# Eindverslag

# Spreadsheat

# Kasper Wendel

4331362

Jonathan Raes  
4300343

Chak Shun Yu  
4302567

Hamza Hussain  
4299957

Maiko Goudriaan  
4302125

Versie 0.3

6-1-14

# Inhoud

Inhoudsopgave

**Algemeen3**

Planning3

Samenwerking3

Overleg3

Versiebeheer4

**Ontwerpproces4**

Problemen4

Technische afwegingen4

UML5

**Verbeterpunten5**

Software5

Vak5

**Individuele feedback**5

Kasper5

Jonathan5

Chak Shun5

Hamza5

Maiko5

# Algemeen

## Heb je je aan de planning kunnen houden?

We hebben elke week een planning gemaakt en laten zien aan de SA. Hierover kregen wij feedback hoe wij beter de planning konden indelen. Zo werd de eerste weken gezegd dat we specifieker moesten zijn in wat we die week wilde gaan doen. Dus duidelijkere sprintbacklogs. Daarnaast waren de planning en sprint te algemeen. Zo hadden we bijvoorbeeld de GUI en formules als één onderwerp neergezet. Hierop werd het commentaar gegeven dat GUI en formules opgedeeld moesten worden in meerdere delen. Dit hebben we de eerste week dan ook gedaan. Naar mate we verder in de weken kwamen, kwamen we er achter dat dit inderdaad noodzakelijk was. Het is namelijk onmogelijk om in één keer alle formules af te hebben en zo ook de GUI. We hebben de formules in de eerste planning opgedeeld per formule. De GUI hebben we pas in de tweede planning opgenomen, omdat we toen de uitleg en de eerste stappen richting de GUI hadden gezet. In de derde planning hebben we de GUI weer verder opgedeeld samen met de controller om stukje bij stukje te implementeren, aangezien we hier iets achterliepen op onze eigen deadlines en de demo.

De eerste weken konden we de planning goed bijhouden. In de kerstvakantie zijn wij iets achter op schema geraakt. Bij het overleg met de SA werd ons aangeraden om nu de laatste deadline in te plannen. Hier zijn we op in gegaan en hebben duidelijke afspraken gemaakt wat we af willen hebben

Dus de planning was onduidelijk en had te algemene onderwerpen.

## Hoe was de samenwerking binnen het team?

De samenwerking binnen het team was goed. We konden goed met elkaar opschieten. We hadden tijdens de eerste bijeenkomst drie tweetallen gemaakt. Per tweetal was je verantwoordelijke voor een gedeelte van de code en project. Helaas viel in de eerste week ook het eerste lid af. Hierdoor hadden we twee tweetallen en een persoon alleen. In de eerste sprintbacklog kon dit nog omdat dit vooral veel oriënteren was. Later hebben we dit bij bijgesteld. Nu waren er per deel twee mensen verantwoordelijk. Zo kon je dus meerdere delen hebben met verschillende mensen. Binnen de tweetallen werd het werk verder verdeeld. In de laatste twee sprintbacklogs veranderede dit weer. Nu was er per onderdeel vaak nog maar een persoon verantwoordelijk, maar dit betekende niet de persoon dat onderdeel alleen kon doen. Vaak was er overleg nodig om de laatste koppelingen te kunnen maken.

## Hoe hebben jullie overleg gepleegd?

Het overleg heeft voornamelijk tijdens de ingeroosterde uren plaatsgevonden. In de eerste sprint hebben we bedacht hoe het programma er ongeveer uit ging zien. Hierop hebben we het werk gezamenlijk verdeeld. In de volgende sprints hadden we een vaste routine. Iedereen vertelde voor zichzelf wat hij had gedaan en wat zijn problemen waren. Nadat iedereen zijn problemen had verteld. Zochten we met zijn allen naar oplossingen. Daarop volgend werd er besproken wie wat verder ging doen in die week.

Naast de vaste vergaderingen hadden we ook een WhatsApp groep. Hier werden vooral korte vragen gesteld en beantwoord. Verder werd en vaak een berichtje ingegooid er als iets nieuws was gecommit. Bij de commit stond vaak ook een korte beschrijving wat er veranderd was. Ook vond er in de wandelgangen af en toe kort overleg plaats.

