Reflektionsrapport över projekt

1dt908 – VT24

Jesper Wingren
Jw223rn@student.lnu.se

Innehållsförteckning

1. Komplexitet i mjukvarudesign och utveckling 3

2. Begrepp inom mjukvaruutveckling 4

3. Agila metoders principer 5

4. Metoder i projekt 6

5. Effektiva grupper 7

6. Mjukvaruutvecklare som individ 8

6.1 Vad innebär mjukvaruutvecklare 8

6.2 Arbetsuppgifter 8

6.3 Lärdomar från gästföreläsningar 8

6.4 Kopplingar mellan projektarbetet och yrket 8

6.5 Min lämplighet som mjukvaruutvecklare 8

7. Litteraturförteckning 10

1. Komplexitet i mjukvarudesign och utveckling

Det första som gör mjukvaruutveckling komplext är den individuella påverkan man har på designen. Det finns standarder och liknande på hur en kodbas ska byggas upp men i slutändan är koden påverkat av den individ som programmerat och hur den individen ser på problemet. Större projekt kan innehålla flera hundra tusen rader kod som utvecklats genom åren och kanske av flera utvecklare. Ska man sedan sätta sig in och förstå all kod som skrivits krävs det många timmar av läsning då det som sagt varierar från person till person hur man programmerar. Detta påverkas även av hur välstrukturerad koden är och hur olika delar samarbetar med varandra [1] [2].

En annan faktor är komplexiteten av programmering och dess olika språk. Att förstå sig på alla språk och hur de kan kombineras samt hur dessa används optimalt är omöjligt. De flesta utvecklare fokuserar därav på ett mindre antal språk där de lär sig mer avancerade koncept och sedan lär sig grunder av andra språk. En bra utvecklare kan lite av allt men mycket om lite, vid ett större projekt är det sällan väsentligt att förstå sig på alla delar av koden men att ha en grundförståelse för det mesta krävs för att kunna vara delaktig på ett bra sätt [1] [2].

Den faktor som är mest aktuell för just mjukvara är att dess livscykel oftast är lång eller oändlig d.v.s. när man utvecklar en applikation ska den vara funktionell längre och även klara större mängder data genom tiden. Detta är till skillnad från en produkt som kanske har en 10-årig livscykel eller till och med en 100-årig så finns det någon gång den tar slut eller har förbrukats. Detta är sällan fallet med mjukvara utan det förväntas vara skalbart, det vill säga gå att utveckla om den växer i antalet användare men även hänga med i teknikens utveckling. Om en mjukvara inte är det så försvinner den ofta ur marknaden efter ett visst tag då den inte uppfyller de krav som sätts. Det är något av de svårast för en utvecklare då de oftast är många funktioner som påverkas av antalet användare eller liknande [1] [2] .

En faktor som påverkar utvecklaren är även den ständiga utvecklingen av mjukvaruutveckling. Att hålla sig uppdaterad på de nyaste metoderna och koncepten krävs mycket tid och lärande. Detta behöver oftast göras på fritiden och inget de företag man år anställd av ger ut tid till utan att vara en utvecklare på jobbet innebär även att vara det på fritiden. En utvecklare som inte följt med utvecklingen de senaste fem åren kommer få det svårt att hitta en anställning och antagligen har problem att lösa nya komplexare problem. Att använda sig av gamla metoder förlänger även oftast tiden då det finns en chans att något nytt mer effektivt sätt har utvecklats [2].

2. Begrepp inom mjukvaruutveckling

Det finns flertalet begrepp inom de olika faserna för mjukvaruutveckling men här kommer några övergripande begrepp.
Analys – Analys är den första delen när man utvecklar en applikation. I denna del så samla man in information från de som ska använda sig av applikationen samt kunden för att få fram en kravlista eller *user-stories* för vad som ska ingå och hur den kan se ut.
Design – I detta steg så börjar planeringen av applikationen där man kollar på hur den ska se ut och vilka funktioner som ska implementeras. En bra start är att rita upp ett första utseende då man kan få en första förståelse av hur den färdiga applikationen kan se ut. Att notera här är att saker efter detta antagligen kommer förändras men gör det möjligt att gå vidare till nästa steg.
Implementering – Här kommer den roliga delen av utvecklingen, själva programmeringen när alla metoder ska implementeras och ett utseende ska formas.
Testning – Här testas koden vilket kan göras på flera sätt både genom att skriva tester samt att flytta runt applikationen och leta efter eventuella buggar som kan dyka upp.
Deployement och underhåll – I detta steg färdigställs applikationen och skickas ut till kund där den installeras och sätts igång. Härifrån är det även viktigt att man håller koll på applikationen och fixar eventuella buggar som uppstår. Så detta steg har inget slut utan oftast så är det något man får göra så länge den används av kund.

