Linnéuniversitetet
Kalmar Växjö

Reflektionsrapport

Projekt och yrkesroll

Författare: Samuel Berg
Termin: VT 24
Kurskod: 1DT908

Reflektionsrapport över projekt

Faktorer som gör mjukvarudesign och utveckling komplext

Att hantera mjukvarudesign och mjukvaruutveckling är komplext av flera skäl. För det första kräver kravhanteringen både noggrannhet och förståelse för användarens behov, vilket kan vara svårt att uppnå från början på grund av de många olika krav som ofta uppstår. Tekniskt sett kan omfattningen av ett projekt öka den tekniska komplexiteten avsevärt, då det finns krav på att integrera olika plattformar och system samt uppfylla säkerhets- och prestandakrav. Dessutom kan ändrade krav och miljöförändringar under utvecklingsprocessen kräva snabba justeringar i design och kod för att möta nya behov eller tekniska begränsningar. Att navigera genom dessa förändringar kräver både flexibilitet och förmåga att anpassa sig snabbt. Mänskliga faktorer spelar också en avgörande roll eftersom olika teammedlemmar har olika bakgrund och expertisområden. Det är därför nödvändigt med en effektiv kommunikation och samordning för att nå framgång. Slutligen måste mjukvaran också vara skalbar för att över tid kunna underhållas för att möta nya krav och ökad användarbelastning. Alla dessa faktorer kräver en genomtänkt planering och en solid arkitektur från början. För att kunna navigera genom ovanstående komplexiteter krävs alltså både teknisk kompetens och en välorganiserad process.

Grundläggande begrepp inom mjukvaruutveckling

I mjukvaruutveckling är det viktigt att förstå de grundläggande begreppen som styr processen. Faserna i mjukvaruutveckling är de olika steg som krävs för att skapa och leverera programvara. Det börjar med kravhantering där behovet av systemet definieras genom att samla in och analysera krav från användare och intressenter. Därefter kommer designfasen där en övergripande arkitektur formuleras och detaljerade planer utarbetas för implementationen. Själva bearbetningen av kod sker i implementationsfasen, följt av testning för att säkerställa att mjukvaran fungerar enligt förväntningarna. Distributionsfasen handlar om att installera och leverera mjukvaran till användare/kunder. Slutligen, underhållsfasen involverar att lösa buggar, göra förbättringar och uppdateringar baserade på feedback och ändrade krav från användaren.

I utvecklingsprocessen spelar olika roller en viktig roll. Utvecklare är de som kodar och implementerar mjukvaran, medan projektledaren tar ansvar för att organisera och övervaka projektet. Arkitekten designar den övergripande strukturen för systemet, medan testare ansvarar för att hitta och rapportera buggar. Designerns för användarupplevelsen (UX) fokuserar på att skapa användarvänliga gränssnitt och upplevelser, medan kravanalytikern samlar in och analyserar krav från användare och intressenter. Kundsupport är en annan viktig roll som tillhandahåller teknisk support och löser användarproblem efter distributionen.

Intressenter i mjukvaruutvecklingen inkluderar användare, kunder, interna intressenter som projektledare, utvecklare och testare, samt externa leverantörer som kan tillhandahålla resurser eller verktyg för utvecklingsprocessen.

Det finns olika metoder inom mjukvaruutveckling som kan anpassas beroende på projektets behov. Vattenfallsmodellen är en linjär metod där varje fas slutförs innan nästa påbörjas. Agil utveckling däremot är en iterativ metod som fokuserar på snabba leveranser och kontinuerlig feedback. Scrum och Kanban är specifika agila ramverk som används för att organisera och hantera arbetsflödet. DevOps är en metod som integrerar utveckling och drift för att förbättra leveranshastigheten och kvaliteten på mjukvaran. Dessa grundläggande begrepp och metoder inom mjukvaruutveckling utgör grunden för att bygga och leverera effektiv och högkvalitativ programvara. Utvecklare kan sedan utforska och anpassa ytterligare tekniker och verktyg baserat på specifika behov och krav inom projektet.

