Projektarbete

Grunder

Projektarbete

Hur gör man på Spotify, på ett modernt ICT-företag?

Se ”Spotify Engineering Culture” (film)

Källa: http://labs.spotify.com/2014/03/27/spotify-engineering-culture-part-1/

Kursmål, lärandemål

Kursens övergripande mål är att ge kunskaper om ingenjörsmässiga arbetsmetoder och att ge grundläggande färdigheter i att använda olika ingenjörsverktyg, med betoning på **projektmetodik**, presentationsteknik och datorn som arbetsverktyg.

Det innebär att studenten efter genomgången kurs skall kunna:

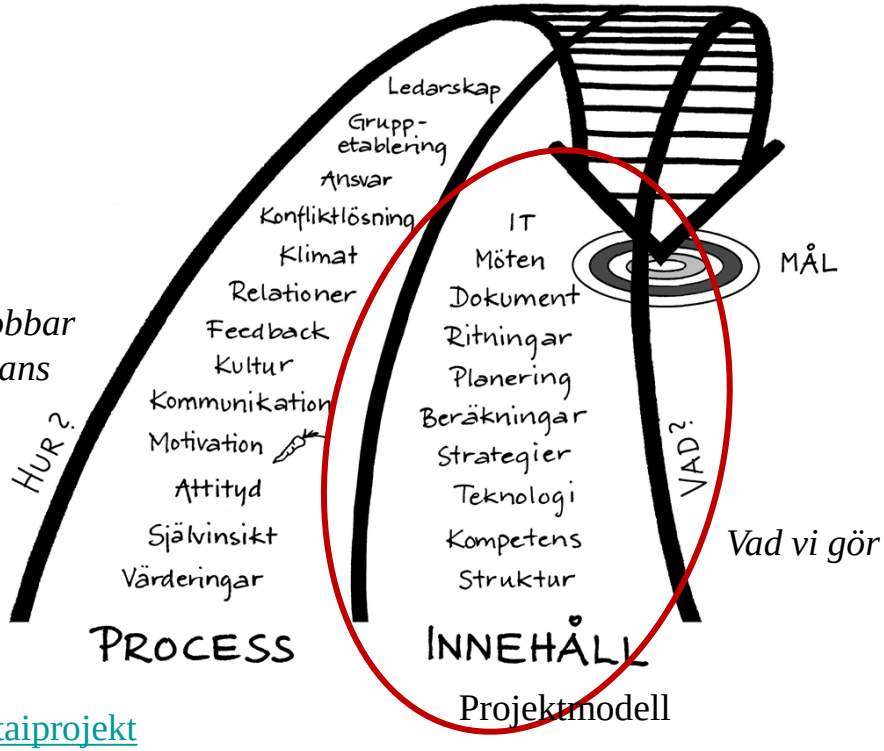
- **beskriva och jämföra olika typer av utvecklingsprocesser/projektprocesser.**
- **kunna delta i, och på ett strukturerat sätt, genomföra ett enkelt projekt i grupp om max 8 studenter.**
- **för enklare problem tillämpa viktiga verktyg och metoder som stödjer vald utvecklingsprocess.**
- **reflektera över genomfört projekt ur angivna aspekter.**
- övning i rapportskrivning.
- kunna tillämpa metoder för muntlig presentationsteknik.
- kunna skapa en enkel hemsida för att presentera resultat och dela infor
- kunna studieplanera enligt vald personlig modell
- kunna reflektera över sin framtida yrkesroll ur olika aspekter.
- kunna ange några perspektiv på "hållbar utveckling" som är relevant för en ingenjör.
- kunna ange några etisk/moraliska aspekter som är relevant för en ingenjör.
- kunna reflektera över gruppdynamiska skeenden i en projektgrupp.

Läraktivitet
- Föreläsning/lektion
- Projektarbete i grupp
- Lektion/handledning
- Kursbok
- Egna studier

Syfte med denna föreläsning

- Introduktion till projektarbete - **överblick**
  - Ta upp delar ur kursboken ur (kapitel 1, 8-12),
  - men även annat
- Tips
- Läs boken!
  Det mesta tas inte upp nu

