Prioriteringssyndromet

Bjørn Henrik Vangstein, FINN.no

14. august 2024

1 Prioriteringsproblemer?

Hva gir mest verdi: å fokusere på få initiativer som gjennomføres raskt, eller å ha mange initiativer i gang samtidig for å utnytte kapasiteten maksimalt? I Lean-filosofien er *one piece flow* et viktig prinsipp for effektive prosesser. *Få initiativer raskt* er altså Leans svar, men dette står i kontrast til masseproduksjonsfilosofien, som baserer seg på prinsippet om stordriftsfordeler *economy of scale*. Spørsmålet skaper ofte fastlåste, dogmatiske debatter. Det er synd; diskusjonen burde ikke handle om dogmatikk, men heller om hvilken metode som objektivt sett skaper mest verdi over tid.

Kontinuerlig forbedring har et rikt utvalg av metoder og verktøy som berører ledelse, organisering og arbeidsprosesser. Hensikten er som regel, hvis vi ser stort på det, å gjøre organisasjoner i stand til skape mest mulig verdi for kunder og brukere med den tiden og kapasiteten vi har tilgjengelig. Vi prøver å fjerne hindringer i verdistrømmen, men effekten av gode prosesser kan raskt ødelegges av uheldige *prioriteringer*. Prioriteringsproblemer er et velkjent tema, spesielt i større organisasjoner. Det burde ikke forundre oss - det å prioritere kan være en kompleks oppgave med mange aspekter å ta stilling til, som blant annet:

- hva som prioriteres[1]
- kriteriene[2] vi prioriterer etter og vektingen av disse
- hvor ofte vi skal prioritere
- hvem som skal prioritere
- hvor mye vi skal prioritere av gangen
- størrelsen på det som prioriteres
- rekkefølgen på det som prioriteres
- kapasitetsfordelingen[3] mellom det som prioriteres

Uansett hvordan vi velger å løse denne oppgaven[4], får prioriteringene våre stor innvirkning på verdiskapingen i de etterfølgende prosessene. Det å prioritere er strengt tatt en optimaliseringsoppgave, og diskusjoner om prioritering dreier seg ofte om hva og hvordan vi skal optimalisere. Denne problemstillingen er det derfor god grunn til å skape mer klarhet i. Det er særlig tre aspekter ved prioritering som påvirker flyten av verdi i organisasjonen.

1.1 Å prioritere feil initiativer

Å sette i gang initiativer som ikke skaper verdi, eller som er ugjennomførbare, ødelegger en organisasjons produktivitet. Det hjelper lite å ha velfungerende organisasjoner og prosesser hvis de settes til å gjøre feil ting. Når vi jobber med kontinuerlig forbedring, utfordrer vi først og fremst hvordan vi jobber. Det er imidlertid enda viktigere å være kritisk til hva vi jobber med. Tydelige mål vil gjøre det atskillig lettere å skille gode initiativer fra de mindre gode. Tydelige mål er likevel ikke nok - mål bør også prioriteres.

[1] Initiativer, oppgaver, tiltak, prosjekter, produkter, idéer m.m.
[2] Hva vi skal oppnå; våre mål
[3] Også kalt ressursfordeling.
[4] Ikke sjelden "løses" denne oppgaven ved rett og slett å la være å prioritere.

1.2 Å prioritere for sjelden

Det vil alltid være en viss risiko for at vi setter i gang feil initiativer, men ved å prioritere hyppig får vi muligheten til å stoppe dem tidlig. Å prioritere hyppigere og fokusere på færre initiativer vil forenkle prioriteringsprosessen, og redusere risikoen for og konsekvensene av uheldige prioriteringer. Verden endrer seg kontinuerlig, derfor er prioriteringer til en viss grad ferskvare - de bør ha en utløpsdato.

1.3 Å sette i gang for mange initiativer

Det er vanligere å høre argumenter for å ha mange initiativer i gang samtidig enn det motsatte. Det ender gjerne med det vi kan kalle haglegevær-prioriteringer hvor vi setter i gang mange initiativer uten innbyrdes prioritering. På denne måten vil det, om ikke annet, være nok arbeid til alle, slik at vår dyrbare kapasitet blir utnyttet fullt ut. Med mange initiativer i gang kan vi jo håpe på at i alle fall noen av dem vil lykkes.

Som vi snart vil se, kan det å sette i gang mange parallelle initiativer gi oss en kostbar organisatorisk forstoppelse. Kostnaden ved å være treg - *forsinkelseskostnaden* - kan overstige kostnaden ved å ha lav kapasitetsutnyttelse. Dessverre er kostnaden ved å være treg sjelden noe vi legger vekt på når vi prioriterer. Ifølge Donald G. Reinertsen er forsinkelseskostnaden den viktigste faktoren å måle i organisasjoner som driver med produktutvikling: "If you only quantify one thing, quantify the cost of delay" [1]. Forsinkelseskostnaden kan være en nøkkel til å gjøre bedre prioriteringer.

2 Noen eksempler med tvillingselskapene A og B

For å kaste lys over problemstillingen starter vi med noen konkrete eksempler. Her brukes omsetning[5] (penger) som parameter for verdiskaping. Senere skal jeg vise hvordan teorien kan brukes når vi prioriterer initiativer som vi ikke kan knytte omsetning til.

Se for deg to selskaper (Selskap A og Selskap B) som i det store og hele er like. De utvikler de samme produktene som lanseres i like markeder. De har samme utviklingskapasitet, kompetanse og erfaring. Begge selskaper har kapasitet til å lansere to produkter pr. år; produktene **P1** og **P2**. Disse produktene er like arbeidskrevende å utvikle, og gir samme omsetning fra og med den dagen de blir lansert. Vi antar at begge produktene krever 6 måneders arbeid å utvikle med full kapasitet, og at de omsetter for 1 million hver pr. måned. Når begge produktene er lansert, omsetter derfor selskapene for 2 millioner pr. måned, altså 24 millioner pr. år. Imidlertid skiller Selskap A og B seg fra hverandre på to punkter: de prioriterier og organiserer utviklingsarbeidet ulikt, noe som påvirker de økonomiske resultatene betydelig.

2.1 Selskap A utvikler produkter sekvensielt

Selskap A har som prinsipp å utvikle kun ett produkt av gangen. Først utvikler og lanserer de P1, før de setter i gang utviklingen av P2. De har kapasitet til å lansere P1 etter 6 måneder og P2 etter 12 måneder. P1 gir derfor en omsetning på 6 millioner det første året. Det andre året har selskapet begge produktene i markedet, og omsetter derfor for 24 millioner. Se figur 1.

Fordi selskap A utvikler ett produkt av gangen, trenger de bare å administrere ett utviklingsprosjekt[6]. Status må rapporteres en gang per måned så lenge produktet er i utviklingsfasen, frem til lansering. I selskap A rapporterer de derfor status 12 ganger per år.

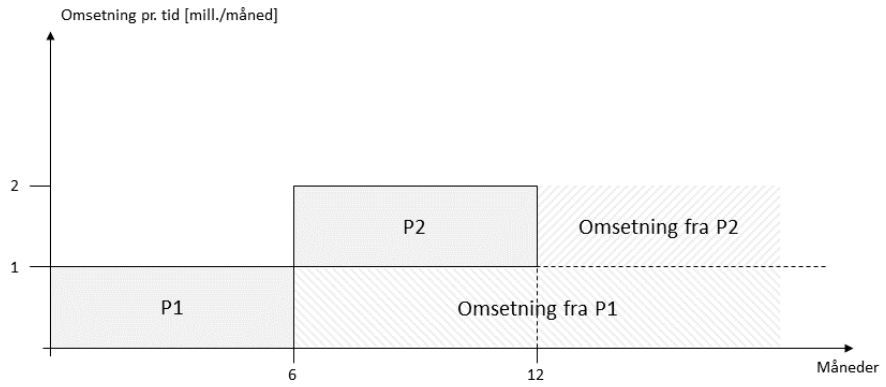
2.2 Selskap B utvikler produkter parallelt

Selskap B har som prinsipp å spre kapasiteten på mange parallelle utviklingsinitiativer. Når et produkt er ferdig, flytter de *ikke* kapasitet til andre pågående utviklingsprosjekter, men setter heller i gang utviklingen av et nytt produkt.

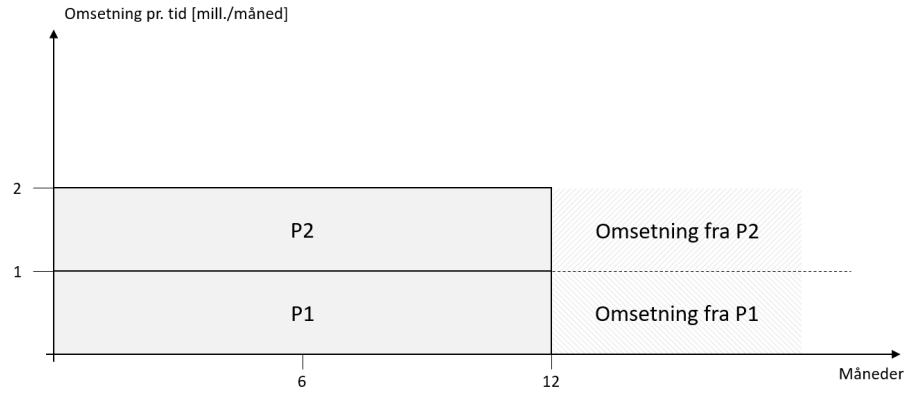
Selskap B fordeler i dette tilfellet utviklingskapasiteten likt mellom P1 og P2, og starter utviklingen av produktene samtidig. Derfor lanserer de både P1 og P2 etter 12 måneder. Selskap B får ingen omsetning det første året, men de omsetter for 24 millioner det andre året, som selskap A. Se figur 2. Selskap B har samme rapporteringsregime som selskap A, men må altså administrere to utviklingsprosjekter samtidig. De rapporterer derfor status 24 ganger per år.

[5] Inntekt fra produkter og tjenester som våre kunder betaler for å bruke.

[6] Ordet 'prosjekt' benyttes her uten å referere til noen spesifikk metodikk, verken vannfall eller smidig. Det brukes kun for å beskrive en tidsavgrenset arbeidsperiode, fra oppstart til produktet er klart for markedet og skaper verdi.



Figur 1: Selskap A utvikler ett produkt av gangen (sekvensielt)



Figur 2: Selskap B starter opp utviklingen av begge produktene samtidig (parallelt).

2.3 Resultatene

Selskap A tjener 6 millioner mer enn selskap B det første året. Hvis vi nå antar at selskapene gjentar prosessen og utvikler to nye produkter hvert år, vil selskap A omsette for 6 millioner mer enn selskap B hvert år - til tross for at de begge lanserer like mange produkter årlig. Selskap Bs metode gir en *forsinkelseskostnad* på 6 millioner hvert år. Selskap A har lavere administrasjonskostnader, og får derfor også et bedre bunnlinjeresultat. Senere skal vi se at de også oppnår andre gevinster.

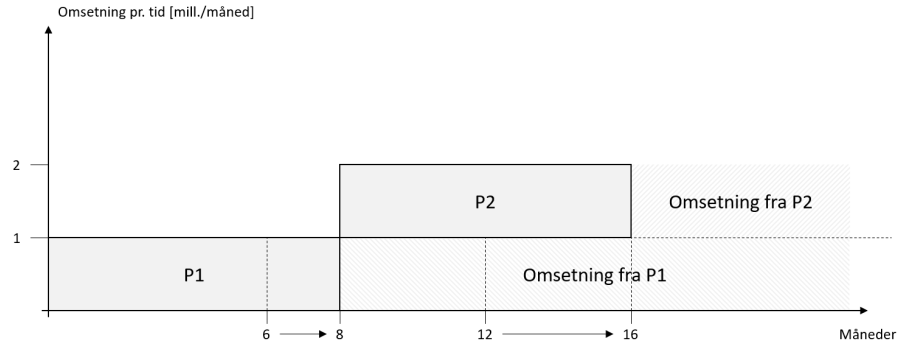
2.4 Innvendinger mot eksempelet

Det er fristende hevde at eksempelet over er for idealisert, og derfor irrelevant. Vi skal derfor drøfte to åpenbare innvendinger.

