Intro.

I sprint 2 blev der udviklet en god del på systemet. Vi fik skrevet vores første ordentlige tests, som tester på data i databasen. Vi fandt ud ad, at der i forbindelse med at kode en task, nemt kan opstå

uforudsete problemer, der også tager tid, og at man så man må beslutte hvordan denne ekstra forbrugte tid skal håndteres og registreres i arbejdsprocessen.

I dette sprint var der primært fokus på kodning og udvikling. Mange af de måder vi havde arbejdet på i sprint1 fortsatte vi med i dette sprint, da der via vores sprint retrospective for sprint 1 ikke var indikation for at lave store ændringer i vores arbejdsprocess. Vi havde ved sprintets start stadig to user stories fra sprint 1 som ikke var færdige. Derfor var de de første vi arbejdede på i dette sprint.

Vi begyndte i løbet Sprint 1 at tale om, om det kunne betale sig, udover at brænde story points, at prøve at brænde timer på et særskilt burn down chart, og vi havde også overvejelser om, om vi skulle prøve det i sprint 2. Vi besluttede at prøve det i sprint 2, og lavede derfor et burn down chart for timer, i håb om at vi kunne spore mere nøjagtigt, hvad vi fik lavet færdigt hvornår.

Ved sprintets start havde vi sprint planning meeting. Her flyttede vi i dialog med product owner de user stories der havde højest prioritet fra product backloggen over i sprint backlog. Vi stod efter dette med en omprioriteret ny version af vores product backlog. Efter dette lavede vi planning poker for, at estimere alle user stories for sprint 2 og nedbryde dem i tasks. Det er vigtigt at nævne, at vi ikke justerede vores burn down chart, selvom vi havde to ufærdige user stories fra sprint 1 med i sprint 2. Dog justerede vi i denne sprint planlægning vores velocity med udgangspunkt i, at vi ikke havde nået alle user stories i sprint 1. Derfor justerede vi vores velocity ned fra 12.5 til 10.0.

I planlægning af sprint 2 havde vi ikke taget højde for, at det meste af mandag vil gå til andre ting end udvikling, dermed mister vi omtrent en dag. Mandag var der både sprint review for hele klassen samt sprint setrospective internt i gruppen. Derudover var der sprint planning meeting for sprint 2. Da vi lavede burn down chart for sprint 2 påførte vi 5 dage til udvikling, og dette resulterede i, at vi havde sat en dag på til udvikling, som vi reelt ikke havde, fordi den var tildelt planlægning af sprintet. Dette betyder i praksis, at det bliver svært, at nå det arbejde der er estimeret for sprintet. Vi nåede heller ikke, at brænde alle de story points vi skulle i sprint 2. Dette gælder begge vores burn down charts.

I sprint 2 blev der hver morgen afholdt daily scrum klokken 9:00. Scrum møderne fungerede efter hensigten, og blev en hjælp til at holde et fælles overblik. Dels over hvad der blev arbejdet på, og dels hvad problemer, hver udvikler stod med, indenfor enkelte tasks. Dette var en stor fordel, da det gav alle i gruppen indsigt i, hvad der kunne være af potentielle problemer i dele af systemet, man ikke selv direkte arbejdede på. På denne måde kunne morgenmøderne bidrage til, at alle i gruppen opnår en mere solid forståelse, af det system man er sammen om at udvikle på. Som udgangspunkt havde alle altid en ny task med fra morgen mødet. Dog med undtagelse af, at man var i gang med en nuværende task som endnu ikke var færdig, til at komme i to verify. Og i praksis er det også svært, at ramme så man bliver færdig med en task ved gårsdagens afslutning, så man er klar til at tage en ny ved hver daily scrum. Tasks kan også godt være estimerede til at tage mere end en arbejdsdag.

Vi talte i løbet af sprintet om, hvorvidt vi skulle prøve, at lave en form for rotation i arbejdet, så alle prøvede at arbejde indfor alle forskellige områder af projektet. Dette var for at undgå, at vi hver især blev for domænespecifikke med hver vores viden om vores pågældende område. Dette ville også have indflydelse på, hvorvidt alle vil være i stand til at ændre på kode alle steder i systemet, jævnfør den af de 12 XP praktikker omkring 'fellow code ownership'. Vi besluttede ikke noget specifikt for dette sprint omkring det, men begyndte i større grad end tidligere at arbejde således, at alle arbejdede på flere forskellige slags ting i systemet end tidligere. Dette vil antageligt også være hensigtsmæssigt, da der i senere sprints vil være et tiltagende behov for, at alle udviklere kan begå sig i alle hjørner af systemet.

