Afstudeerverslag

De ontwikkeling van een Google Adwords Module

Student
Roy van Wensen
r.v.wensen@student.hhs.nl
09002545

Opleiding
HBO Informatica van de Haagse Hogeschool,
locatie Holland Spoor

Examinatoren
Begeleidend examinator: dhr. ir. B. Kuiper
Tweede examinator: dhr. ing. H.F. Schouten

Opdrachtgever
Wolter Tjeenk Willink, eigenaar Traffic Builders BV

Bedrijfsmentor
Dhr. T. Oner, SEM Software Engineer

Afstudeerperiode
27 augustus 2012 tot en met 21 december 2012

Referaat

Deze afstudeerscriptie beschrijft het onderzoek naar de wensen en eisen voor een Google Adwords Module voor het bedrijf Traffic Builders en de wijze waarop deze is ontwikkeld. Daarbij zijn de volgende technieken en tools gebruikt:

- Google Adwords
- Goolge Adwords Editor
- PHP 5.3
- Mysql
- Apache
- UML.

Voorwoord

Dit afstudeerverslag is geschreven in het kader van mijn studie aan de Haagse Hogeschool, academie ICT & Media, opleiding Informatica. Door middel van dit afstudeerverslag wil ik inzage geven in het traject dat ik heb afgelegd om mijn afstudeerperiode tot een succesvol einde te brengen.

Hierbij wil ik Traffic Builders bedanken voor het mogen uitvoeren van de opdracht die in dit verslag wordt beschreven en voor de opgedane kennis tijdens mijn afstuderen. Mijn dank gaat in het bijzonder uit naar Tuncay Oner, Wolter Tjeenk Willink en de SEA consultants die mij tijdens mijn afstudeeropdracht hebben ondersteund. Buiten Traffic Builders wil ik Annemarie Blokland, Jeroen de Klerk en Hans Fontijn bedanken voor hun steun en hulp.

De eerste keer dat ik Traffic Builders binnen liep kan ik me nog goed herinneren. De nieuwe gezichten, nieuw werk en 8 uur per dag op één plek aanwezig zijn – het was allemaal nieuw voor mij. Nu mijn afstudeerperiode ten einde is en ik terugkijk op een leuke en leerzame tijd, kan ik zeggen dat ik een belangrijke ervaring rijker ben. Naast nieuwe vakkennis heb ik kunnen ruiken aan aspecten als collegiale samenwerking, de structuur van een organisatie, werksfeer en de balans tussen werk en privéleven. Dat alles heeft mijn periode bij Traffic Builders extra zin gegeven.

Roy van Wensen

Almere, maandag 7 januari 2013

Inhoudsopgave

1 Inleiding 1

2 Traffic Builders en zijn werkzaamheden 2

2.1 Het bedrijf 2
2.2 Afdelingen 2

3 De afstudeeropdracht 4

3.1 Probleemstelling 4
3.2 Doelstellingen van de afstudeeropdracht 4
3.3 Resultaat 5

4 Aanpak 6

4.1 Projectmethodiek 6
4.2 Toepassing RUP 6
4.2 planning 8

5 Inceptiefase 9

5.1 Plan van aanpak 9
5.2 Onderzoeken 9
5.3 Interviews 12
5.4 Requirements 14
5.5 De enquête 14
5.6 Architectuur 15
5.7 Conclusie inceptiefase 17

6 Elaboratiefase 18

6.1 Uitbreiding requirements 18
6.2 Use cases 19
6.3 Systeem- en databaseontwerp 24
6.4 Aanpak implementatie 26
6.5 Testen 29
6.6 Conclusie elaboratiefase 30

7 Constructiefase iteratie 1 32

7.1 Query's 32
7.2 CSV-export 32
7.3 Meerdere versies van de templates 34
7.4 AdsTemplate klasse 35
7.5 Feedback 36
7.5 Conclusie constructiefase 37

8 Constructiefase iteratie 2 38

8.1 Module omzetten in Rankinspectorstijl 38
8.2 Overige aanpassingen 39
8.3 Implementatie in de live omgeving 40

9 Transitiefase 41

9.1 Voorbereiding en uitwerking 41

10 Verantwoording beroepstaken 43

11 Evaluatie 46

11.1 Evaluatie inceptiefase 46
11.2 Evaluatie elaboratiefase 46
11.3 Evaluatie constructiefase 46
11.4 Evaluatie transitiefase 46

Bronnen 47

Definitielijst 48

1 Inleiding

Deze scriptie hoort bij het afstudeerproject van Roy van Wensen, uitgevoerd in opdracht van Traffic Builders. Dit verslag beschrijft het afstudeerproject, de keuzes die ik heb gemaakt en waarom ik deze gemaakt heb.

Leeswijzer
Hoofdstuk 2 is een beschrijving van het bedrijf Traffic Builders, de werkzaamheden die de verschillende afdelingen binnen Traffic Builders verrichten en mijn plek daarin.

In hoofdstuk 3 staat de situatie beschreven zoals deze was bij aanvang van mijn afstuderen. Hieruit is tevens mijn afstudeeropdracht ontstaan.

In hoofdstuk 4 staat mijn plan van aanpak beschreven. Dit plan van aanpak bevat ook een planning.

In hoofdstuk 5 wordt het verloop van de inceptiefase toegelicht. Daarin staat ook de manier waarop het onderzoek is gedaan en de resultaten van het onderzoek.

In hoofdstuk 6 staat beschreven hoe de eerste stappen van de opdracht richting de ontwikkeling van de Google Adwords Module zijn gezet. Dit is de elaboratiefase.

Hoofdstuk 7 en 8 beschrijven het verloop van de constructiefase in twee iteraties. Tevens wordt toegelicht hoe de ontwikkeling van de Google Adwords Module is verlopen.

Hoofdstuk 9 beschrijft het proces van de overdracht van de nieuwe module aan de opdrachtgever en de gebruikers.

Hoofdstuk 10 bevat de verantwoording van mijn beroepstaken.

Hoofdstuk 11 bestaat uit de evaluatie. Er staat beschreven hoe mijn project is verlopen, wat goed ging en wat voor verbetering vatbaar is.

2 Traffic Builders en zijn werkzaamheden

In dit hoofdstuk staat beschreven wat Traffic Builders doet, welke afdelingen er binnen het bedrijf zijn en wat mijn plek daarin was.

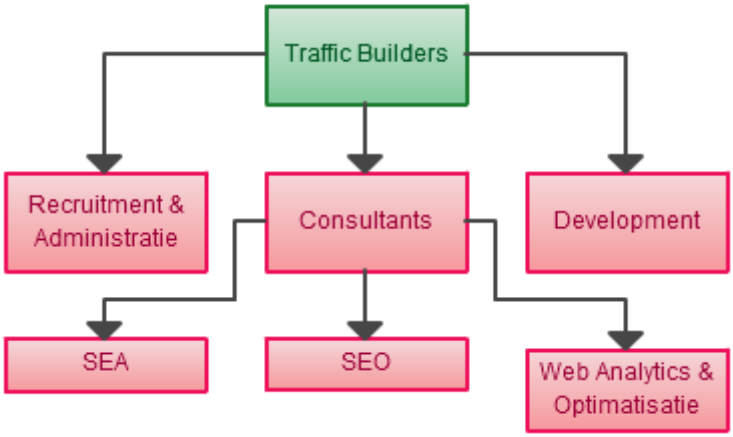
2.1 Het bedrijf

Traffic Builders is een commercieel zoekmachinemarketingbureau, dat is opgericht in 2001. Traffic Builders streeft ernaar "*ambitieuze opdrachtgevers hun online marketingdoelstellingen te helpen overtreffen".* Het bedrijf bestaat uit 25 consultants, copywriters en developers. Zij werken dagelijks voor opdrachtgevers uit de thuiswinkelbranche, reisbranche, financiële dienstverlening en zakelijke dienstverlening, zowel nationaal als internationaal. De werkzaamheden die de werknemers van Traffic Builders verrichten zijn:

- Zoekmachine-optimalisatie (SEO)
- Google Adwords-campagnes (SEA)
- Web Analytics
- Conversie-optimalisatie

2.2 Afdelingen

Er zijn 3 afdelingen binnen Traffic Builders: Recruitment & Administratie, Consultants en de Developmentafdeling. Het organigram is te zien in figuur 1.



Figuur 1 - Afdelingen Traffic Builders

Recruitment & Administratie

De afdeling Recruitment & Administratie zorgt voor de boekhouding en voor de werving van nieuwe medewerkers. Ook handelt zij aanvragen voor stages en afstudeerprojecten af. Tijdens het aanvragen van mijn afstudeerproject heb ik contact gehad met deze afdeling.

SEO-consultants

De afdeling Search Engine Optimalization (vanaf nu SEO) werkt aan het verbeteren van websites van klanten. De verbeteringen zijn gericht op het bevorderen van de vindbaarheid in de organische resultaten van Google. Deze staan ook wel bekend als de gratis resultaten van Google. Werkzaamheden die onder andere uitgevoerd worden om dit te realiseren zijn:

- het maken van site structuur
- page speed optimalisatie
- het aanvragen van vermeldingen op andere websites
- het creëren van nieuwe content.

Traffic Builders heeft een online tool ontwikkeld, genaamd Rankinspector. Deze online tool heeft modules die SEO-consultants in staat stellen om:

- de positie van de eigen website en die van de klanten in Google op bepaalde keywords te monitoren in de organische resultaten van Google;

- de geschiedenis van posities in te zien; deze worden namelijk opgeslagen in Rankinspector. Daarmee kan worden bekeken of er sprake is van stijging of daling in de organische resultaten van Google;
- een concurrentieanalyse uit te voeren. Met deze analyse kan worden bekeken welke online strategieën van concurrenten leiden tot betere posities in Google.

Met behulp van de resultaten uit Rankinspector geven de SEO-consultants advies aan de klanten van Traffic Builders.

SEA-consultants
De Search Engine Advertising (vanaf nu SEA)-consultants van Traffic Builders werken voor klanten zoals het postorderbedrijf Otto en reisorganisator Toerkoop. Deze bedrijven hebben een groot assortiment producten. Veel van deze producten moeten worden geplaatst in Google Adwords, het advertentienetwerk van Google. De SEA-consultants maken voor deze bedrijven advertenties voor de verschillende producten in de Google Adwords Editor of met de andere tools die het bedrijf gebruikt. O2mc en Adcore zijn de tools die momenteel gebruikt worden voor dit soort grote campagnes. Deze tools zijn echter duur en missen functionaliteiten waarmee de SEA-consultants sneller zouden kunnen werken.
De Google Adwords Module die in het kader van mijn afstudeeropdracht ontwikkeld gaat worden, moet het voor de consultants eenvoudiger maken om grote hoeveelheden advertenties sneller aan te maken. De SEA-consultants zijn dan ook degenen die het meest gebruik gaan maken van de nieuwe module.

Consultants Web Analytics & Web Optimalisatie
De afdelingen SEA en SEO zorgen voor nieuwe bezoekers op de websites van klanten. De afdeling Web Analytics & Web Optimalisatie zorgt ervoor dat deze bezoekers daadwerkelijk overgaan tot de gewenste actie. Deze afdeling streeft naar zo hoog mogelijke verdiensten, via het verkopen van een product in een webshop, inschrijving voor een nieuwsbrief, aanvraag voor een offerte of een andere gewenste actie.
Het optimaliseren van websites – een andere taak van deze afdeling – vindt vooral plaats via splittesting. Er worden 2 pagina's gemaakt en naar elke pagina worden 1000 bezoekers geleid. Achteraf wordt bekeken welke pagina het beste resultaat oplevert. Alleen deze pagina wordt dan in de toekomst nog gebruikt.

Development
Op de afdeling Development wordt gewerkt aan de ontwikkeling van Rankinspector. Rankinspector is een tool die posities van websites in zoekresultaten in de gaten houdt aan de hand van bepaalde keywords. Ook worden op deze afdeling nieuwe modules voor Rankinspector ontwikkeld, zoals:

- een vereenvoudigde XML sitemap van een website;
- een module die een Google Shopping Feed maakt van de producten in een webshop;
- een linkbuiling tool die backlinks naar de website opzoekt.

Mijn plaats
De afdeling Development is de plek waar ik mijn afstudeeropdracht heb uitgevoerd. Daar heb ik de Google Adwords Module voor Rankinspector ontwikkeld.

3 De afstudeeropdracht

In dit hoofdstuk zal de afstudeeropdracht nader worden uitgelegd aan de hand van de probleemstelling, de doelstelling en het beoogde resultaat.

3.1 Probleemstelling

De SEA-consultants van Traffic Builders creëren grote Google Adwords campagnes. In deze campagnes moeten van verschillende producten uit verschillende categorieën advertenties worden opgemaakt en ingevoerd. Het opmaken van advertenties houdt in dat er voor elk product een titel, beschrijving en URL moet worden bedacht. Daarna worden de advertenties in Adwords geplaatst en geactiveerd, zodat ze in de betaalde zoekresultaten van Google verschijnen.
Het opmaken en invoeren van de vele verschillende advertenties gebeurt op dit moment handmatig of met bestaande tools. Het handmatig opzetten van grote campagnes is een tijdrovende en dus een kostbare bezigheid. Bovendien kosten de tools waarmee campagnes nu worden ontwikkeld veel geld en missen de functionaliteit om snel advertenties te realiseren.

Traffic Builders wil het creëren van Adwords campagnes versnellen, zodat er aanzienlijk kan worden bespaard op de kosten. In de opmaak van de advertenties zitten vaak maar kleine verschillen die terug te brengen zijn tot een klein aantal variabelen zoals productnaam, prijs, locatie, uitvoering en merk. Daarom wil Traffic Builders een module ontwikkelen waarin een standaardtekst opgegeven kan worden met verschillende parameters. Hiermee kan voor verschillende producten één advertentie worden gegenereerd. Deze moet vervolgens op eenvoudige wijze in Google Adwords geplaatst kunnen worden.

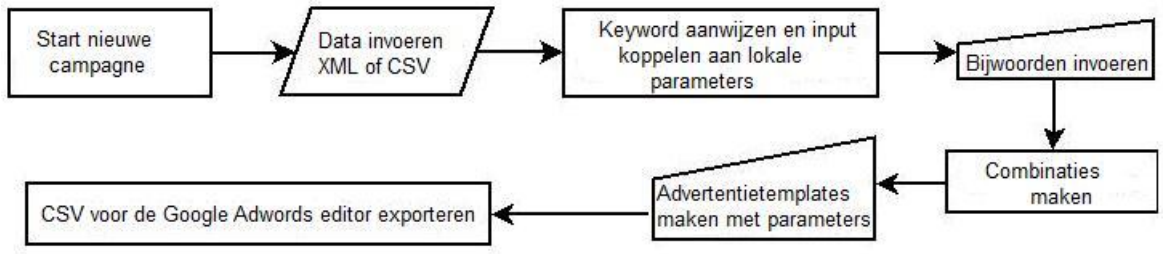
3.2 Doelstellingen van de afstudeeropdracht

Het doel van mijn afstudeeropdracht is het oplossen van het probleem dat in paragraaf 2.2 en 3.1 staat beschreven, door het ontwikkelen van een module voor Rankinspector. Deze module krijgt de naam ‘Google Adwords Module’. Voor deze nieuwe module wordt een eigen database ontwikkeld. De module moet ervoor zorgen dat het proces van het creëren van grote hoeveelheden advertenties in Google Adwords sneller gaat.

In de module moet het mogelijk zijn:

- data aan te leveren in de vorm van XML of CSV;
- dat de gebruiker aangeeft wat de keywords zijn in de CSV of XML;
- dat de gebruiker een koppeling maakt tussen lokale parameters en de aangeleverde data uit XML of CSV;
- dat de gebruiker bijwoorden invoert om combinaties te kunnen maken met de keywords;
- dat de gebruiker advertentietemplates aanmaakt met parameters die tijdens een eerdere stap zijn aangemaakt. Daarmee zijn de vele losse advertenties – en de kans op fouten – verleden tijd.

De module zal uiteindelijk een CSV (Comma Separated Values) produceren die geïmporteerd kan worden in de Google
Adwords Editor.
Het volledige proces staat schematisch weergegeven in figuur 2.



Figuur 2 - Flowchart van de gewenste oplossing

Het afstudeerproject bestaat uit twee onderdelen. Het eerste onderdeel betreft het verrichten van onderzoek. Het tweede onderdeel omvat de ontwikkeling van de Google Adwords Module. Deze twee onderdelen worden hieronder verder toegelicht.

3.2.1 Te verrichten onderzoek

Voordat de Google Adwords Module kan worden ontwikkeld, moet eerst onderzocht worden wat de wensen en eisen van de gebruikers zijn. Deze wensen en eisen worden vervolgens omgezet naar requirements, belangrijke vereisten voor de module.
Tijdens de inceptiefase wordt vervolgens onderzocht welke data opgeslagen moeten worden in de te ontwikkelen database.
Ook moet er onderzoek plaatsvinden naar de opbouw van de CSV, zodat deze geïmporteerd kan worden in de Google Adwords Editor. Dit onderzoek wordt uitgevoerd tijdens de elaboratiefase.

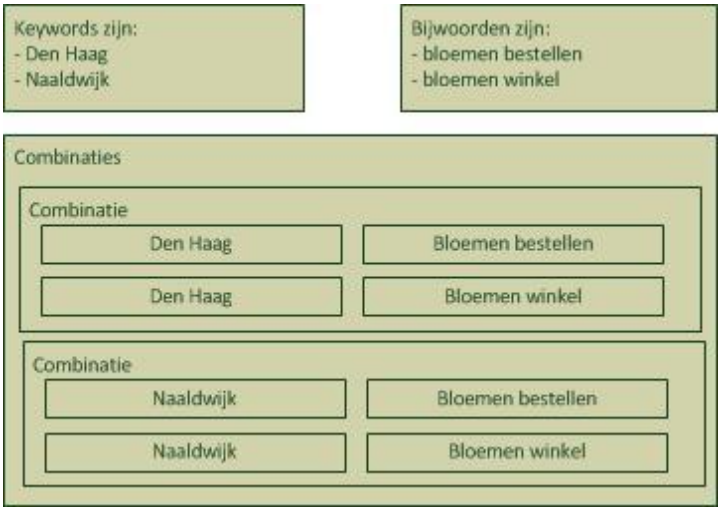
3.2.2 Ontwikkeling van de applicatie

Als de requirements in kaart zijn gebracht en de onderzoeken zijn afgerond, wordt de module ontwikkeld. Om te beginnen wordt een functioneel en technisch ontwerp gemaakt. Daarna volgt een prototype. De presentatie van het prototype levert feedback op die wordt gebruikt voor het realiseren het eindproduct.

3.3 Resultaat

Het uiteindelijke resultaat is een Google Adwords Module waarin de SEA-consultants van Traffic Builders Google Adwords advertenties kunnen maken. De Google Adwords Module moet zorgen voor tijdbesparing. Ook zal het aantal fouten afnemen, doordat alleen de templates nagekeken moeten worden.

De tijdbesparing zal ontstaan doordat de module aan de hand van een lijst met keywords (in CSV- of XML-formaat) en bijwoorden zelf combinaties gaat maken (zie figuur 3). De gebruiker hoeft tijdens het maken van de advertenties alleen nog andere rijen uit de CSV of XML te koppelen aan lokale parameters.



Figuur 3 - Keyword combinaties

Als de combinaties gemaakt zijn en de parameters gekoppeld, moet de gebruiker één of meer advertentietemplates maken met daarin de parameters. Daarna kunnen de advertenties als CSV worden geëxporteerd en in de Google Adwords Editor geüpload. Als een bezoeker dan in Google zoekt op bijvoorbeeld ‘bloemen bestellen Den Haag’ zal er een advertentie worden weergegeven zoals in figuur 4.

Figuur 4 - Uitwerking in Google

4 Aanpak

In dit hoofdstuk staat beschreven welke aanpak ik heb gevolgd tijdens de uitvoering van mijn afstudeeropdracht.

4.1 Projectmethodiek

Traffic Builders heeft geen gestandaardiseerde manier voor het ontwikkelen van applicaties en beschikt ook niet over standaard documentatie. Het bedrijf bekijkt per project welke manier passend is. De applicaties worden meestal in drie fases ontwikkeld: initialisatie, ontwikkeling en oplevering.