Det finns även flera olika roller inom mjukvaruutveckling men här kommer några som används vid utveckling av en applikation. Värt att notera är att dessa roller sällan är en person per roll utan samma person kan ha flera roller inom ett projekt.
Programmerare – Sköter själva programmeringen och utvecklingen av projektet. Denna kan även delas in i front-end utvecklare, back-end utvecklare och så vidare.
Testare – Testar applikationen och letar efter eventuella buggar.
Projektledare/scrum master – Beroende på metod som används vid utveckling kallas dessa olika och arbetar olika men i grund och botten är det båda som håller koll så att projektet rör sig framåt. Har även oftast mest kontakt med kunder eller liknande och för informationen vidare till utvecklarna.

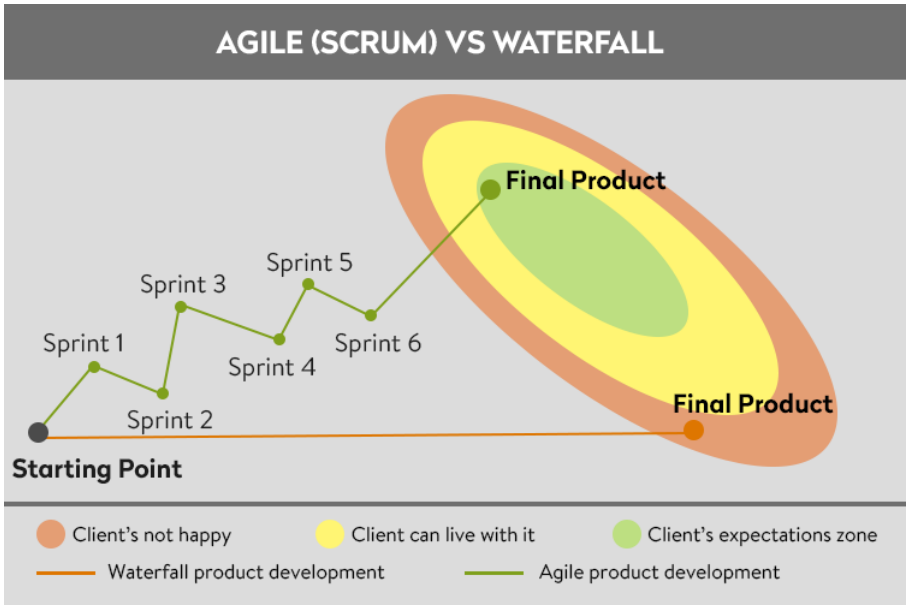
En annan del av utvecklingsprocessen är intressenter som innefattar alla som intresserar sig av applikationen. De roller som nämnts tidigare men även några fler,
Kunder – De som har köpt en applikation är kunden som självklart intresserar sig i den och är även de som kommer använda den.
Företagsledning – I detta begrepp innefattas bland annat chefer, HR arbetare och andra anställda på företag. Dessa individer påverkas av applikationens resultat och intresserar sig därav av applikationens framgång.

Tidigare i texten nämndes en form av metod som kan användas vid utvecklingen men det finns fler som kan användas.
Vattenfall – Man jobbar igenom projektet från start till slut genom att dela upp det i delar och slutför varje del innan man går vidare.
Agil utveckling – Denna innefattar till exempel scrum och går ut på att man under projektets arbetsgång kontinuerligt utvärderar applikationen. Man jobbar i olika cyklar och arbetar med dessa iterativt igenom projektets gång.

