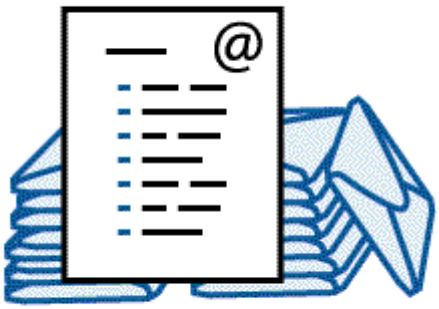
"Realiseren mailinglist-applicatie in ASP.NET "

Afstudeerscriptie

Jeroen Cramer 20021998 VIA2

Periode:
2006.1.2 - 2006.2.1
15 Mei – 6 Oktober



Examinatoren:
K.B. de Jong
T.H.M. Spaan

Afstudeerbedrijf:
Mediaweb Internet Integrators b.v.
Bedrijfsmentor:
P. Van Marwijk

Referaat

Cramer, J.

Dit is de afstudeer scriptie van J. Cramer ten behoeve van de opleiding Vormgeving en Interactie van Ontwerp (VIA) aan de Haagse Hogeschool.

De afstudeerstage vond plaats bij Mediaweb Internet Integrators b.v. te Noordwijk.
De opdracht voor deze afstudeerstage was het ontwerpen en bouwen van een mailingapplicatie in ASP.NET.

De scriptie is bedoeld voor de examinatoren, de gecommitteerde en andere betrokkenen bij de afronding van de stage van de student. Van de lezer wordt verwacht dat deze op HBO - niveau heeft genoten of vergelijkbare ervaring heeft. Voorkennis van de modeleer - techniek UML (unified modeling language), ER-diagrammen en SQL-syntax is noodzakelijk bij het lezen van sommige delen van de scriptie met betrekking tot de werking van de applicatie en de structuur van de database.

Enkele kernwoorden uit dit verslag zijn:
- Afstuderen
- Eindverslag
- Haagse Hogeschool
- 8 fasen – model
- UML
- ASP.NET
- Mailing

Voorwoord

Ik wilde de mensen van mediaweb en het pand AanZee bedanken voor de geboden mogelijkheid meer van mezelf te leren kennen. Ik wilde Rob Hoek en Iskander Prins bedanken voor hun hulp met de scriptie en feedback. Verder wil ik mijn vrienden, klasgenoten en familie bedanken voor de gebrachte ontspanning en hun luisterend oor ten tijde van de stage.

Noordwijk 30 september 2006, Jeroen Cramer

Inhoudsopgave

Inleiding

Dit is de afstudeerscriptie van Jeroen Cramer als de afronding van de afstudeerstage gedurende de periode 2006.1.2 – 2.1 gelopen bij Mediaweb Internet Integrators.
Het doel van de scriptie is inzicht te verschaffen in het verloop van de door de student gelopen afstudeerstage door het uiteenzetten van het proces en de producten van de stage.

De scriptie bevat de volgende delen:

- Hoofdstuk 1: Situatieschets van de afstudeerstage
- Hoofdstuk 2 - 6: Trajectbeschrijving aan de hand van de doorlopen fases met elk een apart hoofdstuk
- Hoofdstuk 7: Aparte stukken van onderdelen uit de voorgaande hoofdstukken die meer diepte vereisten of die de vaart uit de voorgaande hoofdstukken haalde
- Hoofdstuk 8: Evaluatie van het proces en de producten
- Woordenlijst, referentielijst en aanvullende bijlagen

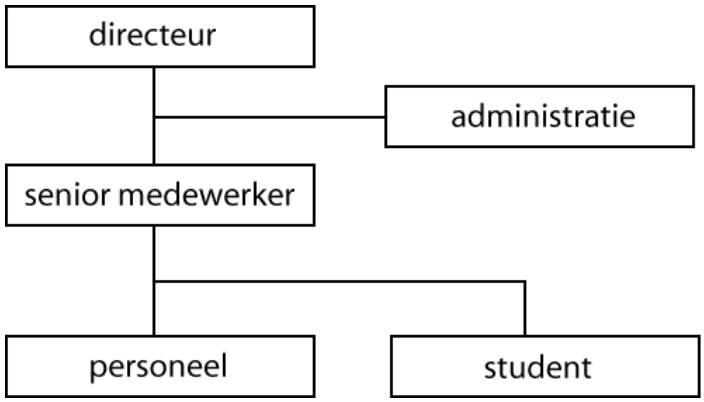
Hoofdstuk 1. Oriëntatie

Het doel van dit hoofdstuk is om de lezer te helpen om zich te oriënteren op het afstudeerproject. Hier wordt de basis gelegd waar de overige hoofdstukken verder op in zullen gaan.

Naast een beschrijving van de opdracht en de werkomgeving worden de gekozen methoden en technieken samen met de stappen die voor het starten van het project hebben plaatsgevonden besproken.

1.1 Bedrijfsoriëntatie

Het bedrijf is in 1995 opgericht door Eric van Hall als onderdeel van The Box Office B.V., een audiovisueel productiebedrijf dat sinds eind jaren tachtig talloze TV, video en filmproducties maakte. Er worden nu geen audiovisuele producties meer gemaakt en het bedrijf heeft zich al geruime tijd gericht op de markt van het Internet, met de nadruk op het maken en hosten van Websites. Naast webdesign en hosting, speelt consultancy een steeds grotere rol binnen het dienstenpakket van het bedrijf. Mediaweb beschikt over veel kennis en ervaring op Internetgebied en streeft ernaar deze waar mogelijk uit te breiden. Zelfstandigheid, betrokken zijn en serieus omgaan met werk zijn punten die worden gewaardeerd.
Het bedrijf bestaat uit de directeur en 5 werknemers en zit samen met 2 andere bedrijven, bureau aan zee en events aan zee, in één pand. Administratieve zaken en receptiewerk worden door een tweetal personen gedaan die door het pand zijn ingehuurd. Voor veel zaken is de senior medewerker, tevens de bedrijfsmentor en opdrachtgever, het aanspreekpunt binnen het bedrijf.
Van tijd tot tijd trekt het bedrijf afstudeerders aan voor het uitvoeren van een opdracht, bij het uitvoeren van deze opdracht worden ze begeleid door een bedrijfsmentor en kunnen ze advies vragen aan de overige leden van het bedrijf.



Figuur 1.1 organogram Mediaweb

Binnen het bedrijf wordt al lange tijd ASP gebruikt voor het maken van de websites en webapplicaties en nu vindt er een geleidelijke overgang naar ASP.NET-omgeving plaats. De keuze om over te gaan naar ASP.NET was dat deze een groot aantal verbeteringen bied ten opzichte ASP.
Twee van de werknemers die ervaren zijn met ASP zijn nu ieder een project in ASP.NET aan het uitvoeren om aan de veranderingen te wennen. De nieuwste werknemer heeft kennis van ASP.NET meegenomen, hij kan en wordt door zijn collega's geraadpleegd.
Bijeenkomsten om praktische ASP.NET problemen te behandelen staan in de planning voor na de zomervakantie.
Omdat het bedrijf in deze overgang zit is bij het formuleren van de afstudeeropdracht rekening gehouden met deze ontwikkeling. In de toekomst zullen alle projecten in ASP.NET gemaakt worden, daarom dient de opdracht uitgewerkt te worden in de ASP.NET omgeving.

1.2 Opdrachtomschrijving

Voor het verzenden van standaard mailings gebruikt Mediaweb al geruime tijd de zelf ontwikkelde Newspublisher waarmee klanten zelf een nieuwsartikel kunnen opstellen en deze als email kunnen verzenden. Mediaweb wil haar mailingdienst vernieuwen zodat dit beter aansluit op eisen en wensen van de klant.
Klanten willen grote hoeveelheden emails per mailing kunnen verzenden. Voor volumineuze mailings wordt nu nog een alternatieve oplossing gebruikt omdat de Newspublisher deze niet aankan. De emails worden in groepen aangemaakt met daarvoor geschreven scripts en daarna klaargezet om verzonden te worden. Deze handmatig uitgevoerde mailings kosten verhoudingsgewijs veel tijd en inspanning.
Deze oplossing wordt ook gebruikt voor het op maat maken van een email voor abonnees die zich bij verschillende categorieën hebben ingeschreven. Ook dit is iets wat Mediaweb graag standaard aan de klanten wil aanbieden, maar omdat newspublisher niet categoriegebonden inhoud voor de mailing ondersteunt moet dit buiten het programma gedaan worden.
Verder willen klanten inzicht hebben in de effecten van een mailing. Op het moment wordt een effectmeting van een mailing door Mediaweb zelf uitgevoerd, een proces dat arbeidsintensief is. De applicatie die de mailings verzorgt is niet toereikend om in te kunnen spelen op deze wensen en het zou meer tijd kosten om de applicatie aan te passen dan deze te vervangen door een nieuwe applicatie.