Een beproefde, gestandaardiseerde projectontwikkelingsmethode is onmisbaar voor een gestructureerde en eenduidige communicatie tussen de betrokken partijen. Daarom heb ik voorgesteld om gebruik te maken van RUP (Rational Unified Process), een methode die overeenkomt met de werkwijze bij Traffic Builders. RUP heeft één fase meer, de inceptiefase. De andere drie fases – elaboratiefase, constructiefase en transitiefase – komen overeen met de werkwijze bij Traffic Builders.
RUP is een iteratieve en incrementele softwareontwikkelingsmethode die gebaseerd is op best practices. De keuze is op RUP gevallen omdat ik hier ervaring mee heb en ook omdat Traffic Builders reeds gewend is aan een iteratieve ontwikkeling. Verder levert RUP duidelijke mijlpalen, zoals de einddatum en de oplevering van producten binnen het bedrijf.

4.2 Toepassing RUP

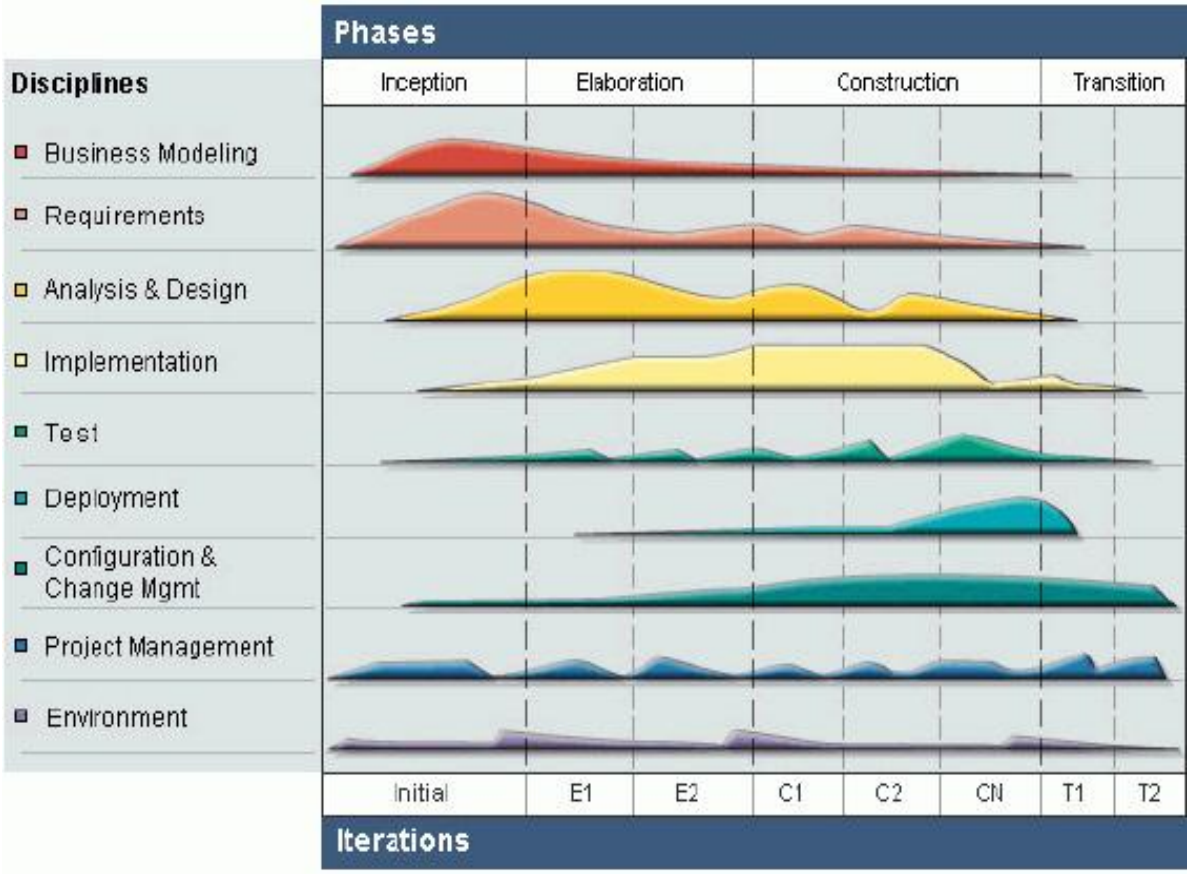
Inceptiefase
[wikipedia 1] “Deze fase dient voor het bepalen van de haalbaarheid van het project en om de inhoud en de afbakeningen te definiëren.” Tevens zullen in deze fase door middel van vooronderzoek de risico’s worden geïnventariseerd en de requirements onderzocht.
De requirements worden gereviewd met de betrokken stakeholders om de gewenste kwaliteit te kunnen behalen en om in een vroeg stadium te ontdekken of er requirements ontbreken.

Elaboratiefase
Als in de inceptiefase duidelijk is geworden dat het project haalbaar is, zullen in de elaboratiefase de functionele requirements en niet-functionele requirements worden opgesteld. Beide worden gedocumenteerd als user story. De user stories definiëren wat er tijdens de afstudeeropdracht gebouwd moet worden.
Ik heb voor user story’s gekozen, omdat zij voor structuur en eenduidigheid zorgen. Alle requirements worden namelijk opgeschreven als ‘*Als <rol>, wil ik dat <doel/wens >*’. Deze notatie zorgt ervoor dat het voor de verschillende stakeholders duidelijk is wat er ontwikkeld gaat worden.
De functionele requirements worden vertaald in use cases en technische ontwerpen, zoals het klassendiagram ERD en RIM.
Tijdens deze fase wordt ook de basis gelegd voor de module. De basis zal een prototype zijn, want “door in een vroeg stadium van het project de stakeholders een grafische voorstelling te geven van het product (prototyping) kunnen stakeholders sturing geven op het product” [Wikipedia 2].

Constructiefase
In de constructiefase wordt de Adwords Module verder ontwikkeld, getest en geïmplementeerd in het huidige systeem. Dit zal in twee iteraties gebeuren. Tijdens de eerste iteratie wordt het prototype door ontwikkeld tot een eerste versie van de module. Deze eerste versie wordt gereviewd en de uitvoer wordt gecontroleerd op validiteit. Tijdens de tweede iteratie worden de laatste aanpassingen doorgevoerd en wordt de module geïmplementeerd in de live omgeving van Rankinspector.

Transitiefase
In de laatste fase van het project zal de module worden getest met een GAT (Gebruikers Acceptatie Test). Hiermee wordt gevalideerd of de ontwikkelde module voldoet aan de eerdere opgestelde requirements. De keuze is op een GAT gevallen en niet op een FAT (Functionele Acceptatie Test) omdat Traffic Builders de focus wil leggen op de niet-functionele eisen. Het functioneren van het systeem is uitgebreid aan bod geweest tijdens de inceptiefase.
Zie verder voor de requirements de toelichting over de inceptiefase in hoofdstuk 5.

Disciplines
Binnen RUP kunnen gedurende de looptijd van een project negen disciplines voorkomen (zie figuur 5). De intensiteit waarmee een bepaalde discipline wordt uitgevoerd, is afhankelijk van de fase waarin het project zich bevindt. De negen disciplines zijn verdeeld over twee hoofdgroepen: ontwikkeling en ondersteuning.
De ontwikkelingsdisciplines zijn: Business Modeling, Requirements Engineering, Analysis and Design, Implementation, Test en Deployment.
De ondersteunende disciplines zijn: Project Management, Environment en Configuration & Change Management.
In dit project zijn voornamelijk de ontwikkelingsdisciplines van toepassing, omdat de ontwikkeling van een module centraal staat.

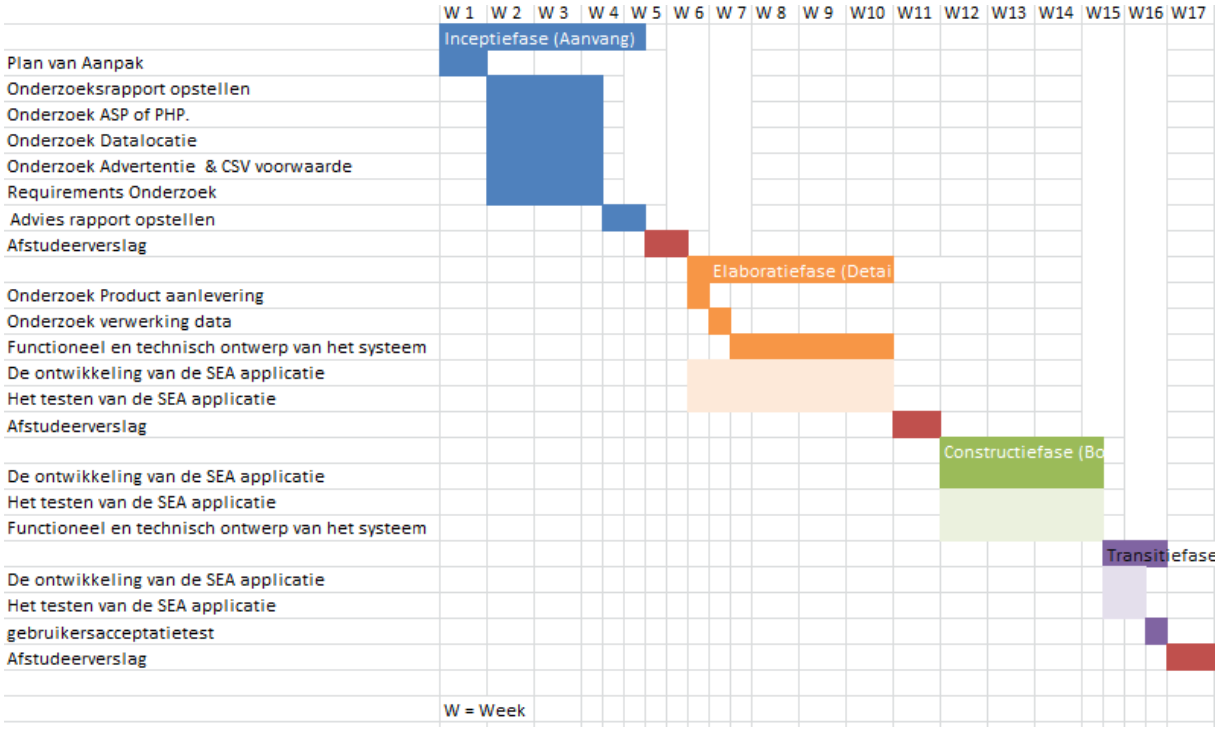


Figuur 5 RUP-diagram (bron ibm.com)

4.2 planning

De grafische weergaven van de achtereenvolgende planningen en de tijdsindeling van het project zien er als volgt uit:

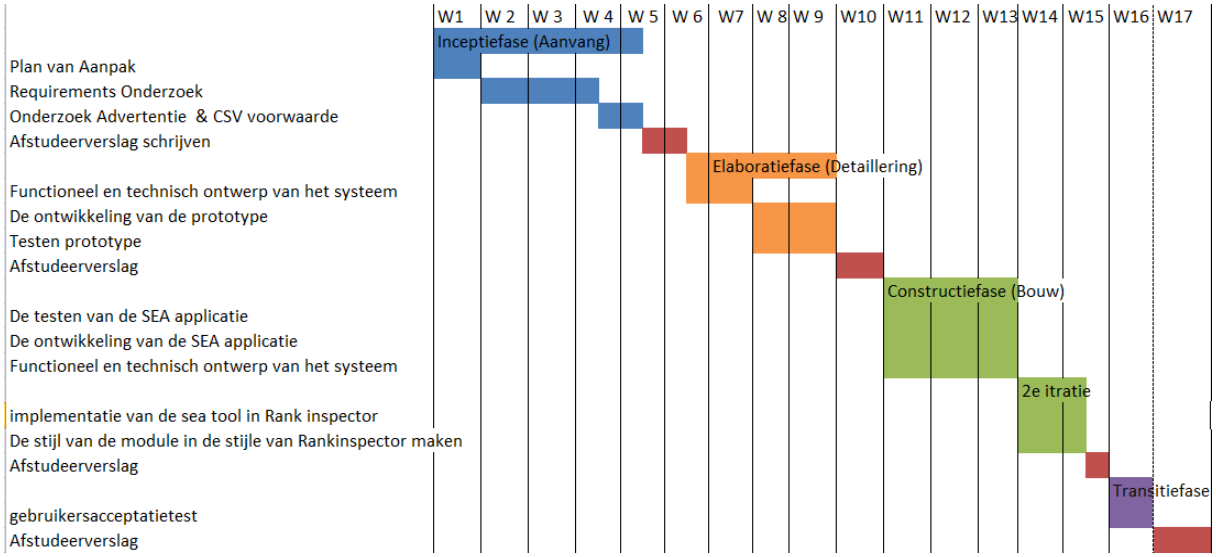
Oorspronkelijk



In de originele planning staan activiteiten die niet uitgevoerd zijn, omdat zij in de loop van het project overbodig bleken te zijn. Afgevallen zijn:

- onderzoeksrapport opstellen;
- onderzoek naar ASP of PHP;
- onderzoek data locatie;
- adviesrapport;
- onderzoek naar verwerking data.

De redenen voor de aanpassingen in de planning worden verderop toegelicht bij de conclusies van elke fase. De nieuwe, uiteindelijke planning ziet er grafisch als volgt uit.



5 Inceptiefase

Tijdens de inceptiefase wordt de haalbaarheid van het project, de inhoud en de begrenzingen bepaald. Aan het einde van de inceptiefase is het oorspronkelijke idee omgezet in een productvisie. De belangrijkste risico's zijn daarmee geïdentificeerd en ingeschat.

5.1 Plan van aanpak

De eerste stap binnen de inceptiefase is het maken van een plan van aanpak. Het plan van aanpak is bedoeld voor de opdrachtgever en de uitvoerder. Door het plan van aanpak is het voor beide partijen duidelijk wat er wel en wat er niet gaat gebeuren.
Voor de aanvang van mijn afstuderen heb ik een afstudeerplan ontwikkeld dat de basis vormde voor het plan van aanpak. Het plan van aanpak bevatte een aantal toevoegingen op het afstudeerplan: de planning, de doelgroepomschrijving, de stakeholders, de risicofactoren en de kwaliteitsborging.

Kwaliteitsborging
Om ervoor te zorgen dat het product voldoet aan de wensen en eisen van de opdrachtgever, de stagebegeleider en de gebruikers zijn de volgende kwaliteitsgaranties opgesteld en uitgevoerd:

- Een regelmatige terugkoppeling naar de stagebegeleider, de opdrachtgever en de gebruikers. Tijdens de terugkoppeling worden de voortgang en het resultaat besproken.
- Controle op het programmeerwerk door de stagebegeleider.
- Inventarisatie van risicofactoren.

Deze punten zijn samen met de stagebegeleider bepaald.
Het complete plan van aanpak is te vinden in bijlage 1.

5.2 Onderzoeken

Nadat de stagebegeleider het plan van aanpak had goedgekeurd, ben ik verder gegaan met de twee onderzoeken.
Het eerste betrof een onderzoek naar de Google Adwords tools die gebruikt worden binnen Traffic Builders. Aanleiding voor dit onderzoek was het advies van mijn stagebegeleider en de SEA-consultants om de te ontwikkelen module te laten aansluiten bij de tools die binnen Traffic Builders al worden gebruikt. Dit onderzoek had als doel om een indruk te krijgen van de tools die momenteel gebruikt worden en de verwachtingen die er van mijn project zijn.
Tijdens dit onderzoek kwam ik complexe termen tegen die om uitleg vroegen. Daarom heb ik besloten deze termen te beschrijven via een terminologieonderzoek, zodat deze voor de rest van het project duidelijk zijn.
In de paragrafen 5.2.1 en 5.2.2 worden de onderzoeken verder uitgediept.

5.2.1 Huidige systemen

Het is belangrijk om de huidige systemen die de SEA-consultants gebruiken te inventariseren, omdat de module die wordt ontwikkeld qua functionaliteit moet lijken op de huidige tools. De tools die Traffic Builders momenteel gebruikt om advertenties op te zetten, zijn O2mc en Adcore. Deze worden gebruikt om grote hoeveelheden advertenties in Google Adwords aan te maken op basis van XML-feeds. Deze tools hebben bepaalde functionaliteiten die ook in de module moeten komen.
Tijdens het inventariseren van de functionaliteiten heb ik deze opgedeeld in verschillende categorieën. Zodoende ontstond structuur in het onderzoek en duidelijkheid in de functionaliteit. Ik heb de functionaliteiten in de volgende vier categorieën ondergebracht:

- Creëren
  alle functionaliteiten die betrekking hebben op het creëren van Adwords campagnes
- Lay-out
  alle functionaliteiten die betrekking hebben op de opmaak van de Adwords tools
- Account
  alle functionaliteiten die betrekking hebben op accounts van de Adwords tools
- Adwords
  alle eisen die Google stelt aan advertenties

Huidige systemen

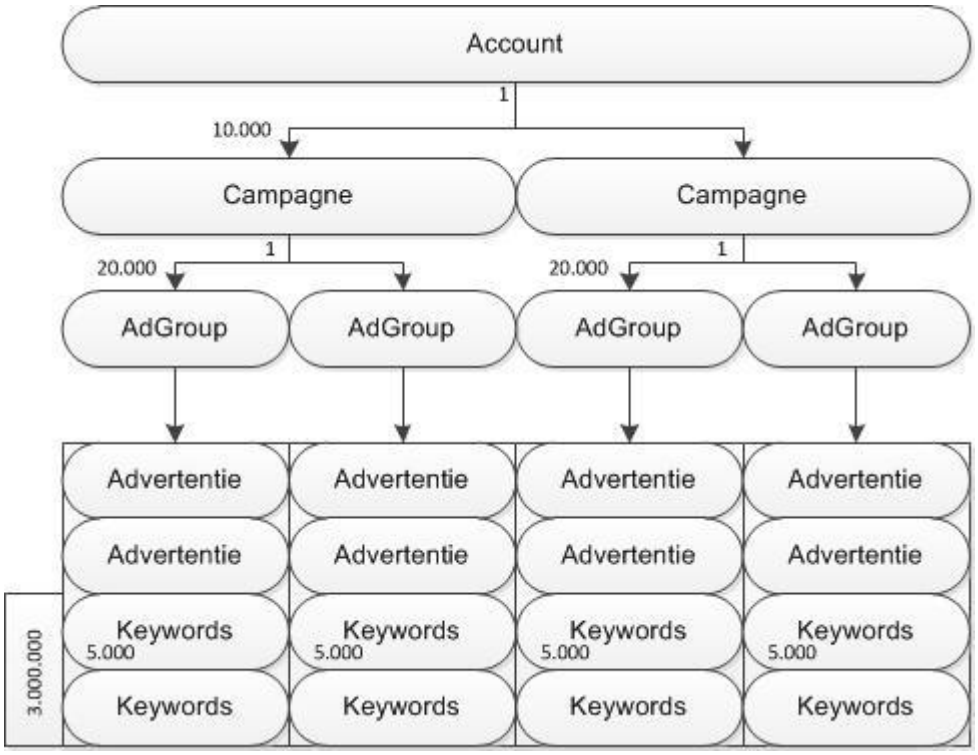
Tabel 1 (samenvatting*)

Nr.	Categorie	Actie	Herkomst functionaliteit
1	Account	Inloggen, aanmelden met gebruikersnaam of e-mailadres en wachtwoord.	Adcore & O2mc
2	Account	Nieuw wachtwoord aanvragen.	Adcore & O2mc
3	Creëren	Het zelf kunnen creëren van nieuwe advertenties door invoegen van XML bestanden.	Adcore
4	Creëren	Het aanmaken van eigen variabelen.	O2mc
5	Creëren	Het maken van nieuwe campagnetemplates met dezelfde xml- file.	Adcore & O2mc
6	Account	Bewerken van account gegevens.	Adcore & O2mc
7	Creëren	Het bewerken van campagnetemplate-instellingen: • campagnenaam aanmaken op basis van variabelen; • budget voor de campagne aangeven; • start- en einddatum aangeven; • status aangeven: actief, gepauzeerd of verwijderd; • taal doelgroep aangeven; • locatie doelgroep aangeven; • aangeven op welke apparaten de advertentie vertoond moet worden: desktop pc's, tablets, smartphones.	Adcore & O2mc

* Zie voor de gehele tabel bijlage 2, hoofdstuk 2.

Structuur van Google Adwords

Figuur 6 geeft een grafisch overzicht van de maximale grootte van Adgroups, campagnes, accounts en advertenties. Deze restricties zijn van belang bij het maken van de CSV-export die later geïmporteerd moet worden in Google Adwords Editor. Worden de maximale aantallen overschreden, dan is de CSV onbruikbaar.



Figuur 6 - Google Adwords structuur en restricties

Tijdens het onderzoeken van de restricties en verplichtingen van Google Adwords heb ik ook ontdekt wat de account structuur is van Google Adwords. De structuur is als volgt:

- Een account kan meerdere campagnes hebben.
- Een campagne kan meerdere Adgroups hebben.
- In één Adgroup kunnen meerdere advertenties zitten met meerdere keywords.

