Adræt innovationsledelse

Søren Raaschou
Posility

Hvordan får virksomheder innovationen til at blomstre og sikrer, at den resulterer i produkter, som kunderne ønsker at købe? Det simple svar findes ikke – for så gjorde alle det bare og innovation ville ikke være en af de vigtigste konkurrence-parametre i dag.

Denne artikel leverer heller ikke innovationsledelsens "silver bullit", men har den lidt mere beskedne ambition at give en anderledes måde at se på innovation og innovationsledelse. Ønsker du ikke at få forstyrret dit syn på at innovation bedst styres top-down og helst skal følge overskuelige faseopdelte processer, er det tid at springe til den næste artikel. Denne artikel vil nemlig forholde sig til top-down og faseopdeling, som Darwins evolutionsteori forholder sig til skabelsesberetningen.

Den påstand, som denne artikel forsøger at belyse, er at innovation er så kompleks og dynamisk i dag, at der skal nye modeller til, og artiklen vil gennem eksempler og teori give et bud på, hvordan en anden metode kunne være.

"I begyndelsen …

… skabte Gud himlen og jorden. Jorden var dengang tomhed og øde, der var mørke over urdybet, og Guds ånd svævede over vandene. Gud sagde: »Der skal være lys!« Og der blev lys".[i]

Det var hvad der skete på den første dag af skabelsen. På anden dagen fulgte hav, jord og planter, og først på tredje dagen skabte Gud *solen*. Hvor mon lyset kom fra de første 2 dage, før solen dukkede op? Hvordan voksede planterne uden den for fotosyntesen så vigtige sol? Paradokset er, at der beskrives en proces, der er let at følge, men den kan bare ikke forklare, hvordan det reelt er gået for sig.

Dette eksempel fra den efter sigende første innovationsproces er ikke taget med for at starte en religiøs diskussion, men for at illustrere, at en 7 faset model kan skabe et overblik over en kaotisk og kompleks proces og kan give god mening, så længe man er enig om, at den er en god forklaring på, hvordan processen skal foregå. Men det er også et eksempel på, at en sådan simplificering ikke kan forklare kompleksiteten, og at man går glip at vigtige erkendelser, hvis man holder fast i den.

Figur 1: Processen i skabelsesberetningen

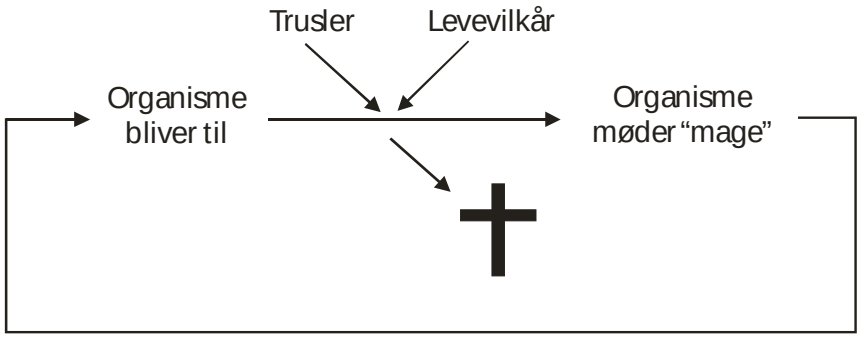
Darwin giver et helt andet sæt af forklaringer på den mangfoldighed og kompleksitet, der findes her på jorden, end skabelsesberetningens om at den var udtænkt og gennemført i én lineær proces. Ifølge Darwins forklaring har ingen skullet overskue designet af hver eneste utroligt

Kilde: Gertsen, F; Hansen, P K; Boer, H (red.) (2006), Innovationsledelse. Center for Industriel Produktion, Aalborg Universitet; ISBN 87-91831-01-6. Open source version http://innovation.cip.dk.

specialiserede art på jorden, men den er opstået ved millioner af gentagelser af en langt simplere proces:

1. En organisme kommer til verden.
2. Enten har den de egenskaber, der skal til for at overleve i dens miljø af levevilkår og trusler, eller også når den aldrig alderen til at bringe sine gener videre. Darwin kalder det: den naturlige selektion.[ii]
3. For dyrenes vedkommende skal de vælges af en mage, for at deres gener kan blandes i en ny organisme med egenskaber, som er en smule anderledes end forældrenes. Darwin kalder det: den seksuelle selektion.[iii]

Der er to mekanismer i spil i gentagelserne: Der skabes hele tiden varians, ved at generne kombineres, hvilket medfører en stor mangfoldighed, og der sker en forbedring af overlevelsesevnen igennem feedback fra omgivelserne og artsfæller.



Figur 2: Processen i evolutionen ifølge Darwin

Er mangfoldighed og overlevelsesevne ikke lige netop, hvad vi ønsker af innovation i dag: at skabe så mange ideer som muligt og hurtigt og sikkert at udvælge de, som vil overleve på markedet som produkter eller ydelser? Det er der næppe nogen, der vil benægte. Og processen har vist sig at virke i virkeligheden.

Hvordan kan vi så oversætte evolutionen til innovation? Modellen ser jo ud som en ganske simpel feedback sløjfe. Men i virkeligheden bliver den voldsomt kompleks, når man tænker på, hvor mange sløjfer, der kører parallelt i verden – reelt en for hver organisme – og hvor forbundne de er, fordi de lever med og af hinanden. I virkeligheden viser det sig jo også at tage tusinde år at nå resultaterne. Eller gør det? Det afhænger jo meget at livscyklusens varighed: nogle organismer kommer gennem en cyklus på bare én dag.

Der kan være mange andre indvendinger imod, at Darwins evolutionsteorier skulle kunne implementeres som innovationsprocesser, men det tjener som et eksempel på, hvordan iterative/gentagne processer er langt bedre til at indfange den kompleksitet, vi omgiver os med og til at justere processen imod målet.

Iterative processer

Når jeg har betragtet processer generelt i erhvervslivet, har det slået mig, hvor meget faseopdelte processer, som f.eks. baseret på state-gate, dominerer, og i hvor ringe grad iterative processer er taget i anvendelse. Det gælder specielt i Vesten, hvorimod Østen især repræsenteret ved Japan, tænker anderledes. Om det hænger sammen med vores opfattelse af det lineært fremadskridende liv over for asiaternes cirkulære opfattelse af reinkarnation, ved jeg for lidt om til at kunne afgøre. Men for mig er tanken nærliggende. Hele industrialiseringens i øvrigt meget succesfulde fokus på opdeling og specialisering af arbejdsopgaver med henblik på en

Kilde: Gertsen, F; Hansen, P K; Boer, H (red.) (2006), Innovationsledelse. Center for Industriel Produktion, Aalborg Universitet; ISBN 87-91831-01-6. Open source version http://innovation.cip.dk.

