Systemutviklingsprosesser

Forelesning 2 - INF1050 Systemutvikling 21.1.2009

Rune Steinberg

International Development Manager ERP

Læringsmål

- Forstå hvorfor systemutviklingsprosessen er viktig
- Forstå de viktigste prinsippene for ulike prosesser
- Få kunnskap om ulike utviklingsprosesser
- Forstå sentrale suksessfaktorer

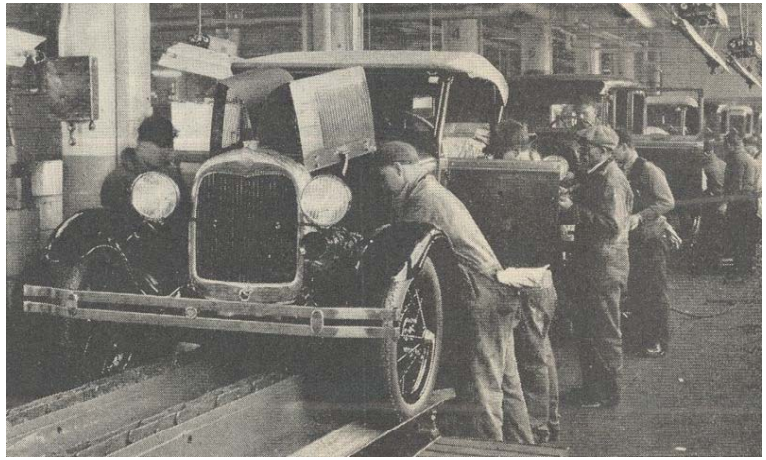
Hva er en systemutviklingsprosess?

- Beskriver hvordan programvare skal produseres ved å angi mekanismer for å styre, kontrollere og organisere arbeidet.
- Kunnskap om systemutviklingsprosessen er sentral i arbeidet med å forbedre produktivitet og kvalitet i systemutvikling

Merk at systemutviklingsprosess, utviklingsprosess og utviklingsprosessmodell betyr det samme

Eksempel fra Ford Model T

- 1908: Stasjonsvis produksjon
- 1913: Samlebånd introdusert
  - Arbeidstid redusert fra 12,5 timer til 1,3 timer
- 1918: Halvparten av alle biler i USA er en Ford
- 1930: Alle bilfabrikker har gått over til samlebånd



Hvorfor er utviklingsprosesser viktig?

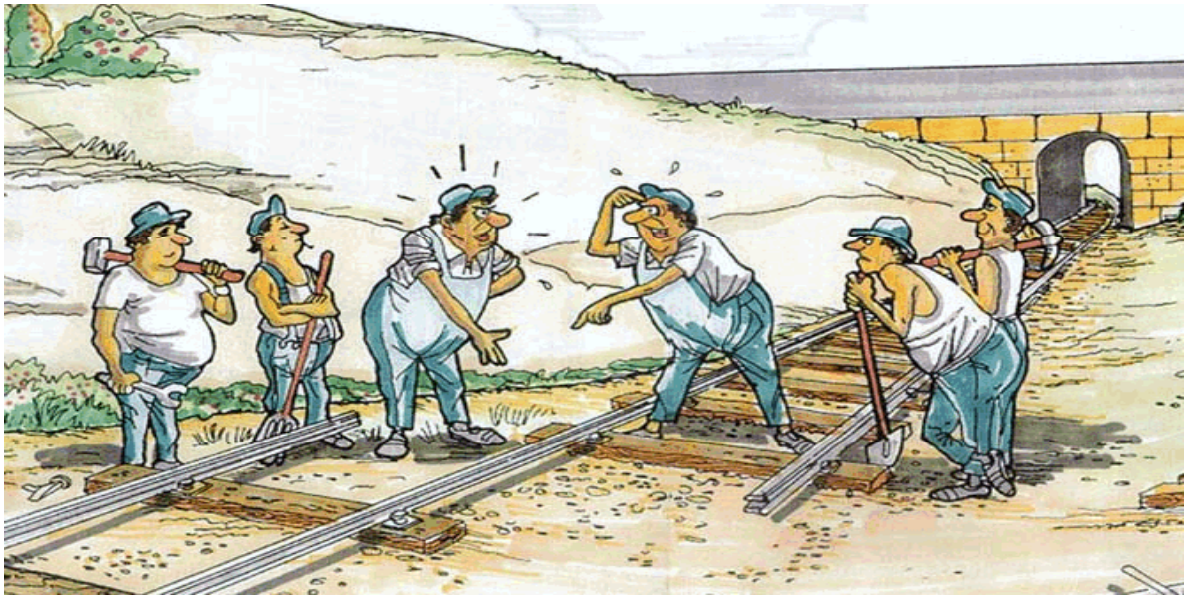
- Hvordan arbeidet utføres har stor påvirkning på produktivitet og kvalitet
- Noen prosesser er bedre egnet enn andre
- Prosesser innfører begreper og felles begrepsforståelse
- Felles begrepsforståelse er nødvendig for godt samarbeid
- Felles begrepsforståelse er nødvendig for standardisering
- Standardisering er nødvendig for forbedring av prosessen

Feil prosess kan ha fatale følger for virksomheten!

Merk at standardisering her betyr innenfor en virksomhet. Det er ikke det samme som en offisiell standard som f. eks. ISO 9000

Systemutvikling, en utfordrende aktivitet?

- TRESS-90: Administrativt system for trygdeetaten
  - Planlagt levert 1993, nedlegges i 1995
  - Opprinnelig budsjett: 383 Mill. Totalt tap over 1 MRD.
- Oslo Sporveier m. fl: Nytt elektronisk billettsystem
  - Planlagt prøvedrift september 2005
  - Fortsatt ikke i drift i 2008



Hvor godt lykkes vi?

Undersøkelse fra norske virksomheter utført av Simula i 2003:

- 76% av prosjektene overskrider budsjettet
- 19% bruker mindre enn budsjettert
- Gjennomsnittlig overskridelse er 41%
- Utviklingsprosessen påvirker utfallet
  - 55% overskridelse ved bruk av fossefallsmodellen
  - 24% overskridelse ved iterative/inkrementelle/evolusjonære metoder

Konklusjon

- Programvare har blitt en nødvendig del av vårt samfunn
- Vi har store utfordringer med å levere med tilfredstillende kvalitet.
- Sannsynligheten for å levere et prosjekt i henhold til tidsplan og budsjett er lav.
- Mange prosjekter feiler fullstendig

Utviklingsprosess og livssyklus

I systemutvikling opererer vi med begrepene utviklingsprosess og livssyklus:

- En livssyklus beskriver hovedaktivitetene fra oppstarten av et prosjekt, til utvikling, drift, og nedleggelse
- En utviklingsprosess beskriver fasene fra oppstart, til utvikling og leveranse

Merk at litteraturen i systemutvikling ikke er entydig på forskjellen mellom livssyklus og utviklingsprosess.

