

ITERATION

LABORATORIES



Software Engineering Project - DAT255

Läsperiod ett 2017

Författare:

Robert Agrell, Jack Ahlkvist, Niklas Baerveldt, Linnéa Bark, Alice Gunnarsson, Max Johansson, Eli Knoph, Jonatan Magnusson, Aron Sandstedt, Matilda Sjöblom, Rasmus Tomasson

Sammanfattning

Med syfte att utföra ett projektarbete med användning av en agil arbetsmetod genomfördes ett försök till att utveckla kolonnkörning på radiostyrda bilar, kallad MOPED. Största fokuset har legat på att lära sig arbeta med en agil arbetsmetod och därefter reflektera kring dess effektivitet. Rapporten fokuserar huvudsakligen på hur scrum applicerades och vilka verktyg gruppen använde för att praktisera metoden. I slutet uppnåddes longitudinell och delvis fungerande lateral autonom körsystem på MOPED:en, vilket presenterades under en demonstration.

Innehåll

| | |
|---|-----------|
| 1 Inledning | 3 |
| 1.1 Bakgrund | 3 |
| 1.2 Syfte | 4 |
| 2 Teori | 5 |
| 2.1 PID-Controller | 5 |
| 2.2 Velocity | 5 |
| 3 Metod | 6 |
| 3.1 Trello | 6 |
| 3.2 KPI | 6 |
| 3.3 Slack | 7 |
| 3.4 Hur teamet applicerade scrum | 7 |
| 4 Produkten och dess process | 8 |
| 4.1 Praxis för att använda nya verktyg och ny teknik | 8 |
| 4.2 Applikationen | 8 |
| 4.3 ACC - Adaptive Cruise Control | 9 |
| 4.4 Lateral styrning | 9 |
| 4.5 Tester | 9 |
| 4.5.1 Testningverktyg | 9 |
| 4.6 Förhållande mellan process, produkt och produktägare | 10 |
| 5 Reflektion och diskussion | 11 |
| 5.1 Scrum | 11 |
| 5.2 Nedbrytning av user stories | 11 |
| 5.3 Teamets process i förhållande till gästföreläsarnas processer . . . | 12 |
| 5.4 Tillbakablickar från sprintar | 13 |
| 5.5 Sprint reviews | 16 |
| 5.6 Reflektion över KPI:er | 17 |
| 6 Slutsats | 19 |
| 7 Referenser | 21 |
| 8 Appendix | 22 |

1 Inledning

Allt eftersom informationsteknik som bransch har vuxit och blivit mer konkurrenskraftig har det blivit aktuellt att optimera hur man arbetar med framförallt mjukvaruutveckling. Mjukvaruindustrin skiljer sig från många andra industrier eftersom den har möjlighet att vara väldigt dynamisk och kan utan att förlora annat än tid gå bakåt i processen och ändra sin produkt efter hand. Vattenfallsmetoden är en metod som fungerande används inom till exempel byggindustrin. Den förklarades av Christian Frithiof på '8 Dudes in a Garage' som en metod där en grupp får en uppgift, planerar uppgiften, jobbar på uppgiften stevvis och får fram en produkt som kunden inte vill ha. Ordet vattenfall förklarar arbetsprocessen, den går stevvis och aldrig tillbaka, vilket är skillnaden mot en agil process. Vattenfallsmetoden har visat sig vara ett dåligt sätt att arbeta på inom mjukvaruutveckling då slutprodukten sällan går att förutse lika konkret. I nuläget anses agila arbetsmetoder vara de mest effektiva att använda. Det är därför aktuellt för IT-studenter att lära sig arbeta agilt. Till inlärningen av en agil arbetssmetod genomfördes ett projekt med mål att utveckla mjukvara till eldrivna modellbilar. Den agila arbetsmetoden scrum praktiseras och följande rapport gör en ansats till reflektion kring metoden. Då fem iterationer, så kallade sprintar, genomfördes kretsar reflektionen främst kring dem samt eftertankar om helheten.

1.1 Bakgrund

Att arbeta agilt innebär att jobba dynamiskt; att anpassa sin arbetsgång och sätta mål efter tid. När ett projekt följer ett agilt arbetssätt är det avgörande med kommunikation mellan utvecklare och kund eller produktägare. Många projekt tar lång tid att utföra och att kraven på produkten ska vara konstanta över hela loppet känns i vårt ständigt föränderliga samhälle som någonting väldigt ovanligt.

Automatisering ökar mer och mer i nuläget då det potentiellt kan leda till högre effektivitet samt säkerhet [1], därmed är självkörande bilar ett aktuellt område att arbeta inom. Många företag väljer idag att storsatsa på utvecklingen av denna teknik, däribland Volvo och Zenuity.

Studenter på Chalmers teknologiska högskola tilldelades uppgiften att utveckla mjukvara på radiostyrda bilar för att uppnå så kallad kolonnkörning. Uppgiften skulle genomföras med hjälp av radiostyrda bilar och dess hårdvara. Bilarna till projektet var utrustade med tre stycken Raspberry Pi-datorer, en ultraljudssensor och en kamera. Om kolonnkörning uppnås i verkliga sammanhang kan det leda till minskad bränsleförbrukning för fordon genom ett minskat luftmotstånd, snabbare transporter, mindre trafikstopp och mindre trafikolyckor [2].

1.2 Syfte

Uppdraget som tilldelats innebär ett försök till utveckling av kolonnkörning. Med huvudfokus på den agila arbetsmetoden scrum har grupper om 10 - 12 personer arbetat mot målet kolonnkörning under skedet av 6 veckor. Syftet med att utveckla kolonnkörning är att nyttja de positiva effekterna som det potentiellt kan föra med sig. Huvudsyftet med rapporten är att reflektera över den agila arbetsmetoden scrum och hur den applicerades under projektet.

2 Teori

Mycket av teorin som används i projektet kommer från kursens föreläsningar och förväntas därav känna till av läsaren. Kunskap från tidigare kurser såsom grundläggande programmering tas inte heller upp på grund av läsarens förkunskaper. Det som är nytt och främmande är PID-Controller och velocity.

