OBLIGATORISK OPPGAVE 1

INF1050 – Gruppe 9

Mars 2012

Henrik Hansen

henrik.hansen@student.jus.uio.no

Innholdsfortegnelse

1 Innledning

Jeg er under oppfatning av at det vil finnes klart flere fordeler enn ulemper med et sentralisert bookingsystem i forhold til et system for hvert enkelt flyselskap. Jeg har valgt å dele de inn to forskjellige kategorier der jeg tror et slikt system vil gi gevinst.

1.1 Service og proaktivitet

Fordeler:

Her tror jeg NorFly har mye å tjene på å investere i et sentralisert bookingsystem. Et slikt system vil utvilsomt øke serviceopplevelsen for sluttbrukeren, da det er mye lettere å bestille på tvers av flyselskapene. Hvis servicepersonellet *manuelt* må aksessere de forskjellige systemene for fly-booking vil dette ta merkbart lengre tid enn hvis alt var på samme sted. Jeg tror en god antakelse er at jo lengre tid det tar, og jo mindre det «går på skinner», jo mer uprofesjonelt vil personellet virke, som igjen vil reflekteres over på flyselskapet, og til slutt på konsernet.

Det vil være enda mer synlig hvis passasjerene skal på «connecting flights», og samtlige av de involverte flyselskapene er en del av NorFly. Kunder kan trolig slå seg til ro med at det går sent i én serviceskranke, men hvis det skjer flere ganger, når kunden attpåtil gjerne er stresset fra før av, er dette meget uheldig for NorFly som konsern. Det samme gjelder hvis kunden ønsker å bestille på gjennom et webgrensesnitt. Opplevelsen vil være bedre hvis alle selskapene er på samme side.

Jeg antar at det også er en stor fordel for konsernet *direkte* at systemet blir sentralisert. Hvis de rette folkene hele tiden blir matet med nyttig informasjon fra et system, er det mye lettere å forutberegne situasjonen, og dermed bli mer proaktive. Jeg ser for meg et system som for eksempel varsler servicepersonellet når de respektive flyene er fulle, og automatisk varsler om andre flyavganger som kan være aktuelle for kundene. Systemet burde selvfølgelig kunne varsle om andre ting også, men jeg mener stikkordet er forutsigbarhet, så det blir en så smertefri kundebehandlingsprosess som mulig.

Ulemper:

Ut ifra «service og proaktivitet»-perspektivet ser jeg ikke noen ulemper, så lenge systemet fungerer som det skal, og møter kravspesifikasjonene til kunden. Men et slikt system vil være ganske stort, og kan inneholde flere bugs som kan virke irriterende på kunden hvis dette medfører lengre behandlingstid. Dog kan nok de fleste bugs unngås med skikkelig testing i de respektive fasene i programmeringen.

1.2 Økonomi

Fordeler:

Et sentralisert system som fungerer som det skal vil utvilsomt være både tid- og kostnadseffektivt. Hvis kompetansen til servicepersonellet er sentrert rundt kun et system, er dette uheldig med tanke på optimal bruk av personellressurser. Hvis konsernet har mangel på personell i et flyselskap i et kort tidsrom kan det være ønskelig å midlertidig forflytte personell fra et flyselskap til et annet, noe som blir vanskelig hvis systemene er relativt ulike. Her vil jeg tro at det er en god del penger å spare, ihvertfall i det lange løp.

Siden konsernet har flere nasjonale og internasjonale ruter med forskjellige flyselskap, hender det nok ofte at de har flere fly som flyr på omtrent samme tidspunkt til samme destinasjon. Her tror jeg det er potensiale til å tjene mer penger med et sentralisert system, enn flere uavhengige systemer. Hvis systemet gir ut konkret informasjon om de forskjellige rutene til servicepersonellet, kan man lett se hvor det er mest penger å tjene på å booke en passasjer. Kunder har forskjellige behov som de ønsker oppfylt, og ved spesielle behov kan flyselskapet ta mer penger uten at kunden blir lurt på noen måte. Det er en «win-win»-situasjon. Et eksempel på dette kan være at systemet sier at «Norwegian» ikke opererer med førsteklasse-billetter, men «SAS» gjør det, mot 1000,- ekstra. Denne informasjonen kan være vanskelig å snappe opp for servicepersonellet hvis du samhandler med flere systemer.

Ulemper:

Et nytt system krever sannsynligvis ulik grad av opplæring, noe som kan koste ganske så mye i et stort internasjonalt konsern. Folk er forskjellige – noen tar ting fort, mens med noen kreves det lengre tid. Ledelsen bør være pessimistiske i antakelsen om hvor lang tid personellet vil bruke på å lære systemet skikkelig, så de ikke går på en økonomisk eller personell smell. Dog er nok dette en liten pris å betale, for gevinsten i det lange løp.

Et slikt omfattende system tar selvsagt lang tid å utvikle skikkelig. Budsjettrammen må være elastisk, eller i det minste må skadeomfanget være relativt lite hvis budsjettrammen sprenges. Det er flere eksempler på at sprengte budsjettrammer kan få katastrofale følger for hele prosjektet, for eksempel i den norske stat.[1] Det må i det minste settes av tilstrekkelig med midler for et så omfattende prosjekt, og det er nok greit å være pessimistisk i antagelsene her også.

