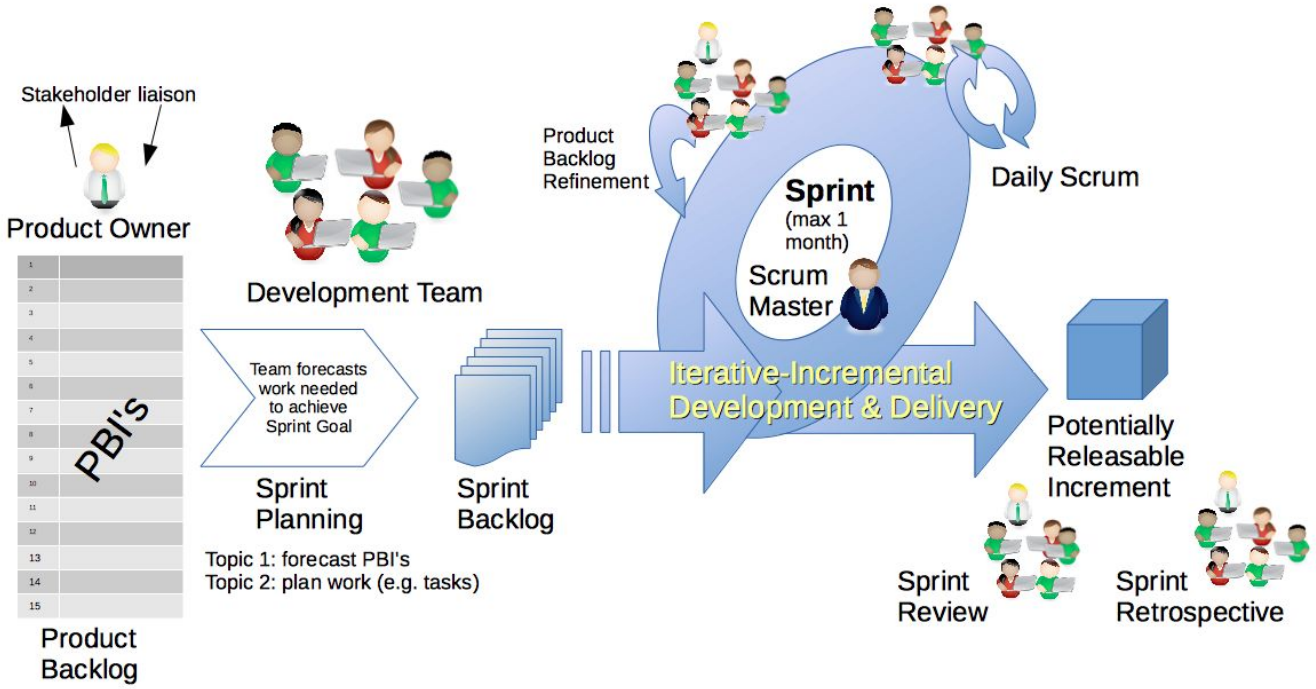
Kravhantering

Tobias Ohlsson, Linnéuniversitetet

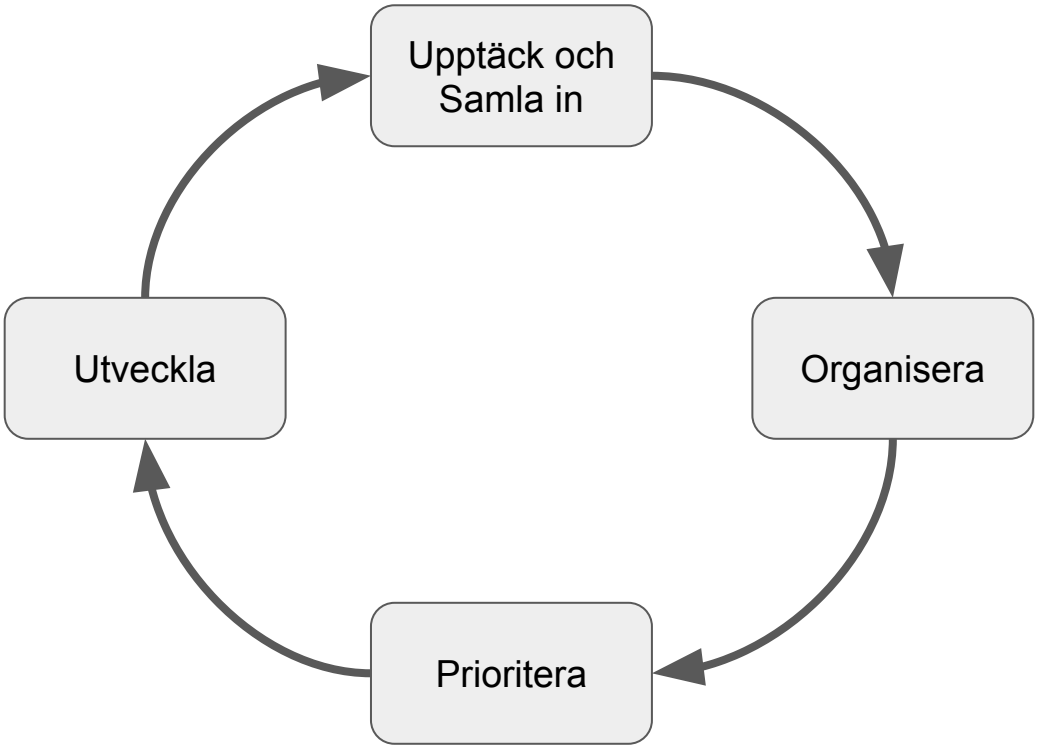
Kravhantering innefattar de roller, artefakter och aktiviteter som genomförs med syfte att hantera krav.

