TDT4140 - Programvareutvikling
Leveranse 8

Antall ord utenom forside og referanser
{ ord }

NTNU | Norwegian University of Science and Technology

Innhold

1 Hva er parprogrammering?

Parprogrammering er en metode innen ekstremprogrammering der to utviklere jobber sammen. Den ene utvikleren, «sjåføren», skriver koden mens den andre, «navigatøren», følger med, leser koden, og gir tilbakemeldinger underveis. Etter en bestemt tid kan utviklere bytte roller [1]. I en meta-analyse gjort av Hannay, Dybå, Arisholm og Sjøberg i 2009 kommer det frem at parprogrammering er et nyttig verktøy for å øke kvaliteten på komplekse oppgaver, og produktivitet på enklere oppgaver [2]. En annen studie, av McDowell et. al. i 2006 viser at parprogrammering har stor virkning som et pedagogisk verktøy, og at det er en god måte for studenter å lære programmering på [1]. Parprogrammering ser også ut til å være nyttig for å ivareta og utvikle studenters interesse for programmering, og særlig blant kvinner, en stadig underrepresentert gruppe i IT-bransjen [1][3].

Parprogrammering differensierer seg fra å bare «kode-sammen» ved at det er en strukturert måte å jobbe på. Den ene utvikleren er den som skriver koden, mens den andre utvikleren er den som følger med og gir tilbakemeldinger [3]. Dette er en viktig forskjell fra å bare kode sammen, hvor det er vanskeligere å få tilbakemeldinger og gi tilbakemeldinger på koden.

2 Problemstilling

I denne oppgaven har jeg valg å ta for meg problemstillingen «Hvordan påvirker parprogrammering samarbeid og produktivitet i et utviklingsteam?». Oppgaver omfavner problemstillingen i lys av teori og erfaringer fra et prosjekt.

3 Hvordan bruke parprogrammering i praksis?

3.1 Erfaringer fra parprogrammering i eget prosjekt

Måten vi brukte og gjennomførte parprogrammering i prosjektet skilte seg ut fra slik det blir presentert i teorien. Hovedsaklig var dette fordi få av oss hadde tidligere erfaring med parprogrammering, og de som hadde det hadde aldri fått en formell opplæring i hvordan det skulle gjøres. Vi hadde derfor ikke en klar forståelse av hvordan vi skulle gjennomføre parprogrammering, og rollefordelingen fulgte derfor ikke strukturen med en «sjåfør» og en «navigatør» som beskrevet i prinsippene til ekstremprogrammering [4]. Praksisen vår var derfor heller nærmere det å «kode-sammen»; der to stykker, på hver sine maskiner, jobbet med den samme koden, og gjerne både skrev og observerte koden samtidig.

Basert på anbefalinger vi hadde fått, og observasjoner av andre prosjekter [4], hadde vi tidlig fokus på å gjennomføre parprogrammering. Vi hadde allerede før starten av første gjennomgang satt av faste tidspunkter for parprogrammering, og fikk dermed benyttet oss av det fra første stund. Dette mener jeg har vært til stor fordel from gruppens arbeid, da vi tidlig fikk utveksle ideer og erfaringer med hverandre, og fikk en god innsikt i hvordan de andre på gruppen jobbet.

Til tross for det tidlige fokuset på parprogrammering, endte vi i starten med å nærmest ha faste par, noe som ikke er optimalt. Det å skifte partner ofte i parprogrammering er nemlig en god ting [4], og gjør at man hele tiden får nye perspektiver på koden man lager. Etter første gjennomgang av prosjektet satte vi derfor konkrete aksjonspunkter om å «bli flikkere på parprogrammering», og dette innebar blant annet det å skifte partner oftere.

Kommunikasjonen vi hadde under denne formen for parprogrammering var noe vi ikke hadde tenkt på i forkant av prosjektet. Underveis kunne den virke lik den som en «navigatør» skal ha; vi ga konstruktive tilbakemeldinger på koden, stilte spørsmål ved logikken og syntaks-bruk, og ga forslag til forbedringer [1]. En utfordring her lå heller i at det ofte var enklere å selv endre på koden til den andre, for så å gi tilbakemelding, enn det var å først gi tilbakemeldingen og diskutere dette med partneren. Dette kommer mest sannsynlig av at vi ikke hadde en sterk rollefordeling, og at begge parter tok på seg begge roller samtidig.