3. Agila metoders principer

Agila metoder grundar sig i att man ska få kontinuerlig feedback på projektet under utvecklingen gång. Man har kundmöten med jämna mellanrum där applikationen diskuteras och man kan då se om man arbetar åt rätt håll. Under dessa sprinter ska man få effektiv feedback på det man levererar. För att det ska vara möjligt behövs att man levererar något som kunden kan ha en åsikt om, som till exempel en funktion att man kan boka biljetter då det är det som kunden vill se. Att komma till ett kundmöte och inte kunna visa något utan bara säga att man har gjort att saker funkar oavsett hur lång tid det än tagit är det något kunden inte kan ge feedback på. Att arbeta agilt tar även bort rollen projektledare i den mån av att någon bestämmer över projektet och man jobbar mer som ett team. Även om det finns en scrum master är det inte samma sak utan den finns mer för att dra projektet framåt. Detta är en stor skillnad emot andra metoder sär man har en tydlig styrning och en person som sköter kundkontakt detta leder till ett mer inkluderande projekt men kan vara en stor förändring för många företag.

Begreppet scrum är en agil metod som då jobbar på detta sätt och används för att i stället för de traditionella som får ett mål och jobba mot men man vet inte om man kommer nå de kunden vill ha. Man får oftast något liknande men scrum gör det möjligt att få kontinuerlig feedback vilket gör att man kommer nå de som kunden vill då man kan bli rättad om man arbetar åt fel håll. Att hålla dagliga stand-up möten, korta möten där man berättar vad man har gjort, vad man ska göra och eventuella problem gör att medlemmar kan få sina problem fixade i stället för att sitta fast om man har problem. Detta kan öka tiden för utvecklingen men kommer göra den färdiga produkten mer likt det kunden vill ha. Det finns dock inga rätt eller fel val av metod utan beroende på projektets natur kan den bästa metoden variera.



[3]

4. Metoder i projekt

Metoden som valts till projektet var scrum. Då det finns många sätt att utveckla denna applikation och utseende kan variera mycket så är kontinuerlig feedback effektivt för att nå en applikation som kunden vill ha. Då utvecklarna i projektet inte har någon speciell kunskap om någon del av utvecklingen så funkar det ännu bättre då man mer samarbetar och lär sig flera delar av utvecklingen.
Att jämföra med vattenfallsmetoden skulle innebära att projektet kunde slutat långt ifrån vad kunden ville ha samt att utan en bra projektledare hade det kunnat bli svårt att flytta projektet framåt om det hade fastnat. Då projektets utvecklare inte heller jobbat med liknande applikationer innan kan det vara svårt att veta hur man gör ett bra användargränssnitt till exempel. Det kan då vara bra att få kundens feedback om vad som är bra och dåligt ur deras synpunkt. Då en av de största svårigheterna och anledningar till att företag inte implementerar scrum i sin utvecklingsprocess är att det blir en stor förändring för utvecklarna men i detta fall så visste utvecklarna inget annat och därmed är det inte fel att använda. Det kommer i slutändan ge en mer korrekt applikation ur kundens synpunkt samt ge utvecklarna en bra insikt i hur scrum fungerar och om det är en metod som de tycker är bra. Då utvecklarna sedan tidigare inte jobbat med scrum men har indirekt jobbat med vattenfallsmetoden då det kan appliceras på andra saker man gjort är det inte fel att man får testa på hur scrum känns.

5. Effektiva grupper

För att utveckla en effektiv grupp som gemensamt arbetar mot ett mål krävs det att man från början skapar en öppen grupp i den mån av att man ska kunna fråga varandra och hjälpas åt. Skulle man misslyckas med detta så försvinner innebörden av att arbeta i grupp och den kommer definitivt inte att arbeta effektivt. Som individ krävs det att man visar öppenhet och flexibilitet för metoder som man själv inte kanske tycker ör bäst men att man litar på de andra medlemmarna. Att ha möten och ses för att diskutera och uppdatera om hur det går och eventuella problem kommer göra gruppen effektivare över tid. När en ny grupp bildas får medlemmarna omedvetet olika roller som skapas naturligt och att man då axlar den rollen krävs för att den ska funka. Dessa roller bildas oftast i hur personen är och vad den är bra på och att en grupp arbetar med medlemmarnas styrkor är ett bra sätt att skapa en effektiv grupp.
Hur medlemmarna agerar och att det vill driva projektet framåt är extremt viktigt då det är det som ska göra det. Utan engagemang hos utvecklarna blir applikationen aldrig bra.
Att hantera problem tillsammans är viktigt så att man inte fastnar på något som en annan medlem kan lösa. Det kan även göra det enklare att hitta en lösning om man har någon att diskutera med. Skulle konflikter uppstå är det viktigt att man som andra medlemmar hjälper till att lösa upp det då man i slutändan arbetar mot samma mål. I detta fall kan det vara bra att ha någon som inofficiellt axlar rollen som ledare och tar och styr upp det om konflikter uppstå. Det finns även personer som i sin natur ör mer tillbakadragen men kanske sitter på bra information och det krävas då att de mer utåt riktade medlemmarna försöker inkludera dessa i diskussioner.