De afstudeeropdracht wordt binnen Mediaweb voor het bedrijf zelf uitgevoerd. Het doel van de afstudeeropdracht is het ontwikkelen van een mailingapplicatie die met de middelen binnen het bedrijf gerealiseerd kan worden en door klanten van Mediaweb gebruikt zal worden. Van deze applicatie wordt verwacht dat deze beschikt over:

- een functionaliteit om een op een template gebaseerde (volumineuze) mailing stapsgewijs te verzorgen
- een ontwerp dat aansluiting vindt op bestaande ontwikkeltechnieken binnen Mediaweb
- de mogelijkheid om in de toekomst uitgebreid te worden door Mediaweb
- een koppeling met de bestaande abonnee databases
- een functionaliteit die inzicht biedt in de effectiviteit van
- een effectenmeting door middel van statistieken toonbaar, in het bijzonder:
    - de weergave van het aantal verzonden emails
    - de verwerking en weergave van bounces en errors
    - de weergave van het aantal geopende links vanuit de mailing

In het kader van de afstudeeropdracht zullen de volgende activiteiten verricht worden:

- Opstellen van een plan van aanpak
- Informatie vergaren voor functioneel ontwerp mailingapplicatie
- Gedocumenteerd functioneel ontwerp maken van mailingapplicatie
- Maken van een grafisch ontwerp
- Maken van een interactief ontwerp
- Informatie en kennis vergaren over programeer- en scripttalen
- software en hardware die bij de realisatie gebruikt worden
- Realisatie functioneel ontwerp
- Testen mailingapplicatie
- Invoering mailingapplicatie

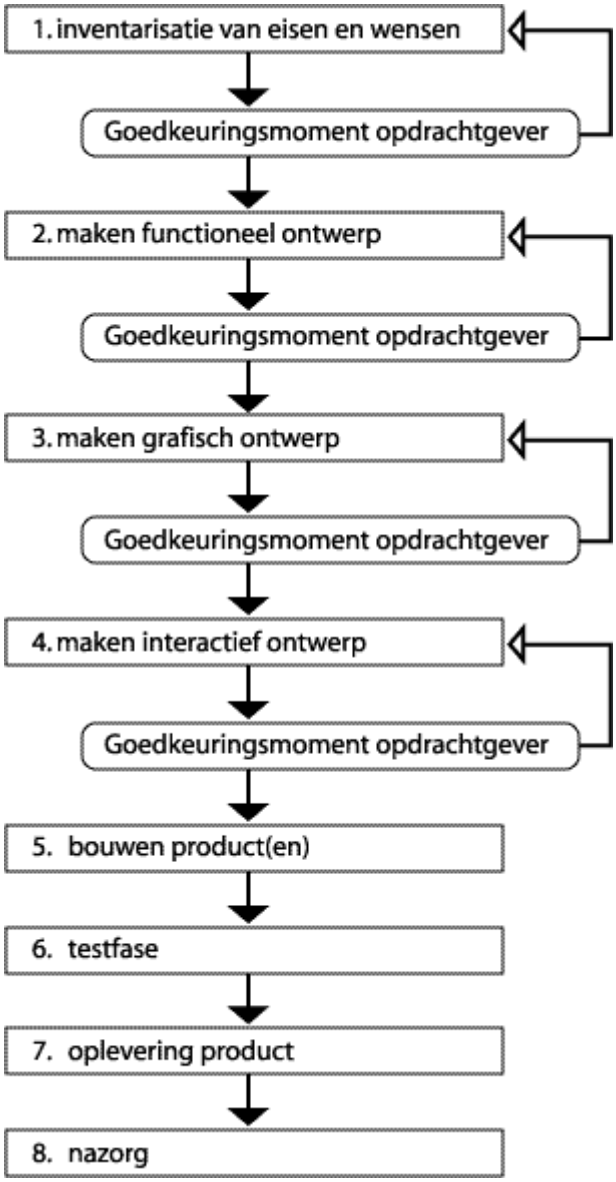
De volgende producten zullen opgeleverd worden:

- Plan van aanpak
- Functioneel ontwerp van de applicatie
- Grafisch ontwerp van de applicatie
- Interactief ontwerp van de applicatie
- Mailingapplicatie
- (kort) rapport over de testresultaten

1.3 Methodiekkeuze

Voor het uitvoeren van het project heb ik gekozen voor het 8 fasen model ontwikkeld door Mediaweb zelf. Deze methode wordt gebruikt voor het maken van websites en webapplicaties voor klanten van het bedrijf.
Zoals de naam al doet vermoeden bestaat de methode uit 8 fases waarvan de 1e 4 fases het ontwerp omvatten en de laatste 4 de realisatie en invoering (*figuur 1.1*). Afhankelijk van feedback en de terugkoppelmomenten met de opdrachtgever kunnen één of meer van de fases opnieuw doorlopen worden maar in principe worden alle fases eenmaal doorlopen. Projectgerichte afspraken en beslissingen zijn niet opgenomen in deze methode en zullen door de projectleden moeten worden opgevangen in bijvoorbeeld een plan van aanpak of onderlinge afspraken.

De belangrijkste reden om deze methode te gebruiken was omdat ik een andere methode wilde toepassen dan de reeds aan mij bekende methoden, IAD en extreme programming. IAD heb ik tijdens mijn opleiding veel gebruikt en voor een kort en klein project als dit vond ik IAD niet echt geschikt. Ik zou niet kunnen inspelen op de voordelen van de methode in dit éénmans project. De pilots zouden niet parallel ontworpen en ontwikkeld kunnen worden, de hoeveelheid en de diepte van de documentatie zou verlaagt moeten worden en gezien de beschikbare tijd kon elke fase waarschijnlijk maar één keer doorlopen worden.
Extreme programming is niet een methode die ik graag zou gebruiken, er wordt geen gebruik gemaakt van documentatie en de realisatie is direct verbonden aan hoe goed de opdrachtgever weet wat hij/zij wil en hoe goed jij bent om dit te achterhalen. Deze methode kan bij verkeerde interpretatie tot gevolg hebben dat de applicatie een flink aantal aanpassingen of herschrijvingen moet doorlopen, deze gestructureerd chaotische vorm van ontwikkelen is te foutgevoelig voor een gebruiker van een nieuwe programmeertaal en kan een behoorlijke deuk in je zelfvertrouwen opleveren wanneer het misloopt.



Figuur 1.2 8 fasen - model

Het 8 fasen model brengt een aantal voordelen met zich mee. Het is bekend binnen het bedrijf en vereist geen aanpassingen van de werknemers om te begrijpen waarmee ik bezig ben, wat in het geval van zowel de bedrijfsmentor als de opdrachtgever een groot voordeel kan zijn.
Het aantal documenten dat aangeleverd moeten worden is beperkt tot 3, meer mag maar is niet noodzakelijk. Om niet al te lang stilstaan bij de documentatie vind ik alleen maar prettig, documentatie dient gemaakt te worden naar de schaal van het project en de kwaliteit en de aard van het eindproduct(en).
Er zijn voorbeelden van de documentatie aanwezig waardoor het voor mij makkelijker wordt om deze te maken, de documenten die ik zal opleveren zijn bij de werknemers bekend waardoor ik eenvoudig feedback kan krijgen zonder veel inspanning van de lezer. Mijn ervaring is dat in een werkomgeving collega's sneller en vaker feedback willen geven wanneer er weinig inspanning zijn vereist.
De voordelen die deze methode met betrekking tot de werkomgeving, het feit dat deze methode nieuw voor mij is en het feit deze methode geschikt is om voor het project te kunnen gebruiken hebben mij voor deze methode laten kiezen.

1.4 Techniekkeuze

Om de verschillende fases en onderdelen binnen het project effectief te doorlopen kan er naast een ontwikkelmethode gebruik gemaakt worden van technieken. De volgende technieken zijn voor dit project gekozen, waarvan de meeste tijdens het project:

- Unified Modeling Language, het onder andere object georiënteerd in kaart brengen van processen en software systemen
- Reverse engineering, het verkrijgen van informatie en het schrijven van ontwerpen en andere documentatie door het uit elkaar halen en analyseren van een bestaande applicatie waarvoor geen documentatie is gemaakt tijdens het ontwikkelen ervan
- Flowcharting, het visualiseren van een proces in stappen met behulp van vastgestelde iconen

UML, Unified Modeling Language

Bij het formuleren van de opdrachtomschrijving was ik niet zeker wat er nodig zou zijn voor het maken van de verschillende ontwerpen maar ik was me wel bewust dat er voor het maken van het functionele ontwerp een logische en beschrijvende weergave nodig zou zijn van de nog niet tastbare applicatie. Een goede weergave zegt vaak evenveel als een groot aantal pagina's tekst en een lezer of een ontwerper kan beter het overzicht houden dan wanneer het enkel in tekst beschreven zou staan.

Ik ging er vanuit dat de interactie met de applicatie, de werking van de applicatie en de onderlinge relatie van de objecten zeker ter sprake moesten komen om een duidelijk beeld te schetsen. UML biedt de mogelijkheid om vanuit verschillende invalshoeken een systeem te kunnen beschrijven en het is een kwestie van keuze welke aspecten gekozen worden. Omdat ik al een nieuwe ontwikkelmethode had gekozen wilde ik niet teveel hooi op mijn vork nemen door hiervoor een nieuwe techniek toe te passen.

Reverse engineering

Bij het maken van het grafisch ontwerp moest ik de huisstijl van de applicatie afstemmen met die van een door het bedrijf gemaakt Content Management System. De gebruikte huisstijl was nooit gedocumenteerd en ik was genoodzaakt om de huisstijl te achterhalen aan de hand van het bestaande product, wat ik heb gedaan met behulp van reverse engineering.

