Kravspesifikasjon

Erik Arisholm

Simula Research Laboratory

&

Institutt for Informatikk

Kravspesifikasjon

- **Hva skal systemet gjøre?**
- **Hvem og hva påvirker krav?**
- **Motivasjon: Hvorfor trenger vi UML?**
  - **Noen resultater fra et UML-eksperiment**
- **Kravspesifikasjon med UML:**
  - **Uttrykke krav som bruksmønstre (use cases)**

Mal for kravspesifikasjon

- **Overordnet systembeskrivelse (visjon)**
  - **Hva er formålet med systemet?**
- **Funksjonelle krav (UML bruksmønstre)**
  - **Hvilke tjenester skal systemet tilby?**
- **Kvalitetsønsker (ikke-funksjonelle krav)**
  - **Eks: Krav til svartider, brukervennlighet, endringsevne, oppetid**
- **Rammer (påvirker funksjonelle og ikke-funksjonelle krav)**
  - **Lover og forskrifter**
  - **Teknologi og samvirke med andre systemer**
  - **Standarder**
  - **Økonomiske rammer og tidsfrister**

Hvordan finne fram til kravene?

- **Intervjuer med kunder og potensielle brukere**
- **Observasjon av eksisterende arbeidsprosesser**
- **Granskning av eksisterende IT-systemer**
- **Granskning av standarder, lover og forskrifter**
- **Inkrementell og evolusjonær (iterativ) systemutvikling, inkludert prototyping**

Unified Modeling Language - UML

Et sett på 8 diagramteknikker, utarbeidet av toneangivende grupperinger innen OO

	Use-Case view	Logi-cal view	Compo-nent view	Concur-rency view	Deploy-ment view
Use Case diagram	■				
Class/object diagram		■			
Sequence diagram		■		■	
Collaboration diagram		■		■	
State diagram		■		■	
Activity diagram		■		■	
Component diagram			■	■	
Deployment diagram				■	■

Hva brukes UML til?

- **Notasjon som støtter opp under modellbasert systemutvikling**
  - **objektorientert analyse ("hva systemet skal gjøre"), og**
  - **objektorientert design ("hvordan systemet skal gjøre det")**
  - **simulering, prototyping, kodegenerering og testing**
- **Dokumentasjon av systemets krav, design og implementasjon**
  - **for andre utviklere (og for deg selv)**
  - **for kommunikasjon med kunde/sluttbruker under utviklingsprosjektet**
    - **Er ikke så sikker på denne påstanden: det er ikke sikkert at kunden forstår UML spesielt godt…**
      - **Enkle prototyper av skjermbilder fungerer kanskje bedre som kommunikasjonsmiddel med sluttbrukere!***

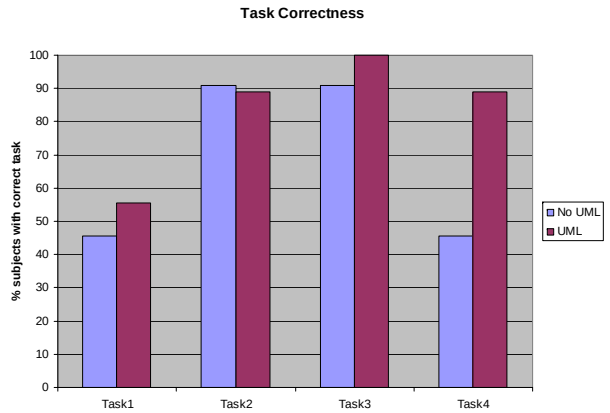
*** Anette C. Lien and Erik Arisholm, Evolutionary Development of Web-applications - Lessons Learned, European Software Process Improvement Conference (EuroSPI'2001), Limerick Institute of Technology, Ireland, November 2001**

Motivasjon: Vitenskapelig evaluering i vårt fag

- **Det er dessverre fortsatt svært vanlig i vårt fag å ta i bruk ny teknologi fordi noen påstår at den er bra, før den har vært gjennom en grundig evaluering av både potensielle kostnader og eventuelle gevinster**
  - **Prosesser, metoder og verktøy for objektorientert programvareutvikling er absolutt ikke noe unntak!**
  - **Det finnes SVÆRT lite vitenskapelig basert kunnskap om *når*, *for hvem*, *hvordan* og *hvorfor* en gitt prosess, metode eller verktøy er nyttig**

