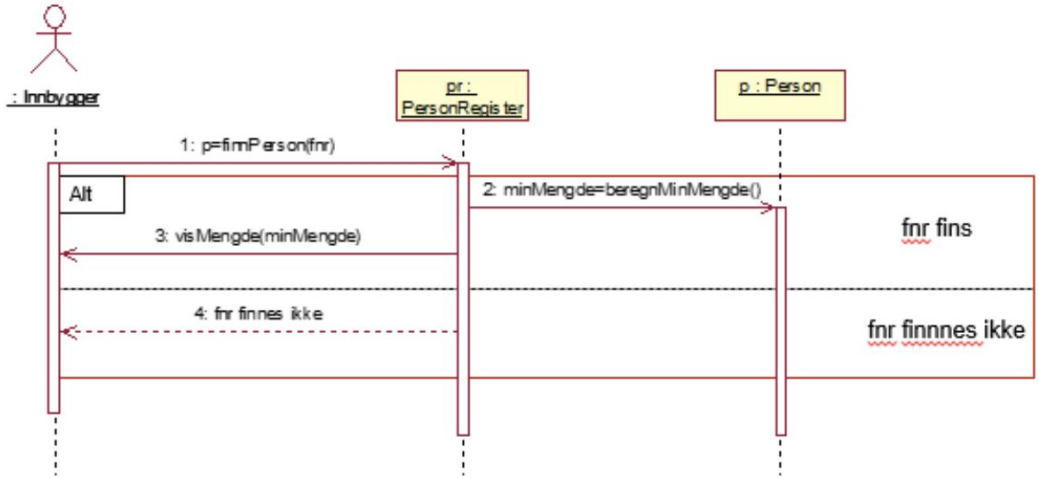
Hovedflyt:
1. Personen oppgir sitt fødselsnummer
2. Systemet finner personen i systemet
3. Systemet beregner den lokale klimagassmengden som personen gjennom sine kjøp har bidratt til
4. Systemet viser klimagassmengden til personen

Alternativ flyt:
Alternativ flyt I, steg 2.1: Ugyldig fødselsnummer
A2.1: Systemet gir feilmelding og avslutter
Alternativ flyt II, steg 2.2: Personen finnes ikke i systemet
A2.2: Systemet gir feilmelding og avslutter

Oppgave 3C

Oppgave 3.c:
Lag et sekvensdiagram for bruksmønsteret ”Beregn Mengde”.

Svar:
Husk å ta utgangspunkt i rekkefølgen i hovedflyt og alternativ flyt i bruksmønsteret (det blir trekk på eksamen hvis bruksmønster og sekvensdiagram ikke samsvarer). Identifiser øvrige objekter og eventuelle undersystem.



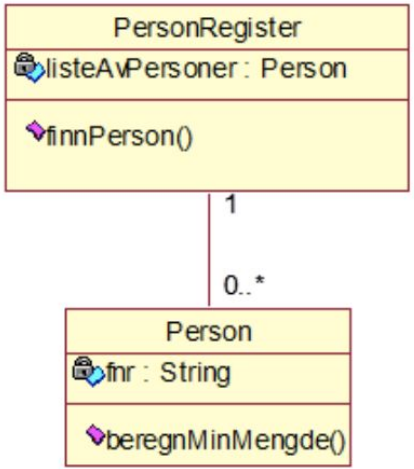
Oppgave 3D

Oppgave 3.d:
Tegn et klassediagram som tilsvarer sekvensdiagrammet fra oppgave 3c. Ta med attributter og metoder i hver klasse, samt assosiasjoner med multiplisitet.

Svar:
NOTIS: Det er svært viktig at alle UML-diagrammene samsvarer med hverandre. I et virkelig system er det mange flere attributter og metoder, men dette er nok for å løse oppgaven.

- beregnMinMengde skal ha en utparameter:
  - beregnMinMengde():Int.
- finnPerson() kan ha en parameter;
  - finnPerson (fnr: int).

(Kan i tillegg ha med Aktøren Innbygger som en klasse, med én til én assosiasjon med PersonRegister.)



Oppgave 4

Oppgave 4:
Ta utgangspunkt i eksempelet med å pante flasker fra forelesningen om objektorientert modellering.