Kortom:

- 1 account mag maximaal 10.000 campagnes hebben.
- 1 campagne mag maximaal 20.000 Adgroups hebben.
- 1 Adgroup mag maximaal 5.000 keywords hebben
- In totaal mogen er niet meer dan 3.000.000 keywords in 1 account zitten.

5.2.2 Terminologie Adwords

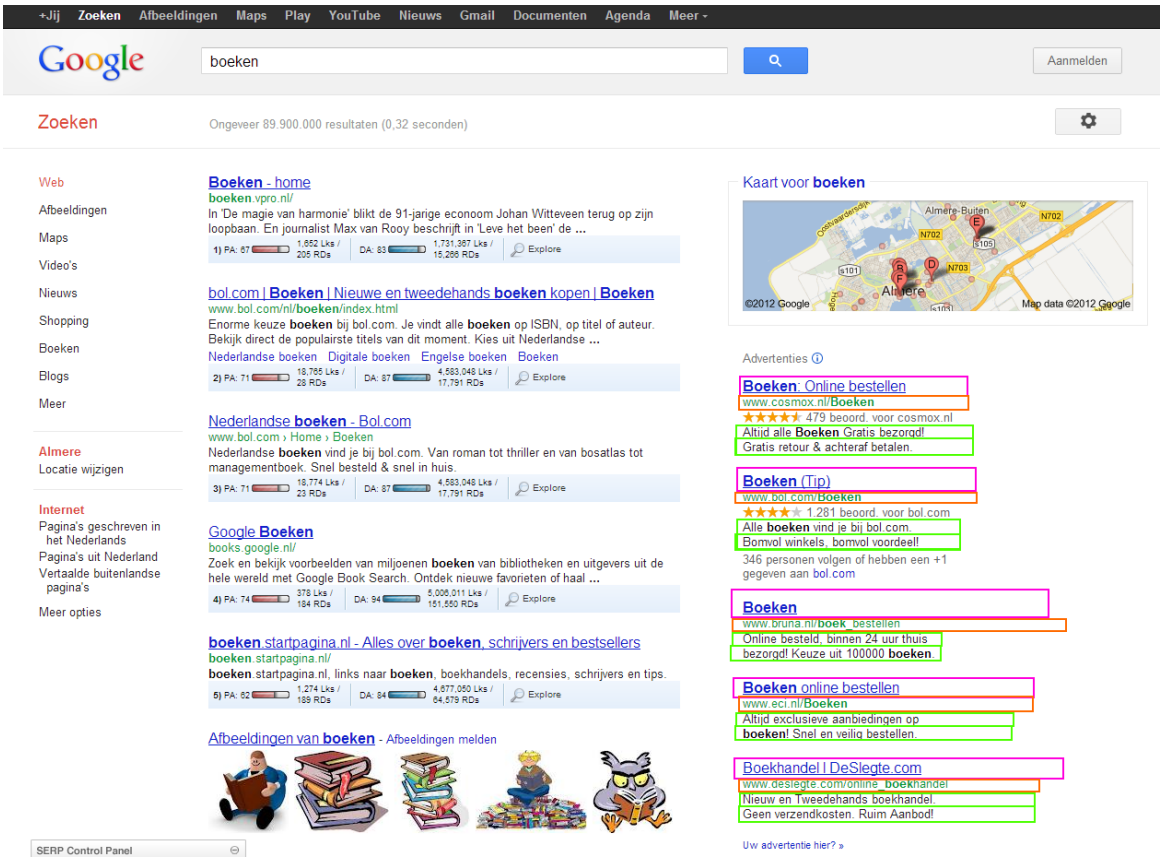
Tijdens het onderzoek naar de bestaande systemen en de structuur van Adwords, is een aantal complexe termen gevonden. Om de betekenis inzichtelijk te krijgen, heb ik een tabel opgesteld met daarin de beschrijvingen van de termen. De uitwerking is te vinden in de definitielijst aan het einde van deze scriptie.

Beeldvorming

Voor een beter begrip van een advertentie in Adwords heb ik twee afbeeldingen gemaakt. In figuur 7 staat een advertentie schematisch weergegeven met de verschillende elementen. In figuur 8 is te zien hoe de advertentie uiteindelijk - in de browser - op het beeldscherm eruitziet. De titel is roze, de display URL oranje en de eerste en tweede omschrijvingsregels zijn groen.

Boeken bestellen	**titel**
www.bruna.nl/**boek_bestellen**	**Display URL**
Bestel online en haal Gratis af in	**Eerste omschrijvingsregel**
de winkel. Keuze uit 100000 **boeken**!	**tweede omschrijvingsregel**

Figuur 7 - Adwords advertentie schematisch



Figuur 8 - Google Adwords advertentie op scherm

5.3 Interviews

De lijst met functionaliteiten uit het vooronderzoek diende als basis voor het opstellen van de interviews met de toekomstige gebruikers.
Ik heb gekozen voor half gestructureerde interviews om een zekere richting te kunnen geven aan de interviewonderwerpen, zonder dat het ten koste zou gaan van de eigen input van de geïnterviewden. Hun input is belangrijk omdat dit inzicht geeft in waar de huidige systemen nog in gebreke blijven. Uit de interviews moeten de belangrijkste requirements gehaald worden. Door middel van de MoSCoW-methode heb ik deze vervolgens geprioriteerd.

Selectie gebruiker-stakeholders
Onder de werknemers is geïnventariseerd wie werkt met Google Adwords. Dit zijn de mensen die zich kwalificeren als stakeholder. Het streven is om drie mensen te interviewen, dertig procent van het totaal aantal SEA-consultants. Ik verwacht dat dat aantal representatief is.

Stakeholders-developer
Ook de developer die de module moet gaan onderhouden, heeft zijn visie gegeven op de module die ontwikkeld gaat worden. Daardoor werd duidelijk wat voor de developer belangrijk is aan het systeem en de ontwikkeling ervan.

Kwaliteit van de interviews
Zodra de interviews waren uitgevoerd en de resultaten geanalyseerd, zijn deze teruggekoppeld naar de geïnterviewden. Op die manier konden de geïnterviewden nog feedback geven.

5.3.1 De interviews

Er zijn drie interviews gehouden met de gebruiker-stakeholders. Twee verliepen zoals gepland, één liep anders dan verwacht. Eerst worden de twee interviews die volgens plan verliepen, besproken. Vervolgens wordt het interview besproken dat anders liep dan verwacht. Daarna volgt een verklaring waarom de laatste anders verliep. Tot slot vat ik het interview met de developer samen.

Interview met Talitha
Als eerste is Talitha geïnterviewd. Talitha is een Adwords consultant bij Traffic Builders en gebruikt O2mc en AdCore. Dit gesprek verliep overzichtelijk. Zij demonstreerde hoe ze de tools gebruikt en wat zij mist. Het belangrijkste punt uit haar verhaal was dat ze het handig zou vinden als de campagnes op haar manier genummerd worden. Momenteel doet ze dit handmatig.
Ook heeft Talitha uitgelegd wat de verschillende soorten matchtypes zijn. Op basis daarvan heb ik besloten dat er voor mijn opdracht duidelijkheid moet komen over de verschillende soorten match types. De verschillende match types zijn te vinden in bijlage 5.
Het complete interviewverslag is te vinden bijlage 3, paragraaf 4.2.

Interview met Martijn
Martijn is als tweede geïnterviewd. Martijn is SEA-consultant en ondersteunt klanten bij hun Adwords campagnes. Hij maakt gebruik van Adcore. Hij heeft mij laten zien hoe hij Adcore gebruikt en wat hij daar handig en onhandig aan vindt.
Eén van de belangrijkste aanbevelingen van Martijn is dat er voor split testen (= onderzoeken welke advertentietekst het meest succesvol is door resultaten van verschillende versies te vergelijken) meerdere advertenties in één Adgroup toegevoegd moeten kunnen worden.
Verder vindt Martijn het lastig dat hij een te lange advertentie alleen kan inkorten door de string van links of rechts te verkleinen. Als hij bijvoorbeeld gebruik maakt van de variabele [$hotel], kan daar een waarde in staan die langer is dan de toegestane veldwaarde van Google. Op dit moment staat de software toe dat hij dan van links of van rechts af verkleint. Dit leidt soms tot vreemd uitziende advertenties. Hij zou het prettig vinden als hij meerdere mogelijkheden op kan geven en de nieuwe module de passende waarde selecteert.
Het complete interviewverslag staat in bijlage 3, paragraaf 4.2.

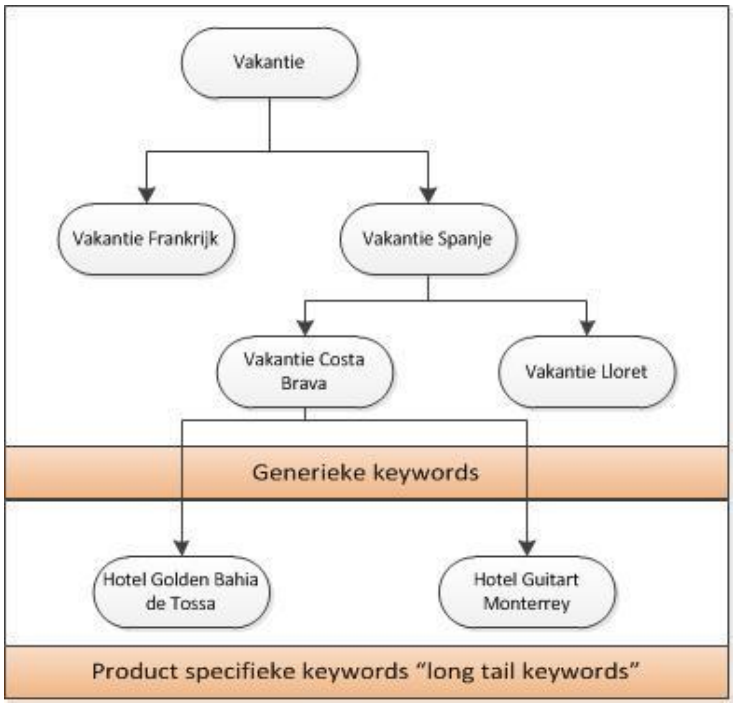
Interview met Wolter
Wolter interviewde ik als derde vanwege zijn vakantie maar had ik als eerste willen interviewen. Hij is namelijk niet alleen één van de toekomstige gebruikers, maar ook de opdrachtgever van dit project. Maar zijn vakantie zorgde niet voor vertraging van het project, omdat ik eerst de anderen heb geïnterviewd.

Wolter is eigenaar en oprichter van Traffic Builders. Hij heeft uitgesproken ideeën over de nieuwe module. Die wijken volgens hem erg af van wat ik tot dan toe te horen had gekregen. Naar zijn mening – en in afwijking van alle andere betrokkenen – moet de nieuwe module namelijk anders zijn dan de O2mc of Adcore tool. Mijn eerder opgestelde vragen waren hierdoor niet meer bruikbaar en daarom heb ik in plaats van een half gestructureerd interview een open interview gehouden.

De twee belangrijkste punten uit het interview met Wolter zijn:

- Hij wil geen module voor product specifieke keywords maar voor generieke keywords. Het verschil tussen die twee is weergegeven in figuur 9. De tools Adcore en O2mc concentreren zich op de productspecifieke of long tail keywords. Wolter wil juist een module voor de generieke keywords.
- In de nieuwe module moet de gebruiker 2 verschillende woorden kunnen invoeren, root keywords en bijwoorden; nu kan dat nog niet. De nieuwe module moet deze twee woorden kunnen combineren tot generieke keywords. Als je bijvoorbeeld als root keyword ‘vakantie’ opgeeft en de bijwoorden zijn ‘Spanje’ en ‘Frankrijk’, dan moet de module daar twee generieke keywords van kunnen maken: ‘vakantie Spanje’ en ‘vakantie Frankrijk’. Deze keywords moeten vervolgens gebruikt kunnen worden om in Adwords mee te adverteren. Zoekt iemand in Google op één van die twee generieke keywords, dan moet de daarbij horende advertentie worden getoond.

Het volledige interview met Wolter is te vinden in bijlage 3, hoofdstuk 4.2.



Figuur 9 – Generieke keywords en long tail keywords

Interview met Tuncay

Tuncay is de enige developer van Traffic Builders. Hij werkt er al vanaf de oprichting van Traffic Builders in 2001. Indertijd was Tuncay de ontwikkelaar van Rankinspector, de eigen tool van Traffic Builders. Alleen al omdat hij de enige developer is, mag hij niet bij de geïnterviewden ontbreken. Hij zorgt immers voor het onderhoud van de nieuwe module van Rankinspector als mijn afstudeerperiode voorbij is. Het moet hem dan ook makkelijk gemaakt worden aanpassingen aan te brengen.

Het belangrijkste uit dit interview is dat de module in PHP ontwikkeld moet worden, net als Rankinspector. De module moet namelijk gebruik gaan maken van de standaard functionaliteiten van Rankinspector, zoals het gebruikersbeheersysteem.
Tijdens het opstellen van het Plan van Aanpak was het nog de bedoeling dat de nieuwe Google Adwords Module een standalone applicatie zou worden. Maar tijdens het interview met Tuncay werd duidelijk dat het een module van Rankinspector moest worden.
Het complete interview is te lezen in bijlage 3, paragraaf 4.2.

5.4 Requirements

Met de interviews is alle informatie verzameld om de requirements in kaart te brengen. De functionele requirements zijn gehaald uit de volgende producten:

- interviews met de gebruikers
- interview met de developer

De requirements die uit de documenten zijn opgemaakt, worden opgeschreven als user story's. De requirements worden hieronder opgesomd en geprioriteerd met de MoSCoW-methode.

5.5 De enquête

MoSCoW-methode

De prioriteit van de verschillende requirements wordt in deze paragraaf bepaald op basis van de MoSCoW-methode. Elke stakeholder brengt daartoe een prioriteit aan in de requirements. Om te voorkomen dat de stakeholders elke requirement even belangrijk vinden, moet bij 60 procent van de requirements "must have" worden opgegeven, bij 25 procent "should have", bij 10 procent "could have" en bij 5 procent "won't have". Deze percentages zijn in samenspraak met de stagebegeleider bepaald als richtlijnen voor de gebruikers.
Er is gekozen voor MoSCoW omdat ik vanuit school veel ervaring heb met deze methode en men bij Traffic Builders ook bekend is met deze methode.

De groep stakeholders-gebruikers bestaat uit meerdere personen. Deze mogen allemaal de enquête invullen en prioriteiten aangeven. Aan de hand daarvan wordt bepaald welke requirements het belangrijkst zijn. De uitwerking zal er als volgt uitzien:

- Als een gebruiker aangeeft dat een requirement een "must have" moet zijn, krijgt dit requirement drie punten, bij "should have" twee punten, bij "could have" één punt en bij "won't have" geen punten.
- Vervolgens worden per requirement de punten opgeteld. Daarna worden de eerste 60 procent van de requirements met de meeste punten als "must have" bestempeld, de volgende 25 procent met "should have", de 10 procent daarna met "could have" en de laatste 5 procent met "won't have".

De enquête is te vinden in bijlage 4.

In overleg met de developer is over de onderstaande requirements besloten dat dit randvoorwaarden zijn in plaats van requirements. Daarom worden deze niet geprioriteerd met MoSCoW, maar worden zij wel toegepast. Tijdens de inceptionfase zal blijken dat het eigenlijk technische beperkingen zijn (zie paragraaf 6.1).

De volgende requirements zijn randvoorwaarden:

NR#	Requirements
1	De module moet in PHP worden ontwikkeld.
2	De DBMS die gebruikt moet worden is MySQL.
3	De module moet een webapplicatie worden.
4	De module moet als module in Rankinspector passen.
5	Voor alle database-handelingen moet de database klassen van Rankinspector gebruiken.
6	Voor het genereren van de html moet x template gebruikt worden.

De resultaten van de prioritering

Tabel 2

NR#	Requirements	MoSCoW prioriteit
1	Als gebruiker wil ik kunnen kiezen voor 4 verschillende matchtypes.	Must have
2	Als gebruiker wil ik nieuwe projecten stapsgewijs kunnen aanmaken.	Must have
3	Als gebruiker wil ik dat de stappen tussentijds worden opgeslagen.	Must have
4	Als gebruiker wil ik XML-bestanden kunnen uploaden.	Must have
5	Als gebruiker wil ik eenvoudig XML-elementen kunnen koppelen aan variabelen.	Must have
6	Als gebruiker wil ik variabelen kunnen plaatsen in advertentietemplates	Must have
7	Als gebruiker wil ik de variabelen zelf een naam kunnen geven.	Must have
8	Als gebruiker wil ik meerdere advertentietemplates kunnen maken.	Must have

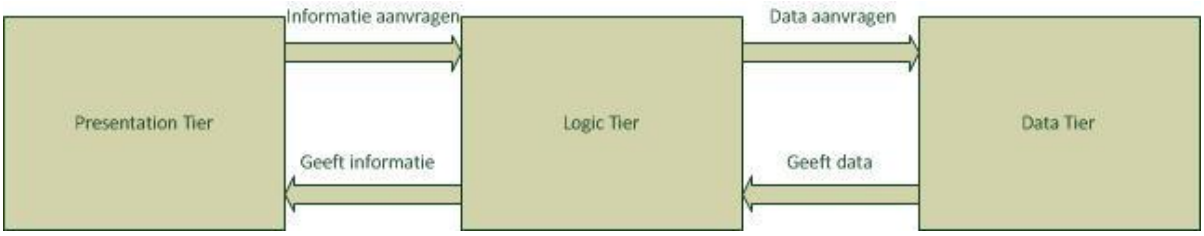
9	Als gebruiker wil ik producten uit een XML kunnen uitsluiten op basis van woorden.	Could have
10	Als gebruiker wil ik vóór de campagnes kunnen bepalen wat het budget is en wat de maximale kosten per klik mogen zijn.	Could have
11	Als gebruiker wil ik alle keywords kunnen uitsluiten door aan te geven wat de minimale en maximale grootte mag zijn van een keyword.	Should have
12	Als gebruiker wil ik meerdere mogelijkheden kunnen opgeven voor de titel, de beschrijvingen en de display url van de advertenties. De mogelijkheid die past binnen de gestelde limieten van Google moet worden gebruikt.	Must have
13	Als gebruiker wil ik per (generiek) keyword kunnen aangeven welke matchtypes daarbij horen.	Must have
14	Als gebruiker wil ik generieke keywords kunnen invoeren.	Must have
15	Als gebruiker wil ik bijwoorden kunnen toevoegen.	Must have
16	Als gebruiker wil ik standaardwaarden kunnen aangeven per project.	Could have
17	Als gebruiker wil ik per generiek keyword kunnen aangeven of dit een aparte campagne moet worden.	Should have
18	Als gebruiker wil ik per generiek keyword kunnen aangeven wat er maximaal per klik betaald mag worden.	Won't have
19	Als gebruiker wil ik per generiek keyword kunnen aangeven wat zijn destination url is.	Should have

Voor de meeste requirements geeft minimaal één gebruiker aan dat hij/zij deze belangrijk vindt. Er is één requirement dat de gebruikers onbelangrijk vinden: requirement 18. De rest heeft het kenmerk "must have", "should have" of "could have".

5.6 Architectuur

Van Rankinspector ontbreekt de documentatie die voor dit afstudeerverslag van belang is. Daarom heb ik zelf de architectuur van Rankinspector in kaart gebracht, evenals de communicatie tussen Rankinspector en Google Adwords.
Hoe het nieuwe systeem moet worden en hoe het huidige systeem werkt, is uitgezocht tijdens de inceptiefase en deels tijdens elaboratiefase. Om het overzichtelijk te houden heb ik er hieronder één geheel van gemaakt.

Architectuurstijl
Rankinspector wordt aangeboden via het internet door het hosten van de tool als een website. Daardoor is Rankinspector wereldwijd te benaderen. De architectuur van Rankinspector bestaat uit een 3-tier architectuurstijl. De 3-tier architectuurstijl is grafisch weer gegeven in figuur 10.