2.1 PID-Controller

En så kallad PID-Controller eller PID-Regulator är en regulator som ofta används inom reglerteknik då ett system är i behov av någon form av konstant reglering för att uppnå ett optimalt tillstånd. PID-förkortningen är en kombination av de tre delarna som en PID-Controller består av, en Proportionerlig del, en Integrerande del samt en deriverande del [3]. Huvudsyftet är att minimera oscillation som kan uppstå när ett system försöker reglera till det optimala tillståndet samt att minimera externa störningskällor. I fallet med bilarna är det optimala tillståndet den optimala hastigheten som bilen bör ha för att därmed upprettihålla det optimala avståndet. Det var därför aktuellt att studera hur en PID-Controller fungerar för att uppnå kolonnkörning. Vad som dock implementerades var en relativt avskalad och modifierad regulator, eftersom en fullt utvecklad PID Controller hade varit överdrivet och onödigt då bilens egna begränsningar strypt större delen av PID Controllern.

2.2 Velocity

Velocity är ett sätt för ett scrum team att mäta hur mycket arbete teamet fått gjort under en sprint. Varje user story får poäng som sedan räknas under sprintens gång i takt med att user stories blir avklarade. Om en user story inte blir fullständigt klar så ska inte dess poäng räknas. När teamet mätt hur mycket de i genomsnitt klarar av under en sprint är tanken att det blir lättare att sätta upp en lämplig arbetsmängd inför kommande sprintar [4].

3 Metod

För att använda scrum med så tillfredsställande resultat som möjligt krävs det ett tydligt arbetssätt och en omfattande gemensam metod att arbeta utifrån. I denna sektion förklaras hur Iteration Laboratories tacklade problemet.

3.1 Trello

Sprintarna gick över en vecka och på fredag förmiddag planerades varje ny sprint med user stories och tillhörande velocity, samt tasks med effort. Gruppens user stories presenterades i ett program vid namn Trello som är ett webb-baserat program där uppgifter kan sammanställas och flyttas runt mellan olika under-rubriker.

I Trello skapades en sprint backlog innehållandes user stories, vilka sedan delades upp i olika tasks. (Se figur 6 i appendix) Tasks var uppgifter som kunde tas i anspråk av en eller flera gruppmedlemmar och flyttas över brädet till rubriken 'Under Progress' och ifall uppgiften blir klar slutligen 'Done'. I Trello fanns även möjlighet att kommentera varje uppgift ifall framsteg gjordes gällande uppgiftens lösande. Under rubriken 'TODO' fanns även uppgifter utan velocity som möten med plats och tid eller torsdagsfika.

3.2 KPI

För att ena gruppen mot gemensamma mål användes tre KPIer.

- Velocity
- Motivationsnivå
- Mötesnärvaro

Motivationsnivå mättes genom att på varje daily scrum fråga varje person hur motiverade de var. Motivationsnivån kunde vara -1, 0 eller 1. Resultatet tillsammans med närvaro sammanställdes i ett dokument på google drive.

3.3 Slack

Hela kursen använde sig av ett team på Slack som kommunikationsverktyg mellan alla studenter, diverse handledare samt föreläsare. Gruppen valde även att skapa ett eget team på Slack, så att teamet kunde skapa egna kanaler att kommunicera i utan att behöva blanda ihop dem med en massa andra kanaler.

I gruppens privata slack-kanal skapades olika kanaler för att lätt kunna hitta rätt information. 'Mötesplats' innehöll endast plats, tid och datum för alla möten. Inget övrigt fick skrivas och varje gruppmedlem var tvungen att 'reagera' på varje möte för att visa att de hade fått informationen om tid och plats. I kanalen 'möten' skedde övrig kommunikation om möten, till exempel om någon skulle bli sen, eller inte kunna närvara överhuvudtaget. Andra kanaler som skapades var 'problem', 'resurser', 'rapporten' och 'appen' som förklarar sig själva. Kanalen 'general' agerade kanal för övrig relevant kommunikation som rörde teamet.

3.4 Hur teamet applicerade scrum

I teamet fanns rollerna scrum master, sekreterare, MOPED-ansvarig och grupperumsbokare för möten. Scrum master och MOPED-ansvarig blev stående roller som Matilda och Alice sattes till. Sekreterare och grupperumsbokare var flytande roller som sattes vid varje tillfälle.

Inför sprint tre delades gruppen in i två mindre grupper. En grupp arbetade med appen och en grupp med ACC.

4 Produkten och dess process

För att applicera scrum behövs ett projekt och för att ett projekt ska ha något mättbart värde så behövs en produkt. Processen under detta projekt har varit krokig och lärorik, denna sektion reflekterar över produktens process.

4.1 Praxis för att använda nya verktyg och ny teknik

Under kursen har mycket ny teknik och nya verktyg introducerats och att använda dessa har varit obligatoriskt för att ta processen framåt. Vad gäller mjukvaruprojekt så är det i första hand API:er som måste studeras men också att planera och välja infallsvinkel för att lösa uppgiften på ett effektivt sätt är viktigt, eftersom det ofta finns flera tillvägagångssätt för att nå samma resultat.

På gruppnvå kom insikten om vikten av kommunikation, att alla var med på vilken väg som valdes för att lösa ett problem och ta till sig ny teknik. Stor projektgrupp innebar stor kompetensbredd och möjligheten att dela upp arbetet att tolka API:er och annat nytt efter tidigare erfarenheter med samma eller liknande uppgifter kom väl till nytta. Det gällde dock att jobba agilt då det visade sig initialt vara svårt att avgöra var gruppens resurser skulle läggas när något främmande introducerades.

På individuell nivå var projektets största uppgift att lära sig att hantera den existerande tekniken. Eftersom det ansågs enklast att utöka mjukvaran som redan fanns på MOPED:en så var det av största vikt att lära sig hur detta fungerade och hur koden kunde bli manipulerad till att utföra uppgiften. Så hur tar en till sig ny teknik på bästa sätt? Först och främst sågs all existerande dokumentation över. När den visade sig inte vara till så stor hjälp så gällde det att använda sina kommunikationskanaler. Då det var känt att många andra satt med samma problem så var det till stor hjälp att kommunicera med de övriga grupperna och utbyta information. Utöver att utbyta sina framgångar med andra i samma sits gällde det att inte vara blyg för att testa sig fram, för metoder att testa koden se nästkommande stycke.

