Post Mortem

DAT255 Software Engineering Project - lp4, vt2013

Elev: Anton Eksell
Grupp: AndroidSquad
Applikation: Coloristance

Rapporten är skriven med examinatorns frågor i åtanke men försöker även beskriva kursens flöde som helhet. Rapporten är en post mortem rapport för kursen men inte för projektet vilken speglas i vissa delar av texten med reflektioner om fortsatta steg.

Introduktion

Projektet bestod i att skapa en applikation till operativ systemet android. Applikationen skulle skapas tillsammans med 3-4 andra studenter med en satt deadline 29/5. Första steget blev att hitta och skapa en grupp vilket gjordes utifrån:

1. Personer som ingick i kandidatgruppen (vilket var en parallell process)
2. Vänner från som gick samma kurs

Denna process resulterade snabbt i en grupp bestående av 5 personer. Valet att vara 5 istället för rekommenderade 4 grundade sig i att en av oss inte kunde närvara första veckorna av personliga skäl. En grupp på 4 skulle drabbats hårdare av frånvaron och vi kände oss trygga med konstellationen.

Gruppen hade grundats och fick namnet AndroidSquad. För att komma fram till vad vi skulle skapa hölls en grundläggande diskussion om vad vi trodde var möjligt, vad som kunde vara intressant och lärorikt och vilka av dessa alternativ som kunde generera ett högt betyg. Vårt val blev att skapa ett spel. Egenskaper vi ansåg viktiga med detta spel var att det var lätt att förstå och roligt att spela och svårt att klara (för att ge det en rimlig livstid). För att få ett urval på spelidéer fick var och en i uppgift att skapa och visualisera ett spelkoncept denne trodde var möjligt att skapa under kursen, sagt och gjort. Med visualiserade och dokumenterade utkast diskuterade gruppen vilket förslag som kändes mest intressant att genomföra, därefter konsulterades Max om vilket utkast han tyckte verkade som ett bra val. Valet föll på Coloristance, något jag blev mycket glad över då detta var min idé och jag hade en tydlig bild av skapelsen framför mig. Gruppen i övrigt verkade inte heller ha några problem med idéen och krävde genast ett utförligare utkast för att kunna sätta sig in i spelidéen.

Processen

Gruppen kände sig till en början mycket rostig i java och helt okunnig i android vilket ledde till att de första veckorna gick till att leta bra information som var lätt att ta till sig och gällde just det som vi

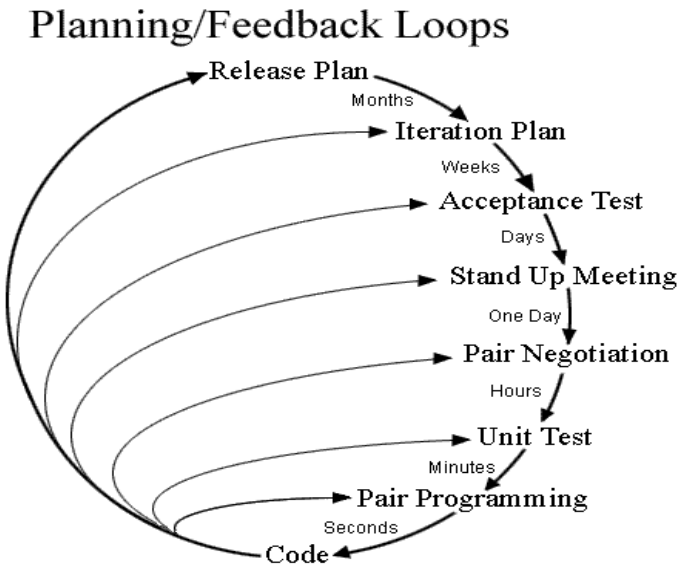
behövde för vårt projekt. Youtube och StackOverflow har i inlärningen varit helt ovärderliga, Youtube för mer generella saker som att lära sig hur man skriver kod för att komma igång i android, StackOverflow för mer specifika saker vad det gäller såväl lösningar på olika problem som faktiska felkoder och anledningar till krascher. Vi började koda utifrån diverse tutorials och delade hela tiden våra källor och de upplevelser vi hade av att använda dem. Denna del av kursen resulterade inte i mycket användbar kod men var vital för att i alla fall få lite kött på benen gällande android-kodande. Efter ca 2-3 veckor hade vi tillräckligt för att komma igång och en uppfattning om lite olika tillvägagångssätt och idéer om nästa steg. Inhämtandet av information fortsatte dock mer fokuserat.

När vi kände att projektet så sakteligen började rulla hade alla fått igång Eclipse, Android ADT och skrivit i alla fall lite kod. Nästa steg blev att programmera mer fokuserat på målet. Här började kaoset med att folk programmerade i samma saker, duplicerade och ibland förstörde varandras kod. Gruppen uppmärksammade detta och tog upp det med handledaren som efter att ha satt sig in i problematiken föreslog att vi skulle läsa på om MVC och försöka tillämpa metoden, sagt och gjort. All funktionalitet togs upp på en whiteboard och delades in under MVC (+ databas), detta förtydligade processen och hur vi kunde försöka dela upp programmerandet emellan oss. MVC fanns därefter i bakhuvudet under hela utvecklingen och kom ofta upp under diskussioner, vanligt var diskussioner om var en metod eller klass hörde hemma. MVC var mycket givande som metod, vi applicerade den inte fullt men metoden gav precis så mycket struktur som behövdes för att vi inte skulle lägga all tillgänglig tid på att lösa conflicts.