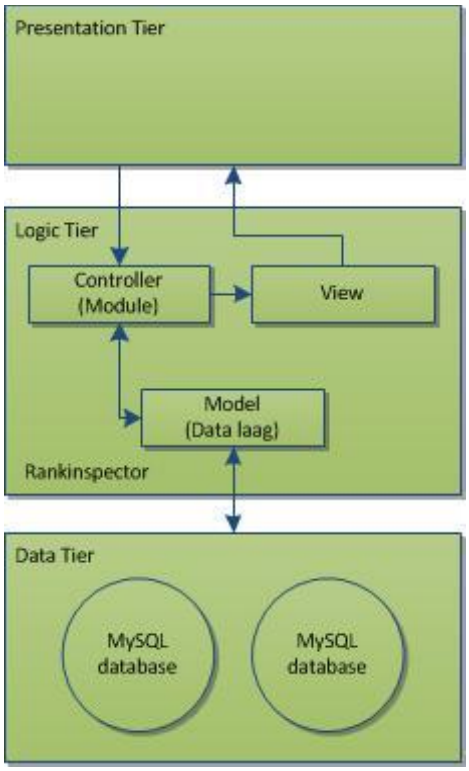
Figuur 10 - Architectuurstijl

Presentation Tier
De Presentation Tier is de toplaag van deze architectuurstijl. In deze laag wordt de informatie getoond aan de gebruiker. Door middel van deze laag kan een bezoeker één van de modules van Rankinspector raadplegen.

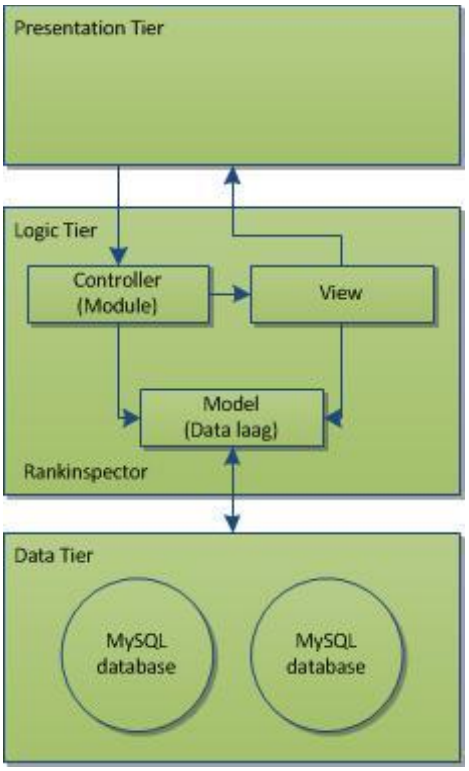
Logic Tier
De Logic Tier is de middelste laag in deze architectuurstijl. In deze laag zit alle logica van Rankinspector. Deze laag handelt de 'request' af van de bezoekers en communiceert met de Data Tier. Deze laag bevat ook alle Business Logic in modules, waarmee de verschillende functionaliteiten van Rankinspector gescheiden worden. In deze laag wordt de nieuwe module toevoegd met nieuwe Business Logic.

Data Tier
Deze laag is verantwoordelijk voor het bewaren en verwerken van data. Op wat voor manier de data verwerkt moet worden en in welke structuur dit moet gebeuren, wordt bepaald door de Logic Tier.

MVC
Rankinspector maakt gebruik van het MVC-model. Traffic Builders heeft dit op zijn eigen manier geïmplementeerd. De controller maakt het model, vraagt de data aan het model en wijst de plek in de view aan waar de data moeten komen staan.
Als Traffic Builders volgens de standaard zou implementeren, dan had de controller het model opgehaald en doorgegeven aan de view. De view had dan zelf bepaald wat waar komt te staan. De verschillen tussen beide stijlen zijn grafisch weergegeven in figuur 11.a en 11.b.



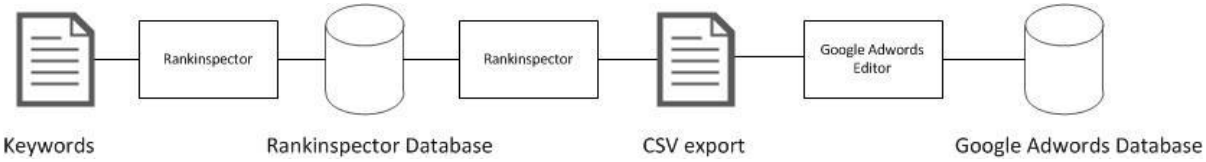
Figuur 11.a – MVC-architectuurstijl zoals toegepast binnen Rankinspector



Figuur 11.b – MVC-architectuurstijl zoals standaard gedefinieerd

Aanpassingen
De toevoegingen voor de nieuwe module zullen worden uitgevoerd in de Data Tier en in de Logic Tier. Er wordt namelijk een module met controllers gemaakt en een nieuwe MySQL database toegevoegd.

Communicatie tussen Rankinspector en Adwords
De communicatie tussen Rankinspector en Google Adwords verloopt via het exporteren en het importeren van een CSV-bestand. In figuur 12 is te zien hoe de data flow verloopt.
Er is bewust voor gekozen om gebruikers zelf de CSV uit Rankinspector te laten exporteren en vervolgens te laten importeren in de Google Adwords Editor.
Een andere manier om de data in de Google Adwords database te krijgen is de Google Adwords API. Maar aan het gebruik van deze API zitten kosten verbonden per mutatie, aanvraag en update. Omdat één van de doelen van dit project is kosten te besparen, is in samenspraak met de stagebegeleider en de opdrachtgever voor de CSV-exporteermethode gekozen. Het gebruik van de Google Adwords Editor is namelijk kosteloos.



Figuur 12 - Overzicht data flow CSV export/import

5.7 Conclusie inceptiefase

In deze paragraaf staan de belangrijkste bevindingen van de inceptiefase.

Van standalone naar module
Oorspronkelijk was het de bedoeling dat de module als standalone applicatie zou worden ontwikkeld. Tijdens het interview met de Tuncay Oner en de opdrachtgever werd duidelijk dat het niet een standalone applicatie moest worden, maar een module van Rankinspector

Onderzoeksrapport
Het voorgenomen tussentijdse onderzoeksrapport is niet opgesteld. In eerste instantie dacht ik dat het belangrijk zou zijn een overkoepelend document te schrijven voor de vier onderzoeken die ik ging uitvoeren. Maar uiteindelijk werden het twee onderzoeken met elk een slothoofdstuk. Een extra document zou meer van hetzelfde geweest zijn.
Dit geldt ook voor het adviesrapport. In beide onderzoeken worden de vervolgstappen in de conclusie vermeld, waardoor een apart adviesrapport overbodig is.

Keuze programmeertaal
Uit de interviews met de opdrachtgever en de developer was duidelijk geworden dat PHP hoe dan ook als ontwikkeltaal gebruikt moest worden, omdat beide stakeholders aangaven dat de nieuwe applicatie die ontwikkeld moet worden een module van Rankinspector moest worden. Onderzoek naar de beste programmeertaal werd daardoor overbodig.

6 Elaboratiefase

De tweede fase binnen RUP is de elaboratiefase. Het merendeel van de functionele requirements (use cases) wordt in deze fase gespecificeerd en de systeemarchitectuur wordt ontworpen.

6.1 Uitbreiding requirements

De belangrijkste aanpassing in deze fase aan het document met requirements betrof het indelen in drie verschillende categorieën: functionele requirements, niet-functionele requirements en technische beperkingen.
Volgens Nicole de Swart moet er onderscheid worden gemaakt tussen een functionele requirement en een gewenste systeemfunctie [Nicole de Swart boek]. Een functionele requirement stelt een eis aan het gedrag van een systeem, terwijl een systeemfunctie dat gedrag zélf representeert: “Een systeemfunctie zet invoer van het systeem om in uitvoer. De actie die het systeem hiervoor uitvoert is de systeemfunctie ofwel het gedrag van het systeem. Een traditioneel functioneel ontwerp is een ontwerp van de gewenste (niet-technische) systeemfuncties en de benodigde systeeminvoer en -uitvoer.”

Functionele requirements
In tabel 3 hieronder staan alleen de nieuwe requirements geplaatst. De volledige lijst requirements en de indeling zijn te vinden in bijlage 7, hoofdstuk 2.
De requirements zijn gehaald uit de volgende documenten:

- **V** = Vooronderzoek
- **G** = Interviews met gebruikers
- **D** = Interview met developer

Nieuwe requirements

Tabel 3

NR#	Requirements	Bron	MoSCoW prioriteit
#21	Als gebruiker wil ik een helpfunctie ter ondersteuning.	V	Should have
#22	Als medewerken van Traffic Builders wil ik de gebruikers kunnen beheren.	D	NVT zie wijziging 3
#23	Als gebruiker wil me kunnen aanmelden met een accountnaam en password.	D	NVT zie wijziging 3
#24	Als gebruiker wil ik advertenties kunnen exporteren en deze vervolgens kunnen importeren in de Google Adwords editor.	D & V	Must have
#25	Als gebruiker wil ik dat advertenties automatisch genummerd kunnen worden.	G	Should have

Niet-functionele requirements
De niet-functionele requirements beschrijven de kwaliteit van Google Adwords Module. Voor de inventarisatie zijn de opdrachtomschrijving, het vooronderzoek, de interviews met de gebruikers en het interview met de developer nog eens doorgenomen.
Dit zijn de niet-functionele requirements die daaruit naar voren kwamen:

- Als gebruiker wil ik dat de module beschikbaar is voor meerdere klanten tegelijk
- Als developer wil ik in de toekomst eenvoudig meerdere bestandstypes voor het importeren van producten kunnen toevoegen

Technische beperkingen
De technische beperkingen en requirements stonden eerst door elkaar. Na het reviewen van de requirements werd het verschil tussen beide duidelijk. De eerste versie van de requirements uit de inceptiefase heb ik voorgelegd aan de opdrachtgever en de developer. Uit hun feedback bleek het belangrijk te zijn om verschil te maken tussen technische beperkingen en requirements: een technische beperking is een verplichting in het kader waarin de module ontwikkeld moet worden en een requirement is een wens of eis aan het systeem.

Technische beperkingen

Tabel 4

NR#	Requirements	Bron
#T1	De module moet in PHP worden gebouwd.	D & V
#T2	De DBMS die gebruikt moet worden is MySQL.	D & V

#T3	De module moet een webapplicatie worden.	D & V
#T4	De module moet als module in Rankinspector passen.	D
#T5	Voor alle databasehandelingen moet de databaseklasse van Rankinspector gebruikt worden.	D
#T6	Voor het genereren van de html moet x template klasse van Rankinspector gebruikt worden.	D

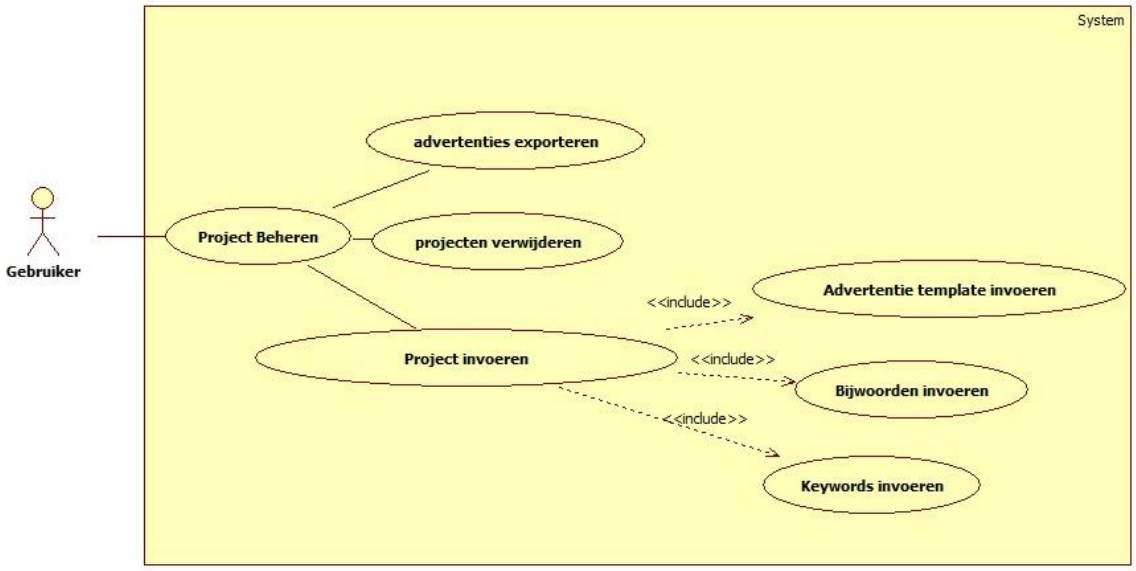
- **V =** Vooronderzoek
- **D** = Interview developer

6.2 Use cases

Op basis van de requirements heb ik een use case diagram en use case omschrijvingen gemaakt, zodat duidelijk wordt hoe de requirements gaan werken. Daarna heb ik de use case views gemaakt. De use case views zijn bedoeld om de communicatie met gebruikers, opdrachtgever en developer eenvoudig en snel te laten verlopen.

6.2.1 Use case diagram

In figuur 13 staat de belangrijkste use case, die met de actor-gebruiker. In bijlage 11 staat een use case waarin ook de beheerder is verwerkt.



Figuur 13 - Use Case Actor-Gebruiker

6.2.2 Use case beschrijvingen

Voor elke use case heb ik een omschrijving gemaakt. In tabel 5 staat de omschrijving voor de use case ‘keywords invoeren’. Deze heb ik hier weergegeven, omdat keywords invoeren een belangrijk onderdeel is van de module.
Elke use case heeft een nummer gekregen en een omschrijving van de actor die hem mag gebruiken. Ook staan de aannames, trigger, flow, alternate flow, requirementnummers en het resultaat erin. De rest van de use cases is te vinden bijlage 11, hoofdstuk 3.

Tabel 5

Naam	Keywords invoeren
Nummer	#7
Omschrijving	De actor wil keywords toevoegen
Actor	Gebruiker, beheerder
Aanname	• Actor is ingelogd. • Actor heeft een project aangemaakt en wil daaraan keywords toevoegen.
Trigger	Gebruiker selecteerde dat hij keywords wil toevoegen.
Flow	1) De actor kiest één van drie manieren om data in te voeren. Daarvoor zijn de volgende keuzes beschikbaar: • Een invoerveld waarin de XML- of CSV-data met de keywords moeten worden geplaatst. • Een input field waar de url geplaatst kan worden naar de XML- of CSV-data. • Een button waar de actor op kan klikken om een XML of CSV bestand te uploaden. 2) De actor selecteert wat voor type data het betreft, XML of CSV. 3) De actor klik op 'volgende'. 4) De actor moet aangeven welk element het keyword is, welk element de destination url is en welk element het 'Cost per Click' (CPC) is. Mochten er in de XML of CSV geen data staan voor de destination url of maximale CPC, dan kan deze erachter als default waarde worden opgegeven. 5) De actor kan extra elementen koppelen, zodat deze later bij het creëren van advertentietemplates gebruikt kunnen worden als variabelen (zie use case #9).
Alternate Flows	-
Requirements nr.	4, 5, 6, 7, 9, 13, 14, 18, 19, 20
Resultaat	De keywords zijn ingevoerd. De actor heeft drie mogelijkheden: • Hij kiest voor annuleren en er wordt niets opgeslagen; • hij kiest voor opslaan en het project wordt opgeslagen; • hij kiest voor 'volgende' en gaat verder met use case #8.

6.2.3 Use case views

Om communicatie met gebruikers, opdrachtgever en developer eenvoudig en snel te laten verlopen, heb ik views gemaakt van de hoofdfunctionaliteiten. Deze views zijn gebaseerd op use cases. De use cases omschrijvingen zijn omgezet naar 5 views, zodat het invoerproces tussentijds opgeslagen kan worden en een gebruiker in een later stadium de draad weer kan oppakken.

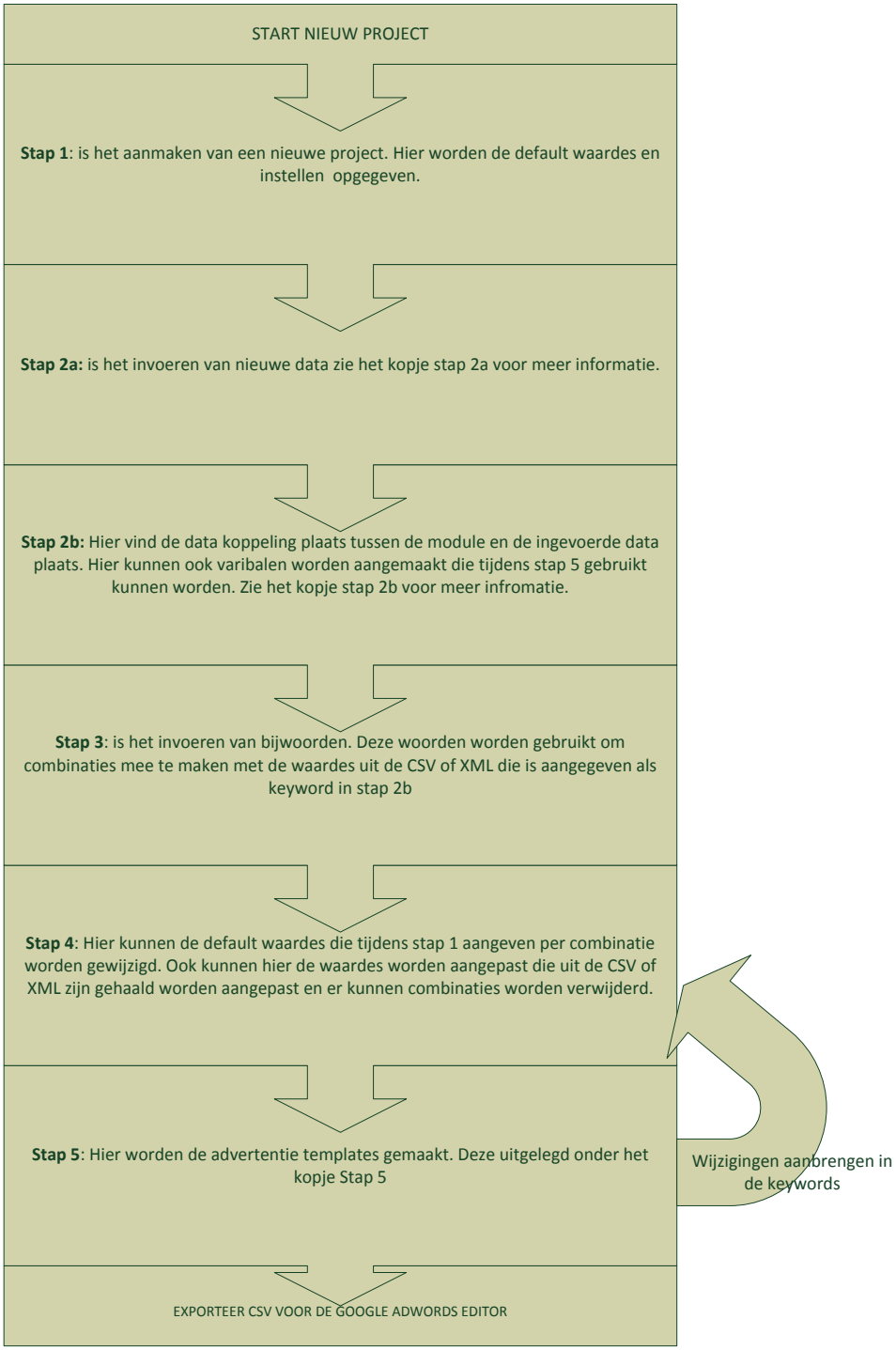
Het tussentijds opslaan is een requirement en de verdeling in 5 stappen is in samenspraak met de opdrachtgever en stagebegeleider tot stand gekomen. Bij elke view staat de herkomst beschreven. Ik heb ervoor gekozen meerdere views uit het document use case in mijn scriptie te plaatsten, omdat het belangrijk is voor het verloop van het afstudeerproject.
De meest cruciale views voor de module zijn hieronder weergegeven. De overige views staan in bijlage 11.

Figuur 14 geeft een overzicht van de 5 stappen die gezet moeten worden voordat de CSV voor de Google Adwords Editor geëxporteerd kan worden.

Er is gekozen om views te maken in plaats van een Mockup, omdat ik sneller een view-teken dan html schrijf en er op korte termijn besluiten moesten worden genomen over de functionaliteit. Zo werd vertraging voorkomen.

Twee stappen 2
In figuur 14 valt op dat er twee stappen 2 zijn: 2a en 2b. Deze horen bij elkaar. Tijdens stap 2a worden de data ingevoerd en tijdelijk opgeslagen. Tijdens stap 2b worden de data gekoppeld en in de database opgeslagen.

De stappen 1, 2b, 3, 4 en 5 kunnen tussentijds worden opgeslagen zodat er in een later stadium verder gegaan kan worden; maar dat is bij stap 2a dus niet het geval. Als de gebruiker na stap 2a besluit te stoppen, gaat stap 2a verloren, omdat de data hier alleen tijdelijk worden opgeslagen.



Figuur 14 - Stappen

Hierna staan 2 van de 5 belangrijkste views voor het creëren van advertenties beschreven.