Reverse engineering is een term die gebruikt wordt voor het uit elkaar halen van een product om de werking ervan te achterhalen, vaak om verbeteringen aan te brengen in het ontwerp. In het ICT- zakboekje wordt reverse engineering beschreven als het herontdekken van de specificaties van een programma uit de code zelf met als uitgangspunt dat de consistentie tussen de documentatie en het systeem niet zorgvuldig is bewaard en dat de kennis van het programma enkel is terug te vinden in de code.

Door het programma te kiezen en het gewenste resultaat te bepalen, in dit geval een huisstijl, kan aan de hand van het uitpluizen van de code en de informatie inpassen in de gekozen methode of techniek alsnog de benodigde documentatie of informatie worden verkregen. Dit is in essentie reverse engineering.

Flowcharting

Hoewel de beslissing om deze techniek te gebruiken pas tijdens het project zelf viel vond ik een kleine verwijzing wel op zijn plaats. Later in het project, tijdens het maken van het interactieve ontwerp, heb ik flowcharts gebruikt om snel de werking van de verschillende onderdelen van de mailingapplicatie uiteen te zetten. Ik hoorde via een collega erover en was benieuwd of deze zo effectief was als hij beweerde. Bij wijze van experiment heb ik deze techniek toen toegepast. Er is meer over te lezen in het hoofdstuk 2, fase 3 interactief ontwerp.

1.5 Gemaakte keuzes voor start

De opdracht vereiste dat het ontwerp geïmplementeerd zou worden in ASP.NET, waar ik geen vergelijkbare voorkennis van had bij het aannemen van de opdracht. Dit had een aantal gevolgen voor het project, bij de start zou ik veel niet weten en ik zou moeten leren tijdens het project.
Mijn voornemen was om het kennisgebrek vanaf de eerste fase aan te pakken om zodoende de problemen in latere stadia op te vangen. Omdat ik niet wist hoe snel ik de praktische kennis zou kunnen vergaren heb ik op voorhand rekening gehouden met de meest waarschijnlijke problemen bij een gebrek tijdens de ontwerpfases. De ontwerpfases zijn de eerste fases die doorloopt worden en zijn vrij van programmeren dus het vergaren van praktische kennis van de ASP.NET omgeving heeft hier het laagste tempo terwijl het profijt ervan het hoogst is. Je weet dan namelijk wat mogelijk is of kan dat eenvoudig(er) achterhalen en kan dat in het ontwerp verwerken.

De ontwerpen zouden per fase minder globaal zijn en meer detail bevatten, dit is normaal bij een ontwikkelmethode zoals IAD waarin het gehele project in steeds groter detail in opvolgende (en itererende) fases wordt uitgewerkt. Hierbij worden alle aspecten van een project in steeds kleinere bouwdelen uitgewerkt totdat men een geschikt ontwerp heeft. Maar in het 8 fasen model krijgt elk aspect van een ontwerp (systeemeisen, functionaliteit producten, interface producten) een fase toegewezen. De details moeten in elke fase al aanwezig moet zijn omdat deze niet nog een keer in een volgende stap uitgewerkt zullen worden. Vanwege de toenemende kennis verwachtte ik pas bij het grafisch ontwerp aspecten van de ASP.NET omgeving te verwerken in het ontwerp.

De kennis voor het realiseren van het ontwerp zou ik gaandeweg opdoen en bij het maken van het functioneel ontwerp zou veel van die kennis nog oppervlakkig zijn. Ik schatte daarom ook dat er geen gedetailleerd ontwerp kon gemaakt worden zonder mogelijk in de

problemen komen door een onrealistisch ontwerp of om (meerdere malen) het uitgewerkte ontwerp aan te passen in latere fases.
Om deze reden wilde ik op voorhand bij het maken van het functioneel ontwerp me richten op het uitwerken van de systeemeisen zonder naar op de ontwikkelomgeving gerichte oplossingen te kijken. Gelijktijdig de mogelijkheden uitzoeken en invulling geven aan systeemeisen zouden zonder de benodigde ervaring enkel tijd opslokken in deze fase en in latere fases, als ik de kennis tijdig bezat zou ik mijn keuze herzien.

Veel van de afwegingen en beslissingen bij het functioneel en grafisch ontwerp zouden op de theorie gebaseerd zijn zonder toetsing met de werkelijkheid. Ik hield daarom rekening met het feit dat ik tijdens het bouwen regelmatig moest controleren of de gekozen aanpak wel zou werken. Ik vond de fase waarin de applicatie werd gebouwd te laat om deze vorm van controle uit te voeren omdat na deze fase maar één terugkoppelmoment zou zijn, wat ik te laat in het project vond.
Om dit op te vangen wilde ik zoveel mogelijk functionaliteit voortijdig uitproberen in een prototype of bij het maken van het interactief ontwerp. Omdat het interactief ontwerp niet bedoelt is om functionaliteit in te plaatsen wilde ik de verschillende functies proberen te los bouwen tijdens het functioneel ontwerp. Als er niet voldoende tijd voor zou zijn dan kon ik terugvallen op het interactief ontwerp.

Hoofdstuk 2. Fase 1, Inventarisatie wensen en eisen

Na de officiële kennismaking met het mijn collega's, het bedrijf, haar buren en de installatie van mijn werkplek ben ik begonnen aan de eerste fase van mijn gekozen ontwikkelmethode.

Voordat ik met de opdrachtgever ben gaan praten wilde ik de wensen en eisen op papier plaatsen die in eerdere gesprekken al boven water waren gekomen. Met dit in gedachten ben ik mijn aantekeningen en de opdrachtomschrijving nagegaan. Mijn bedoeling was om deze lijst in het eerste gesprek te gebruiken om de opdracht met de opdrachtgever op te halen en tegelijkertijd feedback te krijgen om de bestaande lijst uit te breiden. Om de lijst overzichtelijk te maken heb ik de eisen geclusterd in groepen. Alle eisen pasten in één van de volgende groepen:

- Gebruiker, gebruikersgerelateerde zaken zoals beheer en administratie
- Mailinglist, de mailinglist zelf of het proces van er één vervaardigen
- Mailing, het voorbereiden en verzenden van een mailing
- Effectmeting, het meten v.d. effecten v.e. gemaakte mailing
- Email, het vervaardigen en verspreiden v.d. email van de mailing

Ik heb overwogen om nog een indeling te maken op ontwerpaspect (basis, functioneel, interface, operationeel en performance) of prioriteit (noodzakelijk, gewenst en luxe) maar heb daar vanaf gezien omdat de lijst als gesprekstof diende, de lijst bestond nu immers al uit geclusterde gespreksonderwerpen. De andere indelingen zouden pas bij een meer complete lijst van nut zijn wanneer er meer informatie over het project zelf bekend was en er een terugkoppeling met de opdrachtgever was geweest.

Na de lijst afgerond te hebben is er een afspraak gemaakt met de opdrachtgever waarbij de lijst is doorgenomen en het beeld van het eindproduct is uitgebreid, de wensen en eisen die uit dit gesprek zijn gekomen zijn later verwerkt. Omdat er geen voorbeelden waren van eerder gemaakte wensen en eisen - lijsten heb ik er zelf één geformuleerd, deze voorlopige versie zou gebruikt worden bij het goedkeuringsmoment en de feedback die ik van deze lijst zou krijgen zou ik gebruiken om de lijst bij te schaven tot de definitieve versie.
Ik kreeg tijdens de bespreking het verzoek om de wensen en eisen - lijst verder in te delen op prioriteit om zo een beeld te kunnen krijgen van de onderdelen en processen waar de applicatie op zijn minst uit zou bestaan, wat naderhand is gedaan en waarvan hieronder een kopie van de definitieve versie te zien is.

BASIS

- Elke klant die gebruiker is moet een gebruikersprofiel hebben
- Het gebruikersprofiel moet beheerbaar zijn door de medewerkers van Mediaweb
- Werknemers van Mediaweb moeten in staat zijn gemaakte templates voor klanten beschikbaar te maken in hun gebruikersprofiel
- De applicatie moet het aantal verzonden emails bijhouden per gebruiker voor administratieve doeleinden
- De applicatie moet via het internet benaderbaar zijn
- De applicatie moet onderscheid maken tussen de gebruikers onderling

- Mailinglists moeten ingevoerd kunnen worden door gebruikers
- De applicatie moet een koppeling kunnen maken met bestaande databases
- De applicatie moet in staat zijn 10.000 of meer emails in één mailing te verzenden
- De applicatie moet uit een mailinglist dubbele inschrijvingen kunnen halen

- Het maken van een email voor een mailing moet in stappen doorlopen worden
- Onafhankelijk van de technische achtergrond van een gebruiker moet een email voor een mailing gemaakt en verzonden kunnen worden
- Een mailing moet naar alle abonnees van de mailinglist of een selectie van de subgroepen verzonden kunnen worden
- Errors en bounces van een mailing moeten verwerkt kunnen worden door de applicatie en weergegeven worden in de effectmeting

- De gebruiker moet de email van een mailing kunnen zien zoals deze verzonden zou worden en aanpassingen kunnen aanbrengen alvorens deze te verzenden
- De email van een mailing moet naast een voor alle groepen gelijk gedeelte, een voor elke groep apart gedeelte kunnen bevatten
- Een nieuwsbrief moet opgeslagen kunnen worden
- De opmaak van een email voor een mailing wordt met behulp van een template verzorgt
- Een opgeslagen nieuwsbrief moet opgeroepen kunnen worden en vervolgens verzonden