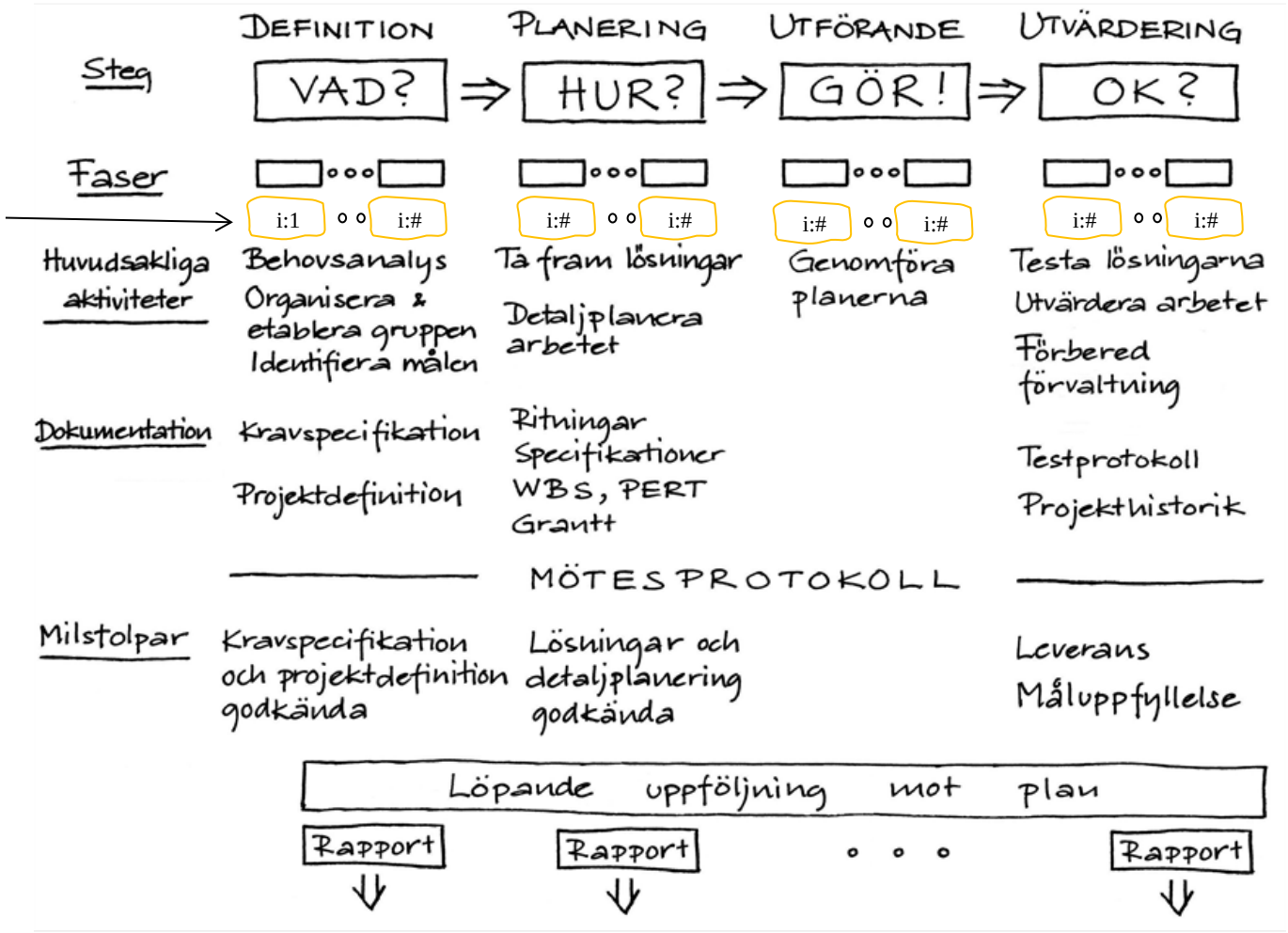
Hur vi jobbar tillsammans



https://www.studentlitteratur.se/campusproducts/arbetaiprojekt/index.html

Ur Arbeta i projekt – individen, gruppen, ledaren. © Sven Eklund

Generell projektmodell

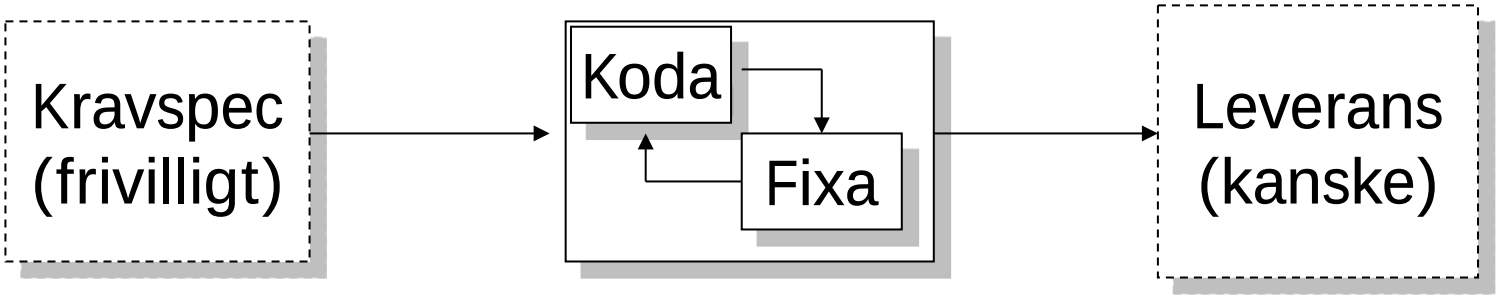


Ur Arbeta i projekt – individen, gruppen, ledaren. © Sven Eklund

Projektmodeller IT, grundläggande

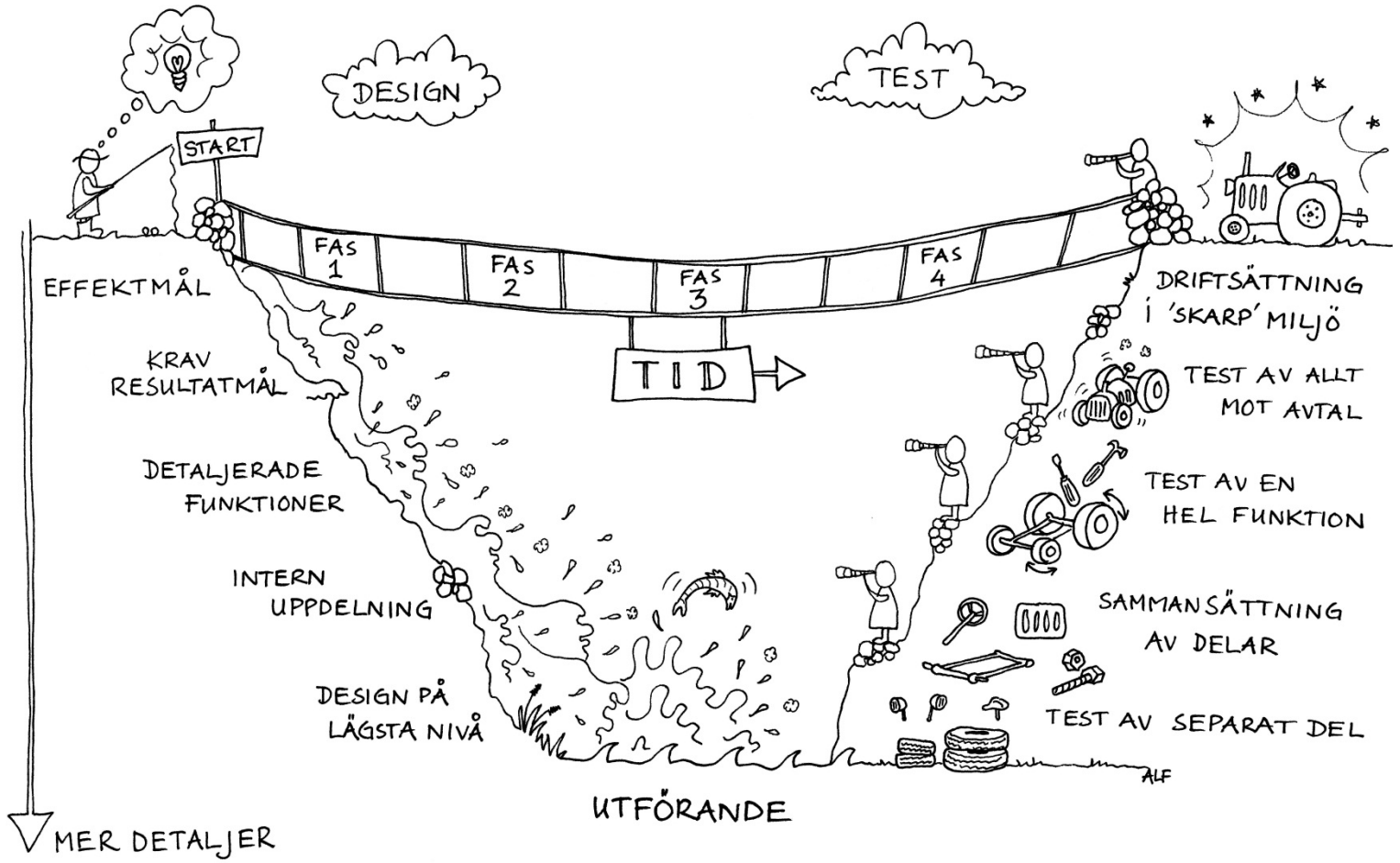
- Koda och fixa ”happy hacking”
- Vattenfall (den klassiska modellen)
- Iterativ och inkrementell
  - Lättrörlig – ”agilt” (agile), ”Scrum”, ”Kanban”, XP (eXtreme Programming), RUP, UP, ...

”Koda och fixa” – ”happy hacking”

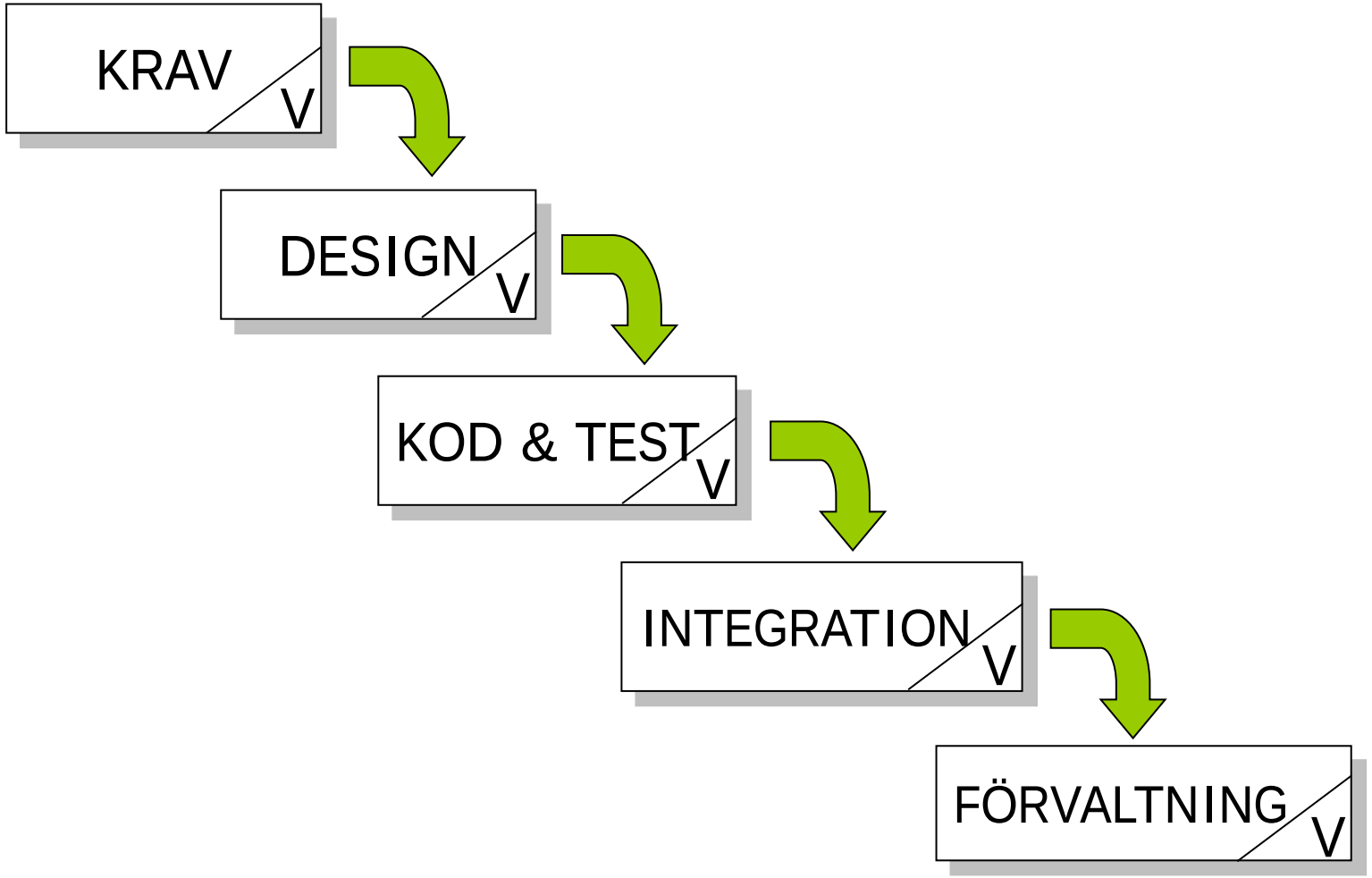


- Hanterar inte problemen vid utveckling
- Kan ändå användas ibland:
  - Ingen overhead => snabb.
  - Kräver ingen process kunskap => oerfaren personal kan användas
  - Användbar för små subprojekt där koden strax skall kastas (GUI-prototyp... Graphical User Interface)

Vattenfallsmodellen



Vattenfallsmodellen



Vattenfallsmodellen

- Också känd som:
  - Den klassiska livscykelmodellen
  - ”Once-through”
  - “Big bang integration”.
  - Sekventiell process
- Processen flödar bara i en riktning, varav namnet vattenfall.
  - Det *går* att gå tillbaka, men det kostar. Processen är byggd på att man inte får en chans till att revidera innevarande steg senare varför man lägger ned mycket energi på att få allt rätt från början innan man går till nästa steg...

Vattenfallsmodellen - positivt

- Indelning i discipliner (=faser) =>
  - möjligt att dela upp arbete mellan utvecklare.
- Arbetsuppdelningen =>
  - utvecklare kan specialisera sig.
  - kallas in när de behövs
- Seniora utvecklare i de tidiga faserna =>
  - de som vet hur det skall se ut och göras
  - juniora utvecklare kan vara produktiva... i de senare faserna

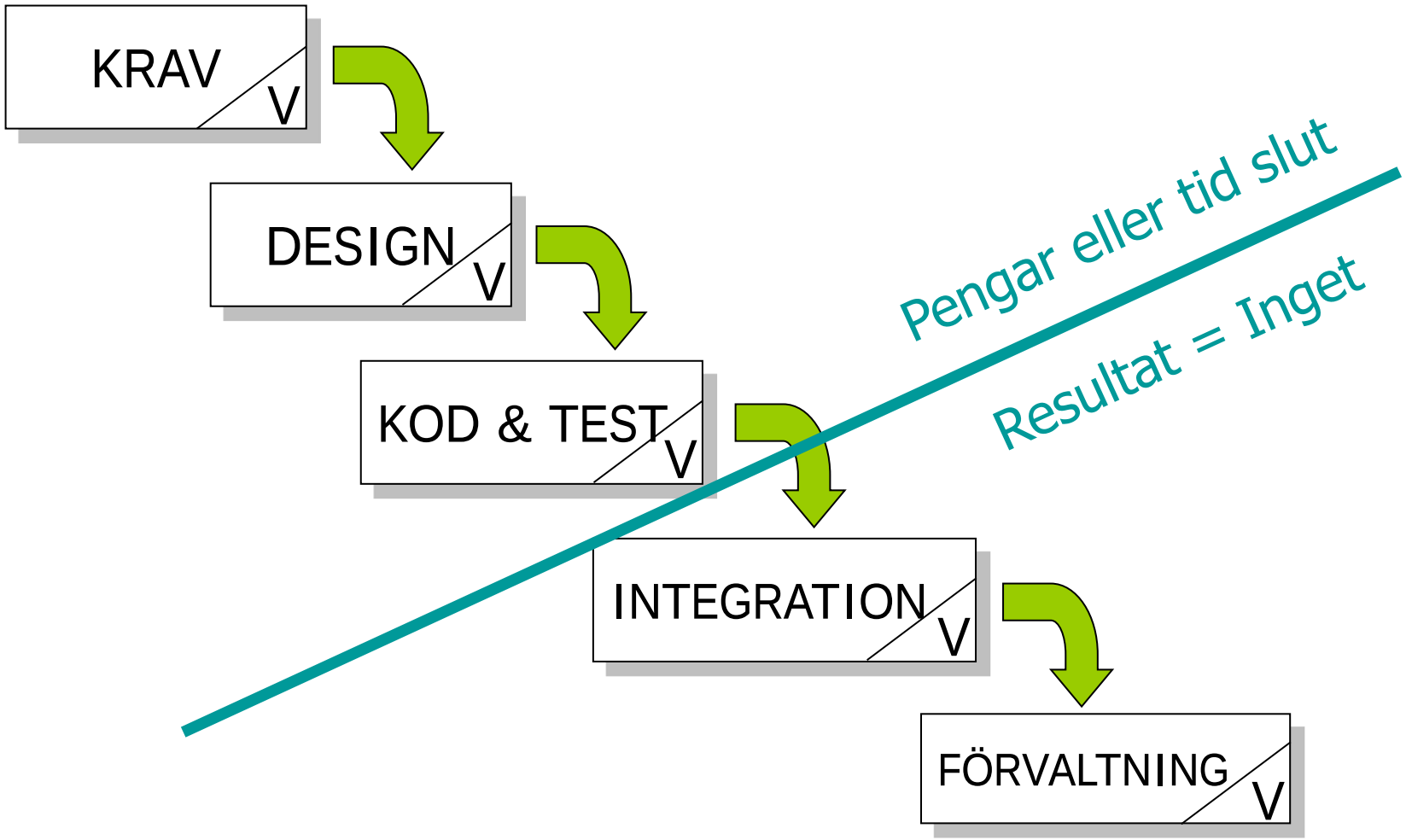
[DeGrace 1990]