Som du som läsare kanske märker hade vi hela tiden en iterativ och inkrementell process som baserades på ett "learn by failure"-tänk, något jag tyckte fungerade väldigt bra. Jag uppskattade även kursens frihet vilken var ovan och välkommen.

Verktyg

De två första verktyg gruppen tog till sig var scrum via Pivotal Tracker och git via github. Vissa delar av scrum kom naturligt så som att dela in roller där jag blev Product Owner samt ScrumMaster för programmeringen, Alexander blev ScrumMaster för allt annat och hela gruppen Team Members. Att försöka bena ut funktioner fört att lägga dem som user stories var en bra process som tydliggjorde vårt mål och skapade diskussioner om prioriteringar och slutmål. Vi hade god kommunikation i gruppen och hade regelbundna uppdateringar om var alla var i sitt arbete. Problemet med scrum kom vid planeringen där ingen gruppmedlem kunde säga hur lång tid en uppgift skulle ta, ofta inte ens hur man skulle gå till väga för att uppnå målet. Detta berodde troligtvis mycket på oerfarenhet i att koda, detta ledde till att gruppen upplevde det mer passande att arbeta tillsammans, ofta med hela gruppen samlad i samma rum och utveckla enligt extremeProgrammings 2-manna koncept. extremePrograming kändes lättare att ta till sig och använda då det hade flera olika tidscykler för att vara så agilt som möjligt (tycker bilden beskriver det bra). De kortare cyklerna var väldigt viktiga för oss mer oerfarana.



Planerandet hölls väldigt generellt i början och mycket tid gick till att dela kunskap istället för att driva projektet framåt i kod. Jag tycker scrum hade poänger men att de kom naturligt för vår grupp och scrum därför totalt sett var mer jobb än nytta, i alla fall för oss som hade en relativt liten grupp med god kommunikation, liten individuell erfarenhet av att koda och gott sammarbete. Vårt fokus på user stories genererade sitt värde i början och blev därefter mer börda än nytta. Sprints tappade ganska snabbt prioritet då i inte var kapabla att förutsäga den tid det skulle ta, ibland gick det alldeles för fort och ibland mycket långsammare än man trott. ExtremeProgramming användes desto mer, att låta diskussioner om olika beslut först hållas av två personer gjorde att processen blev mer genomtänkt än med individuella beslut, samtidigt distraherades inte hela gruppen vid varje enskilt beslut/fråga. Denna sorts programmering användes även på distans/online genom skype-samtal och skärmdelning. Gruppen hade över lag en mycket agil utvecklingsprocess med högt i tak där alla fick möjlighet att lyfta sina tankar. Ett tecken på den agila processen är i min mening också det faktum att vi gick igenom de olika verktygen och tog till oss de delar som effektiviserade och hoppade över resten. Detta gäller även att hur vi tog till oss git. Till en början uppstod dock problem med att ingen av oss kunde git. Flera förstod inte ens var nyttan låg men vi prövade och kom fram till att git gav mer, mycket mer, än vad det tog i form av arbete. För att vi skulle kunna använda git så obehindrat som möjligt skapade vi en lathund för kommandon och förklaringar av deras funktionalitet (Git for Dummies).

Scrum kändes mer passande för professionella programmerare som har lite mer insikt om vad de är på väg att göra, extremePrograming verkar vara väldigt användbart så länge det gäller kreativa beslut där programmeraren/-na måste ta beslut om vägval, krävs inga ”vägval” skulle personerna jobba var för sig oh producera mer. Kanban var en teknik som kändes väldigt agil men också helt fel, tekniken verkade vara gjord för existerande system där brister uppstår och behöver åtgärdas, detta under tiden resterande delar fungerar dugligt. Den strukturen finns inte när både programmet och programmerarna mer eller mindre skapas från grunden, allting kommer att ha brister och behöva jobbas på...

Testning

Gruppen testade koden intensivt, vi hade flera android-enheter att pusha till och gjorde det ofta. Till en början var det mycket "manuell" testning där vi helt enkelt tittade på vad som hände på skärmen, tolkade och letade igenom koden efter ett fel som skulle kunnat orsaka detta fel. Ganska snabb kom fel som man inte grafiskt kunde tyda, gav de en krasch gick det oftast att hitta med hjälp av auto-genererade Log-meddelanden. Ganska snabbt hittade vi hur man skickade egna Log-meddelanden och kunde med hjälp av detta triangulera felen som fortfarande bestod av logiska fel exempelvis NullPointerExceptions. jUnit var något vi var ovana vid och den testningen delades till en början ut som ett separat ansvar, vi fick det att fungera men ansåg att detta testande fortfarande tog mer tid än vad det sparade. Den sista tiden av projektet som bestod av mycket letande efter små-buggar och fixande av dessa kändes det dock som att jUnit skulle ha varit effektivare, lätt att vara klok i efterhand... Eftersom vi sagt att vi skall fortsätta utveckla appen så kommer nog ett av de följande stegen dock bli en mer extensiv JUnit-testning. JUnit blir förmodligen riktigt användbart först vid ett senare tillfälle i kodningen men har man erfarenhet av det är det nog enklast att utveckla allt eftersom projektet växer.