Ulike faser i en livssyklus

1) Kravinnsamling og kravanalyse (hva skal systemet gjøre?)
2) Design (hvordan skal det konstrueres?)
3) Programmering (konstruksjon)
4) Test (ble det riktig?)
5) Installasjon, integrasjon, driftsetting
6) Vedlikehold (feilretting og videreutvikling)

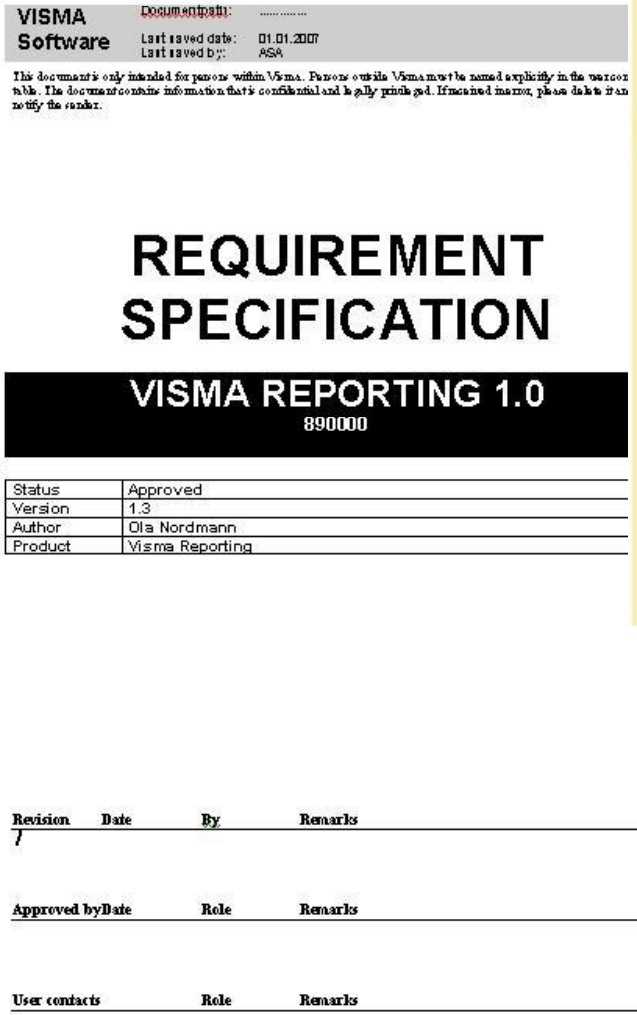
Merk at det finnes flere ulike varianter av faseinndelinger i en livssyklus. Se f. eks. forskjellen mellom GS.

Kravinnsamling og kravanalysefasen

Identifiserer kravene til hva vi ønsker å oppnå med systemet og hvilke begrensinger vi må ta hensyn til. Det innebærer å definere:

- Overordnet målsetting og begrensing
- Funksjonelle krav (hva skal systemet gjøre for brukeren)
- Ikke-funksjonelle (tekniske krav som f. eks. svartider)
- Kravene identifiseres og besluttes av utvalgte *interessenter* (sluttbrukere, driftpersonell, etc.)
- Resultatet er gjerne et dokument som beskriver resultatet av analysen (kravspesifikasjon)

Eksempler på kravdokumenter



VISMA Software

Documentpath: .....
Last saved date: 01.01.2007
Last saved by: ASA

This document is only intended for persons within Visma. Persons outside Visma must be named explicitly in the version table. The document contains information that is confidential and legally privileged. If received in error, please delete it and notify the sender.