Navn: Pante flasker
Aktør: Kunde
Pre-betingelse: Panteautomat er klar til å ta imot pant
Post-betingelse: Kunde får kvittering eller lodd (Røde Kors)

Hovedflyt
1. Kunde setter inn en flaske (panteobjekt)
2. Panteautomaten skanner koden på flasken
3. Panteautomaten godkjenner objekt og pantebeløpet legges til det totale
4. Kunde trykker på kvittering
5. Panteautomat skriver ut kvittering

Oppgave 4A

Oppgave 4.a:
Utvid den tekstlige beskrivelsen med tilfellene at flere flasker pantes og at det er fullt i mottaket av flasker (alternative flyt).

Svar:

Alternativ flyt I, steg 1: Mottak fullt

- A1.1: Systemet gir beskjed om at det er fullt i mottaket av flasker
- A1.2: Systemet returnerer flaske
- A1.3: Systemet går til steg 5 i hovedflyten

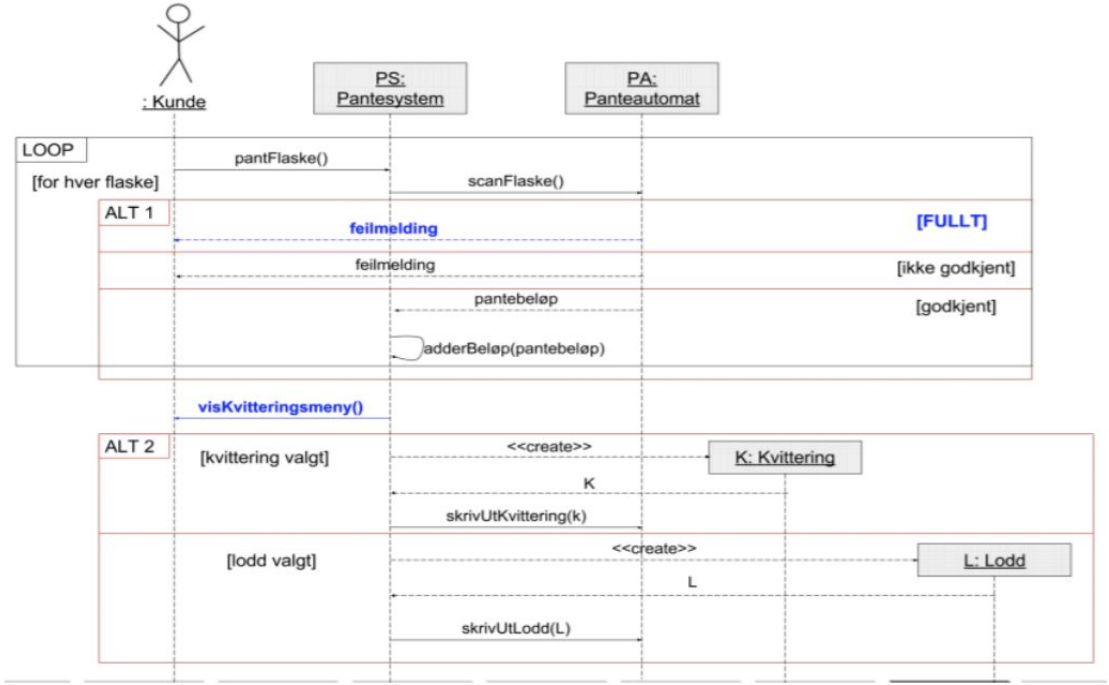
Alternativ flyt II, steg 4: Flere flasker

- A2.1: Kunde velger å pante en ny flaske
- A.2.2: Systemet returnerer til steg 1 i hovedflyten

Oppgave 4B

Oppgave 4.b:
Utvid sekvensdiagrammet med tilfellet at det er fullt i mottaket av flasker.

Svar:



Oppgave 4C

Oppgave 4.c: Utvid aktivitetsdiagrammet med funksjonalitet som legger til pantebeløp på flasker som gir pant.