Insamling, Dokumentation, Analys, Validering, Förändring.

SCRUM


Stakeholder liaison
Product Owner
PBI's
Product Backlog
Development Team
Team forecasts work needed to achieve Sprint Goal
Sprint Planning
Topic 1: forecast PBI's
Topic 2: plan work (e.g. tasks)
Sprint Backlog
Product Backlog Refinement
Sprint
(max 1 month)
Scrum Master
Daily Scrum
Iterative-Incremental Development & Delivery
Potentially Releasable Increment
Sprint Review
Sprint Retrospective

Insamling, Analys, Dokumentation



Kravkällor

Högnivåkrav

- Marknadsanalys, Affärsutveckling

Användarkrav/Systemkrav

- Slutanvändare, Kund, Domänexperter, Utvecklare: Intressenter
- Gamla system, Liknande system, Litteratur, Standarder, Lagar

Kravinsamling

Interaktiv

- Intervjuer, Workshops, Brainstorming
- Fråga, Visa, Arbeta Tillsammans
- Kräver Tid och Tillgång
- Formell kunskap - sådant en intressent kan/vill uppge

Observerande

- Etnografi - studera hur intressenterna faktiskt arbetar i verkligheten
- Datainsamling - samla in data kring hur ett system faktiskt används
- Informell kunskap

Kravanalys

Struktukturera och organisera kraven så att helheten går att förstå

- Hantera dubbletter
- Dela upp sammansatta krav
- Höj/Sänk Nivån
- Klassificera
- Gruppera
- Hitta kopplingar och beroenden
- Realism/Genomförbarhet/Testbarhet
- Motsägelser
- Frågor

Prioritera

Avgör vilka krav som är viktigast att arbeta vidare med

Risk

- Hur svårt, Hur okänt, Hur stort, Genomförbart?

Täckning

- Hur stor del av systemet påverkas av kravet

Nytta

- Hur viktigt är kravet för användare

Utveckling

För kraven vidare i processen

- Dela upp i konkreta arbetsuppgifter som sedan implementeras
- Skriva testfall
- Prototyper/Skisser för ytterligare kravinsamling
- Specificera kravet formellt i en Kravspecifikation

Dokumentation

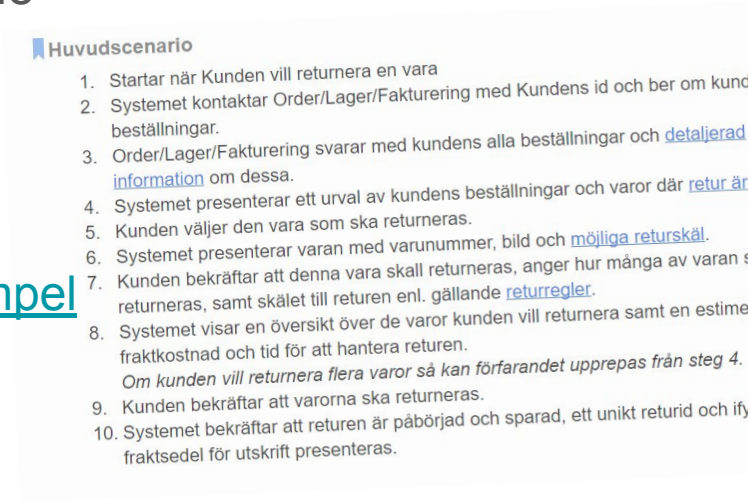
Formella

- Matematiska definitioner av systemets beteende

Naturliga

- Kravlistor - “Systemet skall”
- Visuella - Skisser/Bilder/Flöden
- Scenarios, Användningsfall (Use Cases), Exempel
- User Stories - “Som X vill jag Y så att Z”

Inget är bäst i alla situationer - var inte religiös.