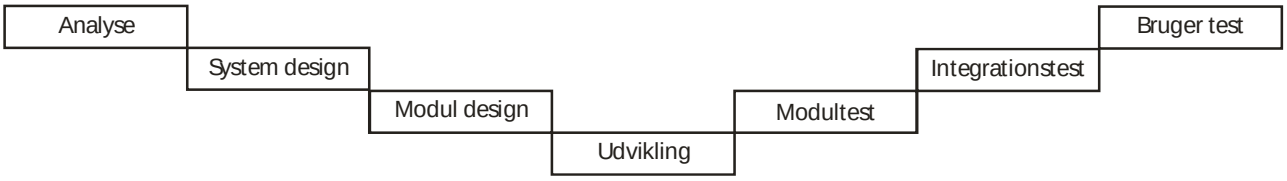
effektiv gennemførsel og automatisering har med sikkerhed også en stor indflydelse på, hvordan vi i vesten tænker i dag.

Ser man f.eks. på de japanske produktionsfilosofier – her i Vesten implementeret under Lean begrebet – er det iterative element gennemgående. Et af de centrale begreber er Kaizen, som er japansk for forbedringer: små forbedringer dag for dag. Kaizen er den løbende, aldrig sluttende forbedringsproces, der forbedrer i små billige trin, og alle fra ledelse til arbejdere er involverede[iv]. Ved at der fokuseres på dag-til-dag ændringer er det små og vedkommende skridt for den enkelte medarbejder, hvilket mindsker forandringsmodstand. Da forandringerne samtidig sker hver dag, er den samlede forandringshastighed høj.

Lean bliver mere og mere udbredt i vesten, og det skyldes, at det leverer resultater – bedre resultater end de traditionelle tankegange. Når de positive effekter bliver bevist for øjnene af os, tilegner vi os gerne processer, som umiddelbart er fremmede for vores sædvanlige måde at tænke på. Østen har ikke eneret på iterative processer – på ét område bliver der i disse år gjort op med den lineære procesforståelse, og iterative processer vinder frem uden at være funderet i Østen. Og de har tilmed gjort det med stor succes: Det er de adrætte software-udviklingsprocesser.

Adræt softwareudvikling

Ligesom på alle andre områder har processer til udvikling af software i mange årtier og er stadig mange steder, baseret på faseopdelte modeller, som afspejler de forskellige arbejdsopgaver som kan identificeres i softwareudvikling: kravene til systemet skal beskrives (analyse), systemets tekniske udformning skal bestemmes (design), systemet skal udvikles og testes. De, som havde behov for i processen at afspejle, at software systemer ofte er modul opbyggede, har haft flere design faser og dertil hørende tests.



Figur 3: V-modellen for softwareudvikling

Modellen, som kommer ud af denne tankegang er egentlig ret smuk at se på, med klare fremadskridende faser og oven i købet symmetrisk, som om en højere magt havde skabt den. Den tegnede model er kendt som V-modellen, men mere kendt er *vandfaldsmodellen*, som den er kaldt, uden at testfaserne er sat op over for deres skabende modpol f.eks. modultest over for modul design. Et meget velvalgt navn, da hele tankegangen bag falder sammen med det lovbestemte i, at vand altid falder nedad – eller glider fremad. Altså at man kun skal gennemføre hver arbejdsopgave én gang.

I midten af 1990'erne begyndte en række praktikere at undre sig over, at der var mange softwareprojekter der blev forsinkede, overskred budgetterne, ikke levede op til brugernes krav, var fulde af fejl osv. Projekter som blev udført efter vandfaldsmodellen og af relativt modne organisationer. De brød med tanken om, at når processen ikke fungerede, behøvede det ikke at være menneskene, der var noget galt med – det kunne også være processen, uanset hvor smuk den ser ud.

Derfor begyndte de at undersøge andre projekter, som godt kunne levere varen. De var interesserede i at finde ud af hvordan de, som var excellente til at gennemføre softwareprojekter til tiden, inden for budgetterne med få fejl og som frem for alt leverede et produkt, som dækkede brugernes behov, gjorde. Og de fandt, at de processer, som i virkeligheden resulterer i

Kilde: Gertsen, F; Hansen, P K; Boer, H (red.) (2006), Innovationsledelse. Center for Industriel Produktion, Aalborg Universitet; ISBN 87-91831-01-6. Open source version http://innovation.cip.dk.

succesfulde projekter, er helt anderledes end vandfaldsmodellen. Selvom det egentligt ofte var beskrevet, at de skulle følge vandfaldsmodellen.

Det lykkedes Kent Beck[v] i 1999 ikke bare at beskrive en helt anderledes empirisk baseret proces, men samtidig at få softwareudviklerne og ikke bare lederne til at interessere sig for *udviklingsprocessen*. Det gjorde han ved at fokusere på udviklernes produkt – nemlig den kode de skriver – og ved at fremhæve de elementer, som udviklerne i forvejen ved fungerer, og forstørre dem til det ekstreme, hvorfor Kent Becks variant af de agile processer er kendt som Extreme Programming.

For eksempel: ”Når det er godt at teste for at opnå kvalitet, hvorfor så ikke gøre det hele tiden?” har ført til tanken om, at hver gang den mindste ændring foretages, skal der testes. For at kunne overkomme det, har denne tankegang ført til, at udviklerne skriver små programstumper, som tester hver enkelte lille del af programmet, når der er behov for det, og behovet er der hver gang der er programmeret i nogle minutter. Og når der alligevel skal skrives tests til automatisk afvikling, hvorfor så ikke gøre det, *inden* funktionaliteten skrives? – så ved udvikler tilmed, hvornår man er færdig til at gå videre med næste funktionalitet.

Eller: ”Når det er så godt at dele viden og lære af hinanden, hvorfor så ikke gøre det hele tiden?” har introduceret par-programmering, hvor udviklerne sidder sammen med ét tastatur, som de skiftes til at bruge.

Begge tanker var meget provokerende ved fremkomsten, hvor vandfaldmodellen herskede, og disse aktiviteter var knyttet til en bestemt fase. Par-programmering er nok den del af Extreme Programming, som er blevet mødt med mest vantro: Hvordan kan produktiviteten undgå at falde, når to gør det samme, som de ellers kunne gøre i parallel? Hvis man tager den tid. som normalt bruges på at spørge kollegaer, på at sidde og prøve ting som ikke virker, på at dele/opbygge viden, på at kun én kender dele af systemet osv., vil mange vide, at det ikke er en helt ubetydelig tid, der spares på kort sigt. Hvad effekten på lang sigt af at al kode er kendt af mindst to personer og alle valg er blevet taget af flere, i forhold til vedligeholdelse osv., er en trossag, fordi det er svært at lave sammenlignende forsøg. Derfor er det også de færreste organisationer, som er ekstreme i deres implementering af par-programming, men gør det, når det har åbenlys effekt, f.eks. når der startes på nye områder eller nye folk læres op.