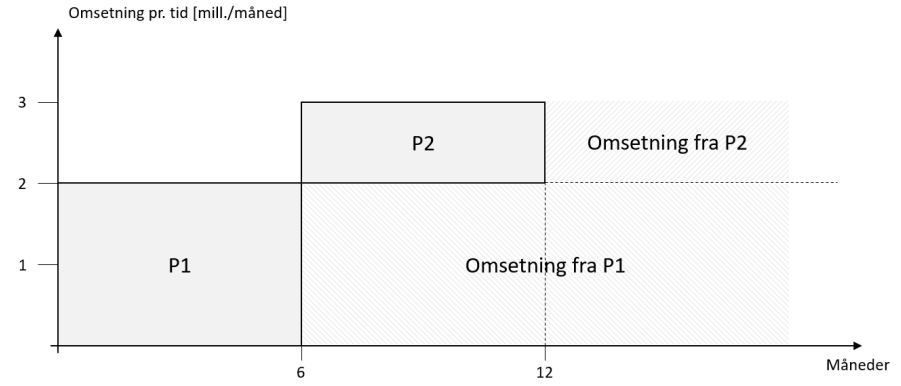
2.4.1 Effekten av redusert kapasitetsutnyttelse med sekvensiell utvikling

Vi kan for hevde at selskap A i praksis ikke ville klare å utnytte all kapasitet hvis de kun arbeider med ett utviklingsinitiativ. Dette er ikke nødvendigvis et sterkt argument. Vi kan like gjerne hevde det motsatte - at selskap B ikke vil kunne utvikle et produkt på kun dobbelt tid med halv kapasitet, men for argumentets skyld skal vi undersøke nærmere hvordan redusert ressurseffektivitet påvirker resultatet.

La oss si at selskap A ikke klarer å utnytte kapasiteten sin fullt ut hvis alle skal jobbe på ett og samme utviklingsprosjekt. Hvis effektiviteten faller med for eksempel 25%, vil det ta åtte måneder å lansere P1, ikke seks. Se figur 3. De rekker riktig nok å tjene fire millioner før selskap B har tjent noe, men selskap A lanserer nå P2 fire måneder inn i det neste året. Dermed tapes ekstrainntekten fra den raske lanseringen av P1. Selskap A lanserer nå to produkter hver 16. måned, mens selskap B lanserer to produkter hver 12. måned. Til tross for fallet i lanseringstakt, har selskapene samme årlige omsetning. Selskap A har fortsatt lavere administrasjonskostnader, og derfor høyere lønnsomhet. I dette eksempelet kan altså selskap A tåle en reduksjon i ressurseffektivitet på 25% og fortsatt oppnå samme omsetning som selskap B. Hvor stort effektivitetstap de kan tåle med sekvensiell utvikling er altså avhengig av produktporteføljen.



Figur 3: Effekten av redusert kapasitetsutnyttelse ved sekvensiell utvikling i Seskap A.



Figur 4: Selskap A har produkter med ulik omsetning. P1 omsetter for det dobbelte av P2.

2.4.2 Forbedring og vedlikehold etter lansering

Premisset om at all kapasitet kan frigjøres umiddelbart etter lansering kan også utfordres. I praksis er vi ikke ferdige med produkter og tjenester når de lanseres. Vi må fortsatt bruke kapasitet på forbedringer og vedlikehold. Hensikten med dette eksempelet er imidlertid å sammenligne de teoretiske forskjellene i resultatene med sekvensiell og parallell utvikling. Behovet for forbedring og vedlikehold har vi både med sekvensiell og parallell utvikling, og det kan derfor neglisjeres i denne sammenligningen. I 3 forklarer jeg hvordan teorien kan brukes i praksis.

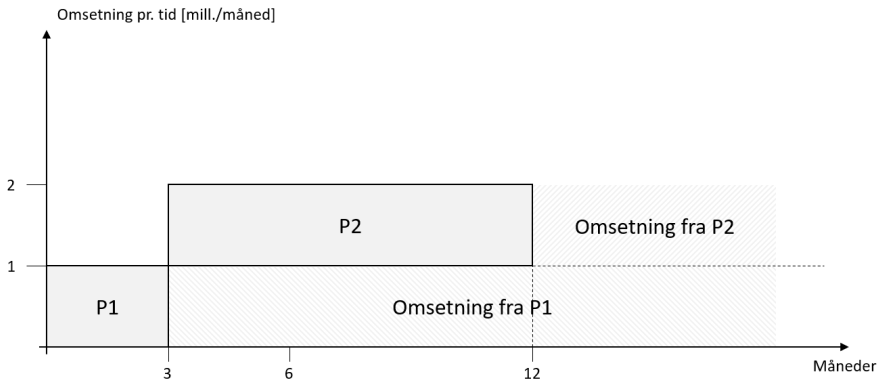
2.5 Differensiering av porteføljen

Eksempelet over er en grov forenkling av praktiske realiteter, men resultatene er likevel interessante nok til at det er verdt å undersøke prinsippene nærmere. I eksempelet over ga produktene P1 og P2 samme omsetning og hadde samme utviklingsomfang. Hva skjer med resultatene hvis vi nyanserer produktporteføljen?

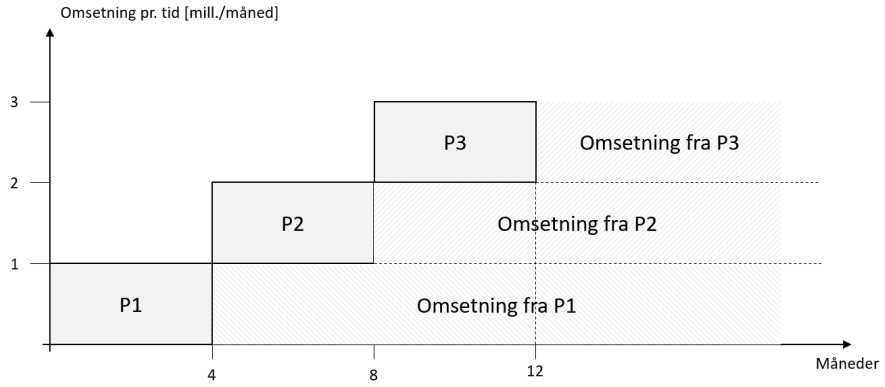
2.5.1 Når P1 og P2 gir ulik omsetning

Vi antar nå at P1 omsetter for det dobbelte av P2, dvs. 2 millioner per måned fra og med lansering. Vi ser at gevinsten ved å være tidlig ute dobles. Se figur 4. Selskap A vil rekke å tjene 12 millioner det første året hvis de utvikler P1 først. Året etter vil begge selskaper omsette for 36 millioner, og selskap Bs forsinkelseskostnad utgjør 33 % av denne omsetningen. Vi ser at når produktenes arbeidsomfang er likt, vil det lønne seg å utvikle produktet med høyest omsetning først, og at sekvensiell utvikling blir enda mer lønnsomt enn parallell utvikling. Selv om vi hadde valgt den minst lønnsomme rekkefølgen (P2 før P1), vil sekvensiell utvikling i dette tilfellet ha vært mer innbringende enn å utvikle produktene parallelt. I dette eksempelet ville Selskap A tåle et ressurseffektivitetsfall på 50 %[7] uten å komme dårligere ut enn Selskap B.

[7]Det vil si at utviklingstiden på både P1 og P2 ville ha økt fra 6 til 9 måneder.



Figur 5: Selskap A har produkter med ulikt arbeidsomfang.



Figur 6: Selskap A utvikler tre produkter sekvensielt.

2.5.2 Når P1 og P2 har ulikt arbeidsomfang

Vi antar nå at P1 og P2 gir samme omsetning, men har ulikt arbeidsomfang, med henholdsvis 3 og 9 måneders utviklingstid med full kapasitet. Se figur 5. Hvis selskap A utvikler P1 før P2, rekker de å tjene 9 millioner før det første året er omme. Hvis de snur rekkefølgen, vil de tjene 3 millioner. Vi ser nå at hvis produktene har ulik omsetning og størrelse, kan prioriteringsrekkefølgen ha stor betydning.

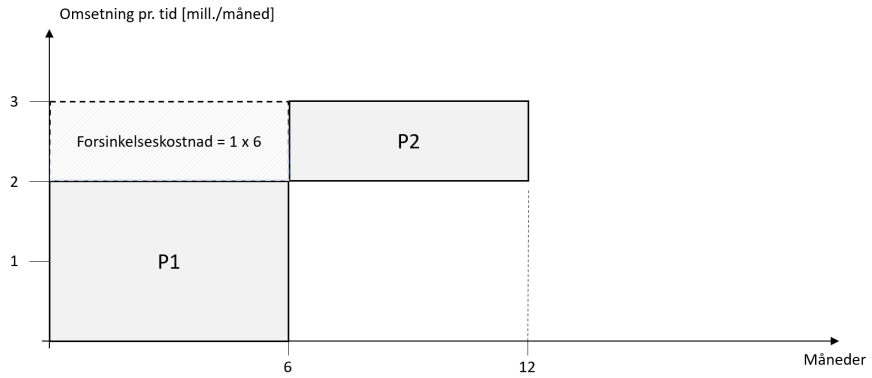
2.5.3 Flere produkter i produktporteføljen

Så langt har vi sammenlignet resultatene i Selskap A og B med kun to produkter i porteføljen, men hvordan blir resultatene med flere produkter? Se figur 6. Med tre like produkter, ser vi at selskap As fordel blir enda større. Hvis både P1, P2 og P3 omsetter for 1 million per måned og krever 4 måneders utviklingsarbeid, vil selskap A omsette for 12 millioner det første året (P1 gir 8, P2 gir 4). Selskap B fordeler fortsatt kapasiteten likt mellom utviklingsprosjektene, og lanserer alle tre produktene etter 12 måneder. De får derfor ingen omsetning det første året.

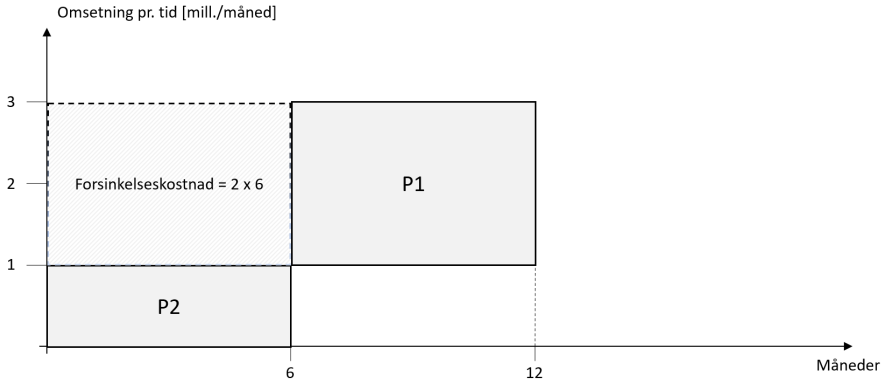
Hvis vi øker til fire produkter med 3 måneders utviklingsarbeid per produkt, tjener selskap A 18 millioner (9+6+3) det første året. Det er tydelig at med flere produkter i porteføljen blir fordelen av seksensiell utvikling større - så lenge ikke ressurseffektiviteten faller for mye. I teorien vil gevinsten ved sekvensiell utvikling av like produkter nærme seg 50% av den totalte porteføljeverdien når antallet produkter øker. Hvis produktene er ulike, kan gevinsten bli enda større.

2.6 Optimal rekkefølge med sekvensiell utvikling

Vi har analysert omsetningen til selskap A med noen konkrete eksempler. Selskap As prioritering handlet om å finne optimal rekkefølge på produktinitiativene. Det er relativt enkelt å se hvilken rekkefølge som er mest innbringende når vi har få produkter og enkle tall å forholde oss til. Vi skal nå generalisere analysen slik at den gjelder alle kombinasjoner av omsetning, omfang og antall produkter.



