

Indholdsfortegnelse

1. PROJEKTETABLERING.....	2
2. FORUNDERSØGELSE	6
3. AGILE METODER.....	13
4. SCRUM	17
5. XP.....	21
6. METODEVALG	26
7. PROJEKTSTRATEGI.....	32
8. TEST I SYSTEMUDVIKLING.....	36
9. ESTIMERING	41
10. ESTIMERING.....	43
11. PRODUKT OG PROCES I SYSTEMUDVIKLING	51
12. FORUNDERSØGELSE.....	54

1. Projektetablering

Gør rede for formålet med projektetablering samt hvilke konkrete aktiviteter, der kan foregå i en projektetablering.

Indledning

- Det er vigtigt at lave projektetablering i forbindelse med et projekt, det gør nemlig alle får den samme forståelse for projektet samt at der socialt etableres et forhold mellem gruppemedlemmerne.
- Formålet med en projektetablering er at projektgruppen skal have forståelse for, hvorfor man laver projektet.
- Giver et overordnet overblik over projektet.
- Samt fordi der skal skabe tillid mellem gruppensmedlemmer.
- Fordele roller, man kunne f.eks. tage forskellige tests, som trap test, for at se hvordan gruppen passer sammen, og derved kan man fordele roller. F.eks. Belbin team roller.
- Projektetablering, handler i sin helhed, om at kende til projektet, og projektetsmedlemmer samt skabe et bånd herimellem.
- Resultatet af projektetableringen er projektgrundlaget.
- Grundlaget er projektets rammer, organisation og arbejdsformer, som skal være klare for både ledelsen og projektdeltagere.

Aktiviteter under projektetablering

HUSK: Snak lidt om hvordan I benyttede jer, af disse aktiviteter under projektet.

Interessentanalyse

- Dem der påvirker projektet, og deltager i det
- Man skal først identificer sine interessenter
- Man kigge også på hver af interessenternes succeskriterier. Succeskriterier er f.eks. brugervenlighed, lav pris, lære noget nyt osv.
- Man skal se om de modstrider hinanden
- Man skal så tænke på hvordan de kan imødegås

Hvilke kommunikationsveje skal I bruge?

- Google
- Skype
- Skal der være en kontrakt?

Projektets organisation

- Hvilke roller?
- Evt. en kontrakt, for at sikre nogle regler
- Brug resultater fra TRAP
- Hvem er styregruppen og referencegruppen?
 - Styregruppe er nogle af de øverste befalende f.eks. en afdelingsleder. **Skal støtte projektet hvis det er i problemer, måske evt. en sponsor**
 - Undergrupper er **selvstyrende undergrupper** af projektet. F.eks. man kunne dele

arbejdsopgaver op, og fordele det ud til undergrupper

- Referencegruppe, dem som holdes orienteret om forløbet og kommer med **gode råd og ideer.**

Også kaldet følgegruppe.

- Valg af arbejdsformer(hvilke værktøjer vil i bruge, hvilken udviklingsmodel, hvilke standarder(f.eks. kodenormer), hvilke test og reviews skal udføres) samt infrastrukturen(Hvad skal der være i jeres arbejdssted? F.eks. tavler og hvordan rummet skal være)

- Nævn evt. organisationsmodel, som man kan opstille projektet efter.
- Organisationsmodeller(Trimvirat-modellen(Flere ledere), netværksmodellen(En for alle og alle for en), MSF-modellen(Seks ligeværdige roller), undergruppemodellen(Projektet deles op i undergrupper), diktatormodellen(Projektlederen tager alle beslutninger), chefprogrammør teamet(Chefprogrammøren bestemmer). Projektgrupperne kan få en følgegruppe som er en rådgivende gruppe(influenter))

TURBO-analyse (Bruges kun under projektetablering i udviklingsprocessen)

- TURBO-analysen
 - Til at finde usikkerhed
 - Man kigger på styrker og svagheder ved f.eks. teknikken
 - Samt lave en projektstrategi, som er en overordnet fremgangsmåde i projektet.
 - Den sigter efter at udnytte mulighederne for succes i projektet.
 - Skal laves under projektetableringen af udviklingsprojektet når man har et produkt, man kan ikke lave det i forundersøgelsen, da man ikke endnu ikke ved om man har et produkt.

Risikoanalyse(Fortsætter hele projektet)

- Tre del elementer
 - Risikoidentifikation
I identifikationen kan man kigge på den interressentanalyse du lavede. Er der interesser der modstrider hinanden? Så skriv dem ned
 - Risikovurdering
Vurdering kan ske ved at man regner sandsynligheden på om det sker, og hvor alvorligt det er.(sandsynlighed * alvor)
 - Risikoimødegåelse
Man skal så kigge på om konsekvensen er lidt eller meget alvorligt
Samt hvordan det håndteres

Projektetableringens etablering

- Man kan vælge symbol
- Et navn til projektet
- Dette gør at man kan identificere sig med projektet
- Skaber fællesskabsfølelse, samt man bliver mindet om projektets formål

Bilag

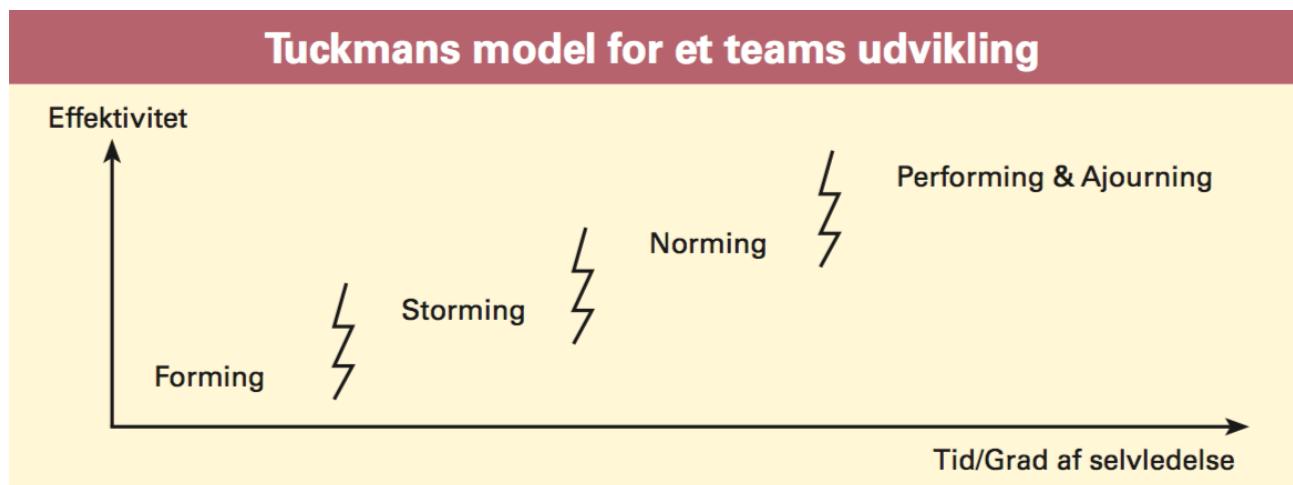
Belbin

Belbins teamroller

Der findes følgende ni teamroller som kan afdækkes af Belbin analysen:

Teamrolle	Bidrag til teamet	Tilladelige svagheder
Idémand	Idémanden er iderig, kreativ og opfindsom. Ser svære problemstillinger fra nye vinkler.	Kan være svag i sin kommunikation. Glemmer og ser ikke praktiske forhold.
Analysator	Analysisk og objektiv. Præcis dømmekraft. Ser de rationelle aspekter i en proces.	Kritisk og skeptisk. Kan have svært ved at begejstre andre.
Specialist	Yder med specialviden. Stærkt fagligt engagement og selvtillid. Meget fokuseret på eget fag.	Tendens til ofte at isolere sig. Vogter sit eget område og bidrager nævert inden for dette.
Opstarter	Dynamisk, højt gearet og rastløs. Udfordrer og skaber pres. Finder løsninger til forhindringer.	Tendens til at lade sig provokeret. Utålmodig og kan have et højt temperament.
Organisator	Disciplineret og loyal. Effektiv i gennemførende faser. Realistisk og praktisk.	Noget ufleksibel. Reagerer langsomt overfor nye forandringer.
Afslutter	Grundig, samvittighedsfuld og præcis. Leder efter fejl og forglemmelser.	Tilbøjelig til at bekymre sig unødvendigt. Utilbøjelig til at delegerer.
Koordinator	Selvsikker og tillidsfuld. God til at prioritere blandt opgaver. Klargør mål og frembringer beslutninger. Har øje for naturlanter.	Tendens til at manipulere og få andre til at gøre arbejdet.
Kontaktskaber	Ekstrovert, entusiastisk, kommunikerende. Udforsker muligheder. Skaber kontakter.	Taber nemt interessen, når entusiasmen har lagt sig.
Formidler	Socialt orienteret og skarpt iagttagende. Hensynsfuld og fleksibel. Skaber et godt klima.	Tilbøjelig til at være ubeslutsom og usikker i afgørende situationer. Kan virke konfliktsky og overfølsom.

En gruppens livscykel



Forklaring

- *Forming*
 - Teamet dannes her
 - Fokuseres på at komme i gang
- *Storming*
 - Valg foretages her
 - Hvem bestemmer?
 - Hvordan træffes beslutningerne?

- Konfliktniveauet når her sit højdepunkt
- Formelle og uformelle normer dannes
- *Norming*
 - Finder fælles mål
 - Roller og normer er næsten på plads
 - Gruppekultur dannes
- *Performing*
 - Teamet arbejder som en enhed
 - Høj autonomi
- *Adjourning*
 - Opgaven fuldføres
 - Teamet opløses

TURBO

Mål og vilkår	Særlig styrke	Særlige svagheder	Mulige beslutninger (strategielementer)
Teknikken		Ny teknologi	Hurtig adgang til konsulenter <small>Ringe til konsulenter hvis vi har brug for hjælp. med tekniken</small>
Udviklerne	Entusiasme		Hurtige delresultater kan bevare entusiasmen
Resultatet	Lille, overskueligt system	Ønske om høj kvalitet <small>Det er en svaghed fordi det ger det svære og det tager længere tid</small>	Kvalitet øges med dygtige folk og ekstern kvalitetsvurdering
Brugerne		Ingen ægte brugere at snakke med	Studere lignende systemer Finde nogle brugere
Omgivelserne		Risiko for afbrydelse p.g.a. andre opgaver	Trinvist forløb, modulært design Hurtigt færdig
Projektets strategi begrundet i projektets særlige mål og vilkår			
Sammenfatning <small>Her formuleres strategi</small>	Projektet skal have et trinvist forløb. For det første for hurtigt at prøve hele den nye metode, for det andet for at kunne justere ambitionsniveauet i tilfælde af afbrydelser, for det tredje for at bevare energien i projektet.		
	Projektet skal være udadvendt. Dels skal vi have inspiration og hjælp, så vi kan lære noget. Dels skal vi have kritik, så vi sikrer en høj kvalitet.		

2. Forundersøgelse

Hvad er en IT-forundersøgelse er. Gør rede for de grundlæggende principper i MUST-metoden. Gør overordnet rede for faserne i metoden.

Indledning

- Indledende analyse- og designarbejde, der danner grundlaget for en beslutning.
- Beslutningen omhandler, hvilke af de skitserede visioner for fremtidige it-anvendelser, der mest hensigtsmæssigt indfrier de forretsmæssige mål og brugernes behov for it-støtte til deres arbejde.
- En forundersøgelse skal producere beslutningsgrundlaget for at igangsætte et implementeringsprojekt
- Man opdeler et it-projekt i et forundersøgelsesprojekt og et implementeringsprojekt.
- Implementeringsprojektet handler om realisering af visionerne.
- En forundersøgelse skal producere beslutningsgrundlaget for at igangsætte et implementeringsprojekt og dette grundlag skal sikres bedre forankring. Derfor går forundersøgelse ud på:
 - At analysere virksomhedens forretnings- og it-strategi samt dens aktuelle mål, behov og muligheder
 - At designe en eller flere visioner om en samlet forandring
 - At vurdere visionerne i forhold til virksomhedens forretnings og it-strategier samt i forhold til grupper af ansatte, relationer mellem afdelinger og i forhold til kunder og leverandører.
 - At fastlægge en strategi og plan for den tekniske og organisatoriske implementering samt at udarbejde økonomisk overslag for visionernes implementering samt løbende at sikre sig feedback fra relevante aktører
- Forundersøgelsen indledes, fordi ledelsen vil forfølge nogle forretningsmæssige mål, fordi en eller flere grupper i en virksomhed har oplevet et problem eller at et problem eller opgaver kan løses af it-anvendelser.
- Forundersøgelsen startes fordi der er noget som er præget af forskellige grad af usikkerhed.
- Forundersøgelsen skal rede trådene ud.
- Resultatet af en forundersøgelse er en rapport måske suppleret med mock ups osv.
- Rapporten indeholder den samlede vision samt vurdering af konsekvenser ved indførsel af visionen. Samt en strategi og plan for implementering.

MUST metoden(Metode til forUndersøgelse i Systemudvikling – og Teori herom)

- IT-anvendelse: It-systemer set i en anvendelsessammenhæng, der inkluderer arbejdsorganisering samt kvalifikationer, som brugerne har brug for – for at kunne udføre arbejdet med den nye teknologi og i den nye organisering.
- Det betyder man ser forundersøgelser i et arbejdsorganisatorisk sammenhæng.
- Kendetegn ved MUST
 - Præsenterer et perspektiv på forundersøgelser, hvor man ser bredt på it-anvendelser
Man ser it-systemer i arbejdsorganisatorisk sammenhæng. Man kigger på, hvis man

implementer dette nye system, hvilke kvalifikationer er så nødvendige, før man kan anvende det.

- Et andet kendetegn er at man ser deltagelse af aktører fra virksomheden
Betyder at ledelse og brugerne skal medvirke i implementeringen
- Tredje kendetegn er at man ikke skal tage virksomhedens udgangspunkt for en forundersøgelse eller umiddelbare løsningsforslag forgivne.
- MUST-metoden hviler sig på fire principper
 - En samlet vision
 - Reel brugerdeltagelse
 - Arbejdsspraksis skal opleves
 - Forankring
- Resultat af MUST
 - En **samling af løsningsforslag** (visioner/systemdefinitioner) -> udvælgelse af én
 - **Konsekvensvurderinger** pr. vision (økonomisk / bredt)
 - **Plan for realisering/anskaffelse** pr. vision etc.

Princippet om en samlet vision

- En samlet vision omfatter forslag til it-systemer og arbejdsorganisering samt kortlægning af de kvalifikationer, som brugerne har behov for
- Tre elementer i samlet vision
 - It udvikling (Nye it systemer)
 - Organisatorisk udvikling (F.eks. man skal lære det nye it system at kende.)
 - Kvalifikationsudvikling (Når vi ændrer et IT system, så skal man helst kunne finde ud af at bruge det nye, kan være folk skal på kursus og uddanne sig)
- Så altså, hvis man ændrer i IT, kan det være man ændre i organisatorisk udvikling som kvalifikationsudvikling, fordi det kræver nye medarbejder med gode kvalifikationer.
- Man skal huske visionens fordele og ulemper
- Det er vigtigt at udvikle visioner der understøtter alle tre elementer.

Princippet om reel brugerdeltagelse

- De berørte ansatte skal deltage aktivt i projektgruppens arbejde
- Det skal være en gensidig læreproces. It-designerne skal kende til arbejdsområdet, hvor de ansatte skal kende til de teknologiske muligheder.
- Man sigter efter 2 effekter:
 - Visionen skal afspejle de ansattes situation
 - Systemerne skal anvendes i overensstemmelse med hensigten.
- Projektgruppen skal udvikle i tre områder, i en forundersøgelse
- Vigtigt citat: Brugere kan bidrage med innovative og konstruktive forslag til forandringer, når de får de rette betingelser. (Ting der kan forhindre os i dette er f.eks. hvis man er introvert, man kan lide sine egne tryggerammer så man tør ikke bidrage)

Princippet om at arbejdsspraksis skal opleves

- Handler om at man skal arbejdsspraksis skal opleves
- For at forstå et fænomen, skal man opleve det på første hånd

- Man kan gøre dette gennem 3 teknikker: Observation, in-situ interview og tænke-højt forsøg
- Say/do problemer. De siger det ene, men gør det andet.
- Nogen fænomener kan man kun komme til at forstå, hvis man forsøger at prøve dem
- MUST-metoden arbejder med to ændringstiltag
- 1: Tanker og forestillinger deltagerne gør sig undervejs i forundersøgelsen
 - Scenerier osv. diskuteres i styregruppen, så kan man diskutere konsekvenserne af dem, foretage ændringer og tilføjelser. Man arbejder i en forestillet verden.
- 2: Konkrete erfaringsskabende eksperimenter, som sigter på ændringer i den fysiske verden.
 - Forsøg af prototyper og gennem eksperimenter og der kan man opdage man mangler krav osv.

Princippet om forankring

- Handler om at informere om og skabe forståelse for samt opbakning til forundersøgelsen mål, visioner og planer.
- Handler om hvorfor og hvordan projektgruppen relaterer sig til projektets andre aktører.
- Det handler at forankre det hos brugerne, udviklerne og styregruppen(afdelingsleder).
 - Hvis brugerne f.eks. ikke vil bruge det så er der ingen pointe, derfor skal man få opbakning omkring det.
- God forankring af forundersøgelse resultater skabes hvis projektgruppen i deres interne arbejde og i de produkter den fremlægger over 3 forankringsregler.
 - **1. Adskil formodninger fra indsamlede informationer.** Vær løbende opmærksom på, hvad der kan føres tilbage til diverse dokumenter, lyd- eller videooptagelser og noter fra interview og observationer m.m., og hvad der er projektgruppens formodninger(antagelser og hypoteser)
 - **2. Afprøv overvejelser, antagelser og hypoteser,** ikke kun konklusioner. Vær åben overfor alle sådanne formodninger, og vær opmærksom på, at det er vigtigt at udfordre og afprøve disse. Synliggør overvejelserne ved at skabe sammenhæng i argumentationen fra problemidentifikation til løsningsforslag. Her kan fx anvendes diagnostisk og virtuel kortlægning.
 - **3. Giv visionerne et anvendelsesorienteret perspektiv.** Visualiser visionerne i forhold til brugssituationer. Brug fx scenarier til at beskrive sammenhæng mellem IT-systemer og arbejdsorganisering. Beskriv visionerne i forhold til de forretningsmæssige mål, så ledelsen kan se, om og hvordan disse indfries.