COMFORT

- De applicatie moet uit een mailinglist foute email adressen kunnen halen
- De applicatie moet de kosten van een mailing kunnen weergegeven

LUXE

- De effecten van emails verzonden naar een mailinglist moeten gemeten kunnen worden
- Bij de effectmeting moeten de verkregen gegevens in zowel tabellen als in grafieken kunnen worden weergegeven
- Mailinglists moeten geïmporteerd kunnen worden door gebruikers
- De mailing moet kunnen worden verzonden volgens een bepaald tijdschema
- Er moet een controle uitgevoerd worden op de mailingemail of deze als spam gezien zal worden

Fig. 2.1 de systeemeisen

Opvallend is dat er aan de interface geen eisen zijn verbonden. De opdrachtgever gaf aan dat de interface net als de rest van de applicatie "logisch" in elkaar moest zitten waarmee hij bedoelde dat de interface en de werking van de applicatie geen verwarring moesten veroorzaken bij de gebruiker maar hem of haar juist moesten ondersteunen. Ik heb het hierbij tijdelijk gelaten met de intentie het "logisch" verder uit te diepen in het functioneel ontwerp en bij het grafisch ontwerp om te zetten in iets tastbaars. De wensen en eisen - lijst is er één van de opdrachtgever, bij de invoering en ontwerp worden de eisen door de ontwerper gesteld en verwerkt. Was de opdrachtgever een extern geweest dan was ik grondiger geweest bij het formuleren van de eisen maar omdat de opdrachtgever de ontwikkelmethode kent en ervaring heeft met het doorlopen van projecten hoeven de doorzichtige eisen niet verwoord te worden maar konden deze verwerkt worden in de ontwerpen.

2.1 Voorbereiden op een functioneel ontwerp

Voordat ik goedkeuring heb gekregen voor de definitieve wensen en eisen - lijst en door kon gaan naar de volgende fase ben ik begonnen ik aan een kladversie van het functioneel

ontwerp. Het beeld van het eindproduct wilde ik alvast vorm gaan geven nu het nog vers in mijn geheugen zat van de voorgaande besprekingen. Het maken zou betrekkelijk weinig tijd in beslag nemen maar bij de bespreking waar anders alleen de wensen en eisenlijst goedgekeurd zou worden kon nu ook de toekomstige applicatie worden besproken.

Op deze manier zou ik alvast feedback kunnen krijgen op het functioneel ontwerp en zouden er eventueel nieuwe systeemeisen uit het gesprek kunnen komen. Daarnaast zou de bespreking meer inhoud voor de opdrachtgever krijgen maar ondertussen niet veel meer van hem vereisen, gezien er geen andere onderwerpen aan het gesprek zouden worden toegevoegd. Er werd namelijk nog steeds over hoe de applicatie eruit zou gaan zien gesproken.

Naast het bespreken van het uiterlijk en structuur van het functioneel ontwerp zijn er een aantal bepalingen gemaakt over de applicatie:

- Het maken van een mailing zou één proces worden, voorheen was daar een tweedeling in gemaakt tussen het maken en het verzenden van de mailing
- Het proces van de mailing wordt in opeenvolgende stappen doorlopen
- De kostenberekening wordt (tijdelijk) achterwege gelaten omdat het businessmodel voor de applicatie nog niet gevormd is.
- Een beheerder/administrator van de applicatie moet toegang hebben tot zowel de mailingfuncties en de overzichten van de gemaakte mailings als het gebruikersbeheer.

Na de bespreking heb ik zowel het voorlopige functioneel ontwerp als de wensen en eisen - lijst aangepast. Het voorlopige functionele ontwerp is daarna gebruikt in fase 2 als basis voor de UML diagrammen en samen vormden zij de definitieve versie van het functioneel ontwerp, meer daarover in het hoofdstuk 3.

2.2 Herindeling systeemeisen

Bij het goedkeuringsmoment waren de eisen gegroepeerd aan de hand van de delen binnen de applicatie die ze vertegenwoordigde, onveranderd sinds de vorige versie, en ingedeeld op prioriteit. De prioriteitstoekenning is gedaan aan de hand van een indeling die op school bij het gebruik van IAD aangeleerd was. De indeling werd gedaan op basis van drie categorieën (basis, comfort en luxe) waarin alle systeemeisen geplaatst worden, de categorie bepaald de volgorde waarin de eis verwerkt wordt en de kans dat deze (gedeeltelijk) gerealiseerd zou worden.

De basis systeemeisen zijn de noodzakelijk voorwaarden voor de applicatie en worden verplicht verwerkt in het (eind)product en worden ook als eerste geïmplementeerd. De comfort systeemeisen hebben een lagere prioriteit als de basiseisen maar zijn door hun toegevoegde waarde aan het ontwerp van de applicatie belangrijk genoeg om minimaal het merendeel ervan terug te vinden in het (eind)product, deze groep wordt logischerwijs na of tegelijk met de basiseisen geïmplementeerd. De categorie luxe systeemeisen zijn de eisen die niet noodzakelijk zijn voor de applicatie maar wel een toegevoegde waarde hebben of gewoonweg veel tijd nodig hebben om verwezenlijkt te worden.
De systeemeisen voor de effectenmetingen zijn verplaatst naar de luxe eisen omdat een werkende applicatie nodig zou zijn voor de metingen en omdat het proces van een effectmeting makkelijker te begrijpen en te implementeren was wanneer er kennis en ervaring was opgedaan over het maken en verzenden van een mailing. De overige systeemeisen die bij de categorie comfort of luxe zijn ingedeeld werden als een extraatje voor de gebruiker beschouwd door de opdrachtgever (*figuur 2.1*).
Met de huidige indeling was eenvoudig een planning te maken voor de realisatie. Basis zou per definitie ingevoerd worden, comfort voor het overgrote deel en luxe waar de tijd het toe zou laten. Deze indeling heb ik doorgenomen met de opdrachtgever en na wat kleine aanpassingen was de wensen en eisen - lijst goedgekeurd.

De wensen en eisen - lijst was overzichtelijk, omvatte alle delen van de applicatie en was samen met de gegeven beschrijving van de applicatie groot genoeg om een start van het functioneel ontwerp mee te maken, dus heb ik afgezien van verdere aanpassingen. De ontwerpaspecten evenals de applicatie zouden bij de volgende fases met behulp van verschillende ontwerpen voldoende aan bod komen dus een herindeling op de ontwerpaspecten, had ik besloten, was overbodig en zag ik af van verdere gesprekken over enkel de systeemeisen. Bij latere terugkoppelmomenten en gesprekken zouden eventueel gemiste of verkeerd geïnterpreteerde eisen boven water komen en in de ontwerpen verwerkt worden.

In de latere fases zijn er geen grote veranderingen in de wensen en eisen – lijst gemaakt en de kleinere aanpassingen zijn direct in de ontwerpen ingevoerd. Ik denk dat pas wanneer er een prototype of een voltooid voorbeeld van de applicatie beschikbaar zou zijn er eventuele grote veranderingen plaats zouden kunnen vinden. Het beeld van de applicatie van de opdrachtgever namelijk zou pas kunnen veranderen als deze getoetst kon worden met een gerealiseerd model ervan.

2.3 Terugkoppelingen met de opdrachtgever

De redenen waarom ik tijdens de verschillende ontwerp fases steeds met de opdrachtgever in overleg ging wanneer een product af was waren om de opdrachtgever op de hoogte te houden, om te toetsen of het product overeen kwam met de wensen van de opdrachtgever en om de opdrachtgever betrokken te houden en hem het gevoel te geven dat hij invloed op het project had.
In de meeste gevallen werd dit gedaan door een kopie aan de opdrachtgever te overhandigen en bij een volgende bijeenkomst de inhoud te bespreken, soms door gewoon een gesprek aan te knopen over het project. De feedback van deze terugkoppel - momenten worden overigens niet hier behandelt maar in het geschikte deel in de bijbehorende fase.
Ik had het voordeel dat mijn opdrachtgever zelf een ICT-specialist was en dat veel van de subproducten tijdens het project begrijpelijk voor hem waren, was hij dit niet geweest dan hadden de besprekingen meer voorbereiding vereist en waren er waarschijnlijk grotere gedeeltes in één keer afgehandeld.

2.4 Maken van het Plan van Aanpak

Omdat binnen deze ontwikkelmethode geen aparte fase of aanpak is bepaald om de organisatie van het project te bespreken heb ik besloten om deze buiten de methode, voornamelijk voor mijzelf, in kaart te brengen.
Ik koos voor het maken van een Plan van Aanpak, het was een beproefde manier om afspraken en organisatorische beslissingen aangaande een project in kaart te brengen en ik zag geen reden om daarvan af te wijken. Later las ik terug in het leerplan dat een soort van Plan van Aanpak werd verwacht. Na het maken van het Plan van Aanpak heb ik deze laten goedkeuren door de opdrachtgever en later beschikbaar gesteld voor de examinatoren bij het bedrijfsbezoek.

2.5 SPAM met betrekking tot de applicatie

Omdat de applicatie grote hoeveelheden emails zou verzenden maakte ik mij zorgen over de invloeden van spam. Elke emailgebruiker heeft er last van en het was niet de bedoeling dat de applicatie als verzendmiddel van spam gebruikt zou worden of per ongeluk als spam verspreider herkend zou worden. Met deze gedachte besloot ik mijn kennis over spam uit te breiden.