REQUIREMENT SPECIFICATION

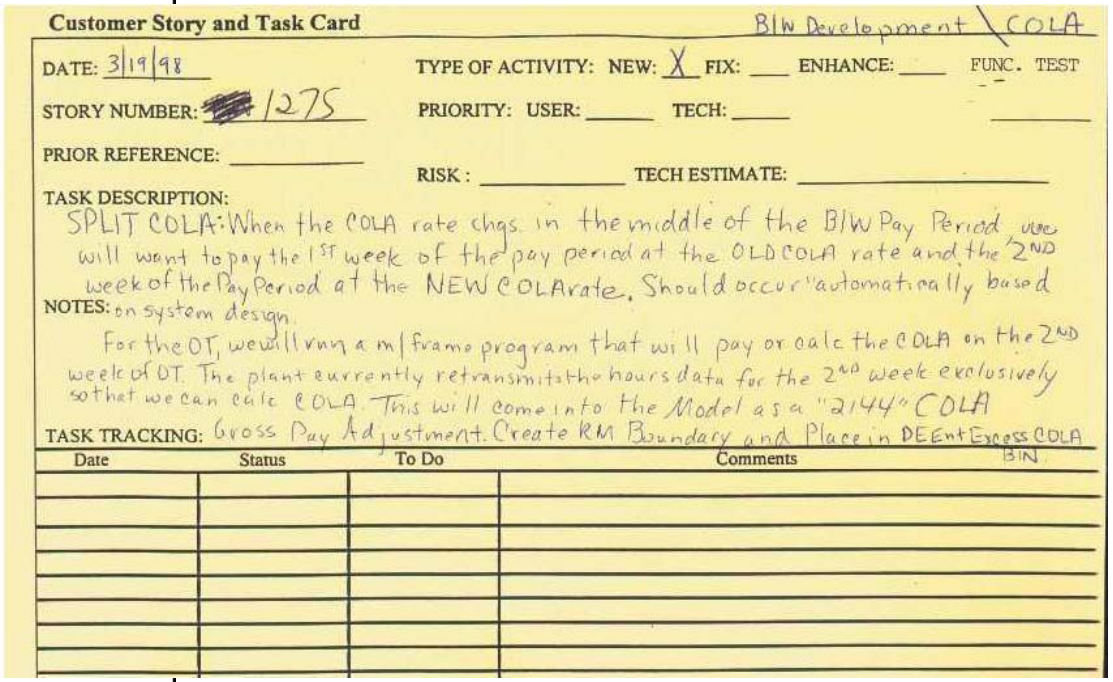
VISMA REPORTING 1.0
890000

Status	Approved
Version	1.3
Author	Ola Nordmann
Product	Visma Reporting

Revision	Date	By	Remarks
1			

Approved by	Date	Role	Remarks

User contacts	Role	Remarks



Customer Story and Task Card B/W Development \ COLA

DATE: 3/19/98 TYPE OF ACTIVITY: NEW: X FIX: ___ ENHANCE: ___ FUNC. TEST

STORY NUMBER: [redacted]1275 PRIORITY: USER: _____ TECH: ____

PRIOR REFERENCE: _____ RISK : _____ TECH ESTIMATE: _____

TASK DESCRIPTION:
SPLIT COLA: When the COLA rate chgs. in the middle of the B/W Pay Period we will want to pay the 1ST week of the pay period at the OLD COLA rate and the 2ND week of the Pay Period at the NEW COLA rate. Should occur automatically based on system design.

NOTES:
For the OT, we will run a m/frame program that will pay or calc the COLA on the 2ND week of OT. The plant currently retransmits the hours data for the 2ND week exclusively so that we can calc COLA. This will come into the Model as a "2144" COLA

TASK TRACKING: Gross Pay Adjustment. Create RM Boundary and Place in DEEntExcessCOLA BIN

Date	Status	To Do	Comments

Designfasen

Gitt kravene må systemet designes. Det innebærer å:

- Delvis design av bruker interaksjon og løsningskonsept
- Design av arkitektur
  - Identifisere hovedkomponenter i systemet
  - Hvilket ansvar hver komponent har
  - Relasjonen mellom komponentene
  - Design beskrives gjerne i egne diagrammer (UML)
- Design gjøres på flere nivåer, overordnet og detaljert
- Resultat beskrives gjerne i UML modeller og system spesifikasjon

Eksempler på resultater fra design

VISMA Software | Specification | Document: | Version: 1.0
Folder: | 12.09.2007
Author: Per Hansen

SYSTEM SPECIFICATION

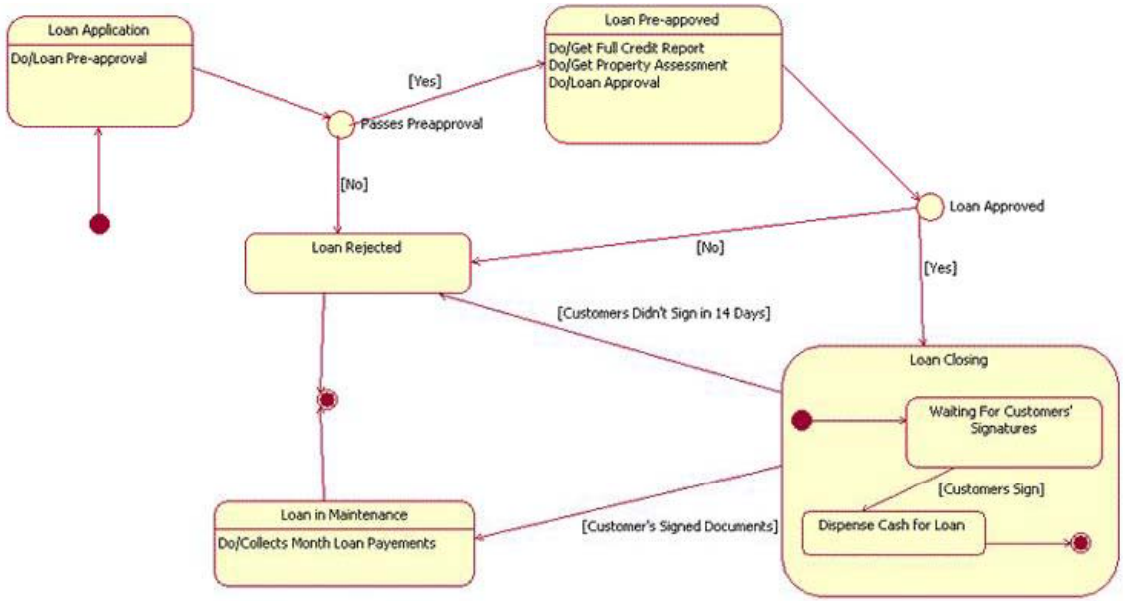
Created by:

Ola Nordmann

Case no: 968989

Case Title: Visma Reporting 1.0

Date: 20070506



Programmeringsfasen

```
as_cr_LETTERS='ABCDEFGHIJKLMNOPQRSTUVWXYZ'
as_cr_Letters=$as_cr_letters$as_cr_LETTERS
as_cr_digits='0123456789'
as_cr_alnum=$as_cr_Letters$as_cr_digits

# Sed expression to map a string onto a valid variable name.
as_tr_sh="eval sed \"y%*+%pp%;s%[^_$as_cr_alnum]%_%g\""
# Sed expression to map a string onto a valid CPP name.
as_tr_cpp="eval sed \"y%*+%pp%;s%[^_$as_cr_alnum]%_%g\""
# Be Bourne compatible
if test -n "${ZSH_VERSION+set}" && (emulate sh) >/dev/null 2>&1; then
  emulate sh
  NULLCMD=:
elif test -n "${BASH_VERSION+set}" && (set -o posix) >/dev/null 2>&1; then
  set -o posix
fi

# Name of the executable.
as_me=`echo "$0" |sed 's,.*[\\/],,'`

if expr a : '\(a\)' >/dev/null 2>&1; then
  as_expr=expr
else
  as_expr=false
fi

rm -f conf$$ conf$$.exe conf$$.file
echo >conf$$.file
if ln -s conf$$.file conf$$ 2>/dev/null; then
  # We could just check for DJGPP; but this test a) works b) is more generic
  # and c) will remain valid once DJGPP supports symlinks (DJGPP 2.04).
  if test -f conf$$.exe; then
```

- Her skjer den endelige *konstruksjonen*
- Inkluderer gjerne ytterligere detaljert design
- Grafisk design