4.2 Applikationen

En mobilapplikation som kunde styra MOPED:en var ett krav från produktägaren. GUI:t skulle vara intuitivt och simpelt. Androidappen som gruppen skapade i detta projekt har en tydlig röd knapp för att sätta på ACC och även en röst som beskriver läget för ACC:n, på eller av. (Se figur 8 i Appendix.) Om man inte är kopplad till en socket visas en röd text med 'not connected', när man är kopplad blir den gröna och det står 'connected'. Två stänger beskriver hur man kan styra MOPED:en manuellt. Det går ej att ändra hastighet när ACC är på, men det går att styra den lateralt. Applikation är simpel och har inga onödiga funktioner för användaren. Skulle man vilja nödstänga av MOPED:en kan man

trycka på 'disconnect', då skickar appen ett break-kommando som stänger av anslutningen och MOPED:en stannar.

4.3 ACC - Adaptive Cruise Control

MOPED:en använder sig av en ultraljudssensor för att mäta avståndet till framförvarande objekt, därefter anpassar den sin hastighet för att hålla ett optimalt avstånd. Då ultrajudssensorn frekvent gav felaktiga mätvärden skrevs en funktion för att filtrera ut orimliga avstånd. Den implementerade hastighetsregleringen var en funktion av avståndet, där ett längre avstånd resulterade i en högre hastighet. Ifall det optimala avståndet, i teamets fall 40cm, infaller skulle MOPED:en stå stilla.

4.4 Lateral styrning

För att styra efter framförvarande MOPED, används kameran tillsammans med en markör som ska sättas på den MOPED som ska följas.

MOPED:en tar kontinuerligt bilder, där varje bild genomsöks efter en markör. Om markören hittas, returneras dess laterala position. Med hjälp av denna position styrs MOPED:ens hjul i markörens riktning.

4.5 Tester

I början av utvecklingen var den första idén bland de flesta grupperna i kursen att utveckla en ACC genom att använda en Java API som innefattade att implementera plugins med hjälp av en server på MOPED:en. Dock ganska tidigt listade några grupper ut att det fanns väldigt mycket lättillgänglig Python-kod på dess huvuddator. Då många grupper redan tidigt fick en hel del problem med sina Java-servrar valde Iteration Laboratories att endast fortsätta sin utveckling genom att skriva i Python.

4.5.1 Testningverktyg

Testerna som har utförts har i första hand varit direkta tester på hårdvaran. Detta för att få en grundläggande förståelse över värden från givarna samt hur körning och styrning fungerade. När det kom till hur appen kommunicerar med en python-server på MOPED:en så kördes serven på en vanlig persondator under testningen.

Testerna på koden som skrivits har utförts genom att använda Pythons inbyggda primitiv assert, och med ett verktyg som heter pytest. Detta för att testa att metoderna som skrivits ger tillbaka det som förväntas av dem. Om testerna skulle misslyckas ger pytest tillbaka läsbara stacktraces.

Eftersom applikationen ItLabsApp är en modifiering av en annan applikation är

testerna gjorda endast på de nya klasser som kopplar denna applikation till pythonservern. Testerna som gjordes var där ACC:n sattes på och av, och värdena den skickade vid hastighetsändring och lateral styrning manuellt. Detta gjordes i Android Studios autogenererade testningsfunktion JUnit4.

4.6 Förhållande mellan process, produkt och produktägare

I denna text åsyftar processen hur arbetet genomförs, produkten det slutgiltiga resultatet och produktägare är den person eller organisation som beställt produkten. Målet med ett projekt är att produktägaren ska få en produkt den är nöjd med inom den givna tidsramen och budgeten. För att detta ska ske är det nödvändigt att processen fungerar på ett bra sätt. Produktägaren är ofta inte insatt i hur arbetet utförs och därför är inte processen intressant för denne men den vill ändå bli uppdaterad under arbetets gång för att säkerställa sig att produkten blir färdigställd till önskvärt skick och i tid. Detta möjliggörs genom att produktägaren har kontinuerliga möten med utvecklarna där de förklarar vilka funktioner som har implementerats hittills och lägger till user stories i product backlog, i övrigt är processen och produktägaren frånskilda och de sammankopplas via produkten.

5 Reflektion och diskussion

Efter och under projektets gång är det viktigt att reflektera för att verkligen få ut någonting av sitt arbete. Definitionen för reflektion som används här är 'assessment of what is in relation to what might or should be and includes feedback designed to reduce the gap'. [5]

5.1 Scrum

Tanken innan projektets början var att teamet skulle ha rollerna scrum master, fast sekreterare, MOPED:ansvarig, en grupprumsbokare och en som meddelade resten av gruppen när och var mötet skulle äga rum. Detta fördes in i teamets sociala kontrakt. Några veckor in på projektet upptäckte dock teamet att vissa ansvarsroller kunde stå kvar medan andra behövde modifieras. De roller som modifierades var fast sekreterare, grupprumsbokare och ansvarig för att meddela tid och plats för möten.

Teamet kom överens om att det var redundant med både grupprumsbokare och ansvarig för att meddela mötens tid och plats. Istället tog en person vid varje sprint planering på sig att boka grupprum inför kommande sprint samt meddela teamet på Slack. Det framkom också att planen om att ha en fast sekreterare blev tungjobbat, därför ändrades rollen till att vara flytande. Allt detta ändrades för att kunna behålla en flexibilitet i teamet, då det kan hända att någon med ansvarsroll inte alltid närvarar.

Inför sprint tre delades teamet upp i två. Detta borde ha gjorts under tidigare sprintar också, då det var många i teamet som hade svårt att veta vad de skulle göra, vilket gjorde arbetsfördelningen ojämnn. Efter uppdelningen av teamet kunde fler se tydligt hur det kunde bidra till respektive grupp och arbetsfördelningen upplevdes mer jämlig.

5.2 Nedbrytning av user stories

Gruppen följde de hänvisningar som gavs under föreläsningarna till hur scrum kan appliceras. Inför varje sprint skapades en Sprint Backlog med ett bestämt antal velocity (22 i vårt fall, då teamet va 11 medlemmar och varje medlem skulle ha två velocity var) som skulle delas ut på alla User Stories. För att gruppen skulle se att det faktiskt hände något under sprinten och att projektet gick framåt så ville gruppen ha så små User Stories som möjligt. Detta för att man ofta skulle kunna flytta saker från 'TODO' till 'Done'.

