

Systemudvikling, SY – Modul 2

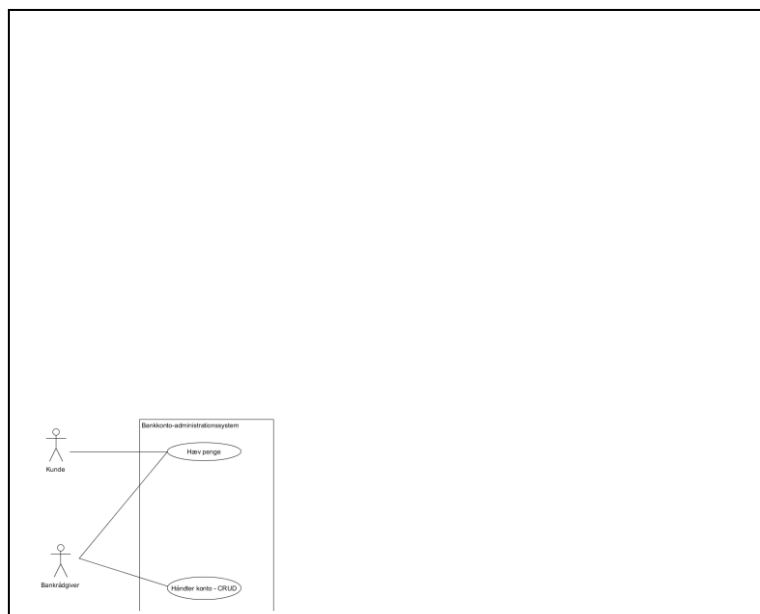
Opgaver:


Opgave 1: Tænke høj test (fra modul 1):

1. Gruppen udvælger en testperson, en testleder og resten er observatører.
2. Testpersonen skal flytte til en udvalgt gruppe.
3. Testen udføres, hvorefter testpersonen flytter tilbage til sin egen gruppe.
4. Resultaterne samles og gruppen laver en konklusion på testen.

Opgave 2: Use cases og use case beskrivelser for Mini bank

Minibank ønsker et system til administration af en kundes konto. Det kunne være jeres konto! Der er påbegyndt et use case diagram for systemet til Mini bank:



1. Forklar diagrammet. Hvad betyder det fx at både bankrådgiveren og kunden kan kommunikere med use casen: *Hæv penge* fx? Hvad betyder det, at det kun er bankrådgiveren, der kan kommunikere med use casen: *Håndter konto – CRUD*? 
2. Ud fra jeres kendskab til, hvad man skal kunne gøre på en bankkonto, bedes I tilføje et par yderligere use cases i ovenstående diagram
3. I skal nu lave en **brief beskrivelse** af use casen: *Hæv penge* for situationen hvor kunden er aktør. *Brief* beskrivelsen bygges op i en lang tekst på følgende måde:

Use case navn: ????

Use case beskrivelse:

En kunde ankommer til en hæveautomat for at hæve penge (indledning).

Kunden bruger systemet til at ??????

Hent inspiration fra Larman s. 63.

4. Herefter skal I tilføje et par varianter til jeres brief use case beskrivelse i **casual format**:

Alternative scenarier:

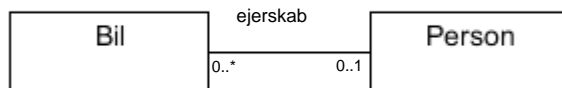
- Hvis kontoen ikke kan identificeres afbrydes processen. Kunden får en fejlmeddelse
 - Hvis
5. Endelig skal I beskrive use casen: *Hæv penge* i *fully dressed* format. Flow of events beskrives som et "happy days" for løb (main succes scenarie) og alternative flows beskrives efterfølgende ved at refererer til trinene i "happy days" hvor afvigelsen sker. Undgå IT detaljer i beskrivelsen.

Use case navn	Hæv penge	
Aktører	Kunde	
Præbetingelse	Kontoen er registreret og der er indsat penge på den	
Postbetingelse	Pengene udbetales og transaktionen er udført	
Frekvens	?? gange jævnt fordelt over dagen . Spidsbelastning???	
Flow of events	Aktør	System
	1. Use-casen starter med at en kunde gerne vil hæve penge fra sin konto	
	2. Kunden angiver kontooplysninger (kreditkort, mobil ...)	3. Finder kontooplysningerne
	4.	5.
	6.	7.
	8.	9.
Alternative flow	3a: Kontoen findes ikke 1: Systemet ???? ?a: Kunden kan ikke huske sin kode.....	

Opgave 3: Færdiggør domænemodellen for et motorregister (øvelse i et tegne domæne- og objektmodeller)

Her er et første udkast til en domænemodel for et simpelt motorregister

Gør modellen færdig:



- Kom med forslag til associationsnavn og multiplicitet (se Larman s. 152 og 153)
- Overvej attributter
- Overvej hvordan I kan illustrere, at en bil har både en ejer og en bruger (se Larman s. 155)
- Kontroller om følgende kan registreres ved at lave et objektdiagram
 - Hans og Grethe ejer en Mercedes i fællesskab
 - Grethe ejer også en Opel, som er en firmabil, der bruges af Sofie
 - En Toyota har haft Hans som ejer, men er nu skrottet
 - Peter har haft en bil, men har ingen pt.

Objektdiagrammet laves ved at tage kopi af domænemodellen, give kopien overskriften objektmodel og arbejde videre på den!