Programmeringsfasen

```
as_cr_LETTERS='ABCDEFGHIJKLMNOPQRSTUVWXYZ'
as_cr_Letters=$as_cr_letters$as_cr_LETTERS
as_cr_digits='0123456789'
as_cr_alnum=$as_cr_Letters$as_cr_digits

# Sed expression to map a string onto a valid variable name.
# Sed expression to map a string onto a valid CPP name.
# Be Bourne compatible
  emulate sh
  NULLCMD=:
elif test -n "${BASH_VERSION+set}" && (set -o posix) >/dev/null 2>&1; then
  set -o posix
fi

# Name of the executable.
as_me=`echo "$0" |sed 's,.*[\\/],,'`

if expr a : '\(a\)' >/dev/null 2>&1; then
  as_expr=expr
else
  as_expr=false
fi

rm -f conf$$ conf$$.exe conf$$.file
echo >conf$$.file
if ln -s conf$$.file conf$$ 2>/dev/null; then
  # We could just check for DJGPP; but this test a) works b) is more generic
  # and c) will remain valid once DJGPP supports symlinks (DJGPP 2.04).
  if test -f conf$$.exe; then
```

Windows Vista består av mer enn 50 Mill. slike kodelinjer. Det gir 1-2 Mill. A4 sider som rager nærmere 100 meter over bakken dersom vi la alle arkene oppå hverandre.

Testfasen

Overordnet er målsettingen er å besvare følgende:

1. Har vi laget riktig system (funksjonelle krav)?
2. Er systemet riktig bygget (tekniske krav)?

Testfasen

Vi må finne flest mulig feil tidligst mulig:

- Er forventninger og krav er riktig?
- Oppfyller systemet kravene?
- Er det lett å lære og å bruke?
- Er det robust under feil bruk?
- etc....

Testfasen gir oss informasjon om kvalitet og risiko, men test kan aldri vise fravær av feil

Hva er en utviklingsprosess?

En utviklingsprosess beskriver en prinsipiell fremgangsmåte for å utvikle et IT system. Prosessen innholder normalt:

- Ulike faser
- Prosessflyt (rekkefølge på faser og aktiviteter)
- Metoder
- Organisasjon

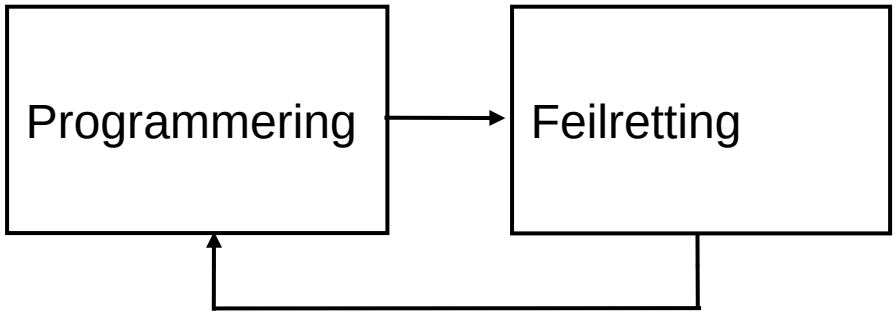
Husk utviklingsprosess og utviklingsprosessmodell betyr det samme. Utviklingsmodell benyttes gjerne som et mer konkret begrep

4 klasser av utviklingsprosesser

- Prøv-og-feil
- Fossefallsmodellen
- Prototyping
- Evolusjonær, iterative, og inkrementelle modell

Merk at disse 4 klassene beskriver prinsippene eller mønstre. Spesifikke og navngitte modeller blir instanser av en av disse.

