# Post Mortem Report

## Tekniker och metoder i projektet

Att göra ett projekt för att skapa en androidapplikation har varit lärorikt på många olika sätt, främst eftersom samtliga i gruppen är vara vid att arbeta i projekt men på ett helt annat sätt än detta. Nedan tar jag upp och diskuterar bland annat fördelar och nackdelar med de tekniker och metoder vi använt oss av under projektets gång.

### Aneroid

Att arbeta med android var förvånansvärt smidigt, eftersom det finns otroligt mycket information och hjälp att tillgå bara genom att söka på google och androids hemsida. Att skriva kod utifrån en ny plattform fungerade därför bra, och ännu bättre mot slutet, eftersom det tar tid att komma in i och få en djupare förståelse för uppbyggnaden av ett androidprojekt. Något som till en början förvirrade var blandningen av xml- och javakod och hur dessa kopplades till varandra, men förståelse för detta uppnåddes relativt lätt genom att följa ett par tutorials. Det största hindret jag upplevde var egentligen att komma igång med tekniken, alla program och metoder som skulle användas, men när vi väl hade allting på plats så fungerade utvecklingen bra. Hade jag gjort om ett liknande projekt idag tror jag att startperioden blivit avsevärt kortare. När jag också väl kommit till insikten om att det finns ett svar på alla frågor jag har om jag bara söker på internet så gick projektet väldigt mycket smidigare. Jag skulle absolut kunna tänka mig att ha att göra med utveckling av mjukvara till exempelvis android i framtiden. Även om jag kanske inte programmerar själv är jag övertygad om att jag kommer stöta på mjukvaruutveckling i många sammanhang, och en djupare förståelse för bland annat vad som är möjligt och inte är möjligt att göra är därför värdefull.

Det som har fungerat mindre bra med utvecklingen beror främst på de tekniska problem som uppstått. Att endast hade tre datorer att programmera på med fyra personer i gruppen var inte optimalt. Utvecklingen var även trögstartad på grund av att vi inte hade tillgång till någon fungerande android-telefon förrän de sista tre veckorna. Att utveckla en GPS och en kamerafunktion med endast emulatorn till hands fungerade inte. Att utveckla något över huvud taget med emulatorn var svårt eftersom denna är obeskrivligt långsam. Den får även applikationen att framstå som seg och det är svårt att få en uppfattning om applikationen ur ett prestanda- och användarperspektiv. När vi sedan fick tillgång till två mobiler tog utvecklingen fart på riktigt. Dock hade båda dessa vår lägsta API level (10), vilket gjorde att vi tyvärr inte hade möjlighet att testa på vår target API (17) under projektets gång, något som möjligtvis hade kunnat göra applikationen bättre.

### Scrum

Att använda sig av scrum och dela upp ett projekt i prioriterade user stories fungerar väldigt bra i just ett sådant här sammanhang när projektet är relativt konkret och går att dela upp i mindre komponenter. Det har varit fördelaktigt att jobba med att genomföra det som är mest prioriterat först eftersom det garanterar att slutprodukten innehåller de viktigaste funktionerna. När sedan det viktigaste finns

med kunde man påbörja att göra det som adderade värde men inte var helt nödvändigt för att applikationen ska fungera, vilket vi tänkte mycket på när vi valde våra user stories. Utöver det har det varit fördelaktigt att kunna omprioritera och förändra vad som krävs av produkten under projektets gång. Det som tog tid var att skriva alla user stories, men när de slutligen motsvarade projektmålet på ett bra sätt var metoden snarare tidsbesparande än tidskrävande.