Vattenfall - problem

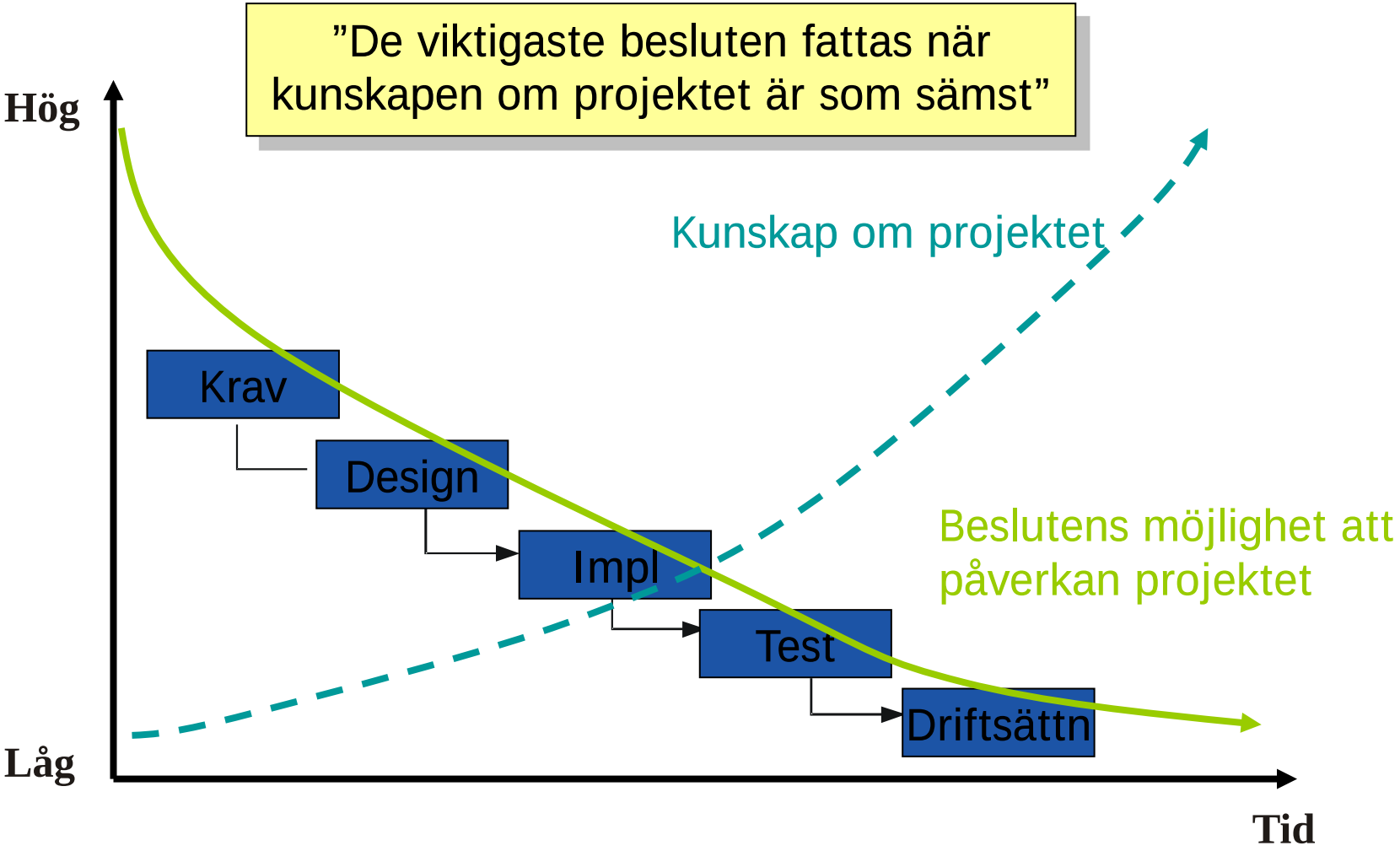
- Förståelse för problemet nås ofta först efter vi börjat med lösningen
  - Krav är vanligen ofullkomliga;
    ”*I Know It When I See It*” (IKIWISI)…
- Osynlig produkt till slutet av projektet.
  - Fokus på projektet (dokument) ej på produkten (kod).
  - Slutanvändarens feedback kommer för sent
  - Det tar lång tid innan problem syns
- Seniora utvecklarna drar vidare efter de tidiga faserna… vem skall då de juniora fråga?

[DeGrace 1990]

Vattenfall – fler problem

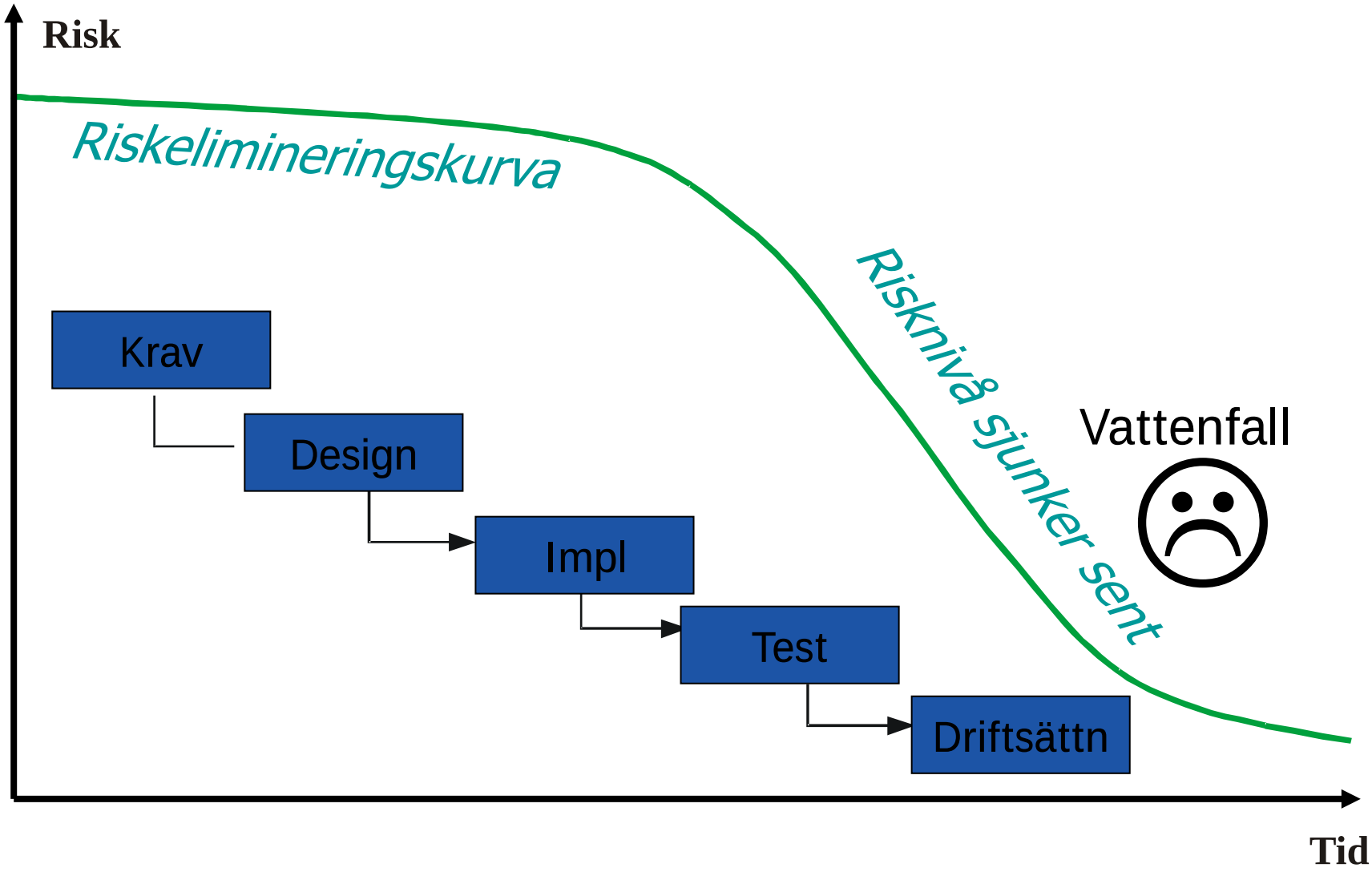


Vattenfall - problem



[Wenell 2001, s 48, modif]

Vattenfall, risker elimineras sent!



Vattenfall, användningsområde

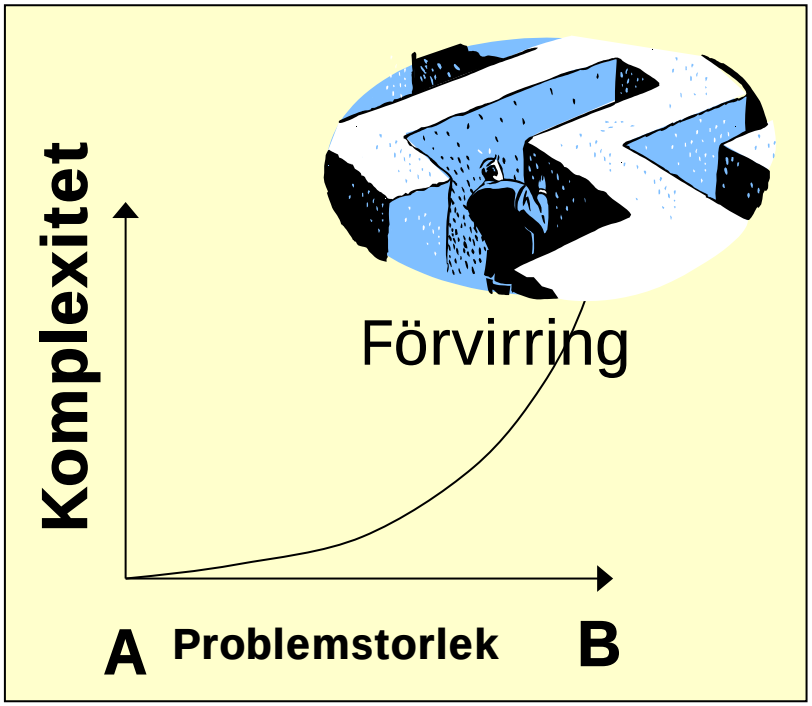
- Kan användas när:
  - kraven är väl kända och
  - arkitekturen är väl känd och
  - det finns tillräckligt med kalendertid för att arbeta sekventiellt
- Exempel på rimligt användande:
  - anpassning av en produkt ur en produktlinje till en viss kund
  - detta har vi gjort förr