Stap 2a:
Hieronder is het invoeren van data weergegeven. Deze view is gebaseerd op use case #7. Deze view is gewijzigd tijdens het reviewen van het prototype (zie paragraaf 6.4). Daarom staan hieronder 2 versies: de originele en de versie na het reviewen van het prototype.

Origineel

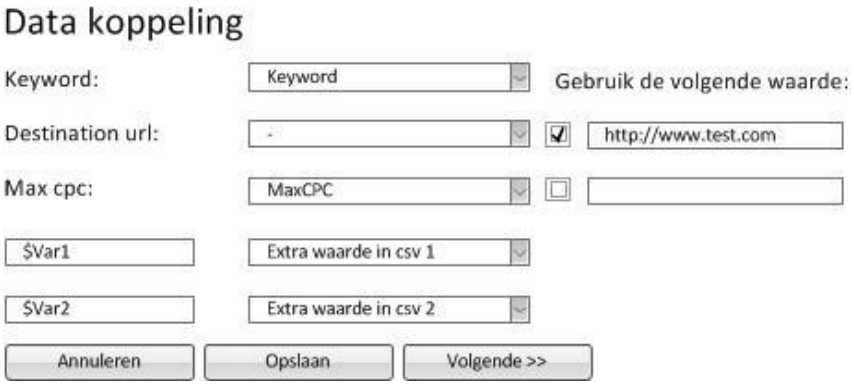
Figuur 15.a – Origineel

Na het reviewen van het prototype

Figuur 15.b - Na review prototype

In figuur 15.a worden alle mogelijkheden weergegeven, in figuur 15.b alleen de geselecteerde.

Stap 2b:
Hieronder staat weergeven hoe de data gekoppeld worden. Deze view is gebaseerd op use case #7.



Figuur 16 - Stap 2b

In dit voorbeeld wordt ervan uitgegaan dat er een CSV is ingevoerd met 2 waardes per regel (oftewel twee rows). Omdat er 2 waardes per regel zijn ingevoerd, komen er 2 regels bij waarin gebruikers variabelen kunnen aanmaken (zie figuur 16). Deze variabelen kunnen vervolgens gebruikt worden tijdens het creëren van de advertentietemplate. In de tabel hieronder staat een voorbeeld van hoe een CSV omgezet zou kunnen worden.

Voorbeeld van een CSV met mogelijke datakoppeling

Tabel 6

	keyword	cpc	dsUrl	land
Rij 1	Den Haag	0.5	http://wwww.testurl.nl/den-haag	Nederland
Rij 2	Brussel	1	http://wwww.testurl.nl/brussel	België
Data Koppeling	Dit is het keyword	Dit is de CPC	Dit is de destination url	Deze row uit de CSV zou je als extra variabelen kunnen gebruiken dan type je bij extra variabelen bijvoorbeeld 'land' in. Als je dan tijdens stap 5 in de advertentie [land] invult komt daar het land te staan

Coach	18.000			
coaching	5.400			
Jobcoach	1.600			
coachen	1.300			

Een gebruiker kan opgeven wat zijn keyword is, zijn destination url en wat de max CPC is. Mocht de url of max CPC niet aanwezig zijn in de CSV, dan kan de gebruiker een standaard waarde invoeren. Dit wordt gedaan door het tekstveld ernaast in te vullen.

Stap 5:
Hieronder is weergegeven hoe de advertentietemplates worden ingevoerd. Deze view is gebaseerd op de use case #9.

Figuur 17 - Advertentietemplates

Een advertentie bestaat in Google uit 5 verschillende onderdelen: Headline, Description Line 1, Description Line 2, Display URL en Destination URL (zie figuur 17). Deze 5 onderdelen moeten worden ingevuld. Hierbij kan gebruik worden gemaakt van de variabelen uit stap 2b (figuur 16).
Te allen tijde kan gebruik worden gemaakt van [keyword], maar een gebruiker kan zelf ook variabelen aanmaken, zoals is uitgelegd bij stap 2b. Een voorbeeld van een advertentietemplate staat hieronder.
In dit voorbeeld zijn [HotelNaam] en [Plaats] variabelen die tijdens stap 2b zijn toegevoegd.
In figuur 18 staat een voorbeeld van een mogelijke uitvoer.

Headline:	Boek het [keyword]!
Display URL	www.booking.com/[HotelNaam]
Description Line 1	Dit hotel is een top hotel in [Plaats]
Description Line 2	Reserveer online tegen de laagste prijzen.

Boek het **Hotel**
www.booking.com/**Mooi-Hotel**
Dit hotel is een top hotel in Parijs
Reserveer tegen de laagste prijzen

Figuur 18 - Voorbeeld uitvoer

Alleen stap 4 en 5 kunnen worden gewijzigd

In figuur 14 is te zien dat bijna alle stappen één richting uit gaan, omdat er geen data aan een bestaand project toegevoegd kunnen worden. Hier is in overleg met de stagebegeleider voor gekozen, zodat de gebruiker tijdens stap 2b zelf de paramaters kan bepalen uit de CSV of XML. Dit betekent dat, als de gebruiker data zou willen toevoegen, de structuur van de CSV of XML exact hetzelfde moet zijn.

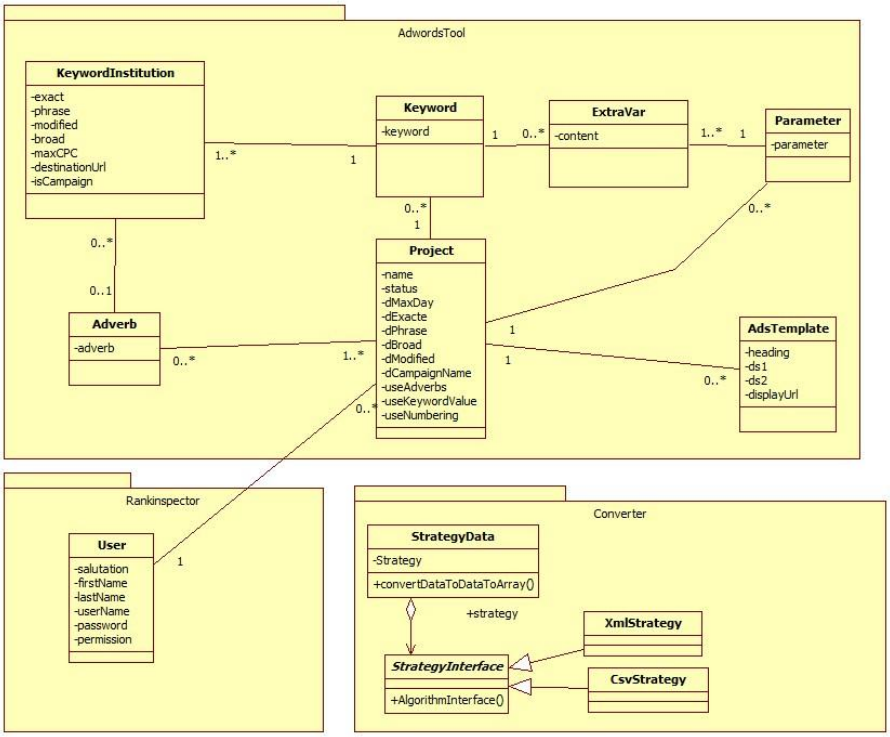
Feedbackgesprek
Toen de eerste versie van de use case views klaar was, heb ik deze voorgelegd aan de stagbegeleider en opdrachtgever, met de bedoeling te achterhalen of de views juist waren. De feedback is te vinden in bijlage 11, hoofdstuk 5.

6.3 Systeem- en databaseontwerp

Het ontwikkelen van het systeem- en databaseontwerp ben ik begonnen met een vertaling van de requirements naar een statisch structuurdiagram. Ik heb daarvoor een klassendiagram gebruikt. Dit diagram beschrijft de klassen, hun attributen, activiteiten (of methoden) en de relatie tussen de klassen van de te ontwikkelen module. Vervolgens heb ik het klassendiagram vertaald naar een databaseontwerp met behulp van ERD (Entity-Relationship Diagram). Het ERD zorgt voor de grafische representatie van het conceptuele datamodel. Het is een visuele weergave van de entiteiten en relaties.

6.3.1 Klassendiagram

Nadat de opdrachtgever de use cases had goedgekeurd, kon het ontwerpen van de applicatie worden vervolgd met het maken van een klassendiagram. In figuur 19 is een compacte versie te zien van het volledige diagram uit bijlage 8. In die bijlage staan ook de Operations (oftewel functionaliteiten) vermeld. Deze zijn hieronder voor de overzichtelijkheid weggelaten.



Figuur 19 - Klassendiagram

Toelichting op de begrippen in het klassendiagram

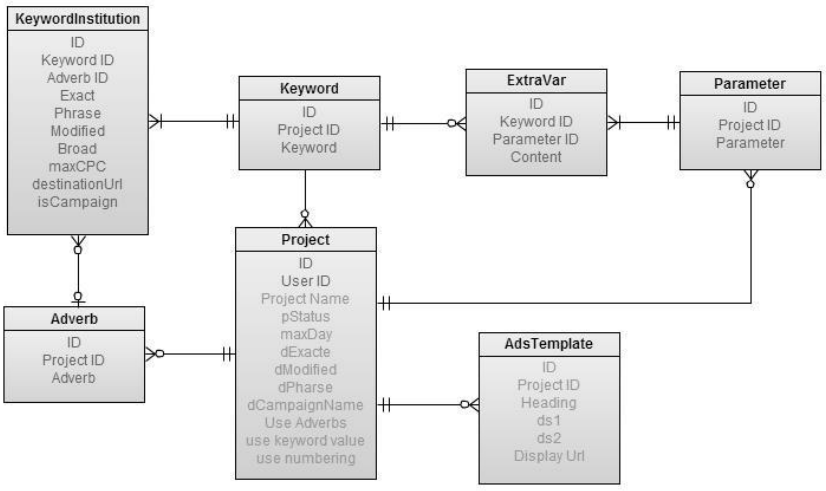
- **Project:** Het project is verantwoordelijk voor de default waarde van het project, het bijhouden van de opties die zijn geselecteerd en de toepassing tijdens het creëren van het project. Het project word tijdens stap 1 gecreëerd. Het lijkt er hier op dat exact, phrase modified, broad, max CPC, destinationUrl en isCampaign dubbel zijn. Maar dat is slechts schijn, want de gegevens die in het project staan, zijn de default waarde voor de Keyword Institution.
- **Keyword:** Dit is een main keyword van een project. Deze word aangemaakt tijdens stap 2a.
- **Adverbs:** Dit is een bijwoord dat is ingevoerd. Bijwoorden worden tijdens stap 3 toegevoegd.

- **KeywordInstitutions:** Dit zijn combinaties van keywords en adverbs. Deze combinaties worden aangemaakt na stap 3. Tijdens stap 4 kunnen combinaties worden verwijderd. Daarom is het noodzakelijk dat combinaties expliciet worden opgeslagen.
  Omdat tijdens stap 4 per keyword ook wijzigingen aangebracht kunnen worden, moet worden opgeslagen of het een campagne is, een max CPC, een destinationUrl, een exact match, een phrase match, een modified match of een broad match.
- **AdsTemplate:** Dit is de advertentietemplate met de parameters erin. Deze wordt aangemaakt tijdens stap 5.
- **Parameter:** Een paramater kan in stap 5 worden ingevoerd in de advertentietemplate. Parameters worden aangemaakt in stap 2b als er extra variabelen zijn ingevuld.
- **ExtraVars:** Dit zijn de waardes uit de CSV of XML die tijdens het creëren van de CSV voor de Goolge Adwords Editor de paramaterwaarden vervangen. ExtraVars worden tijdens stap 2b ingevoerd als er extra variabelen zijn ingevuld.
- **Converter:** Om de ingevoerde data om te zetten naar een array is gekozen voor het gebruik van het Strategy Pattern. De developer had tijdens de interviews aangegeven dat hij in de toekomst eenvoudige nieuwe datatypes zoals TSV of XLS wil kunnen ondersteunen. Het Strategy Pattern maakt dit mogelijk. Wat de developer moet doen bij het ondersteunen van nieuwe datatypen, is een nieuwe Strategy klasse toevoegen, bijvoorbeeld XlsStrategy. Daarin moet de code worden geplaatst die nodig is om XLS om te zetten naar een Array.
  Ik heb het Factory Method Pattern als alternatief overwogen, omdat deze mogelijk ook geschikt was om eenvoudige nieuwe datatypen toe te voegen. Dit bleek niet zo te zijn; het enige doel van Factory is het creëren van objecten. Strategy Pattern bleek beter geschikt te zijn om het probleem van de verschillende datatypen op te lossen.
  Strategy Pattern voorziet in een mechaniek om verschillende algoritmen te selecteren, precies wat ik nodig had.
- **De User klassen van Rankinspector:** Deze zijn verantwoordelijk voor de informatie over de gebruiker die is ingelogd. Deze is gekoppeld aan het klasseproject, zodat duidelijk is welke user welk project gemaakt heeft.

6.3.2 ERD

Een ERD is het uiteindelijke model dat exact weergeeft hoe de database eruit komt te zien, inclusief de verschillende relaties. Dit diagram heb ik gemaakt om inzicht te krijgen in de verbanden tussen de gegevens.
De primaire sleutels en de vreemde sleutels heb ik vastgesteld op basis van de Adwords module uit het klassendiagram (zie figuur 19). De relaties hoeven in het ERD niet te worden beargumenteerd, omdat het ERD alleen een andere representatie is van het klassendiagram.
Wel is een verklaring nodig voor het feit dat er geen koppeling is tussen de projecttabel en de user tabel. De database die ik ga ontwikkelen wordt in figuur 20 weergegeven; de user tabel staat in een andere, reeds ontwikkelde database. Het documenteren van de hele bestaande database in deze ERD is overbodig, omdat ik alleen de vreemde sleutel van de user uit de andere database gebruik. De rest van de bestaande database wordt door mij niet gebruikt.

Referentiële integriteit met de user
Er wordt een koppeling gelegd tussen het project en de User uit de andere database door de primary key van de User-tabel op te slaan bij het project. Omdat de users in een tabel in een andere database staan, is referentiële integriteit een uitdaging. Want als een user uit de ene database verwijderd wordt, zou dat leiden tot corrupte data in de andere database. Daarom is bij de stagebegeleider nagegaan wat er gebeurt als een gebruiker verwijderd wordt uit het systeem van Rankinspector. Als een user verwijderd wordt uit Rankinspector, dan wijzigt alleen de status van een gebruiker. Dat wil zeggen dat hij de status ‘verwijderd’ krijgt, maar wel gehandhaafd blijft in de database.



Figuur 20 - ERD

6.3.3 RIM

Ik heb ervoor gekozen om alleen een RIM te maken; niet een RIM én een RRM. De reden hiervoor was dat ik bekwaam genoeg ben in SQL om meteen een RIM te maken.
Ik heb voor RIM gekozen omdat deze direct te importeren is in DBMS. De datatypes en de lengte van deze gegevens heb ik vastgesteld op basis van de bestaande database van Rankinspector.
De primaire sleutels en de vreemde sleutels heb ik vastgesteld op basis van het ERD. De volledige RIM-tabellen staan in bijlage 8, paragraaf 4.2.

6.4 Aanpak implementatie

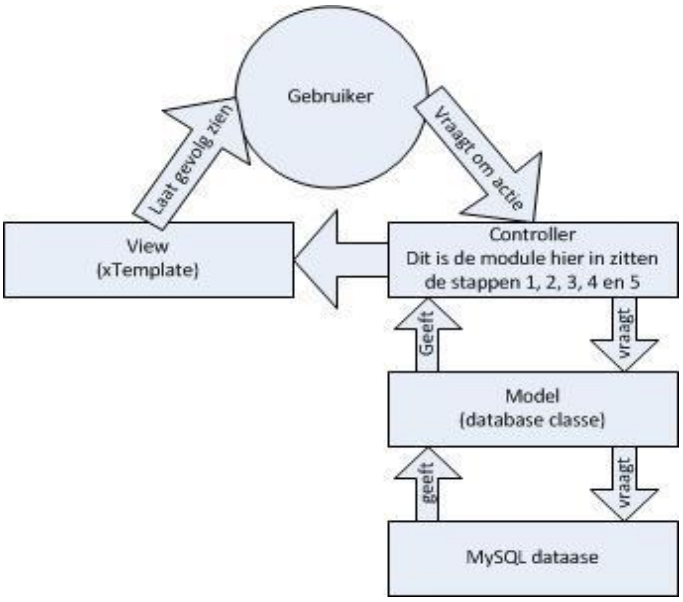
Tijdens de elaboratiefase komt de discipline implementatie aan bod. Dit is het laatste onderdeel van deze fase en betreft het ontwikkelen van een prototype. Een prototype is een product dat laat zien dat en hoe de uiteindelijke module gaat werken. Er is gekozen voor een prototype in plaats van voor bijvoorbeeld een Proof of Concept, omdat een prototype efficiënter is qua tijd. Een Proof of Concept dient slechts ter demonstratie van het product en wordt niet meer doorontwikkeld. Een prototype wordt wel in een later stadium doorontwikkeld.

De 5 stappen die ontwikkeld zijn, staan gedocumenteerd in paragraaf 6.2.3. Van deze 5 stappen is alleen het ‘insert gedeelte’ ontwikkeld; na de ontwikkeling van het prototype is het dus alleen mogelijk om data in te voeren. Dit project draait om het invoeren en exporteren van data naar de Google Adwords Editor. Ik heb ervoor gekozen om hier het invoeren van de data te ontwikkelen en niet het exporteren. Het exporteren is immers niet echt een functionaliteit die impact heeft op de werking van de module. Deze is daarom een stuk minder interessant om te demonstreren.
De functionaliteit bewerken, verwijderen en exporteren worden daarom tijdens de constructiefase ontwikkeld.
Het prototype werd aan het eind van deze fase gedemonstreerd aan de opdrachtgever en stagebegeleider. De feedback die deze presentatie opleverde, is gebruikt in de constructiefase. In bijlage 9 staat de volledige feedback

Tijdens de inceptiefase bleek dat het belangrijk is dat de applicatie als module in Rankinspector moet worden geplaatst. Daarom is ervoor gekozen om een module van Rankinspector te onderzoeken. De module die uiteindelijk is gekozen, is de User Beheer Module. Deze module bevat de insert querys die ik nodig heb voor het prototype.
Na het inspecteren van de User Beheer Module had ik een goed beeld hoe een module van Rankinspector omgaat met het invoeren van gegevens. Ik kon nu beginnen met het ontwikkelen van het prototype.

6.4.1 Hoe werkt Rankinspector met een module

Rankinspector maakt gebruik van een vertakking van het MCV Pattern. Om het MCV Pattern te kunnen toepassen maakt Rankinspector gebruik van standaard klassen. In figuur 21 is te zien hoe Rankinspector het MVC Pattern toepast.



Figuur 21 – Toepassing MVC Pattern in Rankinspector

De controller maakt gebruik van:

- **Restrictions** (verifyAccess): Rankinspector heeft een restrictiesysteem dat content beschikbaar stelt op basis van zijn rol.
- **XTemplate:** De xTemplate klassen is een Template Parser. Deze word aangestuurd en gevuld door de controller.
- **Database-klassen:** De database-klasse van maakt gebruik van PEAR [pear.php.net]. Deze stelt mij in staat om op een abstract manier te communiceren met de database.

6.4.2 Converter

Het Strategy Pattern is gebruikt om de converter te creëren, zodat er later zonder veel wijzigingen nieuwe datatypen ondersteunt kunnen worden. Dit was een wens van de programmeur van Traffic Builders.
Als in de controller een nieuwe klasse StrategyData wordt aangemaakt, moet de controller opgeven welk type data het is, bijvoorbeeld XML. Daarna wordt er in de klasse StrategyData gekeken of de klasse XML Strategy bestaat. Als het niet lukt om de klasse aan te maken – wat inhoudt dat hij niet bestaat – wordt er een default klasse teruggestuurd. De user weet dan dat dit Datatype niet ondersteund wordt.
Door StrategyData dynamisch te creëren hoeft de programmeur alleen een nieuwe StrategyClass toe te voegen als er een nieuw datatype ondersteund moet worden. Op deze manier hoeft er alleen een nieuwe klasse te worden toegevoegd die StrategyInterface implementeert. Vervolgens kan hij direct worden gebruikt zonder verdere aanpassingen in de code.