Faser i metoden

- Forberedelsesfasen(Proyektetablering) er her man skal have styr på planen og projektgrundlaget. Bestemmer rammerne for forundersøgelsen: tid, indhold, økonomi og deltagere. Her sker projektetableringen
 - Aktiviteter som:

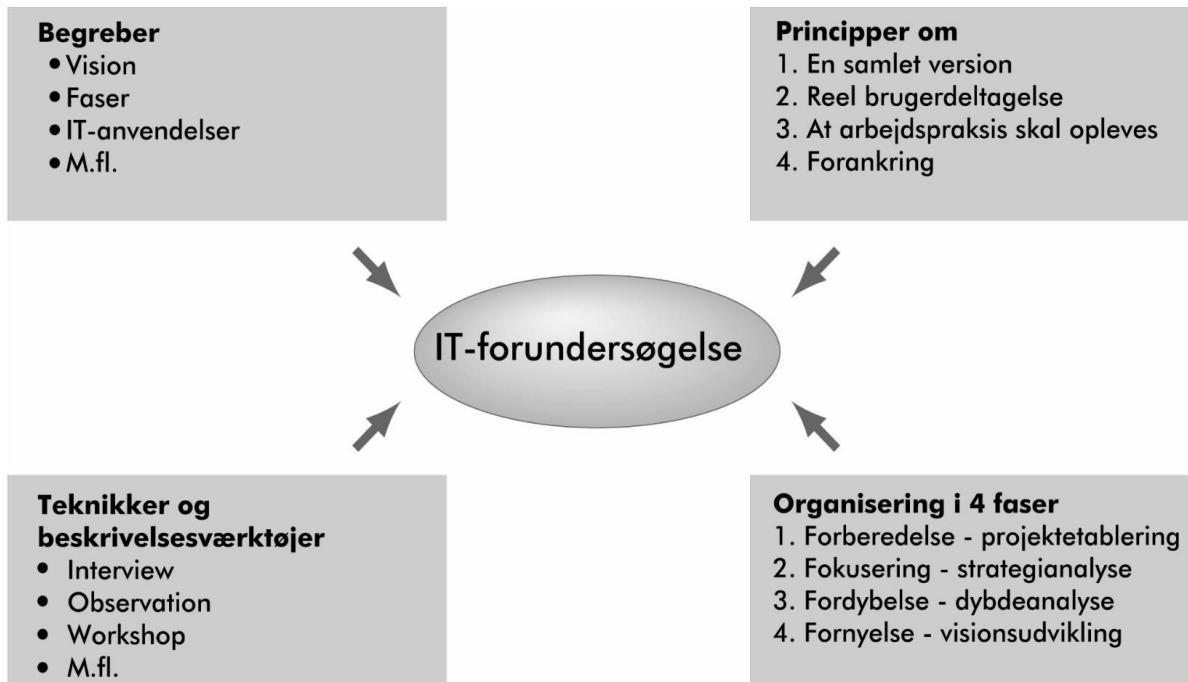
Social etablering

Kortlæg interesserter

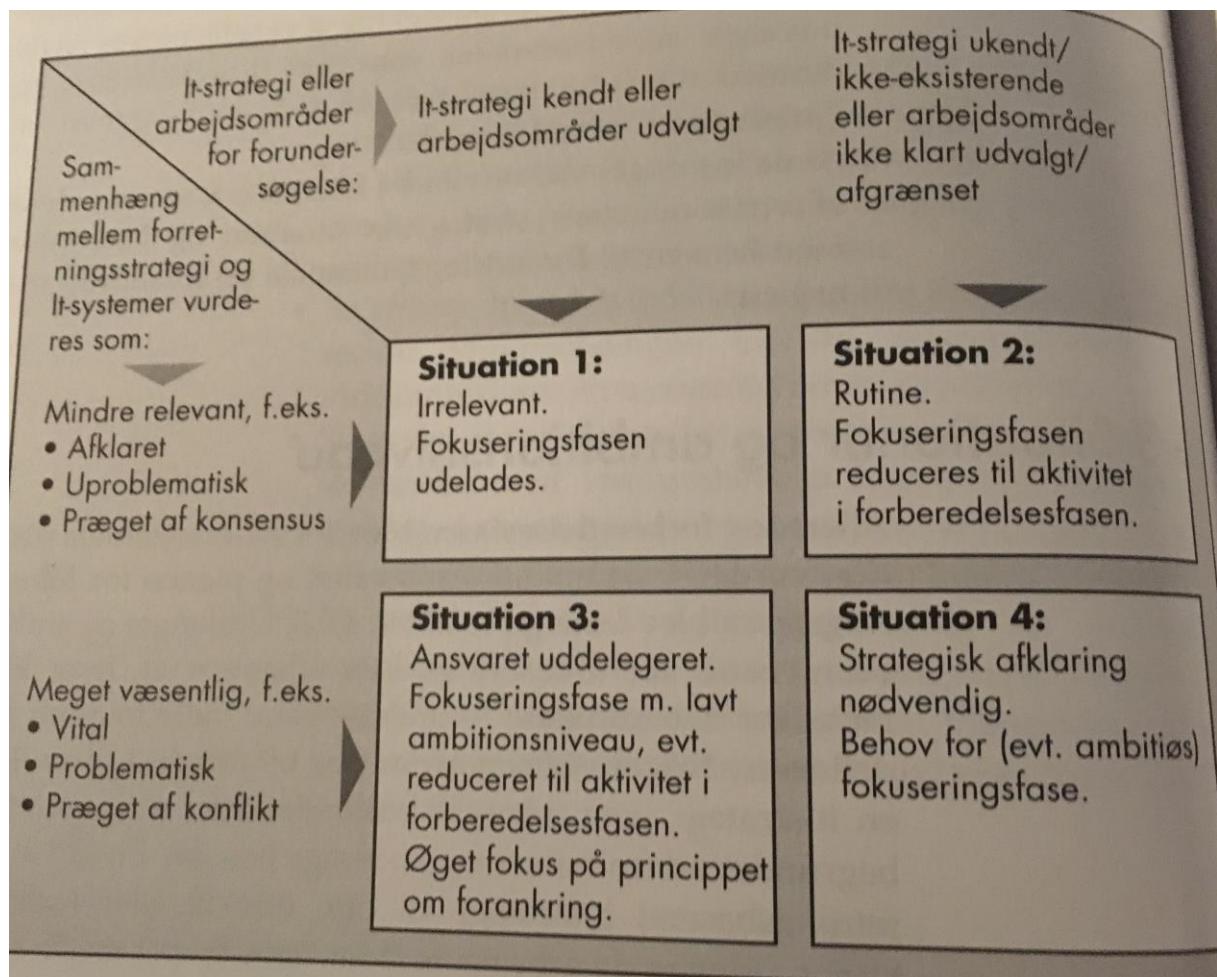
- Fokuseringsfasen(Strategianalyserapport). Leder til en strategianalyserapport. Hvilke arbejdsområder skal forundersøgelsen videre forløb fokusere på. Mellemprodukter som afklaring af relationen mellem forundersøgelsens mål og virksomhedens forretnings- og it-strategi.
Altså virksomheden har it-strategi og forretningsstrategi og de skal understøtte hinanden. Det så derfor vigtigt at vi laver et system, som kan understøtte forretningsstrategien og it-strategien.
Forretningsstrategi er for at sikre virksomhedens overlevelse.
IT-strategi: Det IT-mæssige svar på, hvordan man når det forretningsmæssige mål, IT-midlerne til at realisere og styrke forretningsstrategien.
Fokuseringsfasen omfatter en identifikation og analyse af virksomhedens omgivelser, såsom kunder, leverandører og konkurrenter.
Fokuseringsfasens resultat er en strategianalyserapport, der identifierer og prioriterer de arbejdsområder, som den videre forundersøgelse skal omfatte, og som er i fokus i fordybelsesfasen.
Man kan godt droppe fokuseringsfasen, men det afhænger af hvilken 'situation' man er i. Hvis man er i situation 1, så er man f.eks. afklaret med it-strategien og forretningsstrategien, så kan den droppes, eller hvis det er mindre relevant, i forhold til projektet.
 - AktiviteterAnalyse af forretningsstrategier
Analyse af IT-strategi
- Fordybelsesfasen(Dybdeanalyse). Leder til en analyserapport. Man bestemmer hvilke mål, problemer og behov der skal prioriteres, og hvilke ideer til it-støtte som skal designes mere i detaljer. Resultat er af fasen er en prioriteret liste over mål, problemer og behov samt ideer til it-støtte, ændret arbejdsorganisering og nye brugerkvalifikationer, som er udgangspunkt for fornyelsesfasen.
 - AktiviteterDataindsamling
Analyse af dataindsamling
(Teknik som diagnostiske kort, fordi nu da man er nået længere er det oplagt at opsummere de problemer der er samt så at nedskrive eventuelle løsninger)
- Fornyelsesfasen(Visionsudvikling) leder til den endelige forundersøgelsesrapport.
 - AktiviteterUdvikle og samle ideer
Mock-ups og prototyper
Her bestemmer man hvilke visioner der skal implementeres, og hvordan det skal foregå. Rapporten kan være suppleret af mock-ups, attrapper og prototyper(**Visuelle kort, oplagt at arbejde videre på de problemstillinger man fandt i fordybelsesfasen, man får et overblik over hvilke løsninger der er, så man får eventuelle ideer til implementering**)
- Generelt for alle faser:
 - Der indgår en række aktiviteter
 - Man befinner sig i en 'situation'. Situationerne fortæller hvor dybdegående, man skal gøre de enkelte faser.

Bilag

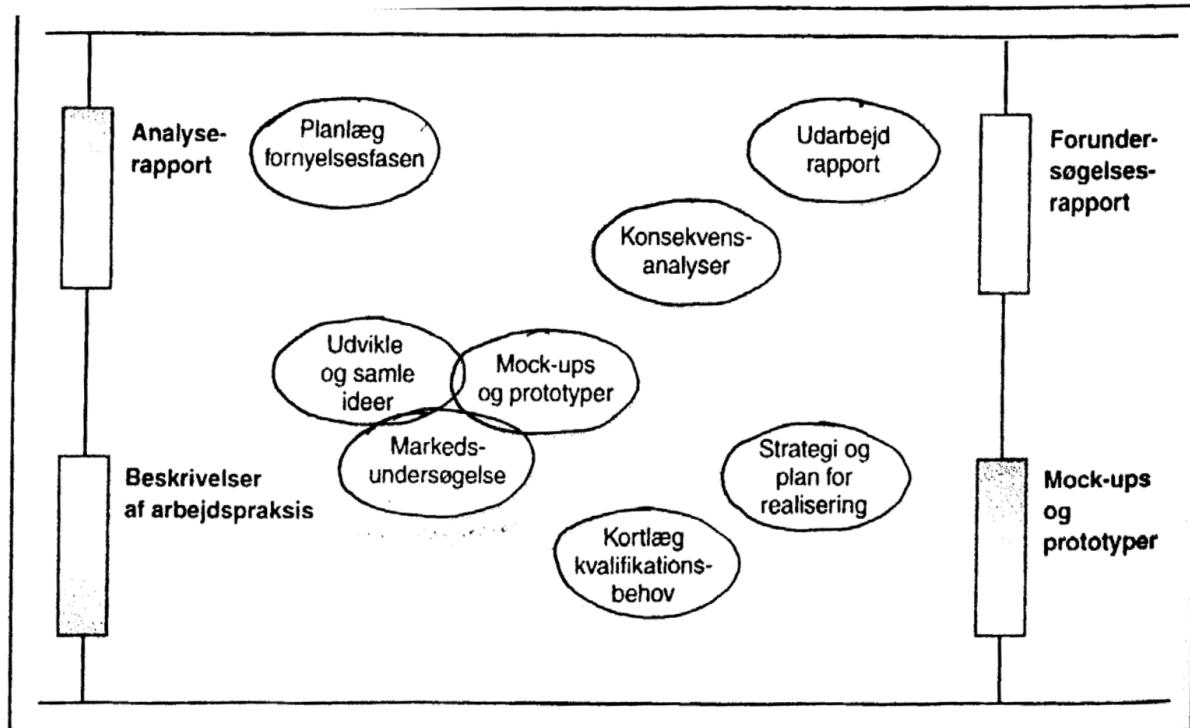
MUST-metoden ser således ud:



Situation eks. for fokuseringfasen



Aktiviteter fornyelsesfasen



Kortlægning

- **To typer kortlægning:**
 - **Diagnostiske kort – problematiske situationer**

Diagnostiske kort: Fordybelsesfasen
 Virtuelt kort: Fordybelses- og fornyelsesfasen(Oplagt i fornyelses)
 Diagnostiske korts ideer til løsning, kan bruges i virtuelt kort.

Problem	Årsag	Konsekvens	Ideer til løsning

- **Virtuelt kort – afprøve idéer til forandring**

Ideer til løsning	Handlinger	Konsekvenser	Vurdering
			Man vurder løsningen, ud fra konsekvenser osv.

Faserne kort

Faser	Fokus	Resultat - beslutninger
Forberedelsesfasen-projektetablering	Rammer for forundersøgelsen: Tid, indhold, økonomi, deltagere	Projektgrundlag + Plan forundersøgelsens udgangspunkt og rammer
Fokuseringsfasen-strategianalyse	Sammenhængen mellem forundersøgelsens mål og virksomhedens forretnings- og IT-strategier	Strategianalyserapport valg af arbejdsområder, der skal undersøges
Fordybelsesfasen-dybdeanalyse	Arbejdsspraksis i udvalgte arbejdsområder	Analyserapport + Beskrivelser af arbejdsspraksis prioritering af hvilke mål, problemer og behov, der skal indfries, og hvilke ideer til IT-støtte der skal konkretiseres
Fornyelsesfasen - visionsudvikling	Visioner om IT-systemer og deres sammenhæng med arbejdsorganisering og kvalifikationer Realiseringsprojektet	Forundersøgelsesrapport + mock-ups og prototyper hvilke visioner skal realiseres, og hvordan skal dette foregå

3. Agile metoder

Hvad karakteriserer de agile metoder. Hvilke agile metoder kender du, og kan du fortælle lidt om en af dem. Gør evt. rede for forskellige metoderetninger.

Indledning

- Når man arbejder med projekter, kan forskellige benytte sig af forskellige typer metoder.
- Det kommer an på, hvad dit projekt går ud på altså forskellige faktorer, og så kan du derfra vurder, hvilke typer metoder det er anbefalet at du benytter.
- Der er f.eks. plandrevne og agile metoder
- Agile udviklingsmetoder anvender timeboxed iterative og inkrementel udvikling. I stedet for at planlægge i dybden i starten af projektet, så er agile metoder åbne for ændringer undervejs og kræver konstant feedback fra brugerne af produktet.
- Hvis agile metoder havde et motto så ville det være embrace change.

Agile metoder

- Det agile manifest
 - **Mennesker og samarbejde** frem for processer og værktøjer [SEP]
 - **Velfungerende software** frem for omfattende dokumentation
 - **Samarbejde med kunden** frem for kontraktforhandling [SEP]
 - **Håndtering af forandringer** frem for fastholdelse af en plan
 - De agile principper
 - Vores højeste prioritet er at stille kunden tilfreds gennem tidlige og løbende afleveringer af værdifuld software [SEP]
 - Vi imødekommer ændringer i krav [SEP] også selvom de fremkommer sent i udviklingen. De agile processer sikrer at ændringer er til kundens fordel. [SEP]
 - Løbende levering af velfungerende software, jo hyppigere des bedre. [SEP]
 - Den mest effektive kommunikationsform er samtale ansigt til ansigt.
- Ud fra det kan man sige at følgende ting karakteriserer agile metoder
- Projektets **overordnede omfang, mål**, betingelser og risikoelementer er **kort dokumenteret**
 - **Korte, iterative, feature-drevne, time-boxede udviklingscykler**
 - **Konstant feedback og konstant forbedring** (daglige møder, sidde sammen, par-programmering, kunderespons osv.)
 - **Kundeinvolvering** [SEP]
 - **Godt design er vigtigt** (refactoring, designet gøres simplere og simplere)

XP – agil metode

- XP har nogle værdier, principper og praksis.
- Man kan sige at man bygger en bro ved hjælp af nogle værdier som gennem principper bruger nogle praksis
- XP går op i 4 værdier: kommunikation, simplicitet, feedback, mod og respekt.

- Kommunikation
 - Det er vigtigt for at lave en form for 'team' følelse og effektiv samarbejde
 - Hvis man har problemer er det en god ide at snakke om det, så finder man måske ud af en person har haft samme fejl før.
- Simplicitet
 - Simpelt design. Simpelt design kommer an på ens egen forståelse af simpelt.
- Feedback
 - XP teams vil have så meget feedback som muligt.
 - Feedback kommer i mange former: Meninger om en ide, hvordan kodet ser ud når du har implementeret ideen, om testsne var nemme at skrive, om testsne kører, hvordan ideen virker når den er sat i gang.
 - Feedback er også med til at gøre ting simpelt, for man kunne stille spg om hvilke af f.eks. 3 løsninger er den simpleste
- Courage
 - Modet til at fortælle sin meninger til sit team skaber tillid
 - Modet til at smide kode væk, som man har brugt altid på, men som ikke fungerer
 - Modet til at lede efter bedre løsninger
 - Modet til at søge efter konkrete svar, skaber feedback.
- Respect
 - Man skal have respekt for sine team mates, ellers fungerer det ikke.

