What To Do

The App that will change your life

Post-Mortem Experience Report

Albin Bjurhall
Michaela Fritiofsson
Emelie Hedqvist
Edvin Nordell
Gustav Swedberg
Henrik Täljedal

Inledning

En gemensam åsikt gruppen har efter avslutad kurs är att det har varit den roligaste och mest kreativa kurs vi haft under kandidattiden på Chalmers. Att vi som grupp själva har fått ta fram och skapa en användbar applikation utifrån våra egna tankar och idéer har gjort att alla gjort sitt yttersta för att bidra till ett bra slutresultat. Gruppen känner att kursen har bidragit till en bättre förståelse för agil systemutveckling och dess för- samt nackdelar. Under kursens gång har syftet med att arbetet bedrivs inkrementellt och iterativt, där mindre leveranser sker löpande för att på ett helt annat sätt möta beställarens krav och önskemål, tydligt uppenbarat sig. Gruppen har också fått god förståelse för hur agil systemutveckling skiljer sig i jämförelse med den traditionella vattenfallsmodellen.

Hela gruppen hade önskat att det hade gått att lägga än mer tid kursen än det har gjorts. Den parallellt genomförda kandidatuppsatsen har mer eller mindre omöjliggjort att mer tid hade kunnat läggas på app-utvecklingen. Vi har spenderat den tid vi har kunnat efter förutsättningarna och arbetsbelastningen har varierat över projektets gång, allt från några få till 30 timmar per vecka. När inlämningen av kandidatarbetet närmade sig sjönk antalet arbetstimmar något, men efter inlämning ökade antalet timmar kraftigt. Alla medlemmar i gruppen har varit ambitiösa och utdraget över hela projektet uppskattas alla ha jobbat ungefär lika mycket. Totala antalet arbetstimmar för gruppen uppskattas till cirka 450 timmar eller cirka 75 timmar per person.

Gruppen var redan från start överens om att projektet skulle vara kul och samtidigt leda till en häftig, och användbar, app som vi själva skulle vilja använda. Till en början användes ingen tydlig strategi eller modell för hur uppgiften skulle genomföras för att nå slutmålet. Men för att kunna slutföra uppgiften krävdes emellertid en tydligt struktur och arbetsgång. Genomgående i rapporten beskrivs arbetsprocessen gruppen antagit samt de metoder som har anamats samt uteslutits under projektets gång.

Arbetsprocessen

I följande kapitel presenteras hur vi har arbetat med att utveckla appen. I kapitlet presenteras hur vi har använt oss av Scrum och eXtreme Programming samt reflektioner vad som har gått bra respektive dåligt med att använda dem. Vidare diskuteras hur verktyg, såsom GitHub, har fungerat att använda.

Scrum - ett arbetssätt som uteslöts

När scrum presenterades genom legobyggandet kände vi att arbetsmetoden kunde vara lämplig för vårt projekt. Vi gillade idéen med korta, iterativa sprinter där vi vid varje avstämningstillfälle var tvungna att presentera våra framsteg för vår kund i syfte att få feedback och godkännande att fortsätta till nästa steg i utvecklingsprocessen. Arbetssättet skulle innebära att vi hela tiden hade små deadlines som vi var tvungna att hålla och med det säkerställa att vi fick en kontinuitet i utvecklingsprocessen.

Således inleddes projektet med en arbetsmetod färgad av Scrums logik. Vid första sprinten hade vi redan skapat user stories, dessa försökte vi sedan skatta och poängsätta utifrån hur mycket arbete vi trodde skulle krävas under en vecka, vilket motsvarade en sprint. Vår betygsättning bestod av tal mellan 1 - 20. Dessa user stories blev även tilldelade en ansvarig person. Efter två sprintar insåg vi emellertid att vi egentligen inte hade någon uppfattning om hur lång tid olika uppgifter tog att slutföra, exempelvis trodde vi att det skulle vara en enkel uppgift att lära oss använda GitHub, vilket i slutändan tog mycket längre tid än vi hade förväntat oss.

Under samma tid hade Thomas Luvö en föreläsning om "Scaling Agile", vilket ökade vår förståelse för hur man tillämpar Scrum. Han gick igenom hur man använde backlog, themes, epics, user stories och tasks etc. Efter lektionen insåg vi ganska snart att vi inte riktigt förstått tanken bakom user stories. Med den nyfunna insikten om vad user stories faktiskt var och kunde användas till delade vi istället upp våra user stories i tasks. I samband med det startade vi också upp en backlog i excel som gav oss en bra överblick över våra tasks, alltså vad vi skulle göra, samtidigt som den bidrog till bättre ordning i projektet.Trots att vi hade strukturerat upp projektet fanns det fortfarande delar som vi ännu inte riktigt bemästrade, och då främst poängsättningen av våra tasks. Vi hade ingen egentlig uppfattning om komplexiteten hos våra tasks och med det inget bra sätt att avgöra vilken ansträngning en viss task krävde. Utöver det utsåg vi inte en ScrumMaster eftersom vi inte riktigt förstod innebörden, och vikten, av rollen.