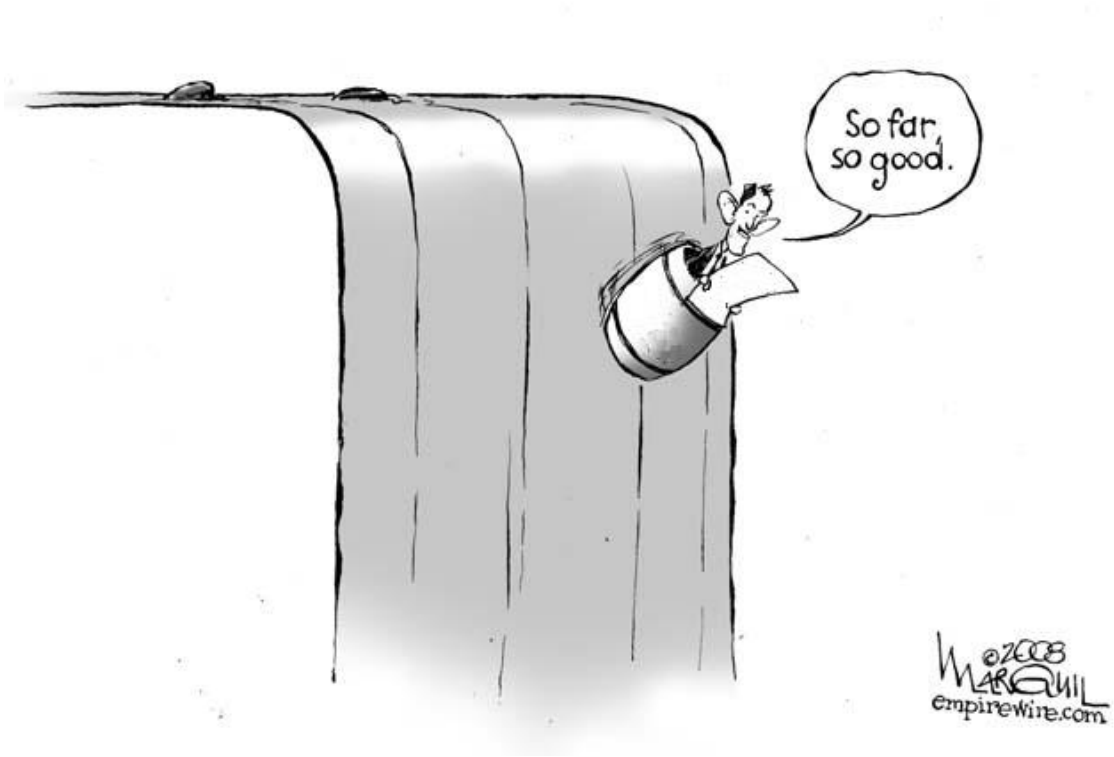
Prøv-og-feil



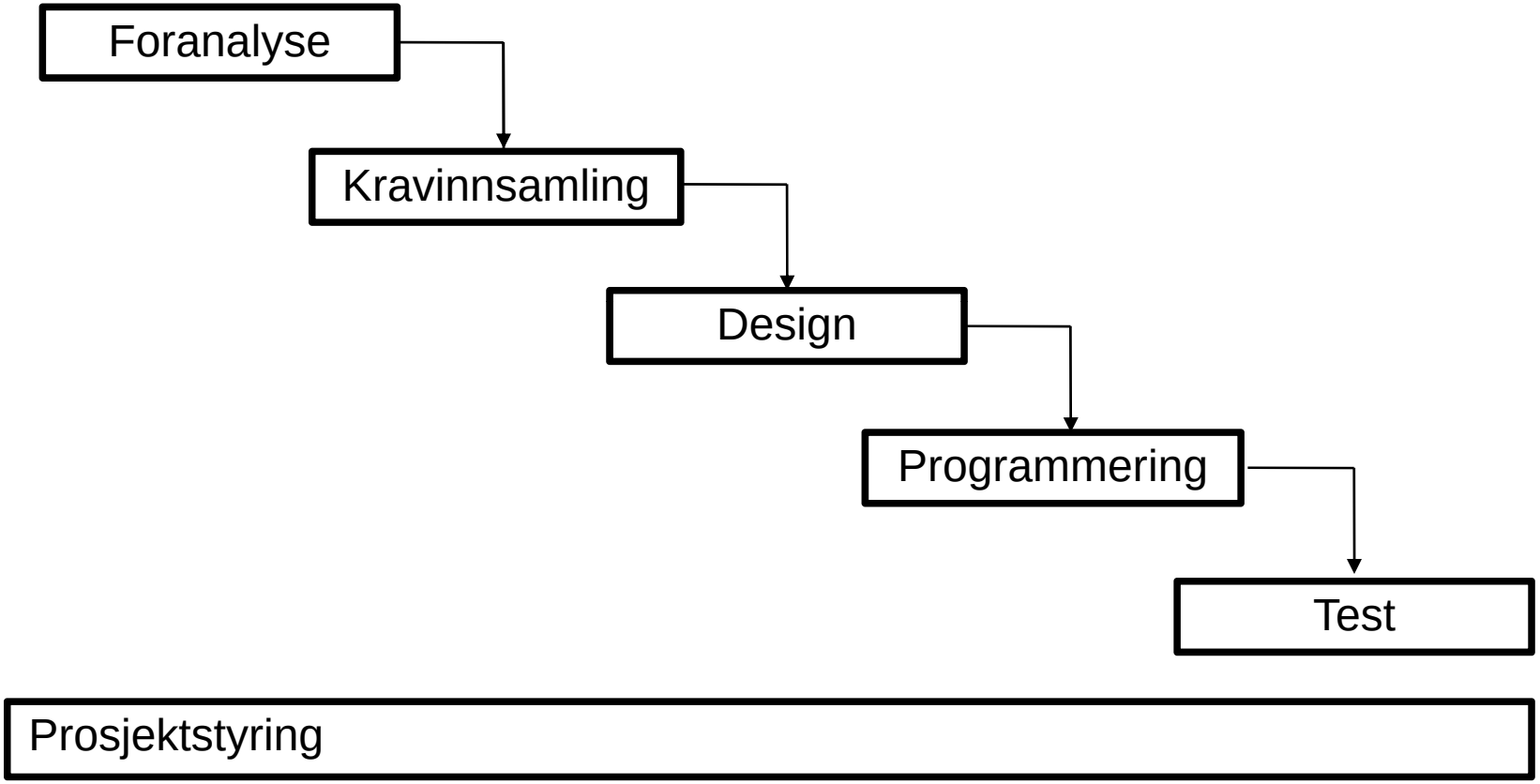
- Ingen planlegging
- Ingen kravanalyse eller designfase
- Ad-Hoc testing