I virkeligheden er Extreme Programming i dag blot en blandt mange empirisk baserede metoder, som beskrives fra slutningen af 1990’erne, og februar 2001 samler Kent Beck en række af personer, som arbejder med ikke-vandfaldsmetoder som f.eks. Alexander Cockburn, Ward Cunningham og Jim Highsmith for at finde ud af, hvad der er fællestrækkene for deres tanker og metoder. Det resulterede i ”Manifesto for Agile Software Development”[vi] som præciserer 4 simple prioriteringer af, hvad der er vigtigst i adrætte udviklingsprocesser:

- ***Individer og interaktion*** over processer og værktøjer
- ***Fungerende software*** over omfattende dokumentation
- ***Kunde samarbejde*** over kontrakt forhandling
- ***Reagere på forandring*** over at følge en plan

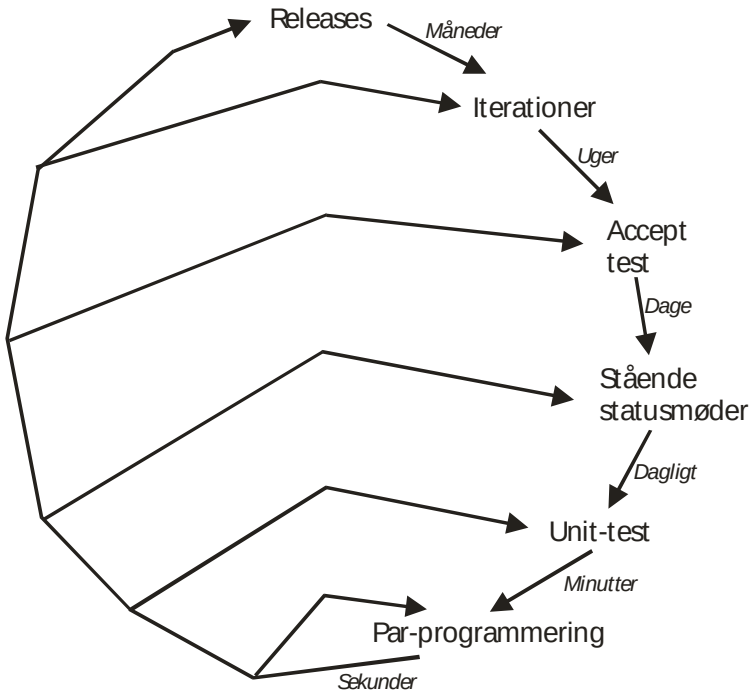
Udsagnene skal ikke forstås som modsætningspar eller som, at det på højresiden ikke er væsentligt, men som at venstresiden er de vigtigste værdier.

Det vigtigste fællestræk for metoderne er, at de er meget adrætte over for forandringer i projektet og dets relationer forstået meget bredt, samtidig med at det empirisk bevist er bedre til at levere den software, som behøves til den tid og økonomi, der forventes. Det har givet dem deres fælles betegnelse.

Kært barn har mange navne, og de adrætte udviklingsmetoder kaldes også af mange iterativ udvikling, fordi alle delelementer udføres i iterationer og derfor gentages mange gange i

Kilde: Gertsen, F; Hansen, P K; Boer, H (red.) (2006), Innovationsledelse. Center for Industriel Produktion, Aalborg Universitet; ISBN 87-91831-01-6. Open source version http://innovation.cip.dk.

løbet af et projektforløb. Ved opstarten af projektet finder projektgruppen ofte en metafor for det system, som de skal udvikle for at have et vokabularium for de begreber som bruges, og der laves en overordnet release plan. Det vil sige, hvad det initialt forventes, at systemet skal kunne ved de følgende releases til brugerne. Bortset fra metaforen gentages alt derefter i iterationer.



Figur 4: Adræt softwareudviklingsprocessen

Som det ses på figuren, itereres der på mange niveauer, fra iterationer der varer sekunder til månedlange iterationer. Typisk leverer et adræt softwareudviklingsprojekt en fungerende version (et release) af det system, der udvikles til slutbrugerne med 1-3 måneders intervaller. Hver release cyklus består af et antal iterationer på 2-4 ugers varighed, som leverer fungerende software til kunden – forstået som den gruppe, der stiller kravene og betaler. De vurderer funktionaliteten og prioriterer, hvad der er vigtigst, at projektet leverer i den næste iteration. Denne prioritering kan omfatte mere funktionalitet, justering af den der findes, tekniske ændringer osv. og det er kunden, som bestemmer på baggrund af udviklernes vurdering af omkostningen og andre konsekvenser som f.eks. stabilitet eller performance.

Releases og iterationer leverer til omgivelserne – de resterende cykler sker inden for projektet. Her er kunden repræsenteret hver dag for at sikre adrætheden i forhold til de forretningsmæssige krav, og denne accepterer funktionaliteten, efterhånden som den bliver klar, deltager på daglige stående statusmøder og har i det hele taget ansvaret for hele tiden at bidrage med den forretningsmæssige viden og prioritering.

Udviklerne itererer i ultra korte intervaller ved at arbejde i par og teste som beskrevet ovenfor, og de deler frustrationer og viden på de daglige stand-up møder. Derudover designer de overordnet systemet i forbindelse med iterationsopstarterne.

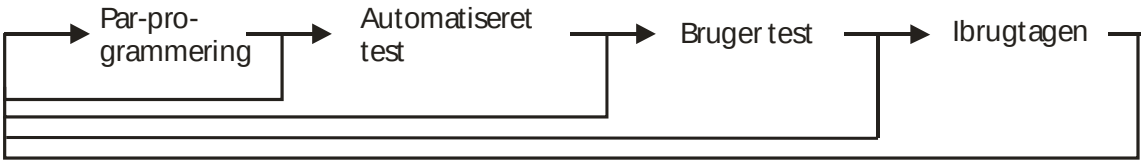
Som implementør af agile processer i flere organisationer er det min opfattelse at succes'en af de agile udviklingsprocesser i høj grad hænger sammen med den høje grad af iterativitet. Iterationer giver feedback til alle centrale elementer i udviklingsprocessen, og giver nye ideer, som forbedrer det foregående. Det gælder kvalitet og fremdrift for pengene, det gælder at dække brugernes behov og det gælder en kontinuert optimering af processen. Fak-

Kilde: Gertsen, F; Hansen, P K; Boer, H (red.) (2006), Innovationsledelse. Center for Industriel Produktion, Aalborg Universitet; ISBN 87-91831-01-6. Open source version http://innovation.cip.dk.

tisk som når to organismer blandes i en ny, som skal klare den næste cyklus, hvor den møder andre levevilkår og trusler end den foregående generation.

Kvalitet og fremdrift

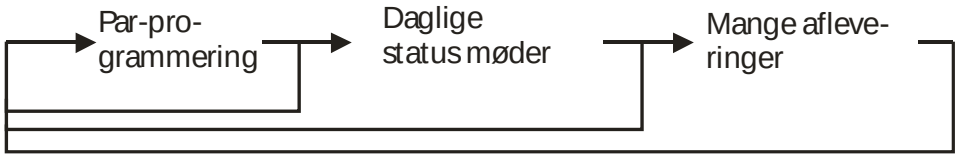
I det foregående er det beskrevet, hvordan kvalitet er indbygget i den adrætte udviklingsproces, ved at der par-programmeres, der skrives automatiserede tests, kunden tester i løbet af og efter en iteration, og brugerne tager systemet i brug efter hver release. Det betyder, at kvalitet er et ansvar for alle – ikke bare en testfunktion. Det betyder også, at alle fokuserer på kvalitet, og det tages i små bidder over længere tid, så det altid er overkommeligt at gøre det nødvendige for at rette "skuden" op.