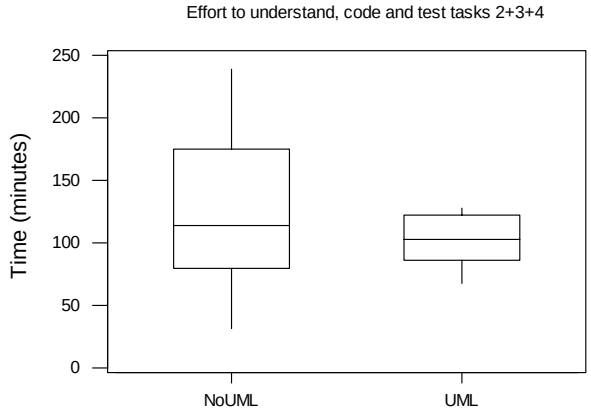
Nytteverdien av UML som dokumentasjon for andre utviklere – resultater fra et kontrollert eksperiment*

(m/Java endringsoppgaver og Tau UML)



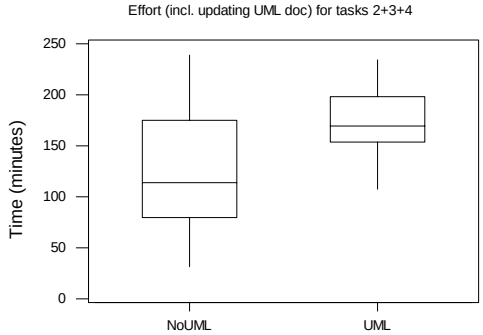
* Erik Arisholm, Samera Afsheen Ali & Siw Elisabeth Hove: "An Initial Controlled Experiment to Evaluate the Effect of UML Design Documentation on the Maintainability of Object-Oriented Software in a Realistic Programming Environment", Simula TR-2003-4

Nytteverdi av UML – tidsforbruk på koding og testing*



* Erik Arisholm, Samera Afsheen Ali & Siw Elisabeth Hove: "An Initial Controlled Experiment to Evaluate the Effect of UML Design Documentation on the Maintainability of Object-Oriented Software in a Realistic Programming Environment", Simula TR-2003-4

Nytteverdi av UML – ekstra tidsforbruk for å oppdatere UML-modellene med Tau UML*



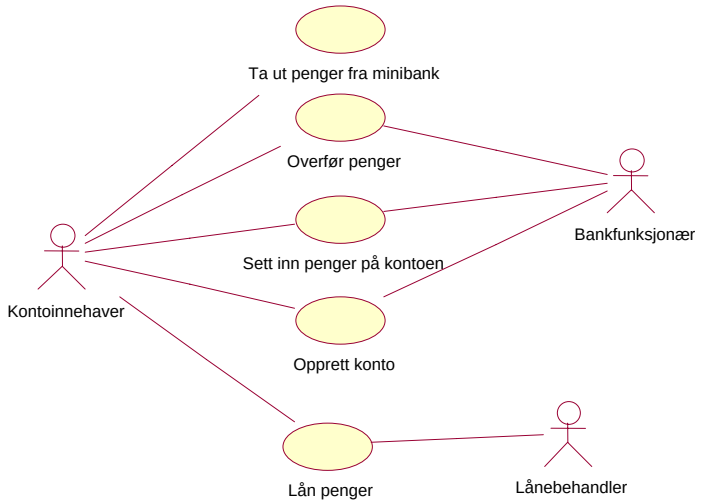
* Erik Arisholm, Samera Afsheen Ali & Siw Elisabeth Hove: "An Initial Controlled Experiment to Evaluate the Effect of UML Design Documentation on the Maintainability of Object-Oriented Software in a Realistic Programming Environment", Simula TR-2003-4

Hva er en bruksmønstermodell?

