Reflektion

Android

Att utveckla en android applikation har varit väldigt lärorikt och kul. Särskilt då jag aldrig gjort något liknande innan. Det gick hyfsat enkelt att komma igång då det finns mycket hjälp att hitta för hur man utvecklar android på nätet. Det är även en bra uppgift för en sådan här kurs då det är enkelt att se resultat och det går bra att utveckla mjukvara utan kunskap om android sedan tidigare. Vad som inte gick lika enkelt var att testa applikationen med automatiska tester vilket jag återkommer till senare. Den sammanlagda tiden som gruppen lagt ner på projektet har estimerats vara ca 700 timmar.

En nackdel som vi upplevde under projeket var att det var svårt att utveckla en applikation utan att ha en android att testa på. På de datorer vi hade tillgång till tog emulatorn väldigt lång tid att ladda och fungerade i vissa fall inte alls. Då det dröjde några veckor innan vi fick tillgång till telefoner var det något som bidrog till att det var svårt att komma framåt med utvecklingen i början. Något annat som bidragit till att resultatet inte blivit optimalt är att vi testade på två androider med android sdk 10 vilket var vårt minimum för applikationen. Vår target sdk var 17 och ett problem var därmed att inte någon nivå över 10 testades förrän sista dagen då en nyare android lånades till redovisningen. Att inte ha testat på en nyare android under projektets gång för att se hur applikationen fungerar på en sådan är naturligtvis en nackdel. I övrigt känns det som en förutsättning för denna kurs att åtminstone två androider per grupp finns tillgängliga att testa på för att inte arbetet ska bli onödigt krångligt, alternativt att gruppen har tillgång till datorer som klarar att köra emulatorn.

Scrum

Att använda sig av Scrum var en fördel för att snabbt kunna komma igång med projektet i början. Genom att använda sig av denna metod bryts ett större projekt ner i hanterbara delar som går att börja med direkt. Detta var en extra stor fördel för vår grupp som inte tidigare utvecklat en android applikation med java. Att då kunna börja med en mindre uppgift och få den avklarad snabbt gjorde att det kändes enklare att senare göra mer avancerade delar.

Nackdelen med att använda sig av Scrum har varit att olika personer i gruppen ändå prioriterade olika funktioner och åtgärder beroende av intresse. Vad som då har varit viktigt är att hela tiden försöka se till slutproduktens resultat och vad som skapar värde i den och följa den prioriteringsordning som bestämdes gemensamt av gruppen i Pivotaltracker.

Tekniken känns effektiv då den inte kräver en så stor initial planeringsprocess utan är mer öppen för att komma igång med själva tillverkandet av produkten direkt. Detta kan för vissa projekt kanske vara en nackdel om helt fel prioritering sker i början. Samtidigt så kommer inte ett projekt med Scrum som metod kunna hålla på i flera år utan att det upptäcks att projektet prioriterat fel funktioner då det är en iterativ process.

Tekniken Scrum skulle jag använda vid ett projekt som i hög grad beror av kundens efterfrågan och där denna är relativt volatil. På så sätt kan man genom user stories stämma av ofta med beställaren om det finns förändrade krav, men också visa upp något som går att använda direkt under utvecklingsprocessen, om än med oimplementerade funktioner.

Att använda tekniken vid ett projekt med väldigt hårt dragna riktlinjer vad gäller struktur skulle inte passa lika bra. Då hade det varit bättre med en större initial planering för att säkerställa att kraven som tagits fram för projektet uppfylls inom en viss tidsram. Samtidigt tror jag ändå på metodiken från scrum att testa parallellt med utveckling av mjukvaran för att hela tiden kunna ha bugg fri kod i den utsräckning det är möjligt. Om en bugg har uppstått i ett initialt skede kan den vara svårare att identifiera ju längre projektet pågår. De tester som vi genomförde gjordes ungefär i mitten av projektet, anledningen till att vi kom igång så sent med dem var eftersom övriga projektet drog ut på tiden då vi haft en del problem med att använda git.