Opgave 4: Domænemodel for Mini bank

Med udgangspunkt i use cases er følgende klasser fundet: En *Konto* klasse til at registrere kontooplysninger og en *Kunde* klasse til at registrere oplysninger om kunden, som har kontoen.

1. Kom med forslag til en domænemodel, hvor I påfører relevante attributter til *Kunde-* og *Kontoklassen* og forbinder dem med en associering, som tilføjes navn og multiplicitet. Lav modellen i UMLet
2. Kontrollere at jeres model er rigtig ved at sætte konkrete data ind i en objektmodel
 - a) Tag først kopi af jeres domænemodel, giv kopien overskriften: Objektmodel. Arbejd videre på den
 - b) Tjek at følgende kan registreres ved at indtegne de aktuelle objekter i objektmodellen
 - a. Hans og Grethe har en kontoen: 1234 i fællesskab. Saldoen er på 1000 kr per 7/7 - 2016
 - b. Grethe også en konto: 5678 alene. Den er på 19999 kr per 7/7 - 2016
 - c. Peter har haft en konto som er nedlagt, men han er stadigvæk registreret i banken

For at kunne registrere pengeoverførsler til kontoen, mangler der en klasse til at registrere hver gang der foretages en hævning eller indsættelse af penge med dato og beløb. Denne klasse kunne man fx kalde for en *Transaktion*.

3. Kom med forslag til attributter på klassen: *Transaktion* og hvordan den skal forbindes til de andre klasser ved associering og multiplicitet.
4. Overvej hvordan det kan registreres, at en transaktion enten kan være en indsættelse eller en hævning
5. Kontroller af følgende kan registreres ved at udvide objektmodellen med de aktuelle objekter:
 - a. 9/7 hæver Grethe 200 kr på konto: 1234 (den der er fælles med Hans)
 - b. 11/7 sætter Hans 500 kr ind på konto: 1234 (den der er fælles med Grethe)
 - c. Hvad er saldoen 11/7?

Der ønskes ikke registret om det er Hans eller Grethe der har hævet eller indsat penge.

6. 14/7 overfører Grethe 100 kr fra konto nr 5678 til fælleskontoen nr 1234. Hvilke konsekvenser har det for domænemodellen?

Opgave 5: Use case beskrivelse for Kajs biler (fortsættelse af opgave fra sidste gang)

Sidste gang fandt I aktører og use cases til ovenstående system som I viste i et UML Use case diagram. I skal nu arbejde videre med beskrivelse af disse use cases.

Use casene i use case diagrammet skal alle beskrives i et *brief* format. De udgør systemets samlede funktionalitet, som godkendes af kunden. Af tidshensyn begrænser vi til de mest kundekritiske.

1. I skal lave en *brief* beskrivelse af use casene: *Udlej bil* og *Aflever bil*. Tag udgangspunkt i jeres opgavebeskrivelser fra sidste gang. Nedenfor er vist eksempel på *brief* beskrivelse af use casen: *Reserver bil*. Husk at use case beskrivelsen bygges op i en lang tekst (Larman kap. 6.1)

Use case: Reserver bil

En kunde henvender sig for at reservere en bil indenfor en ønsket periode (indledning).

Ekspedienten bruger systemet til at tjekke om der er ledige biler, registrere bilønske og kundeoplysninger, oprette reservationen samt udskrive en reservationsbekræftigelse

Use case: Udlej bil

En kunde henvender sig for at hente en reserveret bil. Ekspedienten bruger systemet til at

Use case: Aflever bil

En kunde

Herefter skal use casene prioriteres, og de mest komplekse beskrives *fully dressed*:

2. Hvilken use case mener I skal have den højeste prioritet? Hvorfor?
3. Use casen: *Reserver bil* har fået den højeste prioritet. Den skal I derfor beskrive i *fully dressed* format. Brug nedenstående skabelon. Tag udgangspunkt i jeres brief beskrivelse og mock up's

Use case navn	Reserver bil	
Aktører	Ekspedient	
Præbetingelse	Kunden er ikke registreret. Ønsket prisklasse med tilhørende biler er registreret.	
Postbetingelse	Reservationen er gemt og tilføjet kunde- og prisklasseoplysninger	
Frekvens	20 gange jævnt fordelt over dagen	
Flow of events	Aktør	System
	1. Use-casen starter med at en kunde henvender sig for at reservere en bil	
	2. Ekspedienten.....	3. Systemet
	4.	5.
	6.	7.
	8.	9.
Alternative flow	?a: Der er ingen ledige biler 1: Systemet ???? ?a: Kunden er allerede registreret.....	

Opgave 6: Domænemodel for Kajs biler

Med udgangspunkt i jeres use case beskrivelser fra sidste opgave skal I:

1. Lave en liste over kandidater til klasser
2. Udvælg de klasser I mener er relevante for systemet (det der skal registreres information om)
3. Påføre attributter på klasserne "Kunde", "Reservation", "Prisklasse" og "Bil"
4. Komme med forslag til, hvordan "Prisklasse" - og "Bil" klassen forbindes med en associering. Tilføj navn og multiplicitet. Lav modellen i UMLet
5. Kontroller at det kan registreres at i Prisklasse A koster det fx 750 kr at leje en bil i et døgn og der er 3 biler i prisklassen, fx en VW Up eller lignende, og vis det i en objektmodel