Inför varje ny sprint hade vi ett möte för att uppskatta vad som kommer att hinnas med fram till nästa avstämningstillfälle. Vilket vi tyckte fungerade bra för vårt sätt att arbete då vi, som tidigare nämnts, fick en deadline samtidigt som vi kunde bygga appen steg för steg och addera ny funktionalitet efter hand. Nu I efterhand inser vi dock att vi missade en viktig aspekt med dessa möten, vi utvärderade egentligen aldrig en tidigare sprint utan blickade bara framåt. För att kunna poängsätta en ny task krävs erfarenhet av tidigare poängsättning. Då vi aldrig utvärderade hur många poäng en task borde haft hade vi inte rätt kännedom för att poängsätta en ny, och därför löste vi egentligen aldrig svårigheterna med att poängsätta våra tasks. Värt att nämna här är emellertid att trots att vi till viss del misslyckats med att använda sprintsen på rätt sätt ändå hade haft svårigheter i att poängsatta tasks. Det eftersom sex sprintar ändå inte ansågs som tillräcklig erfarenhet för att kunna poängsätta tasks inom en realistisk ram. Vidare insåg vi efter ett tag att planeringen vid Scrum tog mycket tid i förhållande till hur mycket tid som faktiskt las på själva appen.

Efter att ha försökt använda oss av Scrum-metodiken utan att fullt ut ha lyckats har vi insett att Scrum-metodiken i sin helhet var alltför tidskrävande för vårt projekt, med låg komplexitet och liten omfattning. Vi anser att Scrum lämpar sig bättre vid mer omfattande utvecklingsprojekt med fler arbetstimmar, längre tidsspann och fler inblandade människor.

I ett skolprojekt där tid är en begränsande faktor och omfattningen på projektet inte är i den grad som när Scrum används i vardagen på en arbetsplats känns det som att många delar av metoden är för avancerade. Då inte ScrumMaster eller Product Owner utsågs missade vi hela poängen med att uppnå kontinuitet med backlogen och reflektera över vad som gick fel och vad som gick bra efter varje sprint. Därför känner vi att denna arbetsmetod i sin helhet kan lämpa sig bättre för erfarna programmerare som arbetar med mer komplexa och mer omfattande projekt. Med detta i åtanke började vi att röra oss mot att försöka använda oss av metodiken kring eXtreme Programming.

eXtreme Programming (XP) - ett urval av det bästa

Efter att inte riktigt ha lyckats med att använda Scrum-metodiken började vi titta på hur vi kunde använda oss av eXtreme Programming. Med Scrum-försöket i åtanke valde vi att angripa XP på ett annat sätt. Istället för att försöka anamma arbetsmetoden i sin helthet plockad vi bitar från XP som vi ansåg intressanta och som kunde ge mervärde under arbetsgången.

I Scrum-metodiken fann vi ingen bra hjälp i hur vi skulle koda och därför hade vi från början ingen tydlig struktur när vi skulle sätta oss ner och försökte få fram vår tänkta app. Resultatet blev att vi satt och programmerade själva, mest för att lära känna Android Studio, utan någon riktigt återkoppling från resten av gruppen. Ganska snart insåg vi att detta inte höll då gruppens kompetens var spridd och därför inte utnyttjades på bästa sätt. För att lösa problemet började vi på försök att programmera i grupper om två personer, som inom XP benämns *pair programming.* Att programmera i par föll sig emellertid ganska naturligt då vi är vana att programmera två och två sedan tidigare kurser. Parprogrammeringen möjliggjorde bättre kodkvalitet och synergieffekter av två personers olika kunskaper men även ett roligare sätt att arbeta.

En klar fördel med parprogrammeringen var att vi lättare kunde lösa problem som uppstod, den ene kunde leta efter lösningar på internet medan den andra testade de potentiella lösningarna. Att programmera i par fungerade mycket väl och således höll vi oss till det genom större delen av app-utvecklingen. Varje par var ansvarig för en specifik del, eller funktion, och jobbade i en egen branch (se avsnittet om gitHub för utförligare beskrivning). I paren varvades det mellan att använda en och två datorer. För att inte tappa kopplingen till resten av gruppen träffades vi kontinuerligt och gav feedback på varandras framsteg.

Parprogrammeringen begränsade dock inte möjligheten för par att jobba med andra pars kod. I samband med att vi delade upp utvecklingen genom parprogrammeringen kom vi i gruppen överens om att all kod är allas kod, *inom XP kallat collective code ownership*, vilket skapade förutsättningar för att alla i gruppen optimerade all kod och inte bara "sin egen kod". En annan del av XP som vi arbetat efter är *simple design*, en princip som hela vår app grundar sig i varför den föll sig naturlig att arbeta efter.