## Heeft versiebeheer jullie geholpen?

Versiebeheer heeft ons minder geholpen dan gehoopt. Dit zat namelijk in het onder de knie krijgen van GitHub. Verder hadden wij eenvoudigere omgang met GitHub verwacht. In het bijzonder dan de “online” communicatie in de groep. We hadden een overzichtelijkere communicatie verwacht zoals een forum. Daarnaast miste er ook een overzicht voor de sprints. Uiteindelijke hebben we de functies wel gevonden, maar om de functies te gebruiken koste extra tijd, aangezien we dit tijdens de vergaderingen hadden besproken. Hierdoor hebben we dan ook gekozen om GitHub alleen te gebruiken voor filesharing. Mocht er toch een acuut probleem zijn dan werd dit via WhattsApp besproken. Je bereikt de groep immers via de mobiel sneller dan via de computer.

GitHub heeft ons wel geholpen bij het beheren van de bestanden. Na een iets wat moeilijke start werd alles toch duidelijk. Het voordeel van GitHub ten opzichte van andere bekende programma’s zoals DropBox is dat je GitHub kan koppelen aan Eclipse.

# Ontwerpproces

## Welke problemen zijn er opgelost?

## Welke technologische afwegingen zijn er gemaakt?

Voor de GUI hebben wij gekozen voor Swing. We hadden de keuze uit AWT, Swing, JavaFX en SWT of programma’s die de GUI maken. De laatste optie om een programma te gebruiken om de GUI te maken hebben we gelijk geschrapt, omdat de gegeneerde code vaak moeilijk aan te passen is en je moet het desbetreffende programma ook onder de knie krijgen. Daarna vielen AWT en SWT af. AWT is de voorganger van Swing. Dus al kies je AWT kan je beter Swing nemen. Hier zitten namelijk meer mogelijkheden in. SWT sprak ons minder aan dan Swing en JavaFX. Dit kwam voornamelijk door de hoeveelheid informatie over SWT minder was dan de informatie over Swing en JavaFX. Het verschil tussen Swing en JavaFX is voornamelijk het aantal features. Deze zijn bij JavaFX velen malen hoger dan bij Swing. Bij JavaFX kan je echt alles naar je eigen wens zetten. Dit gaf wel een hogere moeilijkheidsgraad mee. Aangezien wij niemand van ons programmeer ervaring had met GUI hebben we gekozen om bij de basis van Swing te blijven. Hier zitten namelijk genoeg features in om te ontdekken.

We hebben voornamelijk de subklassen van JComponent gebruikt om de GUI in elkaar te zetten. Zo hebben wij o.a. JTable, JScrollBar en JMenuBar gebruikt. Daarnaast hebben JFreeChart gebruikt om de grafieken te maken.

Voor versie beheer hebben wij GitHub gebruikt. Dit werd ons namelijk verplicht. Wij hadden aan het begin graag een andere applicatie voor versie beheer gebuikt. Dit was voornamelijk omdat wij de werking GitHub niet snapte en het ontbreken van de features beschreven bij “heeft versie beheer jullie geholpen”. Na twee weken snapte we GitHub. We hadden verder geen problemen met GitHub behalve de ontbrekende features.

**UML**

**Formules: parsers etc.**

**XML: parser etc.**

We hebben voor de volgende extra features gekozen: grafieken en opmaak van de cellen. Als we eventueel tijd over hebben zouden we ook nog extra formules maken. De grafieken hebben wij wel geïmplementeerd gebruik makend van JFreeChart. Ook hebben wij een kleurenkiezer geïmplementeerd, maar het toepassing op de kleur van de cellen en tekst is nog niet toegepast. Daarnaast zijn de andere opmaak opties ook nog niet toegepast.

Controller

## Hoe heeft UML jullie wel/niet geholpen?

# Verbeterpunten

## Hoe kan de software verbeterd worden?

* Betere testen en meer testen.
* Meer commentaar.
* Tabbladen.
* Uitgebreidere GUI
* Opmaak implementeren.

## Hoe kan dit vak verbeterd worden?

# Individuele feedback(p.p. 200 woorden)

## Sterkere/zwakkere punten, conflicten met andere teamleden, hoe zijn problemen opgelost.