

TNM094 – Medietekniskt kandidatprojekt

Kravhantering och kundkontakt

# Syftet med kravhantering

- Synkroniserar

- krav (från kund)
- behov (hos intressenter)
- plan (för utvecklarna)




- För att förstå problemet

- vad det utvecklade systemet ska kunna göra
- vad det ska bero på andra att göra

- Ska inte

- utforska *hur* problem ska lösas

# Elicitation

- Syfte
  - Samla data
  - Förena olika synsätt och mål
- Källor för elicitation
  - Intervjua intressenter 
  - Tidigare manualer och annan dokumentation över processer, etc.
  - Observera system och processer
  - Praktisera
  - Domänspecifika tekniker  
(t ex “Designprinciper för offentliga digitala tjänster”)
  - Gruppdiskussioner och Brain Storming

# Intressenter

- Kund
  - Slutanvändare
  - Administratörer
  - Domänexperter
  - Marknadsforskare
  - Jurister
  - Teknikexperter
  - Utvecklare / underhållsansvariga
  - Utbildningsansvariga
- etc.

# Att tänka på vid intervju

- Intressenterna är oftast inte utvecklare
  - känner inte till fallgropar
  - vet inte vad som är viktigt eller oviktigt, dyrt eller enkelt
  - vet inte hur kraven leder till färdigt system
- Aktivt lyssnande
  - utforska metodiskt alla aspekter och behov
  - använd listor för att säkert täcka alla aspekter
  - använd följdfrågor för att utforska även det underförstådda
  - diskutera både funktionella och icke-funktionella aspekter

# Att lösa konflikter

- Typiska konflikter
  - inkompatibla krav
  - intressenters inkompatibla behov eller synsätt
  - balans mellan kostnad, funktioner och schema
- Förhandlingssteg
  - agera professionellt och separera människorna från problemen
  - fokusera på intressen, inte positioner
  - titta på de bakomliggande behoven – ”varför” före ”vad”
  - hitta alternativ för ömsesidig vinning
  - insistera på att använda objektiva kriterier
- Prioritera behov
  - informellt eller numeriskt
  - tvingar kunden att välja

# Modellering och prototypning

- Utforska frågor genom att visa
  - Är lösning möjlig och kostnadseffektiv?
  - Utforska alternativa funktioner
  - Hantera ”jag vet när jag ser det”-kunder
  - Assistera i prioritering
- Prototypning
  - Mock-up-gränssnitt
    - ritat på papper eller kartong
    - icke-fungerande GUI i datorn
  - Implementera partiell funktionalitet

# Detaljnivå i kravspecifikationen

- Det finns ingen "rätt" nivå
  - för mycket information gör kravet svårförstått
  - för lite information gör kravet underspecificerat
- Rekommendationer
  - ett krav per rad i en tabell eller post i backlogg
  - undvik kors-referenser
  - gruppera liknande krav



# Kravhantering i Agil utveckling

- Krav är främst kodade som *poster* ("stories")
  - en post ska kunna genomföras i en sprint
  - 5W: who, what, when, where, why
  - "As a <role>, I want <feature/fix>, so that <motivation> (+ <acceptance criteria>)"
- Vad som ska göras kodas som *uppgifter*
  - vilka steg som ska genomföras för en post
  - en uppgift ska kunna genomföras på mindre än en dag
- Hierarkiska poster
  - en slags Work Breakdown Structure (WBS)
    - Grupperar t ex genom tema-poster, episka poster, etc
    - Delar upp större poster i mindre poster

# Exempel

- Post

Som användare vill jag kunna använda mig av Facebook-inloggning så att jag inte behöver skapa ett nytt konto för den här tjänsten.

- Uppgifter

- Analysera Facebook API
- Förbered login-modulen för att tillåta externa konton
- Lägga till Facebook-inloggning i login-modulen
- Inkludera den nya funktionen i GUI:t
- Testa den nya funktionaliteten

# Kravhantering i Agil utveckling

- I Scrum
  - Produkt-backlogg – kravspecifikationen
    - innehåller poster för alla krav som gäller för tillfället
    - kontrolleras av produktägaren
  - Sprint-backlogg – att-göra-listan
    - innehåller alla uppgifter som ska genomföras i sprinten
    - kontrolleras av teamet

# Kravhantering i Agil utveckling

- En backlogg är en prioriterad lista
  - viktigaste posterna (för kunden) listas högst upp
  - hämta från toppen i produkt-backlogg till sprint-backlogg
- En produkt-backlogg är inte fullständig
  - endast de högst prioriterade behöver vara "kompleta"
  - ju högre på listan desto mer detaljerad
  - poster längre ned på listan:  
underspecificerade, otydliga, för stora, osäkra
- Backlogg synliggör arbetet
  - Visar på vad som behöver göras
  - Tydliggör projektets progression
  - Utlämna därför inte "underförstått" arbete