Koden i sig

De flesta av oss var ovana att programmera utan tydlig ledning exempelvis via regleringar gällande vilka interfaces som skulle finnas, vilka loopar som fick användas och vilka anrop som skulle göras. Att inte vara så styrd gav möjlighet till mycket kreativitet men också många snabb-lösningar som man lät stå tillsvidare för att de fungerade och drivet att komma vidare/ att producera mer var större än kodens skönhet. Jag satt, i ett senare skede, mycket med att snygga till både min egen och andras kod ex. gällande duplicerad kod m.m. Att det gjordes då var för att det blev vitalt en bit in i processen då det blivit svårare att greppa hela projektet. Bättre kod krävdes helt enkelt för att effektivt kunna fortsätta. Stressen med att uppnå en minimum viable product fanns dock kvar och vägde i många lägen tyngre än snygg kod. Jobbet som lades på att snygga till kod blev framförallt där det blev en allt för gräslig och oläsbar härva. Kodandet kretsade mest runt programmets funktionalitet och då koden inte syns när programmet körs fick den något lägre omsorg. Gruppen kan sägas ha prioriterat funktion framför skönhet vilket fungerade när projektet fortfarande var relativt småskaligt.

Koden lyckades relativt bra med att använda objekt och hårdkoda så liten del som möjligt. I den inlämnade koden finns dock fortfarande ett antal ställen där ex. villkoren för for-looparna är hårdkodade men i de flesta fall finns en korresponderande variabel som skulle in men siffran var en snabbare och mer lättläst lösning för stunden, detta är något som kommer att ordnas i senare versioner. Framåt kommer heller inte stressa fram ny funktionalitet lika mycket. Istället kommer jag vara mer noggrann med kommentering och testning då det inte känns lika angeläget med "nytt" som med "bra" efter att vi äntligen fått ut en relativt bugg-fri och stabil produkt (MVP).

Tid

Tiden lagd på kursen är svår att bedöma av flera anledningar, ex. video tittandet gjordes till och från skola nästan varje dag och även en hel del hemma. Vissa kvällar satt jag uppe och kodade bra över midnatt och andra dagar låg prio på kandidatarbetet. Prioriteringar mellan kursen och kandidatarbetet gick fram och tillbaka, en sak är säker i alla fall och det är att många timmar lagts på projektet. En ruff uppskattning skulle vara mellan 240 och 280 h för min egen del. Svårt att avgöra hur det var för andra då vi inte alltid satt tillsammans ex. videos och kvällar/nätter har jag ingen aning om hur mycket andra satt med. Men jag vet att de i alla fall jobbat minst 60 % av dessa timmar då vi suttit tillsammans stor del av arbetad tid.

Gruppen

Vår grupp fungerade bra, vi var vana sedan tidigare att jobba i grupp vid olika arbeten. När jag säger bra syftar jag på framförallt på kommunikation och arbete nedlagt. Det var en öppen atmosfär och vid minsta tvekan om ett beslut så gick det att bolla med valfri medlem. Mindre bra skulle ha varit vår ojämnhet i kod-vana, det var ibland svårt att dela problematik gällande ren kod då vi inte alla befann oss på samma ställe gällande kunskap om hur man kodar både generellt och i android. Dock var detta allmänt erkänt och alla drog sitt strå till stacken på ett eller annat sätt genom att ta de delar man kände att man hade grepp och kunde tillföra. Min känsla är att alla i gruppen lärde sig väldigt mycket under kursen och gruppen bibehöll en positiv stämning rakt igenom trotts att vi ibland jobbade under mycket pressade förhållanden.

Avslutande reflektioner

Det har varit en bra kurs där jag/vi lärt oss enormt mycket. Att programera gentemot Android känns väldigt "i tiden" och upplevs som en värdefull kunskap att ha med sig. Efter kursen känner jag mig inte tveksam över att säga att jag skulle kunna programmera en enklare android applikation åt någon. Olika metoder att utföra projektet på var bra att ha med men det kändes som att mycket föll sig naturligt, troligen på grund av att vi har mycket erfarenhet av grupparbeten sedan tidigare och därför utvecklats inom detta område. Den svåra biten var den faktiska kunskapen; att hitta, tillskansa sig och tillämpa kunskap inom ett helt nytt område. Det är den första kursen som jag på eget initiativ aktivt kommer att arbeta vidare med. Coloristance kommer att lanseras på Google store!

Extra stort tack till Max som har varit en klippa i att handleda oss!

Anton Eksell, Industriell Ekonomi –Vt2013