[DeGrace 1990, Boehm 2000 s 8,]

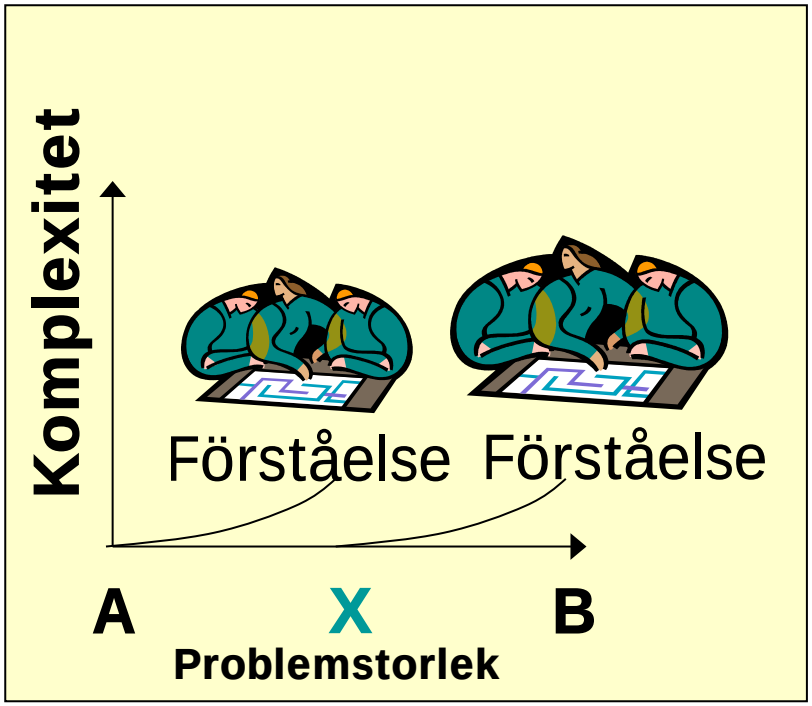
Åtgärda problemen med vattenfall

Dela upp problemet i mindre bitar och
lös bit för bit (“Divide and conquer”)

Vattenfallsmodellen

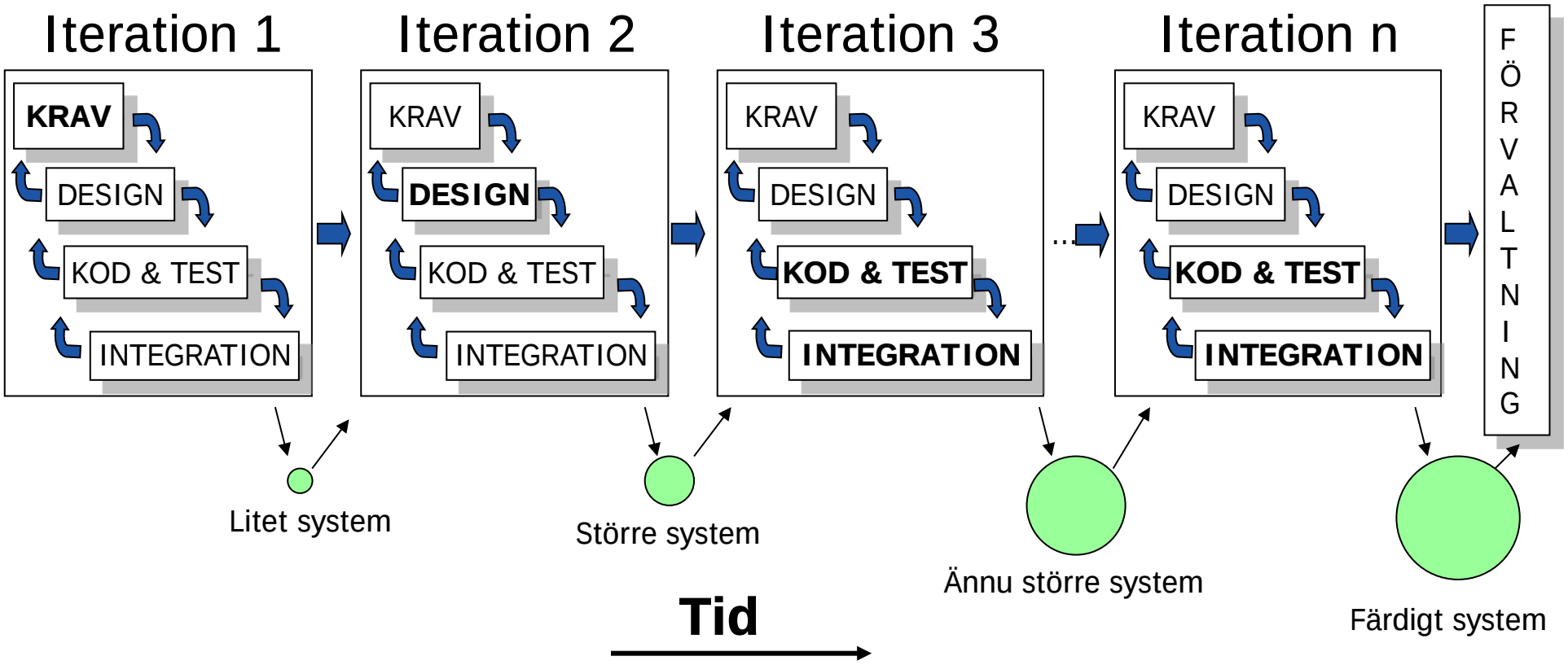


Iterativt & inkrementellt

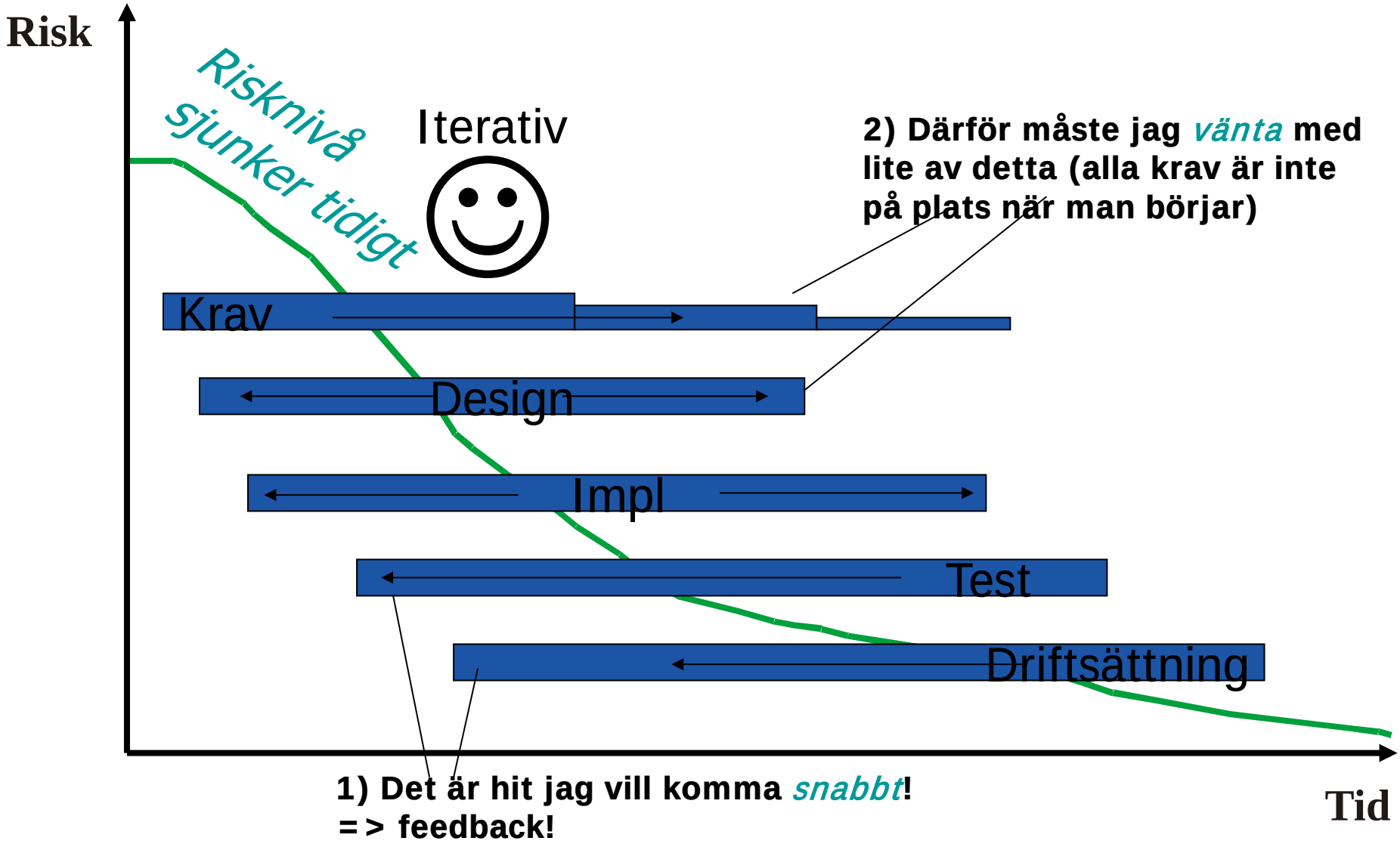


Fritt efter: [Weinberg 1982 s 93]