Pivotaltracker

Ett viktigt hjälpmedel för att kunna applicera metodiken Scrum för projektet har hjälpmedlet Pivotaltracker varit. Genom att lägga in alla user stories där har gruppen kunnat hålla kolla på vad som görs just för tillfället men också vad som ligger närmast i tiden att börja med. Det har även varit praktiskt att endast kunna lägga till en ny user story vid behov samt ändra ordning på redan befintliga. Det har även bidragit till en struktur i projektet och en möjlighet att kunna se hur gruppens arbete utvecklas samt planering av arbetet.

Nackdelar med Pivotaltracker var att det från början kändes lite omständigt att lägga in alla user stories. Men efter ett tag var det ett väldigt bra verktyg för planering och tidsestimering. Pivotaltracker känns som ett hjälpmedel som inte endast kan vara bra vid mjukvaruutveckling, det hade även kunnat tillämpas i andra projekt. Särskilt i projekt där fler personer än fyra är involverade då det kan vara svårt att kommunicera och ha överblick över hur gruppen ligger till. I Pivotaltracker kan alla gruppmedlemmar enkelt gå in och se vem som arbetar med vad och vad som står på agendan att göra samma vecka. I ett projekt med färre personer än fyra hade jag tyckt att det eventuellt känns lite onödigt med Pivotaltracker då det istället kan vara enklare att kommunicera och planera genom att träffas. Att göra Pivotaltracker för ett projekt med en större omfattning tidsmässigt hade känns för omfattande. Men kanske hade Pivotaltracker kunnat gå att applicera även då om det mer omfattande projektet bröts ner i intervall. Inför varje intervall planeras detta sedan i mer detalj. Däremot känns det inte värdeskapande att planera ett längre projekt i detalj för att sedan ändra allting.

Såhär i efterhand hade det varit bättre att fokusera mindre på att skriva så detaljerade user stories som vi gjorde. Det bidrog till ett onödigt arbete med att lägga in alla i Pivotaltracker. Det hade istället varit bättre att göra dem mer omfattande och istället lägga till flera uppgifter under varje. I början av projektet använde vi oss mindre av hjälpmedlet, främst för

att vi programmerade i väldigt mycket mindre omfattning då. I slutet när vi hade ett betydligt högre tempo var Pivotaltracker ett bra hjälpmedel för struktur och för att få de mest prioriterade uppgifterna gjorda.

Git

Att använda versionshantering vid kodning var helt nytt för oss och det var en viss startsträcka för att förstå hur det fungerade och på vilket sätt som det skulle användas. Det tog därför ett par veckor innan gruppen var helt igång och kunde använde detta verktyg. Fördelen med att använda git har först och främst varit att kunna koda parallellt samt att kunna gå tillbaka i historiken och se gamla commits. Det har även varit fördelaktigt då en person kan utveckla en viss del av applikationen samtidigt som någon annan arbetar med en annan klass. Nackdelen har varit att det ibland varit tidskrävande att slå samman koden genom en merge. Det kan ha berott på att vi pushade kod till Github för sällan, samt gjorde commits för sällan. En annan anledning kan ha varit att vi inte utnyttjade funktionen branches i git där olika delar av ett projekt kan separeras så att den funktion som varje person arbetar med inte måste merges med master vid varje push. Detta har bidragit till en onödigt tidskrävande process vilket vi inte förstod i början. Det har däremot varit väldigt lärorikt att använda git.

Inlärningen var som tidigare sagt något tidskrävande men efter att ha kommit igång med git känns det som ett bra och nödvändigt verktyg för att flera ska kunna utveckla funktioner parallellt. Även om det kunde blivit mer optimalt och mindre tidskrävande om vi hade lyckats använda olika brancher under projektets gång. Vid framtida projekt med utveckling av mjukvara kommer jag använda git då det känns som ett tillförlitligt verktyg. Även i de fall som utveckling sker individuellt hade jag kunnat använda git då det är väldigt praktiskt att kunna gå tillbaka och se vad som tidigare gjorts i githubs historik över commits.