Principer för agila metoder för mjukvaruutveckling

Då agila metoder används så ligger fokus på att bryta ned projektet i snabba iterationer, där varje steg resulterar i en användbar version av produkten som kan testas vilket ger feedback på hur varje steg fungerar. Genom detta iterativa tillvägagångssätt möjliggörs en kontinuerlig förbättring och anpassning till kundens behov och prioriteringar under hela utvecklingsprocessen. Samarbete mellan korsfunktionella team är en hörnsten i de agila metoderna. Genom att integrera utvecklare, testare och produktägare i samma team möjliggörs en effektiv delning av kunskap och kompetens. Detta främjar inte bara bättre beslut utan ger också snabbare lösningar på problem som uppstår under utvecklingen. Agila metoder utmärker sig genom sin flexibilitet och anpassningsförmåga. Istället för att hålla sig strikt till förutbestämda planer kan teamet snabbt justera sina strategier och prioriteringar baserat på ny information och/eller förändrade förutsättningar. Kontinuerlig förbättring och reflektion är ytterligare ett viktigt inslag i agila metoder. Genom att regelbundet utvärdera och justera arbetsmetoder och processer kan teamet optimera sin produktivitet och effektivitet över tid. Detta fokus på kontinuerlig utveckling skiljer sig markant från mer traditionella metoder, som ofta har en mer linjär och förutsägbar process. [1]

Sammanfattningsvis, agila metoder för mjukvaruutveckling prioriterar kundnöjdhet, samarbete, flexibilitet och kontinuerlig förbättring. Agila metoders iterativa och flexibla natur gör att de på ett enkelt sätt kan möta kraven i en snabbt föränderlig miljö, där responsivitet och leverans av värde är av yttersta vikt. Hänvisar till ett flertal bilder från min ena källa [1, s20], [1, s25-29], [1, s37], [1, s40-41], [1, s47-48] som visar bra och tydligt hur de olika typer av metoder fungerar visuellt.

Val av mjukvaruutvecklingsmetoder för projekt

I projektet använde vi oss av en typ av metodik som liknar SCRUM, speciellt då det gäller utvecklingens framsteg. Angående versionshanteringen så delade vi först upp det inom de tre grunderna i projektet, databas, mjukvara och användargränssnitt. Inom dessa tre grunder så skapade varje person sin egen bransch där det arbetade med de metoder de tog sig an för att lösa användarberättelserna för den aktuella sprinten.

Då det gäller val av metodik för projektet så förstår jag varför man valt en SCRUM-liknande metod. Detta på grund av att den är väl använd inom branscherna som använder agila-metoder [1, s32]. Däremot så ser jag inga problem med att använda

en vattenfalls-liknande metod istället för detta projekt. Detta tror jag personligen hade fungerat lika bra eller bättre på grund av att det varit tydliga instruktioner från början med vad som skulle inkluderas i projektet och så länge det programmerades på ett skalbart sätt så hade det inte varit några problem att adaptera det kunden hade velat ändra.

Hur skapas effektiva grupper

Skapandet av effektiva grupper involverar en mängd olika faktorer, inklusive individernas egenskaper, deras beteenden och hur väl de interagerar med varandra. För att reflektera över detta och förstå vad som krävs av gruppmedlemmar samt hur problem och konflikter kan hanteras, är det viktigt att undersöka några centrala punkter.

För det första är tydliga mål och syften avgörande för effektiva grupper. Alla medlemmar måste förstå och sträva mot dessa mål för att nå framgång. Det är också viktigt att alla medlemmar har klara roller och ansvarsområden för att undvika förvirring och konflikter.

Kommunikation spelar en avgörande roll i att skapa och upprätthålla en effektiv grupp. En öppen och lyhörd kommunikation gör det möjligt för alla att dela sina tankar och idéer samt att ta emot feedback från andra medlemmar. Respekt och tillit är också grundläggande för en positiv gruppdynamik och detta underlättar samarbete och konfliktlösning.

Flexibilitet är en annan viktig egenskap hos effektiva grupper. Alla gruppmedlemmar måste kunna anpassa sig till förändrade omständigheter och vara villiga att ompröva sina strategier för att uppnå sina mål på bästa sätt.