För att försöka ta fram så små user stories som möjligt kom gruppen ofta överens om ett lite större mål som kändes lämpligt att sätta upp inför kommande sprint, t.ex. "Köra eget pythonscript på MOPED:en". Det kan då delas in en mängd mindre TODOs, så som "Komma in på MOPED:ens TCU via SSH". Att först komma fram till något som gruppen ville se vara färdigt vid sprintens slut

gjorde det lättare att försöka bryta ned uppgiften i så små steg som möjligt. På det här sättet märktes det också om uppgiften skulle bli alldeles för omfattande.

5.3 Teamets process i förhållande till gästföreläsarnas processer

I ett större perspektiv var produktprocessen som beskrivs i denna rapport primärt utfört jämfört med de företag med ett väl utarbetat arbetssätt. Med det sagt betyder det inte att detta arbete inte har varit nödvändigt eller lärorikt. Dock så påvisade gästföreläsningar från Volvo Cars, Zenuity och 8 Dudes in a garage, ett mycket strukturerat och komplext arbete på olika arbetsnivåer.

Om man bortser från de uppenbara storleksskillnaderna mellan detta team och företag såsom Volvo och Zenuity, finns det fortfarande skillnader. Ett såpass äldre och moget företag som Volvo har dessutom redan en välbeprövad arbetsmetod. Likadant gäller för Zenuity, trots att det är relativt nybildat. Antalet mäniskor i vardera företag/grupp skiljde sig åt. På sådant sätt är det svårt att jämföra med detta teams process, då förutsättningarna inte var desamma.

Zenuitys arbetssätt är i teorin relativt likt detta projekt. Det finns en produktägare, PO, som har flera team där teamen skapar sina egna user stories till sprint backlogen. PO:n har dock sina krav och liknande som user stories måste rätta sig efter. Den stora skillnaden förutom projektens storlek är faktiskt arbetssättet. Zenuitys team får lov att välja arbetsmetod, allt från Scrum till någon vattenfallsmedod.

En annan skillnad med detta projekt mot Zenuity var att deras PO sa åt dem vad som skulle prioriteras i deras backlog. I vårt projekt fick gruppen istället själva välja i vilken ordning vi skulle ta oss an våra user stories. Blev inte en user story som gruppen valt i början av sprinten att ta sig an klar, gjorde inte det så mycket. Vi fortsatte istället med den i nästa sprint. Om Zenuity inte blev klar med en user story var de tvungna att jobba över till den var färdig.

8 Dudes In A Garage var ett betydligt mindre företag/start up. Där hade medlemmarna tydliga roller och ibland flera roller per person. Vilket också är svårt att applicera på denna grups process då vi var betydligt fler och med ett mindre projekt. Det var sällan flera medlemmar i teamet fick flera roller än en. Rollerna som delades ut i gruppen har redan diskuterats i denna rapport, och behandlade mer processen än själva produkten.

5.4 Tillbakablickar från sprintar

Under projektets gång har teamet haft gemensamma reflektionstillfällen där det tas upp vad gruppen tycker och tänker om hur sprinten gått. Detta har dokumenterats och sammanställts.

Sprint ett

Scrum var för alla medlemmar i gruppen främmande och i ett försök att förstå vad det innebar lästes flera kursmaterial igenom. Tillvägagångssättet inför planeringen av sprint etts backlog blev då aningen bristfällig. Vaga formuleringar så som 'Läs om plug-ins' användes. Som konsekvens av detta blev aldrig user stories definierade som 'Done' i vår Trello, eftersom formuleringen inte tydligt beskrev vad som faktiskt innebar 'Done'. Det fanns alltså ingen tydlig 'Definition of Done' gemensamt bestämt i gruppen, vilket i sin tur ledde till att scrum boarden gav intrycket att inget hade åstadkommits under första sprinten. Trots att det hade arbetats inom flera områden hade dessa arbeten inte mätts och därför inte blivit done. Därför hade tydligare och mer konkreta uppgifter behövts finnas i sprint backlogen, något som implementerades i nästkommande sprintar.

Visuell data över gruppens KPI under sprint ett, se Figur 1 i appendix.

Ett annat problem som fanns i sprint backlogen var att redan den första sprintens backlog var för specifik om hur en uppgift skulle lösas. Ett exempel ur sprint backlogen i sprint ett: 'As a group we want to be able to write plugins to the TCU'. Den meningen medförde problemet att uppgiften enbart fick lösas på ett sätt. Man var alltså tvungen att skriva plugins och inte bara ren kod till MOPED:en, vilket märktes att det var mycket lättare och smidigare. Denna lösningen visade sig på grund av bristande kunskap kring ämnet vara enormt tidskrävande för gruppen och icke genomförbar under en sprint. Alltså borde formuleringen undvikta att specificera enbart ett lösningsförslag. Dessutom borde grundproblemet med uppgiften ha poängterats tydligare. 'Varför vill teamet skriva plugins?' För att kunna skriva kod och i sin tur få MOPED:en att bete sig på det önskvärda sättet.

Det fanns alltså många user stories och i teorin många saker att göra första sprinten, men när sprinten var slut såg det ut som att det inte åstadkommits något, mycket på grund av de dåligt formulerade user stories och dess brist på 'Definition of Done'. Istället för user stories bundna till tekniska tillvägagångssätt hade de istället kunnat skrivas som 'Som en grupp vill vi köra fungerande kod på MOPED:en för att längre fram kunna implementera ACC'. Det hade resulterat i att man inte är bunden till att få just java plug-ins att fungera, utan istället hade alla möjligheter som löst problemet varit ett accepterat tillvägagångssätt.

Eftersom det var mycket ny information att bekanta sig med under sprint ett däribland API:er samt osammanhangande koddokumentation, hade en bestämd tid för inläsningen kunnat sättas för att få bestämt en konkret 'definition of done'. Då hade det synts tydligt att teamet suttit och läst och då också synts att de åstadkommit något under sprinten. Teamet hade också kunnat försöka bryta ner sina user stories mer, för att de inte skulle uppfattas som aningen abstrakta. Då hade det kunnat bli lättare att uppskatta hur lång tid varje skulle ta och vad som faktiskt skulle göras för att de skulle uppfattas som klara.