Hoewel ik veel verschillende omschrijvingen heb kunnen vinden van spam houd ik de volgende aan: Spam is een email die zonder medeweten of toestemming van de ontvanger is verzonden en waar de ontvanger geen behoefte aan heeft, UBE en UCE zijn hier voorbeelden van. Unsolicited bulk email (UBE) is een mailing waarbij in grote hoeveelheden worden verzonden en die onderling (bijna) identieke emails bevat waarvoor de ontvangers zich niet bewust voor hebben ingeschreven. Unsolicited commercial email (UCE) is hetzelfde als UBE maar heeft commerciële doeleinden.

Tijdens het zoeken naar informatie over SPAM en de tegenmaatregelen vond ik de website van de Mail Abuse Prevention System (MAPS), een non-profit organisatie die zich toewijd aan het beschermen van email systemen tegen spam-verzenders. Op hun site werd naast algemene spam-informatie en de mogelijkheid spammers aan te geven richtlijnen gegeven

voor het beheren van een mailinglist. Met behulp van deze richtlijnen kon ik eenvoudig achterhalen waar op gelet moet worden bij het aanmaken en beheren van een mailinglist, hoewel de applicatie de inschrijving en beheer niet afhandelde was het toch belangrijk om te weten. In grote lijnen werd het volgende beschreven:

- Er moet een duidelijke en eenvoudige methode zijn om uit te schrijven
- Er moeten alternatieven worden geboden voor het uitschrijven
- De aard en frequentie van de mailinglist moest duidelijk aangegeven worden
- Onbestelbare adressen moeten uit de mailinglist worden gehaald
- Bescherm de mailinglist tegen misbruik

Nadat ik meer te weten was gekomen over het begrip spam ging ik nadenken over tegenmaatregelen om de applicatie minder spam-gevoelig te maken. Mijn gedachten gingen voornamelijk uit naar de applicatie minder spam-gevoelig te maken door mailings te controleren aan de hand van in de praktijk gebruikte filtermethoden en door enkel bewuste abonnees in de mailinglist op te nemen.

Door een email voor een mailing de controleren met behulp van de spamfilters zou een gebruiker zien of deze door de filters heen konden komen en waar de foutgevoelige constructies zaten wanneer dit niet het geval was. Ik zou één van twee dingen kunnen doen:

1. Bestaande software invoeren in de applicatie en die vervolgens gebruiken
2. Een eigen controle maken met behulp van woordenlijsten en eventuele andere filtermethoden.

De eerste mogelijkheid was om een bestaande filter te gebruiken om de email te controleren. De meeste programma's hebben de spamfilters ingebouwd samen met andere filterfuncties in een controle om te bepalen waar de email geplaatst moet worden door de mailserver. Aan de hand van controles krijgt de email een score waardoor deze wel of niet binnen de categorie spam valt, naderhand wordt de email dan in de geschikte map geplaatst.
De bekende filterprogramma's gebruiken bij de controle een Bayesian filtermethode, de gebruiker geeft bij deze vorm van filteren aan welke email als spam wordt ervaren waarna het programma op basis van de verhoudingen tussen de aangegeven spam - en 'normale' woorden besluit of een email nu spam is of niet. Deze methode zal in het begin spam doorlaten en gaandeweg leren om de spam te weren, voor een controlemiddel bij de applicatie is dit echter niet geschikt omdat er geen invoer van de ontvangers beschikbaar is.

Gezien ik geen geschikte en betrouwbare losse spam-controlesoftware heb kunnen vinden zou ik dus een bestaand programma voor mailservers kunnen gebruiken maar ik was niet zeker hoe lastig het zou zijn om de filterprogramma's op deze manier de email binnen de applicatie te laten controleren.

De andere optie was het aanmaken van een woordenlijst van woorden die als spam-gevoelig zijn bestempeld door de internetgemeenschap (bv. viagra) samen met de varianten (bv. VIAGRA of V14GR4) en die gebruiken om te filteren. Hoewel de manier minder effectief zou zijn kon deze aanpak wel eenvoudig te implementeren en te beheren zijn.

Buiten het naslagwerk van de term SPAM en het verzinnen van tegenmaatregelen heb ik met de opdrachtgever gesproken over het onderwerp. Buiten het doornemen van spam binnen de applicatie wilde ik nagaan of het verwerven van de abonnees geldig verliep. De opdrachtgever liet me zien hoe de abonnees doorgaans worden ingeschreven en toonde aan dat een bezoeker zich bewust moest inschrijven voor de mailinglist door middel van een webformulier. Dit formulier werd ook gebruikt om een abonnee uit te schrijven. De data van de abonnees werd opgeslagen op de webserver in een database.
Omdat het risico dat een klant SPAM zou verzenden door de bewuste inschrijving en het ten alle tijden kunnen uitschrijven was weggenomen hoefde er alleen rekening gehouden worden met mailings die de indruk van SPAM zouden wekken en dientengevolge geblokkeerd konden worden. De opdrachtgever gaf aan dat alle anti-spam oplossingen konden opgenomen worden als luxe-eis omdat hij deze functies als extra's zag en de implementatie veel tijd zou innemen. Ik heb het naslagwerk bewaard en zou doorgaan met de anti-spam extra's wanneer de luxe-eisen aan de beurt waren, tot die tijd zou ik me richten op het realiseren van de basis- en comforteisen.

2.6 Softwarekeuze: Visual Web Developer 2005

Voor het maken van de mailingapplicatie heb ik gekozen voor Visual Web Developer 2005. Dit programma is een IDE (Integrated Development Environment) voor ASP.NET websites, het programma ontleend haar functionaliteit aan Visual Basic 2005 die met haar functionaliteit de gehele ASP.NET omgeving omvat.
Microsoft's IDE's zijn bij uitstek geschikt om in een door Microsoft gemaakte ontwikkel - omgeving mee te kunnen werken. Het programma was beschikbaar binnen het bedrijf, dus er zou geen sprake zijn nieuwe aanschaffingen en binnen het bedrijf kon er geholpen worden bij gebruik. Er waren een aantal alternatieven voor het programma: Visual Basic 2005 en ASP.NET Web matrix.

Hoewel ik deze waarschijnlijk niet zou gebruiken heb ik Visual Basic 2005 geïnstalleerd om te kijken wat het te bieden had. Toen bleek dat het programma voor het maken van websites hetzelfde aanbood als Visual Web Developer heb ik definitief voor de laatste gekozen. Het Visual Basic 2005 pakket bied veel meer maar leidt daarom ook af, een bijkomend voordeel was dat Visual Web Developer 2005 gratis te downloaden was waar Visual Basic alleen als pakket te krijgen was.

Een van de weinige andere IDE's voor ASP.NET op het moment is ASP.NET web matrix, een door de ASP.NET - gemeenschap gemaakte WYSIWYG (what you see is what you get)-IDE die in qua interface veel overeenkomt met Visual Web Developer.
Het programma ondersteund de programmeertalen VB.NET, C#.NET(C sharp) en J#.NET (java sharp) maar geen C++ of Javascript, het werkt niet met projecten (alle bestanden binnen een website) maar met losse bestanden en het heeft geen IntelliSense zoals de IDE's