Figur 7: Forsinkelseskostnad ved sekvensiell utvikling. P2 venter på P1



Figur 8: Forsinkelseskostnad ved sekvensiell utvikling. P1 venter på P2

En ulempe ved analysen over var at vi måtte beregne omsetningen *i en gitt periode* for å kunne sammenligne resultatene. Dette er egentlig ikke nødvendig fordi forskjellene i resultatene oppstår allerede i oppstartsfasen. Ved å beregne *forsinkelseskostnaden* blir analysen enklere, det vil si omsetningen vi taper i perioden fra et produkt er prioritert til det blir lansert. Dette tidsrommet bestemmes av produktets utviklingstid og utviklingstiden til eventuelt andre produkter som står foran i køen. Kjenner vi forsinkelseskostnaden trenger vi ikke å vite hvor lenge produktene skal leve i markedet for å finne den optimale prioriteringsrekkefølgen.

Metoden for hvordan vi finner rekkefølgen som gir lavest forsinkelseskostnad (og derfor høyest omsetningsvekst) er velkjent. Reinertsen [2] har kalt regelen for **WSJF** - *weighted shortest job first*. Black Swan Farming [3] bruker forkortelsen **CD3** - *Cost of Delay Divided with Duration*.

Svaret er at vi må utvikle produktet med

høyest forsinkelseskostnad pr. utviklingstid først

for så å utvikle produktet med nest høyest forsinkelseskostnad pr. utviklingstid, osv. Forsinkelseskostnaden er i denne sammenhengen produktets løpende omsetning i den aktuelle perioden, dvs. omsetning pr. tid (f. eks. kr/måned). Vi må med andre ord gi høyest prioritet til det produktet som gir oss *raskest vekst i den løpende omsetningen*. Se figur 7 og 8. Vær oppmerksom på at dette ikke nødvendigvis er produktet med høyest løpende omsetning, men produktet med høyest løpende omsetning i forhold til produktets utviklingstid.

Den matematiske forklaringen på denne regelen har jeg vist i appendiks A.1. Optimal rekkefølge med et vilkårlig antall n produkter, kan vi skrive slik:

$$\frac{c_1}{T_1} > \frac{c_2}{T_2} > \frac{c_3}{T_3} \cdots > \frac{c_n}{T_n}$$

Hvor $c_1, c_2, \ldots, c_n$ er hvert produkts spesifikke forsinkelseskostnad. $T_1, T_2, \ldots, T_n$ er utviklingstiden for produktene. Vi utvikler altså produktet med den høyeste verdi på brøken $\frac{c}{T}$ først.

Dette prinsippet gjelder ikke bare produktutvikling; det kan brukes på alle slags initiativer så lenge vi kan tallfeste verdien og gjennomføringstiden på initiativene vi skal prioritere. Vi skal senere se hvordan dette kan gjøres i praksis.

2.6.1 Produkter med lik omsetning eller lik utviklingstid

Hvis vi bruker denne regelen ser vi at når produktene har lik omsetning pr. tid, er det kun utviklingstiden som bestemmer rekkefølgen. Produktet som krever *kortest utviklingstid utvikles først.* Hvis produktene ikke har lik omsetning, men samme utviklingstid, er det omsetningen pr. tid som bestemmer fordelingen. Vi utvikler produktet med *høyest løpende omsetning først.*

2.6.2 Eksempel på bruk av regelen for optimal rekkefølge ved sekvensiell utvikling

Vi gjenbruker et av eksemplene over, hvor P1s løpende omsetning var 2 millioner per måned, og P2 omsatte for 1 million per måned. Begge produktene krevde 6 måneders utviklingstid med full kapasitet. Prioriteringsfaktoren for P1 blir:

$$\text{P1: 2 millioner per måned/6 måneder} = 0,33 \text{ millioner [kroner per måned}^2\text{]}$$[8]

P2s faktor blir:

$$\text{P2: 1 million per måned/6 måneder} = 0,167 \text{ millioner [kroner per måned}^2\text{]}$$

Konklusjonen er altså at det er mest lønnsomt å utvikle P1 først.

Vi legger nå til et tredje produkt i porteføljen; **P3** omsetter for 1 million per måned og krever 4 måneders utviklingstid.

P3s prioriteringsfaktor blir:

$$\text{P3: 1 millioner per måned/4 måneder} = 0,25 \text{ millioner [kroner per måned}^2\text{]}$$

Vi ser at inntektsstrømmen øker raskest ved å utvikle P3 før P2. Optimal rekkefølge blir derfor:

$$P1, P3, P2.$$

Samlet forsinkelseskostnad (i millioner) blir $2 \times 6 + 1 \times (6 + 4) + 1 \times (6 + 4 + 6) = 38$ med denne rekkefølgen. Den minst optimale rekkefølgen er $P2, P3, P1$, som vil gi oss en samlet forsinkelseskostnad på 48 millioner.

2.7 Kapasitetfordeling ved parallell utvikling

Hittil har vi antatt at selskap B fordeler kapasiteten likt mellom P1 og P2 (50 % til hver), men er denne fordelingen optimal hvis P1 og P2 ikke er like? Jeg minner om at når Selskap B har fått et produkt er ferdig, flytter de *ikke* kapasitet til andre pågående utviklingsprosjekter, men starter i stedet utviklingen av et nytt produkt.

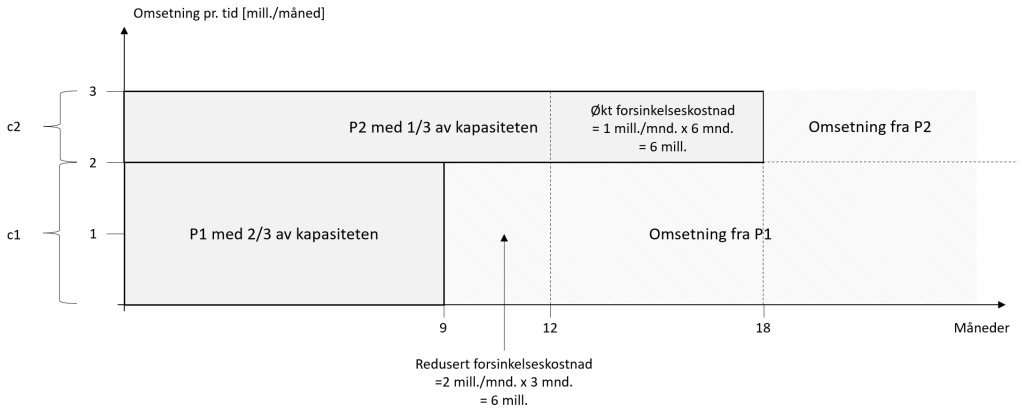
2.7.1 Produkter med ulik omsetning

Vi antar fortsatt at produktet P1 omsetter for det dobbelte av P2, med henholdvis 2 og 1 millioner per måned, og at de er like arbeidskrevende å utvikle. Inspirert av resultatene i selskap A, kan det være fristende for selskap B å tildele P1 en større andel av kapasiteten for å lansere det mest innbringende produktet tidligere. Hvis de for eksempel gir P1 dobbelt så stor kapasitet som P2, henholdsvis $\frac{2}{3}$ og $\frac{1}{3}$ av utviklingskapasiteten, vil P1 bli lansert allerede etter 9 måneder. Se figur 9. Selskap B vil tjene 6 millioner det første året, men med kun $\frac{1}{3}$ igjen av kapasiteten til P2, vil P2 lanseres først etter 18 måneder - 6 måneder senere enn med lik kapasitetsfordeling. Med denne kapasitetsfordelingen ville selskap B ha tapt like mye på en forsinket lansering av P2 som de ville ha tjent på å lansere P1 tidligere.

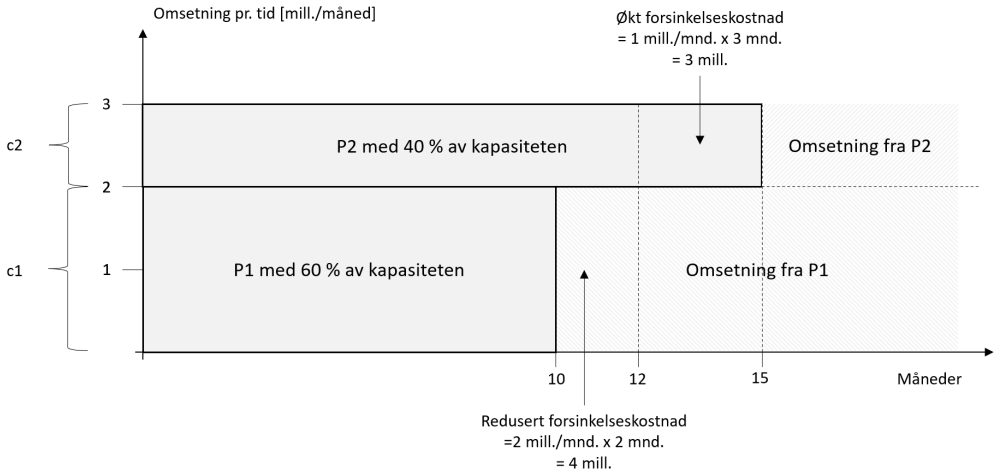
Hvis vi gir enda mer kapasitet til P1, blir resultatet verre. Med 3 ganger så mye kapasitet til P1, med henholdsvis $\frac{3}{4}$ og $\frac{1}{4}$, vil P1 bli lansert etter 8 måneder, og selskapet vil omsette for 8 millioner det første året. Imidlertid vil P2 bli ferdig først etter 24 måneder, og de taper i alt 4 milloner sammenlignet med den opprinnelige kapasitetsfordelingen.

Den optimale kapasitetsfordelingen mellom P1 og P2 er omtrent 59/41. Da vil produktene bli lansert etter henholdsvis 10 og 15 måneder. Det gir 1 million lavere forsinkelseskostnad enn med en 50/50-fordeling. Se figur 10. Hvis begge selskaper prioriterer optimalt får selskap B en forsinkelseskostand som er 11 millioner (23 %) høyere enn selskap A. Selv om selskap A utviklet produktene i den *minst* optimale rekkefølgen ville de ha tjent 5 millioner mer enn selskap B med optimal kapasitetsfordeling.

[8] Forklaringen på denne enheten finner du Appendiks A



Figur 9: Forsinkelseskost ved parallell utvikling med 2:1-fordeling av kapasiteten - vinningen går opp i spinningen.



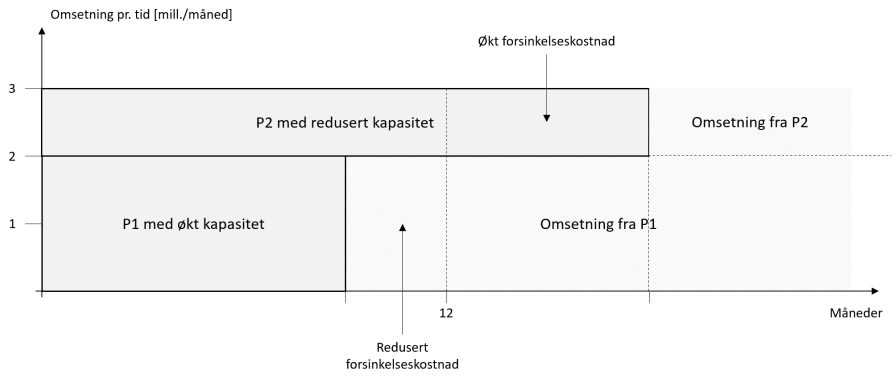
Figur 10: Forsinkelseskost ved parallell utvikling med 3:2-fordeling av kapasiteten gir 1 mill. i gevinst.

2.7.2 Produkter med ulikt arbeidsomfang

Hvis produktene har samme omsetning, men ulikt arbeidsomfang, vil det heller ikke være optimalt å fordele kapasiteten likt. Hvis vi ser på eksempelet over med henholdsvis 3 og 9 måneder utviklingstid, og lik omsetning, vil en 37/63-fordeling være nære det optimale, men Selskap B må nå gi den *største* andelen av kapasiteten til det produktet som krever *mest* arbeid. Med denne produktporteføljen vil selskap Bs forsinkelseskostnad bli 7 millioner (+49 %) enn i selskap A når begge selskapene prioriterer optimalt. Hvis selskap A av en eller annen grunn utvikler P2 først, vil de fortsatt tjene mer enn selskap B. Vi begynner kanskje å ane at det er mer komplisert å prioritere i selskap B.