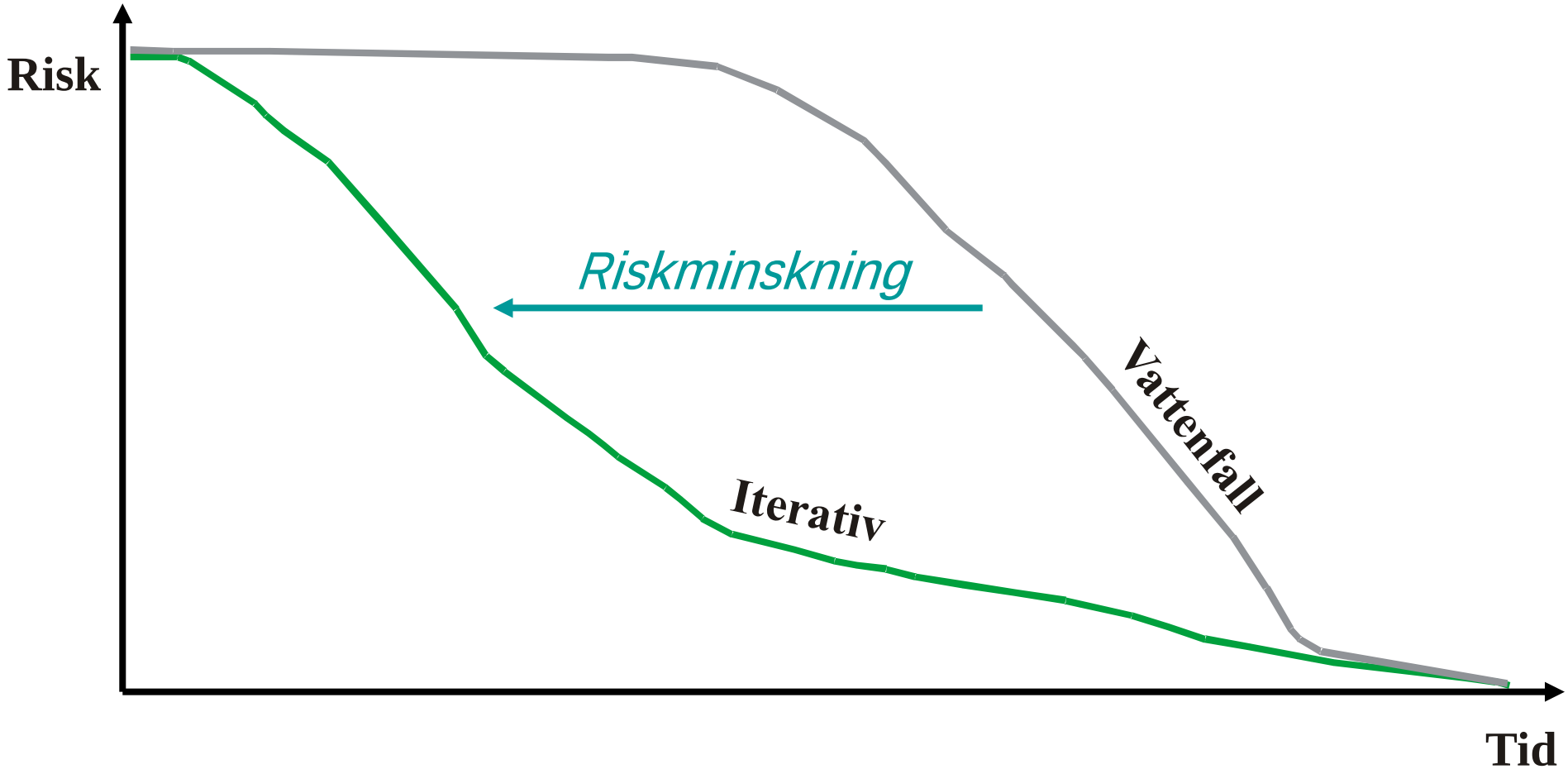
Iterativ och inkrementell utveckling



Attackera (möt) riskerna

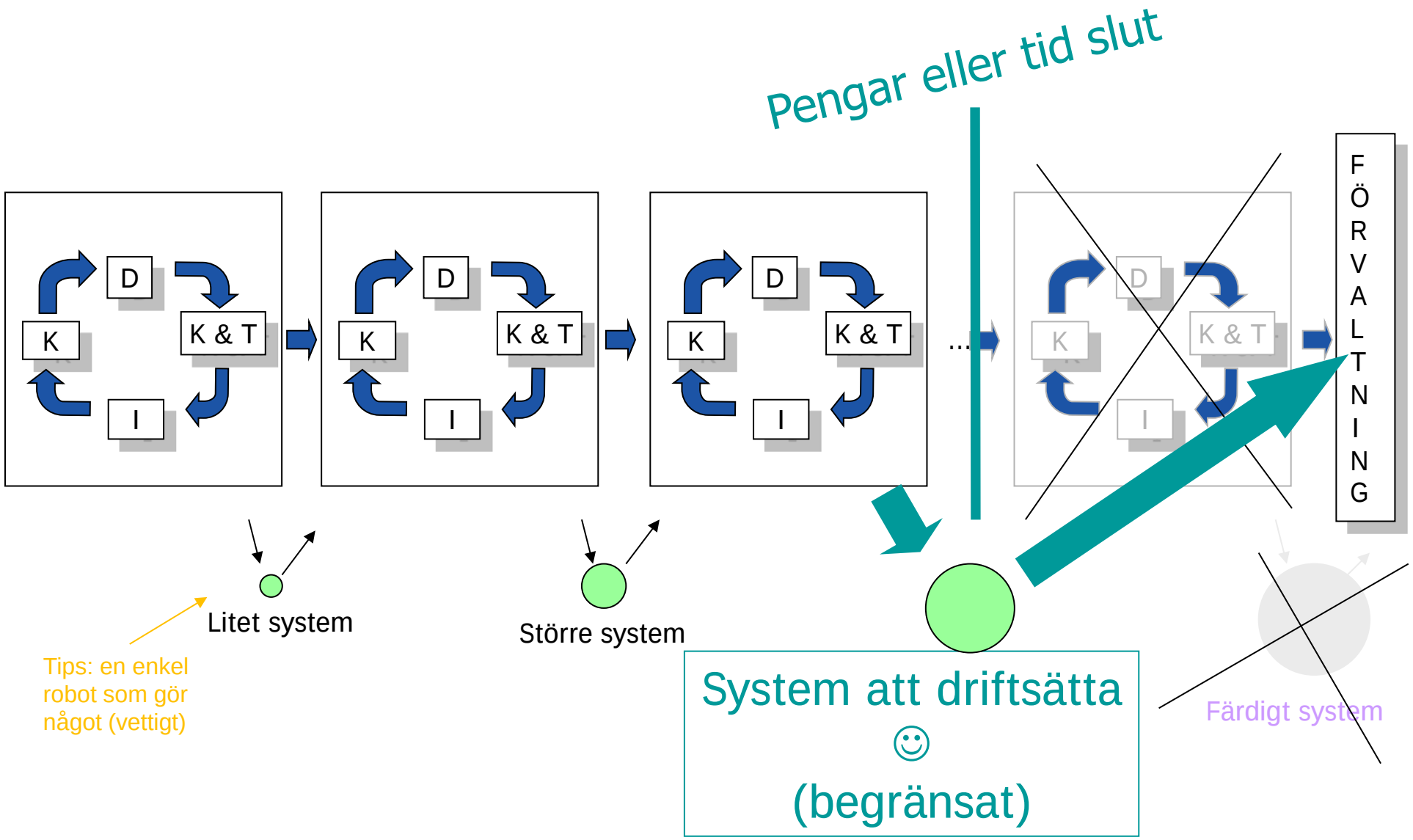


Riskreducering



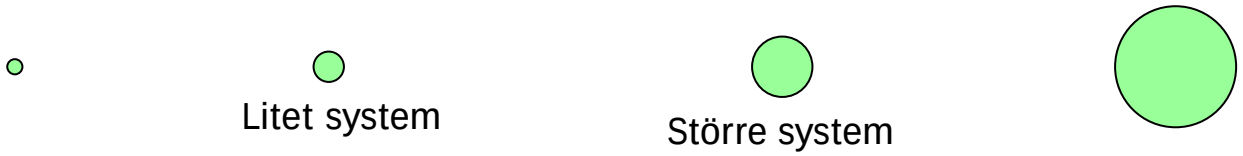
[Kroll 2003, fig 2.1]

Iterativ och inkrementell utveckling - fördelar



Iterativ och inkrementell utveckling

Produktdemonstration efter varje iteration!



Iterativ och inkrementell utveckling

- Lämpar sig för kravförändringar
  - Slår bara fast de kraven som ska byggas närmaste framtiden
- Kontinuerlig integration =>
  - tidigare feedback på arkitektur/designmissar
  - tidigare användarfeedback: ”Rätt produkt?”
  - lättare hitta orsak till buggar
- Risker upptäcks/fixas under tidiga integrationer

Iterativ och inkrementell utveckling kan vara svårstyrd

Processen riskerar bli svårstyrd / osynlig, inga naturliga milstolpar

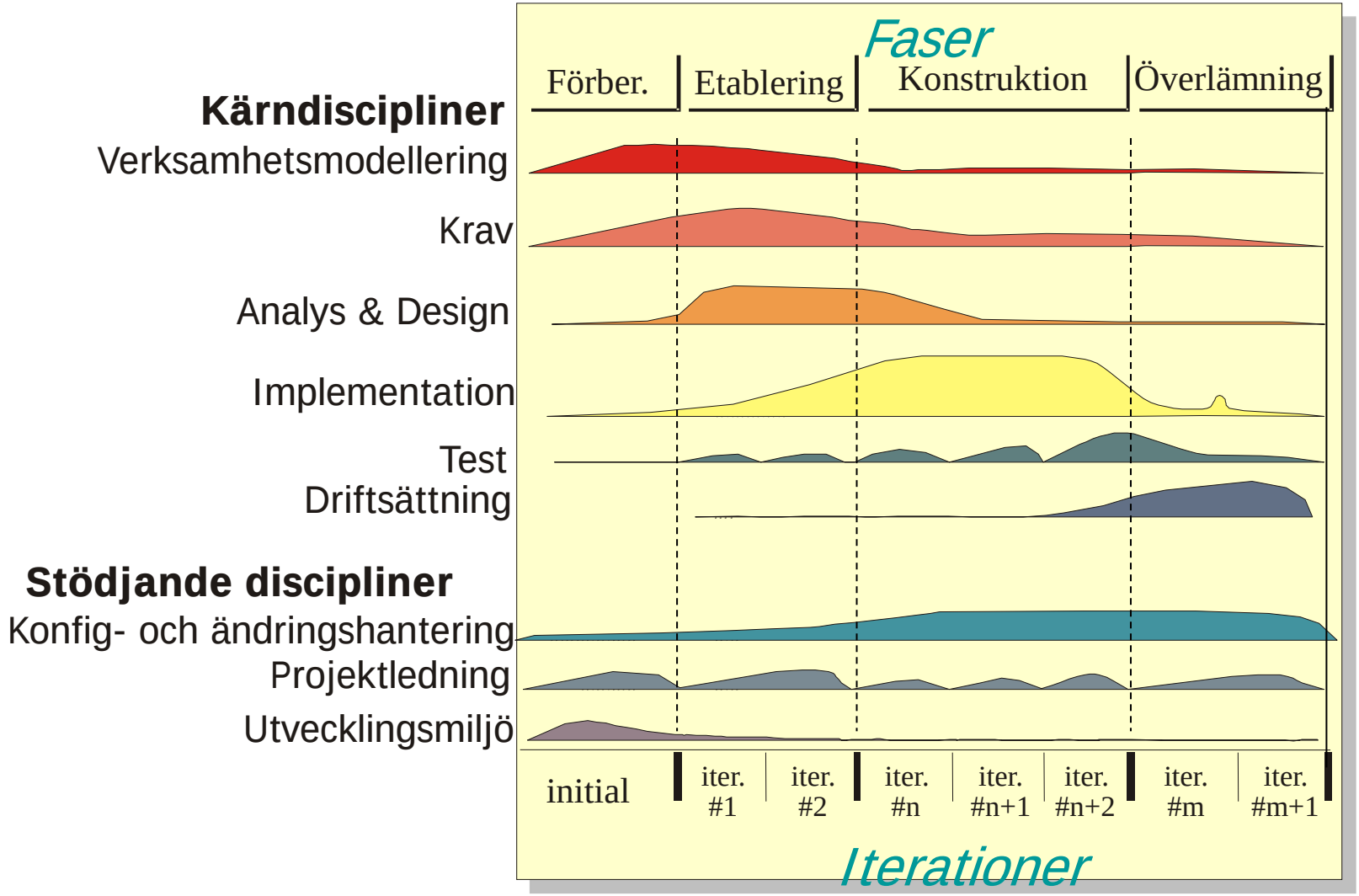
Ingen uppföljning – allt flyter

```
While (System Not Ready) {
        Lite på kraven
        Lite design
        Lite kod och testning
        Integrering
}
Förvaltning
```