Figur 5: Gentagelserne i kvalitetssikring

Det modsatte er desværre alt for tit problemet i projekter, der følger vandfaldsmodellen, fordi forsinkelser i de tidligere faser sjældent flytter den dag, systemet skal tages i anvendelse. Derfor bliver de sidste faser presset i tid, og i vandfaldmodellen er det testen, hvilket har alvorlige konsekvenser for kvaliteten.

De mange feedback sløjfer har også en positiv indvirkning på fremdriften i projekterne, fordi der er mange muligheder for at få hjælp, når man sidder fast i et problem, og fordi man i mange situationer skal stå til regnskab for det, man laver.



Figur 6: Gentagelser der fremmer fremdrift

Par-programmering er et redskab, der sikrer, at der er en at diskutere løsninger med, hvilket mindsker perioderne, hvor en udvikler sidder og spilder tid på noget andre måske ved. Par-programmeringen har også den positive effekt, at viden om, hvordan værktøjer mv. bedst udnyttes, spredes, hvorved alle bliver mere produktive.

Når par-programmeringen ikke rækker i forhold til at finde løsninger, er de daglige stående statusmøder en god platform til at spørge alle, så der ikke spildes mere tid. Sommetider kan problemerne løses, mens gruppen står op, ellers parres problemhaveren med den, der ved bedst. De daglige statusmøder har også den effekt, at alle skal stå til regnskab for, hvad de har nået den foregående dag, og hvad de forventer at nå den forestående dag. Det giver dagligt commitment til at nå et stykke fremad, og summen af disse commitments giver en øget fremdrift.

Det tredje niveau, som øger fremdriften, er, at systemet skal afleveres til brugerne efter hver iteration, hvilket kræver, at tingene er færdige. Ken Schwaber kalder også i hans SCRUM[vii] metode iterationerne for *"sprints"*, hvilket er en god betegnelse, fordi iterationerne opdeler et projekt i en lang række kortdistanceløb fremfor et maratonløb. Effekten af hele tiden at skulle nå et mål, som ligger tæt på i tid og størrelse, ligner den vi kender fra atletikkens verden: Det er lettere at holde farten oppe i en kort periode, og det slider ikke så hårdt. I erhvervsmæssig sammenhæng betyder det et lavere stressniveau. Alle holder farten oppe og

Kilde: Gertsen, F; Hansen, P K; Boer, H (red.) (2006), Innovationsledelse. Center for Industriel Produktion, Aalborg Universitet; ISBN 87-91831-01-6. Open source version http://innovation.cip.dk.

holder fokus på det, der skal leveres, og ikke alt muligt andet, som kunne være sjovt undervejs – og ingen skal i en lang periode yde det ekstraordinære for at nå i mål. Det er ikke så let at komme uger efter planen i løbet af iterationer på 2-4 uger.

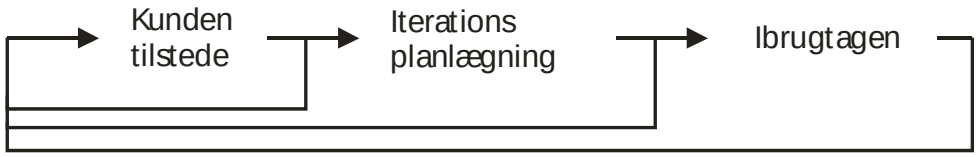
Dække brugernes behov

I manges forståelse af adrætte projekter er den nyskabelse, der ligger i, at kunden i de adrætte udviklingsprojekter gøres til en del af projektet og er til stede hver dag til at prioritere krav og svare på spørgsmål, en afgørende forskel på adrætte projekter og vandfaldsprojekter. Det er også en meget vigtig del.

Kundens tilstedeværelse er i min forståelse vigtig, fordi den gør det muligt for projektet at følge de erfaringer, kunde og brugere får undervejs i processen. Meget software understøtter processer i virksomheder, og det skabes med henblik på effektiviseringer eller lignende. Det resulterer i, at den virkelighed, som findes ved udviklingens start, hvor kravene traditionelt sættes op, erstattes af en anden, når systemet indføres. Den virkelighed, hvori systemet skal eksistere, påvirkes yderligere af forandringer i virksomhedens omgivelser, f.eks. når konkurrencevilkårene ændres af konkurrenter, eller når virksomheden selv tager en anden strategisk retning. Det gør det meget svært at stille de endelige krav op f.eks. et år i forvejen, fordi virkeligheden på brugstidspunktet reelt ikke er kendt.

Derudover er software et meget abstrakt produkt, indtil det er færdigt. Der er mange måder til at tegne, hvad et stykke software skal kunne, men uanset måden vil det kun repræsentere en delmængde, og det kræver stor abstraktionsevne at forestille sig det færdige produkt.

Samlet set mener jeg ikke, at det er muligt at opstille alle de rigtige detaljerede krav til et system, inden man begynder at få erfaringer med systemet, hvor det skal bruges. At opstille alle kravene optimalt svarer i skabelsesberetningen til at skulle beskrive alle dyrs specialiseringer og samspil med andre dyr og planter – uden at have noget kendskab til dem på forhånd.



Figur 7: Gentagelser af kundeinvolvering

Vanskeligheden ved at stille alle krav op initielt, håndterer de adrætte udviklingsprocesser ved at tillade, at kravene ændres med de erfaringer, kunden gør sig undervejs ved at se systemet i brug. I løbet af et projekt på f.eks. et år kan den tilstedeværende kunde korrigere krav 365 gange blot ved den daglige tilstedeværelse, 25-50 gange kan en mindre gruppe prioritere helt nye krav til den kommende iteration, og 4-12 gange tages systemet i brug af de reelle brugere. En stor kontrast til at skulle overskue kravene på én gang.

Et af de vigtige elementer i iterationsplanlægningen er, at kundegruppen laver en prioritering af kravene til iterationen på baggrund af den estimerede omkostning ved at opfylde det enkelte krav. Det har en stor effekt. Jeg har oplevet krav, som initielt var prioriteret helt i top, falde i prioritet, fordi andre krav syntes at give større værdi for pengene, for til slut aldrig at blive implementeret. At det, der i starten var det vigtigste, aldrig rigtig kunne betale sig at lave, viser, hvor svært det kan være at stille alle krav op fra starten.

Det, at softwaren kommer ud til den reelle bruger 4-12 gange i løbet af projektet frem for bare én gang, har også en stor effekt, fordi det er i brugssituationen, mange af de mest værdifulde ideer kommer. Proces og software kommer i interaktion, og det sikrer, at begge dele bevæger sig i samme retning, og det sikrer at man ikke ender med en proces og et system, som

Kilde: Gertsen, F; Hansen, P K; Boer, H (red.) (2006), Innovationsledelse. Center for Industriel Produktion, Aalborg Universitet; ISBN 87-91831-01-6. Open source version http://innovation.cip.dk.

slet ikke passer sammen. Samtidig får mange softwaren mellem hænderne, og mange hoveder genererer flere og bedre ideer end få.

