## Karar Al-Remahy 2021-12-22

Det är mycket som är positivt med den agila metoden, och detta är eftersom det är en filosofi. Man kan göra på annorlunda sätt än SCRUM och ändå följa filosofin som agile förmedlar. För det handlar mer om hur man tänker under arbetets gång.

Uppgiften fick mig att uppleva standups, mobb- och par-programmering samt sprint reviews och retros. Dessa moment var nyttiga för mig då jag fick en mer praktisk bild istället av att försöka lära mig och komma ihåg allt det teoretiska.

Vår grupp var liten, vilket gjorde att vi tålde mobb-programmeringen. Vi hade problem i början med tysta åskådare men vi löste det så småningom. För den delen var jag aldrig den tysta åskådaren för jag har tagit mig an en mentorsroll sedan början av andra kursen eftersom vi har haft problem med lärare som inte var bra. Dessutom har jag mer erfarenhet av Bootstrap vilket också innebär att jag förklarade saker när något relativt svårt behövdes göras till mockupen så för min del var mobb-programmering bara en positiv. Vare sig jag var driver eller i baksätet fungerade det bara bra. Men jag förstår att folk vill spendera tid med koden själva, därför hade par-programmering fungerat bättre. Att vi istället delar upp arbetet så att ett par gör en task medans en annan grupp gör en annan task. Men denna optimering är nog för mycket och det var bra som det var - För alla var på samma sida och tittade på eller skrev samma kod - Blir någon sjuk kan resten av mobben fylla det tomma sätet. Detta är nog anledningen till varför jag tror par-programmering är vanligt i arbetslivet, för om den ena är sjuk kan den andra täcka för - För båda parterna vet vad som ska utföras.

Den agila processen är inte så svår att förstå nu efter att jag har testat mig på det med hjälp av SCRUM och webbshop-uppgiften. Det börjar med en planeringsskede där man går igenom personas, user stories och få till tasks att lägga till i exempelvis en kanban tavla. Dessa lade vi in i kanban tjänsten Trello. Vid varje planering tar vi våra tasks och bestämmer vilken som ska utföras i sprinten och lägger sedan valen i exempelvis kolumnen “Doing”. Man kanske delar upp arbetet och ser till att alla har något task att utföra under sprintens gång. Under arbetet kanske man kommer på flera user stories som man tidigare inte hade föreställt, dessa kan enkelt läggas in i backloggen och tas upp i en senare sprint. Vid avslutat sprint börjar review processen, vi beskriver det vi har hunnit bli klara med. Efteråt kommer sprint retrospective, där går vi igenom saker som gick bra, mindre bra, vad vi bör sluta göra och vad vi bör göra till nästa sprint. Alltså är retrospectiven viktigt för att kunna förbättra gruppens hållning till den agila processen och metoden som i detta fall är SCRUM.

## På tal om Vattenfallsmodellen

Det finns fördelar med vattenfallsmodellen, exempelvis med att det är ett simpelt hierarki, det börjar med A och slutar med F. Kunden kan under varje del kontrollera och besluta om exempelvis man ska börja om eller om man ska lägga ner projektet. Om projektet exempelvis läggs på is, har Vattenfallsmodellen krävt att deltagarna skrivit dokumentation som gör det enklare att tina upp projektet.

Det största problemet med Vattenfallsmodellen är att man utför arbete tidigt i arbetet som kan göra att det blir problem längre in i arbetet. Exempelvis om testare hittar buggar kan hela projektet bli skjuten tillbaka och skapa ytterligare arbete och förseningar. Om man vill ändra på något måste förändringen gå igenom varje process vilket inte är lika effektivt som agila arbetssätt som Scrum.

Många av problemen med Vattenfallsmodellen blir lösta genom Scrum. Scrum mötena man har dagligen och sådant som hamnar i backloggen kan bli fixade så småningom. Scrum metoden är också nyttig i att alla har varsitt jobb för dagen, något Vattenfallsmodellen har problem med för man utför resurs-utdelningen bara i början av delen, inte varje dag som Scrum modellen gör.

Ett ytterligare negativt med Vattenfallsmodellen är att man kan bara få en överblickande vy i pauserna början, mellan samt slutet av varje del, och inte under pågående arbetet men i Scrum kan man se hur arbetet fortskrider dagligen och kan rätta till om det behövs, för arbetet delas ut för 24 timmar för ett sprint som tar exempelvis 2-4 veckor. De dagliga mötena hjälper med detta.

Ännu ett negativt med Vattenfallsmodellen är att om tar för stora beslut måste dessa beslut tas innan utvecklingens gång, annars måste projektet skjutas tillbaka för att gå igenom hierarkin igen.

I Scrum kan dessa beslut tas efter en sprint är klar, eller till och med under utvecklingens gång för man riskerar inte att slänga bort lika mycket arbete som Vattenfallsmodellen. Det är även lättare att börja om med en sprint tack vare ovanstående argument.

## Utöver skoluppgiften har jag också gjort kanban projekt själv utan att jag visste om det.

Jag har ett projekt som sitter på <https://befriendlier.app/> och mitt problem var att jag inte visste vad jag skulle jobba på efter att jag var klar med ett steg, vilket ledde till att inte mycket blev klart eller att jag tappade bort min motivation. Jag använde mig senare av kanban, eller det jag idag vet kallas för kanban och då blev min process smidigare eftersom jag höll både koll på vilka buggar min kod hade och behövdes fixas och även vad som behövdes bli klart.

<https://github.com/users/KararTY/projects/1> Idag använder jag mitt egna “kanban struktur” men den behöver också lite finslipning för jag har saker lite överallt och inte allt på ett ställe, d.v.s på kanban-board för vissa anteckningar ligger på anteckningsblock eller så har jag inte skrivit ner dem någonstans utan det sitter en kommenterad funktion i koden med orden “todo”.

Jag utvecklar med hjälp av Scrum utan att jag tänker på det, för jag kommer på någon idé och skriver in det i kanban-et eller på ett anteckningsblock och sedan jobbar jag på något i listan i någon vecka och testar den medan jag programmerar, sedan släpper jag ut den till live production, men jag har inte kunnat sätta det till ord - Förrän nu, agil utveckling och Scrum. I detta fall finns det ingen Scrum Master eller projektledare utan jag är alltså allt-i-ett och jobbar självständigt och löser de problem som dyker upp själv men hade jag kanske haft kollegor till detta projekt hade jag kanske mest ha varit en projektledare då jag förstår projektet och vilka krav den ska uppfylla men annars hade jag kunnat bara programmera och inte oroat mig om problem - Utan jag hade nog haft möjlighet att lägga in dem i backloggen. Självklart är denna agila arbetsmodell tänkt för en arbetsgrupp och inte för soloprojekt vilket betyder att jag inte använder mig av hela processen. Något viktigt jag kan tänka på till nästa sprint är återkopplingen, eller “Sprint retrospective”, det vill säga vad som gick bra under veckans/sprintens gång och vad som går att förbättra till nästa sprint. Och kanske ha mer tydliga deadlines till mina egna sprints, för i dagsläget räknar jag på att “kanske” bli klar över helgen och planerar inte mycket mer än så.

Jag vet inte exakt hur mitt projekt hade sett ut utan dessa imitationer jag har av agila utvecklingsmetoder, kanske hade mycket tagit längre tid att genomföra eller kanske hade jag inte kunnat komma långt alls med projektet för jag tänker mig nog att den agila filosofin är mer normalt än vad man kan tänka sig, och att vi kanske inte har kunnat sätta ord till varför vi utför handlingar i en viss ordning oavsett om det är till programmering, eller att åka till affären och köpa mjölk…