# Olika sorters krav

- Funktionella krav

- funktionalitet
- data-stöd

- Kvalitets-krav

- prestanda, användbarhet
- säkerhet, pålitlighet
- underhållbarhet
- precision och korrekthet
- schema och kostnad

- Process-krav

- resurser
- dokumentation
- standarder

- Design-krav

- fysisk placering
- fysiska begränsningar
- plattform och gränssnitt
- användare

# Testbarhet

- Testbarhet hos funktionella krav
  - arbetsflöden som visar på normalt och felaktigt användande
  - förhandsvillkor och resulterande tillstånd för viss inmatning etc.
- Testbarhet hos kvalitetskrav
  - Fel:
    - “ska svara snabbt”
    - “ska vara lättanvänt”
    - “ska kunna hantera många uppkopplingar samtidigt”
  - Rätt:
    - “ska svara inom 50 ms, och inom 10 ms i genomsnitt”
    - “åtminstone 50% av målgruppen ska kunna logga in utan tidigare...”
    - “ska hantera 50k uppkopplingar med en maximal fördröjning på...”

# Dokumentation

- Är oftast en organisationsstandard
  - IEEE, DoD, SKF, Ericsson, etc.
  - En del av den interna processen
- I Agil utveckling
  - Res så lätt som möjligt, men inte lättare än så
  - Oftast enkla listor
  - Använd diagram och modeller där de hjälper

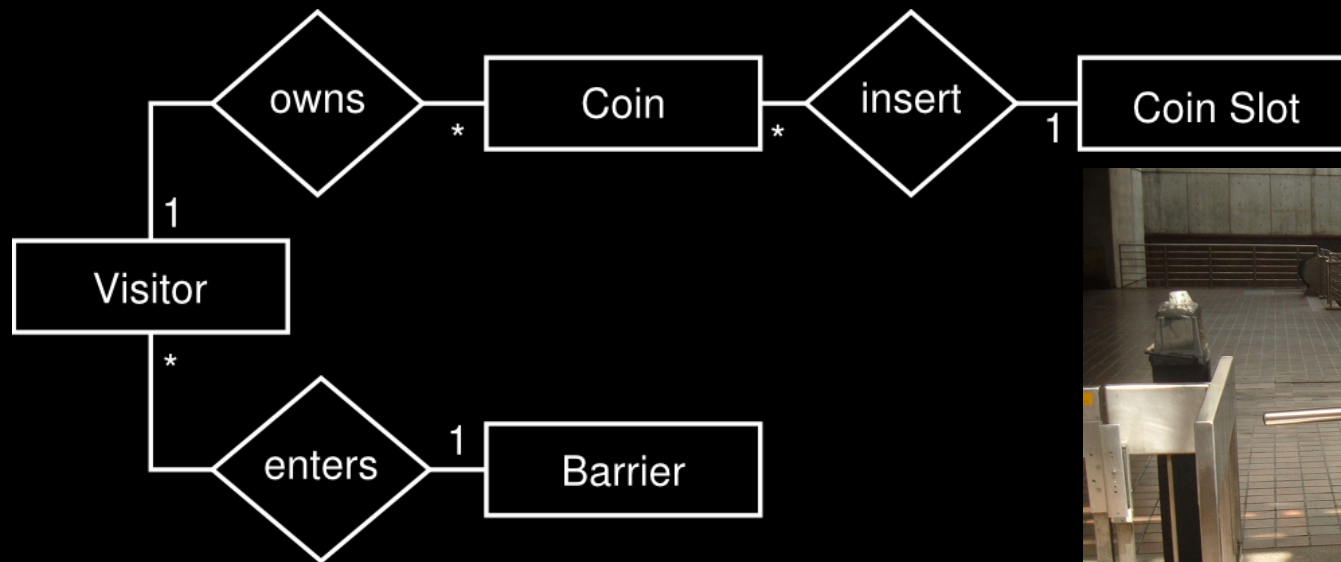
# Modellering

- Ingen teknik passar i alla lägen
  - kombinera tekniker som passar
- Till exempel
  - textuell beskrivning i listor
  - relation mellan aktiviteter, entiteter och operationer
  - funktioners förhandsvillkor och resulterande tillstånd
  - sekvenser av input, operationer och output
- Att tänka på
  - en bild eller figur kan förtydliga avsevärt!
  - kontrollera lämplighet – tydliggörande eller förvirrande
  - dokumentera bara krav (problem), inte lösningar