3.2 Effekten av parprogrammering

Selv om vår praksis ikke fulgte de prinsippene som er beskrevet i teorien, så merket vi likevel at parprogrammering hadde en positiv effekt på produktiviteten vår. Som tidligere nevnt viser meta-analysen fra 2009 at parprogrammering har en positiv effekt på kvaliteten på komplekse oppgaver, og produktiviteten på enklere oppgaver [2]. Der vår gruppe merket det mest var i de komplekse oppgavene. Når vi arbeidet med oppgavene individuelt var det lett å sette seg fast og ikke komme videre, men hvis du hadde en partner du kunne sparre med og diskrete problemet, så dukket løsningene raskere opp.

Om dette derimot er fult og helt på grunn av parprogrammeringen er ikke lett å si. For et viktig faktum å tilføye er at flere på gruppa hadde lite erfaring med å programmere, og særlig større, mer komplekse, prosjekter. Det å da bare kunne ha noen og diskutere med vil alene kunne hjelpe effektiviteten. Parprogrammering har også vist seg å være et effektivt pedagogisk verktøy [1], så man kan heller ikke utelukke at det er kombinasjonen av å ha en partner å diskutere med, og å ha en strukturert måte å jobbe på, som har gitt oss en positiv effekt.

3.3 Forbedringer av praksis

For å kunne forbedre vår praksis med parprogrammering, så må vi først se på hva som fungerte og ikke fungerte. Det som fungerte var at vi hadde et tidlig fokus på parprogrammering, og at vi hadde faste tidspunkter for dette. Dette gjorde at vi fikk benyttet oss av parprogrammering fra første stund, og at vi fikk utvekslet erfaringer og ideer med hverandre. Det som ikke fungerte var at vi ikke hadde en klar rollefordeling, og at vi ikke hadde en god måte å kommunisere på. Dette førte til at vi ikke fikk utnyttet parprogrammering til det fulle potensialet, og at man ofte bare jobbet med koden sin selv for så å forklare til partneren hva man hadde gjort.

En anbefaling til grupper i en liknende situasjon er å tidlig fokusere på parprogrammering, men også være beviste på å holde seg til prinsippene til parprogrammering og ikke nødvendigvis bare kjøre sin egen stil og kode sammen slik vi endte opp med. God kommunikasjon er også viktig, og å gi riktig tilbakemelding til partneren sin er nesten like viktig som å faktisk skrive god kode.

4 Konklusjon

For å svare på problemstillingen har parprogrammering påvirket samarbeidet i den forstand at gruppen raskere kom i gang med å jobbe sammen, og at vi fikk en bedre forståelse av hverandres arbeidsmetoder. Parprogrammering har også påvirket produktiviteten i den forstand at vi fikk mer gjort på kortere tid, og at vi fikk arbeidet mer sammen. Som gruppe

var vår største svakhet at vi ikke hadde en god nok forståelse av hva parprogrammering faktisk innebar, og derfor ikke fikk utnyttet metoden til det fulle potensialet.

Referanser

[1] C. McDowell, L. Werner, H. E. Bullock og J. Fernald, «Pair Programming Improves Student Retention, Confidence, and Program Quality,» *Commun. ACM*, årg. 49, nr. 8, s. 90–95, aug. 2006, ISSN: 0001-0782. DOI: 10.1145/1145287.1145293. adresse: https://doi.org/10.1145/1145287.1145293.

[2] J. E. Hannay, T. Dybå, E. Arisholm og D. I. Sjøberg, «The effectiveness of pair programming: A meta-analysis,» *Information and Software Technology*, årg. 51, nr. 7, s. 1110–1122, 2009, Special Section: Software Engineering for Secure Systems, ISSN: 0950-5849. DOI: https://doi.org/10.1016/j.infsof.2009.02.001. adresse: https://www.sciencedirect.com/science/article/pii/S0950584909000123.

[3] H. M. Kattan, F. Soares, A. Goldman, E. Deboni og E. Guerra, «Swarm or Pair? Strengths and Weaknesses of Pair Programming and Mob Programming,» i *Proceedings of the 19th International Conference on Agile Software Development: Companion*, ser. XP '18, Porto, Portugal: Association for Computing Machinery, 2018, ISBN: 9781450364225. DOI: 10.1145/3234152.3234169. adresse: https://doi.org/10.1145/3234152.3234169.

[4] H. Kniberg, *Scrum and XP from the Trenches*, 2. utg. InfoQ, 2015.