UPPSALA UNIVERSITET CAMPUS GOTLAND
Institutionen för informatik och media
Utbildningsprogram: Kandidatprogram i Systemvetenskap (inriktning Programvaruteknik)
Kursansvarig lärare: Millan Lundgren

Datum: [2022-02-07]

PM inom
Systemutvecklingsmetoder, 7,5 hp

Agila Modeller

Principer lärande

Minhui Zhong

1 Inledning

Agila modeller är en viktig del för systemutveckling processer, det stödjer iterativ och ökar utveckling av mjukvara. Nuvarande har agila modeller blivit populär i samhället, vilket gör att flera och flera människor har stora uppmärksamhet på sådana modeller. Det är viktigt att diskutera och att lära känna av de antagande och principerna som finns i agila modeller, eftersom det ökar våra förståelser om hela agila processer. Genom att ta reda på problem och svårigheter som finns i modeller, kan modeller bättre utvecklas in i en systemutveckling process. I detta PM presenterar jag kort vilka principer som finns in i agila modeller. Vidare beskriver jag även att olika principer kan ge upphov till en distinkt antaganden. Jag beskriver även varför jag anser att vissa antaganden ser annorlunda ut idag. I slutet av PM anger jag tre viktigaste principer som jag själv anser att det är enormt viktigt för en systemutveckling process.

1.1 Syfte

Syften med detta skrivande om agila modeller är att fokusera på de antaganden som finns in i olika principer, att lära och diskutera om de principerna som finns i nutids agila modeller. Syften även med hur antaganden relaterar till principer i en agila process. I och med syften på att människor kan förstå och utveckla modeller bättre. Ytterligare i syften är att inspirera och uppmuntra flera människor att delta in i diskussion om agila processer.

1.2 Metod

Det finns ingen intervju och observation i mitt PM skrivande, utan jag kommer ha en litteraturgenomgång för att besvara alla syften som finns ovanstående. Fördelen av detta att jag själv kan ha en grund förståelse om Agila modeller. Genom att referera förklarningar om principerna som finns i vetenskapligt artikeln, gör mitt uttryck och min tanke starkare. Detta gör även att min text mer trovärdigt, så att läsare kan lita på vad jag har skrivit. Det underlättar att läsare kan förstå vad jag har presenterat. Sådan metod kan klara sig bra med tidsbegränsade. Däremot metoden saknar om tydliga faktum om dagliga erfarenheter och feedback. Det finns begränsar som metoden är inte lika flexibelt som intervju och observation.

2 Resultat

Turk et al (2005, s 63) utgår i sin artikel från den som kallade ”Assumptions Underlying Agile Software – Development Processes”, presenterar olika antagande och principer in i en agila process. På början av artikeln har Turk et al (2005, s 64) presenterat att ” Extreme Programming” (XP) är en lätt och effektiv utvecklingsmetod för att uppnå olika aktuella behov. Enligt Turk et al (2005, s 64) har XP fem grundläggande principerna såsom återkoppling, enkelhet, inkrementella förändringar, omfamning av förändring och kvalitetsarbete. Bland annat Turk et al (2005, s 64) skriver att par programmering är en av de mer välkända XP-metoderna, där två programmerare arbetar tillsammans för att utveckla en enda kod. En programmerare kodar direkt medan den annan observerar, på så sätt att jobba som en granskare och att ge omedelbara feedback. Detta tillvägagångsätt ökar högre produktivitet. Turk et al (2005, s 64) beskriver även att antagande om olika människor ser problem ur olika perspektiv och detta gör ett kombinerat tillvägångassätt för problem är mer effektivt än individuellt tillämpade tillvägångassätt. Turk et al (2005, s 65) nämner att XP-praxis bygger på antagandet att korrigering av krav och design fel senare kostar inte mer om de upptäcktes och togs bort tidigare. Detta antagande gör utvecklare att fokusera mindre på grundlig analys och design i de tidiga faserna, i stället att göra förbättringar under hela projektets gång genom att faktorisera om koden.