PHP-afhankelijke toevoeging

In PHP is het mogelijk om klassen aan te maken op basis van strings. Dat heeft als voordeel dat je de klasse kan aanmaken tijdens het uitvoeren van de creatie die op basis van string wordt meegegeven. Dit ziet er in PHP als volgt uit:

```
public static function create($strategy_ind_id) {
  $classname = $strategy_ind_id . 'Strategy';
  if (class_exists($classname)){
    return new $classname;
  }else{
    //hier zou je een error kunnen geven of als je het volgens regels van de factory method wil doen kun je net als hier onder een ‘default’ Strategy aanmaken
    return new DefaultStrategy;
  }
}
```

Deze benadering heeft het voordeel dat niet van tevoren uitgezocht hoeft te worden welke klassen moeten worden aangemaakt. In plaats daarvan kunnen tijdens het creëren de gewenste klassen worden opgevraagd. Mochten die niet bestaan, dan wordt er een error of een DefaultStrategy teruggestuurd.

Ik heb voor deze wijze van implementeren gekozen omdat:

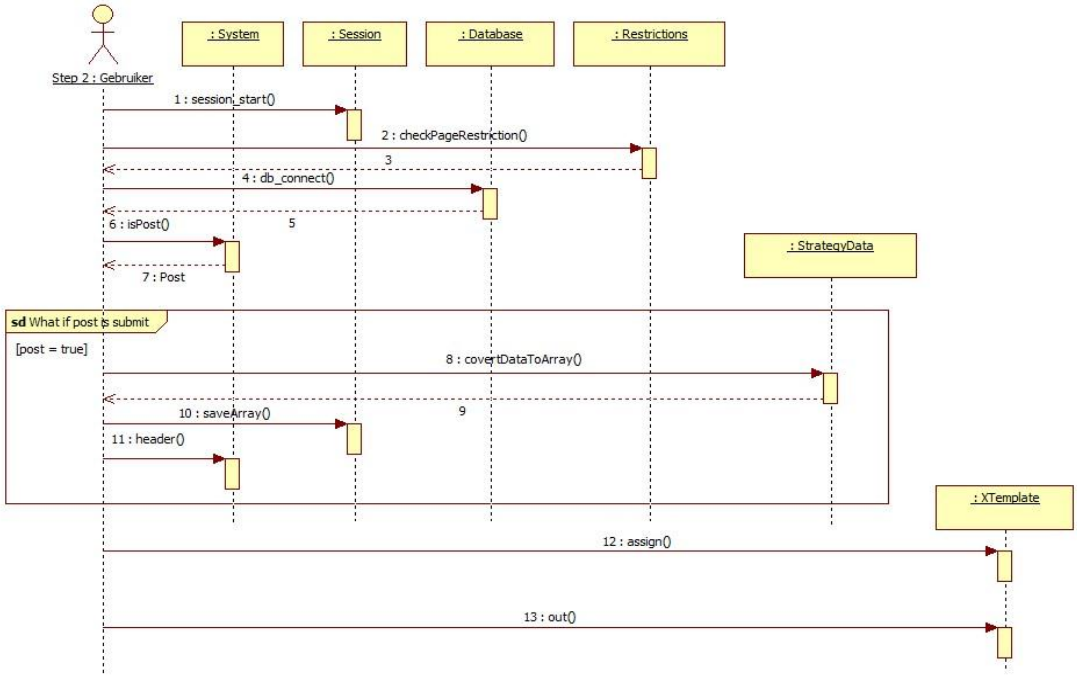
- Op de officiële pagina van PHP [php.net patterns] staat dit als voorbeeld aangegeven. Dit maakt het aannemelijk dat het een goede oplossing is binnen PHP.
- Het probleem dat er niet altijd klassen worden teruggestuurd – wat overigens wel de verantwoordelijkheid is van de Strategy-klasse – opgelost kan worden door in de else een default klasse te plaatsen (zoals in het voorbeeld staat).
- Het zorgt voor minder onderhoud.

6.4.3 De ontwikkeling van de vijf stappen

Elke stap is een controller en heeft dezelfde structuur. Deze structuur is grafisch weergeven in figuur 22 en bestaat uit de volgende onderdelen. Eerst wordt beoordeeld of de gebruiker de rechten heeft om deze actie uit te voeren. Vervolgens worden de gewenste data uit de database gehaald door de database-klasse. Daarna worden de data bewerkt en worden de bewerkingen, indien gewenst, weer in de database opgeslagen. Daarna wordt het resultaat getoond door de view (xTemplate).

Figuur 22 - Controller

Om de werking van de stappen te verduidelijken heb ik stap 2a uitgewerkt. Van deze stap heb ik een Sequence Diagram gemaakt (zie figuur 23). Voor 2a is gekozen omdat deze gebruikmaakt van de Strategy-klasse.



Figuur 23 - Sequence Diagram stap 2a

Bij alle stappen zijn de eerste 4 acties hetzelfde: er wordt altijd eerst een session aangemaakt, er wordt gecontroleerd of deze gebruiker deze pagina mag bezoeken en er wordt een connectie gemaakt met de database. Zie ook figuur 22 om te zien hoe de flow is van een stap.
Ook wordt bij de meeste stappen gekeken of het gaat om een 'post request'. Dat wil zeggen dat er gekeken wordt of de gebruiker deze pagina opvraagt door middel van een formulieraanvraag. Als dit het geval is, moeten de data worden opgeslagen die tijdens deze stap worden aangemaakt of gewijzigd. Op deze manier wordt het opslaan en het opvragen van data op één plek afgehandeld.
In figuur 23 is te zien hoe bij een post request de klasse StrategyData wordt aangemaakt en waar de aangemaakte data naar toe worden gestuurd. Daarna wordt de array, die in de StrategyData is aangemaakt, teruggestuurd naar de session. Zodoende kan deze array vervolgens in stap 2b gebruikt worden om de waardes te koppelen, waarna ze opgeslagen kunnen worden in de database.

6.4.4 Keuzes

Zoals vooraf in de klassendiagrammen is bedacht, bestaat de te ontwikkelen code uit 2 delen: de Google Adwords module zelf en de converter. Als eerste zijn de converter en de klassen uit het klassen diagram gemaakt, omdat de controllers stap 1, 2a, 2b, 3, 4 en 5 van deze klassen en de converter gebruiken.

6.4.5 Feedback

Na de ontwikkeling van het prototype is aan de stagebegeleider om feedback gevraagd. Hij heeft de code en de werking van het prototype bekeken en vond dat er goed werk was geleverd. Daarna is het prototype gereviewd door de opdrachtgever. De volledige feedback is te lezen in bijlage 9.
De belangrijkste feedback behelsde enkele noodzakelijke aanpassingen aan de lay-out, maar de module functioneerde verder zoals gewenst. De nieuwe wensen ten aanzien van de lay-out zijn in samenspraak met de opdrachtgever en de stagebegeleider bestempeld als 'could haves'. Zij zijn dus niet verplicht voor het succesvol afsluiten van het project, maar het zou wel prettig zijn als de aanpassingen erin zitten.

6.5 Testen

Omdat de tijd voor het testtraject beperkt was, heb ik ervoor gekozen om de module informeel te testen op syntax en semantiek. Ik heb de invoer getest en gekeken wat er met de data gebeurde.

Deze testen hoefde ik slechts beperkt uit te voeren, omdat ik uit ervaring al weet waar het vaak fout gaat. In de documentatie staan de informele testgevallen. Zie verder bijlage 12.
Tijdens het testen van het prototype is er een belangrijke fout ontdekt: de module kon in dat stadium niet omgaan met het aanhalingsteken (‘). De test die dit probleem achterhaalde was de belangrijkste en heb ik daarom hieronder weergegeven.
Omdat er tijdens de constructiefase op dezelfde wijze is getest, beschrijf ik alleen hier de manier van testen.

Test aanhalingsteken
Tabel 7

Nr	**1**			
Testcase	Voer nieuw project in (stap 1)			
Doel	Het testen op mogelijke invoer door de gebruiker			
Input		Test 1	Test 2	Test 3
	Projectnaam	‘6kdkd	€Dûj12345678	Netjes
	Dagbudget	Ddd88	8,8 (er staat een **komma**)	8.8 (er staat een **punt**)
	Campagnenaam	.,.txt.dcsv.css.	Dûj12345678	Netjes
Verwachte uitkomst		Test 1	Test 2	Test 3
	Projectnaam	‘6kdkd	€Dûj12345678	Netjes
	Dagbudget	0	8.8 (er staat een **punt**)	8.8 (er staat een **punt**)
	Campagnenaam	.,.txt.dcsv.css.	Dûj12345678	Netjes
Uitkomst		Test 1	Test 2	Test 3
	Projectnaam	Foutmelding	€Dûj12345678	Netjes
	Dagbudget	0	8.8 (er staat een **punt**)	8.8 (er staat een **punt**)
	Campagnenaam	.,.txt.dcsv.css.	Dûj12345678	Netjes
	De prototype van de module blijkt gevoelig voor mysql injecties. Hier moet tijdens de constructiefase een oplossing voor gezocht worden.			
Commentaar	Bij het invullen van de projectnaam, het dagbudget en de campagne wordt getest op verschillende invoer en of deze juist wordt afgehandeld. Er is speciaal getest op het aanhalingsteken (‘) omdat er getest moest worden of de module gevoelig is voor mysql injecties			

6.6 Conclusie elaboratiefase

In deze paragraaf staan de belangrijkste bevindingen van de elaboratiefase.

Onderzoek productaanlevering
Bij de aanvang van het afstudeerproject leek het vooral de bedoeling dat er producten geïmporteerd moesten worden en omgezet in advertenties. Tijdens de Inceptiefase werd duidelijk dat de nieuwe module geen producten moest gaan omzetten naar advertenties, maar dat de module keywords met bijwoorden tot generieke keywords moest combineren (zie figuur 9, paragraaf 5.3). Daarmee werd een onderzoek naar hoe de klanten de producten moesten gaan aanleveren overbodig.

Onderzoek verwerking data
Dit is tijdens het ontwerpen van het functioneel en technisch ontwerp gedaan.

Usecase views
Tijdens het ontwikkelen van de use cases werd duidelijk dat de gebruikers graag iets tastbaars wilden om feedback op te geven. Daarom zijn de views gemaakt op basis van de use cases. Dit heeft voor extra duidelijkheid gezorgd bij de gebruikers en de opdrachtgever, wat de communicatie tussen de stakeholders en mij ten goede kwam.

Onderhoud in de toekomst

Tijdens het ontwikkelen van mijn prototype heb ik rekening gehouden met de werkwijze in de rest van Rankinspector. Dit heb ik gedaan door in mijn module dezelfde mappenstructuur en dezelfde naamgevingen te gebruiken als in Rankinspector. Ook de opbouw van de code en het gebruik van reeds aanwezige klassen – zoals de database en view klasse – zijn op dezelfde wijze geïmplementeerd als in andere modules. Doordat ik mij aan de huidige werkwijze van Traffic Builders heb gehouden, is het onderhoud van de module eenvoudig.

7 Constructiefase iteratie 1

In dit hoofdstuk staat het verloop van de constructiefase beschreven. Hier zijn een aantal delen van de ontwikkelde code beschreven, zoals het CRUD-gedeelte en de export van de CSV. Tot slot worden er voorbeelden gegeven van de uiteindelijke lay-out van de ontwikkelde module.

7.1 Query's

Insert
Tijdens de elaboratiefase heb ik van de 5 stappen inserts gemaakt. Maar deze bleken tijdens het testen (zie paragraaf 6.5) van de in- en uitvoer van de module niet op de meest veilige wijze gemaakt te zijn. Er was te veel gebruik gemaakt van de functie query om de data op te slaan, waardoor het systeem gevoelig werd voor MySQL-injecties.
Om dit te verhelpen zijn tijdens de constructiefase alle insert query's omgezet naar PDO (PHP Data Objects). Op deze manier worden de data op een veilige manier in de database geplaatst. Om MySQL-injecties te voorkomen, is het ook mogelijk om een andere functie van PHP te gebruiken. Hiervoor heb ik niet gekozen omdat verschillende instanties – waaronder PHP.net zelf – dit afraden [php.net/mysql-real-escape-string]. De voornaamste reden om deze functie niet te gebruiken is dat hij de input aanpast voordat deze in de database opgeslagen wordt. Dit is niet wenselijk vanwege het risico dat data dubbel gequote opgeslagen word. PHP.net raadt aan om gebruik te maken van PDO.

Update
De twee stappen die geüpdate kunnen worden zijn stap 4 en 5 (Figuur 14 in paragraaf 6.2.3). Aan het eind van stap 3 worden alle combinaties tussen keywords (stap 2) en bijwoorden (stap 3) gemaakt. Deze combinaties zijn tijdens stap 4 te wijzigen. Bij stap 5 is het wijzigen van advertentietemplates ontwikkeld.

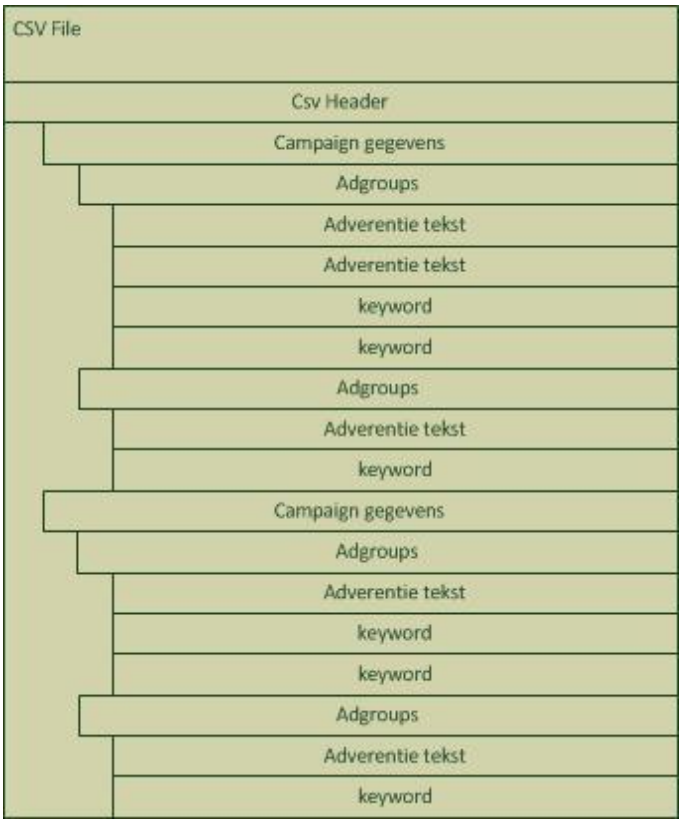
Delete
Het verwijderen van keywords moest gebeuren met een extra check. Als de gebruiker van de module een main keyword verwijdert (zie figuur 24, nummer 1), moeten de onderliggende keyword combinaties ook verwijderd worden (zie figuur 24, nummer 2). Wordt een keyword combinatie verwijderd, dan moet alleen de combinatie verwijderd worden.

Keyword	MaxCPC	Destination URL	Eigen Campagne	Exact	Pharse	Modified	Broad	
▼ leeuwarden 1	0.55	http://www.droomboeket.r	☐	☑	☑	☑	☐	[x]
leeuwarden + bloemen bezorgen 2	0.55	http://www.droomboeket.r	☐	☑	☑	☑	☐	[x]
▶ deventer	0.55	http://www.droomboeket.r	☐	☑	☑	☑	☐	[x]
▶ delft	0.55	http://www.droomboeket.r	☐	☑	☑	☑	☐	[x]

Figuur 24 - Keywords en bijwoorden

7.2 CSV-export

Voor het maken van de CSV-export is gekeken naar het voorbeeld dat Google geeft [csv voorbeeld]. Van dit voorbeeld heb ik een grafische weergave gemaakt om te verduidelijken hoe een CSV eruit moet zien voor de Google Adwords Editor (zie figuur 25).
Door bepaalde velden in te vullen, wordt aangegeven of het om een nieuwe Campaign, Adgroup, advertentietekst of keyword gaat (zie tabel 7).



Figuur 25 - CSV opbouw

Selectie via velden

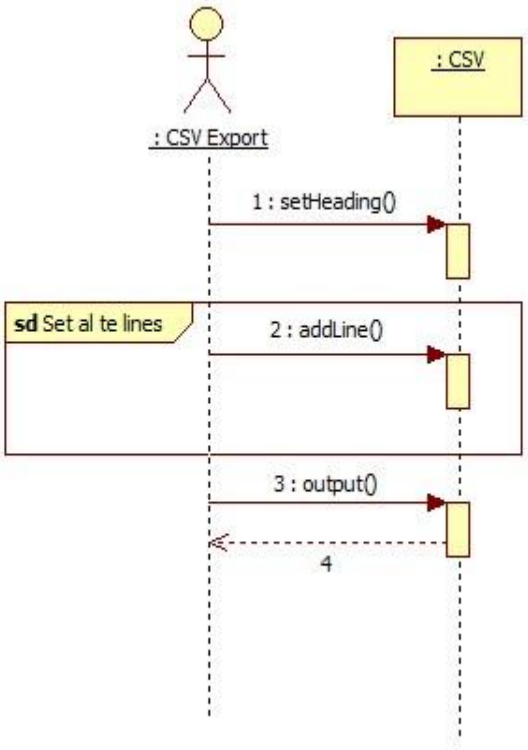
Tabel 5

Welke row	Rijen die ingevuld moeten worden
CSV header	Alle rijen die nodig zijn voor een werkende CSV voor in de Google Adwords editor. De CSV header ziet er als volgt uit: Campaign, Campaign Daily Budget, Languages, Location ID, Location, Ad Schedule, Networks, Ad Group, Max CPC, Flexible Reach, Keyword, Keyword Type, Headline, Description Line 1, Description Line 2, Display URL, Destination URL
Campaign	Campaign en Ad Group zijn verplicht. Flexible Reach en Max CPC zijn optioneel.
Ad group	Campaign, Ad Group, Headline, Description Line 1, Description Line 2, Display URL en Destination URL.
Advertentie	Wanneer aangegeven moet worden dat het om een nieuwe adverentie gaat, moet worden ingevuld; Campaign, Ad Group, Headline, Description Line 1, Description Line 2, Display URL en Destination URL.
Keyword	Campaign, Ad Group, Max CPC, Keyword en Keyword Type.

CSV-klasse

Tijdens het ontwikkelen van de mogelijkheid om een CSV te exporteren, kwam ik erachter dat CSV's aan bepaalde syntaxis moeten voldoen. Als bijvoorbeeld de inhoud van een veld een komma bevat, levert dit normaal gesproken fouten op omdat een komma het einde van een veld aangeeft. Een komma is namelijk de delimiter van een CSV. CSV heeft daar een oplossing voor bedacht: door " " om een veld heen te zetten. Hiermee kan worden aangegeven dat het één veld is. Maar daardoor treed

een nieuw probleem op wanneer de leestekens “ ” in het veld voorkomen. Dit probleem kan worden voorkomen door voor elke “ in het veld een / te zetten.
Elk veld moet dus worden gecontroleerd op komma’s en aanhalingstekens. Ik heb dit op één centrale plek ondergebracht in de CSV-klasse. Bekijk figuur 26 om te zien hoe de CSV-klasse gebruikt kan worden.



Figuur 26 - Sequence Diagram CSV creëren

7.3 Meerdere versies van de templates

Tijdens de inceptiefase was al gebleken dat de gebruikers meerdere versies wilden kunnen opgeven van de title, description line 1, description line 2, display url. Dit heb ik tijdens de elaboratiefase opgelost door tussen de verschillende versies een delimiter te zetten. Zo kan de gebruiker zelf bepalen hoeveel versies hij wil aanmaken. Maar de opdrachtgever gaf tijdens het feedbackmoment aan dat hij niet wilde dat de gebruiker zelf kon bepalen hoeveel versies hij kan aanmaken. De opdrachtgever wilde 3 inputvelden, zodat de gebruiker maximaal 3 versies kan opgeven. Figuur 27 toont hoe dat eruit ziet.

De functionaliteit om met delimiter te werken was reeds ontwikkeld. Daarom is in overleg met de stagebegeleider besloten om die functionaliteit te behouden. Zo blijft de flexibiliteit bestaan. Mocht de opdrachtgever in de toekomst nog extra velden willen toevoegen of verwijderen dan is dit eenvoudig te bewerkstelligen. Dus de verschillende versies van de titel staan in één kolom, de verschillende versies van de description line 1 staan in één kolom, de verschillende versies van de description line 2 staan in één kolom en de verschillende versies van de url staan in één kolom.