Eftersom jag aldrig använt mig av några andra projektmetoder för mjukvaruutveckling innan är det svårt att dra någon slutsats om hur denna metod skiljer sig från andra. Det jag kan jämföra med är mina erfarenheter av helt andra projekt men som även de bygger på att ett flertal personer arbetar tillsammans för att nå en slutprodukt. Tanken med scrum kan definitivt inspirera till upplägget av andra projekt då det bygger på att separera större deluppgifter och eliminera spill i form av onödiga diskussioner och långa möten. Jag tror därmed absolut att det är möjligt att inspireras av scrum-metoden i framtida projekt för att göra dessa mer lean, trots att andra typer av projekt ofta inte går att dela upp i just konkreta user stories som vi gjort i detta fall. Jag upplever alltså scrum som ett väldigt strukturerat och överblickbart sätt att planera och genomföra projekt som har relativt tydliga och avgränsade delkomponenter. Arbetssättet har även gjort att projektets innehåll har tvingats planeras från början och sedan revideras från vecka till vecka, vilket gjort projektmålet väldigt tydligt för samtliga i gruppen. Nackdelen med scrum var att det, speciellt till en början, var svårt att estimera hur mycket som skulle hinnas med under en begränsad tid. I början var det även svårt att veta hur omfattande en viss user story skulle vara att implementera.

**Pivotal Tracker**

Jag upplevde användandet av Pivotal Tracker som något otroligt positivt och ett viktigt hjälpmedel för att kunna applicera metodiken scrum. Metoden är framför allt väldigt lätt att förstå och att se den direkta nyttan av. Det ger en väldigt konkret överblick över vad som gjorts, när det gjorts och av vem, och vad som finns kvar att göra. Det är därför ett väldigt användbart verktyg för att arbeta med ett iterativt projekt som relativt lätt kan delas in i mindre oberoende komponenter, vilket inte nödvändigtvis måste vara ett projekt för mjukvaruutveckling. Trots att delmålen i andra projekt jag gjort tenderat att vara mindre konkreta och mindre avskilda från varandra tror jag definitivt att Pivotal Tracker utan problem skulle kunna användas i dessa sammanhang. Ett verktyg som detta är mycket värdefullt då det ger översikt och kontroll över projektets utveckling för samtliga inblandade. Det som kunde vara svårt var att definiera när en uppgift eller user story blivit uppfylld. Den avvägningen har inte alltid varit självklar då det ofta är möjligt att implementera en uppgift både på en grundläggande och en mer avancerad nivå. Därför valde vi att ha olika user stories om samma funktion men implementerat på olika nivåer, för att verkligen prioritera att göra det viktigaste för funktionaliteten först. Vi gjorde exempelvis att vårt djur kunde äta, gå och leka före vi skapade animeringarna för dessa aktiviteter.

**Git**

Det finns uppenbara fördelar med att använda versionshantering med hjälp av git, det är dock något jag aldrig använt mig av i någon tidigare kurs. Att använda terminalen är även det något helt nytt för mig vilket ärligt talat gjorde att förståelsen för git till en början var obefintlig. Nu i efterhand ser jag det som mycket värdefullt, för att endast programmera på en dator hade absolut inte fungerat. Om jag någon gång i framtiden skulle göra något mjukvaruutvecklingsprojekt som involverade många personer skulle jag absolut kunna tänka mig att använda git, eller liknande, för versionshantering. Detta speciellt eftersom jag nu redan vet hur det fungerar och inte hade behövt lägga någon tid på att lära mig det.

De största fördelarna med git var helt klart att det möjliggjorde parallell utveckling och att det gav möjlighet att gå tillbaka och studera tidigare commits. Användandet av git, när man förstått vad syftet var och hur man skulle använda det, fungerade oftast väldigt smidigt och tog generellt sett inte mycket tid från utvecklingen. Egentligen känns git därför som en relativt tidseffektiv teknik, men upplevdes inte riktigt så i början eftersom vi inte hade full förståelse för vad det innebar. Nackdelen med git upplever jag som att det ibland var tidskrävande att slå samman kod vid en merge och att fel ibland uppstod när fler än en person varit tvungna att arbeta i samma klass. Delvis berodde detta på att vi inte committade mot olika branches förrän i slutet, om vi gjort detta hade det möjliggjort utveckling av separata funktioner utan att behöva pusha till master varje gång. Det vi däremot gjorde bra var att vi, utan något enstaka undantag, höll oss till att aldrig pusha kod som inte var korrekt eller inte gick att köra.