Ta grepp om den iterativa processen

- Styr upp *hela processen*
  - Faser med milstolpar/grindar
    - i XP = release
  - Iterationer inom faserna
  - Mål för varje iteration
- Styr upp *varje iteration*
  - I XP = fast längd (”timebox”) + iterationsplanering i början av varje iteration
  - *(Förslag, jobba veckovis i ert projekt)*

Ta grepp om den iterativa processen – exempel RUP



Spektrum mellan feedbackdriven (lättrörliga) och förutsägelsedriven (”plandrivna”)

Hackare

XP
Crystal Clear

Adaptiv
mjukvaru-
utveckling

Milstolpe-
risk-driven
model

Milstolpe-
plan-driven
model

Vattenfall,
Rigida
kontrakt

Feedback-
driven

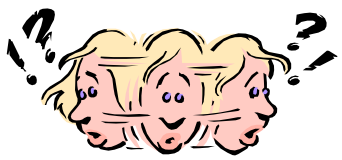
Förutsägelse-
driven

Lättrörliga (agile) metoder

Rational Unified Process (RUP)

Lättrörlig

Kod i fokus



För långt hit
KAOS! < =

Hitta balansen!
(formalitetsnivå)

”Plandriven”

Dokument i fokus

För långt hit
= > Rigiditet

[Boehm 2002, modifierad]

Principer för lättrörlig utveckling

- Viktigast är att *göra kunden nöjd*
- *Välkomna kravförändringar*
- Verksamhetskunniga och utvecklare *arbetar tillsammans*
- Självgående och ansvarstagande *individer*
- Kommunikation *ansikte mot ansikte*
- *Fungerande produkt/system* är det främsta måttet på framsteg.
- Agile verkar för *uthållig utveckling*
- *Teknisk elegans och bra design*
- Enkelhet - konsten att *göra rätt saker*, varken mer eller mindre.
- *Självorganiserande team*
- Gruppen utvärderar och *anpassar regelbundet sitt arbetssätt* för att förbättra sin effektivitet.

Förenklat/omarbetat från: [www.agilesweden.org]

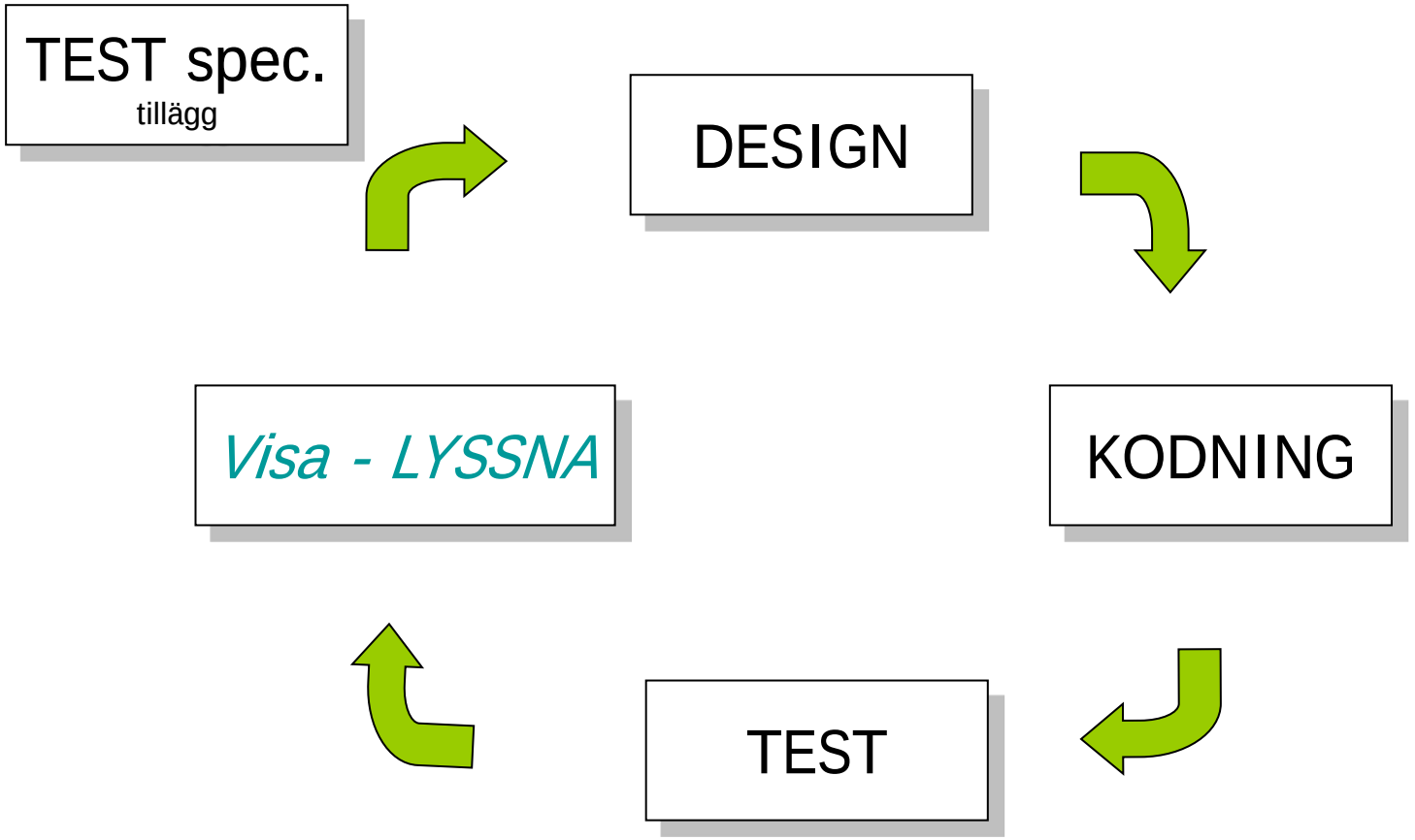
Extreme Programming är

ett sätt att utveckla mjukvara och fokuserar på:

- Excellent användning av *programmeringstekniker*
- Tydlig *kommunikation*
- *Lagarbete* (teamwork)
  - att komma förbi ”Jag vet bättre än alla andra och därför behöver jag bli lämnad ensam för att vara den bästa”

[Beck 2004, kapitel 1]

Extreme Programming (XP)



[Beck 2000]

Tag vanliga ”sunda förnufts-principer” och använd dem *extremt* => namnet.

- *Kodgranskning* är bra...
  - granska ständigt (parprogrammering)!
- *Testning* är bra...
  - testa ständigt (bygg testerna först)
- *Integrationstest* är viktigt...
  - integrera och testa flera gånger varje dag!

[Beck 2000 s xv]

Tag vanliga ”sunda förnufts-principer” och använd dem *extremt* => namnet.

- *Design* är viktig…
  - designa dagligen (refactoring)!
- *Enkelhet* är bra…
  - var nöjd med det enklaste som fungerar!
- *Arkitektur* är viktig…
  - alla jobbar med att förfina arkitekturen ständigt!
- *Små iterationer* är bra…
  - gör iterationerna korta

[Beck 2000 s xv]

Metoder
SCRUM, Kanban – poppis nu

- Lean software development
- RUP (Rational Unified Process)
- Extreme programming
- Crystal Clear
- DSDM
- SCRUM (en term från ryggby)
- MSF Agile (Microsoft Solutuion Framework for Agile Software Projects)
- Kanban
- SEMAT – ny (Software Engineering Method and Theory)
- …

Agilt arbetssätt – några bilder
Jobba ihop

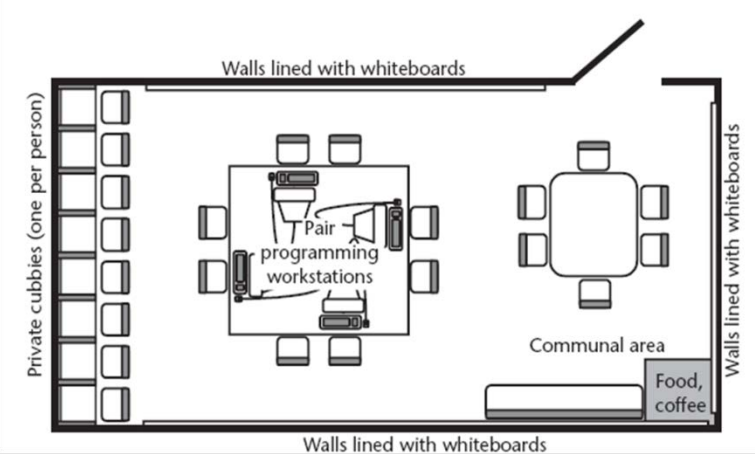
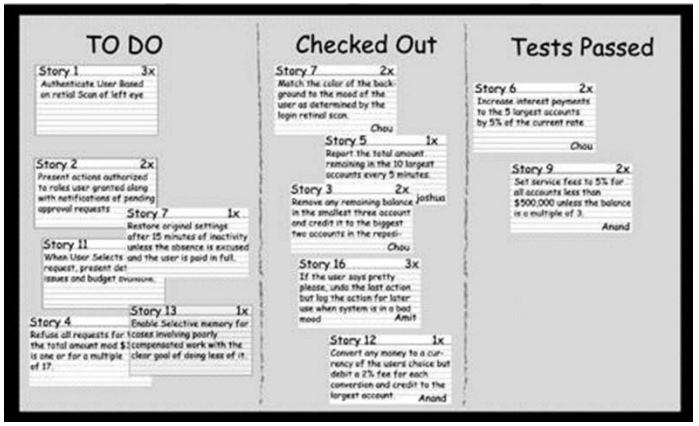
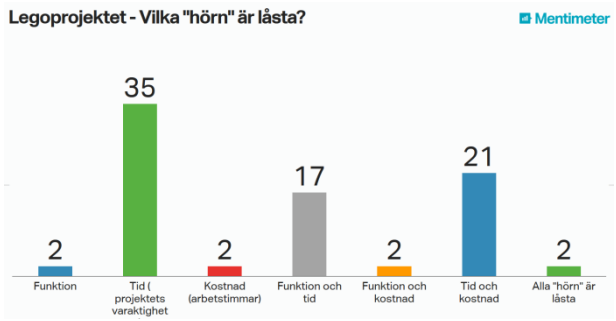


Fig 10-1
Agile Software Development. Evaluating the Methods for Your Organization [Artech House 2005]