När det gäller hantering av problem och konflikter är det avgörande att identifiera och hantera dem så tidigt som möjligt för att undvika att de blir för stora. Trots eventuella konflikter är det viktigt att gruppens övergripande mål och syfte förblir i fokus för att säkerställa en konstruktiv diskussion och ett effektivt resultat. Genom att skapa en miljö som främjar öppen kommunikation, respekt och samarbete kan grupper övervinna hinder och framgångsrikt arbeta mot sina gemensamma mål.

Reflektionsrapport över yrkesroll

Yrkesrollen mjukvaruutvecklare innebär

Rollen som mjukvaruutvecklare är mångfacetterad, det krävs en kombination av kunskap och ansvarskänsla. Rollen innebär att man är ansvarig för att skapa mjukvarusystem och applikationer som fungerar effektivt. Det krävs då att utvecklaren behärskar många olika områden, från att analysera användarbehov till att designa lösningar och skriva kod i olika programmeringsspråk. Testning är också en viktig del av processen där det säkerställs att koden är korrekt och uppfyller kraven. Efter lanseringen av en produkt fortsätter arbetet med underhåll, buggfixar och optimering av prestanda.

Kommunikation och samarbetsförmåga är nyckeln då mjukvaruutvecklare ofta arbetar i team och behöver hålla kontakt med olika intressenter. Att hålla sig uppdaterad med den senaste tekniken och delta i forskning och lärande är avgörande för att fortsätta utvecklas inom yrket. Dessutom spelar dokumentation en viktig roll för att underlätta för framtida utvecklare att förstå och underhålla koden.

Grundläggande arbetsuppgifter för en mjukvaruutvecklare

I början av karriären inom mjukvaruutveckling får man som mjukvaruutvecklare ofta arbeta tillsammans med handledare och assistera i enklare uppgifter. På sikt, efter ungefär ett år, förväntas mjukvaruutvecklare förbättra sina färdigheter och ta på sig större och större ansvar, vilket kan inkludera att behärska nya tekniker och metoder för att hantera mer komplexa projekt och utmaningar.

Inledningsvis kan det handla om att skriva kod för mindre funktioner eller moduler under handledning av mer erfarna utvecklare. Med tiden kommer man att ta på sig större ansvar och börja arbeta mer självständigt inom ett specifikt område av systemet.

När det gäller felsökning och debugging, kommer nybörjare att assistera med att identifiera och fixa mindre buggar under övervakning av erfarna utvecklare. Efter ungefär ett års erfarenhet förväntas man kunna självständigt felsöka och lösa vanliga problem och buggar, och även hjälpa andra teammedlemmar i processen.

Vid enhets- och integrationsprovning, förväntas nybörjare skriva enhetstester för små kodavsnitt och delta i integrationstestning under övervakning. Senare kommer man att ansvara för att skapa och genomföra enhetstester för sina egna kod bidrag och även delta i integrations- och systemtestning.

När det gäller kravanalys och design, förväntas nybörjare assistera med att samla in krav och delta i diskussioner om systemdesign under övervakning. Efter att man skaffat sig tillräcklig erfarenhet bör man kunna bidra till att definiera krav och utforma mindre delar av systemet med handledning.

Gällande dokumentation, kommer nybörjare att assistera med att skriva och uppdatera dokumentation för kod och system. Då man fått tillräcklig erfarenhet förväntas man kunna producera och underhålla tydlig och komplett dokumentation för sina egna arbetsområden.

När det kommer till kommunikation och samarbete, förväntas nybörjare delta i möten och code reviews samt kommunicera med teammedlemmar och ge statusuppdateringar. Efter ett tag bör man effektivt kunna samarbeta med andra utvecklare, designers och intressenter, delta i tekniska diskussioner och ta ansvar för mindre delar av projektet.

Över tid förväntas mjukvaruutvecklare förbättra sina färdigheter, ta på sig större ansvar och vara mer självständiga inom sina arbetsuppgifter. Det kan också innebära att lära sig nya tekniker, verktyg och metoder för att hantera mer komplexa projekt och utmaningar.