2.8 Optimal kapasitetsfordeling med parallell utvikling

Med parallell utvikling setter vi i gang alle initiativene samtidig, og oppgaven er å finne den kapasitetsfordelingen mellom produktene som er mest lønnsom - med andre ord fordelingen som gir den lavest samlede forsinkelseskostnaden for hele porteføljen. Vi så i eksemplene over at dette er en balanseringsøvelse; når vi flytter kapasitet fra ett produkt til et annet, vinner vi tid på det ene, men taper på det andre. Se figur 11. Som vi ser kan en slik omfordeling gi store utslag, og forholdet mellom gevinst og tap er ikke helt opplagt. Hvis vi for eksempel gir ett produkt 99% av kapasiteten, vil det andre produktet sitte igjen med 1%. Da blir utviklingstiden 100 ganger så lang, og det kan vi åpenbart tape mye penger på. Vi kan selvsagt flytte kapasitet fra P1 til P2 når P1 er ferdig. Problemet med dette resonnementet er at hvis vi har fleksible ressurser som kan flyttes fritt mellom utviklingsprosjektene, burde vi utvikle produktene sekvensielt, ikke parallelt. Det skal jeg begrunne senere.



Figur 11: Parallell utvikling. Når vi flytter kapasitet fra ett prosjekt til et annet, vinner vi tid på det ene, men taper på det andre.

I appendiks A.2 har jeg vist at den optimale kapasitetsfordelingen mellom to produkter er:

$$\frac{r_1}{r_2} = \sqrt{\frac{c_1 T_1}{c_2 T_2}}$$

Hvor r_1 og r_2 er produktenes andel av den samlede utviklingskapasiteten. c_1 og c_2 er den spesifikke forsinkelseskostnaden til hvert produkt. T_1 og T_2 er utviklingstiden for produktene.

Ved parallell utvikling beregner vi prioriteringsfaktoren ved å multiplisere forsinkelseskostnaden med utviklingstiden. Dette tallet viser den akkumulerte forsinkelseskostnaden per produkt i utviklingsfasen. Forholdet mellom disse faktorene avgjør optimal kapasitetsfordeling. Vi må altså fordele kapasitet etter løpende omsetning kombinert med utviklingstid, og gi

mest kapasitet til produktet med høyest løpende omsetning multiplisert med utviklingstid.

I formelen inngår det også en kvadratrot. Det gjør kapasitetsfordelingen ulineær, det vil si at det er mindre intuitivt hva som er optimal kapasitetsfordeling hvis vi utvikler produktene parallelt.

2.8.1 Tommelfingerregler kapasitetsfordeling mellom parallelle initiativer

- Gi mest kapasitet til produktet med høyest verdi for cT og minst til det med lavest cT.
- Hvis produktene har like prioriteringsfaktorer ($c_1T_1 = c_2T_2$), skal de dele kapasiteten likt mellom seg. Dette gjelder selvsagt alle kombinasjoner av c og T. Hvis c_1 er dobbelt så stor som c_2, mens $T1$ er halvparten av $T2$, skal de også dele kapasiteten likt.
- Hvis de har samme forsinkelseskostnad c, er det kun utviklingstiden T som bestemmer fordelingen.
- Hvis de har samme utviklingstid T, er det forsinkelseskostnad c som bestemmer fordelingen.
- Kurven for forsinkelseskostnad er u-formet, og derfor rund i bunnen. Det er derfor ikke kritisk at vi finner optimal kapasitetsfordeling helt nøyaktig.

2.8.2 Eksempel på bruk av regelen for optimal kapasitetsfordeling

I et av eksemplene over omsatte P1 for 2 millioner per måned, P2 for 1 million per måned, begge krevde 6 måneders utviklingstid med full kapasitet.

Prioriteringsfaktoren for P1 blir:

$$c_1 T_1 = 2 \times 6 = 12$$

For P2 blir den:

$$c_2 T_2 = 1 \times 6 = 6$$

Vi ser at P1 skal ha større kapasitet enn P2. Forholdet mellom kapasitetandelen deres blir:

$$\frac{r_1}{r_2} = \sqrt{\frac{12}{6}} = \sqrt{2} \approx 1.41$$

Det vil si at P1 skal ha ca. 41 % mer kapasitet enn P2. Fordi den samlede kapasiteten er konstant og lik 100%, blir fordelingen ca. 59% og 41%[9].

Vi legger et tredje produkt til i porteføljen. **P3** omsetter for 1 million per måned og krever 4 måneders utviklingstid.

P3s prioriteringsfaktor blir:

$$c_3 T_3 = 1 \times 4 = 4.$$

P3 skal altså ha minst kapasitet av disse tre produktene.

$$\frac{r_2}{r_3} = \sqrt{\frac{6}{4}} \approx 1.23$$

P2 skal ha ca. 23 % mer kapasitet enn P3. Kapasitetsfordelingen mellom dem blir:

$$P1 \approx 44\%, P2 \approx 31\%, P3 \approx 25\%$$

2.9 Forskjeller mellom sekvensiell og parallell prioritering

Utvikler vi sekvensielt, gir vi produkter med høy omsetning og *kort utviklingstid* høy prioritet. Utvikler vi parallelt, gir vi produkter med høy omsetning og *lang utviklingstid* stor andel av kapasiteten. Vi skjønner kanskje at sekvensiell og parallell metode ikke nødvendigvis vil gi sammenfallende prioriteringer. Denne forskjellen burde vekke vår nysgjerrighet - er den ene metoden alltid mer lønnsom enn den andre?

2.9.1 Produkter med lik omsetning og ulik utviklingstid

Hvis vi har to produkter med lik omsetning, men ulik utviklingstid blir det tydelige forskjeller i prioriteringene mellom sekvensiell og parallell utvikling. Med sekvensiell utvikling utvikler vi produktet med kortest utviklingstid først, og lar det med lang utviklingstid vente. Vi vil altså lansere det mest lønnsomme produktet så raskt som mulig for at det skal gi oss inntekter mens vi utvikler det produktet som er mer arbeidskrevende.

Med parallell utvikling vil vi gi minst kapasitet til det produktet som har kort utviklingstid, og mest kapasitet til det produktet som har lengst utviklingstid. Dermed vil det mest arbeidskrevende produktet forsinke det minst arbeidskrevende produktet, og vi tjener mindre penger mens vi venter på det mest arbeidskrevende produktet. I slike tilfeller vil parallell utvikling være vesentlig mindre lønnsomt enn sekvensiell utvikling. Dette er vist med et eksempel i tabell 1 - 3 hvor vi med parallell utvikling bruker fire ganger så lang tid på å få lansert det første produktet sammenlignet med sekvensiell utvikling. Dette forsinkelsestapet blir betydelig til tross for at hele porteføljen blir lansert kun 2 måneder senere enn med sekvensiell utvikling (en forsinkelse på 18 %.)

Produkt	Oms./mnd.	Omfang (mnd.)
P1	1	1
P2	1	10
Sum	2	11

Tabell 1: Produktportefølje med ulike produkter.

Produkt	Lansering (mnd.)	Oms. år 1
P1	1	11
P2	11	1
Sum		12

Tabell 2: Sekvensiell utvikling.

[9] $59/41 \approx 1.4$ og 59%+41% = 100%

Produkt	Kap. (%)	Lansering (mnd.)	Oms. år 1
P1	24	4	8
P2	76	13	0
Sum	100		8 (-33 %)

Tabell 3: Parallell utvikling.

2.9.2 Produkter med lik utviklingstid og ulik omsetning

Hvis vi har to produkter med ulik omsetning, men lik utviklingstid, blir prioriteringene mer samsvarende. Med sekvensiell utvikling lanserer vi produktet med høyest omsetning først, og lar det med lav omsetning vente. Med parallell utvikling vil vi gi mest kapasitet til det produktet med høyest omsetning, og vil også lansere det mest lønnsomme produktet først. Forskjellene i forsinkelseskostnadene blir mindre. Dette er vist med et eksempel i tabell 4 - 6 hvor vi med parallell utvikling bruker kun 30 % lenger tid på å få lansert det første produktet sammenlignet med sekvensiell utvikling. Til tross for at vi med parallell utvikling bruker mer enn dobbelt så lang tid på å få lansert hele porteføljen, dominerer fremskyndingen av det mest verdifulle produktet inntektsbildet. Forskjellen mellom sekvensiell og parallell utvikling blir mindre. Appendiks A.4 viser flere eksempler på forskjellene i resultatene med sekvensiell og parallell utvikling.

Produkt	Oms./mnd.	Omfang (mnd.)
P1	10	1
P2	1	1
Sum	11	2

Tabell 4: Produktportefølje med ulike produkter.

Produkt	Lansering (mnd.)	Oms. år 1
P1	1	110
P2	2	10
Sum		120

Tabell 5: Sekvensiell utvikling.

Produkt	Kap. (%)	Lansering (mnd.)	Oms. år 1
P1	76	1,3	107
P2	24	4,2	8
Sum	100		115 (-4 %)

Tabell 6: Parallell utvikling.

2.10 Hva er mest lønnsomt - sekvensiell eller parallell utvikling?

I alle de ovennevnte eksemplene kunne vi se at var mer lønnsomt å utvikle produktene sekvensielt enn parallelt. Dette er ikke tilfeldig. I følge modellene over vil i teorien

sekvensiell utvikling alltid være mer lønnsomt enn parallell utvikling

med optimal prioritering, uavhengig av produktenes omsetning og omfang. Dette er vist i appendiks A.3.

Forklaringen er, kort sagt, at med sekvensiell utvikling vil det mest lønnsomme produktet bli lansert så raskt det er mulig, og det vil gi oss inntekter mens vi utvikler resten av porteføljen. Hvis vi utvikler produktene parallelt, sprer vi kapasiteten, og utviklingstiden øker for alle produktene uansett hvordan vi fordeler kapasiteten.

Hvor stor forskjellen blir mellom disse metodene, avhenger av sammensetningen av produktporteføljen. Hvis vi har en portefølje med produkter som har omtrent lik omsetning, men hvor ett produkt har betydelig lengre utviklingstid, vil sekvensiell utvikling være langt mer lønnsomt. Hvis vi imidlertid har en produktportefølje hvor produktene har omtrent samme utviklingstid, men hvor ett produkt har vesentlig større omsetning enn resten av porteføljen, vil forskjellen mellom sekvensiell og parallell utvikling være mindre.

2.11 Andre gevinster ved å være rask

Så langt har vi kun brukt omsetning for å sammenligne sekvensiell og parallell utvikling, men andre perspektiver kan ha vel så stor betydning. Disse perspektivene taler også i favør av seksensiell utvikling.

2.11.1 Redusert risiko

Hvis vi antar at selskapene opererer i et marked med konkurranse, vil det å være først ute med et produkt være fordelaktig, for eksempel i form av større markedsandeler enn de som lanserer senere[10]. Generelt kan vi si at risikoen for uforutsette, uønskede hendelser blir større desto lenger tid vi bruker på å utvikle et produkt. Det er mange faktorer som før eller senere kan svekke våre prioriteringer, som konjunktursvingninger, teknologiske nyvinninger, variasjoner i tilgangen på nøkkelkompetanse og lignende. Risikoen er derfor større med parallelle utviklingsløp. Jo flere parallelle prosjekter, desto større blir risikoen.

2.11.2 Hyppigere prioriteringer og gjenbruk av læring

Vi så i vårt første eksempel at Selskap A kunne prioritere hver 6. måned. Selskap B måtte leve med sine prioriteringer i 12 måneder, med mindre de stoppet pågående utviklingsinitiativer. Det betyr at selskap A lettere kan tilpasse seg endringer enn selskap B. I selskap A vil hvert nytt initiativ som settes i gang allerede fra start få fordeler av erfaringene fra de forutgående initiativene. I selskap B har de mindre muligheter for å gjenbruke læring fordi de starter initiativene samtidig.

2.11.3 Færre avhengigheter

Hvis initiativene har avhengigheter, som for eksempel felles teknisk plattform og infrastruktur, delt nøkkelkompetanse og lignende, kan selskap B ende opp med å bruke enda lengre tid på grunn av venting og flaskehalser. Jo flere initiativer i gang, desto større er sannsynligheten for slike avhengigheter.