XP – Principper

- Principperne:
 - **Humanity** - forholde sig til menneskelige behov som ønske om at være sociale, tilhøre team, udvikle sig, blive forstået osv. [SEP]
 - **Economics** – det du laver skal have forretningsværdi, opfylde forretningsmæssige mål osv. [SEP]
 - **Quality** – kvalitet betaler sig, slæk aldrig på kvaliteten
 - **Baby Steps** – udvikle små bidder

XP – Praksis

- Beskrivelse af nogle praktikker:
 - Pair programming:
 - Som udgangspunkt skal alt kode skabes ved brug af parprogrammering, hvor 2 personer sidder sammen ved en computer.
 - En person driver det hele, mens den anden observerer.
 - Sørger for høj kvalitet.
 - Praktikken tager tid at mestre og kan føles akavet til at starte med.
 - Være åben overfor den anden persons idé.
 - Løbende rokerer rundt og skift roller.
 - Test-first
 - Man tester først.
 - Energized work:

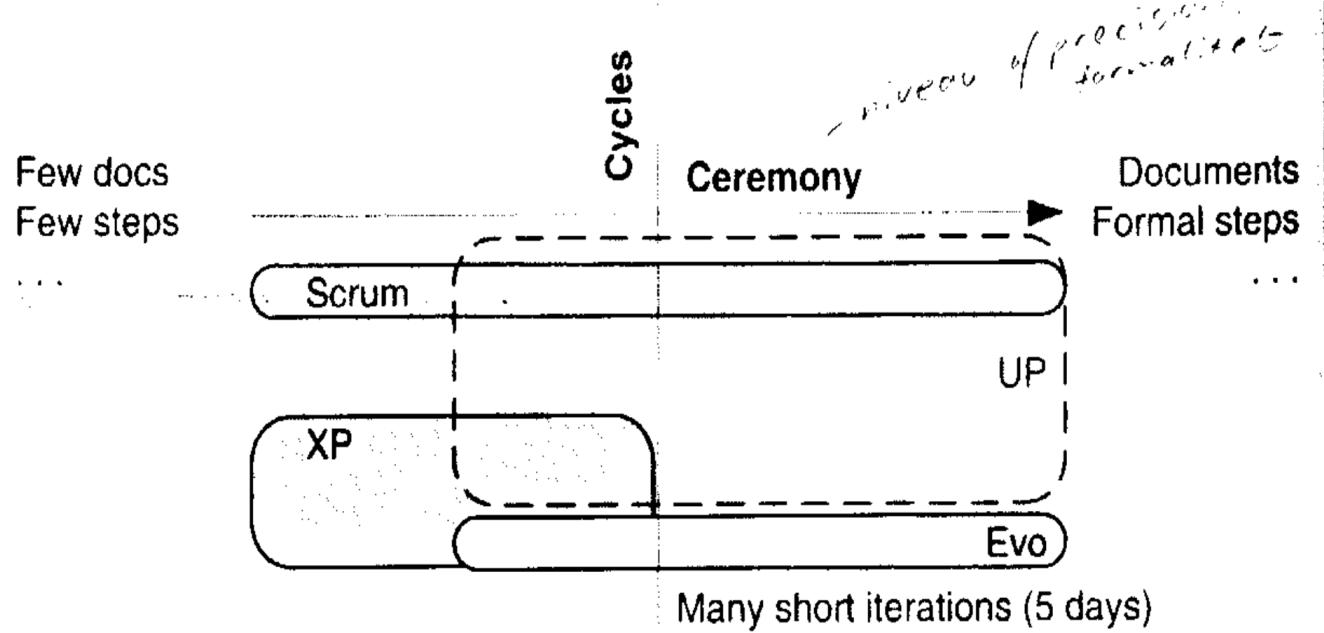
- En arbejdsuge er 37 timer lang, altså er der ingen overarbejde.
 - Er med til at sikre at medlemmerne er klar til at arbejde, da man ikke har ”lange” arbejdsdage.
 - Også med til at sikre at høj motivation er tilstede.
 - Fejl opstår tit når folk er trætte, så det er også med til at sikre høj kvalitet for ens produkt.
 - Sørge for at have en arbejdsplads, som virker energisk og motiverende.
- Whole team:
 - Man har et helt hold med de nødvendige kvalifikationer og kompetencer.
 - Vigtigt med god sammenhold.
 - Have et fælles mål.
 - Vigtigt med god sammenhold.
 - Eksempel:
 - Du har nogle bestemte værdier, som f.eks. kommunikation, hvilket gør du har nogle bestemte principper, for at have værdier som kommunikation, kræver det at der er nogen at kommunikere med, dvs. f.eks. mennesker, og ud fra det princip, laver du så nogle praksis som f.eks. pair programming

Andre metoderetninger

- Plandrevet
 - Støtter sig opad planlægning
- Vandfald vs. XP
 - Krav fastsat fra start, ingen mulighed for ændring
 - Mindre kundefeedback
 - Mere dokumentationspræget
 - Er mere til stabile omgivelser
 - Man skal have plan drevet til kritiske opgaver fordi at der kan være meget dokumentation mht. til lovgivning osv. Der skal være meget dokumentation om hvordan noget fungerer osv.
 - Kan håndtere flere mennesker end agile, da de oftes opdeles i plandrevne

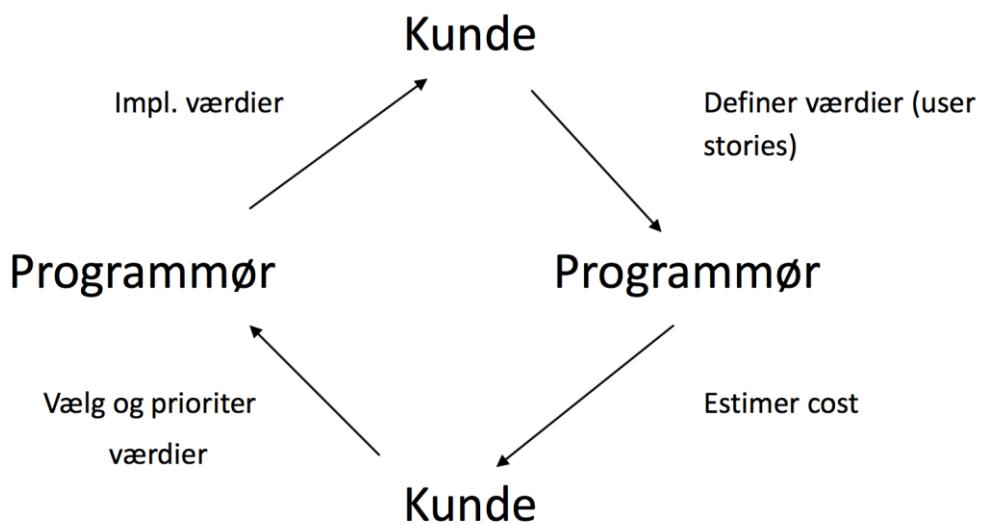
Bilag

Forskellige metoder



XP arbejdsproces

- Proces



4. SCRUM

Gør rede for de forskellige begreber og måder at gøre tingene på i SCRUM. Skitser evt. et systemudviklingsforløb under anvendelse af SCRUM.

Indledning

- En agil metode til at styre et projekt. Typisk software development, men ikke nødvendigvis, den kan bruges til mange ting.
- Kig på det som et framework/skabelon til at styre en proces.
- Scrum er en iterativ og inkrementel framework.
- XP har fokus på programmeringen, på hvordan man skal gøre tingene
- Der er ingen team leader. Der er ingen siger ikke hvis roller er hvis, det noget man bestemmer sammen.
- Scrum teams er understøttet af to roller. ScrumMaster som er coach for teamet, som hjælper deltagerne til at bruge Scrum så godt som muligt.
- Så er der product owner som er den anden rolle, den repræsenterer business, customers eller users og guider teamet så der kan laves et godt produkt. Produkt owner laver en vision for produktet.
- Der er 3 roller i Scrum: ScrumMaster, Product owner og deltagerne. Man kan se på denne analogi:
- Scrum teamet er bilen og følger den sti den bliver ledt hen imod. Product owner er køreren som sørger for de altid er på den rette vej og ScrumMaster er mekanikeren som sørger for at bilen performer så godt som muligt.

Agile metoder

- Det agile manifest
 - **Mennesker og samarbejde** frem for processer og værktøjer^[SEP]
 - **Velfungerende software** frem for omfattende dokumentation
 - **Samarbejde med kunden** frem for kontraktforhandling^[SEP]
 - **Håndtering af forandringer** frem for fastholdelse af en plan
- Kendetegn
 - Projektets **overordnede omfang, mål**, betingelser og risikoelementer er **kort** dokumenteret
 - **Korte, iterative, feature-drevne**(skal kunne have det features brugerne kræver), **time-boxede udviklingscykler**
 - **Konstant feedback og konstant forbedring** (daglige møder, sidde sammen, par-programmering, kunderespons osv.)
 - **Kundeinvolvering**^[SEP]
 - **Godt design er vigtigt** (refactoring, designet gøres simplere og simplere)

Scrum

Overordnet

- Scrum modellen bruger en serie af sprints. Sprints er timeboxed og tager ikke mere en måned, som regel tager det 1-4 uger.

- Man skal planlægge møder i starten af sprint, man skal finde ud af hvor mange ting deltagere vil lave. Man skal lave en backlog som er en liste af tasks som man skal lave under sprint.
- I en sprint skal man tage nogle funktioner som man vil kode. Som man havde som ide før.
- Hver dag i et sprint skal der være en scrum møde hvor ScrumMaster og product owner er med. Mødet skal tage ca. 15 min. Man skal her fortælle hvad man har lavet i går, og skal lave i dag.
- I enden af hver sprint viser man hvad man har lavet til PO.
- Efter hver sprint viser de hvad de har lavet hvor de kan få feedback som de kan lave i næste sprint.
- Består af 5-10 personer
- Feedback loop i Scrum gør at man kan lave ændringer osv løbende.
- Sprint backlog er teamets to-do list for sprintet. Produktet backlog er de funktioner der skal laves.

Scrum Roller

- Tre primære roller, the product owner, the team og scrummasteren.
- Product ownerens rolle er at få maximum business value.
 - Han skrive alt ned der skal laves fra både deltagerne og stake holders, og laver en krav liste.
 - Han bestemmer f.eks. hvad der skal være i næste sprint, for der skabes værdi for ham.
- Product owneren kan være kunden eller andre personer.
- Teamet laver produktet.
- Teamet er self-managing.
- Typisk 5-10 medlemmer.
- ScrumMaster gør alt for at teamet bliver succesfuldt.
- Skal ikke ses som projektleder, han støtter teamet. Han gør alle forstående omkring scrum
- Man kan se dem mere som coaches.

Starting Scrum

- Man starter Scrum med en prioriteret liste om hvad der krav til produktet der skal laves.
- Det er kaldet en product backlog, den der er højst prioriteret står øverst.
- Backloggen vokser med tiden og ændres med tiden.

Scrumboard

- Overblik over hvor langt man er på projektet

Sprint Planning Meeting

- I begyndelsen af hver sprint, så er der en sprint planning meeting.
- Scrum Team, Product owner reviewer product backlog.

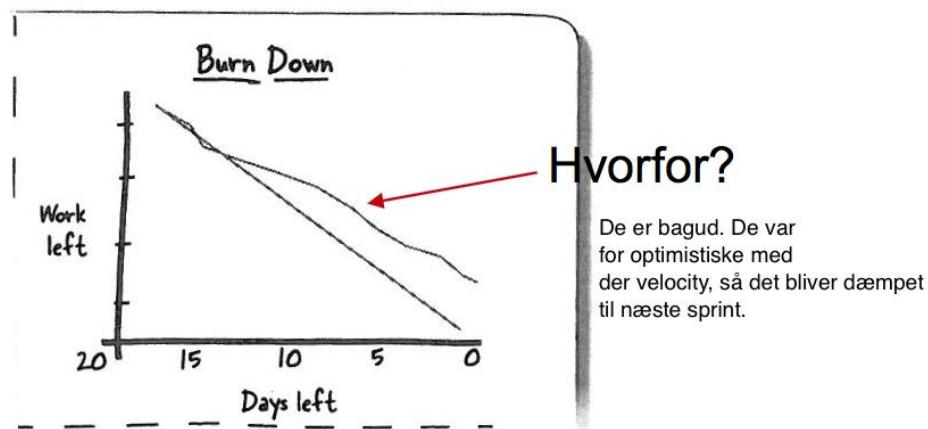
- Scrumt teamet vælger ting fra product backlog som der kan arbejdes med.
- De tager så de task i product backlog og gør dem til mindre opgaver og estimerer dem, det sættes så ind i sprint backlog. Team members vælger så hvilke opgaver de vil arbejde med.

Daily Scrum

- Daily scrum er et møde so varer ca. 15 min, som man møder til hver dag. Detter er også kaldet standup meeting.
- Man fortæller hvor meget man har nået osv. hvad man vil nå til næste gang.
- De skal skrive i sprint backlog hvor meget de mangler.
- Hvis man ikke har nået det man skulle i sin sprint, så er det synd. Man får ikke mere tid. De skal lære at estimere rigtigt.
- Hvilke udfordringer er der? Måske er der andre der kan hjælpe

Burndown chart

- Scrum masteren skriver så på hvor mange timer der er tilbage i sprintet, i sprint burndown chart.
- En burndown chart er som egen en lige linje
- Y aksen er work-left og x aksen er days left



Sprint review meeting

- Efter hvert sprint viser man hvilke product backlog opgaver man har nået
- Man viser f.eks. demoer.

Sprint Retrospective

- Efter sprint review samler de sig for sprint retrospective.
- Her snakker de om hvad der virker og hvad der ikke virker.
- De snakker om hvilke ændringer der skal være til næste sprint, som kunne gøre den mere produktiv.

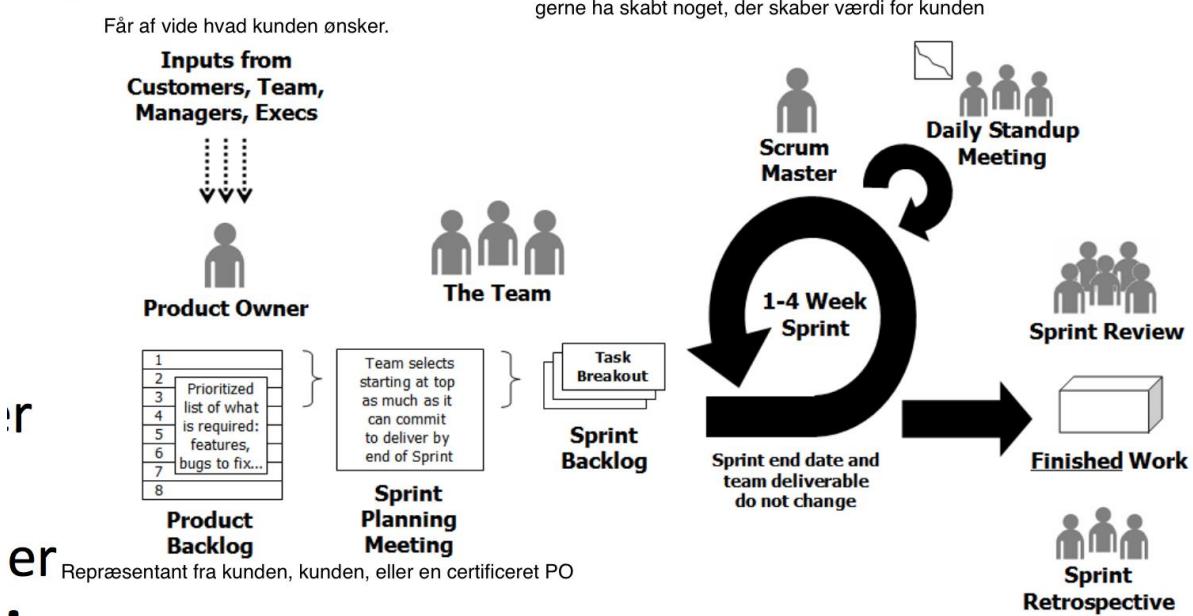
Scrum artifacter

- Den primære artifikat i scrum er produktet selv. Produkt backlog er også et artifikat.
- Andre artifacts er sprint burndown chart og release burndown chart. Burndown charts viser hvor meget arbejde der er tilbage i sprint eller en release. Her kan man se om man følger tidsplanen.

Bilag

Scrum udviklingsforløb

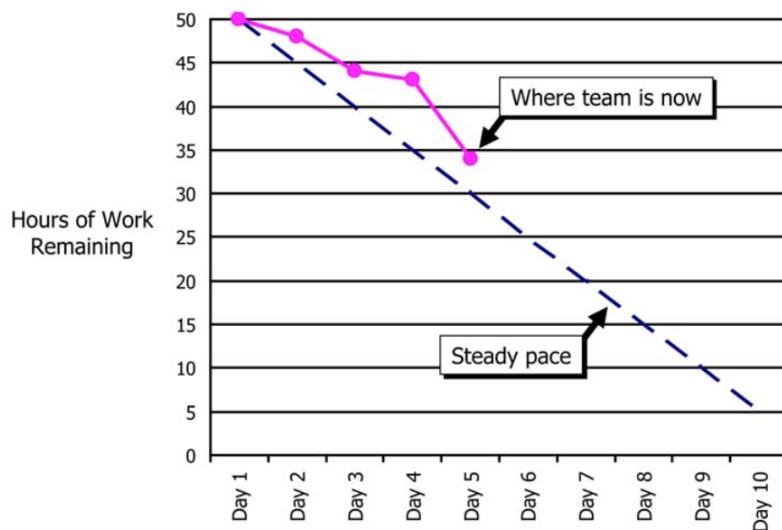
Log



er Repræsentant fra kunden, kunden, eller en certificeret PO

ier

Burndown chart



5. XP

Giv en beskrivelse af XP som ”systemudviklingsmetode”, kom med eksempler på nogle af de værdier og praktikker XP er bygget op omkring. Vælg et par af praktikkerne og beskriv dem nærmere.

Indledning

- Det agile manifest
 - **Mennesker og samarbejde** frem for processer og værktøjer^[SEP]
 - **Velfungerende software** frem for omfattende dokumentation
 - **Samarbejde med kunden** frem for kontraktforhandling^[SEP]
 - **Håndtering af forandringer** frem for fastholdelse af en plan
 - De agile principper
 - Vores højeste prioritet er at stille kunden tilfreds gennem tidlige og løbende afleveringer af værdifuld software^[SEP]
 - Vi imødekommer ændringer i krav,^[SEP]også selvom de fremkommer sent i udviklingen. De agile processer sikrer at ændringer er til kundens fordel.^[SEP]
 - Løbende levering af velfungerende software, jo hyppigere des bedre.^[SEP]
 - Den mest effektive kommunikationsform er samtale ansigt til ansigt.
- Ud fra det kan man sige at følgende ting karakteriserer agile metoder
- Projektets **overordnede omfang, mål**, betingelser og risikoelementer er **kort** dokumenteret
 - **Korte, iterative, feature-drevne, time-boxede udviklingscykler**
 - **Konstant feedback og konstant forbedring** (daglige møder, sidde sammen, par-programmering, kunderespons osv.)
 - **Kundeinvolvering^[SEP]**
 - **Godt design er vigtigt** (refactoring, designet gøres simpelere og simpelere)

XP – agil metode

- XP har nogle værdier, principper og praksis.
- Man kan sige at man bygger en bro ved hjælp af nogle værdier som gennem principper bruger nogle praksis
- Vigtigste ved XP er koden og kvaliteten.
- XP går op i 5 værdier: kommunikation, simplicitet, feedback, mod og respekt.
- Kommunikation
 - Det er vigtigt for at lave en form for ’team’ følelse og effektiv samarbejde
 - Hvis man har problemer er det en god ide at snakke om det, så finder man måske ud af en person har haft samme fejl før.
- Simplicitet
 - Simpelt design. Simpelt design kommer an på ens egen forståelse af simpelt.
- Feedback
 - XP teams vil have så meget feedback som muligt.
 - Feedback kommer i mange former: Meninger om en ide, hvordan kodet ser ud når du har implementeret ideen, om testsne var nemme at skrive, om testsne kører, hvordan ideen virker når den er sat i gang.

- Feedback er også med til at gøre ting simpelt, for man kunne stille spg om hvilke af f.eks. 3 løsninger er den simpleste
- Courage
 - Modet til at fortælle sin meninger til sit team skaber tillid
 - Modet til at smide kode væk, som man har brugt altid på, men som ikke fungerer
 - Modet til at lede efter bedre løsninger
 - Modet til at søge efter konkrete svar, skaber feedback.
- Respect
 - Man skal have respekt for sine team mates, ellers fungerer det ikke.
- NÆVN DE 4 VARIABLE, tid, pris, kvalitet, omfang.
- Man leverer altid til den givne tid, pris og kvalitet, man ændrer kun på omfanget.

XP – Principper

- Principperne:
- **Humanity** - forholde sig til menneskelige behov som ønske om at være sociale, tilhøre team, udvikle sig, blive forstået osv. [SEP]
- **Economics** – det du laver skal have forretningsværdi, opfylde forretningsmæssige mål osv. [SEP]
- **Quality** – kvalitet betaler sig, slæk aldrig på kvaliteten
- **Baby Steps** – udvikle små bidder [SEP]

XP roller

- Kunde
 - Leverer user stories, accepttests, træffer beslutninger og prioriterer
- Udvikler
 - Unittest, estimering, programmering
- Projektleder
 - Problemløser, fjerner forhindringer, ekstern kommunikator, koordinator
- Tester
 - Hjælpe kunden med at skrive og gennemføre accepttest, sørge for testværktøjerne virker, sørge for integrationsmiljøet fungerer

Iteration Planning

- Kunderne bestemmer hvilke user stories der skal implementeres i hver iteration
- Kunden kan ikke ændre på user stories når de er i gang, men kan give feedback til næste gang
- Hvis man ikke når alle user stories som man lovede skal man rette velocity til. Hvis man kun nåede 35 point i iterationen er den velocity i næste iteration 35 point.