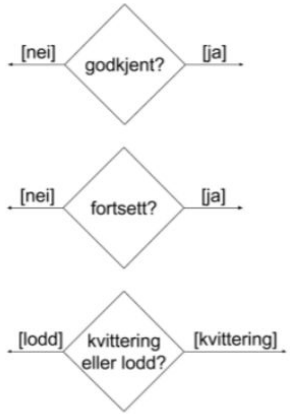
Sett inn panteobjekt
Ta ut panteobjekt
Legg til pantebeløp
Skriv ut lodd
Skriv ut kvittering

Svar del 1:
1. Identifiser aktivitetene som inngår i “Pant av flasker”

Notasjon - aktivitet angis med avrundet rektangel (med verbfrase).

2. Identifiser de ulike **avgjørelsene** og mulige **utfall**
Godkjent? → Er panteobjektet gyldig? (Med andre ord: Gir det pant?)
Fortsett? → Ønsker brukeren å pante flere flasker?
Kvittering eller lodd? → Ønsker brukeren kvittering eller lodd?

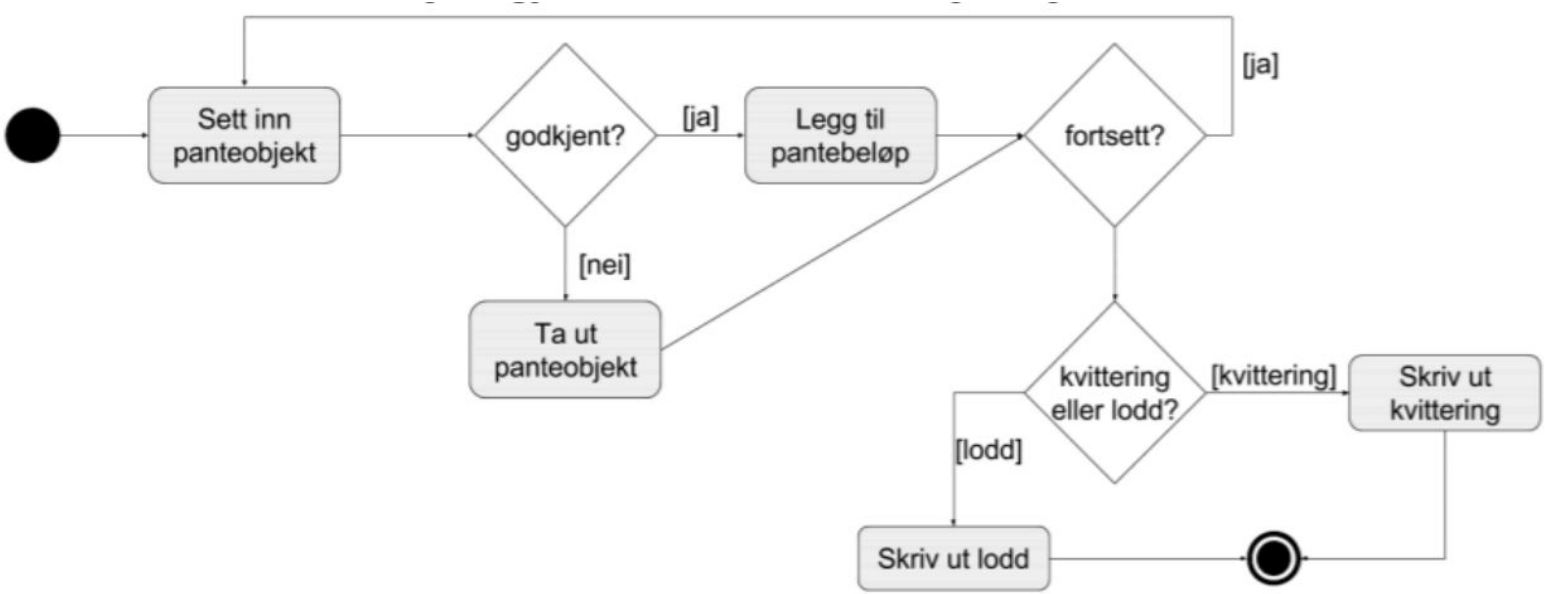
Notasjon - avgjørelser representeres med valgdiamanter
Notasjon - utfallene angis ved piler til neste aktivitet



Oppgave 4C

Svar del 2:

Sett sammen aktiviteter og avgjørelser til et helhetlig diagram



Oppgave 4D

Oppgave 4.d:
Hvordan kunne du modellert use caset uten å bruke klassen Pantesystem i klasse- og sekvensdiagrammet?