Det finnes muligheter for suboptimalisering ved at for eksempel: (i) En interessent «skriker høyest» og får mesteparten av sine ønsker oppfylt, mens andre blir satt på sidelinjen. (ii) De som er med på kravspesifikasjons-runden tenker kun på sine behov(feks økonomiske) og glemmer sluttbrukerens behov. (iii) Fokus på funksjonaliteter blir prioritert over brukervennlighet.

[1] Bedre samordning og styring av store og/eller strategisk viktige IKT-prosjekter i staten(pdf) - http://www.regjeringen.no/upload/FAD/Vedlegg/IKT-politikk/Bedre_IKTsamordning.pdf (21/02-12)

2 Interessenter

2.1 Interessenter i et systemutviklingsprosjekt

Det kan være flere interessenter(«stakeholders») i et systemutviklingsprosjekt, men jeg tror de viktigste vil være de som er nevnt under. Jeg har prøvd å sette det opp med de interessentene som har mest interesse av systemet øverst. Det er viktig å få med seg at kunde kan både være potensielle flypassasjerer, og kunde som i «kunde som kjøper systemet». Kunde som kjøper systemet vil ofte være ledelse, og jeg har prøvd å skille «kundene» så godt som mulig.

- Alle som har et ønske om at systemet skal bli utviklet.
  - Først og fremst er det interessentene på utviklingsstadiet. Med det mener jeg dem som er med på å utvikle systemet på en eller annen måte. Dette inkluderer gjerne kunderepresentant,utviklerteamet og eventuell prosjektleder(eller «ScrumMaster» e.l.) og testteam.
  - Kunden er også interessert i at systemet skal bli utviklet. Dette inkluderer ledelsen, sluttbrukerne, flypassasjerer, investorer/sponsorer også videre.

- Alle som er interesserte i systemet på en eller annen måte.

  - Denne gruppen kan variere veldig fra system til system. Men noen system må for eksempel kontrolleres eller lignende. Da kan myndighetene, i form av en arbeidsgruppe, kontrollorgan, eller lignende være en interessent fordi de må kunne *forstå* systemet for å kunne gjøre sin jobb. Det samme gjelder også hvis det er et ekstern organ som ikke er statlig som må kunne aksessere systemet. Det er likegyldig om det er lovbestemt kontrollering eller ikke, hvis noen andre en den tradisjonelle sluttbrukeren skal bruke systemet, må det legges opp til det.

  - Denne gruppen har egentlig ingen begrensninger hvem som kan være i den. Men alle har selvsagt ikke noe de skulle sagt for det om. For eksempel vil gjerne en konkurrent være stakeholder, men konkurrenten vil ikke ha noe innflytelse med tanke på utviklingen av systemet selv om han er en interessent.

2.2 Interessenter i NorFly-prosjektet

Jeg har valgt å sette opp interessenter i systemet til NorFly, som jeg ser det, i en tabell for å gjøre det mest mulig oversiktlig. Sluttbruker kan være både potensiell flypassasjer(ved å bruke webgrensesnittet) eller serviceskrankepersonell o.l.

Interessenter	Interesser	Ansvarsområde
Kunde(i form av ledelse, ikke sluttbruker)	Å få et sentralisert system som er effektivt(på alle mulige måter) og intuitivt, som gir god oversikt over passasjerer, flyavganger, priser m.m. Et velfungerende system i bunn og grunn.	Uavhengig av hvilken sys.utv.pros som skal brukes, må *kravspecsene* være så gode som mulig, og gjenspeile sluttbrukers, samt kundes behov, sånn at systemet kan bli så nærme kundes ønske som mulig. Formidle dette videre til kunderep. Sette av nok midler.
Kunderepresentant	Hovedansvar->Fornøyd ledelse/sluttbrukere. Formidle sluttbrukernes/ev. andre interessenters ønsker til utviklerne. Overblikk over progresjon, komme med innspill underveis.	Formidle sluttbrukerenes/ev. andre interessenters ønsker til uviklerene. Derfor være godt informert om kundens ønsker til systemet. Holde ledelse godt informert underveis om status.
Utviklerteamet	Velfungerende system, uten «bugs», som kunden er fornøyd med. Tid-, kostnad- og ressurseffektivt. Fornøyde kunder/sluttbrukere.	Hovedansvar -> Få et system som kunden har bruk for og som fungerer som det skal. Bør ikke overstride budsjett- og tidsrammer. Følge Prosjektleders e.l. sine instrukser.
Prosjektleder (eller tilsvarende)	Samme som utviklerteamet, men på et overordnet plan. Hovedinteresse -> Gi kunden det han trenger.	Samme som utviklerteamet, men på et overordnet plan. Holde kunderep. underrettet om progresjon underveis, være så tilgjengelig som mulig for kunde. Være utviklernes ansikt utad.
Sluttbruker – ansatt i NorFly	Burde være det samme som kunde's interesser(pkt 1), men på et mer konkret plan. Å få et system som er raskt, effektivt og lett/intuitivt å bruke. Det må huskes at det mange forskjellige selskaper som er representert, og har egne ønsker, så denne gruppen blir veldig stor.	Rapportere «bugs» til kunderep, komme med forslag til funksjoner(inkrementer hvis smidig sys.utv.pros. er/blir brukt.)
Kunde(i form av flypassasjer, ikke kunde av systemet - ledelse)	Å komme presis til destinasjonen, med best mulig service. Burde ikke merke systemet i det hele tatt – gå så glatt som mulig. Kan ha også ha	Finne fram til serviceskranken. Ha med pass/ID... Eventuelt bruke webgrensesnittet.