# Relationen mellan kravets delar

- ER-diagram
  - ER-modeller eller klass-diagram (UML)



# Förhandsvillkor och resultat

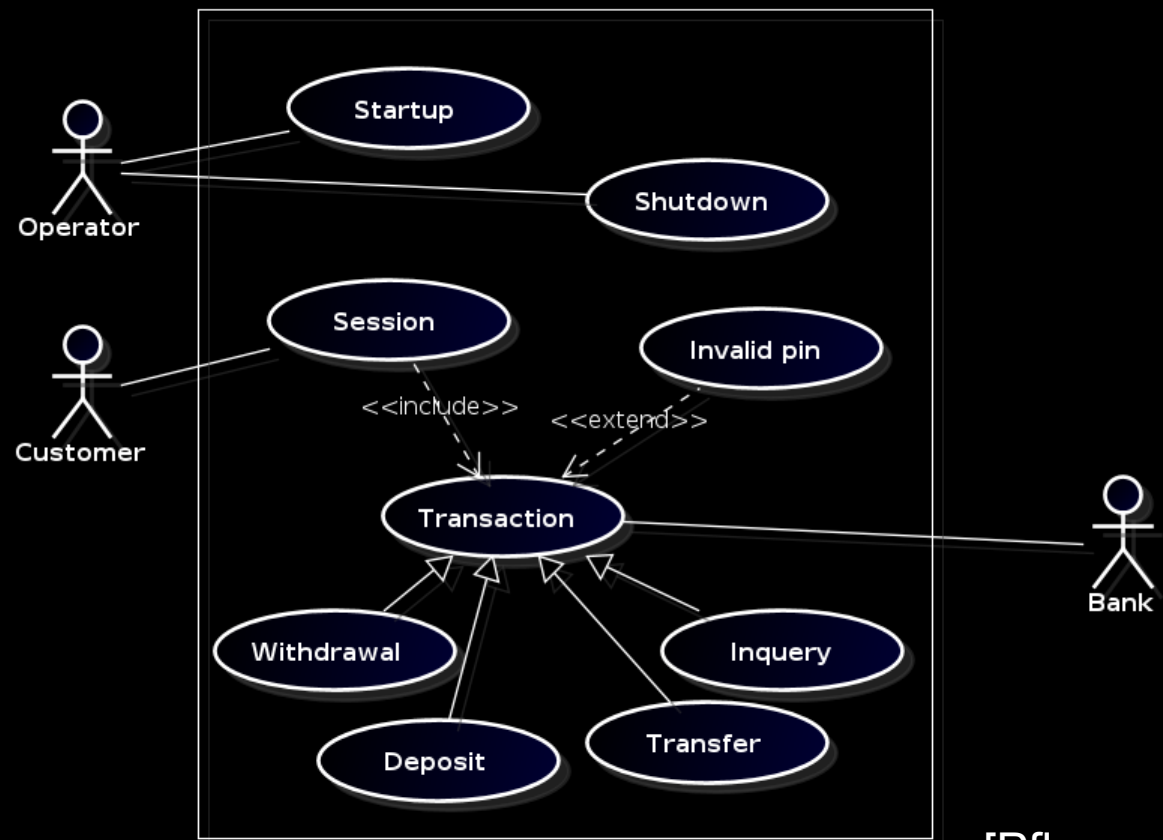
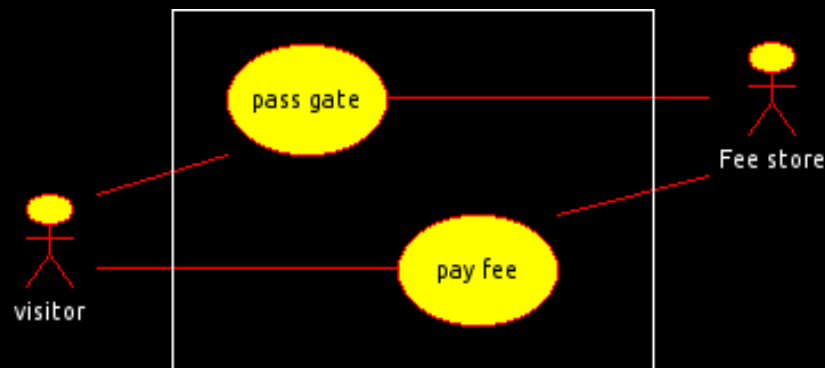
- Funktionell notation
  - matematiska uttryck
  - beslutstabeller
- Logiska uttryck
  - Första ordningens logik
    - $\text{num\_coins} \leq \text{num\_entries}$
    - $(\text{num\_coins} > \text{num\_entries}) \Leftrightarrow (\text{barrier} = \text{unlocked})$
    - $(\text{barrier} = \text{locked}) \Leftrightarrow \neg \text{may\_enter}$
  - Temporallogik

coin entered	F	T	F
slug entered	F	-	T
pushing	T	T	-
buzzing			X
rotating		X	
reject event	X		