Læring

Ovenfor er det beskrevet, hvordan læringen omkring produktet og den sammenhæng, den virker i, foregår. Kunden bliver mere og mere bevidst omkring produktet, og samtidig lærer softwareudviklerne mere og mere om kundens forretning og begynder at bidrage med ideer og alternative løsningsforslag, som dækker samme behov til en lavere pris.

Det er også beskrevet hvordan, softwareudviklerne udveksler viden omkring teknologien gennem par-programmering og daglige statusmøder. Dette suppleres ofte med fælles designdiskussion ved indgangen til en ny iteration, hvor de nye krav er kendte. Det både dygtiggør udviklerne fagligt og er med til at sprede kendskabet til hele systemet.

Den tredje dimension, hvori der foregår læring, er procesdimensionen. Mest brugt er det at afslutte iterationer og releases med en procesevaluering med henblik på at optimere processerne. Disse evalueringer er mere vedkommende end evalueringer af færdige projekter, fordi de ting, der kommer frem, påvirker de selv samme mennesker fra den følgende dag. Evalueringerne giver et kontinuert fokus på hele tiden at gøre det bedre, som afspejler sig i resultaterne.

Fra adræt software udvikling til adræt innovation

Som det fremgår af det ovenstående har man inden for softwareudvikling med stor succes gjort op med de lineære processer og har erstattet dem med cirkulære processer. Forbedringerne er sket på de områder, hvor der tidligere var så store problemer, dvs. der er opnået billigere og mere præcise leverancer, højere kvalitet og systemer, der bedre opfylder kundernes krav, samtidig med at evnen til at lære og forbedre processerne er forøget.

Hvis tilsvarende forbedringer kunne opnås i forhold til innovation, ville det ikke være så dårligt, men hvad har innovation med adræt softwareudvikling at gøre?

Målet for to processer har det til fælles, at det handler om at skabe en forandring. Den situation, som er gældende, er ikke holdbar og processen startes for at finde den bedste vej videre. I begge tilfælde gælder det om at bryde med de eksisterende forestillinger om, hvordan tingene skal være, og gerne komme ud med noget, som er markant mere profitabelt.

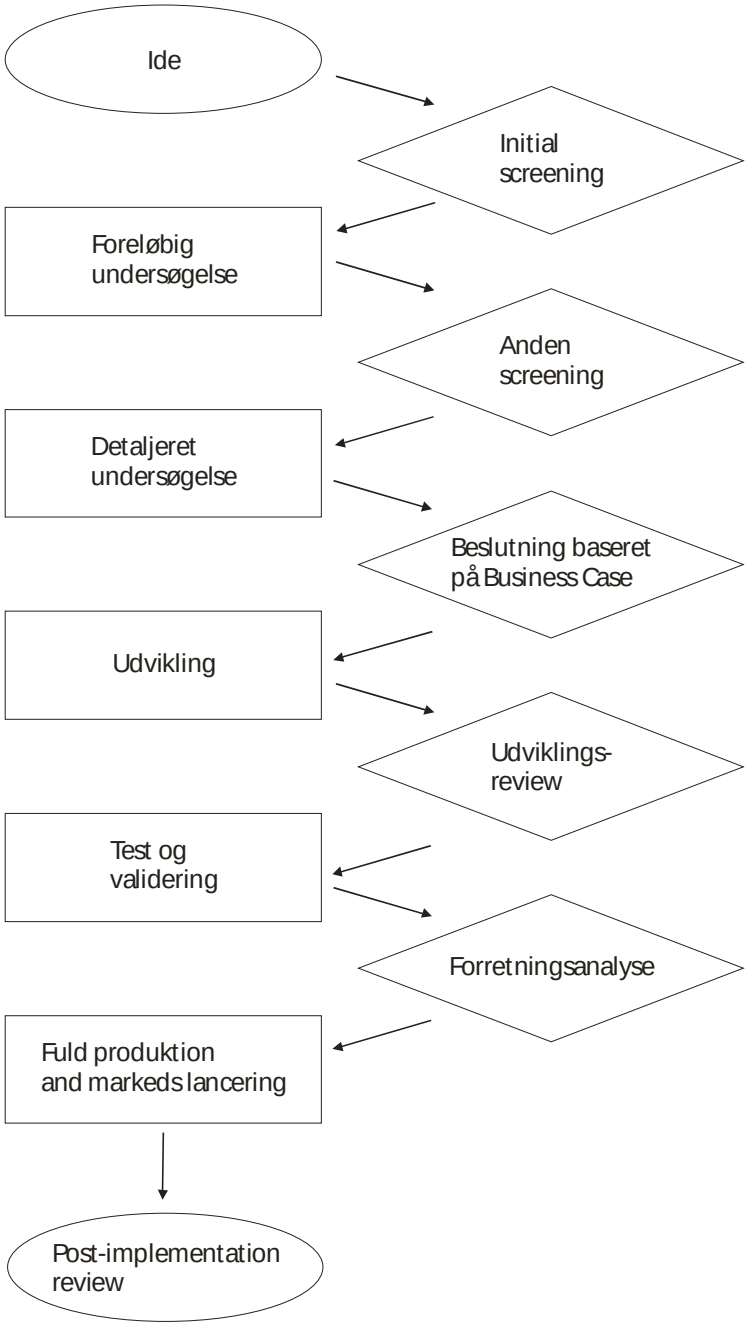
Produktet af de to processer har det til fælles, at det er uforudsigeligt. Ingen kender det endelige resultat, fordi det skal dække et behov, som findes i fremtiden. Derfor er udgangspunktet abstrakt og svært at beskrive, og at processen handler i høj grad om at blive klogere på problemstillingen og de mulige løsninger. På grund af uforudsigeligheden er produktet også resultatet af en beslutning om, at produktet er godt nok nu. I begge tilfælde ville processen kunne fortsætte i det uendelige med at prøve flere ideer af, men når det skal kommercialiseres, tages en beslutning om, at produktet er godt nok nu.

Redskaberne til at nå målet har også det til fælles, at der er utrolig mange parametre at skrue på. Software er i den forstand ”soft” – der er ingen begrænsninger for, hvad der er muligt, som der for eksempel er i den fysiske verden – og det betyder, at der er uendelig mange kombinationsmuligheder. I innovation – selv af fysiske produkter – er mulighederne i løsningsrummet også uendelige, fordi innovation kan ske i mange dimensioner på en gang f.eks. i produktets egenskaber, i optimeringer af for eksempel fremstillings-, logistikprocesser eller i positioneringen af produktet[viii] på markedet for at opnå højere priser. Det kræver mange forskellige kompetencer at få det til at gå op i en højere enhed i det færdige produkt. Kompetencer som lige så godt kan inspirere og supplere hinanden i løbet af innovationsprocessen. På samme måde som i adræt softwareudvikling.

Kilde: Gertsen, F; Hansen, P K; Boer, H (red.) (2006), Innovationsledelse. Center for Industriel Produktion, Aalborg Universitet; ISBN 87-91831-01-6. Open source version http://innovation.cip.dk.