	«særønsker», som for eksempel førsteklasse-billetter, å få sjekket bagasjen fram til siste destinasjon, skifte fly og mye mer, og burde få oppfylt dette, så langt det er mulig. Ellers ønske om et vellfungerende webgrensesnitt	
Datatilsynet/personvernnemnda	Sjekke at personopplysninger blir behandlet på en skikkelig måte, med tanke på en utrolig stor mengde personopplysninger i et sentralisert register. Garantert interessant, jf personopplysningsloven[1]. Må derfor legges opp til kontroller o.l.	Kontrollere at NorFly følger rettsreglene mtp behandling av personopplysninger, jf gjeldende lover.

3 Utviklingsprosess

3.1 Argumenter for å velge plandreven utviklingsprosess ved «fossefallmetoden»

Det kan være flere grunner til at en plandreven utviklingsprosess er foretrukket. Men hvis plandreven utviklingsprosess i det hele tatt skal vurderes, bør det tenkes på om kravspesifikasjonene er beskrevet godt nok. Hvis så og si hele systemet kan spesifiseres på forhånd, og det er mulig å forutse alle funksjonene systemet skal ha på forhånd og relativt lite må forandres på underveis eller i etterkant kan det være en fordel å velge plandreven utviklingsprosess, ved å bruke fossefallmodellen(«the waterfall model») Hvis dette ikke er mulig, uten særdeles mye arbeid, er nok andre metoder bedre å bruke, eller en kombinasjon av forskjellige metoder.

Det som kan være en fordel ved å bruke fossefallmetoden, slik jeg ser det, er at koordineringen av arbeidet vil sannsynligvis bli mye lettere. Dette er nyttig da et slikt system som NorFly ønsker seg, som skal samhandle mellom alle de forskjellige flyselskapene, nødvendigvis vil være særdeles komplekst, og ikke minst stort. Da kan en klar og veldefinert plan hjelpe prosjektlederen med å strukturere arbeidet og fordele arbeidsressursene smart. Fossefallmodellen krever normalt at arbeidet du gjør dokumenteres nøye etter hver fase, så prosjektlederen kan enkelt se hvordan teamet

[1] LOV 2000-04-14 nr 31: Lov om behandling av personopplysninger (http://www.lovdata.no/all/hl-20000414-031.html (Hentet 20/02-12))

ligger an i forhold til planen(e). Denne informasjon kan være veldig viktig for kunden. Inkrementell utvikling har også vist seg å være mest effektiv når det utvikles mindre systemer, gjerne der teamet kan være på samme plass, og «uformell diskusjon» er lett å få til, og her kreves det er stort team sannsynligvis.[1]

Det er også påstått at systemer som skal ha veldig lang levetid, krever veldig bra dokumentasjon, som er en av kjennetegnene til plandrevet utvikling. Dette er fordi da må gjerne forskjellige team være borti samme system, og hvis det er dårlig teknisk dokumentert, kan det bli problematisk å forandre på koden. (Dog mener noen tilhengere av smidig utvikling at dette er tøv, siden ofte så glemmes det å oppdatere dokumentasjonen når systemet oppdateres. Det burde da heller fokuseres på ryddig og fin kode, som igjen er lett forståelig for neste team)[2] Det sies også at det kreves mindre kompetanse av utviklerne ved fossefallmetoden.[3] I store, veletablerte firmaer er det ofte normal praksis å operere med plandreven systemutvikling, siden det er det «vi alltid har gjort». Da kan det være vanskelig for noen ildsjeler å «banke igjennom» et forslag om å heller skifte til en smidig metode. Da kan det heller være en fordel å holde seg til plandreven metode, da det ikke krever massive ressurser for å lære arbeiderne nye arbeidsmetoder.

Argumentene for å velge plandreven systemutviklingsmetode blir kort forklart da:

- Lettere å strukturere arbeidet og ressursene. Spesielt hvis teamet befinner seg forskjellige steder, geografisk.
- Ofte bedre dokumentasjon til kunden, og fremtidige utviklingsteam.
- Åpner for bedre kontroll av progresjon versus budsjett/planer.
- Utviklerfirmaet har dette som normal «business-practise», og det vil være får tid- og/eller kostnadskrevende å gå over til en annen metode, spesielt hvis teamet og/eller firmaet er stort og har veletablerte retningslinjer å følge.
- Krever gjerne mindre kompetanse av utviklerne enn ved smidige metoder. Dette fordi at kravene er så detaljerte og veldefinerte, at det er bare å transformere det til programkode – krever minimalt med fantasi.