The Halfway Point

- Halvvejs gennem iterationen, holder teamet et møde.
- Halvdelen af historierne skulle gerne være færdige nu.

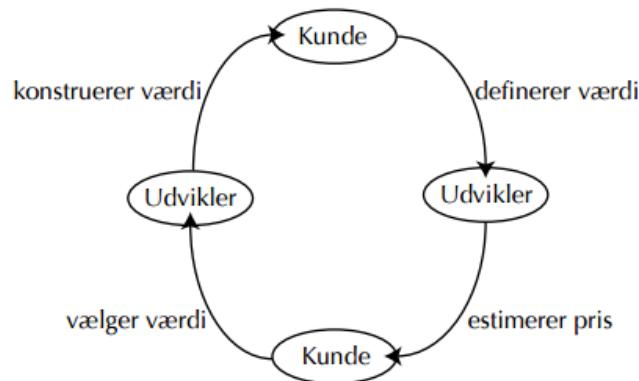
- De skal så se på om de kan nå resten af user stories, og hvis de ikke kan det, kan kunderne vælge at tage en user story fra.
- Det er vigtigt at se at det user stories der bliver færdige og ikke bare opgaver. F.eks. forskellige opgaver fra forskellige user stories som ikke er samlet.

Iterating

- Ved enden af hver iteration skal man demonstrere hvad man har lavet.
- Kunderne skal så fortælle hvad de synes

Extreme Programming:

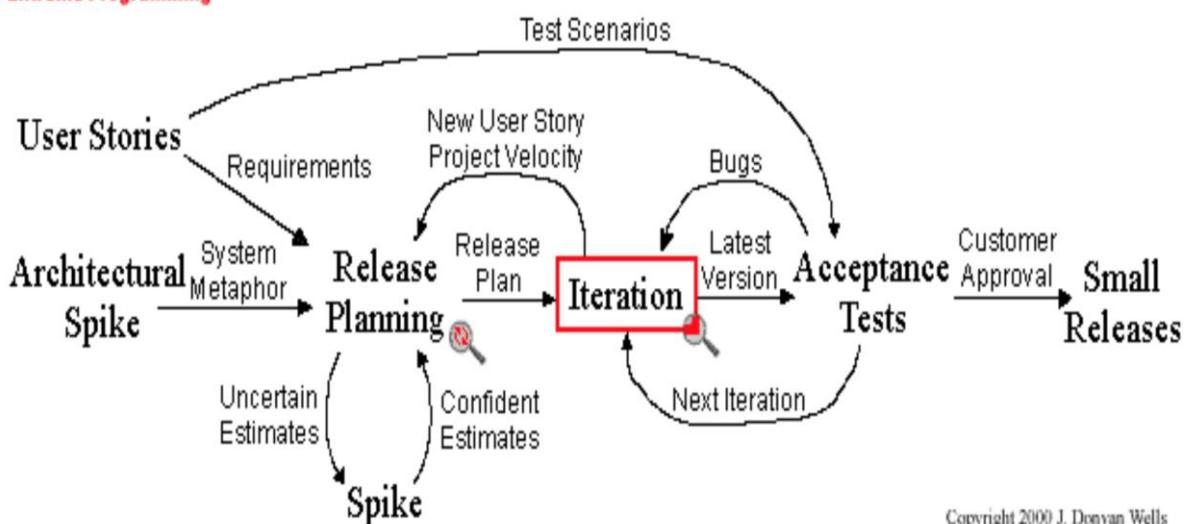
- Overordnet set består systemudvikling med XP af følgende fire aktiviteter:
 1. At programmere. – Betyder allermest i XP
 2. At teste. – Teste, færre fejl
 3. At lytte. – Imødegå kundens krav
 4. At designe - Hvis det er kompliceret er design vigtigt
- Ved XP er der følgende hovedprincipper:
 - Hurtig feedback.
 - Antag simpelhed.
 - Forandring skal være trinvis.
 - Tag forandring til dig.
 - Udfør kvalitets arbejde.
- Meget vægt og tillid til det enkelte individ.
 - Man pålægges ikke arbejdsopgaver, men går i selv ind og vælge det man ønsker.
- Teamet skal gå efter et fælles mål.
- Bestræber sig på at imødekomme kundens ønsker.
- Ligger meget vægt på kommunikation.
- Af spejler sig fra ordinære systemudviklingsmetoder ved at have meget vægt på det sociale igennem metodens værdier, principper og praksis.
- Tid, kvalitet og pris er altid faste, men omfanget kan justeres undervejs.
- Code review: Hvor du hele tiden til har en til at tjekke andres kode, for så man ikke laver fejl.
- User stories:
 - Beskriver de opgaver, som skal laves for kunden.
 - Skal udarbejdes i samarbejde med kunden.
 - Meget vigtigt, at de har værdi for kunden.
 - Skal være mulige at teste (accepttest).
 - Udføre en user story, så man har en velocity at tage udgangspunkt i.
 - Sætte estimater på hvor lang tid en story tager at udføre ud fra ens estimat.
- Task planning:
 - Muligheden for at splitte user stories op i mindre tasks, så man får et bedre overblik.



- God idé at estimere task'ne i units, så man har en idé om hvor mange timer det tager at udføre.
- Planning Game:
 - Udfører iterationer og releases igen og igen.
 - Præsentation overfor kunden, når man er færdig med en iteration.
 - Planning game består af:
 - Iterationer:
 - Over projektets forløb udføres der en række iterationer.
 - Planlagt i samarbejde med kunden, så kunden får de ønskede user stories udført.
 - Tilstræbe sig på, at få sine user stories til at passe med ens velocity.
 - I slutningen af hver iteration tilpasse teamets velocity, hvis det er nødvendigt, så den passer bedre til den næste iteration.
 - Sørge for at lave passende automatiske tests-first.
 - Testne skal afspejle kundens krav, så systemet er i stand til at udføre det som kunden ønsker.
 - Med til at øge udviklingshastigheden, da man ikke skal bruge nær så meget tid på at lave fejlsøgning.
 - Hvorfor tests first?
 - Hurtigere at kode systemet
 - Du ved hvad funktionaliteten kræver, hvis du laver tests først, da det giver dig et overblik over kravet for funktionen.
 - Sparer dig tid, hvis du laver testene først frem for senere.
 - Hurtig feedback fordi du hurtigt kan teste, du kan se om du er på rette vej.
 - Release planning:
 - Løbende mulighed for at udbygge systemet med flere user stories, da man løbende har kontakt med kunden.
 - Udviklere skal give hver user story points.
 - Releases skal ikke ske for ofte – kan virke forstyrrende overfor kunden.
 - Release ca. hver 3. måned.
 - Hvis man undervejs er bagud, har man mulighed for at gå i ”afslutter mode”, hvor man fokuserer på at få det mest essentielle færdiggjort.
 - Hellere have kvalitet fremfor kvantitet.
 - Man laver spikes, for at finde svar på svære tekniske spørgsmål. En spike solution er en simple program, som kan opdage potentielle løsninger.
 - Til at starte med kan XP godt være besværligt, men løbende bliver man bedre til de forskellige teknikker.
 - Eksempelvis bliver man bedre til at estimerer, udarbejde user stories med kunden, blive vant til de forskellige praktikker osv.
 - Beskrivelse af praktikker:
 - Pair programming:
 - Som udgangspunkt skal alt kode skabes ved brug af parprogrammering, hvor 2 personer sidder sammen ved en computer.
 - En person driver det hele, mens den anden observerer.

- Sørger for høj kvalitet.
 - Praktikken tager tid at mestre og kan føles akavet til at starte med.
 - Være åben overfor den anden persons idé.
 - Løbende rokerer rundt og skift roller.
- Energized work:
 - En arbejdsuge er 37 timer lang, altså er der ingen overarbejde.
 - Er med til at sikre at medlemmerne er klar til at arbejde, da man ikke har ”lange” arbejdsdage.
 - Også med til at sikre at høj motivation er tilstede.
 - Fejl opstår tit når folk er trætte, så det er også med til at sikre høj kvalitet for ens produkt.
 - Sørge for at have en arbejdsplads, som virker energisk og motiverende.
 - Whole team:
 - Man har et helt hold med de nødvendige kvalifikationer og kompetencer.
 - Vigtigt med god sammenhold.
 - Have et fælles mål.

Bilag



6. Metodevalg

Gør rede for hvordan en systemudviklingsmetode vælges, hvilke faktorer kigger man på osv.

Indledning

- For at komme i gang med et projekt, skal man vælge en metode som passer bedst til ens projekt.
- Projekter varierer så meget, og det kommer bl.a. an på hvem man laver et projekt for, samt hvilke produkt der skal produceres ud fra projektet
- Man skal derfor kigge på en række faktorer, når man skal i gang med metodevalget, til projektet.
- Cockburn mener der er en metode til alle projekter, men ikke en metoder passer på alle projekter.

Hvilke faktorer kigger man på?

- Man skal kigge på faktorer som:
 - TURBO
 - Projektets/applikationens karakteristika
 - Ledelsesmæssige karakteristika
 - Teknikker
 - Personlige karakteristika

TURBO

- TURBO er bl.a. til at finde usikkerhed.
- Man kigger nemlig på særlige styrker og svagheder ved f.eks. teknikken og udviklerne
- Hvis der er meget usikkerhed er det anbefalet at arbejde agilt
- Hvis der ikke er usikkerhed, kan du vælge at arbejde med alt
- Der knytter sig til analyseskemaet en analyseteknik, som omfatter 4 trin, og fem kasser med byggeklodser, det vil sige ideer til, hvad man kan fylde i skemaet.

Mål og vilkår	Særlig styrke	Særlige svagheder	Mulige beslutninger (strategielementer)
Teknikken		Ny teknologi	Hurtig adgang til konsulenter Ringe til konsulenter hvis vi har brug for hjælp med teknikken
Udviklerne	Entusiasme		Hurtige delresultater kan bevare entusiasmen
Resultatet	Lille, overskueligt system	Ønske om høj kvalitet <small>Det er en svaghed fordi det ger et svært og det tager længere tid</small>	Kvalitet øges med dygtige folk og eksterne kvalitetsvurdering
Brugerne		Ingen ægte brugere at snakke med	Studere lignende systemer Finde nogle brugere
Omgivelserne		Risiko for afbrydelse p.g.a. andre opgaver	Trinvist forløb, modulært design Hurtigt færdig
Projektets strategi begrundet i projektets særlige mål og vilkår			
Sammenfatning <small>Her formuleres strategi</small>	<p>Projektet skal have et trinvist forløb. For det første for hurtigt at prøve hele den nye metode, for det andet for at kunne justere ambitionsniveauet i tilfælde af afbrydelser, for det tredje for at bevare energien i projektet.</p> <p>Projektet skal være uadadvendt. Dels skal vi have inspiration og hjælp, så vi kan lære noget. Dels skal vi have kritik, så vi sikrer en høj kvalitet.</p>		

Projektets/applikationens karakteristika

- Her kigger man på
 - Projektetsmål
 - Projektetskompleksitet
 - Projektets størrelse
 - Projektets omgivelse
- Kan målene reagere på forandringer? Så skal man arbejde agilt.
- Hvis projektet er komplekst, er det anbefalet at benytte sig af plandrevet metoder. Hvis man f.eks. kigger på at lave et system til et hospital, kan man sige det ret komplekst fordi det handler om at redde menneskeliv. Et komplekst projekt kræver meget dokumentation, og det derfor det anbefalet med plandrevne metoder
- Hvis der er mange mennesker i projektet er det mere ideelt at arbejde med plandrevne, fordi der deles om op i teams, hvor man f.eks. i Scrum, ville have det svært med at have et standup meeting med 100 mennesker.
- Hvis projektets omgivelser er meget stabile, så kan man bruge plandrevne metoder, men hvis de er ustabile er det bedst man bruger agile. Agile metoder kan bedst tilpasse sig forandringer fordi man benytter sig af iterationer, og dermed nemt foretager sig ændringer undervejs

Ledelsesmæssige karakteristika

- McGregors, X-syn(dovent) og Y-syn(Ansvarlig)
- Kræves der høj grad af planlægning? Agile metoder benytter sig ikke af planlægning som sådan, kun når projektet skal til at slutte, hvor plandrevne støtter sig op ad planlægning. Hvis lederen også er kontrollerende kan det være svært med agile metoder, fordi det kræver mange frihedsgrader, fordi når man arbejder med agile, er en gruppestyrende. En leder kan ikke kigge efter hvad hvert XP par laver. Man stoler på sine programmører i agile metoder. Hvis lederen er kontrollerende kan det være

han ikke stoler på hvad hans medarbejder har lavet, og det vil blive et problem i agile metoder, hvis lederen lige pludselig vil ha' lavet det på en anden måde, end det et XP par har lavet f.eks.

- Har man mulighed for at snakke med kunden. Agile metoder sætter meget fokus på der er meget kontakt til kunden

Karakteristika mht. teknikker

- Hvilke teknikker har i tænkt jer, at benytte jer af?
- Dette ved man selvfølgelig før når man har valgt metode, men hvis man har lavet lignende projekter før og har haft succes med det, ved man måske allerede vælge teknikker man vil bruge.
- Simpelt design = agilt
- Test først = agilt

Personlige karakteristika

- Alle kunder skal være CRACK(Collaborative, Representative, Authorized, Committed, Knowlegeable), både i plandrevne og agile
- Men i agile metoder kræver man næsten at kunden er fuld tid, da agile metoder ligger sig op ad person til person kommunikation. Her kan man se på det agile manifest som siger noget om samarbejde med kunde frem for kontraktforhandling, samt deres mennesker og samarbejde frem for processer og værktøjer. I plandrevne er det oftest envejs kommunikation, hvor man har en kontrakt med kunden.
- Agile metoder kræver gode programmører fordi det er så selvstændigt
- Hvis man har mange grader af frihed, er agile metoder en mulighed.

Øvelser til metodevalg

- Man kan lave Houston Matrix – Todd Little(Der er et regneark til det. Dag 6, man kigger på f.eks. faktorer og giver den rating, f.eks. størrelsen af teamet(fra 1-10))
- Man kan også bruge Cockburns skala
- Man kan bruge Boehm dimensioner(Personnel, dynamism, culture(selvledelse og kaos = agilt), size, criticality)
- Disse øvelser, kan man bruge til hvad ens projekts passer mest til.

Konklusion

- Ud fra dette, skal man kigge på de forskellige karakteristika.
- Hvordan passer med dit projekt ind i dette, karakteristika?
- Heller den mere til de agile eller de plandrevne?
- Nævn eksempler fra dit projekt, hvis nødvendigt.

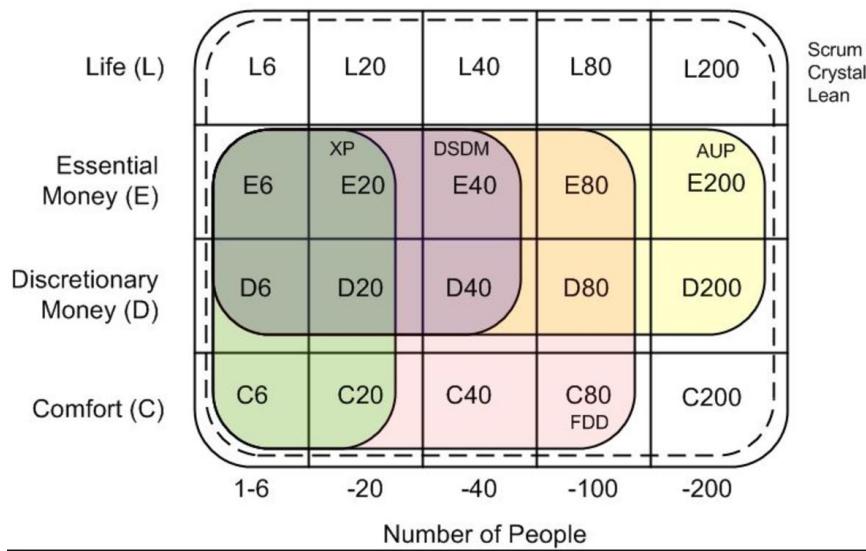
Øvrige ting

Cockburn mht. udviklernes metodeforståelse

- Som sagt, skal man være rimelig øvet, før man kan benytte sig af agile metoder.
- Cockburn har så opstillet nogle niveauer, som beskriver den enkeltenes metodeforståelse.
- Se bilag for et billede.
- **Note: Nævn ikke dette, den er kun med i dispositionen for en sikkerhedsskyld.**

Bilag

Cockburns skala

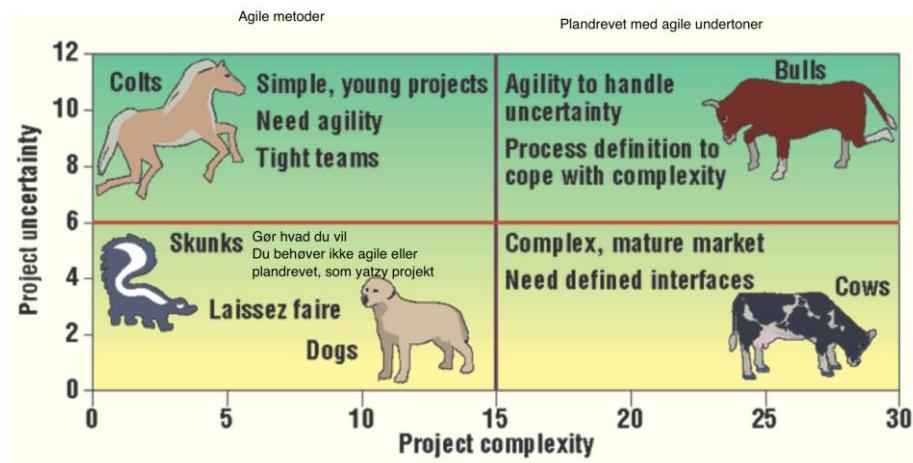


Note mht. til Cockburns skala: Scrum fylder hele skalaen fordi det er muligt at man i Scrum, kan deles op i Scrum grupper.