Turk et al (2005, s 66) påstår att agile alliance är definition av värderingarna och målen för agil mjukvaruutveckling. Enligt Turk et al (2005, s 68) är principen ” Our highest priority is to satisfy the customer through early and continuous delivery of valuable software” att mjukvara är utvecklad för att utföra tjänster som tillför värde för användare vid leveranstillfället. Då ser principerna fyra ”Deliver working software frequently” och fem ”Working software is the primary measure of progress” som konsekvenser av den första principen. De tre principerna har fokus på att utvecklare måste ha i åtanke att kundernas behov utvecklas. Däremot menar Turk et al (2005, s 69) att det finns synlighetsantagandet och iterationsantagandet in i de tre principerna. Synlighet är att projekts synlighet kan uppnås enbart genom leverans av arbetsbod och iteration handlar om ett projekt kan alltid struktureras i korta upprepningar av fast tid. Turk et al (2005, ss 69–70) visar att principen två ”Business people and developers must work together daily through the projekt” att interaktionen mellan utvecklare och slutanvändare i agila processer handlar om att lösa funktionsrelaterade problem och att bestämma insatsens omfattning. Antaganden till denna princip är att kundinteraktion och teamets kommunikation. Antaganden om kundinteraktion påstår att kundteam är tillgängliga för frekvent interaktion när utvecklare behöver det och antaganden om teamets kommunikation är att utvecklare är placerade i tid och plats så de kan hålla intensiv kommunikation med varandra.

Turk et al (2005, s 70) presenterar om principen åtta ”The most efficient and effective method of conveying information to and within a development team is face to face conversation” har antagnden om ansikte mot ansikte, det som författarna vill presentera att interaktion ansikte mot ansikte är den mest produktiva metoden för att kommunicera med kunder och bland utvecklare. Turk et al (2005, s 71) påpekar även att antaganden om dokumentation handlar om omfattande och konsekvent av dokumentation och mjukvarumodeller är kontraproduktivt, det vill säga att det är mer tillförlitligt att bestämma specifikationer och design utifrån kod än från andra dokument eftersom dokument kanske inte alltid hålls uppdaterade när koden förändras. I och med koden är den mest tillförlitliga beskrivningen av vad ett system gör och hur det konstruerades. Principen tre ”Welcome changing requirements, even late in development” har antagandena om förändrade krav och kostnaden för förändring, detta presenterar Turk et al (2005, s 72) i sin artikel. Tolkningen av förändrade krav är att kraven alltid utvecklas på grund av förändringar av teknik, kundbehov, affärsområden eller till och med förvärv av nya kunder. Antaganden om kostnaden för förändring med andra ord är att kostnaden för förändring ökar inte dramatiskt över tiden.

Enligt Turk et al (2005, ss 72–74) är principen sex ”Build projects around motivated individuals. Givet them the environment and support they need, and trust them to get the job done”, principen sju ”The best architectures, requirements, and designs emerge from self-organizing teams” and principen tolv ”Project teams evaluate their effectiveness at regular intervals and adjust their behavior accordingly” bidrar till antagandena teamets erfarenhet, självutvärdering och självorganisering. Teamets erfarenhet menar att utvecklare har den erfarenhet som behövs för att definiera och anpassa sina processer på lämpligt sätt. Självutvärdering är därmed att teamen kan och vill utvärdera sig själva och de bästa arkitekturerna kraven och design kommer från självorganiserande team. Turk et al (2005, ss 74–75) nämner att principen nio ”Agile processes promote sustainable development” och tio ”Continous attention to technical excellence and good design enhances agility” har antagandena om kvalitetssäkring. Kvalitetssäkring handlar om utvärdering av mjukvaruartefakter kan begränsas till frekventa informella intervjuer, recensioner och kodtestning. Sist men inte minst principen elva ”Simplicity is essential” har antaganden om det applikationsspecifika utveckling och kontinuerlig om design. Det applikationsspecifika utveckling handlar om att återanvändbarhet och allmänhet inte bör vara mål för applikationsspecifika mjukvaruutveckling och kontinuerlig om design är att system kan fortlöpande om design och fortfarande behålla sin strukturella och konceptuella integritet.

2.1 Kan det betyda att vissa antaganden skulle se annorlunda ut idag? Vilken/Vilka i så fall och varför anser du det?

Som jag nämnde tidigt att tekniken alltid förändrar sig över tiden och detta gör att vissa antaganden som fanns i 2005 skulle se litet annorlunda ut än idag. Själv anser jag att antagandet om ansikte mot ansikte, förändrade krav och