Huvudscenario

1. Startar när Kunden vill returnera en vara
2. Systemet kontaktar Order/Lager/Fakturering med Kundens id och ber om kund beställningar.
3. Order/Lager/Fakturering svarar med kundens alla beställningar och detaljerad information om dessa.
4. Systemet presenterar ett urval av kundens beställningar och varor där retur är
5. Kunden väljer den vara som ska returneras.
6. Systemet presenterar varan med varunummer, bild och möjliga returskäl.
7. Kunden bekräftar att denna vara skall returneras, anger hur många av varan s returneras, samt skälet till returen enl. gällande returregler.
8. Systemet visar en översikt över de varor kunden vill returnera samt en estime fraktkostnad och tid för att hantera returen.
   Om kunden vill returnera flera varor så kan förfarandet upprepas från steg 4.
9. Kunden bekräftar att varorna ska returneras.
10. Systemet bekräftar att returen är påbörjad och sparad, ett unikt returid och ify fraktsedel för utskrift presenteras.

Krav bör ha

- Unikt id
- Tydligt förklarande namn
- Referenser (vem har skrivit, vem är källan, när)
- Status (implementerat, testat, ogiltigt, …)
- Tydlig beskrivning av kravet som går att förstå
- Prioritet
- Beroenden till andra krav (X måste göras först)
- Spårbarhet till testfall
- Spårbarhet till implementation

Kravdokument

Vision

- Hög nivå
- Problem, Intressenter, Användare, Marknad, USP

Kravspecifikation

- De organiserade kraven
- Relativt låg nivå, relativt detaljerat
- Kan ligga till grund för ett kontrakt

Kravspecifikation

- Problembeskrivning och Övergripande mål
- Intressenter (Kund, Utvecklare, Användare, Tredjepart, Samhälle)
- Slutanvändare (Roller, Grupper)
- Ordlista (Definitioner av viktiga begrepp och förkortningar)
- Krav
  - Per delsystem/användarroll (om relevant)
  - Funktionella (Kanske hänvisning till product backlog)
  - Ickefunktionella (Kanske fokus på dessa då de påverkar helheten)
  - Nivå: Användarkrav & Systemkrav
- Exempel RetSys (tidigt i ett projekt, med fokus på scenarios)
- Exempel Ganesha (en klassisk “färdig” kravspec)

Validering

Löser vi rätt problem?

- Implementera → Leverera → Återkoppling

Hitta problem i kraven:

- Är kraven giltiga? Användare, kund och omvärlden kan ändra sig, vet vi mer?
- Är kraven konsekventa? Finns det motsägelser?
- Är kraven kompletta? Har vi missat någon del eller användare?
- Är kraven realistiska? Kan vi åstadkomme detta? Kostnad?
- Är kraven testbara? Kan vi objektivt säga att kraven är uppfyllda?

Tekniker för Validering

Implementera → Leverera → Återkoppling

Inspektera kraven

- Internt
- Externt (Kund & Slutanvändare)

Prototyper

- Enklare skisser eller implementationer
- Säkerställ giltighet (visa slutanvändare)
- Säkerställ genomförbarhet (teknisk prototyp)

Testfall

- Säkerställ testbarhet genom att skriva testfall

Förändring

Över tid kommer kraven att behöva ändras

- X var svårare/tog längre tid än vi trodde
- “Ja det var det vi bestämde, men nu när vi ser det är det inte vad vi vill ha”
- Teknik X är nu obsolet
- Konkurrent X släppte precis feature Z

Omförhandling

- Det sämsta alternativet är att hålla fast vid dåliga krav

Förändringsanalys

- Ändrar vi ett krav riskerar vi att påverka flera andra krav
- Vilken kostnad har förändring av ett krav

Tips

Tekniker för kravhantering ska givetvis kombineras och anpassas

Krav fokuserar på "vad/vilken/vem" inte "hur", jämför med user stories.

- Vad vill användare X egentligen uppnå?
- Vilken information behöver mjukvaran för att kunna hjälpa X
- Vilken information vill X ha tillbaka?

Engagemang hos kund och slutanvändare är ett måste

- Fråga innan så känner sig folk delaktiga
- Acceptera och uppmuntra förändring
- Leverera och samla återkoppling på ett strukturerat sätt
- Hantera föreslagna förändringar så känner användare att de kan påverka.
- Kräv tid med slutanvändare

Hitta rätt nivå på dokumentation

- Vem ska läsa, hur stort team, distribuerat team, testning, användardokumentation, återanvändning
- Förändra det som inte funkar!