3.2 Fordeler og ulemper ved bruk av «Scrum»

Her har jeg forsøkt å sette opp en liste over hva jeg ser på som fordeler og ulemper ved å ta i bruk Scrum.

Fordeler ved å bruke Scrum:

- Prosjektet, som er forholdsvis stort, blir brutt ned i mindre og mer forståelige deler, og hele teamet vet hele tiden hvor langt de har kommet. Dette vil mest trolig lede til en raskere og mer effektivt bruk av tiden til teamet. Et system som er “fungerende” vil bli levert mye raskere ved Scrum enn ved fossefallsmetode for eksempel.

---

1 Ian Sommerville – Software Engineering, 9th Edition(2010) Kap 3.2

2 Sommerville(2012) Kap 3.2 pkt 5.

3 Sommerville(2012) Kap 3.2 pkt 9.

- Bruk av Scrum, og andre smidige metoder, gjør det særdeles mye lettere å implementere forandringer i systemet. Etterhvert som tiden går, får du et bedre og mer modent system, da oppdateringene blir levert i inkrementer i gitte tidsintervaller, gjerne vær 2-4 uke.
- Utvilsomt vil det ønskes nye funksjonaliteter, siden det er så mange interessenter som har ønsker om hvordan systemet skal være. Ved bruk av Scrum vil det evalueres hvordan de ligger ann i forhold til kravene/ønskene, etter hver sprint. Dette vil også si at risikoen for et "katastrofeprosjekt" blir minimalisert, siden systemet er oppe til evaluering i løpet av hver sprint.

Ulemper ved å bruke Scrum:

- At «user storiene» skal være korte og konsise, kan medføre at kravene ikke blir spesifisert så bra. Dette kan igjen medføre at prosjektet sklir langt utover planer/budsjetter. User stories kan derfor gjøre det vanskeligere å estimere hvor lang tid, og hvor mye det vil koste å utvikle et system, spesielt hvis det er stort.
- Det er vanskelig/umulig å gjennomføre hvis ikke alle i teamet er plassert på samme sted, siden denne modellen krever mye korte uformelle møter,samtaler o.l.
- Scrum krever også høyere kompetanse på utviklerne enn en tradisjonell fossefallsmetode, eller tilsvarende.
- Krever mye involvering og input fra en kunderepresentant, som igjen må vite hva sluttbrukerne og resten av interessentene vil ha. Dette krever mye tid, og hvis dette er noe NorFly ikke kan avse, blir det vanskelig å gjennomføre.

3.3 En liste over user stories i prioritert rekkefølge

1. Som en flypassasjer(kunde), ønsker jeg å kunne bestille flybilletter til alle flyselskapene i konsernet, uten unødvendige hindre og registreringer(alt på en plass), for en best mulig opplevelse.
2. Som en (slutt)bruker av systemet, ønsker jeg å se hvilke fly som er i rute og ikke, for å være oppdatert hele tiden.
3. Som (slutt)bruker av systemet, ønsker jeg å se hvilke flyselskap som flyr hvor, og hva prisforskjellen på de forskjellige selskapene er for mest mulig inntjening, og for å kunne yte best mulig service til kundene.
4. Som bruker av systemet, ønsker jeg et oversiktlig og "light-weighted"-system, som "alle" kan bruke.
5. Som bruker av systemet, ønsker jeg en lett og oversiktlig måte å hente ut statistikk på, i form av grafer og tabeller, for økonomiske hensyn m.m.
6. Som kunde(investor, ledelse etc) ønsker jeg å se hvilke fly som har fylt opp mindre enn 60% av kapasiteten sin, for økonomiske hensyn.
7. Som (slutt)bruker av systemet, ønsker jeg samhandling med de største hotell- og leiebilselskapenes systemer, så flypassasjerene lett kan bestille hotell og leiebil igjennom vårt system.
8. Som flypassasjer(kunde) ønsker jeg se hvilke flyselskap som tilbyr førsteklasse("first class") og enkel oversikt over prisforskjell på de forskjellige "klassene", for personlig hensyn.

9. Som kunde(investor, ledelse etc) av systemet, ønsker jeg samhandling med et forsikringsselskaps system, sånn at vi kan tilby avbestillingsforsikring og ulykkesforsikring på én og samme plass til passasjeren vårs.
10. Som bruker/kontrollør av systemet(datatilsynet) ønsker jeg full tilgang til systemet, eller mulighet for å raskt få dette, for å drive kontroll med behandling av personopplysningene i systemet.

3.4 Tre-ukers sprint med Scrum

I den første sprint backlogen ville jeg tatt med punkt 1-3. Dette er fordi at dette er essensielle funksjonaliteter som programmet må ha med for å i det hele tatt ha nytteverdi.