Figuur 27 - Advertentietemplates invoeren

7.4 AdsTemplate klasse

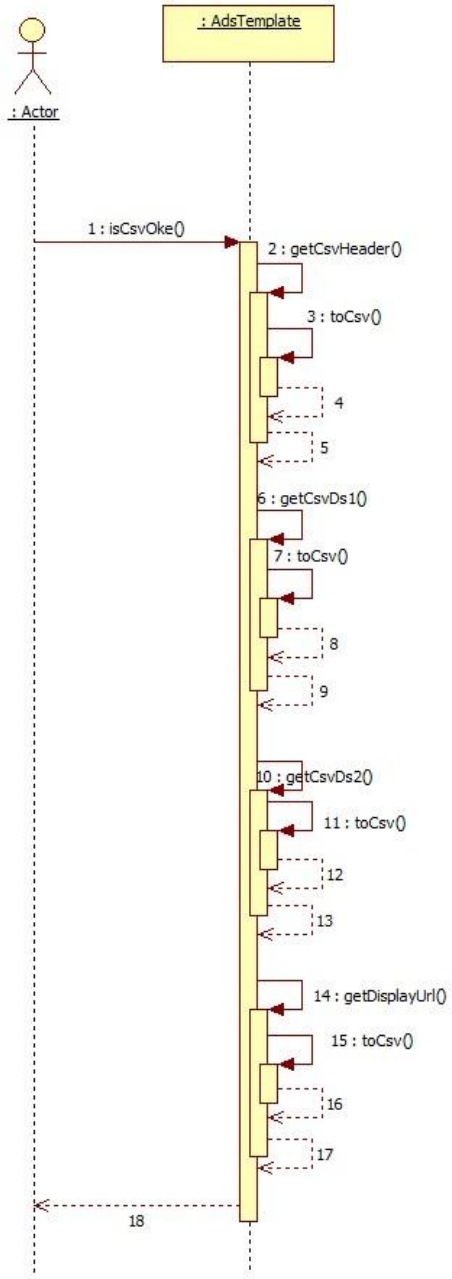
Alle klassen die in het klassendiagram in paragraaf 6.3.1 staan, zijn ontwikkeld. De werking van AdsTemplate wordt besproken omdat deze template data bewerkt en controleert. Ik heb de klasse AdsTemplate uit het klassendiagram uitgebreid (zie figuur 28).

Figuur 28 - AdsTemplate klassendiagram

In dit figuur is te zien dat alle standaard getters en setters aanwezig zijn. Er zijn ook extra functionaliteiten aanwezig, zoals de getCsvHeader, getCsvDs1, getCsvDs2 en getCsvDisplayUrl. Deze functionaliteiten roepen allemaal de functionaliteit toCsv aan. Deze functie stuurt de passende ds1 terug. Want in toCSV wordt de Ds1 uit elkaar gehaald op de delimeter en worden de versies bekeken; de eerste die past wordt teruggestuurd. Past er niet één, dan stuurt hij een lege string terug.

Bij getCsvDisplayUrl worden spaties vervangen door streepjes. Zou dit niet worden gedaan, dan stond er in de uiteindelijke advertentieteksten '%20' in plaats van een spatie. Dit komt omdat in de display url spaties niet toegestaan zijn. '%20' vindt de opdrachtgever onwenselijk, hij geeft de voorkeur aan een streepje.

De functionaliteit isCsvOke checkt alle vier de getCsv-functies. Als er één een lege string door toCsv teruggestuurd wordt, is de CSV niet juist. Wanneer de CSV onjuist is, stuurt de functie terug dat de CSV onjuist is. Wanneer alles in orde is, stuurt hij terug dat het goed is en dus gebruikt kan worden in de CSV. Zie figuur 29 voor het sequentiediagram van isCsvOke.



Figuur 29 – Sequentiediagram isCsvOke

7.5 Feedback

Nadat ik de eerste volledige versie van het systeem had voltooid, heb ik een presentatie gegeven over de werking van de module. De opdrachtgever reageerde positief op de aanpassingen aan het prototype. De taalverbeteringen waren goed en ook de manier waarop de advertenties ingevoerd moeten worden, kreeg groen licht.

De inputvelden waren voorzien van tekstuele ondersteuning en alle verwijder- en updatefunctionaliteit werkte naar wens. De enige opmerking was dat de tabellen nog niet met dhtmlx werkten (dit is een manier van tabellen weergeven in de GUI) en dat de door mij ontwikkelde module nog dezelfde style moest krijgen als de overige modules van Rankinspector. Ik heb voorgesteld deze aanpassingen door te voeren in de volgende iteratie. Dit vond de opdrachtgever een goed idee.
De stagebegeleider heeft gekeken naar de code, de locatie en de mappenstructuur. Hij vond dat sommige bestanden in een andere map hoorden. Het programmeerwerk vond hij goed. Tot slot vond hij dat er gebruik gemaakt moet worden van een taalbestand, zodat de module later ook aangeboden kan worden in het Engels.

7.5 Conclusie constructiefase

In dit hoofdstuk staan de belangrijkste bevindingen uit de constructiefase, eerste iteratie.

Twee iteraties
Tijdens mijn afstudeerproject heeft de focus gelegen op de ontwikkeling van de techniek en niet op de GUI. Daarom heb ik mij tot en met deze fase weinig bezig gehouden met de grafische weergave. Halverwege deze fase werd echter tijdens een tussentijdse feedback duidelijk dat dat wel belangrijk is. Voordat de module in gebruik genomen kon worden, moest hij dezelfde grafische stijl hebben als de andere modules in Rankinspector. Dat hield onder andere in dat de GUI-tabellen gebruik moesten maken van dhtmlx. Ik vond het verstandig om dat niet in deze iteratie te realiseren, omdat ik niet bekend was met dhtmlx en ik me tijdens deze iteratie vooral wilde concentreren op de functionaliteiten van de applicatie. Daarom heb ik ervoor gekozen de constructiefase in tweeën te delen: de eerste voor de werking en de tweede voor de GUI.

8 Constructiefase iteratie 2

In dit hoofdstuk staat beschreven wat er gebeurd is tijdens de tweede iteratie van de constructiefase. De lay-out is op een aantal punten aangepast, vooral de tabellen. De taalbestanden zijn gecreëerd en de module is in de live omgeving geplaatst.

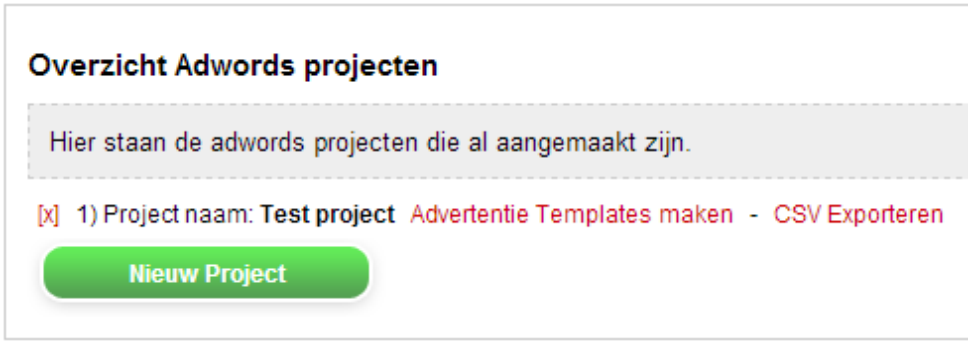
8.1 Module omzetten in Rankinspectorstijl

De focus van het project ligt op het maken van functionaliteiten. Daarom is tijdens de ontwikkeling alleen rekening gehouden met de structuur van pagina's en niet met de vormgeving. De view verwerkt geen berekeningen en voert geen controles uit van de gegevens die getoond worden. Wel zijn de views door het MVC-pattern gescheiden van de functionaliteit, zodat het aanpassen van de lay-out in een later stadium overzichtelijk kan verlopen.
Tijdens deze iteratie is het de bedoeling om de module in dezelfde stijl te krijgen als andere modules in Rankinspector.

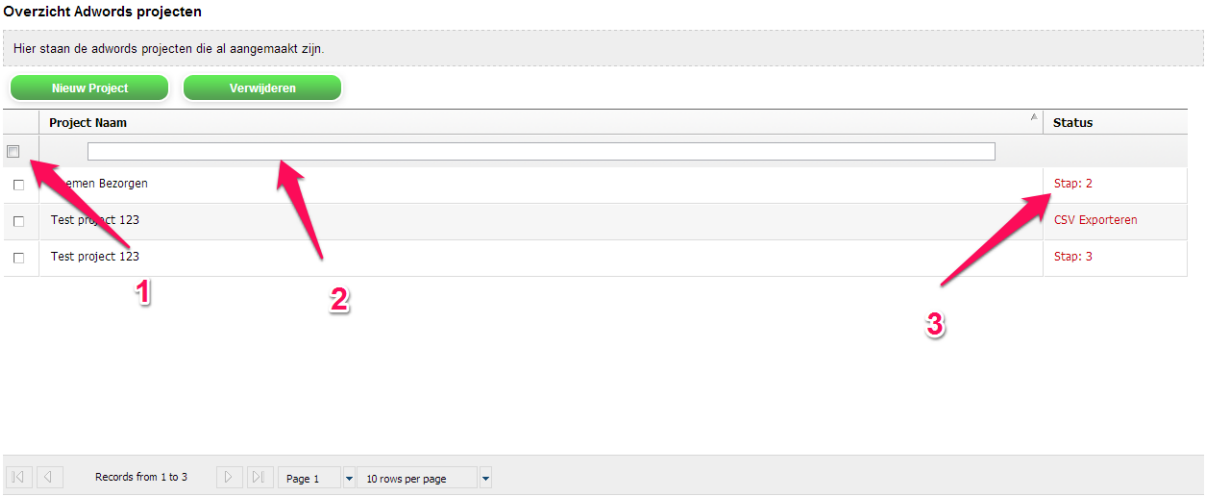
Werking dynamische tabel
Ik heb de documentatie van dhtmlx gebruikt om te zien hoe de GUI-tabellen moeten worden opgebouwd. In de documentatie [dhtmlx documentatie] zijn goede voorbeelden en tutorials te vinden over de opbouw en bewerking van een tabel.
Als je een tabel dynamisch wilt laden, dan wordt er gebruik gemaakt van een asynchrone aanvraag (Ajax Call). Deze haalt data uit een PHP-bestand. In dit PHP-bestand worden de data uit de database opgevraagd en opgemaakt als CSV. Vervolgens wordt de CSV teruggestuurd naar de Ajax Call. Daarna moet in Javascriptcode worden aangeven welke tabel-headers en rijen data verwacht moet worden. Vervolgens moet aangeven worden welke data uit de rij de unieke key bevatten, welke data getoond gaan worden en op welke manier.

Overzicht Adwordsprojecten
De lijst met het overzicht van de projecten word dynamisch in de tabel geladen. De tabel kan met behulp van Ajax worden aangepast zonder dat de pagina ververst hoeft te worden. Er kunnen nu meerdere projecten tegelijk verwijderd worden. Het ontwerp was eerst zoals in figuur 28 te zien is. Nu is het ontwerp geworden zoals in figuur 29 te zien is.

In figuur 29 is bij nummer 1 te zien hoe in één keer alle projecten geselecteerd kunnen worden. Vervolgens kan op 'verwijderen' geklikt worden.
Bij nummer 2 is een zoekbalk te zien. Hiermee kunnen gebruikers projecten zoeken door de naam in te vullen. Dan worden dynamisch de juiste projecten geladen.
Met een klik bij nummer 3 wordt het project afgemaakt of wordt een CSV voor Google Adwords geëxporteerd.



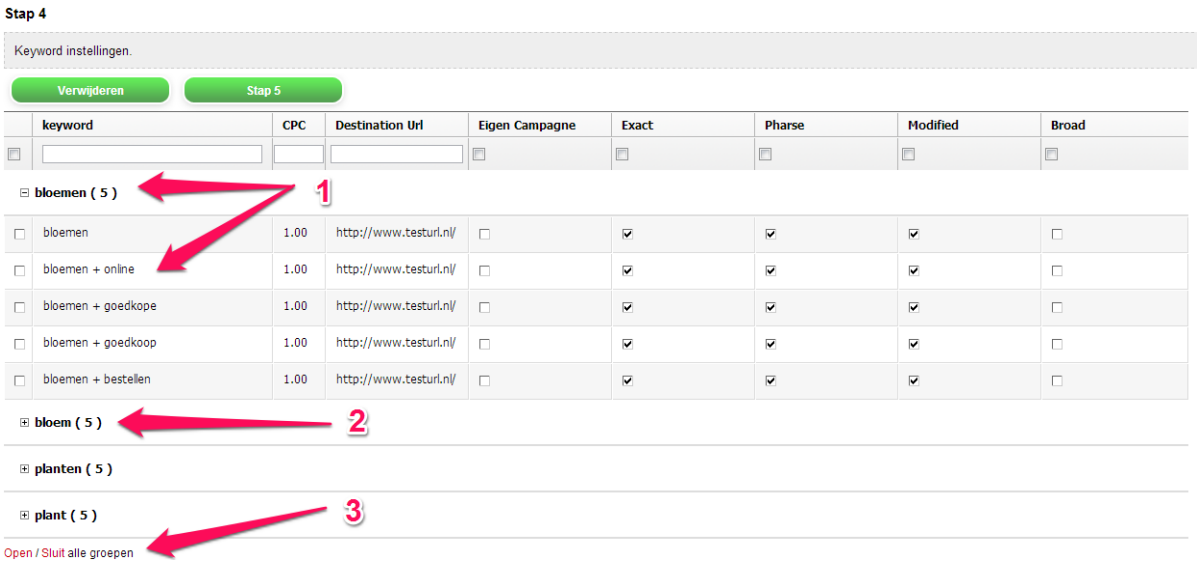
Figuur 28 - Het oude ontwerp van het projectoverzicht



Figuur 29 - Het nieuwe ontwerp van het projectoverzicht.

Keywords wijzigen

In figuur 24 in paragraaf 7.1 is te zien hoe de lay-out van het wijzigen van de keywords er na de eerste iteratie uit zag. Tijdens de tweede iteratie is de lay-out gewijzigd naar de situatie in figuur 30. In de tabel wordt alles gegroepeerd op de main keywords.



Figuur 30 - Het nieuwe ontwerp voor het wijzigen van de keywords

In figuur 30 zijn nummers te zien die nieuwe functionaliteiten aanduiden:

1. Toont een uitgeklapte groep
2. Toont een ingeklapte groep
3. Toont de mogelijkheid om alle groepen te openen of te sluiten

8.2 Overige aanpassingen

Tijdens de tweede iteratie zijn, naast bovenstaande wijzigingen, nog een aantal veranderingen doorgevoerd:

- Er zijn taalbestanden ingevoerd. Dit houdt in dat de tekst in de templates is vervangen door variabelen. Deze variabelen worden gevuld in een taalbestand. Bijvoorbeeld: als in de template wordt gezet {welkom}, dan wordt in het Nederlandse taalbestand gezet {welkom} = ‘Goedendag’ en in het Engelse taalbestand {welkom} = ‘Good Day’.
- De bestanden zijn in de juiste mappen geplaatst en in de code zijn de locaties van de bestanden aangepast.

- De update- en delete-functionaliteit van het overzicht en het weergeven van de keywords zijn aangepast zodat deze om kunnen gaan met Ajax-aanvragen.

8.3 Implementatie in de live omgeving

Bij het ontwikkelen van het prototype en in de constructiefase heb ik gebruik gemaakt van een kopie van de laatste versie van Rankinspector. Toen de productie klaar was heb ik alle nieuw aangemaakte mappen en bestanden in een map gezet en de map in een zip geplaatst. De zip is op de server uitgepakt. Op deze manier zijn de bestanden in de juiste mappen in de live omgeving gekomen. Vervolgens heb ik de RIM gebruikt om de nieuwe database aan te maken op de live omgeving.

9 Transitiefase

In dit hoofdstuk staat beschreven wat er gebeurd is tijdens de transitiefase. Deze fase is bedoeld:

- om te achterhalen of de module aan vooraf opgestelde requirements voldoet;
- voor de inproductiename (deployment);
- voor de nazorg;
- en voor de overdracht van de verantwoordelijkheden.

De transitiefase wordt afgesloten met een inventarisatie van de opgedane ervaringen (lessons learned).Deze heb ik verwerkt in een evaluatie die te lezen is in hoofdstuk 11. In het kader van dit hoofdstuk heb ik mij vooral bezig gehouden met de inproductiename van de module en met de vraag of de module voldeed aan de vooraf opgestelde requirements.

9.1 Voorbereiding en uitwerking

Tijdens de voorbereiding is eerst een presentatie gemaakt waarin de werking van de module werd uitgelegd. Alle gebruikers-stakeholders waren uitgenodigd voor deze presentatie om hen in één keer allemaal te kunnen inlichten.
Vervolgens is aan de hand van de requirements uit de elaboratiefase een enquête opgesteld om te achterhalen of de gebruikers vinden dat de module werkt zoals hij zou moeten werken. De requirements uit de elaboratiefase waren gebaseerd op de wensen en eisen van de gebruikers. Nu was het moment daar om te achterhalen of aan deze wensen en eisen is voldaan.

Tijdens de presentatie waren de reacties van de gebruikers zeer positief. Zij vonden dat er geluisterd was naar hun wensen en eisen en waren tevreden over het resultaat.
De enquête hield ik na afloop van de presentatie. Alle 8 aanwezige gebruikers kwamen daarin tot dezelfde conclusie: alle wensen zijn geïmplementeerd, behalve 11 en 25.
Hieronder staat de tabel met de vragen en de resultaten.

Aan requirement 11 is niet voldaan omdat andere requirements belangrijker gevonden werden. Ik heb dit in samenspraak met mijn stagebegeleider bepaald.
Over requirement 25 is meerdere keren met de opdrachtgever en de gebruikers gesproken. Zij kwamen niet tot een eensluidend oordeel over de manier van nummeren. Het project liet onvoldoende tijd om voor dit dilemma – dat was geprioriteerd als should have – een oplossing te vinden.

In onderstaande tabel met requirements zijn de MoSCoW-prioriteiten uit de elaboratiefase toegevoegd. Zodoende is in één oogopslag te zien of er voldaan is aan alle must have's. De requirementnummers in de eerste kolom zorgen voor de herleidbaarheid.

Requirements

Tabel 8

Req nr	Vragen	Voldaan?	MoSCoW prioriteit
1	De 4 verschillende matchtypes kunnen worden gekozen.	**Ja** / Nee	Must have
2	Nieuwe projecten kunnen stap voor stap worden aangemaakt.	**Ja** / Nee	Must have
3	Tussentijds worden de stappen opgeslagen.	**Ja** / Nee	Must have
4	XML-bestanden kunnen worden geüpload.	**Ja** / Nee	Must have
5	XML-elementen kunnen eenvoudig worden gekoppeld aan variabelen.	**Ja** / Nee	Must have
6	Parameters kunnen worden geplaatst in advertentietemplates.	**Ja** / Nee	Must have
7	De namen van parameters kunnen zelf gekozen worden.	**Ja** / Nee	Must have
8	Meerdere advertentietemplates kunnen worden aangemaakt.	**Ja** / Nee	Must have
9	Keywords kunnen worden uitgesloten in een XML.	**Ja** / Nee	Could have
10	Als gebruiker wil ik voor de campagnes kunnen bepalen wat het budget is en wat de maximale kosten per klik mogen zijn.	**Ja** / Nee	Could have
11	Keywords kunnen worden uitgesloten door aan te geven wat de minimale en maximale grootte mag zijn van een keyword.	Ja / **Nee**	Should have
12	Er kunnen meerdere mogelijkheden worden opgegeven voor de titel, de beschrijvingen en de display url van de advertenties. De mogelijkheid die past binnen de gestelde limieten van Google moet worden gebruikt.	**Ja** / Nee	Must have
13	Per (generiek) keyword kan worden aangeven welke matchtypes daarbij horen.	**Ja** / Nee	Must have

14	Generieke keywords kunnen worden ingevoerd.	**Ja** / Nee	Must have
15	Bijwoorden kunnen worden toegevoegd.	**Ja** / Nee	Must have
16	De standaard waarden kunnen per project worden aangegeven.	**Ja** / Nee	Could have
17	Per keyword kan worden aangegeven of dit een aparte campagne moet worden.	**Ja** / Nee	Should have
18	Per keyword kan worden aangegeven wat er maximaal per klik betaald mag worden.	**Ja** / Nee	Won't have
19	Per keyword kan worden aangegeven wat de destination url is.	**Ja** / Nee	Should have
21	Er is een helpfunctie ter ondersteuning.	**Ja** / Nee	Must have
25	De advertenties kunnen automatisch genummerd worden.	Ja / **Nee**	Should have

10 Verantwoording beroepstaken

In het afstudeerplan zijn 5 beroepstaken gedefinieerd die ik als doelstelling had opgenomen in mijn afstudeerproject. Het gaat om de volgende beroepstaken:

- 1.1 Het selecteren van methoden, technieken en tools;
- 1.4 uitvoeren van analyse door definitie van requirements (achteraf ingevoerd);
- 2.1 opstellen van een gegevensmodel voor een database;
- 3.1 ontwerpen van een softwarearchitectuur;
- 3.3 het bouwen van een applicatie.