2.11.4 Lavere administrasjonskostnader

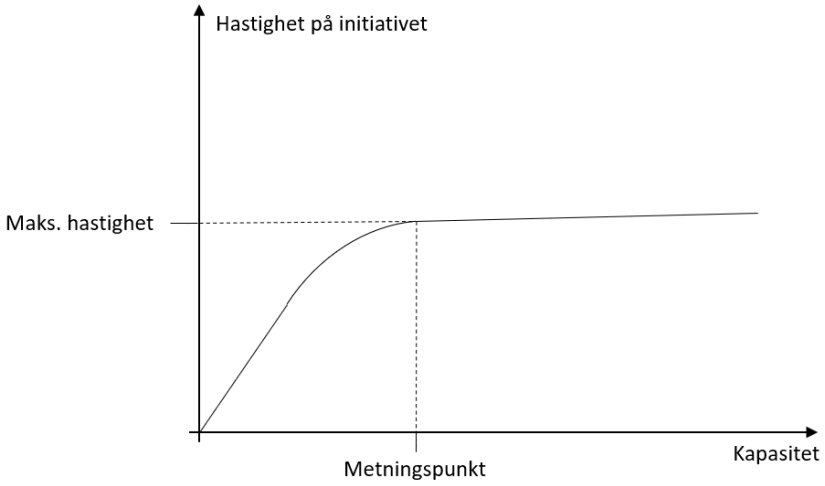
Jo flere samtidige initiativer, desto mer kapasitet må vi bruke på å administrere dem. Som vi har sett vil parallell utvikling resultere i lengre utviklingstid for alle initiativer sammenlignet med sekvensiell utvikling. De løpende administrasjonskostnadene øker, og som regel er det koordineringen på tvers av pågående initiativer som bidrar mest til den administrative kompleksiteten. Sett fra brukerens perspektiv, er ikke dette merarbeidet verdiøkende - det gjør ikke produktene bedre.

3 Teori møter praksis

Ifølge teorien vil sekvensiell utvikling gi høyere omsetning, lavere administrasjonskostnader, raskere læring og lavere risiko sammenlignet med parallell utvikling. Premisset for denne konklusjonen er at vi har fleksibel kapasitet, og at initiativene kan akselereres ved å tilføre dem mer kapasitet. I praksis er dette sjelden tilfelle. De fleste initiativer når et metningspunkt hvor ytterligere kapasitet ikke lenger vil gi økt fremdrift. Se figur 12. Ressurseffektiviteten kan da bli så lav at vi kan skape mer verdi ved å fordele kapasiteten på flere parallelle initiativer.

Betyr dette at vi i praksis skal velge parallell og forkaste sekvensiell utvikling? Svaret er nei; vi kan bruke en hybridmetode som baserer seg på det viktigste fortrinnet i den sekvensielle metoden. Vi har bevist at sekvensiell utvikling er mest lønnsomt fordi vi så tidlig som mulig ferdigstiller det initiativet som gir høyest verdi i forhold til arbeidsinnsatsen. Vi har også sett at det gir store fordeler å jobbe med *få* initiativer om gangen. Disse prinsippene vil vi holde fast ved når vi prioriterer i den virkelige verden.

[10] *The first mover advantage*: Et konkurransefortrinn ved å være den første til å bringe et nytt produkt eller en ny tjeneste til markedet, som sterkere merkevaregjenkjenning og kundelojalitet.



Figur 12: Punktet hvor mer kapasitet ikke gir mer fart.

3.1 Mer skalerbare initiativer

Alle initiativer har et metningspunkt som begrenser farten. Dette får oss til å spre kapasiteten på flere initiativer, og vi vet nå at dette øker forsinkelsestapet. Det burde motivere oss til å prøve å skyve på metningspunktet - det vil si at vi gjør initiativene mer skalerbare. Dette oppnår vi f. eks. ved å:

- Redusere arbeidsomfanget ved å ta ut elementer som ikke gir kunde- og brukerverdi.
- Fjerne flaskehalser[11], dvs. ledd som begrenser farten i hele prosessen.
- Bryte større oppgaver opp i mindre deler. Det kan øke flyten og kontrollen, og fjerne avhengigheter og kompleksitet.
- Planlegge med *mindre enn* 100 % kapasitetsutnyttelse for å ha slakk nok til å håndtere uforutsette hendelser.

3.2 Hybridløsningen: Parallelle prosjekter med sekvensiell prioritering

Vi har konkludert med at vi ønsker en god balanse; *høyest mulig fart på initiativet som gir den største verdien pr. arbeidstime*, men samtidig ha så *høy ressurseffektivitet som mulig*. Det første oppnår vi ved å prioritere etter formelen for optimal rekkefølge med sekvensiell utvikling, dvs. gi høyest prioritet til det produktet som gir oss *mest verdi pr. utviklingstid*, osv.:

$$\frac{c_1}{T_1} > \frac{c_2}{T_2} > \frac{c_3}{T_3} \cdots > \frac{c_n}{T_n}$$

Ideelt sett burde vi få P1 ferdig før P2, P2 før P3 osv. Fordi initiativene har et metningspunkt, velger vi likevel å sette i gang flere, samtidig som vi sørger for at initiativet med høyest prioritet (P1) blir ferdig så fort som praktisk mulig. P1 skal ha maksimal fart, og skal derfor få så stor andel av kapasiteten at det ikke ville blitt tidligere ferdig hvis det hadde fått mer kapasitet. Hvis vi har ledig kapasitet til overs, bør vi gi den til P2 etter samme prinsipp. Slik kan vi fortsette å fordele kapasitet til P3, P4 og så videre til vi ikke har mer kapasitet igjen. Da har vi både fordelt kapasiteten optimalt og funnet ut hvor mange samtidige initiativer det er lønnsomt å ha i gang. Vi bør i tillegg vurdere hvor komplekst det er å administrere disse initiativene. Avhengigheter kan bli en avgjørende flaskehals, og ved å stanse initiativer vil vi i såfall kunne gi betydelig mer fart i resten av porteføljen.

[11] Det er alltid en flaskehals, f. eks. en komponent, et system, en delleveranse, spisskompetanse osv. som andre er avhengige av for å komme videre.

3.3 Smidig kapasitetsfordeling med "Pluss én, minus én"-metoden

Vi kan gjøre denne kapasitetsfordelingen konkret og praktisk ved først å spørre: Hvis P1 hadde fått enda én medarbeider[12] med på laget, ville P1 da blitt ferdig tidligere? Hvis svaret er ja, gir vi P1 én medarbeider til med på laget. Hva med to? Tre? Slik fortsetter vi å spørre inntil svaret blir nei. Hvis vi da har ledig kapasitet igjen, gir vi den til P2 ved å stille det samme spørsmålet. Hvis P1 imidlertid svarte nei første gang, snur vi spørsmålene. Hvis P1 hadde fått *mindre* kapasitet, ville P1 da blitt ferdig *senere*? Hvis svaret er nei, kan vi flytte kapasitet fra P1 til P2, og fortsette å spørre slik gjennom hele porteføljen. Resultatet kan bli at noen initiativer må vente, men vi vil likevel ende opp med en god balansen mellom flyteffektivitet og ressurseffektivitet. Situasjonen vil antageligvis endre seg over tid, og vi bør derfor gjøre denne vurdering regelmessig.

3.4 Høy fart er viktigere enn høy kapasitetsutnyttelse

Det er likevel viktig å huske at høy fart på det høyest prioriterte initiativet kan være viktigere enn maksimal kapasitetsutnyttelse. Forsinkelseskostnadene kan være vesentlig høyere enn kapasitetskostnadene, og da kan vi tåle redusert utnyttelse så lenge vi oppnår maksimal fart på det mest lønnsomme initiativet. Det kan være bedre med for stor enn for liten kapasitet til det høyest prioriterte initiativet.

4 Initiativer uten kroneverdi eller andre definerte effekter

Så langt har vi basert teori og eksempler på at vi har produkter som gir oss en inntekt, men hva om det vi skal prioritere ikke er knyttet til penger? Og hva om det vi skal prioritere ikke er produkter, men helt andre typer initiativer som ikke har så konkret definerte effekter?

4.1 Initiativer uten kroneverdi

Kroneverdiene vi har brukt i eksemplene så langt er i matematisk forstand en vekting av de ulike produktene. Denne vektingen kan representere hva som helst. Matematikken forblir den samme, og er uavhengig av hva vektingen representerer. Initiativene trenger derfor ikke å være produkter, og det vi prioriterer trenger heller ikke å være utviklingsprosjekter. Modellene kan brukes for alle initiativer eller oppgaver som skal skape en eller annen form for verdi. Metoden krever imidlertid at vi klarer å sette en relevant tallverdi på det vi skal prioritere. I produkt- og teknologiutvikling kan det være effekter som flere besøk, brukere, klikk eller transaksjoner vi ønsker å oppnå. I salg prioriteres kanskje ulike tilbud basert på definerte kontraktsverdier. I HR kan effektene vi ser etter være antall søkere på utlyste stillinger, sykefraværsandel osv. Vi kan vekte initiativene etter de effektene som er relevante for oss, og bruke de samme prioriteringsmetodene som vi allerede har skissert.

4.2 Initiativer uten kvantiserte effekter

Hvis vi ikke kan sette eksakte tallverdier på initiativene våre, kan vi fortsatt bruke prioriteringsteorien ved å rangere initiativenes verdi i forhold til hverandre, og tildele dem tallverdier basert på denne rangeringen. Hvis vi for eksempel har tre initiativer, kan vi gi det initiativet som vi mener vil skape mest verdi tallet 3, det neste mest verdifulle initiativet tallet 2, og det siste 1. Hvis vi ønsker å tallfeste initiativenes verdi mer nyansert, kan vi bruke en annen skala, for eksempel 100, 50, 25 osv. Her er det bare fantasien som setter grenser.

Hvis vi ikke klarer å rangere oppgavene etter verdi, kan vi like godt anta at alle oppgavene har samme verdi, for enkelhetsskyld gi alle verdien 1. Da vil arbeidsomfanget alene avgjøre rekkefølgen (sekvensielt) eller kapasitetsfordelingen (parallelt). Som vi har sett sier matematikken at den minste oppgaven skal gjøres først hvis vi jobber sekvensielt. Ved parallell jobbing gir vi mest kapasitet til oppgaven med størst omfang.

[12] Medarbeider med relevant kompetanse. Medarbeider kan også tolkes som en metafor på den minste praktiske kapasitetsenheten vi kan disponere. For eksempel kan et team være den minste enheten.

4.3 Initiativer uten nøyaktig tidsestimat

Teorien vår er basert på at vi vet hvor lang tid det vil ta å gjennomføre initiativene. Hvordan prioriterer vi hvis vi ikke vet presist hvor lang tid det vil ta å gjennomføre dem? Vi kan tenke på samme måte som over, hvor vi ikke hadde eksakte tall for verdi. Vi kan rangere initiativene etter relativt omfang. Dermed ender vi opp med relativ vekting av både verdi og omfang. Det er ikke nødvendig å bruke samme skala for verdi og omfang.

Appendiks A.5 viser noen eksempler på hvordan denne metoden kan brukes i praksis.

4.4 Når vi ikke vet noe om verdi eller omfang

Hvis vi ikke har nok innsikt til å anslå hvor lang tid initiativene tar å fullføre, innebærer sekvensiell utvikling en risiko for at ett initiativ kan blokkere andre i uforholdsmessig lang tid, og dette kan skape store forsinkelseskostnader. Da kan risikoen bli lavere ved å utvikle parallelt fordi vi i det minste vil få fremdrift på noen initiativer. Samtidig vet vi at parallellisering også vil skape økte forsinkelseskostnader. Initiativer med stort omfang beslaglegger en stor andel av kapasiteten, og forsinker mer lønnsomme initiativer med mindre omfang. En praktisk løsning på dette dilemmaet er å sette en fast tidsramme (*timebox* eller *timeout*) for alle initiativer som settes i gang. Når denne tidsgrensen nås, stoppes igangsatte initiativer som ikke er ferdige, slik at vi kan sette i gang det neste initiativet. Initiativene som stoppes legges tilbake i køen og prioriteres opp mot de andre initiativene. Tidsrammen settes slik at majoriteten[13] av oppgavene blir ferdige innenfor denne perioden. Denne løsningen er en nødløsning vi kan bruke når vi ikke har nok informasjon om det vi prioriterer. Men la ikke dette bli en sovepute. Som vi har sett, kan tapene ved feilprioritering bli så store at det sannsynligvis er lønnsomt å bruke tid på å få en viss oversikt over både verdi og omfang.