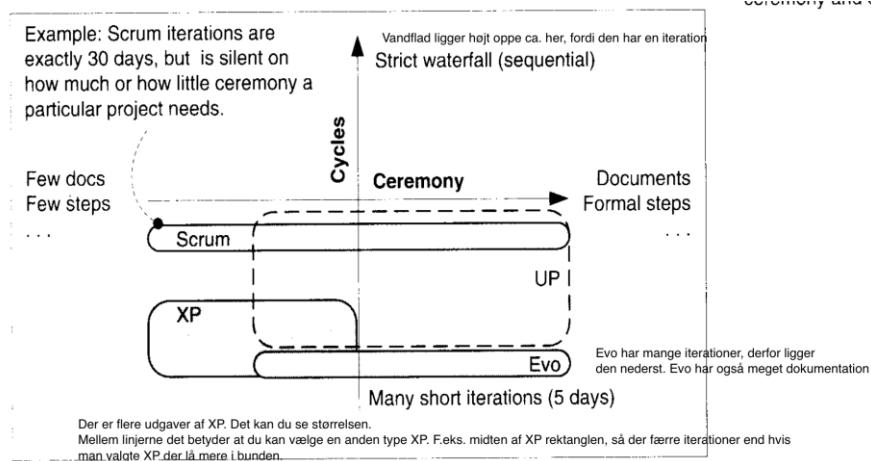
Houston Matrix – Todd Little

Complexity attributes and their scores*					
Attribute	Complexity score				
	1	3	5	7	10
Team size	1	5	15	40	100
Mission criticality	Speculative	Small user base	Established market	Mission-critical with large user base	Safety-critical with significant exposure
Team location	Same room	Same building	Within driving distance	Same time zone +/- 2 hrs.	Multisite, worldwide
Team capacity	Established team of experts	New team of experts	Mixed team of experts and novices	Team with limited experience and a few experts	New team of mostly novices
Domain knowledge gaps	Developers know the domain as well as expert users	Developers know the domain fairly well	Developers require some domain assistance	Developers have exposure to the domain	Developers have no idea about the domain
Dependencies	None	Limited, well insulated	Moderate	Significant	Tight integration with several projects

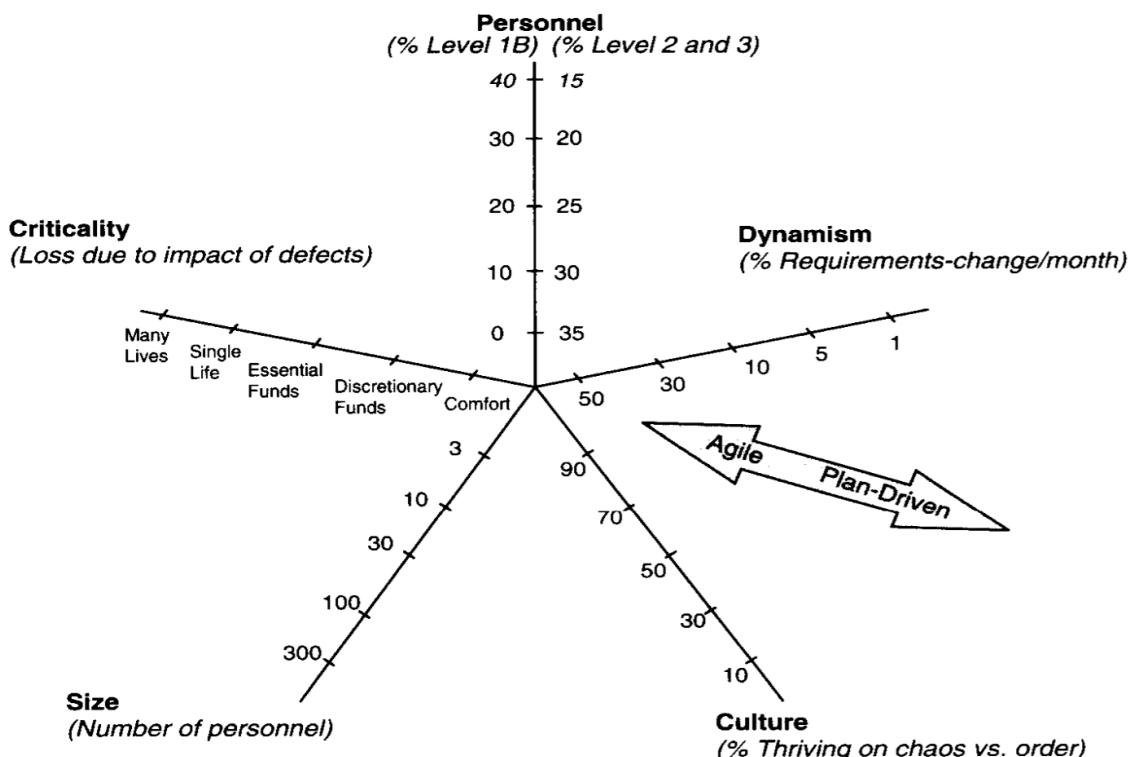
* Minimally complex = 1, highly complex = 10.



Klassifikation af metoder



Boehnms dimensioner



Cockburns metodeforståelse

Table 2-1 Levels of Software Method Understanding and Use (after Cockburn)

Level	Characteristics
3	Able to revise a method (break its rules) to fit an unprecedented new situation
2	Able to tailor a method to fit a precedented new situation
1A	With training, able to perform discretionary method steps (e.g., sizing stories to fit increments, composing patterns, compound refactoring, complex COTS integration). With experience, can become Level 2.
1B	With training, able to perform procedural method steps (e.g., coding a simple method, simple refactoring, following coding standards and CM procedures, running tests). With experience, can master some Level 1A skills.
-1	May have technical skills, but unable or unwilling to collaborate or follow shared methods.

7. Projektstrategi

Gør rede for hvordan man finder frem til en strategi for ens projekt, eks. Forklar hvordan man anvender et såkaldte TURBO-skema under udarbejdelsen af en projektstrategi.

Indledning

- Projektstrategien er den overordnede løsning på projektetsopgave.
- Strategien er den røde tråd, den bærende ide, i projektplanen.
- Strategi: Overordnet fremgangsmåde i et projekt.
- Projektstrategien er projektlederens middel til at få overblik over planen.
- Projektdeltagerne får et bedre overblik igennem dette.
- Opgave: Et projekts mål og vilkår (De forhold som kan gøre projektet svært)
- Vilkår: Et projekts betingelser og ressourcer

Projektstrategi

Projektstrategis forhold.

- Der skal laves en projektstrategi over
 - Ikke fastlagte forhold
 - Udviklere
 - Teknik
 - Åbenhed for omverden
 - Release-metode
- Én samlet strategi, ikke flere delløsninger hvori ”strategi” indgår
 - Hverken store eller små problemer skal løses med en ny strategi
 - Projektet skal samles under en stor koordineret strategi
- Eksempler på strategier:
 - Eksperimenter til projekter med stor usikkerhed, for at danne et større overblik
 - Hvis der er rigtig mange involveret, kan stram styring være en strategi
 - Hvis målet er et at blive hurtigt færdigt, kan strategien være at tilpasse et standardsystem
 - Hvis der ingen særlige problemer er, kan strategien være at blive hurtig færdig
- Strategiske beslutninger skal tages så styrker udnyttes og der koncentreres om de sværreste problemer
 - ”En situationsbestemt strategi, der sigter på at udnytte muligheder for succes i et projekt”
- Projektlederen
 - Han/hendes middel til at få overblik over planen
 - Klar formulering = bedre muligheder til videregiver overblik = lettere at planlægge, da projektdeltagere kan forholde sig til projektlederens hensigt
- Hvis man ingen svagheder og styrker har, er det oplagt at sige strategien er hurtig færdig, så man undgår der kommer problemer.
- En strategi er vigtig til udviklingsprojekter, da det hjælper med at løse en ny opgave bedst muligt, da kombinationen af vilkår og mål er ny og uprøvet.
- Alle projekter støder på problemer, og vi kan ikke forvente at de løser sig selv, og derfor er det vigtigt, at man på forhånd har taget højde for styrker og svagheder, for at kunne udarbejde en strategi der er tilpasset til ens projekt, for at imødegå kommende problematikker

- Et projekt uden en strategi, er et projekt hvor der ikke er taget højde for de væsentlige problemer og hvordan de eventuelt skal løses
- Dette kan ende i brandslukning (brug af hjælp af f.eks. leder), der er en meget uprofessionel løsning på problemer
- For at bestemme sin strategi, kan man gøre brug af et TURBO-skema, der skal hjælpe en på vej med at definere sin strategi. TURBO står for: Teknikken, Udviklerne, Resultatet, Brugerne og Omgivelserne.
- Man skal kun vælge de styrker og svagheder som 'vejer' mest. De er de vigtigste.
 - Generelt går TURBO ud på:
 - at analysere særlige mål og vilkår
 - Overvej strategiske beslutninger
 - Afvej svagheder
 - Sammenfat strategien

Mål og vilkår	Særlig styrke	Særlige svagheder	Mulige beslutninger (strategielementer)
Teknikken			
Udviklerne			
Resultatet			
Brugerne			
Omgivelserne			
Projektets strategi begrundet i projektets særlige mål og vilkår			
Sammenfatning			

- Analyser mål og vilkår ved at stille spørgsmål som:
 - Hvad er særlig svært eller særlig usikkert ved teknikken, udviklerne osv.
 - Hvad er den særlige styrke ved disse?
 - Overvej til hvert problem hvilke modtræk projektet kan tage for at imødegå disse
 - Overvej samtidig for hver styrke, hvordan den aflaster projektet og hjælper det på vej
- Hvis ikke dit svar passer ind i en af boksene, så lav en ny for at tilpasse den til lige netop jeres projekt
- Afvej svagheder
 - Udvælg de største problemer
 - Her er det vigtigt at man kun tager reelle problemer, og udelader eventuelt klynk
- Sammenfat strategien
 - Udvælg strategielementer blandt de løsninger der er 'tungest'
 - Strategielementerne skal udvælges, så de styrker hverandre og løser opgaven.
 - Rediger selve formuleringen af strategien så den lever op til de 4 krav.
Kig ved kendtegnet ved en god strategi forneden.
- Udvælg strategi elementer der tilpasser sig problemerne, og er med til at løse dem
 - Strategielementerne er strategiens byggesten
 - En god strategi er kendtegnet ved:

- Enkelt formulering, 3-5 slagord
 - Logisk svar på projektets mål og særlige vilkår
 - Skal mest bygge på kendte og velprøvede elementer
 - Strategien skal ligge inde for projektets formåen

- De fem mål og vilkår:
 - Teknikken
 - Teknisk platform ny/gammel eller unik platform
 - Nye værktøjer? Kan f.eks. være en svaghed, da I skal lære noget nyt.
 - Udviklere:
 - Skal der hentes brugere udefra?
 - Kræver det specialister
 - Hvordan sørges der for godt samarbejde?
 - Resultattype:
 - Hvilket resultat?
 - Outsource eller forbedre virksomheden f.eks.
 - Hvordan skal resultatet implementeres?
 - Trinvist
 - På én gang
 - Eksperimenter er altid et oplagt strategielement
 - Ved stor usikkerhed er åbenhed over for nye forslag fra brugerne vigtig at fastholde gennem hele projektet
 - Brugerne:
 - Graden af brugerinddragelse
 - Kan både forbedre og forsinke et projekt(F.eks. måske kan de ønske for mange features, men samtidig komme med gode ideer.)
 - Det er vigtigt at udviklerne hurtigt får opbygget tillid til brugerne, ellers så finder brugerne nogle nye
 - Projektets omgivelser:
 - Åbent/lukket projekt? I nogle tilfælde kan det være godt at projektet er hemmeligt, f.eks. vi man snakker om noget vedr. landets sikkerhed.
 - Prøve at undgå at projektet er afhængigt af mange andre projekter
 - Kan være svært at lave et sundt projekt
 - Magtfulde interesser kan have unrealistiske forventninger
 - Stærkt modstridende interesser om projektet
 - Her er det projektlederens job at få opbygget en tillid og goodwill
 - Få eksterne konsulenter til at vurdere projektet for at bygge større troværdighed omkring projektet
 - Goodwill kan skabes ved at lave små hurtige og synlige successer.
 - Tiden:
 - Det kan være en fordel at arbejde hurtigt
 - Udarbejde en prototype

- Brug TURBO i udviklingsforløbet
 - Først opbygge en strategi når du har noget at arbejde med.

- F.eks. skal det ikke laves i starten i forundersøgelsen, for der er du ikke sikker på du har et konkret produkt. Man ved nemlig ikke, i starten af forundersøgelsen om det er nødvendigt med et produkt.

Bilag

Mål og vilkår	Særlig styrke	Særlige svagheder	Mulige beslutninger (strategielementer)
Teknikken		Ny teknologi	Hurtig adgang til konsulenter
Udviklerne	Entusiasme		Hurtige delresultater kan bevare entusiasmen
Resultatet	Lille, overskueligt system	Ønske om høj kvalitet	Kvalitet øges med dygtige folk og ekstern kvalitetsvurdering
Brugerne		Ingen ægte brugere at snakke med	Studere lignende systemer Finde nogle brugere
Omgivelserne		Risiko for afbrydelse p.g.a. andre opgaver	Trinvist forløb, modulært design Hurtigt færdig
Projektets strategi begrundet i projektets særlige mål og vilkår			
Sammenfatning	<p>Projektet skal have et trinvist forløb. For det første for hurtigt at prøve hele den nye metode, for det andet for at kunne justere ambitionsniveauet i tilfælde af afbrydelser, for det tredje for at bevare energien i projektet.</p> <p>Projektet skal være udadvendt. Dels skal vi have inspiration og hjælp, så vi kan lære noget. Dels skal vi have kritik, så vi sikrer en høj kvalitet.</p>		

8. Test i systemudvikling

Gør rede for hvilken rolle test har i en agil systemudviklingsproces, det kan være test i eks. XP. Hvad er forskellen på testens rolle i et agilt udviklingsprojekt kontra test i ikke agile metoder eks. UP.

Du er velkommen til at drage erfaringer med test ind fra jeres eget projekt.

Indledning

- Man kan teste på forskellige måder. Hver metode kan have deres egen 'præference' på hvordan de helst vil teste.
- Test i systemudvikling handler om at finde frem til hvad det rigtige system er. Det rigtige system skal kunne leves op til kundens opstillede krav til funktionalitet, data og kvalitet.
- Vi tester dermed om systemet lever op til kravene. Når vi laver tests så gør vi det fordi for at sikre kvalitet. Kvalitet er bl.a. udseende, performance, holdbarhed, brugervenlighed, korrekthed
- Vi tester for at se om f.eks. en metode virker, for at sikre man ikke støder ind i problemet senere.
- **Formål med test**
 - Fejlfinding
 - Dokumentation af kvalitet
 - Leverede op til det specificerede(Det kunden ønsker)
 - Virker efter hensigten

Operationelle

- Her udføres det der skal testes
- Hvordan er testene gået
- Er den godkendt?
- Lav fejlrappport til udviklere
- Testteknikker
- Whitebox
 - Ser på strukturen inde i programmet
 - Selve koden
 - Man kan se på om en kode kører for langt osv.
 - Der er nogle dækningskriterier for whitebox
 - Statementdækning (Statements skal gennemløbes mindst 1 gang, hvis 100% dækning)
 - Forgreningsdækning (If/else, switch/case, for og while identificeres og testes)
 - Stidækning(Stier skal gennemløbes og testes. F.eks. hvis en metode kalder har en if og else sætning som kalder på to forskellige metoder i hver, så skal denne dækning teste begge 'stier')
- Blackbox
 - Man ved ikke hvilket kode programmet indeholder, man ved kun det input og output der kommer ud.
 - F.eks. når du bruger java biblioteker, du ved ikke indholdet af koden men ved hvordan inputtet og outputtet skal være.

- Du kender kun specifikationen, så du ved hvad den skal kunne ku' men ikke koden.
Så du ved pre og post betingelser.
- Man skal identificere gyldige og ikke-gyldige værdier(gyldige er positive tal, for ugyldige er negative)
- Ækvivalensmængder/klasser
- Man skal kun teste en case af gangen. F.eks. prisen for en voksen er en case, og rabatten for en voksen er en case.
- Graybox
 - Blanding af white og blackbox test
 - Man kender til koden
 - Det er en blackbox test, men testcases er lavet af folk der kender til den indre struktur.

Testens rolle i XP

- Sikre kvalitet samt at der dokumenteres for fremgang. Det bruges egentlig til at se om der sker fremgang.
- Nogle af de praktikker i XP er
 - Continuous Integration
Man skal integrere og teste systemet løbende. F.eks. hver anden time. Derfor er det vigtigt med ten-minute build, fordi når man integrerer systemet skal det testes.
 - Test-first(Man laver testcases til automatisk test(unit tests) før programmeringen påbegyndes)
Testen fejler indtil programmet er lavet^[SEP]
 - Hvorfor tests first?
Man laver ikke overflødig kode, man ved hvor man arbejder henimod Hurtigere at kode systemet
Du ved hvad funktionaliteten kræver, hvis du laver tests først, da det giver dig et overblik over kravet for funktionen.
Sparer dig tid, hvis du laver testene først frem for senere.
Hurtig feedback fordi du hurtigt kan teste, du kan se om du er på rette vej.
 - Ten-minute build. Testen skal være hurtig til at afvikle. At builde systemet og køre en automatisk test må ikke tage mere. Det skal være sådan fordi man skal teste flere gange, og hvis det er langsomt kan det være man ikke får det gjort.
- Nævn trekanten i XP
- Det er mere oplagt at test first i agile metode fordi man løbende laver testene i hver iteration. Hvis man lavede test first i vandfaldsmodellen kunne det f.eks. være man lavede en masse test som man senere ikke havde brug for fordi kravene har ændret sig, så laver man en masse unødvendig kode.

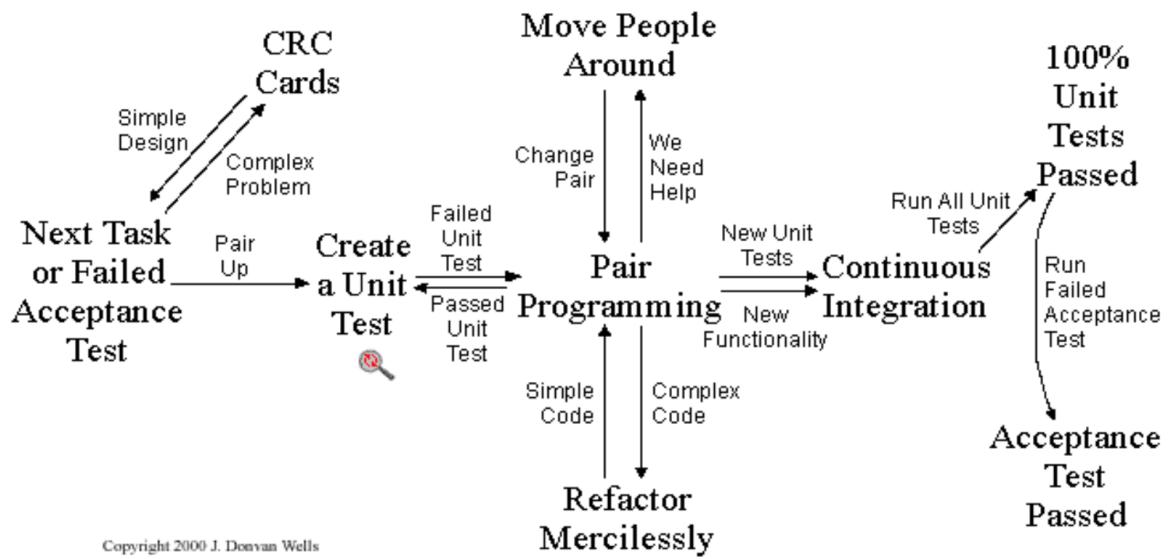
Agile test vs. plandrevne

- XP test er bygget op omkring test, du tester hele tiden fordi du er først færdig når din du har fået accepttest.
- Man tester løbende i XP, men



Collective Code Ownership

Zoom Out



Continuous integration betyder Hvis vi laver en sinus knap,
så skal den integreres med GULen med det samme.
Man skal integrere med det samme, så folk kan huske hvad man
har lavet osv.