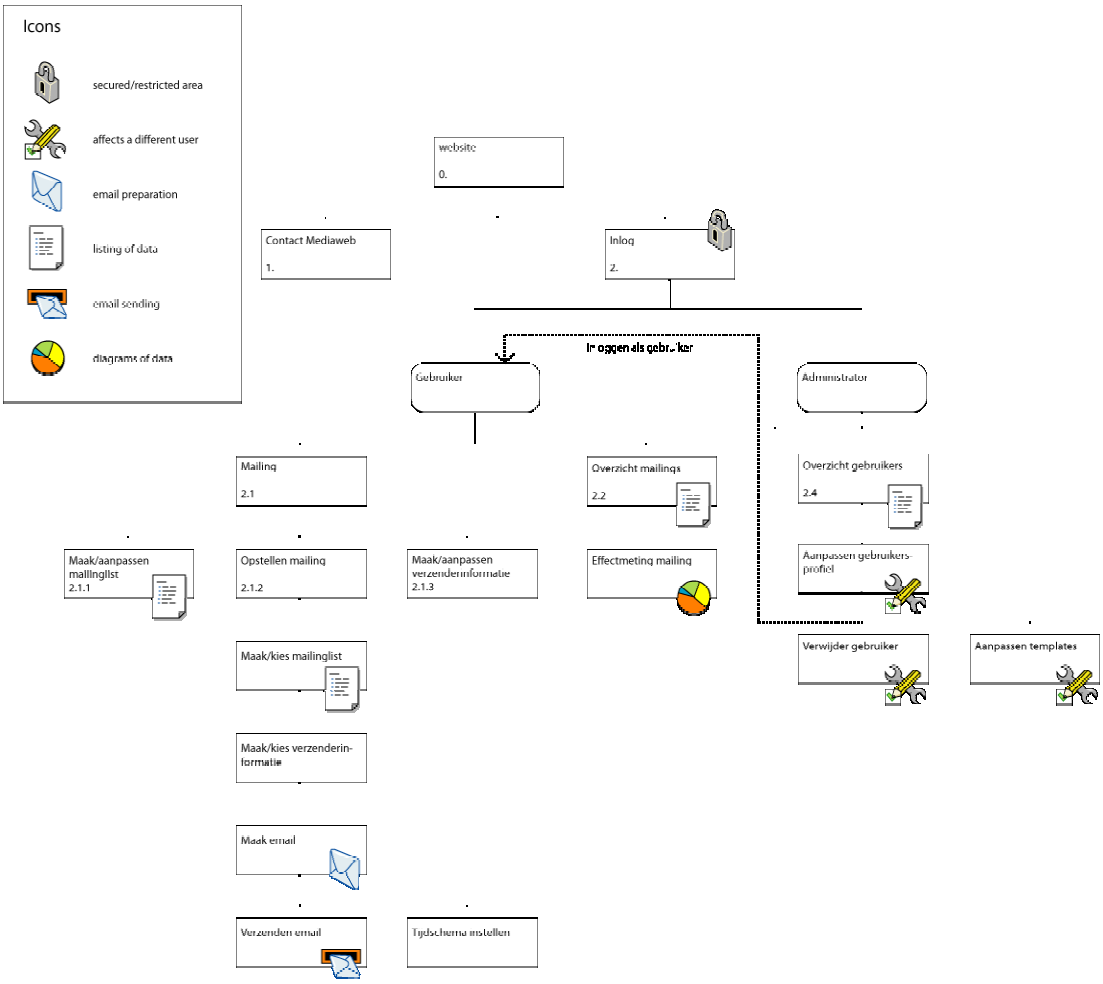
van Microsoft. Met deze laatste functie kan bij een functie of een object de bijbehorende code worden aangeboden in een lijst en elk item in de lijst is voorzien van een korte beschrijving van wat deze omvat.
In mijn thuisomgeving en voor projecten voor mezelf zou ik ASP.NET webmatrix gebruikt hebben, voornamelijk omdat het programma gratis is, maar omdat dit programma enkele belangrijke functionaliteiten mistte, nog in ontwikkeling is en
niet op werk gebruikt werd door mijn collega's heb ik het niet gebruikt.

Bij zowel Visual Basic als Visual Web Developer werd SQL server 2005 geleverd, ik had begrepen dat het bedrijf dezelfde serversoftware gebruikte dus was er geen reden om een ander te zoeken. De mailingapplicatie moest wel te gebruiken zijn in combinatie met de server en de daarop draaiende databases.

Hoofdstuk 3. Fase 2, Maken functioneel ontwerp

Ten tijden van het maken van het voorlopige functionele ontwerp ben ik met toestemming gaan kijken naar functionele ontwerpen van oude projecten. Als ik het 8 fasen - model zou willen gebruiken moesten de documentatie ook overeenkomen, zolang het niet in strijd zou zijn met de doelen van het project.
De functionele ontwerpen verschilden per project maar er waren twee terugkerende gedeeltes te onderscheiden: een navigatieschema van de website en een beschrijving van de verschillende onderdelen van de website. Voor een klant zou dit inderdaad voldoende zijn, hetzelfde geld voor een ervaren websitebouwer wanneer de website niet te complex in elkaar zou zitten.

Het voorlopige functionele ontwerp werd ook niet meer als een korte beschrijving per onderdeel en een navigatieschema wat voldoende zou zijn voor het uitlokken van feedback, waarvoor ik het zou maken. De definitieve versie moest meer bevatten als het voor het bouwen gebruikt zou gaan worden.
Omdat de mailingapplicatie veel nieuwe en ingewikkelde componenten zou bevatten besloot ik al gauw dat dit niet voldoende zou zijn om als basis te dienen om de applicatie te bouwen. Ik mistte gewoonweg de ervaring en de expertise zou ik gaandeweg vergaren. Om de applicatie te kunnen bouwen wilde ik meer informatie in het document plaatsen, genoeg om de gewenste functionaliteit van de applicatie weer te geven, en op een dusdanige wijze dat die informatie overzichtelijk was en snel opgenomen kon worden. De vereiste concentratie zou bij het bouwen anders teveel bij het opnemen van de informatie liggen in plaats van het implementeren ervan.



Figuur 2.2 Navigatieschema applicatie

3.1 Ontwerp beslissingen

Aan de hand van de bespreking van het voorlopig functioneel ontwerp zijn er bij het maken van het ontwerp een aantal beslissingen genomen over de werking van de mailingapplicatie. Na overleg met de opdrachtgever zijn deze vastgezet in het ontwerp, in de volgende paragraven worden ze kort behandeld.

Het maken van een mailing zou één doorlopend proces worden met een mogelijke onderbreking bij het selecteren of aanmaken van de mailinglist of afzendergegevens. Dit om het maken van een mailing overzichtelijk te houden door er doorloopbare stappen van te maken. Verder kon de foutgevoeligheid op deze manier beperkt worden door de gebruiker door de stappen heen te leiden.

De invoer van de gebruiker wordt tot een minimum beperkt, de meeste data wordt selecteerbaar aangeboden om zo fouten door verkeerde of schadelijke invoer te weren. In ieder geval een deel van de gebruikers zal weinig tot geen ervaring hebben met het maken en verzenden van een mailing. Door veel keuzes op voorhand beschikbaar te maken moest het proces eenvoudiger te gebruiken zijn, ook voor meer ervaren gebruikers.

Tijdens het doornemen van een verbeterde functioneel ontwerp met een collega kwam hij met de vraag of er per klant één gebruiker zou zijn of meerdere en in het geval van meerdere gebruikers of deze beperkte toegang zouden kunnen krijgen. In mijn eerdere

gesprekken met de opdrachtgever had ik begrepen dat er per klant een account zou zijn en daarom geen beperkingen hoefde te krijgen buiten het onderscheid tussen gebruiker of beheerder maar er zouden voordelen aan zitten om meerdere gebruikersgroepen in het ontwerp toe te voegen wanneer er meerdere accounts per klant zouden komen of wanneer de applicatie in de toekomst uitgebreid zou worden. Met dit in gedachte heb ik het onderwerp bij een volgend gesprek met de opdrachtgever naar voren gebracht om zijn gedachten te horen.
De opdrachtgever heeft me toen duidelijk gemaakt dat de applicatie eventueel als module gebruikt zou worden en dat een klant maar één account bezit. Er konden op een later tijdstip uitbreidingen komen op de applicatie maar daarvoor was het niet nodig om onderscheid te maken tussen de rechten per gebruiker.
Uiteindelijk zijn er dus voor het gebruik van de applicatie twee niveaus vastgesteld: de gebruiker en de beheerder. Per klant is er één gebruiker, deze hebben de mogelijkheid om een mailinglist samen te stellen, alvast afzendergegevens te maken of een gehele mailing verzorgen. Enkel personeel van Mediaweb kan als beheerder inloggen in de mailingapplicatie. Een beheerder zal naast het beheren van de gebruikersprofielen de mogelijkheid krijgen als een andere gebruiker in te loggen en voor deze gebruiker een mailing te verzorgen.

Omdat een deel van de comforteisen zeker ingevoerd zouden worden zijn in het functioneel ontwerp zowel de basis als de comforteisen verwerkt, omdat niet zeker was of de luxe-eisen aan bod zouden komen zijn deze voorlopig buiten beschouwingen gehouden. Wanneer de basis- en comforteisen geïmplementeerd waren zouden de luxe in een volgende iteratie toegevoegd worden, na aanvullende stukken voor het functioneel en eventueel grafisch te hebben gemaakt zouden deze eisen zijn gerealiseerd.

3.2 Het functioneel ontwerp verder uitbreiden

Ik heb een tijd gespeeld met de gedachte om voor het functioneel ontwerp een aanvullend document te maken dat meer uitgebreid zou zijn en het liefst de gehele applicatie in detail zou omvatten. Deze wilde ik gebruiken voor het bouwen van de applicatie maar heb uiteindelijk gekozen om het bestaande functionele ontwerp uit te breiden.
De combinatie van documenten die in de ontwerpfases worden geschreven omvatten de werking, het uiterlijk en de eisen van de opdrachtgever en wanneer deze zorgvuldig gemaakt worden zijn ze voldoende om de eindproducten te vervaardigen. Als ik met behulp van een normaal functioneel ontwerp de applicatie niet kon maken dan moest ik deze aanpassen zodat dit wel mogelijk was.

Verder heb ik na een aantal keren overwegen gekozen om de applicatie niet in alle detail weer te geven omdat dit het ontwerp geen goed zou doen. Ik miste gewoonweg de expertise van de ontwikkelomgeving en hoewel ik zekerheid betreffende het ontwerp wilde hebben zou het een slecht idee zijn geweest om het functioneel ontwerp aan te vullen met bijvoorbeeld een XUAN-model. (In dit model wordt per venster en per object op interactie- en programmaniveau beschreven wat er kan gebeuren.)
De gewenste mate van detail kon namelijk niet reëel verwerkt worden zonder dat er fouten in het ontwerp zouden sluipen door de gebrekkige kennis. Uitgebreid uitzoeken zou niet de gewenste resultaten leveren omdat het niet zeker was dat ik de gevonden informatie correct zou interpreteren, daarvoor mistte ik de ervaring en kon ik mezelf klemzetten met het ontwerp.