5 Oppsummering

Hvordan vi prioriterer har stor betydning for organisasjonens evne til å skape verdi. Sammenlignet med kostnadene som påløper i de senere stadiene av et prosjekt eller en prosess, er riktig prioritering en enkel og kostnadseffektiv måte å forbedre resultatene på. En analytisk tilnærming til prioritering kan dermed gi store gevinster for hele organisasjonen.

Ved å følge noen tommelfingerregler kan prioriteringene bli vesentlig bedre. En av de mest virkningsfulle grepene er *å prioritere få initiativer om gangen* og sørge for at disse blir gjennomført raskt. Ved *å redusere antall parallelle initiativer* oppnår vi flere fordeler, som raskere vekst, lavere risiko, raskere læring, færre avhengigheter og mindre administrasjon. Organisasjonen blir kort sagt smidigere og mer produktiv.

Prioritering handler ikke bare om hvilke initiativer som skal gjennomføres, men også om hvilken *rekkefølge* de skal gjennomføres i, og *kapasitetsfordelingen* mellom dem. Vi bør sørge for at initiativet som gir den største verdien pr. arbeidstime får høyest mulig fart. Å *øke organisasjonens fleksibilitet* være lønnsomt fordi det blir enklere å sette inn ressursene der de gir størst verdi.

Prioritering er ferskvare; *prioriter hyppig*. Verden endrer seg kontinuerlig, og det som er riktig prioritering i dag, er kanskje mindre optimalt i morgen. Det er alltid en risiko for å prioritere feil, men feilprioriteringer kan vi lære mye av. Prioriterer vi hyppig, kan vi raskere stanse uheldige initiativer.

En prioriteringsmetodikk som *legger vekt på forsinkelseskostnader og arbeidsomfang*, vil bidra til at organisasjonen kontinuerlig jobber med de riktige tingene, i riktig rekkefølge og med riktig kapasitet. Unngå overtenking; gode estimater er kanskje ti ganger bedre enn dårlige, men dårlige estimater er hundre ganger bedre enn ingen estimater.

Den største prioriteringsfeilen du kan gjøre, er å unngå å prioritere. Hvis ikke du prioriterer, vil noen andre gjøre det – men hvem, og på hvilket grunnlag? Én ting kan du være sikker på: resultatet blir sjeldent optimalt hvis du overlater prioritering til tilfeldighetene.

[13] Det er vanlig å sette rammen slik at 80 - 90 % av initiativene rekker å bli ferdige i løpet av tidsrammen.

Referanser

[1] Reinertsen, D. G. (2009). *The Principles of Product Development Flow: Second Generation Lean Product Development*. Celeritas Publishing. s. 31-32.

[2] Reinertsen, D. G. (2009). *The Principles of Product Development Flow: Second Generation Lean Product Development*. Celeritas Publishing. s. 191-195.

[3] Black Swan Farming, Cost of Delay Divided by Duration

A Appendiks

A.1 Optimal prioriteringsrekkefølge med sekvensiell utvikling

Vi utvikler produktene sekvensielt, og vil finne rekkefølgen produktene skal utvikles i som gir oss lavest forsinkelseskostnad (eng. "Cost of Delay") for hele porteføljen. For å beregne den totale forsinkelseskostnaden bruker vi hvert produkts løpende omsetning, dvs. omsetning per tid. Vi kan kalle dette produktets *spesifikke forsinkelseskostnad* som vi symboliserer med funksjonen $c(t)$. Denne forteller oss hvor mye vi taper per tidsenhet mens produktet venter på å bli lansert. Ventetiden er summen av produktets egen utviklingstid og utviklingstiden til de produktene som står foran i køen. Produktenes omsetning vil variere over tid, og derfor er $c(t)$ en tidsavhengig funksjon.

A.1.1 Optimal rekkefølge når omsetningen varierer over tid

Vi antar først at vi har to produkter med tilhørende spesifikke forsinkelseskostnad c og utviklingstid T:

- P1: $c_1(t)$ og T_1
- P2: $c_2(t)$ og T_2

Utviklingstiden er tiden det tar å ferdigstille produktet med full utviklingskapasitet, og vi starter utviklingen ved tiden T_0. Samlet forsinkelseskostnad C når vi utvikler produktene i rekkefølgen $P1, P2$, blir:

$$C_{12} = \int_{T_0}^{T_0+T_1} c_1(t)dt + \int_{T_0}^{T_0+T_1+T_2} c_2(t)dt$$

Hvis vi bytter om rekkefølgen til $P2, P1$, blir forsinkelseskosntaden:

$$C_{21} = \int_{T_0}^{T_0+T_2} c_2(t)dt + \int_{T_0}^{T_0+T_1+T_2} c_1(t)dt$$

Samlet forsinkelseskostnad er med andre ord arealet under produktenes omsetningskurver i utviklingsperioden. Hvis $c_1(t)$ og $c_1(t)$ har vesentlige tidsvariasjoner, kan vi ikke utlede en generell regel for optimal rekkefølge. Da må vi beregne C_{12} og C_{21} separat og sammeligne resultatene. Hvis vi for eksempel har med tydelige sesongvariasjoner å gjøre, vil den samlede forsinkelseskostnaden i stor grad påvirkes av rekkefølgen vi utvikler produktene i.

A.1.2 Optimal rekkefølge konstant omsetning over tid

For å finne en enklere regel for optimal rekkefølge, antar vi at omsetningen er konstant over tid, dvs. $c(t) = c$. Hvis vi likevel har noe variasjon, kan vi bruke den gjennomsnittlige omsetningen per produkt i den aktuelle perioden. Restulatet blir ikke lenger avhengig av oppstartstidspunktet T_0. Integralene over forenkles, og forsinkelseskostnaden som skapes av produktets egen utviklingstid blir derfor bare cT.

Vi regner først med kun to produkter med tilhørende spesifikke forsinkelseskostnader og utviklingstider:

- P1: c_1 og T_1
- P2: c_2 og T_2

Samlet forsinkelseskostnad når vi utvikler produktene i rekkefølgen $P1, P2$ blir:

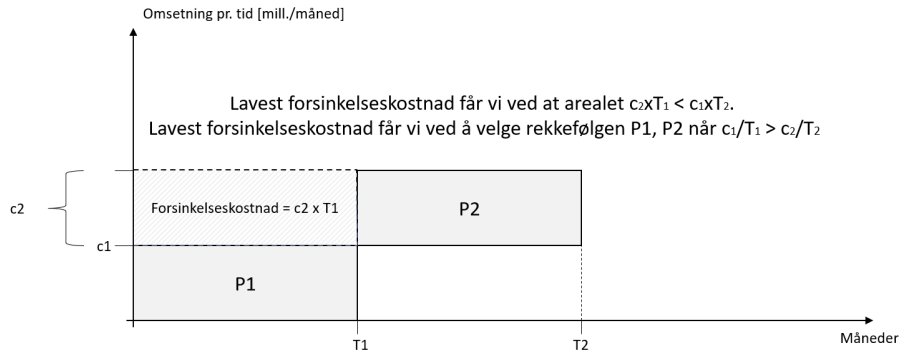
$$C_{12} = c_1T_1 + c_2(T_1 + T_2)$$

Se figur 13. Hvis vi bytter om rekkefølgen til $P2, P1$ får vi:

$$C_{21} = c_2T_2 + c_1(T_1 + T_2)$$

Vi ønsker den rekkefølgen som gir oss lavest samlet forsinkelseskostnad. Det finner vi ved å sette opp en ulikhet hvor vi krever at rekkefølgen $P1, P2$ skal gi lavere forsinkelseskostnad enn rekkefølgen $P2, P1$:

$$C_{12} < C_{21}$$



Figur 13: Forsinkelseskostnaden ved at P2 venter på P1.

Se figur 13.

$$\begin{gathered} c_1T_1 + c_2(T_1 + T_2) < c_2T_2 + c_1(T_1 + T_2) \\ \Downarrow \\ c_1T_1 + c_2T_1 + c_2T_2 < c_2T_2 + c_1T_1 + c_1T_2 \end{gathered}$$

Leddene c_1T_1 og c_2T_2 er forsinkelseskostnaden som hvert produkt skaper med sin egen utviklingstid. Den er uavhengig av rekkefølgen vi utvikler produktene i. Vi kan stryke disse leddene, og vi står igjen med:

$$c_2T_1 < c_1T_2$$

Dersom rekkefølgen $P1, P2$ skal gi lavest forsinkelseskostnad, må forsinkelseskostnaden som skapes av at P2 venter på at vi utvikler P1 være lavere enn forsinkelseskostnaden som skapes av at P1 venter på P2. Det vi taper på å la $P2$ vente på $P1$ tilsvarer arealet c_2T_1 i figur 13. Vi skal altså velge den rekkefølgen som gir minst areal cT. Hvis vi snur om på faktorene i ulikheten, får vi svaret på en form som er lettere å tolke:

$$\frac{c_1}{T_1} > \frac{c_2}{T_2}$$

Dette betyr at vi får lavest forsinkelseskostnad hvis vi utvikler produktet med høyest løpende omsetning per utviklingstid først. Dette er med andre ord det initiativet som gir oss størst verdiøkning i forhold til tiden vi bruker på å gjennomføre initiativet.

Hvis vi har tre produkter ($P1, P2$ og $P3$), kan vi gjøre den samme analysen for P2 og P3. Regelen blir den samme for den optimale rekkefølgen for P2 og P3:

$$\frac{c_2}{T_2} > \frac{c_3}{T_3}$$

Med n produkter kan vi skrive regelen slik:

$$\frac{c_1}{T_1} > \frac{c_2}{T_2} > \frac{c_3}{T_3} \cdots > \frac{c_n}{T_n}$$

Hvor $c_1, c_2, \ldots, c_n$ er hvert produkts spesifikke forsinkelseskostnad. $T_1, T_2, \ldots, T_n$ er utviklingstiden for produktene. Vi utvikler produktet med den høyeste verdi på brøken $\frac{c}{T}$ først. Hvis omsetning måles i kroner og tid måles i måneder, blir enheten for denne brøken

$$[c/T] = [kroner/tid^2]$$

Dette er stigningstallet til omsetningskurven, dvs. endringen i løpende omsetning mens produktet utvikles. Dette kan også tolke som omsetningens *akselerasjon*.

A.2 Optimal kapasitetsfordeling med parallell utvikling

Vi velger også her å se på den samlede forsinkelseskostnaden for hele porteføljen. I dette tilfellet blir situasjonen ganske annerledes fordi vi utvikler alle produktene i parallell. Vi har en begrenset, kontant utviklingskapasitet. Optimaliseringsoppgaven handler om å fordele kapasiteten mellom alle utviklingsprosjektene slik at de samlede forsinkelseskostnader blir lavest mulig. Andelen av kapasiteten produktene får til utvikling er:

$$r_1, r_2, r_3, \dots r_n$$

Det betyr at produkt $P1$ får r_1, $P2$ får r_2, osv. Summen av disse er lik 1:

$$r_1 + r_2 + r_3 \cdots + r_n = 1$$

Disse r-ene er med andre ord andelen av den totale kapasiteten som produktene tildeles, og den totale kapasiteten er 100 %. Vi må også anta at utviklingsprosjektene er lineært skalerbare, dvs. at utviklingstiden halveres når kapasitetsandelen dobles, osv.

Den laveste forsinkelseskostnaden vi kan oppnå for produktet $P1$ er c_1T_1 hvor T_1 er den tiden det tar å utvikle $P1$ hvis det hadde fått hele utviklingskapasiteten.

Fordi vi med parallell utvikling ikke kan bruke all kapasitet på ett produkt, blir utviklingstiden lengre for samtlige produkter. Utviklingstiden øker fra T_1 til T_1/r_1, T_2 til T_2/r_2 osv. hvor r_n er et tall mellom 0 og 1, men aldri 0 eller 1. Det betyr at hvert produkt nå tar lengre tid å utvikle enn hvis vi hadde brukt all tilgjengelig kapasitet på hvert av dem. Denne forsinkelsen er med andre ord en form for venting som med sekvensiell utvikling. Med parallell utvikling venter vi ikke med å starte opp utviklingsprosjektene, men vi setter ned tempoet i samtlige prosjekter fordi vi sprer kapasiteten.