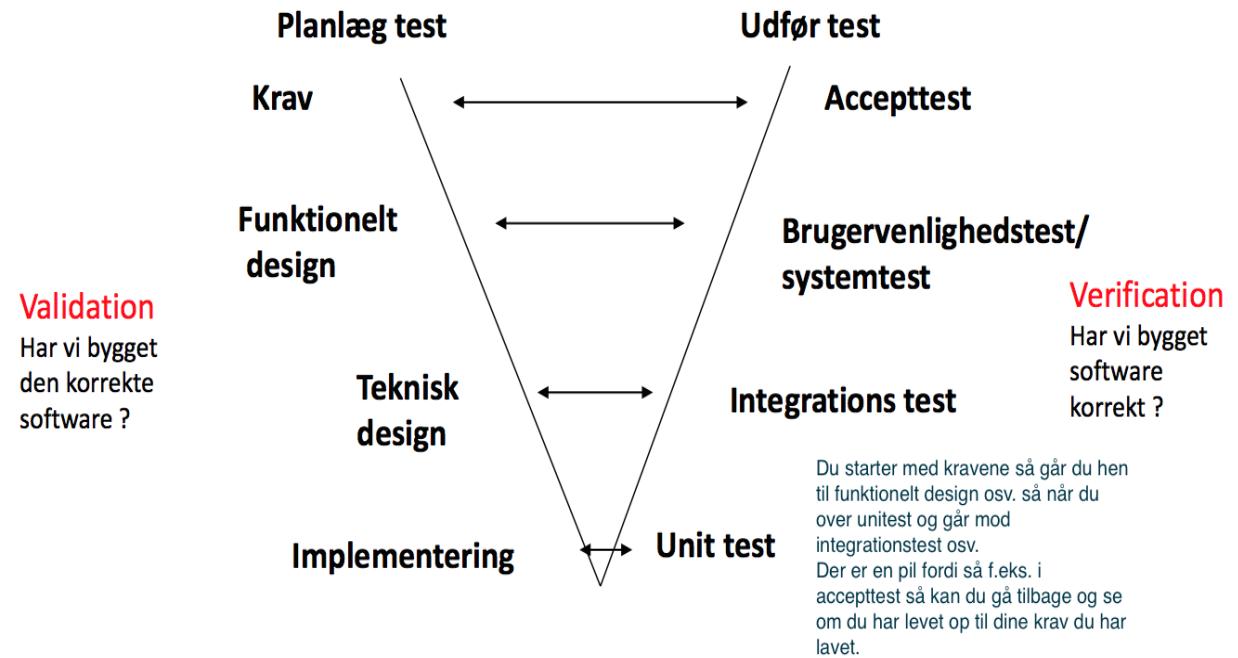
Efter man har lavet tests first, kan det være at man efter vil lave flere tests det er derfor der
står 'new unit tests' og så kører man alle unit tests.

Bilag

V-model

V-model

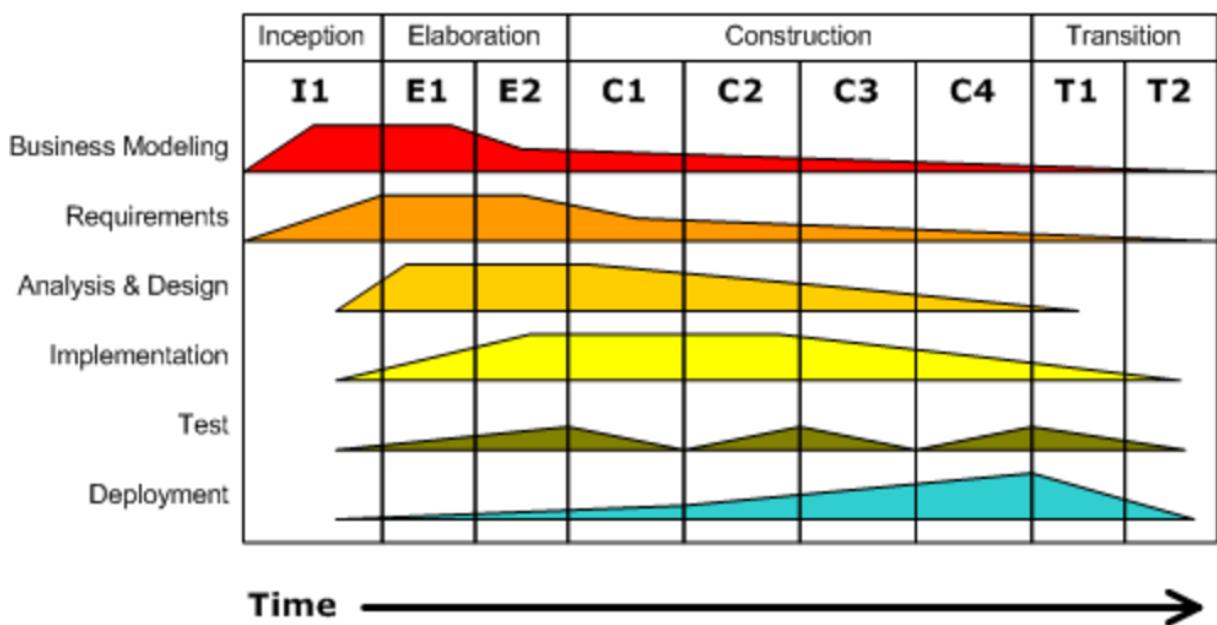
V-modellen er en testmodel, der er udviklet til den traditionelle fase model!



Test i UP

Iterative Development

Business value is delivered incrementally in time-boxed cross-discipline iterations.



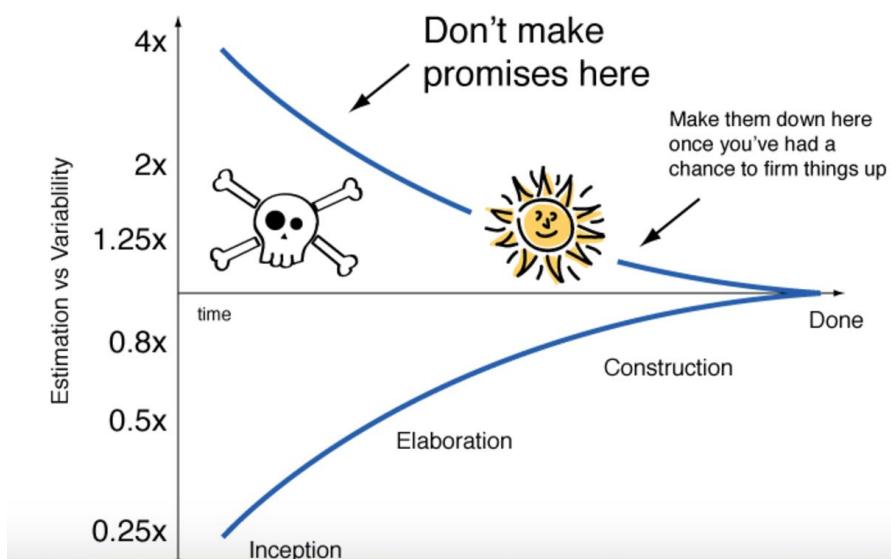
9. Estimering

Beskriv hovedideerne i estimering i forbindelse med projektstyring. Forklar hvilken rolle estimering har i en agil udviklingsproces med XP og/eller SCRUM. Hvilken estimeringsteknik bruges til estimering i XP og/eller SCRUM.

Indledning

- Estimat, betyder egentlig forudsigelse af et projektets forbrug af personressourcer.
- Formålet med estimering er at sikre at planen er realistisk med hensyn til personressourcer
- Man kan sige at det er et kvalificeret gæt på tidsforbrug.
- Estimerer kræver erfaring, fra tidlige projekter.
- Estimatet er vigtigt fordi markedet ofte fast tid og fast pris
- Estimerings er erfaringsbaseret
- Hvorfor er estimerer vigtigt?
 - Omgivelserne har ret til et tidligt estimat, markedet kræver ofte fast tid og fast pris.

Cone of uncertainty



Estimering bliver nemmere og nemmere jo længere du er i projektet.
Ikke lov noget for tidligt

Estimering i XP

- Estimering spiller en stor rolle i XP. Hver user story skal estimeres og hver iteration fylder antal units.
- De skal kunne håndtere forandring, så er det vigtigt at lave en hurtig estimering på hvor langtid en opgave vil tage. Du kan ikke lave et samlet estimat fordi der hele tiden kan komme nye user stories.

- Så man kan sige at der skal udføres 30 units(pga. ens velocity) pr. iteration, så skal man sørge for at have 30 units af user stories. Fordi så ved man hvor mange user stories man kan udføre pr iteration.
- Man bruger planning poker fordi det er en hurtig estimeringsmetode samt at den bliver forankret mellem gruppemedlemmerne fordi alle kommer med inputs og alle er indforstået med det.
- Estimater kan dog afvige, så man benytter sig af yesterday's weather, hvor man ændrer ens velocity efter hvad man nåede sidste iteration.
- I XP regulerer man omfanget, så man fjerner user stories som f.eks. ikke er så nødvendige i første release.
- Man estimerer ved hjælp af planning poker. Fortæl om jeres projekt

Estimering på flere måder

- Estimer størrelsen af systemet
 - Linjer kode
 - Antal use case points
 - Antal story points
- Estimér indsats
 - Mande måneder(f.eks. hvor mange personer er der hver måned til at arbejde på projektet som tager 12 mende(en person)-måneder og hvornår vil projektet være færdigt med de antal medarbejder. Så f.eks. være 1,3 mande måneder med 9 arbejdere.)
- Estimer tiden
 - Måneder
- Fortæl derefter om andre estimeringsmetoder. F.eks. use case points

10. Estimering

Gør rede for hovedideerne i estimering i forbindelse med projektstyring. Vælg en af estimeringsteknikkerne use case points, trepunktstimering, eller cocomo og beskriv principperne i den. Kender du andre estimeringsteknikker?

Til dette spørgsmål er det ikke nødvendigt at kunne alle udregninger, men derimod skal man vide hvad de forskellige ting går ud på og hvilken effekt de har til sidst for UCP beregningen.

Indledning

- Estimering skal ses som en ruteplan forstået på den måde at det er en plan for hvor lang tid projektplanen tager.
- Altså hvor lang tid tager det at kommer fra A til B.
- Kort sagt får vi en varighed om hvor lang tid vores rejse/projekt ca. kommer til at tage.
- Estimat, betyder egentlig forudsigelse af et projektets forbrug af personressourcer.
- Formålet med estimering er at sikre at planen er realistisk med hensyn til personressourcer
- Man kan sige at det er et kvalificeret gæt på tidsforbrug.
- Estimater kræver erfaring, fra tidligere projekter.
- Estimatet er vigtigt fordi markedet ofte fast tid og fast pris
- Der er forskellige måder, hvorpå man kan estimerer.

Use case points

- Mere moderne og passer bedre til den objektorienterede verden
- Use case point omhandler at man prioriterer at estimere hvor lang tid ens use case kommer til at tage.
- Man ønsker at gøre det for at kende hvor lang tid en del opgave af projektet kommer til at tage.
- Inden for UCP er der 3 vigtige faktorer: UCCP(Utilpassede Use case Point), TCF(Teknisk kompleksitets faktor), ECF(Omgivelsernes kompleksitets faktor).

Kort opsummering af use case points

- I use case points, kigger man på hvor komplekse use casene og aktørerne er. Man kigger også på de tekniske forhold samt omgivelsernes forhold.
- Man skal vurder om use casene er komplekse, f.eks. ved at kigge på hvor mange transaktioner der er. Man kigger også på hvor komplettest use casen er ved at kigge på hvordan use casen bliver tilgået. F.eks. gennem grafisk grænseflade.
- Man kigger så på de tekniske forhold, f.eks. er det nemt at bruge? Er det nemt at ændre?
- Man kigger også på omgivelserne, f.eks. udviklernes erfaring. Vægter mere end de tekniske
- Ud fra det regner man så kompleksiteten sammen, hvor man så får resultatet om hvor langtid det tager at lave alle use casene, i timer.

Formel for udregning:

- UCP = UUCP*TCF(Teknik)*ECF(Omgivelser)...
- Timeforbruget udregnes som følgende: ESTIMAT = UCP * PF

UCCP(Utilpassede Use case Point)

- UUCP findes ved at lægge UUCW(Utilpasses use case vægtning) og UAW(Utilpasset aktør vægtning) sammen.
- UUCW tager sit udgangspunkt i antallet af aktiviteter/punkter en use case består af. Så altså jo flere task en use case har jo større er kompleksiteten af den givende use case.
- UUCW laves i et skema som sket nedenfor.

Niveau	Beskrivelse	Point	Antal Use Cases	Sum P * Antal
Simpel	Simpelt brugerinterface. Involverer kun en enkelt database tabel. Består af mindre end 3 aktiviteter og kræver implementering i mindre end 5 klasser.	5	6	30
Middel	Mere interface design. Involverer 2 eller flere database tabeller. Består af mellem 3 og 7 aktiviteter og kræver implementering i 5 – 10 klasser.	10	8	80
Kompleks	Kompleks brugerinterface eller kompleks anvendelse af produkt. Involverer 3 eller flere database tabeller. Mere end 7 aktiviteter og kræver implementering i mere end 10 klasser	15	4	60
UUCW Total				<u>170</u>

Bemærk: De viste antal use cases er udelukkende eksempler

- Der findes kategorier en use case kan blive placeret i, når man taler om UUCW. Her er taler om Simpel, Middel og Kompleks.
- Man finder frem til den endelige UUCW værdi ved at kategorisere use casene og multiplicere antallet af use cases i den enkelte kategori, med pointene. Værdierne lægges så sammen. UUCW er så værdien af summen af samtlige point.
- UAW omhandler den samlede kompleksitet af de involverede aktører. Den værdi findes på nogenlunde samme måde som UUCW findes på.
- Her er det bare aktørerne der er i fokus. De kategoriseres efter kompleksitet.
- De får henholdsvis 1,2,3 point. Herefter multipliceres antallet af aktører i hver kategori med pointene og de point bliver så lagt sammen og udgøre så UAW værdien.

Kategorierne:

- 1. (1 point) Simpel
Aktøren repræsenterer et andet system, med et veldefineret interaktions interface
- 2. (point) Middel
Aktøren repræsenterer et andet system som tilgås via en protokol(tcp eller internet protokol)
- 3.(3 point) Kompleks
Aktøren er en person der tilgår produktet via et brugerinterface.

TCF(Teknisk kompleksitets factor)

- Måden man finder TCF på er ved at evaluerer 13 tekniske områder.

Technical Factor	Description	Weight	Perceived Complexity	Calculated Factor (Weight * Perceived Complexity)
T1	Distributed System	2	1	2
T2	Performance	1	3	3
T3	End User Efficiency	1	3	3
T4	Complex Internal Processing	1	3	3
T5	Reusability	1	0	0
T6	Easy to Install	0.5	0	0
T7	Easy to Use	0.5	5	2.5
T8	Portable	2	0	0
T9	Easy to Change	1	3	3
T10	Concurrency	1	0	0
T11	Special Security Features	1	0	0
T12	Provides Direct Access for Third Parties	1	3	3
T13	Special User Training Facilities Are Required	1	0	0
Technical Total Factor				19.5

Afvejning(Weight)

- Afvejningen er et udtryk for den relative indflydelse på kompleksiteten.
- F.eks. er værdien 2 et udtryk for at det pågældende område har en stor effekt på kompleksiteten.
- 0.5 er udtryk for en meget lille effekt på kompleksiteten. Værdierne er konstante.

Opfattet kompleksitet(Percieved complexity)

- 0.5 er udtryk for en meget lille effekt på kompleksiteten.
- Værdierne er konstante
- Projektgruppen udfylder skemaet med disse værdier mellem 0-5. 0 = ligegyldigt, 1=lav kompleksitet, 3= middel, 5=høj kompleksitet. 3=Tvivl
- Værdierne for afvejning og den opfattede kompleksitet multipliceres og de 13 resultater lægge sammen.
- Resultatet er et samlet udtryk for ”Den tekniske kompleksitet”
- ”. Den tekniske kompleksitets faktor(TCF) er et udtryk for om teknikken øges eller mindske tidsforbruget i gennemførelsen af projektet.”
- Det er fastlagt værktøj af tiden maksimalt kan øges med 30% og mindskes med 40 %.. Derfor findes der 2 formler(En tilnærmedesvis og en nøjagtig) som sikrer at TCF værdien ligger mellem 0.6 og 1.3.
- $TCF = 0.6 \text{ eller } 1.3 + (0.01 * TTF)$
- Hvis den endelige værdi overstiger 1 er projekt teknologisk kompleks, under 1 er det modsatte.

Omgivelsernes kompleksitets faktor(ECF)

- ECF går ud på at vurderer en omgivelser.
- Omgivelser kan nemlig have ligeså stor effekt som det tekniske punkt når det kommer til at estimere ens projekt.

Environmental Factor	Description	Weight	Perceived Impact	Calculated Factor (Weight*Perceived Complexity)
E1	Familiarity With UML	1.5	5	7.5
E2	Part-Time Workers	-1	0	0
E3	Analyst Capability	0.5	5	2.5
E4	Application Experience	0.5	0	0
E5	Object-Oriented Experience	1	5	5
E6	Motivation	1	5	5
E7	Difficult Programming Language	-1	0	0
E8	Stable Requirements	2	3	6
Environmental Total Factor				26

Afvejning(Weight)

- Negativ tal har en stor effekt på UCP endelige værdi.
- Afvejning betyder, hvor meget betydning det har.
- Værdierne kan være konstante, men som tidligere med TCF er vurderingen helt oppe til projektgruppen.

Opfattet indflydelse(Percied impact)

- Projekt gruppen fastsætter værdierne, ud fra deres holdninger.
 - Værdien 5 = stærk positiv indflydelse. 1 = negativ indflydelse.
 - Måden hvor man finder ECF er den samme som TCF beregningen.
 - Produktet af de 8 forhold lægges sammen og udgører den "Totale omgivelses faktor".
 - Desto højere ECF værdi, desto bedre er det, eftersom høje værdi øger muligheden for at opnå en positiv indflydelse på den samlede UCP beregning.
 - For at kunne bruge UCP til estimering, mangler vi at finde PF. PF er baseret på erfaring fra tidligere projekter.
- Udregning:
- PF = timer/UCP ← Timer = hvor lang tid de tidligere projekt tog. UCP = UCP værdien for det tidligere projekt.
 - Estimat = UCP*PF

Trepunkttestimering(Succesiv kalkulation):

- **Estimer 3 tal for hver aktivitet**
 - Den korteste tid, man forstiller sig, det kan laves på.
 - Den mest sandsynlige tid
 - Den længste tid man forestiller sig det kan laves på.
- Jo større afvigelser der er mellem den korteste og den længste jo større risiko er der for mangler at identificere væsentlige risici.
- Herefter beregner man middel værdien($m = (a+3b+c)/5$), Standardafvigelse($s = (c-a)/5$)
Hvis standardafvigelse er stor, indebærer det en stor usikkerhed i forhold til projektet.

Aktivitet	a	b	c	Trepunktsestimering		Reelt forbrug
				m	s	
Lav analyse UML diagrammer	50	75	100	75	10	100
Osv.	Osv.					