Som tidigare nämnts arbetade vi genom sprinter som var en av de delar vi tog med oss från det initiala försöket med Scrum. I varje sprint arbetade vi med att implementera och förbättra design och kod, vilket inom XP behandlas som *design improvement*. Ytterligare en del som vi tog med oss från Scrum är *Small releases* vilket även är en del inom XP.

Genomgående i kodningen använde vi *coding standards,* detta för att underlätta arbetet för varandra och vid varje metod skrev vi en förklaring till vad metoden gjorde eller hade för syfte. *UserStories* är även en del av XP som vi redan, med varierande framgång, arbetat efter.

Många av delarna som ingår i XP kunde inte användas och dessa var *whole team, continious integration, Test-Driven development, customer test, small releases, sustainable pace and metaphores*. Många av dessa delar var för omfattande, resurskrävande och tidskrävande för vårt projekt. *Whole team* innebär bland annat att gruppen arbetar i ett rum, detta har inte varit möjligt för oss vare sig resurs- eller tidsmässigt. Vi tror däremot på idéen att arbeta nära varandra, då det möjliggör transparans och snabb tillgänglighet till viktig information. *Countinious integration* menar att varje natt skall en " working new version" släppas, detta är ingenting vi applicerade. Att vi inte använde oss av delen *Test-Driven development* i arbetsmetoden är för att det hade varit alltför

omfattande att arbeta på det sättet. Denna del menar att ingen rad kod skrivs innan ett test för raden har utformats. Delen *customer test* användes inte eftersom ingen kod skrevs av vår kund,Morgan, som försäkrade att kundens krav uppfylldes. Att arbeta efter *small releases* var inte möjligt i och med att omfattningen på vår app innebar att när vår app var klar så var det en release. Detta är någonting som kan tillämpas på större projekt. I vårt projekt var det svårt att tillämpa metoden *sustainable pace* då detta inte är vårt dagliga arbete om 40 arbetstimmar per vecka. Fokus låg på att få ihop appen och den tid detta krävde lades på apputvecklingen. Detta kunde absolut innebära mer än 40 timmarsarbetsvecka i kombination med kandidatarbetet. Vi tycker att detta är en vettig del av XP då det ger upphov mer långsiktigt hållbart arbetsliv. Att vi inte använde oss av *metaphores* föll sig naturligt då vi inte tidigare har hört talas om en liknande idé som vår app var. Däremot ser vi ingen nackdel i det då app-idén inte var särskillt svår att förklara.

Eftersom vi använde de delar av XP-metodiken vi tyckte passade omfånget på vårt projekt var denna teknik i jämförelse med Scrum mer effektiv i tidsanvändningen i förhållande till den tid som lades på projektet. Så skulle möjligtvis inte vara fallet om alla delar av XP skulle anammas. Vi får känslan av att denna metod passar både stora och små projekt då det är en metod som lätt kan skalas upp och ner. Detta är en metod vi känner att man kan använda i många olika typer av projekt i framtiden. Ifall projektet däremot skull involvera väldigt många människor skulle däremot en annan metod möjligtvis användas.

Vår slutliga arbetsmetod - en blandning efter våra kriterier

Vid reflektion efter det att projektet avslutats ser vi saker tydligare, vi har egentligen aldrig följt någon agil arbetsmetod fullt ut. Som redan nämnt testade vi först Scrum för att sedan testa XP. Slutligen kan vi konstatera att vi under slutskeendet av projektet arbetat efter båda delar, eller en optimerad kombination som passade vår grupps sätt att arbeta. Vi använde oss av backlogen för att få en överblick om vad som måste göras och till när. Vi slutade att poängsätta arbetsuppgifterna pågrund av tidigare nämnda anledningar. Vi fortsatte jobba med pair programming under en längre tid för att sedan i slutskeendet av apputveckligen sitta mer ihop som en hel grupp. Detta upplägg passade oss bra då appen till en början bestod av tre separata arbetsdelar som kom att integreras till en i slutet av projektet.

Verktyg som använts under projektets gång

Git och GitHub - hatet och förälskelsen

Vår första sprint handlade till stor del om att förstå och lära oss GitHub, ett verktyg som används för att sammanfoga kod. GitHub kom att bli till stor hjälp under projektets gång. Detta verktygs komplexitet underskattades av alla medlemmar, vilket kom att leda till suboptimeringar i programmeringen till en början. De olika programmeringsgrupperna om två arbetade på tre olika projekt. Detta var möjligt för oss eftersom projektet bestod från början av tre olika definierade delar: Design, funktionalitet samt platssökning. Med design menar vi appens grafiska utseende, alltså XML koden. Med funktionalitet menar vi koden för vad som hände då vi exempelvis tryckte på en knapp och med platssökning så menar vi koden för att kunna genomföra en sökning av närliggande platser med hjälp av Google Places API Web Services.