Høy risiko for å feile

Fossefallsmodellen

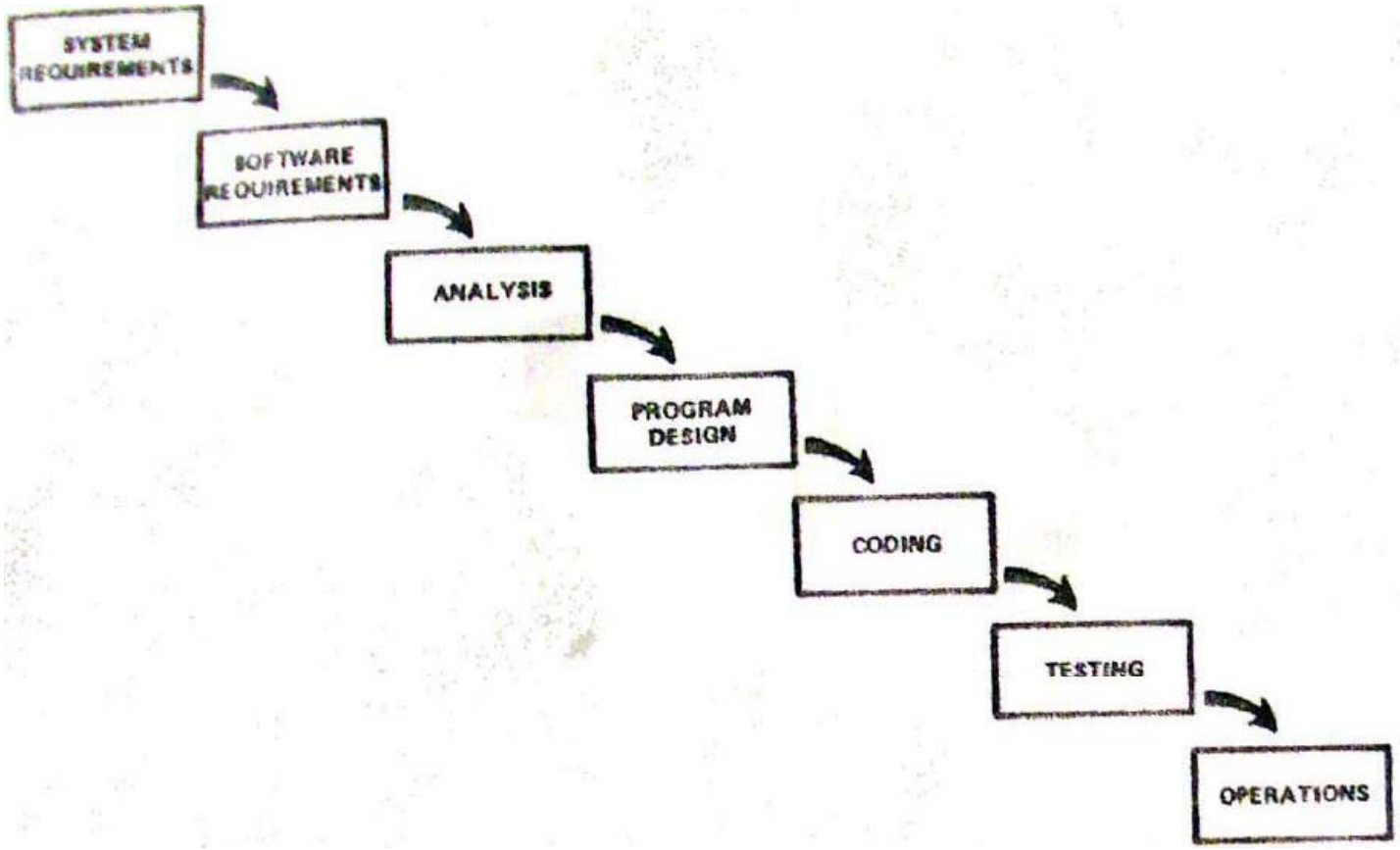


Fossefallsmodellen



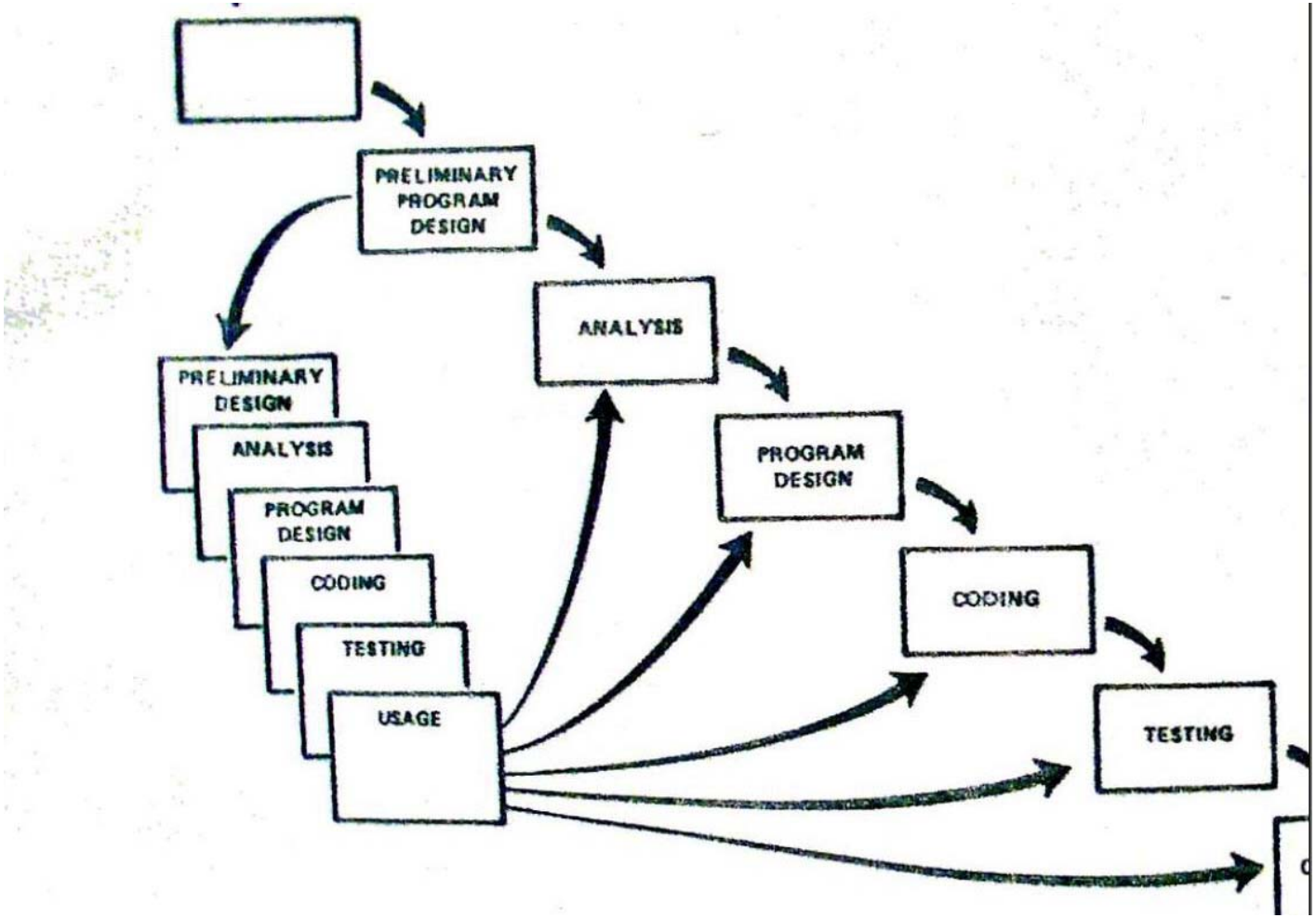
Varianter av fossefallsmodellen

Den første utgaven fra 1970



Varianter av fossefallsmodellen

Den første utgaven fra 1970



Hovedprinsippet i fossefallsmodellen

- Utvikling er en forutsigbar produksjonsprosess
- En pålitelig og detaljert plan kan etableres ved oppstart
- Kravene kvalitetssikres ved at de dokumenteres og gjennomgås før programmeringen starter
- Hver fase avsluttes før neste fase kan begynne
- Endringer i planen skal normalt ikke skje
- Programvaren antas å bli korrekt utviklet i første forsøk
- Systemet kan ikke utprøves før det er helt ferdig
- En repitisjon av prosessen vil levere samme resultat

Fordeler med fossefallsmodellen

- En av de første forsøk på å standardisere systemutvikling (DoD Military Standard 2167)
- Påtvinger disiplin med tydelig start og stopp i hver fase
- Konseptuelt enkel, enkel å forstå og kontrollere for ledere, enkel å undervise
- Alle krav og design gjøres før programmering. Sparer mye kostnader hvis feil oppdages på dette stadiet

(DoD = Department of Defense, USA)

Problemer med fossefallsmodellen

Tre viktige observasjoner

1. Er det mulig å forstå hvordan et IT-system vil fungere ved å lese fra hundre til flere tusen sider med dokumenter?
2. Er det først når vi sitter foran en datamaskin og prøver et system at vi oppdager feilene?
3. Hvordan kan vi planlegge en testfase med en gitt slutt dato uten at vi vet noe om feilraten i systemet?