Innovation som faseopdelt

Mange innovationsprocesser bliver ligesom i traditionel softwareudvikling beskrevet som faseopdelte processer. De ser for eksempel ud som hos Robert G. Cooper[ix].



Figur 8: Faseopdelt innovationsproces

Faserne er ide generering, indledende undersøgelse, detaljeret undersøgelse, udvikling, test samt produktion og markedsføring. Mellem hver fase er der et beslutningspunkt (rhomberne på figuren) hvor ideen enten forkastes eller forsættes i næste fase.

Det er påfaldende, hvordan opdelingen af processen, som i vandfaldsmodellen er baseret på de aktiviteter, der indgår: Ideerne skal genereres i den første fase, og disse evalueres derefter på baggrund af økonomi, risiko, tilstedeværende kompetencer etc. af to omgange. De ide-

Kilde: Gertsen, F; Hansen, P K; Boer, H (red.) (2006), Innovationsledelse. Center for Industriel Produktion, Aalborg Universitet; ISBN 87-91831-01-6. Open source version http://innovation.cip.dk.

er, som slipper gennem screeningen, udvikles, og det testes i forhold til produktion og marked, om ideen endeligt kan gennemføres. Passerer produktet også denne gate, sættes det i produktion, og markedsføring og salg initieres. Tanken er, at tusinde af ideer i passagen af de mange gates efterhånden reduceres til ganske få overlevelsesdygtige ideer, som tusinde haletudser, der bliver til få frøer. Ikke som de højerestående dyr, som får meget færre afkom.

Dette er ikke en særlig adræt proces. Hvordan håndteres ideer, som opstår i løbet af processen f.eks. hos de, der udvikler produktet? Er det nødt til at blive til en ny ide, som starter forfra i processen? Eller endnu værre hvis ideen udvikler sig når den bliver testet i forhold til markedet – hvor lang tid går der så, til en ny version kan være parat til at komme på markedet? Eller hvis der sker noget afgørende på markedet, mens produktet er på vej gennem processen?

Processen er designet efter at få sorteret dårlige ideer fra tidligt for at spare omkostninger, men den er ikke i stand til at håndtere hvis ideen forbedres undervejs eller vilkårene for produktet ændrer sig. Det kan også koste mange penge.

Omvendt kan man risikere, at gode ideer bliver filtreret fra på et for tidligt tidspunkt, fordi det i de indledende vurderinger vurderes, at det er umuligt at udvikle eller producere – før udviklingsafdeling eller produktionsafdelingen har haft mulighed for at lave innovation i forhold til for eksempel deres teknologier eller processer. Den tabte gevinst, der ligger gemt her, kan i sagens natur ikke gøres op, men da det måske er de ideer, konkurrenterne også har opgivet, kan der ligge potentielle cashcows her.

Hvordan kan innovationsprocessen gøres mere adræt? Ser vi tilbage på de principper, som den adrætte softwareudvikling hylder, er det

- tæt interaktion mellem de mennesker, som deltager i udviklingen, og kunder/brugere
- fokus på at have et fungerende produkt
- iterationer på mange niveauer for at maksimere feedback på produkt og proces
- og frem for alt at have adræthed i forhold til ydre og indre påvirkninger i projektet

I det følgende afsnit vil jeg forsøge at give et bud på, hvordan disse principper kan mappes til innovation for at skabe en mere adræt proces.

Tæt interaktion i innovationsprocessen

Innovation kan være drevet af forskellige kræfter, f.eks. teknologi presser produkterne frem, anderledes design er vigtigste mål eller trækket fra kundeønsker etc. Der er også forskellige grader af radikalitet i innovationen. Det har naturligvis indflydelse på, hvilke grupper som er den primære kraft i innovationen i virksomheder, dvs. hvem der sætter dagsorden i forhold til ideernes skabelse og udvikling. I faseopdelte modeller som Coopers vil det typisk være dem, der sidder på ide-fasen, og derfor er det primært deres ideer, som bliver til nye og forbedrede produkter, men andres ideer kan blive opfattet som støj, fordi de kommer så sent i processen, at det bliver dyrt at implementere dem.

Hvis man, som jeg, tror på, at den gode ide er uafhængig af tid og rum, at interaktionen mellem mange forskellige interessenter med forskellige baggrunde øger chancen for den gode ide, og at ideer udvikler sig efterhånden som de bliver gjort mere konkrete, er de faseopdelte modeller ikke hensigtsmæssige for innovation. Det gælder om i stedet for at etablere en proces, hvor feedback og ideer hele tiden fødes tilbage for at forbedre produktet.

I adræt softwareudvikling er interessenterne primært kunden (den som betaler for udviklingen) og brugerne (dem som bruger softwaren), men det kan også være mange andre, hvis det f.eks. er en softwarevirksomhed, som sælger et produkt. I sådan et tilfælde omfatter interessenterne også marketing, salg, supportorganisation osv. Deres deltagelse i udviklingspro-

Kilde: Gertsen, F; Hansen, P K; Boer, H (red.) (2006), Innovationsledelse. Center for Industriel Produktion, Aalborg Universitet; ISBN 87-91831-01-6. Open source version http://innovation.cip.dk.

cessen vil sjældent være på daglig basis, men snarere i iterations- eller release-planlægningerne.

Ser man på andre typer af produkter, f.eks. fysiske, kommer andre interessenter på banen: designere, produktionsfolk, leverandører osv. Det store antal af bidragydere til processen gør det hele mere komplekst, men det er samtidig et overflødighedshorn af ideer og dermed muligheder for at løbe fra konkurrenterne. Denne større kompleksitet gør det ikke lettere at fange alt i faserne i en faseopdelt model – tværtimod er der behov for endnu mere interaktion for at få synergien mellem ideerne frem.

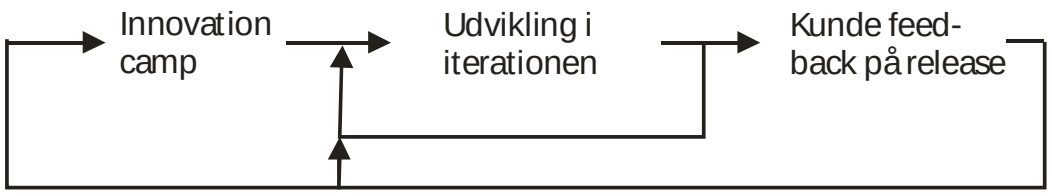
Følger vi tankegangen i adræt softwareudvikling, skulle innovationsprocessen starte med at skabe en vision/metafor for processen og derefter have cykler af forskellig længde, hvor de forskellige interessenter i processen kommer med deres ideer og anden input.

Metaforen kunne for eksempel skabes i en innovation camp – et begreb som er ved at blive mere udbredt. En innovation camp er designet til at skabe et meget koncentreret samarbejde i et par døgn, hvor kreativitet, research og test spiller sammen for at finde holdbare ideer. En super intensiv proces, hvor hele livscyklus for produktideer gennemspilles i timekorte iterationer. Deltagerne er en bred gruppe, som repræsenterer virksomhedens mangfoldighed og inspiratorer udefra. Inspirationen kan komme fra personer, som har visionerne og viden om, hvilke trends som skal ligge i fremtidens produkter. Roberto Verganti kalder disse key interpreters[x], fordi de er i stand til at fortolke brugernes kommende behov.