In dit hoofdstuk beschrijf ik op welke manier ik de beroepstaken heb toegepast. Elke beroepstaak wordt in cursieve tekst toegelicht.

1.1 Selecteren van methoden, technieken en tools [niveau 4]
"Selecteren en adviseren over een projectmanagementmethode. Selecteren en adviseren over methoden en technieken om een systeem te ontwikkelen, te beheren en te testen. Selecteren en adviseren over ontwikkeltools die een onderdeel (gaan) vormen van de ontwikkel-, beheer-, test- en productieomgeving. Opzetten van de ontwikkel-, beheer-, test- en productieomgeving. Kunnen werken met de geselecteerde methoden binnen de gerealiseerde omgeving."

Traffic Builders maakt geen gebruik van een gestandaardiseerde projectmanagementmethode. Voor dit project was het noodzakelijk dat er wel een gestandaardiseerde projectmanagementmethode werd gebruikt die aansloot bij het bedrijf. De keuze viel, met volledige instemming van de opdrachtgever, op RUP. Deze methode kreeg de voorkeur omdat hij het best aansloot bij de fasen van inititialisatie, ontwikkeling en oplevering die Traffic Builders doorgaans met projecten doorloopt. De fase die bij Traffic Builders ontbreekt is de inceptiefase. De andere drie – elaboratiefase, constructiefase en transitiefase – komen overeen met de fasering bij Traffic Builders.

Traffic Builders maakt gebruik van een ontwikkelomgeving, een testomgeving en een live omgeving. Daar heb ik tijdens dit project gebruik van gemaakt.

Omdat ik een projectmanagementmethode heb gekozen die aansluit bij het project, het bedrijf en de wisselende requirements vind ik dat ik deze beroepstaak heb behaald.

1.4 Uitvoeren van analyse door definitie van requirements [niveau 3]
"Inventariseren, analyseren en beschrijven van de gewenste informatievoorziening, zodat wordt vastgelegd welke processen objecten creëren en gebruiken. Analyseren, opstellen en beschrijven van de requirements (en hun samenhang) van elke te automatiseren informatiefunctie en van het informatiesysteem als geheel. Toetsen van de requirements op volledigheid, duidelijkheid en consistentie."

Deze beroepstaak heb ik toegevoegd omdat hij mij een groot deel van de afstudeerperiode heeft bezig gehouden.
In dit afstudeerproject ging het om een module met verschillende requirements. Er was sprake van 3 stakeholders met tegengestelde wensen en eisen ten aanzien van de module. Tijden het project zijn er verschillende requirements bij gekomen. Twee voorbeelden van belangrijke veranderingen die de gedachtewisselingen over de requirements opleverden:

- In het begin van het project lag de focus vooral op de techniek en de functionaliteit. Tijdens verschillende feedbackmomenten werd duidelijk dat de layout ook belangrijk was.
- In het begin van het project was het de bedoeling dat het project tot een standalone applicatie zou leiden. Tijdens de inceptiefase werd echter duidelijk dat het een module moest worden van Rankinspector. Dit is een ingrijpende verandering geweest, onder meer omdat sommige requirements niet meer ontwikkeld hoefden te worden. In plaats daarvan moest worden onderzocht hoe Rankinspector werkt.

Het onderzoeken en achterhalen van requirements heb ik op een gestructureerde manier uitgevoerd. Ik heb vooronderzoek gedaan naar bestaande tools (zie paragraaf 5.2.1) om vast te stellen welke functionaliteiten zij aanbieden. Dit vooronderzoek was noodzakelijk omdat de nieuwe module moest lijken op de bestaande tools.
De lijst met functionaliteiten die het vooronderzoek opleverde, heb ik gestructureerd ingezet als basis voor de interviews (zie paragraaf 5.3). De requirements die uit dit vooronderzoek naar voren kwamen,

heb ik systematisch geprioriteerd volgens de MoSCoW-methode. Ik heb dit in nauw overleg met de betrokken stakeholders gedaan. Zo konden zij aangeven wat ze belangrijk en minder belangrijk vonden(zie paragraaf 5.5). Dat is namelijk essentieel voor de toekomstige gebruikswaarde van de nieuwe module.
Mijn keuze voor user story's was een keuze voor structuur en eenduidigheid. Alle requirements worden namelijk opgeschreven als "*Als <rol>, wil ik dat <doel/wens >*". Deze notatie zorgt ervoor dat het voor de verschillende stakeholders duidelijk was wat er ontwikkeld ging worden.

Ik vind dat ik deze beroepstaak heb behaald omdat de requirements:

- traceerbaar zijn;
- volledig en dekkend zijn;
- duidelijk zijn voor de verschillende stakeholders;
- geprioriteerd en gevalideerd zijn door de stakeholders.

2.1 Opstellen van een gegevensmodel voor een database[niveau 3]
"Analyseren en beschrijven van gegevens over objecten in hun onderlinge samenhang. Toetsen van het gegevensmodel op volledigheid, duidelijkheid, consistentie en andere normen."

Er is een ERD en een RIM gemaakt waarin de samenhang tussen de objecten te zien is (zie paragraaf 6.3.2 en 6.3.3). In deze diagrammen zijn binaire relaties terug te vinden.
Ik heb ervoor gekozen alleen een RIM te maken en niet een RIM én een RRM. Ik achtte mijzelf deskundig genoeg in SQL om meteen een RIM te maken, die bovendien direct te importeren is in het DBMS.
Het vaststellen van de datatypes en de lengte van deze gegevens heb ik uitgevoerd op basis van de bestaande database van Rankinspector. Het ERD-diagram is gemaakt op basis van het klassendiagram (zie paragraaf 6.3.1). Om de kwaliteit van de ontwerpen vast te stellen, heb ik gebruik gemaakt van review-momenten met de stagebegeleider, zodat ik zijn grote technische kennis kon benutten.

Ik vind dat ik deze beroepstaak heb behaald, omdat ik:

- de juiste objecten heb gekozen;
- de objecten een duidelijke beschrijving heb gegeven;
- de samenhang tussen de verschillende objecten op correcte wijze heb vastgelegd.

3.1 Ontwerpen van softwarearchitectuur [niveau 3]
"Bepalen en beschrijven van de architectuur van applicatie(s). Bepalen en beschrijven van de verschillende subsystemen en componenten waarin een applicatie is onderverdeeld en de relaties tussen die delen."

Eerst heb ik uitgezocht wat de huidige architectuurstijl is van Rankinspector. Hierdoor werd duidelijk dat Rankinspector werkt met een 3-tier architectuur. Binnen de Logic Tier maakt Rankinspector gebruik van een vertakking van het MVC. Omdat ik de module op dezelfde wijze wilde ontwikkelen als de andere modules van Rankinspector, heb ik onderzocht wat de verschillen zijn tussen de officiële versie van MVC en de versie van Rankinspector (zie paragraaf 5.6).

Bij het ontwerp heb ik gekozen voor een klassen- en een use casediagram omdat men bij Traffic Builders bekend was met deze diagrammen. Zodoende was er weinig uitleg nodig tijdens de feedbackmomenten (zie paragraaf 6.2 en 6.3).
Van de use casediagrammen zijn views gemaakt die gebaseerd zijn op de use caseomschrijvingen (zie paragraaf 6.2.3).

Tijdens het interview met de developer bleek dat hij voor het importeren van data in de toekomst eenvoudig nieuwe datatypen wilde kunnen ondersteunen. Daarom heb ik onderzocht of het Factory Method Pattern voor dat doel geschikt was. Dat bleek niet het geval. Vervolgonderzoek wees uit dat het Strategy Pattern wel de vereiste kwaliteiten had.

Om de kwaliteit van de ontwerpen vast te stellen heb ik gebruik gemaakt van een aantal reviewmomenten met de stagebegeleider. Daarnaast waren er reviewmomenten met de opdrachtgever over de werking van het systeem, niet over de implementatie.

Ik vind dat ik deze beroepstaak heb gehaald omdat ik:
- de architectuur begrijpelijk heb beschreven voor de stakeholders;
- rekening heb gehouden met de belangen van de verschillende stakeholders;
- rekening heb gehouden met de beveiliging.

3.3 Het bouwen van de applicatie [niveau 3]
"Verfijnen en/of transformeren van het (detail)ontwerp van de systeemdelen (subsystemen, componenten, modules) van een applicatie. Bouwen en documenteren van de systeemdelen. Systeemdelen samenstellen tot een werkende applicatie."

De module heb ik zelfstandig ontwikkeld in een ontwikkelomgeving van Rankinspector. Tijdens de ontwikkeling heb ik gekeken hoe Strategy Pattern het beste geïmplementeerd kon worden. De klassen uit het klassendiagram zijn ontwikkeld en er is een CSV-klasse ontwikkeld om aan de juiste syntaxis van CSV te voldoen.

In paragraaf 6.4 staat hoe ik ben begonnen met het ontwikkelen van het prototype. Vervolgens is in hoofdstuk 7 te zien hoe ik de eerste versie van het product creëer. In hoofdstuk 8 is beschreven hoe ik de Rankinspector stijl voor de module heb geïmplementeerd.
Tijdens het ontwikkelen van de module heb ik rekening gehouden met de rest van Rankinspector door de mappenstructuur en de naamgevingen hetzelfde te maken. Ook de opbouw van de code en het gebruik van reeds aanwezige klassen, zoals de database en viewklasse, zijn op dezelfde wijze geïmplementeerd. Ik heb mij dus aan de huidige werkwijze van Traffic Builders gehouden, zodat het toekomstige onderhoud eenvoudiger is.

Ik vind dat ik deze beroepstaak heb behaald omdat ik:
- rekening heb gehouden met het hergebruik;
- rekening heb gehouden met de wijzigbaarheid;
- de applicatie op de juiste wijze heb samengesteld en werkend heb opgeleverd;
- gebruik heb gemaakt van een ontwikkelomgeving met testomgeving;
- op de juiste wijze gebruik heb gemaakt van de mogelijkheden van de programmeertaal.

11 Evaluatie

11.1 Evaluatie inceptiefase

Deze fase begon, zoals het hoort, met het maken van een plan van aanpak. Dit verliep goed, al had ik wat moeite met de planning omdat nog niet duidelijk was wat er moest gebeuren. Achteraf bezien had ik beter een globale planning kunnen maken en later per fase een gedetailleerde.
Ik heb mij aan de faseplanningen gehouden; mijn inschatting van de tijdsduur per fase was dus goed.
De opdrachtgever was bij de start van mijn project twee weken op vakantie. Om geen vertraging op te lopen heb ik er toen voor gekozen de interviews over de module alvast met de andere toekomstige gebruikers te voeren. Hierdoor moest het beeld dat ik van de nieuwe module kreeg, bij terugkeer van de opdrachtgever worden bijgesteld. Maar ik had rekening gehouden met verschillen in de wensen en eisen van de stakeholders. En ik was erop voorbereid dat het moeite zou kosten om één idee over de module te vormen onder de verschillende stakeholders. Hiervoor heb ik voldoende tijd genomen, waardoor de gebruikers tevreden zijn over het resultaat.

11.2 Evaluatie elaboratiefase

Het in kaart brengen van Rankinspector was een uitdaging omdat er geen documentatie van het systeem bestond. Er was alleen commentaar in de code bijgevoegd. Met de antwoorden van mijn geduldige stagebegeleider op al mijn vragen kon ik alsnog in kaart brengen hoe Rankinspector werkt.

Voor de rest verliep deze fase gestructureerd; ik wist goed wat ik ging doen en hoe ik het ging aanpakken. Het maken van de use cases verliep goed, maar ik merkte dat er behoefte was aan views. De opdrachtgever en de gebruikers wilden graag iets tastbaars zien, zodat ze een goed beeld zouden krijgen van het plan. Daarom heb ik er tijdens het maken van de use cases voor gekozen ook views te maken. Dit bleek een verstandige keuze. Het heeft ervoor gezorgd dat ik in een vroeg stadium van het project te weten kwam of mijn idee overeenkwam met het idee van de opdrachtgever en gebruikers.
In eerste instantie was het de bedoeling dat de applicatie geen module zou worden maar een standalone applicatie. Daardoor had ik enige tijd nodig om de functionaliteiten van Rankinspector te begrijpen. Die tijd heeft wel in mijn voordeel gewerkt, omdat ik veel ‘standaard’ functionaliteiten ontdekte in Rankinspector die ik niet zelf hoefde te ontwikkelen.

Het prototype wilde ik goed opbouwen. Omdat er bijna geen documentatie aanwezig was van Rankinspector kostte de bouw van het prototype tijdens deze fase redelijk wat tijd. Die had ik onder meer nodig om de gedocumenteerde klassen en libraries zoals PEAR.net en dhtmlx – de enig beschikbare documentatie – te raadplegen. Daarmee kon ik de vragen die ik had vaak wel oplossen.
Achteraf bezien had ik in deze fase beter gestructureerd de architectuur van Rankinspector en van de ontwikkelde module kunnen onderzoeken. Door mijn focus op het creëren van betrokkenheid bij de gebruikers en opdrachtgever heb ik daar te weinig tijd aan besteed. Ik wist aan het eind van de elaboratiefase hoe Rankinspector globaal werkt, maar een volgende keer wil ik dat exact weten.

11.3 Evaluatie constructiefase

De constructiefase vond ik een leuke fase die goed en vlot verliep. De module begon vorm te krijgen. De fout die ik ontdekte tijdens het testen van het prototype was snel hersteld.
Ik ben blij dat ik er halverwege de eerste iteratie voor heb gekozen heb om deze fase in twee iteraties te doen. Dat maakte de gebruikers en mijzelf duidelijk dat het in de eerste fase ging om de ontwikkeling van de functionaliteit en tijdens de tweede iteratie om de vormgeving. Die gerichte concentratie kwam de kwaliteit ten goede en werkte prettig.
Wat goed ging tijdens deze fase is het creëren van nuttige feedback, betrokkenheid en een goed product. Het maken van sequentiediagrammen zou ik de volgende keer uitgebreider en meer gestructureerd aanpakken.

11.4 Evaluatie transitiefase

Ik heb tijdens deze fase een presentatie gegeven die ik goed had voorbereid en die daardoor goed verliep. De aanwezigen waren niet alleen enthousiast over het product, maar ook stellig in hun overtuiging dat ze de nieuwe module gingen gebruiken. Dit vond ik leuk om te horen want daar had ik het uiteindelijk natuurlijk voor gedaan. Ik heb geprobeerd een product te maken dat aansluit bij de behoeften van de gebruikers. Tijdens deze laatste fase is duidelijk geworden dat ik daarin geslaagd ben.

Bronnen

In dit hoofdstuk staan de bronnen beschreven die ik voor mijn afstudeerproject heb geraadpleegd.

[ibm.com]
SOA terminology overview, Part 2: Development processes, models, and assets
http://www.ibm.com/developerworks/library/ws-soa-term2/index.html
Geraadpleegd op 8-11-2012

[Nicole de Swart boek]
Handboek requirements - Brug tussen business en ICT
Geraadpleegd op 24-10-2012

[Wikipedia 1]
Rational Unified Process
http://nl.wikipedia.org/wiki/Rational_Unified_Process#Fasering
Geraadpleegd op 9-11-2012

[Wikipedia 2]
Rational Unified Process
http://nl.wikipedia.org/wiki/Rational_Unified_Process#Maak_prototypes
Geraadpleegd op 9-11-2012

[pear.php.net]
PEAR DB
http://pear.php.net/manual/nl/package.database.db.php
15-11-2012

[pear.php.net/ execute]
Introduction - Prepare & Execute
http://pear.php.net/manual/nl/package.database.db.intro-execute.php
15-11-2012

[php.net/mysql-real-escape-string]
mysql_real_escape_string
http://php.net/manual/en/function.mysql-real-escape-string.php
17-10-2012

[csv voorbeeld]
Sample CSV to Import to AdWords Editor
https://docs.google.com/spreadsheet/ccc?key=0AuHDrmCX8MtydFdadzFHZVgzZXlpbW16Q283djZs
RVE#gid=0
17-10-2012

[dhtmlx documentatie]
Dhtmlx documentatie
http://docs.dhtmlx.com/doku.php
17-10-2012

[Insert Speed]
8.3.2.1. Speed of INSERT Statements
http://dev.mysql.com/doc/refman/5.0/en/insert-speed.html
26-10

[Wikipedia 3]
Model-view-controller-model
http://nl.wikipedia.org/wiki/Model-view-controller-model
Geraadpleegd op 29-11-2012

[php.net patterns]
Pattern
http://php.net/manual/de/language.oop5.patterns.php
Geraadpleegd op 11-12-2012

Definitielijst

<table>
<tr><th>Woord</th><th>Beschrijving</th></tr>
<tr><td>Match Types</td><td>Match Types zijn de vier verschillende mogelijkheden om in Google te adverteren op de zoekwoorden. De vier verschillende matchtypes zijn: broad match, phrase match, exact match en modified broad match. Wat deze vier verschillende types precies inhouden is te vinden in de bijlage over match types, zie document 5.</td></tr>
<tr><td>Adwords Account</td><td>Dit is een account van Google, waarin Adwords kan worden aangemaakt of toegevoegd.</td></tr>
<tr><td>Campagne</td><td>Hier worden de algemene instellingen geplaatst voor de advertenties die eronder geplaatst worden.</td></tr>
<tr><td>Adgroup</td><td>Hier worden de algemene instellingen ingevoerd voor de onderliggende advertenties.</td></tr>
<tr><td>Variabele</td><td>Een variabele – bijv. {HotelNaam} – vertegenwoordigt een waarde uit een XML-element.</td></tr>
<tr><td>Advertentie</td><td>Een advertentie bestaat in Google Adwords uit 5 onderdelen: Headline, Description Line 1, Description Line 2, Display URL en Destination URL.</td></tr>
<tr><td>Advertentietemplate</td><td>Een advertentie bestaat uit 5 onderdelen: titel, Description Line 1, Description Line 2, Display URL en Destination URL. Deze 5 onderdelen moeten worden ingevuld met variabelen. Hieronder staat een voorbeeld om aan te geven hoe een advertentietemplate eruit kan zien.

<table>
<tr><td>Headline:</td><td>Boek het {HotelNaam}!</td></tr>
<tr><td>Description Line 1</td><td>Dit hotel is een top hotel in {Plaats}</td></tr>
<tr><td>Description Line 2</td><td>Reserveer online tegen de laagste prijzen.</td></tr>
<tr><td>Display URL</td><td>www.booking.com/{HotelNaam}</td></tr>
<tr><td>Destination URL</td><td>{dUrl}</td></tr>
</table>

Een advertentietemplate die verwerkt wordt tot een advertentie kan er als volgt uitzien.

<table>
<tr><td>Headline:</td><td>Boek het Nouvel hotel!</td></tr>
<tr><td>Description Line 1</td><td>Dit hotel is een top hotel in Lons le Saunier</td></tr>
<tr><td>Description Line 2</td><td>Reserveer online tegen de laagste prijzen.</td></tr>
<tr><td>Display URL</td><td>www.booking.com/Nouvel-Hotel</td></tr>
<tr><td>Destination URL</td><td>http://www.booking.com/ hotel/fr/nouvel-lons-le-saunier.nl.html</td></tr>
</table>
</td></tr>
<tr><td>Adsense</td><td>Het programma van Google waarmee tekstadvertenties op websites weergegeven kunnen worden en waarvoor de site-eigenaar een vergoeding per klik krijgt.</td></tr>
<tr><td>Click</td><td>Een click is de registratie van het feit dat iemand op een advertentie-uiting heeft geklikt en daarmee de achterliggende informatie opvraagt.</td></tr>
<tr><td>Titel</td><td>Dit is de titel van een tekstadvertentie bij Google Adwords.</td></tr>
<tr><td>CPC</td><td>Cost Per Click. De prijs die een adverteerder betaalt als een klant op zijn advertentie klikt.</td></tr>
<tr><td>Destination URL</td><td>Het product url.</td></tr>
<tr><td>Omschrijving</td><td>Een omschrijving bestaat uit 2 regels. Het is de tekst onder de titel in een tekstadvertentie van Google Adwords. Er zijn 2 x 35 karakters aan tekst beschikbaar.</td></tr>
<tr><td>SEA</td><td>Beter bekend als Search Engine Advertising oftewel het tegen betaling plaatsen van advertenties boven en naast de ‘natuurlijke’ zoekresultaten. Bij SEA kunnen 3 doelstellingen worden onderscheiden. Ten eerste zoveel mogelijk vertoningen (bereik) genereren. Ten tweede zoveel mogelijk clicks realiseren. Tot slot zoveel mogelijk conversies realiseren.</td></tr>
</table>