Samlet forsinkelseskostnad for to produkter blir:

$$C = \frac{c_1T_1}{r_1} + \frac{c_2T_2}{r_2}$$

hvor $r_1 + r_2 = 1$. Derfor kan vi skrive:

$$C = \frac{c_1T_1}{r_1} + \frac{c_2T_2}{(1 - r_1)}$$

Dette er summen av to speilvendte hyperbler som er avhengige av hverandre gjennom $r_1 + r_2 = 1$. Det ene leddet i funksjonen øker når den andre minker, og vice versa. Summen av dem gir en U-formet kurve med et definert minimumspunkt. Se figur 14. Fordi både c_1 og c_2 er positive, ser vi at summen C blir en kurve med kun ett minimumspunkt - den r-verdien som gir lavest samlet forsinkelseskostnad. Vi ser også at når r går mot 1 eller 0, går C mot ∞.

Vi finner dette minimumspunktet ved å derivere uttrykket for C over med hensyn på r_1, sette den deriverte lik null, løse ligningen med hensyn på r_1, og bruke at $r_2 = 1 - r_1$.

$$\frac{dC}{dr_1} = \frac{c_1T_1}{r_1^2} - \frac{c_2T_2}{(1 - r_1)^2} = \frac{c_1T_1}{r_1^2} - \frac{c_2T_2}{r_2^2} = 0$$

Da finner vi at den optimale kapasitetsfordelingen mellom produktene P_1 og P_2 er:

$$\frac{r_1}{r_2} = \sqrt{\frac{c_1T_1}{c_2T_2}}$$

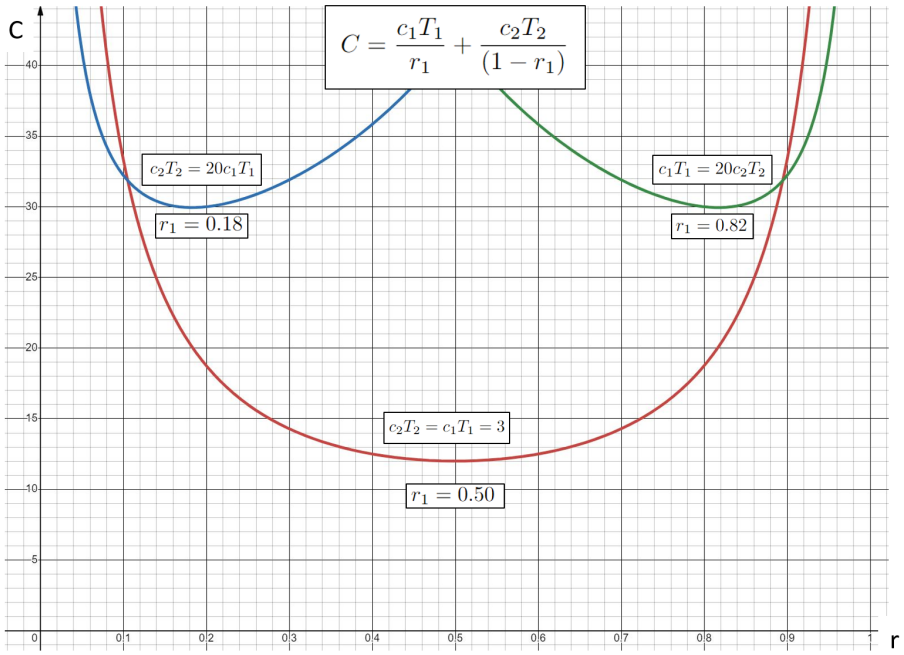
Med flere produkter i porteføljen kan vi utvide resonnementet, og vi får:

$$\frac{r_1}{r_2} = \sqrt{\frac{c_1T_1}{c_2T_2}}$$

$$\frac{r_2}{r_3} = \sqrt{\frac{c_2T_2}{c_3T_3}}$$

$$\frac{r_n}{r_{n+1}} = \sqrt{\frac{c_nT_n}{c_{n+1}T_{n+1}}}$$

Hvor $r_1, r_2, \dots, r_n$ er produktenes andel av kapasiteten, $c_1, c_2, \dots, c_n$ er hvert produkts forsinkelseskostnad. $T_1, T_2, \dots, T_n$ er utviklingstiden for produktene.



Figur 14: Kurven for forsinkelseskostnad ved parallell utvikling med ulike verdier for c_nT_n og optimal r_n.

A.3 Hva gir lavest forsinkelseskostnad - sekvensiell eller parallell?

Vi vil nå se om en av disse metodene alltid vil gi oss teoretisk lavest forsinkelseskostnad. Vi antar at vi prioriterer optimalt, det vil si optimal rekkefølge i det sekvensielle tilfellet, og optimal kapasitetsfordeling i det parallelle tilfellet. Vi skal altså løse ulikheten $C_{sekvensiell} < C_{parallell}$:

$$c_1T_1 + c_2(T_1 + T_2) < \frac{c_1T_1}{r_1} + \frac{c_2T_2}{(1-r_1)}$$
$$\Downarrow$$
$$c_1T_1 + c_2T_1 + c_2T_2 < \frac{c_1T_1}{r_1} + \frac{c_2T_2}{(1-r_1)}$$

for alle verdier av c,T og r. Å løse denne ulikheten direkte med algebra innebærer mye skriving, men det er lettere å analysere ligningen med noen kritiske verdier. Vi vet allerede at forsinkelseskostnadskurven for parallell utvikling er u-formet med et definert minimumspunkt, og at r aldri blir 0 eller 1.

Grenseverdi nr. 1: P1 har mye større forsinkelseskostnad enn P2
Det vil si at $c_1 >> c_2$, og da er $c_1T_1 >> c_2T_1$ og kan vi stryke leddet c_2T_1. Vi får:

$$c_1T_1 + c_2T_2 < \frac{c_1T_1}{r_1} + \frac{c_2T_2}{(1-r_1)}$$

Det er nevneren i brøken på høyre side som skiller sekvensielt fra parallelt, og fordi $0 < r < 1$ vil venstre side av ulikheten alltid være minst. Det betyr at $C_{sekvensiell} < C_{parallell}$.

Grenseverdi nr. 2: P2 har mye større forsinkelseskostnad enn P1
Det vil si at $c_2 >> c_1$. Da er $c_1T_1 << c_2(T_1 + T_2)$ og kan vi stryke leddet c_1T_1 på venstre side.

$$c_2(T_1 + T_2) < \frac{c_1T_1}{r_1} + \frac{c_2T_2}{(1-r_1)}$$

Fordi c_2 utvikles sist i optimal sekvensiell utvikling, er $\frac{c_1}{T_1} > \frac{c_2}{T_2}$. Her er også $c_2 >> c_1$, og da må $T_2 >> T_1$. Vi kan derfor stryke T_1 på venstre side av ulikheten.

$$c_2T_2 < \frac{c_1T_1}{r_1} + \frac{c_2T_2}{(1-r_1)}$$

Leddet c_2T_2 går igjen på begge sider av ulikheten, men på høyre side er den dividert med $1-r_1$. som er mindre enn 1 og større enn null. Dermed er venstre side alltid mindre enn høyre. Det betyr at $C_{sekvensiell} < C_{parallell}$.

Grenseverdi nr. 3: P1 krever mye lenger utviklingstid enn P2

Det vil si at $T_1 >> T_2$. Vi kan bruke resonnementet fra forrige punkt. Fordi c_1 per definisjon utvikles først i sekvensiell utvikling, er $\frac{c_1}{T_1} > \frac{c_2}{T_2}$. Da må også $c_1 >> c_2$.

$$\begin{gathered} c_1T_1 + c_2T_1 + c_2T_2 < \frac{c_1T_1}{r_1} + \frac{c_2T_2}{(1-r_1)} \\ \Downarrow \\ T_1(c_1 + c_2) + c_2T_2 < \frac{c_1T_1}{r_1} + \frac{c_2T_2}{(1-r_1)} \end{gathered}$$

Vi kan stryke leddet c_2 i parentesen og c_2T_2 på venstre side.

$$c_1T_1 < \frac{c_1T_1}{r_1} + \frac{c_2T_2}{(1-r_1)}$$

Fordi $0 < r_1 < 1$ vil venstre side alltid være mindre enn høyre. Det betyr at $C_{sekvensiell} < C_{parallell}$.

Grenseverdi nr. 4: P2 krever mye lenger utviklingstid enn P1

Vi antar at $T_2 >> T_1$.

$$\begin{gathered} c_1T_1 + c_2T_1 + c_2T_2 < \frac{c_1T_1}{r_1} + \frac{c_2T_2}{(1-r_1)} \\ \Downarrow \\ c_1T_1 + c_2(T_1 + T_2) < \frac{c_1T_1}{r_1} + \frac{c_2T_2}{(1-r_1)} \end{gathered}$$

Vi kan stryke T_1 i parentesen på venstre side:

$$c_1T_1 + c_2T_2 < \frac{c_1T_1}{r_1} + \frac{c_2T_2}{(1-r_1)}$$

Fordi $0 < r_1 < 1$ vil venstre side alltid være mindre enn høyre. Det betyr at $C_{sekvensiell} < C_{parallell}$.

Likevektspunktet: P1 og P2 har samme akkumulerte forsinkelseskostnad

Vi antar at $c_1T_1 = c_2T_2$.

$$C_{sekvensiell} = c_1T_1 + c_2T_1 + c_2T_2 = 2c_1T_1 + c_2T_1$$

$$C_{parallell} = \frac{c_1T_1}{r_1} + \frac{c_2T_2}{(1-r_1)} = 2c_1T_1 + 2c_2T_2 = 4c_1T_1 = 4c_2T_2$$

fordi optimal kapasitetsfordeling med parallell utvikling her gir $r_1 = r_2 = \frac{1}{2}$.

I det sekvensielle tilfellet ser vi at det kritiske leddet på venstre siden er c_2T_1. Vi undersøker hva som skjer med venstre side med ulike verdier. $c_2 = c_1$:

$$C_{sekvensiell} = 3c_1T_1 = 3c_2T_2$$

Her ser vi at sekvensiell utvikling gir 25 % lavere forsinkelseskostnad enn parallell utvikling.

Hvis vi istedenfor antar at $c_2 >> c_1$, og husker på at $c_1T_1 = c_2T_2$, må i såfall $T_2 << T_1$. Det vil imidlertid ikke skje med optimal sekvensiell utvikling, og dette er derfor ikke en reell mulighet.

Til slutt antar vi at $c_2 << c_1$: Det har vi allerede analysert, og fikk:

$$C_{sekvensiell} = c_1T_1 + c_2T_2 = 2c_1T_1 = 2c_2T_2$$

$$C_{parallell} = \frac{c_1T_1}{r_1} + \frac{c_2T_2}{(1-r_1)} = 4c_1T_1 = 4c_2T_2$$

Vi ser at i dette tilfellet går forsinkelseskostnaden ved sekvensiell utvikling ned mot halvparten av det vi får ved parallell utvikling.

Vi har nå undersøkt de kritiske verdiene, dvs. grenseverdiene og minimumspunktet til den omvendte U-kurven for $C_{parallell}$, og ser at det ikke finnes kombinasjoner av omsetning, utviklingstid og kapasitetsfordeling hvor parallell utvikling vil gi lavere forsinkelseskostnad enn sekvensiell utvikling.

A.4 Illustrative regneeksempler

I eksemplene under er forsinkelseskostnadene beregnet med optimal rekkefølge og kapasitetsfordeling (avrundet).

A.4.1 Like produkter

Produkt	Oms./mnd.	Omfang (mnd.)
P1	1	4
P2	1	4
P3	1	4
Sum	3	12

Tabell 7: Produktportefølje med like produkter.

Metode	Forsinkelseskostnad
Sekvensiell	24
Parallell	36
Økning	12 (50 %)

Tabell 8: Akkumulert forsinkelseskostnad.

Produkt	Lansering	Oms. år 1	Oms. år 2	Oms. totalt
P1	4	8	12	20
P2	8	4	12	16
P3	12	0	12	12
Sum		12	36	48

Tabell 9: Sekvensiell utvikling.

Produkt	Kap. (%)	Lansering	Oms. år 1	Oms. år 2	Oms. totalt
P1	33	12	0	12	12
P2	33	12	0	12	12
P3	33	12	0	12	12
Sum			0	36	36

Tabell 10: Parallell utvikling.