- **En komplett bruksmønstermodell består av to komponenter**
  - ***UML bruksmønsterdiagrammer* som gir en visuell representasjon av modellen (oversikt over aktører, bruksmønstre og sammenhengen mellom dem)**
  - ***tekstlige spesifikasjoner* av hvert enkelt bruksmønster**
- **Modellen beskriver hva systemet skal gjøre**
  - **Aktør: Beskriver en bestemt type bruker av systemet**
    - **Aktører kan være mennesker eller andre informasjonssystemer**
    - **UML symbol for en aktør:**

  - **Bruksmønster (use case)**
    - **beskriver en bestemt interaksjon mellom systemet og aktøren**
    - **UML symbol for et bruksmønster:**

Eksempel: Bank bruksmønsterdiagram



Aktører (1)

- En aktør beskriver en (og kun en) ROLLE som en bruker (en person eller et annet system) har i forhold til systemet som skal utvikles
  - En person (Ola) kan ha mange roller

    - Aktør 1: “Kontoinnehaver”
    - Aktør 2: “Bankfunksjonær”
    - Men “Ola” er ingen aktør i banksystemet

Aktører (2)

- Aktører avgrenser systemets bruksområder:
  - Hvilke typer brukere vil systemet ha?
  - Hvilke MÅL har disse brukerne?
    - Målene til aktørene dikterer hvilke tjenester systemet skal tilby
  - Primære aktører initierer bruksmønstre som oppfyller deres mål
  - Sekundære aktører muliggjør utførelsen av bruksmønstre
  - Hvilke aktører et system har, og hvorvidt en aktør er *primær* eller *sekundær* avhenger av hvor man setter grensene mellom system og dets omgivelser

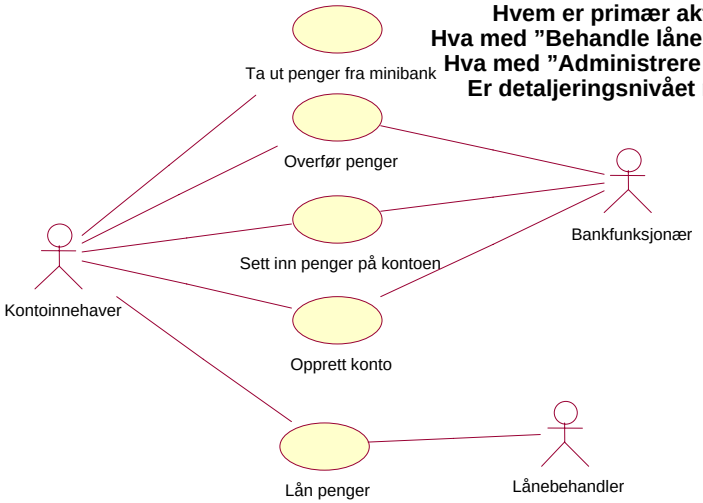
Aktører (3)

- Aktører har MÅL. Målene er gode utgangspunkt for å finne bruksmønstre:
  - “Kontoinnehaver”:
    - Mål1: Ta ut penger
    - Mål2: Overføre penger
    - Mål3: Sette inn penger
    - Mål4: Låne penger
  - “Bankfunksjonær”
    - Mål1:Behandle lånesøknad

Hva er et bruksmønster?

- Et bruksmønster beskriver en komplett sekvens av interaksjoner mellom aktør og system for å oppnå et bestemt MÅL
- Navnet på et bruksmønster representerer en brukssituasjon hvor alt går bra (”happy day scenario”). Tips! Bruk *aktive* setninger
- Eksempler ():
  - *Ta ut* penger fra kontoen (ikke ”uttak”)
  - *Overfør* penger (ikke ”pengeoverføring”)
  - *Sett inn* penger på kontoen (ikke ”innskudd”)
  - *Opprett* konto

Eksempel: Bank bruksmønsterdiagram



Hvem er primær aktør?
Hva med "Behandle lånesøknad"?
Hva med "Administrere konto"?
Er detaljeringsnivået riktig?

Hvordan lage bruksmønstermodeller?

- Det er ofte vanskelig å finne de "riktige" bruksmønstrene
  - Hvor STORT er et bruksmønster?
  - Hvor DETALJERT er et bruksmønster?
  - Hva MANGE bruksmønstre består modellen av?
- Det finnes ikke noe fasitsvar, men her er noen enkle retningslinjer:
  - 1: Identifiser grensen mellom systemet og omgivelser. Det bestemmer hva som er primære aktører
  - 2: Dersom du lar målene til det du anser som de primære aktørene styre valg av bruksmønstrene har du et godt utgangspunkt å jobbe med
  - 3: Dersom modellen blir svært kompleks kan man velge et høyere abstraksjonsnivå (med "større" bruksmønstre på øverste nivå)
    - I videregående kurs (bl.a. Inf3120 – Software Engineering) vil dere også lære mer avanserte UML-konstruksjoner som gjør det mulig å definere mer struktur på store bruksmønstermodeller (<<include>>, <<extend>>, m.m.)