**Testning av applikationen**

Under projektets gång genomfördes både automatiska tester, acceptanstester samt tester av kod genom logcat-utskrifter. Detta var nödvändigt för att kontrollera att de implementerade funktionerna fungerade som förväntat. Ingen av gruppdeltagarna har i någon större omfattning tidigare arbetat med enhetstester. Att testa applikationen, framför allt genom att automatiskt navigera mellan olika aktiviteter upplevde vi därför som mycket svårt. Därför använde vi oss av enklare enhetstester med hjälp av exempelvis assertEquals men till allra högsta grad av acceptanstester och utskrifter i logcat för att testa samma funktioner, vilket visade sig fungera bra. Jag upplevde ofta enklare tester, som utskrifter i logcat, som väldigt användbart för att felsöka under programmeringens gång och kontrollera att variabler förändras som förväntat av en viss händelse. Fler automatiska tester hade dock varit fördelaktiga att ha för att slippa manuellt testa samtliga funktioner varje gång en förändring gjorts. Dock är jag inte säker på att, med den begränsade förkunskapen av testning som gruppen hade, att vi verkligen sparat tid på att skriva mer omfattande automatiska tester.

## Arbetet i projektet

Under projektet delade vi varje vecka upp vad som skulle vara avklarat till nästkommande vecka, samtidigt studerade vi hur mycket vi hunnit med veckan innan för att få en bättre uppfattning om hur lång tid olika saker tar att genomföra. Projektets takt var medvetet högre i slutet, främst på grund av att vi skrev kandidatarbete på samma gång och med samma gruppdeltagare, men även för att

programmeringen var avsevärt lättare mot slutet. När väl en viss nivå av förståelse hade uppnåtts upplevde jag det som att utvecklingen gick väldigt snabbt. Under projektets gång gjorde vi två releases, en under tiden och en i slutet. Det upplevde jag som bra främst för att man kunde testa på att göra releasen innan den slutgiltiga och se hur det fungerade, men även för att det gav ett delmål under projektets gång. Den andra releasen innehöll signifikanta förbättringar gentemot den första, och det var intressant att se hur stor skillnad det blivit på relativt kort tid.

**Min arbetsinsats**

Under dessa få veckor, framför allt mot slutet, har jag lärt mig mycket, speciellt i form av att söka efter information, lösa problem och programmera. Det jag framför allt har bidragit med och gjort bra i projektet är implementation av GPS-funktionen, att förbättra sparningsfunktionen av data på interna minnet samt att införa tidsaspekten i form av att vår "moodbar" ska minska efter ett visst antal timmar och att vårt djurs ålder ska visas. Till en början hade vi problem med att spara djurets humör på det interna minnet vilket ledde till att vi gjorde vi en mindre bra lösning för att få det att fungera. Vi fick då onödigt många variabler i våra model-klasser och skapade inte objekt där vi borde vilket motverkade objektorienteringen. Detta satte jag mig dock och löste vilket jag tror bidrog till en klar förbättring av strukturen i koden. Jag insåg därför verkligen, även om jag egentligen visste det innan, att det i många fall är mer effektivt att tänka till både en och två gånger innan man börjar implementera någonting för att sedan slippa göra om det.

Något jag verkligen tar med mig från denna kursen är en förbättrad problemlösningsförmåga. Teknikproblem har aldrig varit min starkare sida och har alltid frustrerat mig. Jag känner dock nu, efter att ha varit tvungen att lösa en uppsjö av problem med eclipse och git att det inte är så svårt som jag trodde och att nästan alla problem har redan upptäckts av någon annan och har även en logisk förklaring. Det känns riktigt bra att kunna ha löst en massa problem själv.

**Gruppens arbetsinsats**

Med tanke på att ingen i gruppen tidigare testat androiduteckling eller använt git eller scrum är jag väldigt nöjd med att vi lyckades med att implementera alla de huvudfunktioner och lite till av det som vi diskuterade under första handledningsmötet. Det märks tydligt hur vi fick otroligt mycket mer gjort per utvecklingstimme de senare veckorna än de första. Den långsammare starten beror på att det tar lite tid att få en djupare förståelse för vissa delar innan man har testat lite olika lösningar och sett konsekvenserna. Det känns helt enkelt som ett learning by doing-koncept. Under de första veckorna när vi inte kommit igång än läste vi på mycket om hur vi skulle göra men det märktes att detta var svårt att ta till sig utan att faktiskt kunna testa det.