Angående extreme programming-praktikker anvendte vi ikke alle de 12 eksisterende i dette sprint, men derimod dem vi på daværende tidspunkt synes der gav mening. Vi besluttede, at undlade at anvende pair programming. Det fungerede i princippet fint, da det gav alle 4 i gruppen mulighed for at sidde med individuel kodning. Dog påvirkede det vores arbejdsprocess. Som konsekvens var det nu en mere manuel udfordring, at få delt kodeviden og opnå fælles forståelse. En anden konsekvens er, at det nemt kan ske, at der i denne situation bliver arbejdet på flere user stories, end hvis man sidder i par. Dette betyder, at der arbejdes parallelt på flere stories, end man ellers ville. Og dette kan afspejle sig i, at man ikke får brændt så mange story points per løbende dag på burn down chartet, som man ellers hvis man havde arbejdet på få user stories ad gangen. Ovennævnte arbejdsgang vil resultere i, at flere user stories der er blevet arbejdet parallelt på, ofte vil være færdige på samme tid, og derfor kan indgå i den samme brænding. Derfor ses der tendens til store knæk på vores burn down sidst i sprintet.

Sprint 2 bar præg af, at vi undervejs i sprintet opdagede, at vi manglede tasks på flere user stories. Vi indså i dette sprint, at man kan komme til at arbejde på, eller blive tvunget til at arbejde på noget nyt der skal løses, før man kan komme videre med den egentlige task man er igang med. Vi valgte i denne situation, at lave nye tasks, til de ekstra opgaver der skulle bruges tid på, eller som vi havde brugt tid på. Dette er vigtigt, da tiden skal registreres til den task man arbejder på, og for at undgå at tiden skrider på en aktuel task, fordi man lægger andre opgaver ind under den. Dette har også noget at gøre med erfaring i, at kunne se hvad der reelt ligger i en given task. En anden ting vi opdagede i dette sprint var, hvor vigtigt det er at kommunikere undervejs i arbejdsprocessen, for at sikre en fælles forståelse af den ønskede funktionalitet. Et eksempel på dette var, at på vores updateuser profile funktionalitet. hvor systembruger kan opdatere oplysninger på sin brugerprofil, passede de input felter der er i brugerfladen ikke med de attributter der er på en user i databasen. Dette er relevant fordi det tager tid at lave om, og sandsynligvis vil gøre det sværere at holde sig indenfor taskens estimerede tid.

I dette sprint fandt også vi ud af, at det var praktisk at notere numre på alle tasks, der indikerer hvilken user story de hører til. Dette var en hjælp da vi i kraft at undlade at pair programme, ofte arbejdede på to eller flere user stories ad gangen. Når vi estimerede user stories ved sprint planning meeting og nedbrød dem i tasks, estimerede vi også timer til hver task. Efter hånden som vi arbejdede os gennem tasks, noterede vi også forbrugt tid på hver task. Dette var for at blive bedre til, at overskue hvor lang tid en given task vil tage.

En gennemgående del af dette sprint var, at lave test-first. De tests der blev implementeret i dette sprint, adskiller sig tidligere tests ved, at der ikke længere testes på simpel data, men på data fra databasen som hører til systemet. Dette blev lavet tests til stort set alle user stories. Vi fandt dog ud ad, at hver gang der var blevet opdateret noget et sted i systemet, påvirkede det i mange tilfælde vores tests, så de tests der hidtil havde fungeret, ikke fungerede længere. Dette resulterede i, at vi var nødt til at omskrive vores tests, så de passede til de ændringer der var lavet. Man må antage, at dette medførte overflødigt arbejde i et eller andet omfang, der ikke er dokumenteret i vores timeregneskab. Dette kunne undgås ved eventuelt at vente med, at skrive testen indtil man er sikker på, hvad det er der skal testes for. Men igen kan det være en del af agil udvikling, at der skal foretages ændringer der kræver nye test scenarier.

Under sprint 2 opdagede vi, at når vi udviklede og blev færdige med user stories, kom vi til at stå med enkeltstående søjler af funktionalitet. Vi udviklede søjler der virker som enkeltstående elementer, og der opstår her en udfordring omkring, at få disse adskilte spor af funktionalitet koblet sammen og integreret med hinanden. Dette ses første gang i dette sprint. Dette kommer af, at vi ikke fra starten af, og i forbindelse med sprint planning meeting, havde gennemskuet om en given funktionalitet skal ligge det ene eller det andet sted. Et eksempel på dette problem og løsningen af det, var da vi skulle implementere den funktionalitet, der skulle tjekke om en systembrugers experience points skulle opdateres, efter at han havde indtastet en træningssession. Der var skrevet user story til hele denne funktionalitet, som om den var en søjle for sig selv. Men vi endte med at implementere denne funktionalitet, på den søjle af funktionalitet der allerede eksisterede for user, og det giver god mening, da det er for en specifik user, at experience points skal udregnes, ligesom experience points også er en attribut der ligger på en user i systemet.

Sammenfattende:

Det kan ses på vores burn down for sprint 2, at der ikke er brændt noget de første par dage. Dette korresponderer fint med der dels var to user stories vi allerede var bagud med fra sprint 1, og at vi dels havde kalkuleret med at have hele mandag, altså 4,5 dage. I virkeligheden havde vi kun 3.5 dage, da hele mandag gik til sprint review, sprint retrospective og sprint planning. Dette forklarer kombineret med, at vi ofte arbejdede på mange user stories parallelt, at vi først får brændt story points på vores burn down sidst på ugen. Angående tests kan man eventuelt optimere workflowet, så man kommer til, at skulle skrive de samme tests færrest muligt antal gange.