6. Mjukvaruutvecklare som individ

Projektet har gett en bra inblick i hur det är att arbeta som mjukvaruutvecklare och gör det möjligt att reflektera på hur jag har uppfattat yrkesrollen.

6.1 Vad innebär mjukvaruutvecklare

Att vara mjukvaruutvecklare innebär inte bara att sitta och koda från morgon till kväll utan det har krävts många timmar planering och diskussioner. Att kunna samarbeta med andra personer är extremt viktigt då det är som att lägga ett stort pussel att utveckla en applikation där allt ska passa ihop. Man måste även kunna ta kritik och anpassa sig efter det när det kommer samt att låta alla i gruppen ha en åsikt om design idéer. Att lite på varandra och fördelat arbetet är viktigt så man inte tar på sig för mycket arbete men det krävs även så att alla i gruppen vill nå en bra applikation. Det kan vara frustrerande om man anser att man har gjort mycket men andra i gruppen gjort mindre och det kommer alltid att vara så när man arbetar i grupp. I stället för att i de fallet sura ihop så är det bättre att hjälpa de att komma i gång och pusha på så projektet kommer framåt. När man senare kommer ut i arbetslivet tror jag det är viktigt att man samlar på sig erfarenhet av andra utvecklare men även att man vågar uttrycka sig om man anser att något kanske år lite gammaldags. Om en erfaren utvecklare säger att en metod ska användas men du vet att det finns en bättre metod så ska man våga säga detta men att man då även sedan kan backa om det inte anses vara rätt. I ett senare skede är det även viktigt att man läser på och lär sig nya saker och vågar lyssna på yngre och andra utvecklare så man hela tiden fortsätter utvecklas som mjukvaruutvecklare.

6.2 Arbetsuppgifter

Arbetsuppgifterna kan variera och man kan jobba med flera delar av applikationen men är även viktigt att man förstår sig på alla delar annars blir det svårt att veta hur man ska agera. Till exempel kan den som arbetar med användargränssnittet inte förstå något om databasen för då är det omöjligt att veta vad man ska ha med utan en ständig kommunikation är bra.

6.3 Lärdomar från gästföreläsningar

Av gästföreläsningarna tar jag med mig mestadels att företag arbetar på olika sätt men att man numera arbetar mycket med scrum och teamarbete. För att kunna arbeta som utvecklare måste man kunna ha diskussioner med kollegor och inte bara borra ner huvudet och koda hela tiden även om det stundtals kan vara fallet. Kommunikationen med kund är något som också pratas mycket om då man annars inte vet vad som eftersöks.

6.4 Kopplingar mellan projektarbetet och yrket

Det jag gör i projektet är väldigt kopplat till yrket då man i en mindre utsträckning gör det som yrket innefattar. Just kommunikation med kund blir lite annorlunda men just att arbeta tillsammans är verkligen något som använts i projektet.

6.5 Min lämplighet som mjukvaruutvecklare

Enligt min åsikt så skulle jag passa som utvecklare då jag har en bra förmåga att samarbeta med andra och vara flexibel men även att jag kan fokusera på ren kodning och fokusera på det. Att just ha den kombinationen som kunskap tror jag är viktigt för

utvecklare nuförtiden. Enligt mig är jag någon som gillar att styra och ställa lite men samtidigt kan jag ta ett steg tillbaka om jag anser att de är rätt i det fallet. Jag hamnar sällan i konflikter med andra utan är bra på att i stället komma fram till en gemensam lösning än att ha en pågående konflikt över något som antagligen är rätt ointressant.

7. Litteraturförteckning

[1] Ju, ”Why Software are So Complex,” *Medium,* 31 December 2022.

[2] ”geeksforgeeks,” 5 December 2023. [Online]. Available: https://www.geeksforgeeks.org/7-reasons-why-software-development-is-so-hard/. [Använd 12 Mars 2024].

[3] K. Pogrebnoy och O. Yatskevich, ”Agile or Waterfall: Choose the Right Approach to Your Software Project Management,” *codetiburon,* 20 Augusti 2020.