Problemer med fossefallsmodellen

- Adekvate krav kan ofte ikke defineres på forhånd
  - Brukerne er ikke alltid sikre på hva de behøver
    - «Jeg vet hva jeg behøver når jeg ser det»
  - Endringer i eksterne forutsetninger er ikke forutsigbare
- Støtter ikke endring av krav underveis
  - Brukerne endrer oppfatning underveis i prosessen
  - Støtter ikke tilpassning til endrede eksterne forutsetninger
- Evaluering og test utføres til slutt
  - Feil og mangler oppdages for sent (dette gir høye kostnader)

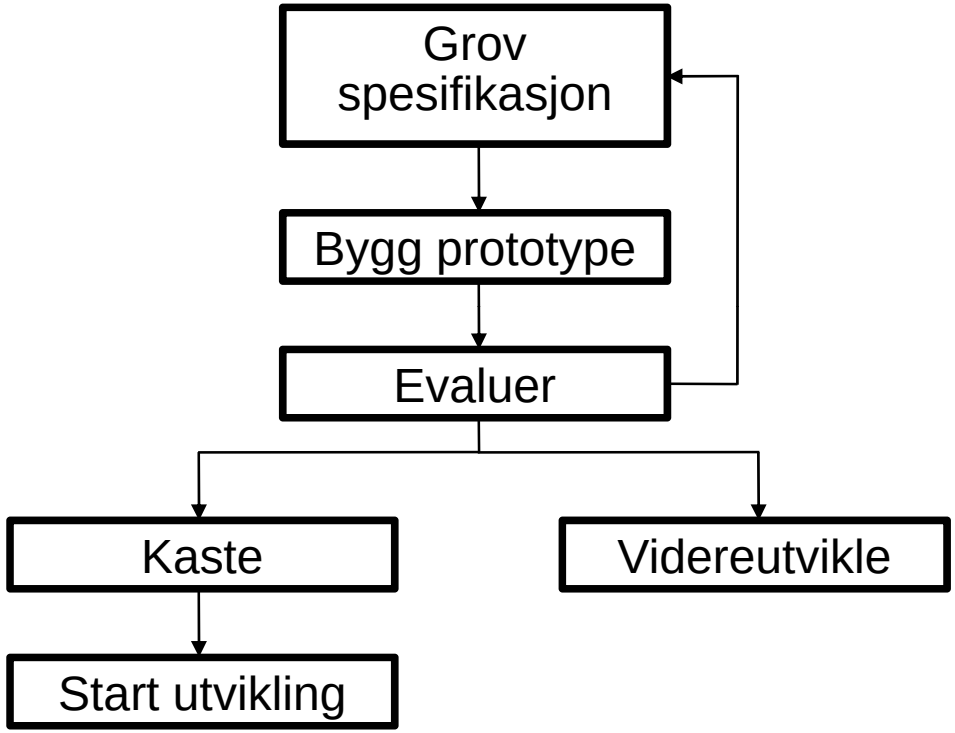
Vi har behov for bedre modeller som støtter evaluering og endring mye tidligere i utviklingen

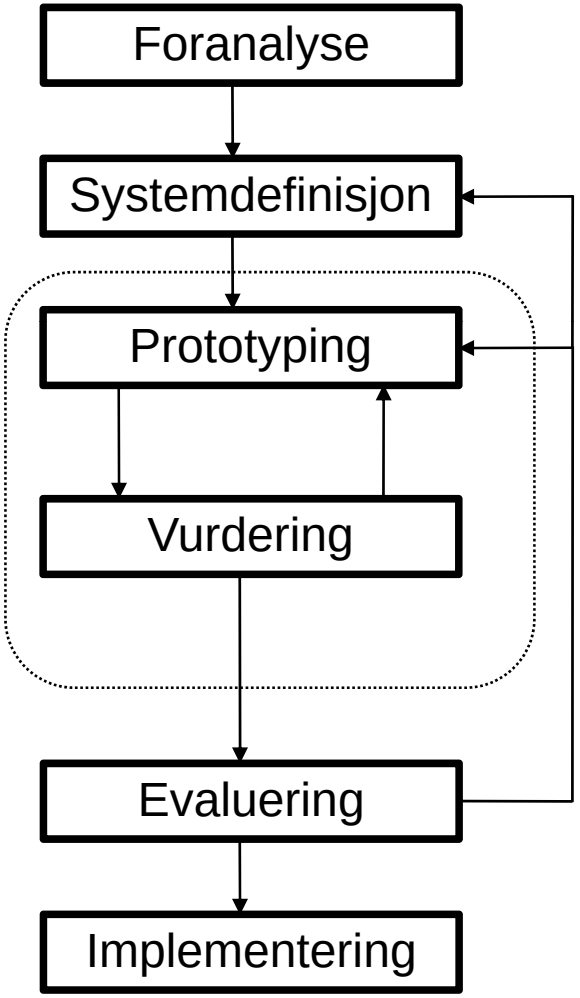
Prototyping

En prototype er en initiell versjon hvis formål er å demonstrere konsepter, utforske designvalg, og evaluere forståelsen av identifiserte krav. Formålet er å sikre at det riktig systemet utvikles.

