1 Introductie

1.1 Het project en diens context

Het idee voor het project stamt uit een probleem dat ikzelf ervoer bij mijn lessen informatica op de middelbare school: soms wil je samen met iemand in het zelfde bestand code schrijven, maar dit is met de meeste *editor*s niet mogelijk. Destijds vond ik *Cloud9*, een *editor* waarin dit wel mogelijk was. Sindsdien heb ik zelf echter de wens gehad zelf zo'n *editor* te maken. Zodoende heb ik voor aanvang van het vak een groepje bijeengezocht en geprobeerd dit project met hen te kunnen uitvoeren.

1.2 Groepsstructuur

Op de eerste dag van het vak werd al meteen begonnen met het verdelen van organisatorische taken binnen de groep. Het resultaat hiervan is te zien in tabel **??**. De volgende dag werd een verdeling van de programmeertaken gemaakt. In deze verdeling werd een duidelijk onderscheid gemaakt tussen de twee componenten van ons project: de *client* en de *server*.

1.3 Persoonlijke rol

Zoals te zien is in tabel **??** was ikzelf voorzitter van de vergaderingen en leider van de subgroep die verantwoordelijk was voor het implementeren van de *server*-infrastructuur. Uit dit laatste volgt logischerwijs dat ik op het gebied van programmeren geschaard werd onder de *server*-groep.

1.4 Verwachte en daadwerkelijke resultaten

Zoals ons ook was verteld door de vakcoördinator en assistenten van het vak, waren onze beoogde projectresultaten aan de optimistische kant. Naast de basisimplementatie voor de *editor* hadden wij in ons projectverslag uitgebreide vervolgplannen om op deze basis verder te bouwen. Uiteindelijk bleek dat niet van deze extra functionaliteit daadwerkelijk gerealiseerd kon worden, gezien de inherente complexiteit van ons basisproduct.

Het grootste compromis wat betreft onze verwachtingen en het uiteindelijke resultaat kwam in de vorm van *locks*. Origineel hadden we voor ogen dat een gebruiker overal een bestand zou kunnen bewerken. Het bleek echter dat een dergelijke implementatie voor de duur van 4 weken te hoog gegrepen was. In plaats daarvan werd de *editor* zo gemaakt dat een gebruiker eerst een regio moest *locken*, alvorens deze dit aangegeven gebied kon bewerken. Dit betekende ook dat er per gebied enkel één gebruiker aanpassingen kan maken.

2 Methode

2.1 Ontwikkelingsaanpak

Zoals benoemd in sectie 1.2 was onze groep wat betreft het implementeren van het project verdeeld in twee sub-groepen. Hieruit volgde dat de ontwikkeling dus ook in twee parallelle stromen verliep. Het bepalen van de taken die iedere week op de planning moesten komen verschilde gedurende de 4 weken van het project; in het begin werd veel *ad-hoc* bedacht, terwijl in latere weken steeds een planning op de vrijdag werd gemaakt voor de daaropvolgende week. Het bijhouden van de taken verliep sinds het begin echter wel op een op dag één vastgestelde wijze: informatie over individuele taken werd bijgehouden op *Github* m.b.v *issues* (figuur **??**) en de staat van projecten—het zij in ontwerp-, productie- of testfase—werd in het projectbord van de *Github-repo* bijgehouden (figuur **??**).

2.2 Persoonlijke afhandeling van taken

Achter het kiezen van taken zat geen uitgebreid doordachte strategie. Binnen mijn *server*-groep koos ieder een taak die op dat moment vaak het meest urgent was, of waarvan andere taken afhingen. Wel had ieder van ons binnen de groep een bepaald deel van de *server*-infrastructuur waaraan hij tot dan toe het meest had gewerkt en waar hijzelf dus het meest over wist. In mijn geval was dit het bestandssysteem en de datastructuren die hiermee samenhingen.

Het voltooien van taken gebeurde vaak op twee manieren: in het geval van een kleine taak probeerde ik dit zo snel mogelijk te implementeren en samen te voegen met het werk van anderen; maar in het geval van grotere taken—waarvan er veel waren—zocht ik vaak een groepsgenoot op om deze mee te voltooien. Dit laatste gebeurde vaak met behulp van het eerder genoemde *Cloud9*. Zo bleek dus ook maar weer tijdens het uitvoeren van dit project dat een specifieke *editor* die gebouwt is om samenwerken aan bestanden mogelijk te maken een uiterst nuttig gereedschap is.

2.3 Aanpak bij nieuwe technologieën en concepten

Origineel zouden we het project geheel in C++ schrijven, een taal die voor velen nieuw zou zijn geweest. Hiervan werd uiteindelijk, gezien de complexiteit, van afgezien. Uiteindelijk zijn er aan de kant van de server wel nieuwe technologieën gebruikt. Deze zijn aan de hand van documentatie bestudeerd door sommigen en uitgelegd aan de anderen in de *server*-groep.

3 Geleerde lessen

3.1 Verwachtingen vooraf

Het was helaas al snel duidelijk dat de doelen die wij onszelf voorafgaand aan het project gesteld hadden veel te ver gegrepen waren. De assistenten hadden ons hiervoor gewaarschuwd, maar met mijn optimistische instelling wilde ik het toch proberen. Naderhand zie ik wel in dat een eenvoudiger project misschien minder voldoening had gebracht, maar uiteindelijk wel beter te realiseren was.

3.2 Groepsstructuur

Aan het begin van het project hielden wij vast aan een rigide groepsstructuur. Dit betekende dat er voor beslissingen een duidelijke verantwoordelijkheid was en dat deze persoon ook aan de rest van de groep verantwoordelijkheid moest afleggen. Deze structuur verviel gaandeweg het project en zorgde bij mij persoonlijk geregeld voor irritatie. Nu namen plots mensen belangrijke beslissingen die de hele groep aangingen, zonder dat dit met de (gehele) groep overlegd werd. Er ontstond een semi-democratie waarin niet iedereen kon meestemmen of enkel de stem van sommigen werd meegeteld.

3.3 SCRUM

Gedurende het project bleek dat, gezien de complexiteit van ons basisproduct, dat het voor de 4 weken die wij hadden moeilijk was om geheel *SCRUM* te werken. Zeker in de eerste 2 weken moest eerst nog een basis gelegd worden, voordat er daadwerkelijk iteratief gewerkt kon worden. Uiteindelijk kon aan de kant van de *server* in week 3 en 4 daadwerkelijk gewerkt worden op basis van een iteratief proces.

4 Conclusie