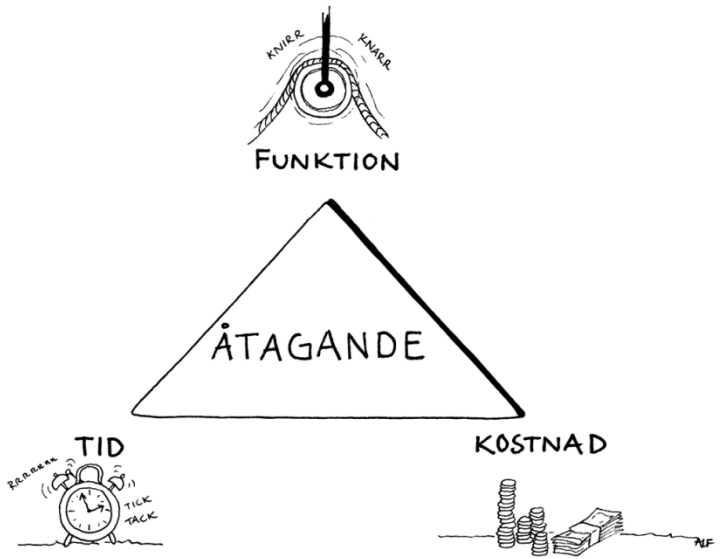


Projektets åtagande – ”lås inte alla hörn”

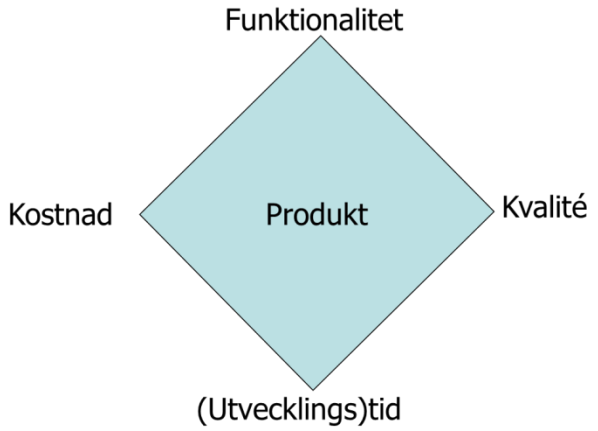
- Om ”Funktion” hålls ”öppen”
  - Krav enl MoSCoW (i TimeBox)
    - Must
    - Should
    - Could
    - Won’t /Wanted



Go to **www.govote.at** and use the code **86 74 09**

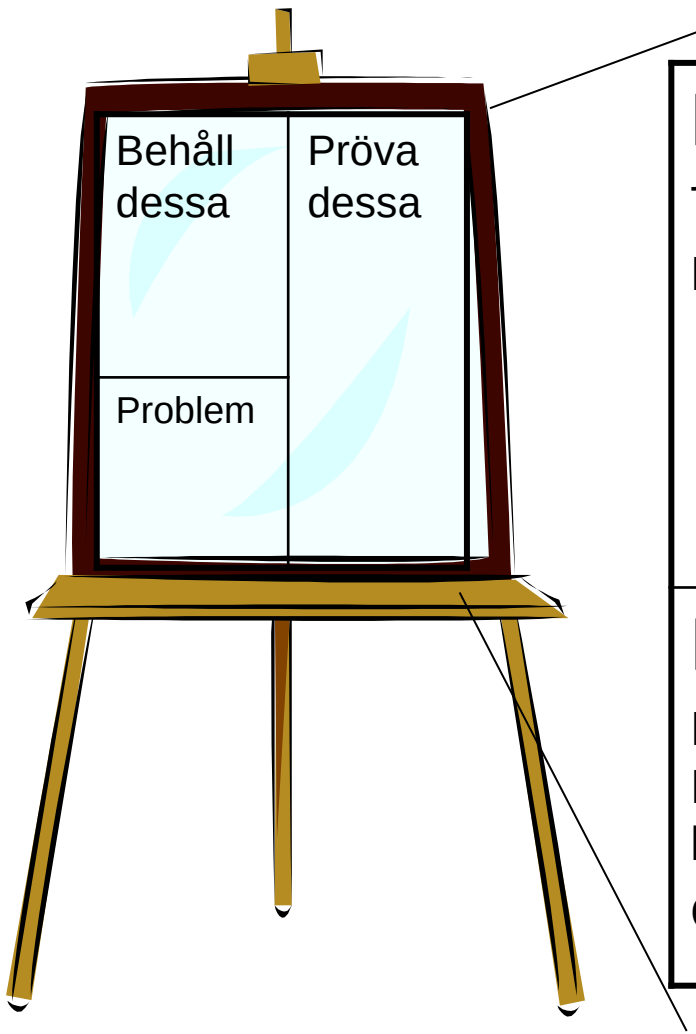


Arbeta i projekt – individen, gruppen, ledaren. © Sven Eklund



[Beck 2000]

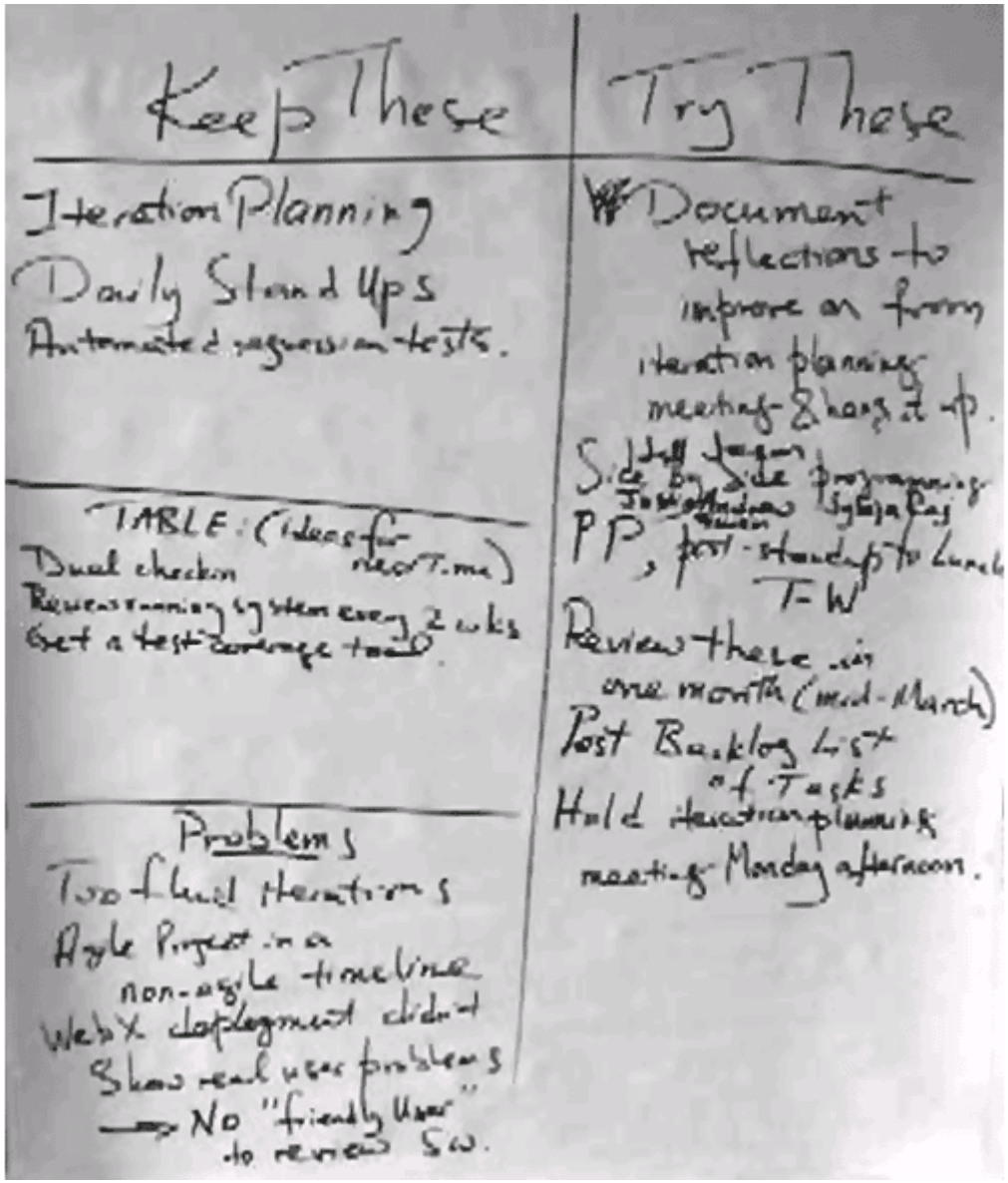
Reflektionsseminarie - arbetssätt



Behåll dessa	**Pröva dessa**
Tyst tid (kl 9-12) Dagliga möten	Partestning ”Böter” för sen ankomst Programmerarna hjälper testarna
Problem	
För många avbrott Leverans av buggig kod Ojämn arbetsfördelning	

[Cockburn 2005, fig 3-6 modif]

Reflektionsseminarie - arbetsätt

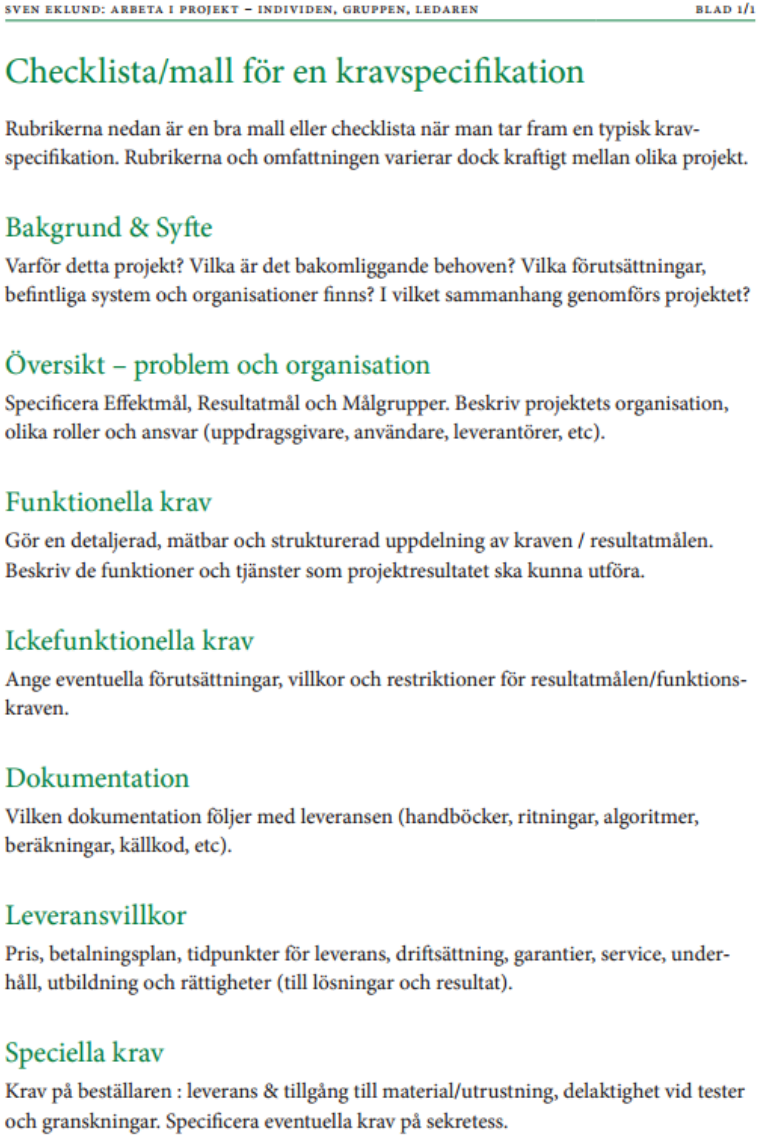


Reflektera 15-60 minuter i hela teamet för att hitta sätt att förbättra arbetssättet. Gör det en gång i veckan, varannan vecka eller en gång i månaden. Behövs mer/oftare i början av ett projekt.