Som grupp har vi främst arbetat tillsammans i par men även själva. Vi har inte haft några direkta problem och kommunikationen har fungerat smärtfritt, främst eftersom vi jobbat så mycket tillsammans. Jag skulle uppskatta att vi lagt 700 timmar på detta projektet, med en jämn fördelning av timmarna mellan projektets deltagare. En stor del av denna tid har vi lagt på programmering och att lösa problem med git eller eclipse. Dokumentationen tog också tid men att skriva är något vi är

vana vid vilket gav oss väldigt mycket högre output per timma. Arbetet i gruppen har i övrigt fungerat väldigt bra eftersom vi är vana att arbeta tillsammans. Alla i gruppen har givetvis inte gjort riktigt samma saker och lika stora delar av varje moment men arbetsfördelningen har blivit jämn och oftast effektiv.

Det som fungerat mindre bra är de tekniska problem vi haft i form av brist på datorer och androidmobiler och även i form av att installera git och eclipse. Det har tyvärr sinkat oss onödigt mycket och ofta känts som om vi slösat tid. Det gjorde oss även väldigt beroende av att ständigt jobba tillsammans, vilket vi dock löste okej genom att mestadels programmera parvis. Det tog även lite tid för oss att vänja oss vid den begränsade mängden hjälp som fanns att tillgå, och därför kände vi oss till en början lite hjälplösa. Detta släppte dock verkligen på slutet när vi kom till insikt med att vi var tvungna och kunde lösa våra problem själva. Något som hade varit intressant att testa på hade varit att blanda grupper och göra projekt tillsammans med någon som exempelvis läser IT. Jag tror att jag hade kunnat lära sig mycket på att blanda helt olika bakgrunder och sätt att se på problem och kanske se helt nya lösningar. Risken med detta skulle varit att vi då inte tvingats lösa våra tekniska problem själva.

## Framtida projekt

Om eller när jag genomför ett liknande projekt i framtiden skulle jag börja med att ha en annan inställning till vilka saker som jag programmeringsmässigt skulle klara av att göra. Nu tänker jag att det går att genomföra mycket mer än jag trodde innan, bara man tar sig tid, eftersom det finns så mycket information att inhämta. Nu har jag dessutom grundläggande kunskaper som skulle göra det mycket lättare att komma igång. I ett framtida projekt skulle det även bli avsevärt mycket enklare att strukturera projektet då jag förstår förutsättningar och verktyg på en helt annan nivå än jag gjorde till en början. Jag tror bland annat att jag hade använt git på ett bättre sätt genom att committa mot branches. Jag tror även, som jag tidigare nämnde, att jag skulle lära mig mycket på att arbeta med någon som kan mycket mer om programmering än jag.

## Kommentarer på kursens upplägg

Det är tydligt att androidutveckling är byggt för open innovation och att all information i stor utsträckning finns för att ”vem som helst” med lite programmeringskunskaper ska kunna sitta hemma på kammaren och producera en mer eller mindre bra applikation. Det är lite av det som gör kursen svår att definiera. Att ha en kurs i något som är gjort för självinlärning och learning by doing/failing är svårt, speciellt om inte studenten (jag, som i-are) var inställd på att det är så det går till. Mina förväntningar motsvarade därmed inte riktigt vad kursen innehöll, vilket inte nödvändigtvis måste innebära att kursen innehöll fel saker, utan snarare att tydligheten behöver öka. Tidigare programmeringskurser vi haft har innehållit labbar vilka har inneburit hjälp så fort något blir fel. Det behöver därför vara tydligt att man i denna kursen bör söka svaret på problemen själv både en, två och tre gånger innan man kontaktar någon om hjälp. Jag inser och vet nu i efterhand att det är bra att inte alltid ha tillgång till hjälp så fort något litet problem dyker upp, och det hade underlättat om det tydligt framgick från början att detta är fallet. I övrigt hade det uppskattats om föreläsningar lades tidigt i kursen och något tillfälle för komma igång och ladda ner programmen (git och eclipse) hade hållits tidigt.