Daarom heb ik besloten dat het uitzoeken en direct implementeren van de functionaliteit per gedeelte beter zou werken als het uitwerken van een ontwerp waar gaten in zaten, op deze manier kon ik de problemen direct verhelpen wanneer ze zich voordeden.
Om dit te kunnen doen moest in het ontwerp de applicatie en haar werking worden weergegeven buiten de implementatieomgeving om. Het ontwerp moest gedetailleerd genoeg zijn om de werking weer te geven maar zonder de nadruk van de praktische uitvoering ervan om deze reden heb ik gekozen om de applicatie te beschrijven a.d.h.v. de Unified Modeling Language (UML), meer daarover in het volgende gedeelte.

3.3 Toevoegen van UML

Om het functioneel ontwerp aan te vullen tot een volwaardig document had ik besloten om een aantal UML-diagrammen toe te voegen. In het verleden heb ik het vaak gebruikt om bestaande applicaties en systemen te beschrijven maar het kan net zo eenvoudig voor ideeën gebruikt worden. Een bijkomend voordeel was dat met deze techniek een diagram op een later tijdstip eenvoudig meer invulling gegeven kan worden door het overzichtelijke karakter van de diagrammen en de mogelijkheid om ze zonder veel moeite uit te breiden.

Van de Unified Modeling Language heb ik het klassendiagram, use case diagram en toestandsdiagram gekozen om de applicatie weer te geven. Op deze manier kon ik de belangrijke aspecten van de te maken applicatie beschrijven. Het klassendiagram zou alle betrokken objecten en onderlinge relaties beschrijven, het use case diagram zou de interactie tussen het systeem en haar omgeving beschrijven en het toestanddiagram zou de werking van de applicatie beschrijven. Een deel van de UML – diagrammen zijn opgenomen in de hoofdstuk 10 als referentie.

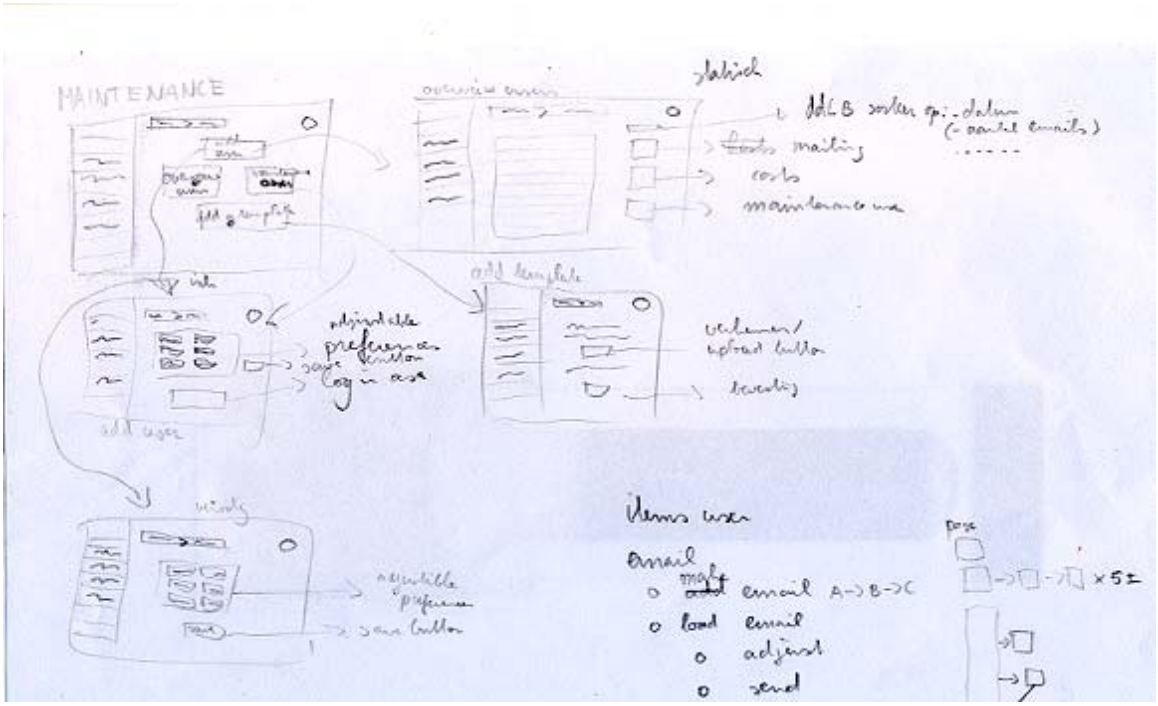
In de het door de diagrammen gemaakte model zijn enkel de basissysteemeisen verwerkt. Van de luxe-eisen was in de vorige fase al besloten dat deze in een volgende iteratie verwerkt zouden worden, de comforteisen konden nog niet ingevoerd worden omdat de betrokken gegevens niet beschikbaar waren of de beslissingen niet waren gemaakt. Het businessmodel was namelijk nog niet gemaakt en zou pas op een later tijdstip worden behandeld wanneer de applicatie een tastbare vorm had aangenomen. Bij het gebruik van de abonneedatabases waren enkel leesrechten toegekend dus mochten er geen aanpassingen in aangebracht worden.

De UML – diagrammen waren bedoeld om de processen te kunnen bouwen en de processen vervolgens uit te breiden tot een werkend geheel. Door het gebrek aan ervaring en kennis verwachtte ik dat er gaten in het ontwerp zouden zitten zelfs als ik mijn best deed om deze in detail uit te werken, dit zou vooral het geval zijn bij het klassendiagram en de toestandsdiagrammen. Tijdens het bouwen van de applicatie zou ik deze gaten en onregelmatigheden tegenkomen en ter plekke oplossen. Wanneer de applicatie afgerond zou zijn kon ik als dat gewenst werd een definitieve versie maken die gedetailleerd de werking van de applicatie weer zou geven.

Niet alle medewerkers waren bekend met deze techniek en de opdrachtgever was er één van, hij kon wel globaal achterhalen wat er in de diagrammen stond maar de fijnere nuances waren lastig te bevatten. Om deze reden konden het navigatieschema en de beschrijvingen niet achterwege gelaten worden, ze waren nodig om het gat te vullen. Het document moest per slot van rekening voor iedereen toegankelijk zijn. Ik heb na een ronde feedback de twee originele delen herschreven zodat alle onderdelen van het functionele ontwerp beter op elkaar aansloten.

3.4 Tussentijdse schetsen

Net als bij het opzetten van de wensen en eisenlijst heb ik een klein begin gemaakt met de volgende fase, deze keer niet zozeer om feedback te ontvangen maar om te kijken hoe het functioneel ontwerp vorm zou krijgen en om te controleren of ik zaken over het hoofd had gezien.
Hoewel de schetsen zelf weinig om handen hadden heeft het maken van schetsen aan de hand van het gemaakte functionele ontwerp geholpen om een tastbaar beeld te schetsen van de applicatie en om de stappen van het mailingproces te structureren.
Tot ik verder ben gegaan met het grafisch model is er niets gedaan met het uitwerken van deze schetsen, in de tussentijd heb ik wel de schetsen gebruikt bij het oefenen met ASP.NET en toen er aan het grafisch ontwerp gewerkt kon ik met die ervaringen het ontwerp opstellen.



Figuur 3.1 navigatieschets gebruikersbeheer

3.5 Controle van het ontwerp

Omdat het 8 fasen – model nieuw voor mij was wilde ik nagaan of de ontwikkelmethode inderdaad voldoende was voor dit project. Bij elke fase ben ik nagegaan of het product van de fase voldoende was voor het doel waarmee die fase doorlopen werd. Bij de eerste fase is een aantal keren overwogen om de wensen en eisen – lijst anders in te delen en één keer is dat ook gedaan aan de hand van de prioriteit.
Al bij het naslaan van de bestaande functionele ontwerpen vond ik dat deze voor dit project niet voldoende zouden zijn als ik een soortgelijk ontwerp zou maken. Het ontwerp was niet voldoende om de applicatie mee te bouwen. Na geruime tijd erover nagedacht te hebben en wat proefschetsen te hebben gemaakt heb ik het functioneel uitgebreid met een model en een globale beschrijving van de verschillende gebruikersgroepen.

Nadat ik het functioneel ontwerp had voltooid heb ik deze een aantal keer nagekeken, zelfstandig en later met een collega. Er waren een paar kleine aanpassingen nodig om het compleet te maken. Bij de terugkoppelmomenten van het grafisch ontwerp en het interactief ontwerp konden wanneer er afwijkingen of verbeteringen waren bepaald deze doorgevoerd worden in het originele functionele ontwerp. Ik beschouwde het functioneel ontwerp als voltooid en het is in een bespreking met de opdrachtgever goedgekeurd.

3.6 Module keuzes voor de applicatie

Op dit punt was het nuttig om te gaan kijken wat ik allemaal nodig zou kunnen hebben buiten de ontwikkelomgeving en de editor voor het schrijven van de applicatie. Mijn intentie was om zoveel mogelijk zelf te maken zonder terug te vallen op modules, hoewel ze voordelen met zich meebrengen kosten ze geld, leer je er weinig van en vereist het wat pas en meetwerk om het in de applicatie toe te voegen.
Daar staat tegenover dat modules kant en klaar zijn, doorgaans minder fouten als zelfgeschreven code bevatten en vaak een compleet kennisgebied omvatten die de programmeur niet eigen hoeft te maken (overigens is wel aan te raden om dit alsnog te doen al is het oppervlakkig).
Dus voor belangrijke, ingewikkelde of veel tijd kostende onderdelen binnen de applicatie zou ik het gebruik van modules overwegen. Hierbij zal ik eerst kijken wat Mediaweb al bezit alvorens naar andere modules te kijken.