# Relationer

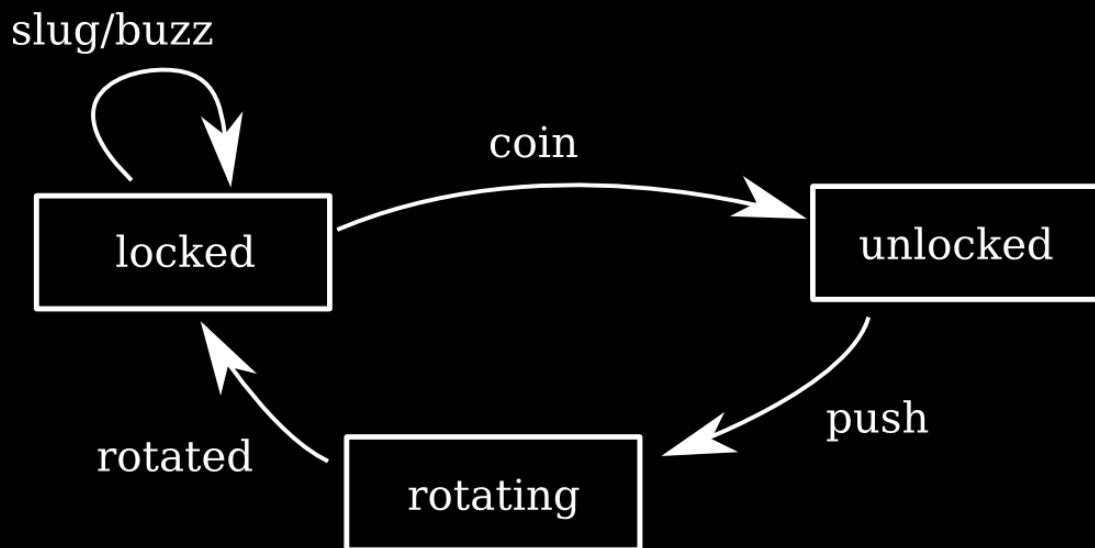
- UML

- Use case-diagram
- Komponent-diagram  
etc

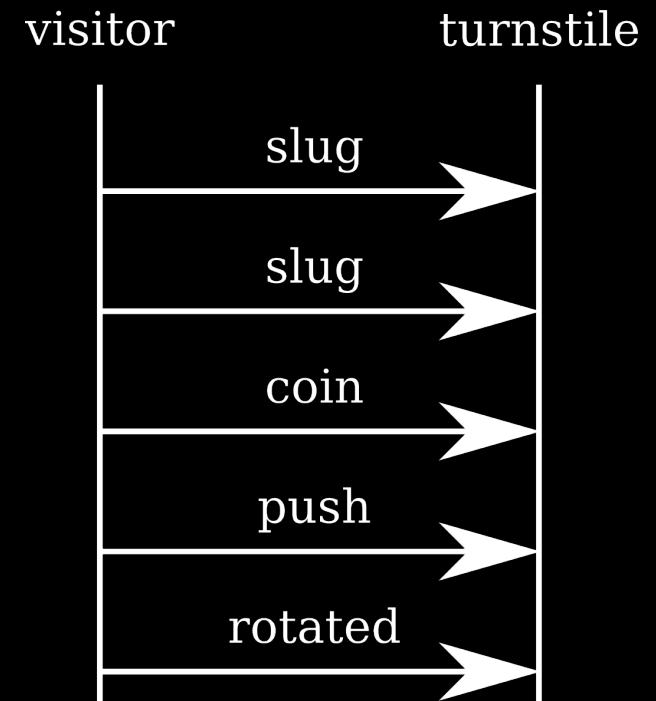


# Sekvenser

- Tillståndsmaskiner
  - Text UML tillståndsdigram, Petri-nät



- Händelsespårning
  - Text UML sekvensdiagram



# För ert projekt

- I början – projektplanen
  - en vision – vad är det för system vi siktar på
  - övergripande krav – tema-poster
- Under projektets gång
  - dela upp det viktigaste temat i mer detaljerade krav
  - förfina och lägg till detaljer
  - skapa mätbarhet
  - plocka de viktigaste och tydligaste kraven i varje sprint