Ved at starte ud med en bred gruppe i et intenst forløb sikrer man at få virksomhedens samlede viden omkring produktet, kunderne, fremstillingsprocesserne osv. samlet i den initielle ideskabelse, og dermed bliver ideernes holdbarhed testet med det samme, og samtidig startes en synergi omkring det nye produkt i virksomheden.

Efter innovation camp'en sammensættes den gruppe, som kommer til at lave den primære udvikling. Hvem det er, afhænger af retningen som sættes på camp'en – det kan være designerne, hvis målet er et helt nyt design, ingeniørerne, hvis det er teknologien, som er fremtrædende osv. Uanset hvem der er primær, vil andre interessenter skulle deltage i den daglige udvikling, og når iterationer og releases planlægges og evalueres.

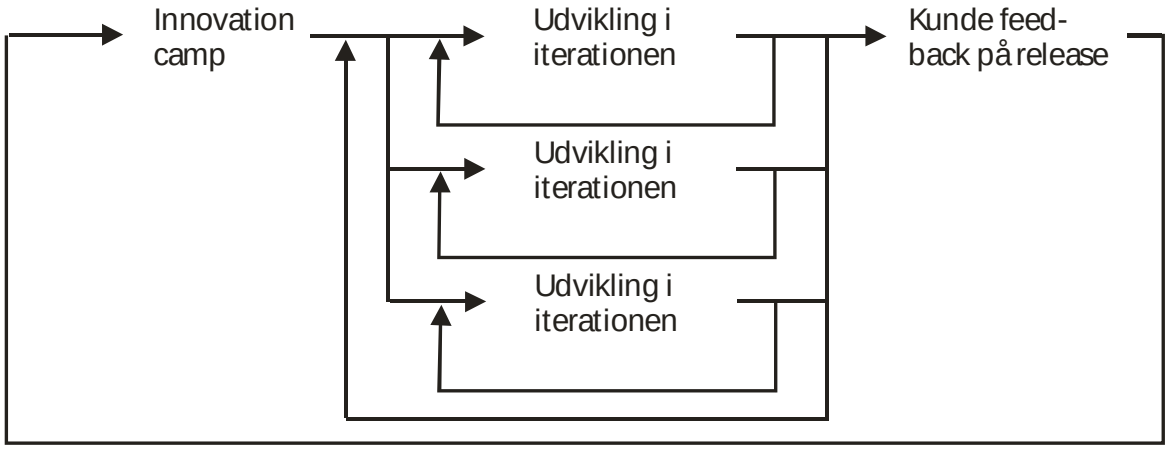
Følges opdelingen i adræt software, vil de nærmeste interessenter og repræsentanter for brugerne være deltagerne i planlægningen af den kommende iteration, mens de rigtige kunders oplevelser samles systematisk sammen efter hvert release og bruges i planlægningen af den kommende release. Efter behov kan innovation camp'en gentages efter hvert eller nogle af release'ne for at øge iderigdommen, men ikke som erstatning for, at alle bidrager med de ideer, der opstår mens der arbejdes med produktet i iterationerne.



Figur 9: Adræt innovationsproces

Mange innovationsprojekter vil ikke kun have udvikling i en dimension, men derimod arbejdes med teknologi, design og processer parallelt. Det vil være for omfattende at have daglige møder med alle involverede, hvorfor den daglige udvikling kan ske i delteams, mens iterationer og releases planlægges sammen, på baggrund af de ideer som opstår i de enkelte teams. En vigtig funktion, når udviklingen er opdelt, er også at holde metaforen/visionen ved lige.

Kilde: Gertsen, F; Hansen, P K; Boer, H (red.) (2006), Innovationsledelse. Center for Industriel Produktion, Aalborg Universitet; ISBN 87-91831-01-6. Open source version http://innovation.cip.dk.



Figur 10: Paralleliseret adræt innovationsproces

Ved at fokusere på at udnytte feedback hos alle, som er involveret i processen, sikres at ideudviklingen forsætter gennem et helt projekt og ikke kun i begyndelsen. Det kan resultere i værdifulde nye features til produktet, til lavere produktionspris osv., og idet ideerne også stammer fra erfaringer gjort med produktet, bliver de meget relevante for brugerne. De enkelte udviklingsforløb kan påvirke hinanden på en lignende måde, som to forskellige dyrearter påvirker hinanden. For eksempel har evolutionen af løver og antiloper sikkert være med til at gøre begge parter bedre i stand til at overleve. I hvert fald mere markante og unikke – egenskaber de fleste ønsker for deres produkter.

Fokus på fungerende produkt

For mange inden for softwareudvikling har det været overraskende, hvilken effekt det har hele tiden at fokusere på at have et produkt, der virker og som får mere og mere funktionalitet efterhånden som iterationer og releases gennemføres. Det er meget lettere for de fleste mennesker at forholde sig til det konkrete produkt frem for en abstraktion over produktet. Det bliver lettere for de fleste at give feedback på de ideer, der ligger til grund for produktet, og få nye ideer, som er relevante for produktet.

At komme derhen, hvor man troede på at det kunne lade sig gøre at have en fungerende version af softwaren med få ugers mellemrum, var vanskeligt i vandfaldsmodellens tid, hvor der først fandtes et samlet system, efter modulerne var integreret. Ligeså langt vil der være for produktionsvirksomheder at tro på, at det kan lade sig gøre at integrere produktionsprocessen i innovationsprocessen på en sådan måde, at der hele tiden er et fungerende produkt.

Min påstand vil være, at man kan komme langt, hvis man tror på effekten af at kunne præsentere et fungerende produkt med korte intervaller – ikke et perfekt produkt i de første mange iterationer, for det har software folkene heller ikke, men et der fungere som det vil gøre ved release. Det at der er nogen, der har vist vejen og har skabt processer og værktøjer, har gjort en stor forskel i softwareudvikling.

I mange år har man kunnet lave produktmodeller af andre materialer, IT simulation er blevet mere og mere virkeligt med årene, produktionslinier bliver mere og mere fleksible med hensyn til at kunne lave mindre og mindre stykstørrelser osv. Der er rigtig mange muligheder til rådighed allerede i dag – og jeg tror på at alle vil kunne finde en kombination, som kan give de ønskelige effekter af feedback, når de, der skal udføre det, tror på, at det kan lade sig gøre.

Det seneste skud på stammen i forhold til at kunne lave produktioner med lave styktal – faktisk enkelt produktion – er den såkaldte FAB[xi], der er en slags ”3D printer”, som bygger produktet op af lag på lag af plastic, metalpulver eller andre materialer. I dag er en FAB så

Kilde: Gertsen, F; Hansen, P K; Boer, H (red.) (2006), Innovationsledelse. Center for Industriel Produktion, Aalborg Universitet; ISBN 87-91831-01-6. Open source version http://innovation.cip.dk.

billig som $20.000, men det forudsiges, at den vil følge samme udvikling som den personlige computer og ende med en pris på $1.000 og dermed blive hvermands eje. Og hvorfor ikke? Direktøren i IBM, Thomas Watson sagde i 1943 at "jeg tror, der er et verdensmarked for måske 5 computere".