Eks. på dokumentation af estimater fra en trepunktsestimering

Trepunkttestimering

- Opdel faser i aktiviteter
- Vurder middelværdi og usikkerhed for hver aktivitet

- For de aktiviteter, hvor usikkerheden vurderes at være for høj fortages(Standardafvigelse og varians høj)en nedbrydning i opgaver – og for hver opgave gentages vurdering(herfra betegnelsen successiv(følgende efter hinanden)

Cocomo – COnstructive COst MOdel:

- Det er en algoritmisk model
- Man indsætter så programstørrelsen og faktorer i en formel hvorved man får den forventede indsats der kræves i projektet, målt i personmåneder.

Kort opsummering af Cocomo

- Man har 5 skalafaktorer og nogle indsats faktorer.
- Hvor god kender din gruppen opgaven?
- Er kendskabet højt, så det f.eks. 0, hvis det lavt så f.eks. 5.
- Kendskab, hvor tit har vi lavet sådan noget før.
- Fleksibilitet, skal vi tilpasse os stramme krav og eksterne grænseflader.
- Afklarethed, hvor godt er de mulige risici afklarede
- Projektgruppen, hvor godt sammentømret er projektgruppen
- Modenhed, hvor god en udviklingsproces har vi
- Disse værdier skal man vurder fra 0-8
- Man har så nogle indsatsfaktorer, som man vurder fra 0,73-1,74. Man kigger på hver, hvilke krav der er til systemet samt andre ting som programmeringsevnerne . Man kigger på selve opgaven og ikke gruppen som i skalafaktorer.
- Indsatsfaktorer multipliceres.
- Man skal også kigge på linjer af kode(Man estimere kodestørrelsen på grundlag af en nedbrydning af kravspecifikation i mindre dele og altså estimere de enkelte dele hver for sig, for så at summere dem til sidst.)
- Ud fra det regner man ud hvor mange personmåder man skal bruge på projektet.
- Den tid der kræves afhænger ikke af antal personer på projektet men af den samlede indsats
- Ofte starter man med relativt få folk, bygger op til mange hvorpå man begrænser antal personer igen
- Man bruger så nogle konstanter som de har fået fra erfaringer fra andre projekter, som de så har lavet en algoritmisk model, til at udregne estimerater.

COCOMO II

- $PM(\text{Personmåned}) = 2,94 * \text{KSLOC}^E * M$
- $E = 0,91 + 0,01 * \sum Sf_i$
- $Sf_i (i=1..5)$ er de 5 skalafaktorer
- KSLOC = produktstørrelse i kodelinjer/1000(Kilo source Lines of Code)
- M = indsatsfaktorer multipliceret
- $TDEV(\text{Udviklingstiden i kalendermåned}) = 3,67 * PM^F$
 $F = 0,28 + 0,2 * 0,01 * \sum Sf_i$
3,67 osv. er teknologiske konstanter.
- Altså den tid der kræver afhænger ikke af antal personer på projektet men af den samlede indsats.

- Ofte starter man med relativt få folk, bygger op til mange hvorpå man begrænser antal personer igen.

COCOMO II (effort)

$$PM = 2,94 \times KSLOC^E \times M$$

$$E = 0,91 + 0,01 \times \sum_{i=1}^5 SF_i$$

$$M = \prod_{i=1}^{17} EM_i$$

SF=Skalafaktorer; EM = Indsatsfaktorer

De 5 skalafaktorer

- Værdier = 0-8
- Er kendskabet højt, så det f.eks. 0, hvis det lavt så f.eks. 5.
- Kendskab, hvor tit har vi lavet sådan noget før.
- Fleksibilitet, skal vi tilpasse os stramme krav og eksterne grænseflader.
- Afklarethed, hvor godt er de mulige risici afklarede
- Projektgruppen, hvor godt sammentømret er projektgruppen
- Modenhed, hvor god en udviklingsproces har vi

Andre estimeringsteknikker

- Planning Poker
- Analogi teknikken(Sammenligning af projekter med et lignende projekt. Indgår i en grad i alle andre estimeringsteknikker(Kvalificeret gæt))

Bilag

COCOMO

Indsatsfaktorer

- Indsatsfaktorer (COCOMO II)
 1. Krav til systemets pålidelighed under drift
 2. Størrelsen af databasen
 3. Systemets kompleksitet
 4. Udvikling til genbrug
 5. Dokumentationsbehov
 6. Krav til hastighed under drift
 7. Krav til centrallager under drift
 8. Hyppighed af ændringer i den tekniske platform
 9. Ventetid på resultater af testkørsler
 10. Analytikernes evner
 11. Udviklernes erfaring med forretningsområdet
 12. Programmørernes evner
 13. Udviklernes erfaring med den tekniske platform
 14. Udviklernes erfaring med udviklingsmiljøet
 15. Anvendelsen af moderne udviklingsmetoder
 16. Tilgængelighed af programudviklingsværktøjer
 17. Krav til afleveringstidspunkt

AARHU

Eks. indsatsfaktorer i COCOMO

Indsatsfaktorer (0,73-1,74)

Faktor	Navn	Meget lav	Lav	Normal	Høj	Meget høj	Ekstra høj
RELY	Pålidelighed	Let besværligt	Let at håndtere	Moderat besvær	Større økonomisk skade	Risiko for menneskeliv	-
		0,75	0,88	1	1,15	1,40	
DATA	Databasestørrelse	-					-
			0,94	1	1,08	1,16	
CPLX	Produktkompleksitet	Meget simpel kode	Simpel kode	Moderat kode	Ret kompleks kode	Meget kompleks kode	Yderst kompleks kode
		0,75	0,88	1	1,15	1,30	1,65
RUSE	Genbrug	-	Ingen	Inden for projektet	Inden for lignende projekter	Inden for ensartede projekter	Generel brug
			0,89	1	1,16	1,34	1,56
DOCU	Dokumentation	Meget lavt behov	Behov for basal dokumentation	Middel behov	Stort behov, også bagefter	Meget stort behov	-
		0,85	0,93	1	1,08	1,17	
TIME	Udførelsestid	-	-	50 % af tilgængelige ressourcer (ingen specielle krav)	70 %	85 %	95 %
ACAP	Udvikleres evner	Ret lav	Lav	Middel	Høj	Meget høj	-
		1,50	1,22	1	0,83	0,67	
PCAP	Programmørers evner	Ret lav	Lav	Middel	Høj	Meget høj	-
		1,37	1,16	1	0,87	0,74	
PCON	Gruppens stabilitet	48 % udskiftning / år	24 % / år	12 % / år	6 % / år	3 % / år	-
		1,26	1,11	1	0,91	0,83	

Værdierne for alle indsatsfaktorerne ganges sammen og ganges på PM for at få den justerede faktor –

Faktor > 1 – mere tid

Faktor = 1 – normale vilkår

Faktor < 1 – mindre tid

Obs. Flere faktorer kan ses i tabel på opgavesedlen!

Eks. de 5 skalafaktorer

Skalafaktorer

Faktor	Navn	Meget lav	Lav	Normal	Høj	Meget høj	Ekstra høj
PREC	Kendskab	Ingen	Stort set ingen	Nogen	Rimeligt velkendt	Ret stort	Meget grundigt
		6,20	4,96	3,72	2,48	1,24	0
FLEX	Fleksibilitet	Praktisk taget ingen	En smule fleksibel	Rimeligt fleksibel	En del fælles normer	Visse fælles normer	Enighed om mål alene
		5,07	4,05	3,04	2,03	1,01	0
RESL	Afklarelighed	Mange store risici	En ret stor mængde risici	En del risici	Generel god forståelse	Stort set forstået	Total forståelse
		7,07	5,65	4,24	2,83	1,41	0
TEAM	Samarbejde	Gennemgribende problemer	En hel del problemer	Basalt samarbejde, mindre problemer	Solidt samarbejde	Meget høj grad af samarbejde	Den gode gruppe
		5,48	4,38	3,29	2,19	1,10	0
PMAT	Modenhed	CMM, niv. 1 (lav)	CMM, niv. 1 (høj)	CMM niv. 2	CMM niv. 3	CMM niv. 4	CMM niv. 5
		7,80	6,24	4,68	3,12	1,56	0

11. Produkt og proces i systemudvikling

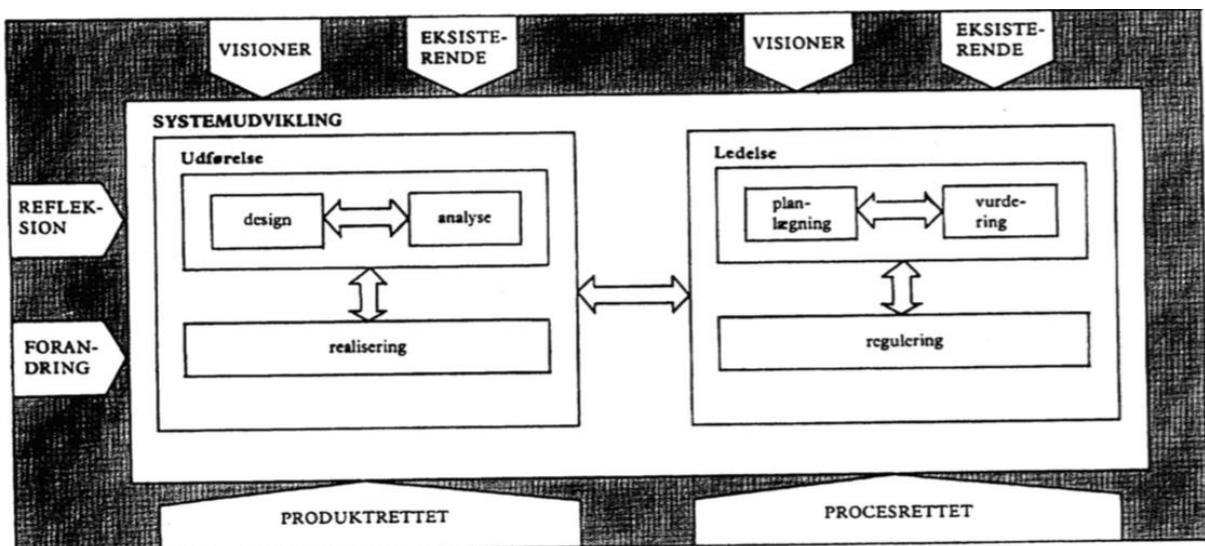
Gør kort rede for, hvad begreberne produkt og proces dækker over i systemudvikling, forklar hvordan der er sammenhæng mellem kvalitet i produkt og kvalitet i proces. Tag evt. udgangspunkt i figuren nedenfor. Kom evt. med eksempler på, hvad der er produkt og proces i forhold til en metode, du kender.

Indledning

- Produkt definition: Det endelige system med tilhørende dokumentation samt mellemresultater, der laves med direkte henblik på at nå frem til slutresultat.
- Proces i systemudvikling: Den samlede arbejdsproces med at udvikle produktet. Det kunne f.eks. være projektplaner. Processen har interne og eksterne sider. Den eksterne handler om styringen af processen. Den interne proces er om udviklernes tilfredshed, personligt og socialt i gruppen.

Sammenhængen mellem produkt og proces

- Produkt og proces har 3 under aktiviteter.
- For produktarbejde drejer det sig om analyse, design og realisering
- For proces drejer det sig om vurdering, planlægning og regulering.
- Der skal være tæt sammenspil mellem de udførende (produktrettede) og ledende aktiviteter (procesrettede)
- F.eks. kan du ikke lave en plan uden at kende til de produktrettede forhold ved projektet. Ligeledes kan man ikke lave et produkt uden at kende til de procesrettede forhold.
- Produkt og proces er afhængige af hinanden, du kan ikke lave et godt produkt uden en god proces, ellers skal man være heldige.
- Du kan ikke lave en proces uden et produkt og omvendt.



- Der er dobbeltrettede pile i figuren fordi at ordentlig systemudviklingsarbejde er en iterativ proces. (Kan ses på udviklingsmetoden Unified Process)
- Visioner i figuren, er hvordan vi forestiller os at systemet skal være

- Eksisterende er de systemer der allerede findes.

Proces i figuren

- I et nystartet projekt, er, at projektlederen planlægger projektets forløb.
- **Planlægningen** bygger på den smule kendskab man har til produktet i denne periode.
- Senere når man møder planlægge milepæle, skal projektlederen **vurdere** produktets tilstand. (Har man nået de forventede resultater? Er kvaliteten den man ønskede? osv)
- Hvis vurderinger ikke svarer til det forventede, kan projektlederen gøre ind og **regulere** på forløbet, ved f.eks. at tilføre flere ressourcer, ændre på planen osv,
- Tilslut kan det være en ide at lave en samlet vurdering i form af en procesrapport.

Produkt i figuren

- Man laver først analyse f.eks. analyseklasse så designklasse og derefter realiserer man det.

Kvalitet

- Kvalitet er et individuelt begreb, afhængig af oplevelse
- F.eks. kunne jeg synes at kvalitet i en termokande kan f.eks. være at den er tæt, og den holder varmen samt at den ikke ruster. Det kunne også være robusthed. Kvalitet i proces kunne også være at om termokanden er produceret på en miljøvenlig måde.
- Indenfor kvalitet snakker man om kvalitetsstyring
 - Kvalitetsmål, for at aftale og dokumentere præcise forventninger
 - Kvalitetssikring, for at indbygge kvaliteten i løsningerne
 - Kvalitetskontrol, for at måle hvor godt det lykkedes
 - Kvalitetsforbedring, for at løbende at øge produktiviteten og kvaliteten. (Proces)
- Kvalitetsforbedring sørger for at de tre første områder udføres bedre og billigere.

Kvalitet i produkt

- Kvalitet i et softwareprodukt vil beskrives af dens ikke-funktionelle krav. F.eks. brugervenlighed
- Arbejde med kvalitet kan i almindelighed opdeles i tre punkter:
 - Opstilling af kvalitetsmål (F.eks. hvis man snakker om en opvaskemaskine, kunne et mål være at den ikke må larme for meget, skal være målbar. Produktet skal være sikkert, effektivitet, genbrugbart)
 - Sikring af kvalitet (Det skal være nemt at ændre. F.eks. med tre-lags-arkitektur, er det nemt at ændre gui fra swing til javafx. UP er mere kvalitetsikret fordi man løbende tester, samt man laver diagrammer for at sikre at det er muligt tingene kan lave således)
 - Kvalitetskontrol(Laver tests, review)
- Andre test af kvalitetsegenskaber kan tjekkes ved review eller inspektion.

Kvalitet i proces

- Man skal kigge på om det er den eksterne eller interne proces, der er i fokus.
- Købere er selvfølgelig mere interesseret i den eksterne end den interne. Nogen gange kan det interne forhold dog godt have en betydning, f.eks. hvis det et dårligt arbejdsmiljø.
- Kvalitet i proces, kan her også deles i 3 punkter
 - Mål for processen (Både internt og eksternt. Hvilke skabeloner vil du bruge(Intern)?, målet er at opstille planer)
 - Kvalitetssikring (I den interne, kan man snakke om kvalitetsikring på den måde at man kan finde frem til den bedste arbejdsmetode. Den eksterne del, handler om at følge planerne)
 - Kontrol af kvaliteten (I den interne kan man gøre således at man følger op på forløbet med jævne mellemrum. Lav evalueringer af f.eks. arbejdsmiljøet. Kontrol af den eksterne gøres gennem indsamling af information om forløbet i forhold til de planer der er opstillet. Du kan f.eks. se om du har nået de milepæle du skulle. Burndown chart(Ekstern))
- Evaluering er vigtig del af kvalitetsforbedring af processen. Det er vigtigt fordi så kan du forbedre dig for hver gang du laver et projekt. Dette kaldes for Software Process Improvement(SPI)

Produkt og proces i metoder

- Produkt
 - Slutprodukt, selve systemet f.eks. aalborg bryghus
 - UML-diagrammer, kravspecifikation, user stories er mellemprodukter
- Proces
 - Planlægning (Ekstern)
 - Evaluering af forløbet (Ekstern)
 - Sprint retrospective (Ekstern, SCRUM, ser på hvad der fungerer og hvad der ikke gør)
 - Sprint review (Ekstern)
 - Det sociale, selve gruppen f.eks. frihedsgrader i gruppen eller arbejdsmiljøet (Intern)

12. Forundersøgelse

Gør rede for de fire faser i en forundersøgelse. Gør specielt rede for aktiviteterne og produkterne i fordybelsesfasen og fornyelsesfasen.

Du er velkommen til at drage erfaringer ind fra jeres eget projekt.

Indledning

- En forundersøgelse skal producere beslutningsgrundlaget for at igangsætte et implementeringsprojekt og dette grundlag skal sikres bedre forankring. Derfor går forundersøgelse ud på:
 - At analysere virksomhedens forretnings- og it-strategi samt dens aktuelle mål, behov og muligheder
 - At designe en eller flere visioner om en samlet forandring
 - At vurdere visionerne i forhold til virksomhedens forretnings og it-strategier samt i forhold til grupper af ansatte, relationer mellem afdelinger og i forhold til kunder og leverandører.
 - At fastlægge en strategi og plan for den tekniske og organisatoriske implementering samt at udarbejde økonomisk overslag for visionernes implementering samt løbende at sikre sig feedback fra relevante aktører
- Forundersøgelsen indledes, fordi ledelsen vil forfølge nogle forretningsmæssige mål, fordi en eller flere grupper i en virksomhed har oplevet et problem eller at et problem eller opgaver kan løses af it-anvendelser.
- Forundersøgelsen startes fordi der er noget som er præget af forskellige grad af usikkerhed.
- Forundersøgelsen skal rede trådene ud.

Forberedelsesfasen

- Projektetablering.
- Udformer planer til resten af faserne
- Man skal her fået afklaret og nå til enighed om rammerne for forundersøgelsen: F.eks. hvad skal opgaven gå ud på? Hvilke betingelser skal gælde for dens løsning? Hvordan vil projektgruppen gøre opgaven an?
- Man skal starte med at blive enig om emner som formål, ambitionsniveau, økonomiske og tekniske betingelser, ledelse og styring af projektet osv.
- Kilden til at begynde en forundersøgelse er altid at, ”nogen” har opstillet mål, har udtrykt et behov eller har oplevet et problem, der er koblet til it, eller at der på anden måde er opstået ideer til nye it-systemer eller nye it-anvendelser. Herefter begynder man så forberedelsesfasen.
- Man skal også teambygge, for at få social etablering.
- Man skal udform projektgrundlag og plan.

Fokuseringsfasen

- I fokuseringsfasen skal projektgruppen afklare og afstemme forundersøgelsens mål med virksomhedens forretnings- og it-strategi, for derigennem at identificere og afgrænse de arbejdsmråder, som forundersøgelsen skal undersøge it-støtte til.
- Forretningsstrategi: Svaret på, hvordan man når målet, handlinger der skal sikre virksomhedens overlevelse
- IT-strategi: Det IT-mæssige svar på, hvordan man når det forretningsmæssige mål, IT-midlerne til at realisere og styrke forretningsstrategien.
- Virksomheden har it-strategi og forretningsstrategi og de skal understøtte hinanden.
- Som resultat af fokuseringsfasen udvælges de arbejdsmråder, som it-systemerne skal anvendes til.
- Fokuseringsfasen sikrer, at projektet og dets resultater kan relateres og vurderes i forhold til overordnede strategiske mål.
- Fokuseringsfasen omfatter en identifikation og analyse af virksomhedens omgivelser, såsom kunder, leverandører og konkurrenter.
- Fokuseringsfasens resultat er en strategianalyserapport, der identificerer og prioriterer de arbejdsmråder, som den videre forundersøgelse skal omfatte, og som er i fokus i fordybelsesfasen.