4 Prosjektplan ved plandreven utviklingsprosess.

4.1 Detaljert prosjektplan

1. Introduksjon

   Dette systemet har som hovedoppgave å fungere som et "bookingsystem" for flyreiser. De som skal bruke systemet er primært ansatte i et flyselskap underlagt NorFly, eller flypassasjeren direkte, igjennom et webbasert grensesnitt. Det webbaserte grensesnittet skal være et tillegg til systemet som de ansatte i flyselskapene bruker. Det som skal prioriteres høyest, er at systemet skal bli levert raskt og at prosjektet ikke overskrider budsjettrammen noe særlig. Brukervennlighet skal prioriteres høyere enn å ha mange "fancy" funksjoner. Budsjettrammen er satt til 15 millioner kr, over en tidsramme på 120 uker, men begge deler er av en viss fleksibilitet.
2. Oversikt over personellresursser

   Teamet skal bestå av 6-8 programmerere, pluss en prosjektleder. Først og fremst skal kravene spesifiseres nøye. Programmererene skal være med på denne prosessen, så langt det er nødvendig. Prosjektlederen skal være med hele tiden, sammen med kunderep. Programmererene har sine fleste oppgaver under implementasjonsfasen, og hvem som skal gjøre hva blir etablert senere(på møter og lignende). Andre involverte vil være et separat testteam, og kunden, såfremt kunden selv ønsker det.
3. Risikoanalyse

   Det er en god del ting som kan gå galt i dette prosjektet. Jeg har prøvd å sette opp de mest aktuelle risikoene som kan oppstå, punktvis. Jeg kommer mer innpå risikoene i 4.2

   - Kapasitet på systemet kan være for liten, slik at belastningen blir for stor,og systemet klarer ikke å prosessere alle henvendelsene, 24/7.
   - Da dette systemet vil ta relativt lang tid å utvikle, kan ledelsen skiftes ut i dette tidsrommet, og dette kan medføre nye krav, eller i verste fall avbrytelse av prosjektet.
   - Nye krav kommer til. Mer problematisk hvis plandreven systemutvikling er valgt.
   - Utviklerne slutter, eller blir utilgjengelige av andre årsaker
   - Det er ikke mulig å rekruttere utviklere med rett kompetanse
   - Flyselskap kjøper seg ut av NorFly, systemet må forandres på. Dette kan skje når som

helst, gjerne lenge etter at utviklerteamet er ferdig med prosjektet.
- Tids- og eller budsjettramme “sprenges”
- Systemet inneholder sikkerhetshull som er vanskelige å oppdage.

4. Krav til innkjøp av hardware og software

   Alt som trengs av hardware og software er allerede på plass, da utviklerselskapet er relativt godt etablert. Dette inkluderer “programmeringskit” for de fleste programmeringspråk, samt hardware å kjøre og teste systemet på.
5. Oppdeling av arbeidsoppgaver

   Aktivitetene som skal gjennomføres er som følger, i prioritert rekkefølge, hvis vi går ut ifra at vi bruker fossefallmetoden: (i) Kravspesifisering. Her møtes interessentene og utviklerne til en diskusjon og spesifisering av kravene. Dokumentasjon blir laget. Både funksjonelle og ikke-funksjonelle krav skal/kan spesifiseres her. (ii) Analyse og Design. Her blir kravene omgjort til et til en arkitekturelt design. (iii) Koding og implementasjon. Her blir designet omgjort til faktisk programvare. Dokumentasjon lages. (IV) Testing av programvaren. Blir ofte gjort av eget testteam.(V) Installering av systemet. (VI) Vedlikehold. Her blir programvaren «patchet» for å møte nye krav eller for å fikse «bugs». (VII) Dokumentasjon for hele prosjektet. Dette inkluderer alt av dokumentasjon, både teknisk og til sluttbruker. Milepælene i prosjektet er forklart i aktivitetsplanen i 5.1.

6. Tidsplan for prosjektet

   Hvor lang tid hver aktivitet vil ta er vanskelig å forhåndsbestemme, men det regnes med at kodingen vil ta ca 50% av planlagt tid. Analyse og Design ca 10-15% av tiden, testing og feilretting 10-15%. Resten av tiden går til det resterende arbeidet. For øvrig er det regnet med utviklingen av hele systemet vil ta rundt 120 uker.
7. Dokumentasjon

   Etter hver fase/aktivitet skal det som er nødvendig at dokumentasjon føres, og gis til kunden når prosjektet er ferdig.