Sprint två

Inför sprint två hade teamet ett långt möte med reflektion över hur första sprinten hade gått. Gruppen kom överens om nya metoder som skulle användas framöver. Teamet bestämde att ha lösare formuleringar kring tillvägagångssättet med större fokus på vad det egentligen är som ska uppnås; se till att skriva kortare user stories och att skriva konkretare 'Definition of Done'.

Att skriva kortare och tydligare user stories klarade gruppen av bra, men teamet var fortfarande lite för bundna till tekniska tillvägagångssätt. Vi skrev bland annat att vi vill använda oss av SSH och 'eventuellt' FileZilla. Vi upptäckte sedan tidigt att det skulle bli för tidskrävande att lära sig FileZilla när vi lika gärna bara kunde använda oss av Git, som vi redan kan, vilket gjorde att hela den uppgiften försvann. I övrigt var sprint två en stor förbättring i jämförelse med sprint ett. Vi uppfyllde hela vår velocity som var utsatt för sprinten och tydliga framsteg hade åstadkommits.

Visuell data över gruppens KPI under sprint två, se Figur 2 i appendix.

Sprint tre

Efter sprint två kunde gruppen se att vad som behövde göras var av tydligt olika karaktär och gruppen skulle kunna delas i två. På så sätt skulle gruppen kunna bli mer effektiv. Det bestämdes att ena halvan av gruppen jobbar med ACC och den andra halvan med appen. På det sättet blev det lättare att sys-selsätta flera och kunna jobba parallellt med varandra. Efter uppdelning av den ursprungliga gruppen bestämdes det att vardera gruppen skriver sina egna user stories eftersom grupperna vet olika mycket om de olika områdena.

Uppdelningen fungerade bra och vardera grupperna uppfyllde nästan sin satta velocity. Varför inte all velocity uppfylldes berodde på en skillnad i 'Definition of done' mellan våra user stories och tasks. Definition av klar på några user stories stämde inte helt överens med definitionen på vissa tasks. Som konsekvens var tasks färdiga men huruvida deras sammanhangande user stories var det var oklart. Detta antydde att gruppen fortfarande inte riktigt varit tydlig med ett 'Definition of done' specifikt i våra user stories. Dessutom uppstod problem då inte alla i gruppen hade tillgång till SD-kortet och därmed inte kunde testa ko-

den för ACC. Det här gjorde att mycket kod var skriven men inte testad inför produktägarmötet. Ett sådant tekniskt problem borde kunnat förutses redan i början av projektet då hela arbetet inte borde stanna upp på grund av att en gruppmedlem inte är närvarande, likt den så kallade bus faktorn [6]. För att undvika det borde det bestämts att SD-kortet alltid ska sitta i MOPED:en, då den alltid varit på campus.

Visuell data över gruppens KPI under sprint tre, se Figur 3 i appendix.

Sprint fyra

Eftersom uppdelningen av gruppen var uppskattad och visat resultat fortsatte vi med den även under sprint fyra. Det blev uppenbart att vardera gruppen kunde dela på sig ännu mer. ACC-gruppen delades upp i undersökning av kameras och longitudinell navigering. App-gruppen delades upp i utveckling av appen och rapportskrivande. Det här fungerade bra då alla hade tydliga mål att jobba mot. Det fortsattes även att skriva user stories i två grupper då det fungerade bra tidigare sprint.

Problemet under sprint fyra blev dock ett logistiskt problem med laddare av MOPED:ernas batterier. De grupperna som för tillfället hade laddare var inte tillgängliga via kursens team på Slack. Det här gjorde att gruppen återigen hamnat i en position där vi inte kan testa kod, vilket gjorde att en av gruppens user stories inte blev uppfyllda. Problemet togs upp under scrum of scrums och löstes genom att när en grupp har en laddare skriver den gruppen det i kursens slack-team, och ser till att vara allmänt mer aktiva där genom att bland annat sätta på notifikationer under veckodagar.

Visuell data över gruppens KPI under sprint fyra, se Figur 4 i appendix.

Sprint fem

Under sprint fem fortsatte uppdelningen av teamet. User stories skapades dock i hela gruppen då app-gruppen upplevde att det inte fanns mycket kvar att göra förutom att snygga till GUI. På grund av det här blev rollerna i teamet återigen oklara, då mycket av produkten började bli färdigställd. Det gjorde att många inte hade mycket att göra under sprinten.

För att åtgärda detta borde fler av teamets medlemmar börjat ägna sig åt att implementera en fungerande platooning, då detta fram tills sprint fem varit nedprioriterat. Fram tills sprint fem har teamet fokuserat främst på att få fram en bra ACC. Istället borde det ha utgåttifrån att ACC kommer att fungera, så att det även prioriterats att fokusera på nästa steg; platooning.

Visuell data över gruppens KPI under sprint fem, se Figur 5 i appendix.

5.5 Sprint reviews

Som tidigare nämnt har gruppen varit relativt ojämnn i arbete som visat resultat och inte alltid haft möjlighet att testa sprintarnas arbete ordentligt. En följd av detta har blivit att gruppen inte alltid haft något konkret att visa upp under sprint reviews.

Varje sprint review har dock givit gruppen mycket kunskap. I början fick teamet hjälp och tips att förbättra vårt sätt att använda scrum samt tips om sätt att lösa uppgiften utan en server. Det har också varit mycket lärrikt att se vad alla andra grupper gör, för att sedan kunna byta information med varandra om hur alla löst sina respektive problem. Det här har gjort att gruppen känts motiverad och inspirerad inför nästa sprint, då många nya idéer och tankar väckts.

Sprint reviews har även varit ett bra sätt att kommunicera fram vad produktägare tycker om gruppens nuvarande produkt och planer för nästa sprint. På grund av att alla grupper samlats och haft gemensamt samtal med produktägare får också alla grupper samma chans för frågor samt samma information om slutmål. Det har gjort att det inte råder skilda meningar mellan grupper om vad det är alla jobbar mot.

Sammanfattningsvis hade gruppen önskat att vi haft mer att visa under varje sprint review. Mötena har dock inte varit förgäves då de alltid inspirerat oss och vi kommit fram till sätt att jobba mot vårt slutgiltiga mål.

Innan projektet började hade gruppen 200 i velocity för en sprint. Detta för att representera 20 timmar per person, som var tio vid tillfället. När velocity och effort börjades sättas på våra user stories och tasks följdes inte detta längre då det var svårt att uppskatta så exakt hur lång tid en task skulle ta. På grund av detta blev det otydligt vilket system gruppen skulle använda samt hur mycket och vad en specifik siffra betyder. Istället bör teamet ha kommit överens om ett nytt system för velocity som alla i gruppen förstod och kunde använda sig av. Som det är just nu är gruppens mätta velocity lite missvisande.