Lärdomar från gästföreläsningar om yrkesroll

Gästföreläsningarna har givit en blick in i hur ett möjligt arbetsliv inom detta yrke hade varit samt givit exempel på de metoder som används inom just det företaget, detta inkluderar hur det hade gått till.

Projektets relation till yrkesroll

I det aktuella projektet har jag identifierat flera element som kan kopplas till arbetsuppgifterna för en mjukvaruutvecklare. För det första så återfinns kravanalysen i användarberättelserna. Design inkluderas genom att vi skulle göra en applikation med ett gränssnitt för hur man interagerar med applikationens funktioner. Programmering ingår genom hela projektets gång då detta är något som behövdes för att kunna presentera funktionaliteter samt för att ha något att testa. Tester har genomförts kontinuerligt under hela projektet för att säkerställa att det fungerar som planerat. Testningen genomfördes genom manuell testning och inte genom unit test. Debugging genomfördes när fel upptäcktes under testningen. Underhållsaspekten var däremot mindre relevant eftersom projektet var avgränsat till kursens ramar och inte hade några framtida krav eller utvecklingsmöjligheter. Samarbete och kommunikation var avgörande i projektet då det genomfördes i grupp där alla medlemmarna inte hade tidigare erfarenhet av att arbeta tillsammans. Det krävdes tydlig och effektiv kommunikation för att säkerställa att alla var på samma sida gällande målen och visionerna för projektet. Lärande var en kontinuerlig process i projektet då nya teknologier och verktyg skulle användas. Detta krävde att gruppmedlemmarna snabbt fick utveckla nya färdigheter för att lösa problem och utmaningar som uppstod längs vägen. Dokumentationsbehovet var minimalt och begränsades till en kort sammanfattning av hur applikationen skulle köras. Även kända problem och buggar skulle dokumenteras.

Är jag lämplig till att bli mjukvaruutvecklare

Med erfarenhet inom mjukvaruutveckling och förmågan att snabbt lära mig nya teknologier och koncept, anser jag mig vara väl rustad för rollen som mjukvaruutvecklare. Jag har arbetat med en mängd olika projekt som har gett mig djupgående kunskap inom området. Min förmåga att lösa komplexa problem är en av mina starka sidor. Att tänka kreativt och innovativt är en del av min arbetsmetodik, och

jag är alltid beredd att utforska nya idéer och tekniker för att förbättra produkter och processer. Min förmåga att samarbeta och kommunicera effektivt gör mig till en tillgång i projektgrupper, där jag aktivt bidrar till en positiv och produktiv arbetsmiljö. Jag är engagerad i mitt arbete och strävar alltid efter att leverera högkvalitativ kod i tid och inom budget. Dessutom är jag driven av en passion för mjukvaruutveckling och ser fram emot att bidra till teamets framgång genom min professionalism och ständiga strävan efter personlig och professionell utveckling.

Referenser

Majoriteten av all min information kommer från erfarenhet eller från kontakter inom branschen. Möjligen så kommer en del information från sådant jag hört från gästföreläsningarna/föreläsningar men oklart [2-5].

[1] T. Andersson Gidlund, "Föreläsning 1: Introduktion, projekt och git.". Föreläst på, Mjukvaruutvecklingsprojekt, Linnéuniversitetet, Nov. 2023
[2] Computer Science LNU. "Guest lecture - Bontouch - iOS-, Android and cross platform development," Computer Science LNU, YouTube, Oct. 6, 2021. https://www.youtube.com/watch?v=Ex-aDzkas4A
[3] Computer Science LNU. "1DT308/902 - Interview with Patricia Pettersson H&M," Computer Science LNU, YouTube, Dec. 14, 2021. https://www.youtube.com/watch?v=6-02p9r02-E
[4] Computer Science LNU. "1DT308/902 - Gästföreläsning Yaskawa, ingenjörens roll," Computer Science LNU, YouTube, Dec. 21, 2021. https://www.youtube.com/watch?v=qvVmaHgSyN0
[5] Computer Science LNU. "CS Branschintervjuer," YouTube. [Online]. Tillgänglig: https://www.youtube.com/playlist?list=PL70wNv4dBdJzyGS914nrC8Onzxqjdmpte. [Åtkomst: Mar. 24, 2024].