4.2 Usikkerhetsmatrisen vil finnes til slutt i dette dokumentet, pga tekniske problemer med å få det inn her...

5 Aktivitetsplan, søylediagram og PERT-diagram.

Innledning

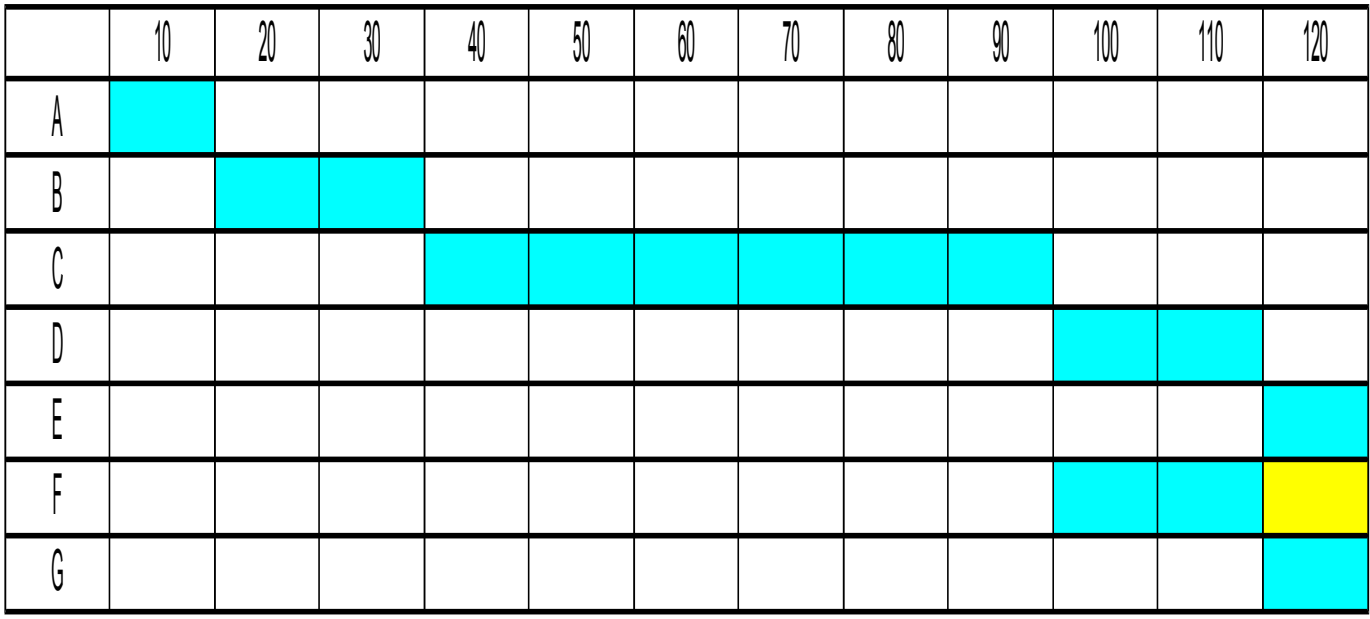
Den kritiske veien er merket med røde uthevete linjer I PERT-diagrammet. Røde linjer fungerer også for å vise avhengighet(kombinert kritisk vei og avhengighet for ordens skyld, ellers hadde det blitt så mange streker) og det samme gjør grønne linjer. Den kritiske veien går da igjennom A-B-C-D-E-G.

Søylediagrammet fungerer som en oversikt over delaktivitetene i utviklingsfasen, med verdier i henhold til aktivitetsplan som er over. Y-aksen er «aktivitetsnavn», og X-aksen er tid i uker. Blå viser til hvor lang tid aktiviteten vil ta i uker, og slakken er merket med gult. Milepælene er merket i aktivitetsplanen som M1, M2, M3 og M4 under *Aktivitetsnr.*