teamets erfarenhet skulle se litet annorlunda ut idag. Anledningen till detta är att användares behov har förändrats och systemet har utvecklats snabbare än tidigare. Nuvarande har vi stora utmaningar än 2005 därför vi behöver många avancerade tekniker som kan underlätta vårs dagliga liv. Idag är det inte ovanligt att ett utvecklingsteam på ett enda projekt ska spridda ut i stora geografiska områden som involverar många tidzoner, då blir ansikte mot ansikte en svårighet. Det är inte lika effektivare tidigare än nu att umgås med ansikte mot ansikte eftersom det finns flera hinder som kan förhindra människor att ta sig till träffas i dåtid. Nu har människors behov utvecklats och det finns många system kan hjälpa till att människor tar sig fram till ett framgångsrikt resultat. Människor inte längre tar mycket tid att träffas ansikte mot ansikte utan med hjälp av något system kan människor lätt att träffas med varandra. På så sätt kan människor spara mycket tid via dagliga kommunikationer. Olika utmaningar dyker upp via olika tillfället därför det är enormt viktigt att teamet har tillräckligt erfarenheter att klara dessa utmaningar. Detta har stora behov för utvecklare kan anpassa sina processer på lämpligt sätt.

2.2 Välj ut 3 principer som du anser vara viktigast, förklara varför du anser dem vara viktigast?

Den första principen som jag anser det är viktigt är att principen ett eftersom denna princip har stora fokus på kundernas behov, i och med det är en grund för principerna fyra och fem. Genom att snabbt får feedback och återkoppling om projektets framsteg, kan systemet bättre utvecklas enligt behov. Principen två anser jag också att det är enormt viktigt för agila processer. Det beskriver tydligt att interaktion mellan utvecklare och slutanvändare i agila processer handlar om att lösa funktionsrelaterade problem och att bestämma omfattning av insatsen. Denna princip sker inte bara i början av ett projekt utan hela processen. Goda interaktion mellan användare och utvecklare ökar bättre förståelse för varandras problem och behov, vilket minskar problem med mänskligt samarbete, därmed ökar chansen till ett lyckat projektresultat. Slutligen jag anser även att principen nio är viktigt eftersom nya utmaningar dyker upp när system har utvecklats. Dessa utmaningar kan förhindra system på olika sätt vilket gör att system har svårighet att förbättra. Då kan en långvarig och hållbar lösning bidrar till ett framgångsrikt resultat.

3 Diskussion/Reflektioner

Agila processer spelar en viktig roll för systemutveckling och det är en av de bästa metod kan förbättra system och utveckling på ett hållbart sätt. En samband utifrån resultatet är att principerna och antagandena som finns in i Agila modeller är nyttigt för systemutveckling och de bidrar till att en djupare förståelse om hela agila processen. Det är inte svårt att upptäcka kunders behov är ett centralt mål för agila modeller. Samtidigt är goda samarbete och kommunikation viktigt för systemutveckling. Olika människor ser problem ur olika perspektiv, precis som författarna lyfter att en kombinera lösning med olika människor för svårigheter är mer effektivt än individuellts lösning. Ensamhet har svårt att lyckas in i en systemutveckling utan det kan leda till flera stora problem. Det är inte räcker att utvecklare samarbetar med utvecklare utan också att utvecklare samarbetar med slutanvändare i agila processer. Oavsett människor upptäckte fel tidigt eller inte, kostar lika att korrigera fel enligt behov. Då har utvecklare mindre fokus på analys och design i tidiga faserna. Detta tyda inte på att XP påverkar systemutveckling på ett negativt sätt.

In i agila principen har vi flera antagandet påverkar systemutveckling enormt mycket. Såsom antagandet om iteration att uppdela ett komplext projekt åt små bitar och att upprepa samma struktur åt en fast tid. Detta kan på stora viss att minska tid på systemutveckling process. Kundinteraktion är en viktig princip av agila modeller eftersom systemutveckling är baserad på kunders behov, det vill säga att återkoppling och feedback av kunder kan system utvecklas bättre. Utifrån inlärning av principerna och antagandena har jag en tanke hur dessa principer och antagande kan användas för systemutveckling? Denna frågeställning belyser enligt mig en viktig del systemutveckling, nämligen de har stora betydelser för systemutveckling. Min svaret till detta är att det inte är ett hållbart sätt att välja en enda princip och antagande eftersom det alltid finns begränsa för att utveckla en avancera system. Kombinerat med flera principer och antagande kan främja systemutveckling på ett hållbart sätt.

4 Referenslista

Turk, D., France, R. & Rumpe, B. (2005). Assumptions Underlyning Agile Software-Development Processes. *Journal of Database Management,* ss 62–87.