5.6 Reflektion över KPI:er

I diagrammen som sammanställer gruppens KPIer bredvid varandra sprint för sprint (se figur 1-5 i appendix) utläses att närvaren har legat runt 70 procent under projektets gång och att motivationsnivån har varit kring 60 procent under varje sprint. Hur mycket arbete som utförts pendlar betydligt mer, det värdet har som lägst varit nere på 25 procent och som högst varit uppe på 100 procent. I och med detta dras slutsatsen att motivationsnivå och närvare inte haft någon större inverkan på hur arbetet har förefallit, vilket antogs när KPIerna sattes upp. Då förmodades att en hög närvare skulle resultera i en mer motiverad grupp och en större mängd arbete avklarat. Vice versa skulle en mer motiverad grupp leda till högre närvare och mer arbete gjort. KPIerna har alltså inte fungerat som tänkt.

Mängden arbetstimmar har uppskattningsvis varit lika stor under sprintarna. Vad beträffar ineffektiva och effektiva arbetstimmar har ingen åtskillnad gjorts, utan båda räknas in under begreppet arbetstimmar. Trots det varierar stapeln som visar hur mycket arbete som utförts kraftigt mellan de olika sprintarna. Att stapeln efter en lång arbetsvecka visar att endast 25 procent av arbetet är utfört (som i sprint ett) ses som mer stötande än motiverande för gruppen. När samma prestation ibland belönas för att i nästa vecka ses som ett misslyckande skapas en förvirring. Förvirringen skapades främst av det sätt som arbetseffektivitet mättes då det inte var särskilt optimalt. Arbete räknades som klart ifall en user story kunde markeras som avklarad. Det ledde i sin tur till att om en user story var undermåligt skriven i den mening att den t.ex. hade en för specifik eller ospecifik beskrivning kunde den aldrig markeras som klar. KPIIn velocity mätte således inte främst gruppens arbetsinsats, utan snarare gruppens förmåga att skriva en väldefinierad backlog. Ett namnbyte av KPIIn hade varit på sin plats för att slippa att gruppen skulle känna sig misslyckad. Ett sådant namnbyte hade möjligen också lett till att gruppen tidigare såg att felet låg i att backloggen var dåligt skriven. Vilket i sin tur hade resulterat i att gruppen la mer energi på att skriva en mer väldefinierad backlog.

Ytterligare en anledning till att stapeln med utfört arbete har varierat så mycket i mängd under de olika sprintarna kan härledas till att en velocity-siffra sattes på våra user stories först efter att alla user stories för sprinten var skrivna. Efter att en user story hade fått ett värde sattes sedan värden till dess tasks. Således hade gruppen en totalsiffra för varje sprint som våra user stories skulle uppfylla. Vilket i sin tur ledde till att gruppen såg till att user stories och tasks blev tillräckligt mycket/lite värda för att passa in. Det betyder att arbetsmängden kan siffermässigt se lika stor ut varje vecka, men den verkliga arbetsbördan kan variera. Bättre hade det varit ifall en user story i taget hade skapats, för att sedan delas upp i flera olika tasks. Efter det borde gruppen ha satt värden på tasksen som sedan tillsammans ledde till ett värde på user storyn. Därefter skulle backloggen ha fyllts på tills arbetsbördan motsvarade det värde gruppen tyckte var en rimlig arbetsmängd för veckan.

Hur väl gruppmedlemmarna kände till vilken uppgift var och en skulle utföra och hur svår den var i förhållande till den personen förmodades ligga till grunden för motivationsnivån. Vilket stämde till viss del, t.ex. var gruppen mest motiverad dagen efter att sprint ett hade tagit slut, fredagen den 15/9. Under sprint ett hade gruppen haft problem med att sätta upp en server under hela veckan, men sista dagen på sprinten under scrum of scrums-mötet på Lindholmen framkom det att en möjlighet var att koda genom Phyton istället och på det sättet slippa sätta upp en server. Vilket resulterade i den höga motivationsnivån eftersom gruppen nu löst ett stort hinder som hindrat dem från att veta hur de skulle gå vidare. Den fortsatta arbetsgången blev nu tydligare.

Problemet med KPI:n motivationsnivå var att ingen i gruppen riktigt visste vad som skulle vägas in när en svarade på hur motiverad en var. Det fanns ingen tydlig riktslinje att endast vissa faktorer skulle vägas in när en svarade på den frågan. Det innebär att en hög motivationsnivå kan reflektera hur bra man trivs i gruppen, hur mycket man sovit under natten eller hur vädret är ute. Ju längre projektet pågick desto mer roll spelade de senare punkterna in. T.ex. yttrades meningar som ”Det regnar ute idag, så jag är -1”. I och med det visade inte KPI:n det gruppen först hade tänkt att den skulle göra. Regn kan alltså påverka KPI:n motivationsnivå negativt, men KPI:n som handlar om arbetsprestation positivt då det regniga vädret ledde till att personen satt inne och arbetade istället för att gå ut. För att förhindra detta borde gruppen ha varit tydliga med att motivationsnivån är kopplad till projektet och inte yttra faktorer. Vilket såklart är svårt, så istället hade vi kunnat ställa frågan ’hur motiverad är du att utföra den uppgift du har framför dig just nu?’.

Utebliven närvaro på mötena behöver inte gå rakt ut över arbetsinsats eftersom det i sig inte innebär att man inte arbetar hemma istället eller senare på dagen. Men det kan göra det, eftersom utebliven närvaro resulterar i att det blir svårare att få hjälp av andra gruppmedlemmar och att veta vad de andra arbetar med. Alla våra möten har legat schemalagda på morgonen, vilket har lett till att många inte orkat ta sig till skolan i tid eller inte kommit alls. En irritation bland de som är på morgonmötena och dem som inte är det har under projektets gång vuxit fram. Vilket i sin tur leder till en lägre motivationsnivå i gruppen. När gruppen utvärderar dessa morgonmöten finns en antydan till att visst engagemang saknats under mötena. Trots det tycker majoriteten att mötena har varit bra, även om en del möten gärna fått legat senare på dagen. En möjlighet för att ha fått bättre sammanhållning inom gruppen hade varit om de som inte kom på mötena hade skrivit på Slack vad de arbetade med. ’Mötena har tvingat mig till att jobba med projektet och gjort så att jag bidragit mer’ är en mening som majoriteten av gruppmedlemmarna håller med om. Vilket antagligen ligger till grunden för det bra betyg alla sedan gav morgonmötena.