Hvilke av disse setningene tror du representerer et bruksmønster?

- "Logg inn på systemet"
- "Lei en bil"
- "Behandle retur av leiebil"
- "Øk produksjonen med 20%"
- "Varesalg"

Hint:

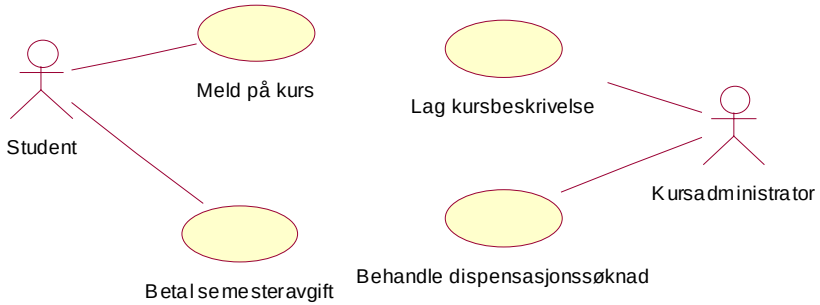
- Hvor går systemgrensen, hvem er primær aktør og hvilke(t) mål oppfyller bruksmønsteret?
- Kan bruksmønsteret beskrives som et komplett hendelsesforløp med en veldefinert start og slutt?

Mal for spesifikasjon av et bruksmønster

Et bruksmønster er ikke komplett uten en tekstlig beskrivelse som skal bestå av følgende informasjon

- Navn: *Hva heter bruksmønsteret*
- Aktør: *Hvem initierer bruksmønsteret*
- Pre-betingelse: *Beskriver hva som må være oppfylt for at bruksmønsteret skal kunne utføres (opsjon)*
- Post-betingelse: *Beskriver tilstand til system og aktør etter at bruktsmønsteret er utført, både ved normal hendelsesflyt og variasjoner. (opsjon)*
- Trigger: *Et situasjon eller hendelse som initierer bruksmønsteret*
- Normal hendelsesflyt: *Hva skjer når alt går bra, dvs når målet til aktøren oppnås på enklest mulig måte*
- Variasjoner: *Beskriv spesialtilfellene (unntak fra normal hendelsesflyt) separat. Format: <Betingelse>: <konsekvenser>*
- Relatert informasjon: *For eksempel ikke-funksjonelle krav ("svartid må være bedre enn 10 sekunder") (opsjon)*

Eksempel: Første utkast til bruksmønstermodell for kursregistrering



Spesifikasjon av ”Meld på kurs”

Aktør: Student

Trigger: Student ønsker å melde seg på et kurs

Pre-betingelse: Student har betalt semesteravgift og er logget inn på systemet

Post-betingelse: Student er meldt på kurset eller student har sendt søknad om dispensasjon

Normal Hendelsesflyt:

1. Studenten velger kurs
2. Systemet sjekker om det er ledig plass på kurset
3. Systemet sjekker at studenten kvalifiserer til å ta kurset
4. Systemet registrerer studenten på kurset og oppdaterer antall oppmeldte på kurset

Spesifikasjon av ”Meld på kurs” (forts.)

Variasjoner:

1a. Kurset holdes ikke dette semesteret: (Er dette en mulig variasjon?)

1. Studenten velger et annet kurs eller avslutter

2a. Kurset er fullt: (Er dette en mulig variasjon?)

1. Systemet gir melding og avslutter registreringen

3a. Kurset forutsetter andre kurs:

1. Systemet sjekker om studenten har tatt kursene

1a. Studenten har tatt kursene som forutsettes:

1. Systemet fortsetter registreringen

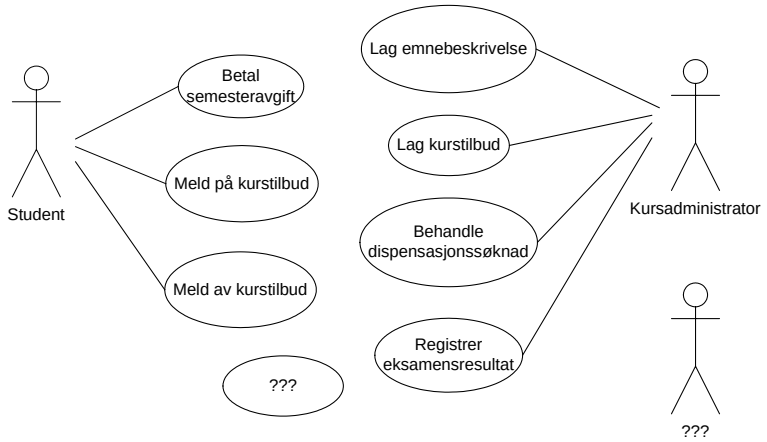
1b. Studenten har IKKE tatt kursene som forutsettes:

1. Systemet spør om studenten ønsker å søke om dispensasjon
2. ……

En vanlig problemstilling

- **”Systemadministrator” el.l. er ofte en aktør som utfører administrative oppgaver i systemet, for eksempel å opprette, slette eller endre brukere.**
- **Alternativ 1: ”Systemadministrator” har ett bruksmønsteret**
  - **”Administrer bruker”**
- **Alternativ 2: ”Systemadministrator” har tre bruksmønstre**
  - **”Opprett ny bruker”**
  - **”Slett bruker”**
  - **”Endre bruker”**
- **Begge alternativer er OK, og beskriver samme funksjonalitet**
- **Fordeler og ulemper ved begge alternativer:**
  - **Alt. 1:**
    - **+ Felles funksjonalitet mellom opprett, slett og endre kan beskrives på ett sted**
    - **- Ofte litt vanskelig å avgjøre hva som er ”normal hendelsesflyt” og hva som er variasjoner**
  - **Alt. 2:**
    - **+ Ofte blir ”normal hendelsesflyt” enklere å identifisere samt færre variasjoner**
    - **- Bruksmønsterdiagrammet kan bli stort og uoversiktlig**
    - **- Felles funksjonalitet beskrives flere steder (men NB! Det finnes løsninger på dette som forklares i videregående kurs)**

Revidert bruksmønstermodell, alt. 1



Bør "Lag x" inkludere både "ny x", "endre x" og "slette x", eller bør man dele opp?

Mulig høynivå spesifikasjon av "Lag emnebeskrivelse"

Aktør: Kursadministrator

Trigger: Kursadministrator ønsker å opprette, endre eller slette en emnebeskrivelse

Normal Hendelsesflyt:

1. Kursadministrator velger emnekode
2. Systemet sjekker om emnekoden eksisterer
3. Kursadministratoren oppdaterer beskrivelsen av kurset (her kan det være mange underpunkter, for eksempel registrering av hvilke andre kurs som forutsettes)
4. Systemet registrerer den nye informasjonen

…

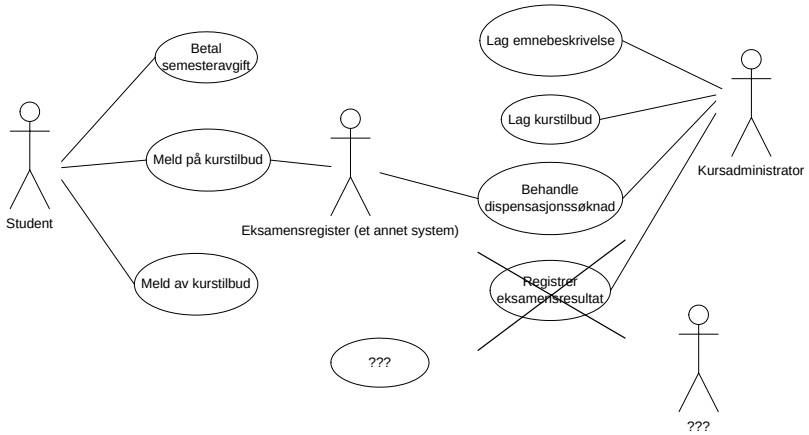
Variasjoner:

2a. Emnekoden eksisterer ikke: (dvs, "Opprett ny emnekode" er en variasjon)

1. Systemet spør om ny emnekode skal opprettes og oppretter i så fall ny emnekode

…

Revidert bruksmønstermodell, alt. 2



Valgt systemgrense bestemmer hvilke bruksmønstre, primære og sekundære aktører modellen inneholder.

I denne modellen er "Eksamensregister" en sekundær aktør i modellen til "Kursregistreringssystem".

Dessuten er "Kursregistreringssystem" er en primær aktør i den tilsvarende modellen for systemet "Eksamensregister"