Mulighederne er mange, men naturligvis er det en udfordring at bringe det til det ekstreme, som de adrætte softwaremetoder har gjort, hvor produktet er i en ny stabil tilstand hver 2-4 uger.

Iterativ innovation

For mange produkter og virksomheder vil det virke uoverskueligt at skulle release nye versioner af produkter hver 1.-3. måneder, men f.eks. Apple har siden den første iPod så dagens lys for 4 år siden, leveret 4 generationer (iPod, iPod mini, iPod shuffle og iPod nano), samtidig med at kapaciteten er vokset i de enkelte generationer. De har fået farveskærme, de kan afspille flere formater bl.a. billeder og video, der er kommet en special brandet version med U2 osv.

Ikke alle spring har været lige store: Mellem generationerne har der været arbejdet meget med designet, mens der inden for en generation er blevet ændret på nogle hardwarekomponenter og lagt nye features ind i softwaren osv.

Men summen har givet et hav af releases og et kæmpe salg, og selv om Apple ikke har været prisførende, har de alligevel taget en meget stor del af markedet – 80% af det amerikanske marked i oktober 2004[xii]. De har simpelthen ikke levnet konkurrenterne nogen chance, fordi de samtidig med den hurtige udvikling i produktet har lavet innovation med processerne, således at konkurrenterne nu også har svært ved at være med på prisen.

Det har uden tvivl krævet meget hos Apple at nå op på den meget høje releasefrekvens, men det har også givet resultater. Den første forudsætning for at nå dertil er, at man tror på, at det kan lade sig gøre, og man er parat til at indrette produktionen efter innovationstakten. Altså at produktionen hurtigt kan lægges om til at producere de nye releases, uanset hvilke forandringer der kommer i produktet. Det er naturligvis lettere, når produktionen deltager i planlægning og evaluering af iterationerne, da de får muligheden for at lave den innovation, der skal til hos dem imellem releasene. Derfor vil produktionstunge produkter kræve, at produktionen har deres egen sløjfe i figur 10, som koordineres for hver iteration.

Hvis det kan lykkes for produktionen og de andre spillere i innovationsprocessen at komme op i en frekvens, der ligner adræt software udvikling og lytte til den feedback, som kommer fra andre interessenter undervejs, hvorfor skulle det så ikke kunne lykkes at frembringe produkter hurtigere, billigere, med bedre kvalitet og med bedre understøttelse af brugernes behov, som det lykkes for adrætte softwareprojekter?

Konklusioner

Faseopdelte innovationsprocesser har måske haft sin berettigelse, dengang forandringshastigheden var lavere end i dag. Hastigheden og det faktum, at produkter er langt mere komplekse i dage, med både hardware og software, mekanik og elektronik, med design som en stadig vigtigere faktor osv., gør, at vi må tænke anderledes. Evolutionen og adræt softwareudvikling har vist, at en elegant og effektiv måde at håndtere den store kompleksitet på, er med iterative processer.

Selvfølgelig kan det lade sig gøre at erstatte den traditionelle faseopdelte innovationsproces med en mere evolutionær og adræt proces. Fremfor at til- og fravalg i processen er uigenkaldelige, giver en høj grad af gentagelse af processerne mulighed for at tilpasse produktet løbende til de feedback, som kan etableres fra forskellige dele af virksomheden, for eksterne rådgivere etc. og selvfølgelig fra kunderne.

Kilde: Gertsen, F; Hansen, P K; Boer, H (red.) (2006), Innovationsledelse. Center for Industriel Produktion, Aalborg Universitet; ISBN 87-91831-01-6. Open source version http://innovation.cip.dk.

Tanken i adræt softwareudvikling er jo mere jo bedre, så det gælder om at få frekvensen og kvaliteten i feedback og ideindsamling op. Når de øges, vil mange mindre forbedringer i produkt, i fremstillingsprocesser, i positionering på markedet etc. tilsammen give en forandringshastighed, som kan sætte konkurrenterne pas. De mange små ændringer vil blive til store forandringer for brugerne, og det kan reelt set fra brugernes og konkurrenternes vinkel være én radikal innovation, selvom virksomheden selv betragter det som inkrementel innovation.

Samtidig giver en ekstrem iterativ proces mange chancer for at lære og blive dygtigere til at gennemføre processen. Kilden til den oprindelige viden er gået tabt på vej fra Kina eller Grækenland, men førende amerikanske universiteter har eftervist, at der skal 21 gentagelser til, før man behersker noget excellent. Alene af den grund er en iterativ proces, hvor forløbet gentages ugentligt eller månedligt meget mere attraktiv end en proces, der varer 2 år. Deltagere i 2 årige processer skal faktisk bruge et helt arbejdsliv på at blive excellent for blot at være klar til pensionen.

Så selvom det virker umuligt at lægge alle processer om til at følge den adrætte innovations proces med korte iterationer og kort tid mellem hver release, så gør det alligevel før dine konkurrenter. Innovation er en vigtig konkurrenceparameter i vores del af verden, hvor omkostningerne er høje, og den, der kommer først, får den bedste pris for sine varer.

Gør som du plejer at gøre, og du vil opnå de resultater du plejer. Vil du have andre resultater, så gør noget nyt - for eksempel adræt innovation.

Kildeliste

[i] 1. Mosebog 1,1-2
[ii] Charles Darwin : Arternes oprindelse, 1859
[iii] Charles Darwin : Menneskets afstamning, 1871
[iv] John Bicheno : The New Lean Toolbox, 2004
[v] Kent Beck: Extreme Programming Explained. 1999
[vi] www.agilealliance.com, www.agilemanifesto.org
[vii] Ken Schwaber: Agile Software Development with SCRUM, 2001
[viii] Dave Francis & John Bessant : Targeting innovation and implications for capability development, 2004
[ix] Robert G. Cooper : Winning at New Products, 1993
[x] Roberto Verganti: Managing Design Driven Innovation for Competitive Advantage, 2005.
[xi] Neil Gershenfeld: FAB. The Coming Revolution on Your Desktop – from Personal Computers to Personal Fabrication, 2005
[xii] http://quote.bloomberg.com/apps/news?pid=10000103&sid=a58iozj_2jXM

Om forfatterne:

Søren Raaschou, direktør i Posility, har gennem en årrække arbejdet med ledelse, projektledelse og udvikling af processer i en række af Danmarks førende virksomheder. Han er uddannet civilingeniør, Master in Management of Technology, og indenfor NLP, og arbejder med opgaver hvor teknologi, processer og mennesker spiller tæt sammen. Senest har han fokuseret på at implementere adrætte udviklingsprocesser og adræt projektledelse.

Kilde: Gertsen, F; Hansen, P K; Boer, H (red.) (2006), Innovationsledelse. Center for Industriel Produktion, Aalborg Universitet; ISBN 87-91831-01-6. Open source version http://innovation.cip.dk.