6 Slutsats

I efterhand kan man tydligt se vad vi ville, var vi är och vad gruppen borde ha gjort för att komma dit. Ett genomgående problem är att gruppen varit mindre bra på att använda vår KPI velocity och ha ett gemensamt 'Definition of done'. Det här har lett till att missförstånd ofta dykt upp i slutet av sprintarna när det visat sig att alla inte delat samma åsikt om vad en user story innehåller och hur den ska uppfyllas. Vad gruppen hade kunnat göra för att åtgärda detta är att ha kommit överens om en 'Definition of done' i början av projektet. På så sätt hade det kunnat bli lättare att komma överens om vad en user story innehåller och hur alla kan se att den uppfyllts. Ett exempel på ett 'Definition of done' hade kunnat vara att göra fler enhetstester som måste gå igenom innan en task kan ses som 'done'. Istället har gruppen haft ett vagt 'när det fungerar' som 'Definition of done'.

I slutskedet av kursen insåg vi att vi borde ha fokuserat på kameran och lateral autonom körsättning tidigare genom att t.ex. ha en tredje grupp. Ett annat problem vi hade var att ett fåtal personer var mer insatta i python-koden än de andra vilket ledde till att det i slutet av projektet var svårt för dem att få hjälp av folk som hade jobbat med andra uppgifter som de var klara med. Detta hade kunnat lösas genom att tidigare i kursen bryta ner slutmålen i mindre från varandra oberoende delmål samt se till att flera hade kameran som ansvarsområde.

Gruppen har efter projektet fått lärdom om hur agila arbetsprocesser går till. Tidigare erfarenheter av grupperbeten skiljer sig väldigt mycket från detta projekt, då gruppen nu fått möjlighet att helt själva lägga upp arbetet som de vill. Det har också uppmuntrats att samarbeta över gruppgränser och alla sätt att ta till sig information har tillåtits.

I grupperbeten tvingas medlemmarna att ta hänsyn till de andra medlemmarnas förutsättningar och tidigare kunskaper. Det är lätt att bara fokusera på sin egen arbetsinsats och missa hur helheten blir till. Trello har hjälpt oss att se hur små uppgifter som individuella gruppmedlemmar har löst binds samman till att bli lösningar av ett större problem, som ingen hade kunnat lösa själv på egen hand.

I början av projektet arbetade vi som vi alltid tidigare hade gjort. Vi valde en uppgift som skulle lösas och hur den skulle lösas. Den person som senare fick den uppgiften kände sig då redan från början begränsad att lösa uppgiften på det sätt som hade bestämts. Redan där begränsas tänkandet hos medlemmarna och gruppen låste sig till att lösa problemen på ett bestämt sätt. Att enskilt få känna frihet och självbestämmande över sitt arbete har visat positiv inverkan hos personer, vilket är motsatsen till om gruppen från början bestämmar hur enskilda gruppmedlemmar ska lösa en uppgift. Förståelse för detta växte under projektet fram hos alla medlemmar och ansvarat för hur en uppgift skulle lösas

flyttades över från gruppen till enskilda personer.

Vikten vid att strukturera upp kod och skriva vettiga kommentarer blev tydlig när vi fick tillgång till MOPED:ens repository på GitHub. Mycket tid gick åt till att tyda metoder som låg där. Tidigare har inte så många i gruppen haft erfarenhet av att ärva så stora projekt och av den anledningen inte förstått varför det är så viktigt.

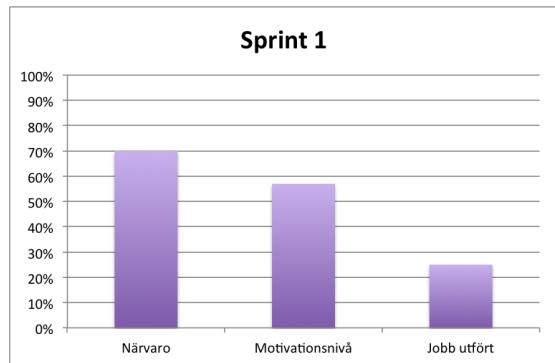
Som helhet känner vi oss nöjda med det vi har åstadkommit, både produktmässigt och processmässigt. Alla har bidragit efter sin förmåga, och det har varit spännande att få dela erfarenhet mellan sektionsgränserna.

7 Referenser

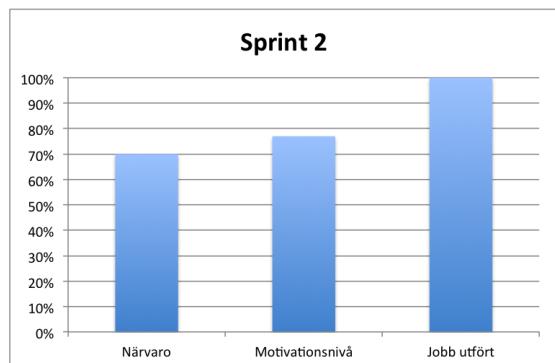
Referenser

- [1] J. Wong, “How will automation affect society.” [Online]. Tillgänglig: <https://www.weforum.org/agenda/2015/01/how-will-automation-affect-society/>, Jan 2015. Hämtad: 17 Oktober, 2017.
- [2] Volvo, “Driving automation is driving progress.” [Online]. Tillgänglig: <http://www.volvotrucks.com/en-en/about-us/automation.html>, 2015. Hämtad: 17 Oktober, 2017.
- [3] P. Avery, “Introduction to pid control.” [Online]. Tillgänglig: <http://www.machinedesign.com/sensors/introduction-pid-control>, 2009. Hämtad: 18 Oktober, 2017.
- [4] ScrumInc, “Velocity.” [Online]. Tillgänglig: <https://www.scruminc.com/velocity/>, 2017. Hämtad: 18 oktober, 2017.
- [5] R. Smith, “Formative evaluation and the scholarship of teaching and learning. new directions for teaching and learning, vol. 88.” Book, 2001. pp. 51-62.
- [6] M. Bowler, “Truck factor.” [Online]. Tillgänglig: <http://www.agileadvice.com/2005/05/15/agilemanagement/truck-factor/>, Maj 2005. Hämtad: 13 Oktober, 2017.