[Cockburn 2005, kapitel 3]

Rekommendation för projektet

- Starta med Kravspecifikation och Projektdefinition
  - Kravspecifikation, se mall
    - Funktionella krav
      - Hitta motståndare
      - Köra svänga backa (hoppa)
      - ”Knuffa” motståndare
      - Inte köra ut från banan
    - Ickefunktionella krav
      - Låg tyngdpunkt – svår att välta
      - Programmering i C
      - Framdrivning – vilka hjul
      - Storlek, vikt,
    - Speciella krav
      - Krav för godkänt i kursen
        - Webbsida
        - Handledningsmöten (förberedelser)
        - Inlämningar, deadlines
        - Mm
      - Tävlingsregler



SVEN EKLUND: ARBETA I PROJEKT – INDIVIDEN, GRUPPEN, LEDAREN BLAD 1/1

Checklista/mall för en kravspecifikation

Rubrikerna nedan är en bra mall eller checklista när man tar fram en typisk kravspecifikation. Rubrikerna och omfattningen varierar dock kraftigt mellan olika projekt.

Bakgrund & Syfte

Varför detta projekt? Vilka är det bakomliggande behoven? Vilka förutsättningar, befintliga system och organisationer finns? I vilket sammanhang genomförs projektet?

Översikt – problem och organisation

Specificera Effektmål, Resultatmål och Målgrupper. Beskriv projektets organisation, olika roller och ansvar (uppdragsgivare, användare, leverantörer, etc).

Funktionella krav

Gör en detaljerad, mätbar och strukturerad uppdelning av kraven / resultatmålen. Beskriv de funktioner och tjänster som projektresultatet ska kunna utföra.

Ickefunktionella krav

Ange eventuella förutsättningar, villkor och restriktioner för resultatmålen/funktionskraven.

Dokumentation

Vilken dokumentation följer med leveransen (handböcker, ritningar, algoritmer, beräkningar, källkod, etc).

Leveransvillkor

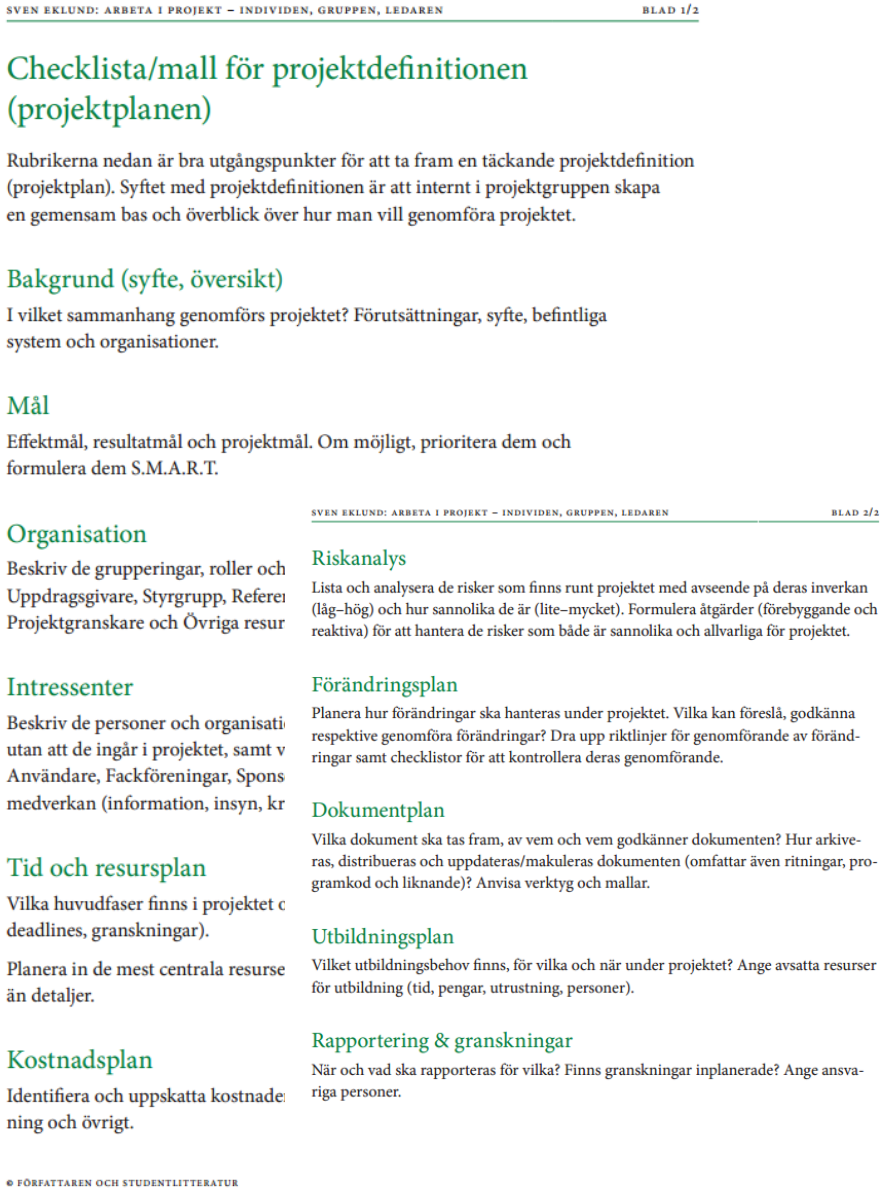
Pris, betalningsplan, tidpunkter för leverans, driftsättning, garantier, service, underhåll, utbildning och rättigheter (till lösningar och resultat).

Speciella krav

Krav på beställaren : leverans & tillgång till material/utrustning, delaktighet vid tester och granskningar. Specificera eventuella krav på sekretess.

Rekommendation för projektet

- Starta med Kravspecifikation och Projektdefinition
  - Projektdefinition, se mall
    - Jobba iterativt,
      - Veckovis
      - Varje iteration ett ”vattenfall”
      - Mål för iterationer
      - Reflektion varje fredag?
      - Delgivning och presentation inom gruppen
    - Jobba inkrementellt
      - Varje iteration har en ”produktdemo”
    - Ansvarsområden
    - Riskhantering
      - Låt risker styra planering, attackera risker
    - Låt Projektdefinitionen bli projektplan, tillfoga
      - WBS
      - PERT-schema
      - Gantt



SVEN EKLUND: ARBETA I PROJEKT – INDIVIDEN, GRUPPEN, LEDAREN BLAD 1/2

Checklista/mall för projektdefinitionen (projektplanen)

Rubrikerna nedan är bra utgångspunkter för att ta fram en täckande projektdefinition (projektplan). Syftet med projektdefinitionen är att internt i projektgruppen skapa en gemensam bas och överblick över hur man vill genomföra projektet.

Bakgrund (syfte, översikt)

I vilket sammanhang genomförs projektet? Förutsättningar, syfte, befintliga system och organisationer.

Mål

Effektmål, resultatmål och projektmål. Om möjligt, prioritera dem och formulera dem S.M.A.R.T.

Organisation

Beskriv de grupperingar, roller och
Uppdragsgivare, Styrgrupp, Referer
Projektgranskare och Övriga resur

Intressenter

Beskriv de personer och organisati
utan att de ingår i projektet, samt v
Användare, Fackföreningar, Spons
medverkan (information, insyn, kr

Tid och resursplan

Vilka huvudfaser finns i projektet c
deadlines, granskningar).

Planera in de mest centrala resurse
än detaljer.

Kostnadsplan

Identifiera och uppskatta kostnade
ning och övrigt.

© FÖRFATTAREN OCH STUDENTLITTERATUR

SVEN EKLUND: ARBETA I PROJEKT – INDIVIDEN, GRUPPEN, LEDAREN BLAD 2/2

Riskanalys

Lista och analysera de risker som finns runt projektet med avseende på deras inverkan (låg–hög) och hur sannolika de är (lite–mycket). Formulera åtgärder (förebyggande och reaktiva) för att hantera de risker som både är sannolika och allvarliga för projektet.

Förändringsplan

Planera hur förändringar ska hanteras under projektet. Vilka kan föreslå, godkänna respektive genomföra förändringar? Dra upp riktlinjer för genomförande av förändringar samt checklistor för att kontrollera deras genomförande.

Dokumentplan

Vilka dokument ska tas fram, av vem och vem godkänner dokumenten? Hur arkiveras, distribueras och uppdateras/makuleras dokumenten (omfattar även ritningar, programkod och liknande)? Anvisa verktyg och mallar.

Utbildningsplan

Vilket utbildningsbehov finns, för vilka och när under projektet? Ange avsatta resurser för utbildning (tid, pengar, utrustning, personer).

Rapportering & granskningar

När och vad ska rapporteras för vilka? Finns granskningar inplanerade? Ange ansvariga personer.

Rekommendation för projektet

(Ange i projektdefinitionen)

- Hitta tider att jobba ihop! Bokför i projektdefinitionen
  - Så många tider som möjligt, sedan kan man ställa in
- Jobba med brainstorming, utnyttja att ni är flera
- Kommunicera med varandra
  - Meddela frånvaro
  - Be om hjälp av varandra
  - Kräv information och förklaring (interna presentationer)
  - Ta ansvar
- Minnesanteckna för att undvika missförstånd
  - ”det var inte vad vi sa”
  - ”skulle jag göra det?”
- Ingen i denna kurs har lyckats med Microsoft Project
  - Använd Excel och Word?
- Om dokument
  - Ofta är processen att ta fram dokument viktigare än dokumentet självt!
  - Välj dokument som ger nytta i någon form!

Slutsatser

- Det finns problem med projekt
- Processer och metoder hjälper till att motverka problemen
- Det gäller att *välja* =>
  - kunskap om processer projektmodeller, metoder…; fördelar, nackdelar och till vad de passar är essentiellt
- Varje projektmetod/ramverk måste *anpassas* till:
  - din egen *organisation*
  - problem*domänen* och
  - ditt *projekt*
- *Beskriv metod i projektdefinitionen*

Frågor ?