- Introdusert for å avhjelpe problemene med fossefallsmodellen
- En mer strukturert utgave av prøv-og-feil
- Tilbyr flere varianter:
  - Bruk-og-kast prototyping
  - Evolusjonær prototyping

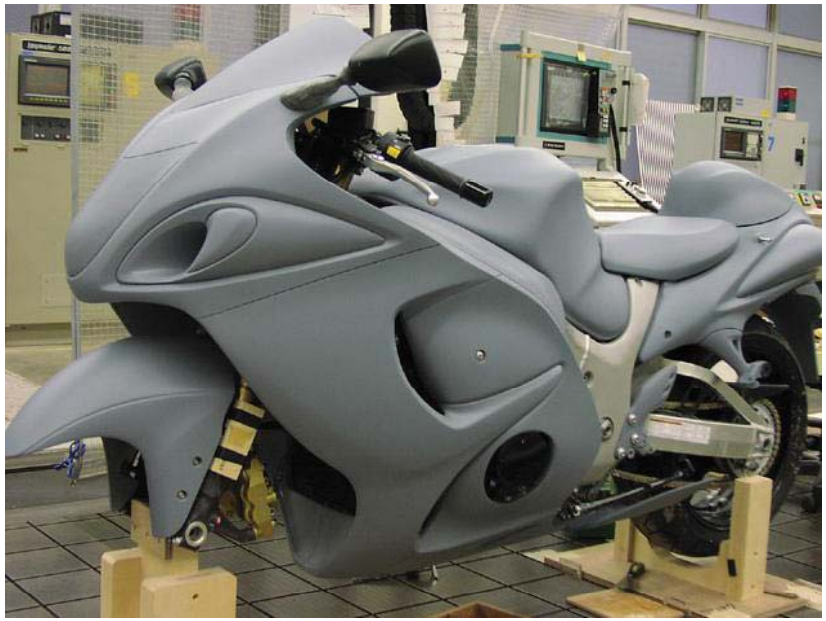
Prototyping





Fordeler med prototyping

- Gir en visuell og tidlig presentasjon av et tenkt slutt resultat
  - Husk: «Jeg vet hva jeg behøver når jeg ser det»
- Forbedrer forståelsen av behov og løsningskonsept



Ulemper med prototyping

- Krav til hurtighet fører til kompromisser på kvalitet
- Lite fokus på arkitektur og andre tekniske kvaliteter
- Lite fokus på vedlikehold
- Interessenter betrakter en kjørende prototype som ferdig system

Prototyping benyttes idag mer som en teknikk for å studere krav og løsningsforslag enn som en fullstendig utviklingsmodell

Fra fossefall til evolusjon

- Fossefallsmodellen har flere viktige ulemper. Prototyping og evolusjonære modeller har oppstått som følge av denne erfaringen
- Fossefallsmodellen baseres på antagelsen om at systemutvikling er en forutsigbar og repeterbar produksjonsprosess (er det riktig?)
- Prototyping og den evolusjonære modellen antar at systemutvikling ikke er forutsigbar eller repeterbar

Evolusjonære modeller

Del prosjektet opp i mindre selvstendige mini-prosjekter som kalles *iterasjoner*. Hver iterasjon må levere en fungerende del av systemet. Dette kalles et *inkrement*. Formålet er å kontrollere at:

- Prosjektgruppen forstår interessentenes behov
- Interessentene bekrefter at prosjektgruppen forstår interessentenes behov
- Teknologi, verktøy, og metoder virker som forventet

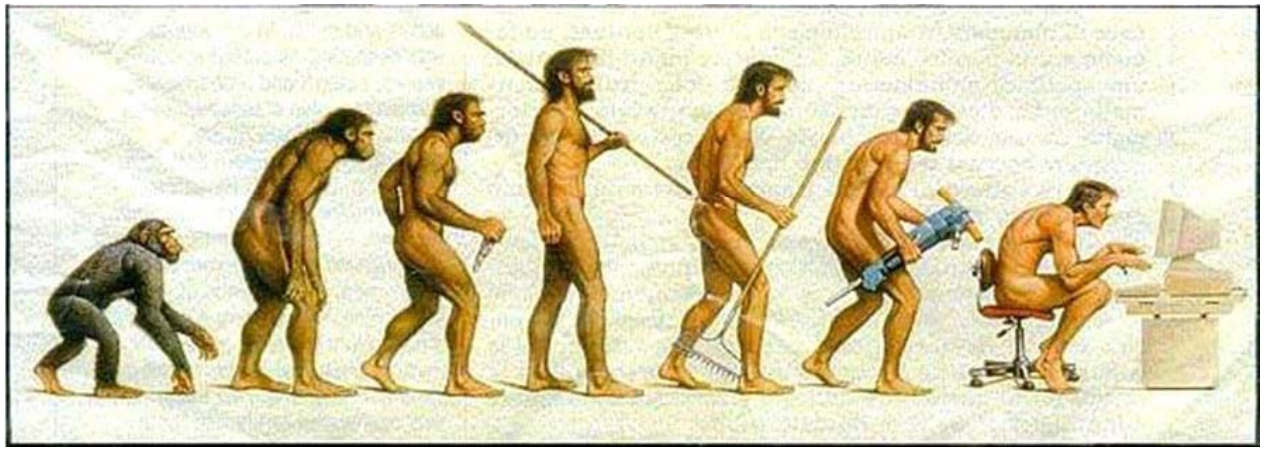
Mange små iterasjoner med leveranser og evaluering leder prosjektet i riktig retning. Dersom noe er galt oppdages dette tidlig

Definisjon på iterativ og inkrementell utvikling

- En *iterativ* utviklingsprosess er en utviklingsprosess som består av flere mindre sekvensielt ordnede mini-prosjekter som kalles iterasjoner
- Hver iterasjon er et selvstendig mini-prosjekt som består av kravinnsamling, design, implementering, og test
- Målsettingen med hver iterasjon er å levere en fungerende, stabil, integrert del av det totale systemet

En begrepsavklaring

- Evolusjonær utvikling introduserer prinsippet om adaptiv endring i form av evaluering og justering av planen
- Iterativ og inkrementell utvikling viser hvordan



"It is not the strongest species that survive, nor the most intelligent, but the most responsive to change"
[Sir Charles Darwin, 1871]

Endring


THE PROJECT REQUIREMENTS ARE FORMING IN MY MIND.
NOW THEY'RE CHANGING... CHANGING... CHANGING... OKAY. NO, WAIT... CHANGING... CHANGING... DONE.
NATURALLY, I WON'T BE SHARING ANY OF THESE THOUGHTS WITH ENGINEERING.
I BUDGETED FOR SOME GOONS TO BEAT IT OUT OF YOU.
DILBERT © United Feature Syndicate, Inc.

Prinsipper

- Ingen fullstendig kravspesifikasjon skrives ved oppstart
- Regelmessige leveranser (inkrementer) til interessentene
- Utviklingen foregår stegvis med nye inkrementer
- Eksplisitte endring og bearbeiding av tidligere resultater er innebygget i modellen
- Regelmessig endring av planene basert på evalueringer utført av interessentene