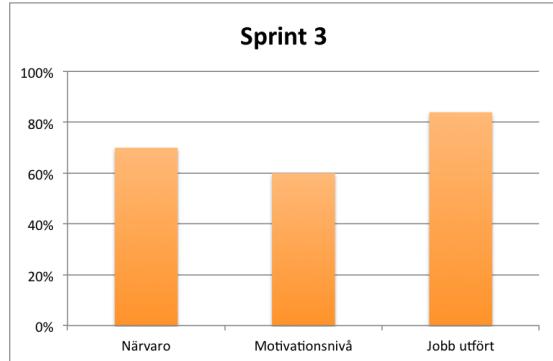
8 Appendix



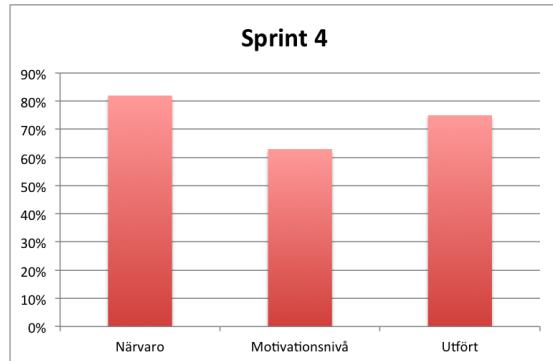
Figur 1: Sprint etts KPI-mätningar. Värt att notera är att gruppens taggnivå endast är baserat på de som närvarat vid veckans Daily Scrums.



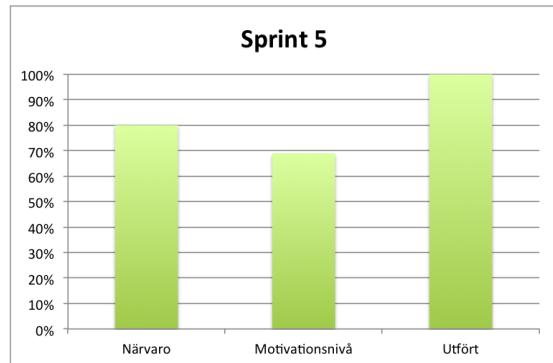
Figur 2: Sprint tvås KPI-mätningar. Värt att notera är att gruppens taggnivå endast är baserat på de som närvarat vid veckans Daily Scrums.



Figur 3: Sprint tres KPI-mätningar. Värt att notera är att gruppens taggnivå endast är baserat på de som närvarat vid veckans Daily Scrums.



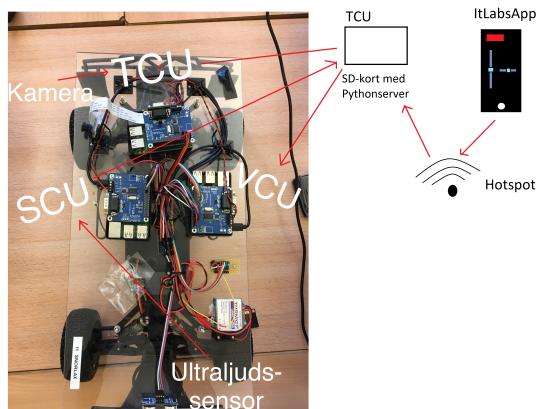
Figur 4: Sprint fyras KPI-mätningar. Värt att notera är att gruppens taggnivå endast är baserat på de som närvarat vid veckans Daily Scrums.



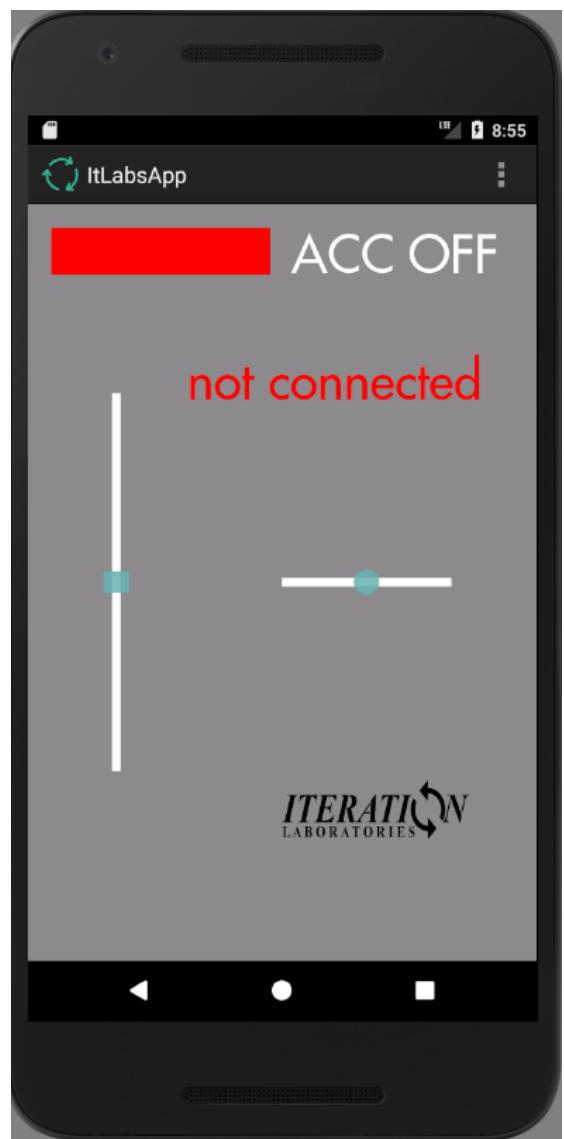
Figur 5: Sprint fems KPI-mätningar. Värt att notera är att gruppens taggnivå endast är baserat på de som närvarat vid veckans Daily Scrums.



Figur 6: Ett exempel på en Sprint Backlog i applikationen Trello, med olika tavlor såsom TODO, Under Progress och Done.



Figur 7: En grov förklaring på hur MO-PED:en ser ut och dess komponenter.



Figur 8: Iteration Laboratories applikation för MOPED:en.