Fordybelsesfasen

- Fordybelsesfasen er en dybdegående analyse af de udvalgte arbejdsmråder.
- Man skal kende til arbejdspraksis, og evt. komme med løsninger hertil.
- Denne fase udgør den centrale analyseorienterede del af en forundersøgelse
- Hovedresultat her, er en liste over mål, problemer og behov i den nuværende arbejdspraksis samt ideer til it-støtte og ændret arbejdsorganisering.
- Man laver her en analyserapport og supplerende beskrivelser af den nuværende arbejdspraksis.
- Den viden som man udvikler i fordybelsesfasen kan senere blive et problem, fordi når man i fornyelsesfasen f.eks. laver en prototype og så går man nogle problemer i møde.

Fasens mulige aktiviteter

- Planlæg fordybelsesfasen

- Aktiviteterne er indsamling af data, analyse af disse og præsentation af resultaterne
- Man skal klar lægge disse aktiviteter.

- Dataindsamling

- Man skal undersøge hvordan den enkelte arbejdsopgave bidrager til produktion af afdelingens samlede ydelse
- Man skal lave interviews suppleret af dokumentanalyse og observation

- Man skal opleve arbejdspraksis(tage derhen og opleve den) og derfra opnå adgang til konkrete erfaringer med de udvalgte arbejdsmråder.
- Fordybelsesfasens fokus bestemmes af det overordnede formål med forundersøgelsen. Fokus på opgaven kunne f.eks. være roller, arbejdsformer osv.

- Analyse

- Man skal her bruge det data man har indsamlet og bearbejde det på en måde, hvorpå projektgruppen kan bruge det. Interviews, videoklips m.m.
- Man skal gøre dette for at det så senere hen kan kommunikeres bredere, fx til styregruppen.
- Retningslinjer for analysen
 - Ikke spring til konklusioner for hurtigt
 - Synliggør data, overvejelser og resultater
 - Involver virksomheden eller afdelingen i overvejelser og refleksioner
- I afslutningen af fordybelsesfasen skal man forholde sig eksplisit til forandringsparatheden. Det handler om at kunne måle afstanden mellem nu-situationen og den ønskede tilstand og sammenholde dette med tidligere erfaringer med forandringer i virksomheden samt holdninger til forandringer hos virksomhedens ansatte.

- Præsentation

- Omfatter alt fra noter, efter det første interview til den ende analyserapport.
- Oplysningerne fra dataindsamlingen skal bearbejdes af nogen fra projektgruppen. Man skal skabe sig et overblik over dem, og det bruger man beskrivelsesværktøjer til, som f.eks. en kommunikationsmodel eller diagnostiske kort. Til sidst får man analyserapporten
- Præsenter det du har fundet frem til, dermed har du nogle ideer til løsninger, og der skal så kigges på hvad du vil gå mere i dybden i.

- Overvej ændret fokus for fasen

- Efter dataindsamlingen, analyse og præsentation kan det være man skal genoverveje fokus for fordybelsesfasen

- Rapportering

- Resultat er af fasen er en prioriteret liste over mål, problemer og behov samt ideer til it-støtte, ændret arbejdsorganisering og nye brugerkvalifikationer, som er udgangspunkt for fornyelsesfasen.
- Målgruppen for rapporten er styregruppen men også andre berørte ansatte i virksomheden.
- Folk som ikke har fulgt analysearbejdet i detaljer, er det i langt de fleste tilfælde nødvendigt at supplere de forskellige beskrivelser med tekst, som sammenfatter resultatet af analyserne.

- Teksten har tre formål:
 - Den kortlægger baggrunden for mål, problemer og behov samt for valget af fokuspunkter i dybdeanalyesen.
 - Den sammenkæder analyserne og beskrivelserne, dvs.
 - beskriver sammenhæng i mål, problemer og behov,
 - etablerer sammenhæng mellem mål, problemer, behov og ideer til løsninger.
 - Den samler og prioriterer mål, problemer, behov og løsningsideer.
- Således kunne disposition til analyserapporten se ud

Forsøgssesfaseren

- Afslutningen på forundersøgelsen
- Der udvikles en vision, til de problemer der er.
- Man skal lave visioner om de mål, behov og muligheder der blev lavet i fordybelsesfasen
- Samlede vision betyder at visionen skal omhandle såvel IT-systemernes funktion, grænseflade og den tekniske platform som arbejdets organisering og de kvalifikationer der er brug for.
- Man skal også tænke på fordele, ulemper og omkostninger i forbindelse med implementering af visionerne
- Afprøve ideer ved hjælp af mock ups, prototyper osv.

Fasens mulige aktiviteter

- Planlæg forsøgssesfaseren

- Forsøgssesfaseren skal nedbrydes i en række aktiviteter. Man skal også vælge hvilke beskrivelsesværktøjer og teknikker der understøtter disse aktiviteter.

- Markedsundersøgelse

- Man skal finde ud af om der er nogen systemer som indfrier nogle af virksomhedens behov, som blev afdækket i fordybelsesfasen. Det kan man ved at besøge messer, eller anmeldelser.
- Man skal så afprøve dette system, for at se om det kan bruges.

- Udvikle og samle ideer

- Gælder om at samle og videreudvikle de ideer som kom fra interviews, til nye IT-systemer og arbejdsorganisering,

- Mock-ups og prototyper

- Eksperimenter med mock-ups og prototyper er meget effektive teknikker til at visualisere og simulere udvalgte dele af visionernes it-systemer, inden disse implementeres.

- Kortlægning af kvalifikationsniveau

- Man skal sikre at fremtidige brugere, opnår tilstrækkelig kvalifikationer til at anvende den nye teknologi i den foreslæde arbejdsorganisering.

- Konsekvensanalyse

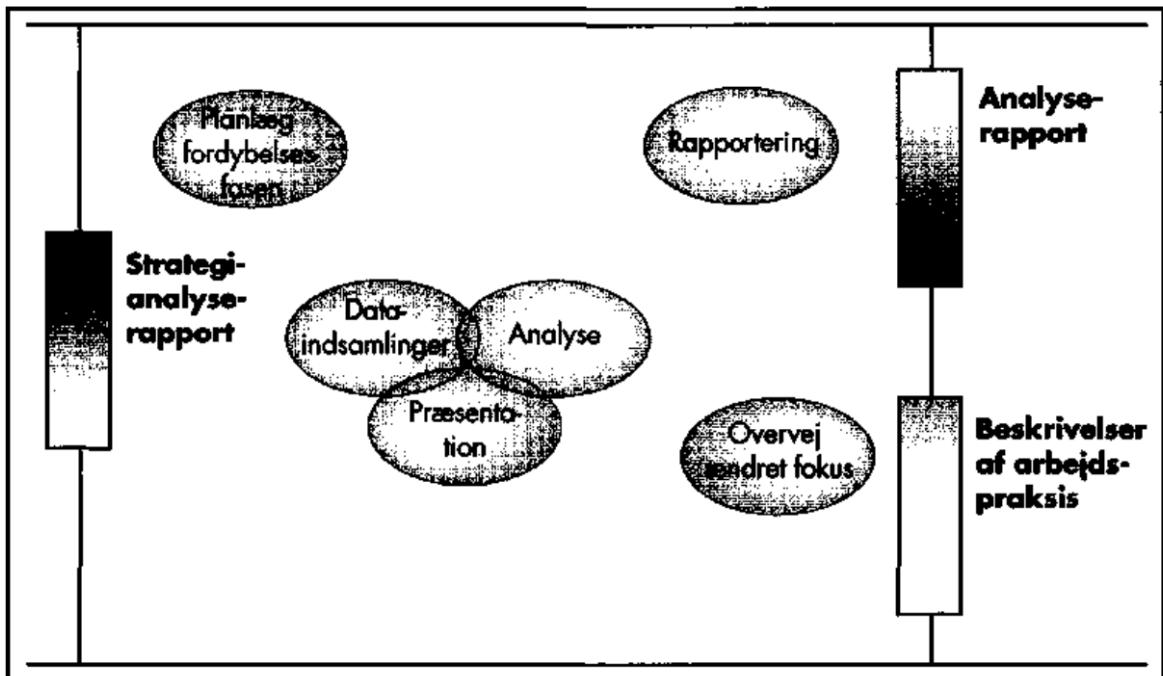
- Man skal kortlægge de fordele og ulemper, man på dette tidspunkt kan forudse, at en implementering kan føre hen til.
- Konsekvenserne for virksomheden, for ansatte osv.
- Konsekvenserne kunne være økonomi, arbejdsforhold, produkter og serviceydelser.

- Strategi og plan for implementering

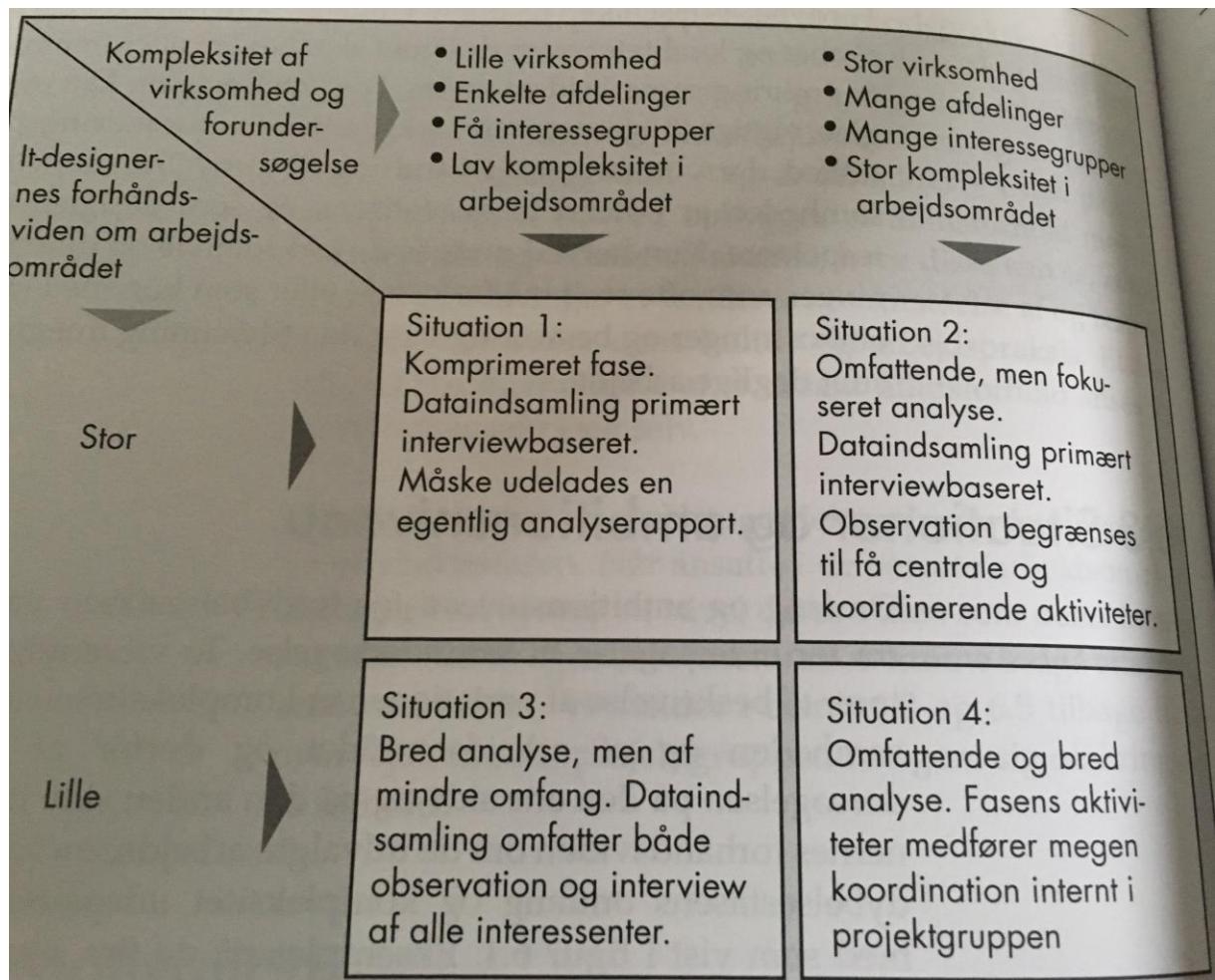
- Her skal man finde en strategi og en plan for den tekniske og organisatoriske implementering af visionerne.
- Strategier – skal forstås som en overordnet metode eller fremgangsmåde for implementeringsprojektet, omfanget af de aktiviteter, dette omfatter, og den økonomi, det involverer, kan være afgørende for om visionerne implementeres fuldt ud.
- Man skal se fordele og ulemperne ved de strategier man laver.
- Den bedste strategi for implementeringsprojektet afhænger både af karakteren af de foreslæde it-systemer og af deres omfang og kompleksitet:
 - Er der tale om standardsystemer eller nyudvikling.
 - Organisatoriske faktorer som den overordnede situation for hele it-projektet, der blev analyseret i fokuseringsfasen.
 - Virksomhedens forandringsparathed.
 - Traditioner og erfaringer med tidlige it-projekter.
 - Ledelsesstil
 - Organisationskultur. Risikostyring er et vigtig
- Risikostyring er et vigtigt element i strategien for et implementeringsprojekt.

Bilag

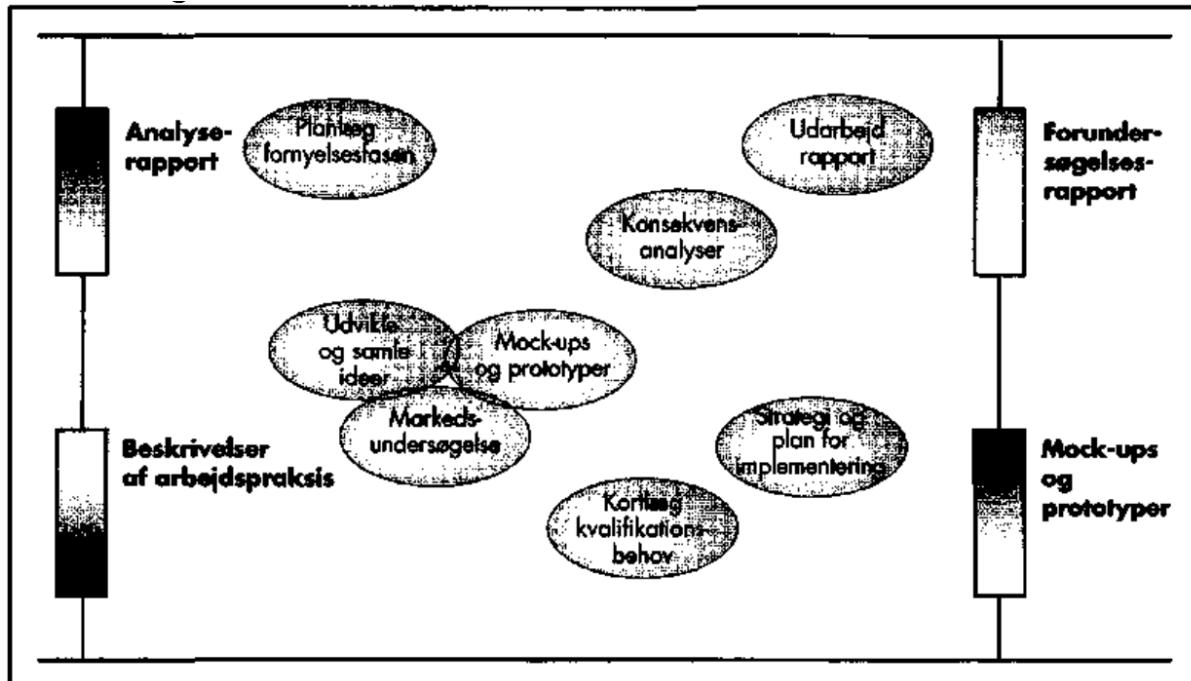
Fordybelsesfasen – Mulige aktiviteter



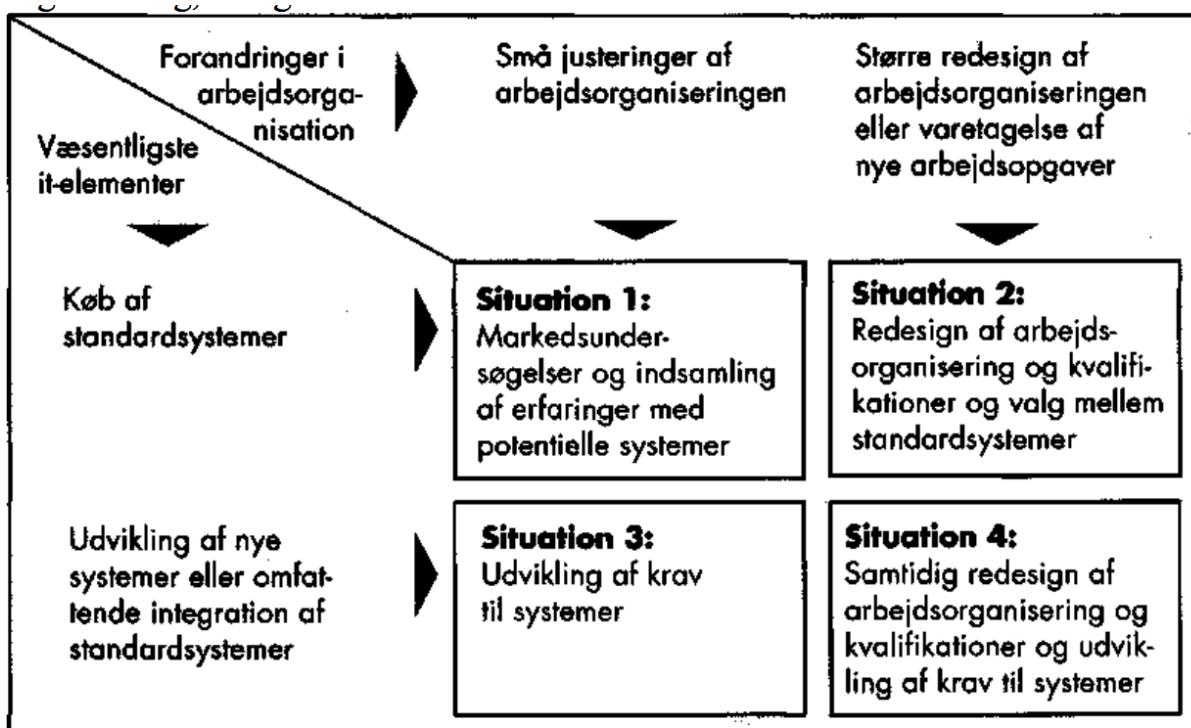
Figur 6.2. Produkter og aktiviteter, der potentielt kan indgå i fordybelsesfasen



Fornyelsesfasen – Mulige aktiviteter



Figur 7.2. Produkter og aktiviteter, der potentielt kan indgå i fornyelsesfasen



Figur 7.1. Fire typiske situationer for en fornyelsesfase