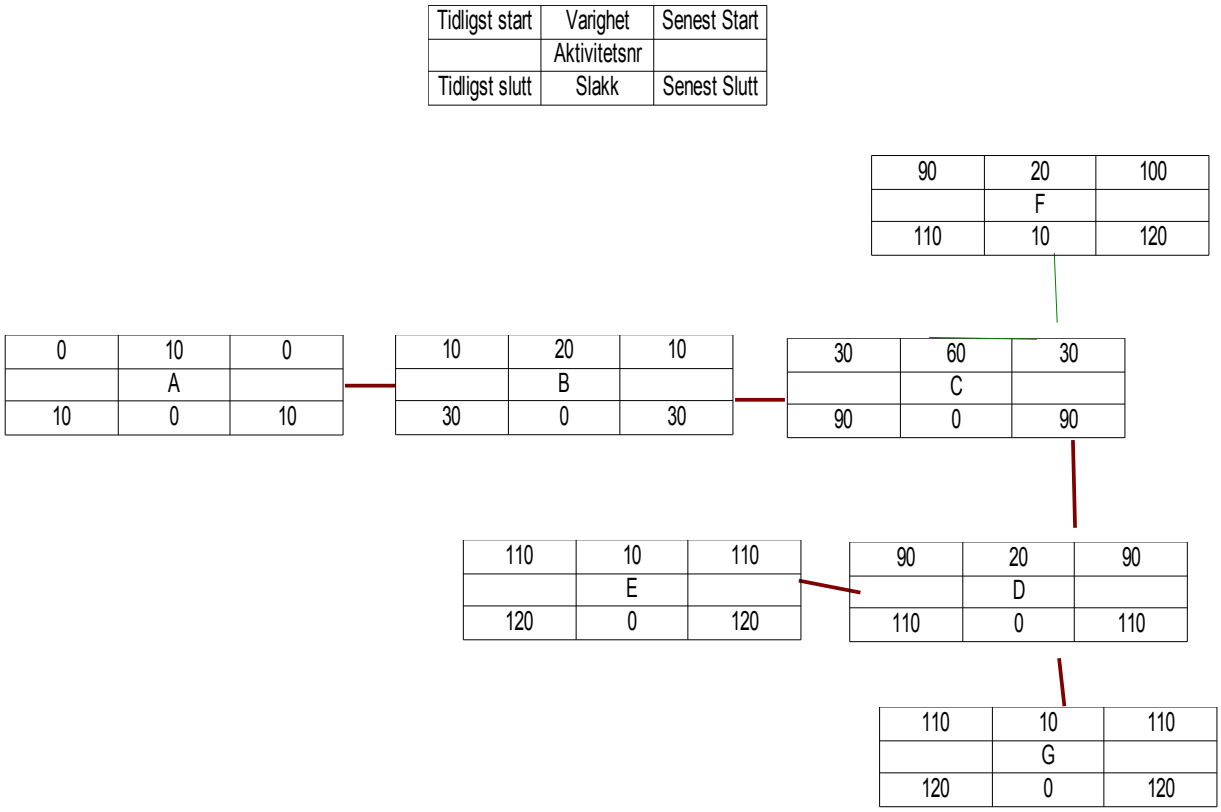
5.1 Aktivitetsplan

AKTIVITETSNR.	AKTIVITET	TIDSFORBRUK(UKER)	AVHENGIGHET
A	Kravspesifisering	10	N/A
B(M1)	Analyse & Design	20	A
C	Koding og implementasjon	60	B
D(M2)	Testing av systemet	20	C
E(M3)	Installering av systemet	10	D
F	Vedlikehold av Systemet	20	C
G(M4)	Dokumentering	10	D

5.2 Søylediagram over delaktivitetene



5.3 PERT-diagram over delaktivitetene



6 Funksjonelle- og ikke-funksjonelle krav

6.1 Funksjonelle krav

- Systemet må kunne ha oversikt over hvilke fly innad konsernet som går når og fra/til hvor.
  - Systemet må ha automatisk og manuell innsamling av denne type informasjon.
  - Informasjonen må organiseres på en måte som gjør at et sånt system er mulig.
- Systemet må kunne ta imot bestillinger på flyreiser for samtlige flyselskap
  - Må ha full oversikt over alle flyselskapene i konsernet, se overstående krav.
  - Må tillate input fra registrert kunde(se et par punkt under), og ikke tillate «ugyldig» input. Kunde må få valg om å skrive ut billetter på automater på flyplass, eller fra sin egen datamaskin.
- Systemet må kunne vise prisforskjell på de forskjellige flyselskapene på identiske flyruter.
  - Første punkt må være oppfulgt.
  - Må vises fint grafisk strukturert.
- Det må kunne registreres nye brukere i systemet.
  - Systemet må ha et registreringsskjema der nødvendig info blir inputtet.
  - Systemet må ikke tillate at brukeren får gå til neste steg før all obligatorisk informasjon

er fylt inn i registreringsskjemaet, og at verdiene er «gyldige».
  - Hvis alt er greit, må systemet lage brukeren med korrekte rettigheter.
- Det må kunne bestilles leiebil, hotell og diverse forsikringer fra systemet.
  - Systemet må være kompatibelt med de aktuelle systemene, sånn at bestilling, rapportering osv kan gjøres mellom systemene.
- Systemet må ha både en lokalversjon, samt et webgrensesnitt(som bygger på lokalversjonen)
  - Se andre krav om hva systemene må kunne gjøre.
  - Må kunne være kompatibel med diverse internettprotokoller o.l. for å kunne ha et webgrensesnitt for flypassasjeren.
- Systemet må kunne oppdage inntrengere på systemet.
  - Systemet må ha diverse mekanismer for å oppdage og eliminere disse truslene(Avanserte IPS'er/IDS'er)
- Systemet må kunne printe ut statistikk, i form av grafer o.l., i .pdf-format.
- Systemet skal gi forskjellige rettigheter i systemet og webgrensesnittet, alt etter om de er en ansatt, administrator eller potensiell flypassasjer som bruker systemet.
  - Kan gjøres på mange forskjellige måter. Feks samkjøre med ansatte-lister.

6.2 Ikke-funksjonelle krav

Produktkrav	Organisatoriske Krav	Eksterne Krav
Systemet må være «døgnåpent». Ingen/minimalt med nedetid.	Det må alltid være tilstrekkelig med kompetente programmerere.	All omsetning må føres på en forsvarlig måte, og skattes etter loven.
Systemet må være sikkert.	Det må prioriteres å bruke gjenbruk i utviklingsfasen.	Personopplysninger må lagres sikkert og forsvarlig.
Systemet må være brukervennlig.	Alt av fravær i teamet må planlegges og eventuelle vikarer må være på plass i god tid.	
Det må være mer enn nok plass på serverene.		
Systemet må være raskt.		
Systemet må inneholde ingen eller minimalt med feil.		