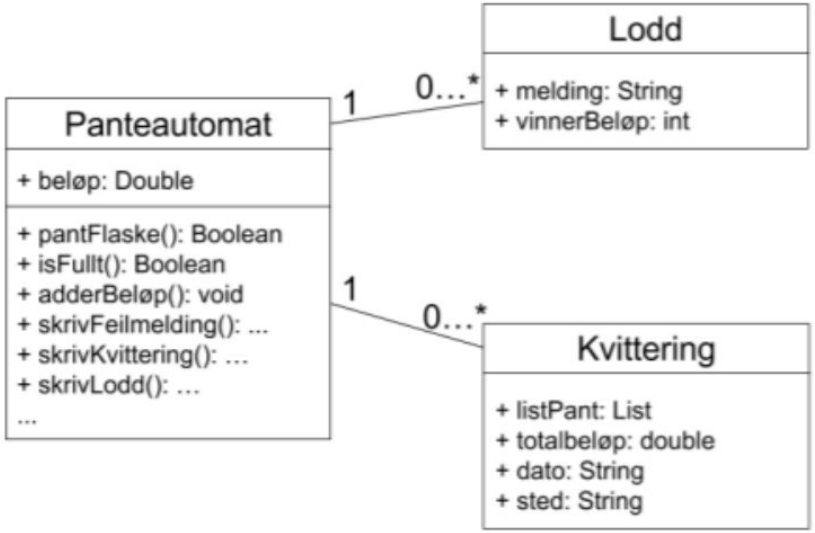
Svar del 1:

- Vi ønsker ikke å **miste funksjonalitet**
- Vi må derfor legge **funksjonaliteten** i Pantesystem i **andre klasser.** Denne eller disse klassene tar over ansvaret for denne funksjonaliteten.
- Den opprinnelige løsningen hadde følgende klasser:

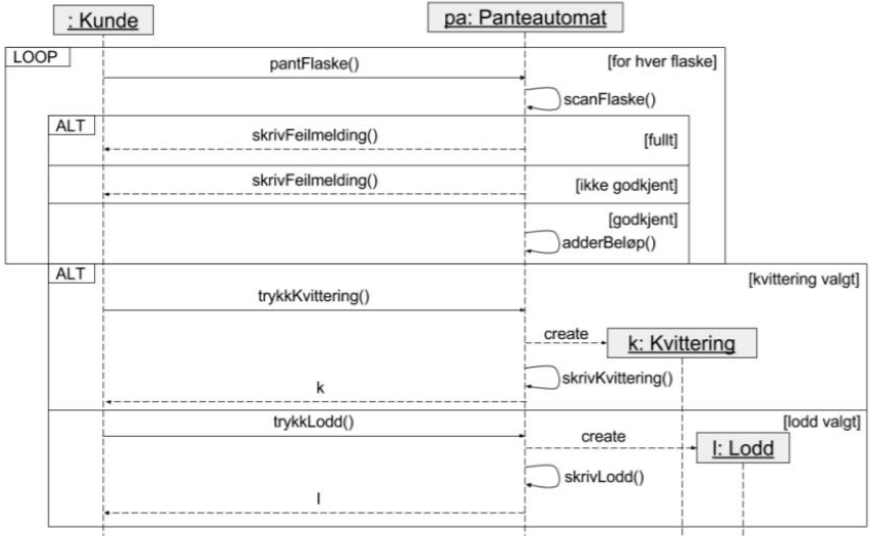
1. Pantesystem
2. Panteautomat → *Vi flytter funksjonaliteten fra Pantesystem til denne klassen*
3. Kvittering
4. Lodd

Oppgave 4D

Svar del 2:
Nytt **klassediagram** hvor innholdet i Pantesystem legges hos Panteautomat:



Nytt **sekvensdiagram**:



Andre fine videoer:

Aktivitetsdiagram

Sekvensdiagram

Use Case-diagram

emmatv@uio.no