A.4.2 Produkter med ulik omsetning, samme utviklingsomfang.

Produkt	Oms./mnd.	Omfang (mnd.)
P1	4	4
P2	2	4
P3	1	4
Sum	7	12

Tabell 11: Produktportefølje med like produkter.

Metode	Forsinkelseskostnad
Sekvensiell	44
Parallell	78
Økning	34 (77 %)

Tabell 12: Akkumulert forsinkelseskostnad.

Produkt	Lansering	Oms. år 1	Oms. år 2	Oms. totalt
P1	4	32	48	20
P2	8	8	24	16
P3	12	0	12	12
Sum		40	84	124

Tabell 13: Sekvensiell utvikling.

Produkt	Kap. (%)	Lansering	Oms. år 1	Oms. år 2	Oms. totalt
P1	45	9	13	48	61
P2	32	13	0	23	23
P3	23	18	0	6	6
Sum			13	77	90

Tabell 14: Parallell utvikling. Tallene i tabellen er avrundet.

A.4.3 Produkter med lik omsetning, ulikt utviklingsomfang.

Produkt	Oms./mnd.	Omfang (mnd.)
P1	1	2
P2	1	4
P3	1	6
Sum	3	12

Tabell 15: Produktportefølje med like produkter.

Metode	Forsinkelseskostnad
Sekvensiell	20
Parallell	34
Økning	14 (72 %)

Tabell 16: Akkumulert forsinkelseskostnad.

Produkt	Lansering	Oms. år 1	Oms. år 2	Oms. totalt
P1	2	10	12	22
P2	6	6	12	18
P3	12	0	12	12
Sum		16	36	52

Tabell 17: Sekvensiell utvikling.

Produkt	Kap. (%)	Lansering	Oms. år 1	Oms. år 2	Oms. totalt
P1	24	8	4	12	16
P2	34	12	0	12	12
P3	42	14	0	10	10
Sum			4	34	38

Tabell 18: Parallell utvikling. Tallene i tabellen er avrundet.

A.4.4 Produkter med ulik omsetning og utviklingsomfang.

Produkt	Oms./mnd.	Omfang (mnd.)
P1	4	2
P2	2	4
P3	1	6
Sum	3	12

Tabell 19: Produktportefølje med like produkter.

Metode	Forsinkelseskostnad
Sekvensiell	32
Parallell	66
Økning	14 (105 %)

Tabell 20: Akkumulert forsinkelseskostnad.

Produkt	Lansering	Oms. år 1	Oms. år 2	Oms. totalt
P1	2	40	48	88
P2	6	12	24	36
P3	12	0	12	12
Sum		52	84	136

Tabell 21: Sekvensiell utvikling.

Produkt	Kap. (%)	Lansering	Oms. år 1	Oms. år 2	Oms. totalt
P1	35	6	25	48	16
P2	35	11	1	24	12
P3	30	20	0	4	10
Sum			26	76	102

Tabell 22: Parallell utvikling. Tallene i tabellen er avrundet.

A.4.5 Produkter med ulik omsetning og utviklingsomfang.

Produkt	Oms./mnd.	Omfang (mnd.)
P1	4	6
P2	2	4
P3	1	2
Sum	3	12

Tabell 23: Produktportefølje med like produkter.

Metode	Forsinkelseskostnad
Sekvensiell	56
Parallell	84
Økning	28 (49 %)

Tabell 24: Akkumulert forsinkelseskostnad.

Produkt	Lansering	Oms. år 1	Oms. år 2	Oms. totalt
P1	6	24	48	72
P2	10	4	24	28
P3	12	0	12	12
Sum		28	84	112

Tabell 25: Sekvensiell utvikling.

Produkt	Kap. (%)	Lansering	Oms. år 1	Oms. år 2	Oms. totalt
P1	54	11	3	48	51
P2	31	13	0	22	22
P3	15	13	0	11	11
Sum			3	81	84

Tabell 26: Parallell utvikling. Tallene i tabellen er avrundet.

A.5 Praktiske eksempler på prioritering uten nøyaktige estimater

A.5.1 Eksempel 1 Kundetilfredshet

En bedrift ønsker å gjennomføre tre initiativer for å forbedre kundetilfredsheten. De har følgende initiativer de skal prioritere:

- Initiativ A: Implementere en ny kundeportal for selvbetjening.
- Initiativ B: Starte et lojalitetsprogram for eksisterende kunder.
- Initiativ C: Forbedre kundeserviceopplæringen for ansatte.

Bedriften har ikke eksakte tallverdier for hvor mye hvert initiativ vil øke kundetilfredsheten, men de kan rangere initiativene etter hvor mye verdi de tror hvert av dem vil skape. De baserer vurderingene på erfaring og innsikt. De rangerer først initiativene innbyrdes etter verdi:

1. Initiativ A (kundeportal) mener de vil ha størst effekt på kundetilfredsheten fordi det gir kundene mer kontroll og øker tilgjengeligheten på tjenestene.
2. Initiativ C (kundeserviceopplæring) vurderes som nest viktigst fordi det direkte påvirker kundens opplevelse når de er i kontakt med bedriften.
3. Initiativ B (lojalitetsprogram) vurderes som minst viktig, selv om det fortsatt har en positiv effekt, men ikke like stor som de to andre.

De som gjør vurderingen mener at det er betydelige forskjeller mellom initiativenes verdi, og de velger skalaen 0-100:

- Initiativ A får verdi 100.
- Initiativ C får verdi 50.
- Initiativ B får verdi 25.

Med denne skalaen har de rom for å justere vurderingene, og eventuelt kunne vurdere andre initiativer senere. Neste steg er å rangere initiativene innbyrdes etter arbeidsomfang. Disse vurderingene er de mer usikre på, og de mener de ikke har grunnlag for en mer nyansert skala enn 1-3:

- Initiativ A (kundeportal): Dette vil sannsynligvis kreve mye arbeid, inkludert utvikling, testing og implementering. Omfanget vurderes som stort, og får tallverdien 3.
- Initiativ B (lojalitetsprogram): Dette vil kreve en viss innsats for å designe, markedsføre og administrere programmet. Omfanget vurderes som middels, og får tallverdien 2.
- Initiativ C (kundeserviceopplæring): Dette krever organisering av opplæringsøkter og materiale, men er ikke like omfattende som de andre initiativene. Omfanget vurderes som lavt,og settes til 1.

Vurderingene over er satt opp i tabell 27. Her har de også beregnet prioriteringsnøkkelen for sekvensiell gjennomføring c/T.

Initiativ	Verdi	Arbeidsomfang	Verdi/Omfang
A Kundeportal	100	3	33,3
B Lojalitetsprogram	25	2	12,5
C Kundeserviceopplæring	50	1	50

Tabell 27: Relativ vekting av verdi og arbeidsomfang.

I tabell 28 har de sortert initiativene etter prioriteringsregelen for sekvensiell gjennomføring. Fordi de også vurderer å gjennomføre initiativene parallelt, har de også bergent hva som er optimal kapasitetfordeling.

Initiativ	Rekkefølge	Kapasitetsfordeling (%)
C Kundeserviceopplæring	1	22,5
A Kundeportal	2	55
B Lojalitetsprogram	3	22,5

Tabell 28: Prioriteringsrekkefølge og kapasitetsfordeling med relativ vekting.

Kapasitetsfordelingen har de beregnet med formelen:

$$\frac{r_A}{r_B} = \sqrt{\frac{c_A T_A}{c_B T_B}}$$

osv.

Hvis de velger å gjennomføre initiativene parallelt, ser vi at initiativ B og C skal ha like stor andel av kapasiteten selv om de ikke skaper samme verdi, fordi verdi multiplisert med omfang gir samme tall: $c_B T_B = 25 \times 2 = 50, c_C T_C = 50 \times 1 = 50$.

Vi ser igjen at sekvensiell og parallell organisering av arbeidet gir ulik prioritering, og at vi kan gjøre viktige beregninger med kun relative tall for verdi og omfang ved å bruke formlene for samlet forsinkelseskostnad fra appendiks A1 og A2. Gjennomfører vi initiativene sekvensielt, ferdigstiller vi C før vi starter med A. Jobber vi parallelt, vil C få under halvparten av initiativ As kapasitet, og vi vil bruke over fire ganger så lang tid på initiativ C[14]. Det er mer enn vi bruker på å få ferdig både A og C hvis vi hadde jobbet sekvensielt. Med parallell jobbing bruker vi ca. 50 % lengre tid på å få ferdig alle tre initiativene. Beregner vi samlet forsinkelseskostnad for henholdsvis sekvensiell og paralell gjennomføring med tallverdiene over finner vi at sekvensiell gjennomføring gir oss for ca. 65 % raskere verdirealisering.

A.5.2 Eksempel 2 Rekruttering

I dette eksempelet ser vi på en rekrutteringsavdeling som skal skaffe dyktige folk til selskapet sitt. Avdelingen har tre ulike initiativer de vil gjennomføre for å øke rekrutteringstakten.

- Initiativ A: Implementere et nytt søkerhåndteringssystem ("Rekrutt").
- Initiativ B: Starte et program for å oppmuntre ansatte til å anbefale kandidater ("Tips").
- Initiativ C: Forbedre onboarding-prosessen for nye ansatte ("Velkommen").

De vurderer og rangerer initiativene etter antatt effekt på rekrutteringstakten:

- Initiativ A ("Rekrutt"): Dette vil forbedre effektiviteten i rekrutteringsprosessen betydelig, redusere tid til ansettelse, og gi bedre oversikt over kandidatene. Vurdert som høy verdi.
- Initiativ B ("Tips"): Dette kan redusere kostnader ved rekruttering og gi kandidater av høy kvalitet. Vurdert som middels verdi.
- Initiativ C ("Velkommen"): Dette kan forbedre nyansattes opplevelse og redusere turnover, men har ikke en umiddelbar effekt på rekrutteringsprosessen. Vurdert som lavere verdi.

De vurderer og rangerer initiativene etter antatt arbeidsomfang:

- Initiativ A ("Rekrutt"): Krever betydelig innsats for å velge, implementere og trene ansatte i bruk av systemet. Vurdert som stort omfang.
- Initiativ B ("Tips"): Krever innsats for å sette opp programmet og markedsføre det til ansatte, men er mindre omfattende enn A. Vurdert som middels omfang.
- Initiativ C ("Velkommen"): Krever mindre innsats å utvikle og implementere nye onboarding-rutiner sammenlignet med de andre initiativene. Vurdert som lav omfang.

De som vurderer initiativene mener de kan nyansere vektingen av både verdi og arbeidsomfang i den grad at de bruker skalaen 1-10:

Initiativ	Verdi	Arbeidsomfang	Verdi/omfang
A Rekrutt	10	10	1,0
B Tips	6	7	0,86
C Velkommen	2	3	0,67

Tabell 29: Relativ vekting av verdi og arbeidsomfang.

Ved å bruke formlene i appendiks A1 og A2 finner vi den optimale rekkefølgen hvis vi gjennomfører ett initiativ av gangen, og optimal kapasitetsfordeling hvis vi gjennomfører dem parallelt.

[14] Fordi initiativ C får mindre enn en fjerdedel av samlet kapasitet.

Initiativ	Rekkefølge	Kapasitetsfordeling (%)
A Rekrutt	1	53
B Tips	2	34
C Velkommen	3	13

Tabell 30: Prioriteringsrekkefølge og kapasitetsfordeling med relativ vekting.

I dette eksempelet ser vi at det er et visst samsvar i prioriteringene med sekvensiell og parallell gjennomføring. Gjennomfører vi ett initiativ av gangen, tar det bare ca. 15 % kortere tid å gjennomført alle tre initiativene sammenlignet med parallell gjennomføring[15]. Likevel blir det første initiativet ferdig på halvparten av tiden, og gir oss verdi mens vi jobber med de to andre. Hvis vi bruker tallverdiene våre, finner vi at sekvensiell gjennomføring gir oss for ca. 48 % raskere verdirealisering.

LinkedIn-profil.

[15]Samlet omfang ved sekvensiell gjennomføring er $10 + 7 + 3 = 20$. Ved parallell gjennomføring får initiativ C 13 % av kapasiteten. Cs omfang er 3. $3/0,13 \approx 23$ som er 15 % mer enn 20.