Fördelar och nackdelar med en iterativ process

Då ingen i gruppen tidigare gjort ett projekt inom mjukvaruutveckling hade vi varken någon förkunskap kring andra metodiker för systemutveckling eller någon förutfattad mening kring arbetsättet Scrum. Fördelen som jag ser med Scrum är främst möjligheten att kunna göra förändringar och omprioriteringar i vad som krävs av en produkt under projektets gång. Det känns även som ett mer effektivt sätt att uppnå en mjukvara som fungerar buggfritt än om utvecklingen av hela programvaran sker innan mer omfattande testning. Ur ett perspektiv där en kund har beställt en produkt kan det även vara av vikt att kunna förändra detaljer i ett projekt under tidens gång, särskilt om det handlar om en mer omfattande beställning. Nackdelen med Scrum var mest beroende av en ovana att hela tiden arbeta mot mindre mål. Men också att det var svårt att estimera vilka user stories som egentligen skulle hinnas med inom en viss begränsad tid.

Min arbetsinsats

Med det faktum att aldrig tidigare ha använt git eller programmerat en android applikation så känns det över förväntan att se hur mycket som går att lära sig under endast några veckor. Från att aldrig ha åstadkommit något större på helt egen hand utan hjälp från handledare har ett helt program skapats. Det som jag framförallt har gjort bra är att testa android appen och genom att använda LogCat försöka spåra var fel har uppstått. Jag har även fått insikt i den stora omfattning av hjälp för att utveckla android som finns på nätet. Utöver det har även kommentarer i koden varit något som jag gjort bra. Det faktum att vi använt två olika androider att testa på har också varit bra då vi kunnat se att applikationen fungerade på båda.

Innan vi fått de olika programmen installerade försökte jag sätta mig in i hur vi skulle programmera. Men det var svårt att ta till sig teori om hur man ska koda utan att kunna testa om det fungerar, så jag läste mycket i början utan att ta till mig särskilt mycket. När sedan programmet fanns installerat kändes det genast enklare och det var då lättare att lära sig. I nästa projekt skulle jag se till att försöka lösa de tekniska problemen så snabbt som möjligt. I början var det också enkelt att ge upp för lätt, men mot slutet har tålamodet ökat och jag har insett att det mesta går att lösa genom att söka på internet. Ett annat problem var att vi inte hade någon android att testa på de första veckorna vilket var en utmaning då emulatorn inte ville visa sig på min dator. En annan sak som jag inte är helt nöjd med är att vi inte lyckades med att testa flera aktiviteter i applikationen med hjälp av automatiska tester.

Gruppens arbetsinsats

Arbetet gruppen emellan har fungerat bra under kursens gång, kanske främst för att vi tidigare under våren gjort kandidatarbete tillsammans och därmed har blivit rejält sammansvetsade vid det här laget. På så sätt var det också enkelt att prioritera och planera när vi skulle ha möten och när vi skulle ses för att diskutera olika aspekter. Den inledande planeringen av projektet fungerade även den bra och det var enkelt att komma på en idé. Under projektets gång har vi programmerat i par för att underlätta själva utvecklandet. Det har fungerat bra och vi har bytt person att programmera med ett flertal gånger. Vissa delar av projektet som ansågs enklare har utvecklats individuellt. Vi har även åstadkommit två releaser under projektets gång och efter att vi gjort den första har arbetet fungerat bättre och takten ökade också mycket efter den.

En orsak till att arbetet inte fungerat optimalt var kanske tron att vi skulle få mer handledning och hjälp med själva programmeringen än vad som var fallet. Om vi istället haft inställningen från början att vi skulle behöva googla och ta reda på allt själva hade slutresultatet antagligen blivit ett bättre projekt.

Inför framtida projekt

I ett liknande projekt framöver hade förhoppningsvis git kunnat innebära ett mer strukturerat arbete för varje person genom att arbeta med branches. Det tillämpades inte i detta projekt men hade kunnat innebära en fördel. Det hade nog bland annat fått konsekvensen att det blivit enklare att ha separerade arbetsuppgifter. Genom att separera arbetsuppgifterna kanske också klasserna hade kunnat bli ännu mindre beroende av varandra, vilket är en viktig grundsten i objektorienterad programmering. I ett framtida projekt hade jag också haft med mig mer tålamod att ta reda på saker och mer målmedvetet söka efter dem.