Det tog oss två sprinter att förstå och synkronsiera ihop allas datorer med GitHub. Vi började med att försöka "snabb-lära" oss GitHub genom det som vi trodde var den enklare versionen att förstå, alltså den grafiska vyn som kan laddas ner. Vi insåg efter ett tag att vi behövde förstå den bakomliggande logiken till GitHub och övergick därmed till att använda oss av terminalen. Den metod vi trodde var en genväg visade sig vara en omväg. Då alla inom gruppen tillslut fick GitHub att fungera och förstod logiken kunde vi därefter återgå till det trevligare interfacet. Många i gruppen ansåg att det gav en bättre överblick över vad som hade utförts och inte. För att uppnå en snabbare inlärningsprocess hade det varit önskvärt att ha en manual på kursens hemsida med GitHubs vanligaste kortkommandon.

Gruppen hade som mål att lansera appen på Google Play. Då alla gruppmedlemmar skall ut i världen för att studera eller jobba skulle vi tycka att det varit otroligt roligt att testa appen på annan ort och se om appen skulle ge mervärde till användaren.

Branches - och dess storhet

Då vi arbetade i par under programmeringen och hade tre väldigt tydligt uppdelade arbetsområden (platssökning, design och funktionalitet) insåg vi storheten med branches, dock kanske inte hur man optimalt använder dessa. Varje par skapade en egen branch och kunde därmed arbeta klart med sin del av uppgiften för att sedan under projektets gång integrera de delar av projektet i dess naturliga ordning. Stegen vi gick efter var att först sammankoppla designen med funktionaliteten för att därefter sammankoppla funktionaliteten med platssökningen. Därefter jobbade vi alla i samma branch. Egentligen arbetade vi på två olika appar under större delen av tiden, en app som var fokuserad på design samt funktionalitet och en app som utförde de uppgifter appen var ansedd för men utan egentlig design. Det var otroligt smidigt att koppla ihop de olika delarna eftersom och vi upplevde inga egentliga problem med verktyget då vi väl förstått det. GitHub kom till slut att bli en otrolig hjälp för oss under projektets gång.

Hade vi däremot förstått hur branches optimalt användes hade vi inte arbetat på det sättet vi gjorde. Vi hade istället haft 3 olika branches men hela tiden synkat emot master. Då vi inte gjorde detta fick vi problem då vi hade kopplat samman de 3 olika arbetsområdena, vi jobbade emot en branch som inte var master. En annan branch än master använde vi alltså som master, vilket egentligen inte är optimalt. För att andra utvecklare och intressenter skall kunna få en överblick över vår repository bör de kunna se allt i master. I slutet av projektet fick vi istället synka vår "masterbranch" till master.

Backlog

För att sätta upp en backlog använde vi oss av Excel. Vid större projekt finns behov av att använda ett annat verktyg än just Excel för att sätta upp en backlog, möjligtvis organiseringstjänsten Trello. Till ett projekt av denna storlek bidrog Excel med en bra överskådlighet samt hjälpte till att anordna en bra struktur för våra uppgifter, user stories, epics samt themes.

Initiala skisser

I början av arbetets gång, när vi precis bestämt oss för vilken app vi skulle utveckla, togs det fram skisser som illustrerade exempel på de olika layouterna som så småningom skulle utgöra grunden till

appens design. Då vår grupp aldrig arbetat i ett software project innan var detta en väldigt bra hjälp på vägen då det gav en initial riktning att arbeta mot. Bilderna gjorde att vi alla hade en gemensam uppfattning om hur vår slutprodukt skulle se ut, om än inte specifikt. Vi utnyttjade alltså befintlig kunskap i programmet Photoshop för att underlätta samarbetet. Detta är antagligen inget effektivt arbetssätt i större projekt men för oss var det till stor hjälp.

Avslutande Reflektioner

Som inledningsvis nämndes är hela gruppen överens om att detta varit en otroligt rolig, spännande och lärorik kurs. Om vi hade fått göra om projektet med precis samma villkor och förutsättningar hade vi först och främst lärt oss använda GitHub på rätt sätt för att undvika de suboptimeringar som uppstod. Vidare hade vi då förstått värdet i att efterhand synka med masterbranchen och på så sätt hålla samtliga medlemmar uppdaterade om hur vi låg till och hur resterande delar av koden var uppbyggd. Det hade också underlättat vid den slutliga sammanslagningen av alla funktioner. Ett annat sätt att underlätta sammanslagningen hade varit att se till att alla grupper arbetade med koden mer nära varandra istället för på varsitt håll.