Produktkravene kan måles ved å gjøre brukerundersøkelser. For eksempel kan brukervennlighet måles ved å se hvor mange feil brukerne gjør i gjennomsnitt, og se hva som er den største grunnen til feilene. Hastigheten på systemet kan måles ved å se hvor lang tid det i gjennomsnitt tar for sluttbrukeren og gjøre ønskete handlinger, og finne ut eventuelt hvor

flaskehalsene er. Dette måles ved å se hvor lang tid det tar før brukeren kaller på metoden/operasjonen til oppgaven blir utført. At systemet inneholder minimalt med bugs kan måles med hvor mange feilmeldinger som kommer opp i testfasen, eller ved andre brukerundersøkelser. Det kan også måles ved å saumfare/korrekturlese koden.

Hvor sikkert systemet er kan måles ved å gjøre stress-tester / «penetration» tester. Det gjøres ved å prøve ut diverse metoder for å prøve å aksessere systemet, uten å ha tilstrekkelig med tilgang. Ellers er det et valg man må ta, hvor sterk kryptering NorFly ønsker, og hvor høy grad av autentisering som behøves. Pålitelighet, og ønsket om å ha systemet «døgnåpent», kan måles ved å se hvordan oppetiden er i prosent. Det kan også måles med hvor stor feilraten er - «mean time beetween failures»(MTBF). Størrelseskravet må verifiseres ved å regne ut hvor mye plass som kreves mot hvor mye som er tilgjengelig. Det må også regnes med litt «slingringsmunn», det er bedre med for mye plass enn for liten. Det må kontinuerlig sjekkes at det er nok plass på serverne. Det som er felles for alle kravene er at de må passere testene av kravene for å bli godkjent. Hvis kravene ikke blir oppfylt, må de fikses slik at det er bra nok, før de kan aksepteres.

At gjenbruk i utviklingsfasen hele tiden må prioriteres, kan måles ved å se hvor mange prosent av programvaren som består av gjenbrukt programvare, og måle det opp mot kravet. At hele tiden må være nok utviklere kan måles ved å hele tiden sjekke det opp mot prosjektplaner, og hele tiden bruke «bedre-føre-var-prinsippet». Det samme gjelder hvis det kreves vikarer eller lignende. De eksterne kravene jeg har tatt med, må måles ved å føre statistikk, og se hvordan de ligger ann i henhold til lovverket. Hvis personopplysningsloven feks sier at det kreves tilstrekkelig sikkerhetstiltak for å sikre opplysningene, må dette sjekkes opp mot hva som faktisk blir gjort. Er det ikke bra nok, må det fikses før det kan aksepteres.

Usikkerhets-element #	Beskrivelse av usikkerhetselement	Kundens vurdering		Leverandørens vurdering		Tiltak	Ansvarlig
		Sannsynlighet	Konsekvens	Sannsynlighet	Konsekvens		
1	Belastning på systemet for stor, og systemet klarer ikke å prosessere alle henvendelser, 24/7.	Stor	Stor	Stor	Stor	Å lage systemet og det systemet avhenger av med nok kapasitet til å behandle nok henvendelser på en gang. Ingen flaskehalser.	Utvikler/Leverandør.
2	Utskifting av ledelse. Kan medføre at ny ledelse har andre krav, eller i verste fall annullering/avbrytelse av prosjektet.	Middels/Stor	Middels/Stor	Middels/Stor	Middels/Stor	Prøve å forutse slike ting så bra som mulig, selv om det kan være vanskelig. Tilpasse seg ny situasjon/nye krav på best mulig måte.	Utvikler(å være tilbøyelig for tilpassing) og Kunde.
3	Nye krav generelt.	Stor	Middels	Stor	Middels	Være tilbøyelig for nye krav. Bruke lang tid på kravspecsene i første omgang, så muligheten for at nye krav dukker opp underveis minimeres.	Utvikler, kunderep. De som er med på kravspesifikasjonsmøtene.
4	Utvikler(e) slutter, eller blir utilgjengelig av andre grunner.	Middels	N/A	Middels	Katastrofalt	Såfremt det er mulig, alltid ha backup av personell, slik at det ikke bare er en person som kan gjøre en ting.	Utvikler/Leverandør.
5	Det er ikke mulig å rekruttere utviklere med høy nok, eller rett, kompetanse.	Lav	Middels/Stor	Lav	Stor	Alltid ha utviklere som har rett, eller god nok, kompetanse. Bedre-føre-var prinsipp!	Utvikler/Leverandør
6	Flyselskap kjøper seg ut av NorFly. System må forandres deretter.	Stor	Middels	Stor	Middels	Dette kommer til å skje fra tid til annen. Bedre-føre-var prinsippet burde spille inn her og, og sånne ting må forutsees og planlegges så forandring i systemet går smertefritt.	Utvikler/Leverandør og Kunde(å holde utvikler oppdatert om sånne ting)
7	Tids- og/eller budsjettrammen «sprenges»	Middels/Stor	Middels/Stor	Middels/Stor	Middels/Stor	Her er det mye som kan ha noe å si. Feks god planlegging, bra team, bra kommunikasjon er med på å forebygge at dette skjer.	Kunde og Utviklere/Leverandør.
8	Systemet inneholder sikkerhetshull som er vanskelige å oppdage	Stor	Middels/Stor	Stor	Middels/Stor	God testing. Konstant sjekke etter dårlig kode.	Utviklere/Leverandør