Op dit punt waren er 2 type modules die ik overwoog te gebruiken, ASP.NET voorzag de codeur in voldoende mogelijkheden met betrekking tot beveiliging, gebruikersprofielen en verbindingen met databronnen maar had een vrij kale email en tekstverzorging.

Een web tekst - editor voor de tekst van de email zou misschien handig zijn maar niet noodzakelijk omdat met een beetje inspanning er zelf één gebouwd kon worden, het is per slot van rekening veelal een kwestie van het voorzien van een tekst van toepasselijke HTML-tags. Deze kwestie zou pas relevant worden wanneer er daadwerkelijk emails gemaakt konden worden, tot die tijd heb ik enkel rekening gehouden met de mogelijkheid dat er een texteditor gebruikt kan worden.

Een mail module zou kunnen toevoegen aan het mailingproces afhankelijk van welke toegevoegde functionaliteit het bood. Een email verzenden kon zonder een module maar dat zou gebruik maken van CDONTS, die gevoelig was voor fouten en veeleisend was voor de performance van de server. Nu zouden deze beperkingen met onderzoek en inspanning wel op te vangen zijn maar het was niet zeker of daar de tijd voor beschikbaar zou zijn en of het zou lukken. Om die redenen was het al nuttig om te kijken wat voor modules er aangeboden werden.

AspEmail was één van de mailing modules op de markt, het is gratis te downloaden maar door een licentie te kopen worden er meer functies vrijgegeven die toegevoegde waarde hebben zoals een alternatieve tekst wanneer de ontvanger geen HTML in de emailberichten ondersteund of het aanbrengen van afbeeldingen in het tekstgedeelte van de email. Mediaweb had reeds de licentie gekocht van AspEmail en maakte geruime tijd gebruik van de module, dus er zouden geen aanschafkosten zijn mocht ik voor deze module kiezen.

De module bied onder andere de volgende mogelijkheden:

- De mogelijkheid email via een externe SMTP server te versturen via de codeerwerk in plaats van verzenden
- De module staat meerdere ontvangers, CC's, BCC's, Reply-To's en file attachments toe
- Ondersteund het verzenden van emails in HTML en tekst format evenals embedded afbeeldingen en geluid
- Verschillende SMTP authenticatie methoden
- Grote volumes email in de wachtrij laten staan en er kan voorrang bij het verzenden worden verleend.
- Een mislukte verzending kan meerdere keren opnieuw geprobeerd verzonden te worden.
- Afhandeling van bounced (niet verzendbare of geweigerde) email

AspNetEmail was een andere die module hoog in aanzien stond. Het is alleen als commercieel product te krijgen en is de afgelopen 3 jaar bestempeld als de lezerskeuze voor mailing modules door asp.netPRO, een webblad voor professionele asp.net gebruikers waar onder andere artikelen over modules en boeken in staan.
De prijs voor de module wisselt tussen de $128,- en de $599,- afhankelijk van welke licentie aangeschaft zou worden.

Naast grofweg hetzelfde te kunnen als AspEmail maar dan in meer detail bied AspNetEmail ook andere mogelijkheden zoals het gebruik van mail templates, het invoegen van ASP.NET objecten zoals datagrids en het gebruiken van informatie uit datareaders en datasets in de email.

Afhankelijk van wat er ten tijde van het bouwen van het "email verzenden"-bouwdeel nodig blijkt te zijn en wat er in de toekomst van de mailingapplicatie nodig is zal ik een keuze maken tussen (tijdelijk) geen module, één van deze twee modules of zoeken naar beter oplossing.

Hoofdstuk 4. Fase 3, Maken grafisch ontwerp

Bij het teruglezen van de structuur van een grafisch ontwerp heb ik besloten dat net als bij het functioneel ontwerp het grafisch ontwerp aangevuld moest worden wilde ik een geschikte applicatie maken. Een aantal schetsen van de kernpagina's voldeed namelijk niet wanneer niet bepaald werd hoe de vormgeving door de hele applicatie doorgevoerd zou worden. Het grafisch ontwerp zou een (soort van) huisstijl worden van de applicatie waarin naast een aantal uitgewerkte schetsen informatie over de vormgeving en de gebruikersinterface te vinden zou zijn.

Voor het maken van de lay-out was ik vrijgelaten door de opdrachtgever, er was gezegd dat later het uiterlijk wel rechtgetrokken kon worden. Ik was van mening dat het net zo goed in één keer goed gedaan kon worden, hoewel er geen eisen aan het uiterlijk waren gesteld wilde ik een uiterlijk maken waarin de opdrachtgever en ik ons beiden konden vinden. Omdat de opdrachtgever eerder producten voor klanten had gemaakt en wist wat deze mensen wilden, was ik van plan zijn feedback te gebruiken om een op zijn minst acceptabel uiterlijk voor de mailingapplicatie te maken. De opzet was om tijdens het uitdenken van het grafische ontwerp met behulp van het bespreken van voorbeelden met de opdrachtgever meer over de gebruikersinterface te weten te komen en zodra er een ontwerp voltooid was deze door te nemen.

4.1 Voorbereidend werk voor het ontwerp

Om een ontwerp te kunnen maken ben ik in eerste instantie op het internet gaan kijken om inspiratie op te doen en richtlijnen te formuleren voor het ontwerp. Voor de richtlijnen heb ik artikelen gezocht over webdesign en bij de voorbeelden heb ik de goede en slechte eigenschappen genoteerd. Ik heb weinig resultaten geboekt bij het zoeken naar pagina's over webdesign en heb daarna besloten voornamelijk de regels van Schneiderman toe te passen (*meer daarover in 4.5*).

Voordat ik begon met schetsen heb ik mezelf een aantal vragen gesteld over hoe ik de applicatie zag en welke invloed dat zou hebben op de vormgeving. Omdat de applicatie voor een langere tijd gebruikt zou kunnen worden waarbij er geconcentreerd te werk wordt gegaan moest deze niet te druk zijn. De uitstraling van de applicatie moest hetzelfde als het bedrijf zijn dus professioneel en toch een tikkeltje eigenzinnig. Het invoegen van iconen wilde ik vanaf zien, geschikte binnen de stijl van de applicatie passende iconen ontwerpen vergt tijd en voor consistentie zouden er snel een tiental gemaakt moeten worden.

Met mijn voorbereiding afgerond ben ik schetsen gaan maken en bij het volgende gesprek met de opdrachtgever heb ik twee digitale schetsen uitgewerkt die qua uiterlijk ver uit elkaar lagen, door twee van elkaar afwijkende schetsen aan te leveren had de opdrachtgever meer keuzemogelijkheden om zijn wensen voor het uiterlijk van de website op te formuleren. Een tweede overweging was dat mensen vaak niet precies weten wat ze wel willen maar wel wat ze niet willen, de twee afwijkende proefschetsen waren dus ook bedoeld om van de opdrachtgever te horen wat hij niet wilde waarna het eenvoudiger zou zijn om het te hebben over wat hij wel wil. Bij meer als twee schetsen dacht ik dat het effect verzwakt zou worden doordat de aandacht dan teveel verwijderd zou worden van wat de opdrachtgever in gedachte had. De schetsen waren niet meer als gesprekspunten om voldoende informatie te krijgen en zo echte schetsen te kunnen maken.

4.2 Terugkoppeling met de opdrachtgever

Tijdens de bespreking van de gemaakte proefschetsen bleek dat de mailingapplicatie eventueel als onderdeel van een door Mediaweb gemaakt Content Management System gebruikt zou worden. Het CMS had al een huisstijl en geopperd werd dat deze gebruikt kon worden. De opdrachtgever was namelijk meer geïnteresseerd in de invulling van de functionaliteit en van de gebruikersinterface als het uiterlijk ervan.
Een grafisch ontwerp is maar gedeeltelijk geschikt om de functionaliteit te tonen, het kan enkel een statische weergave van een dynamisch systeem geven. Gezien ik op dit punt in het project achter op mijn planning liep wilde ik de huisstijl van het CMS of een aangepaste versie overnemen om de eventuele invoering te vereenvoudigen en daarna doorgaan met een Interactief Ontwerp maken. Een Interactief Ontwerp kon nu eenmaal beter als een grafisch ontwerp de werking van een applicatie weergeven.
Door de reeds goedgekeurde huisstijl over te nemen kon het Grafisch Ontwerp sneller afgerond worden en kon ik het interactief ontwerp gaan maken waar de opdrachtgever geïnteresseerd in was.

4.3 Inbreng van een Content Management System

Tijdens de bespreking had ik geïnformeerd of er van de huisstijl van het CMS een document was opgesteld maar dit bleek niet het geval. Ik besloot daarom om met behulp van het bestaande Content Management System de huisstijl vast te stellen.
Na een controle of de huisstijl van het CMS (met namen de interface) wel geschikt was voor de mailingapplicatie ben ik aan de hand van het bestaande product een huisstijl gaan formuleren op basis van de website en de Cascading Stylesheet van de website.

Omdat er geen eerdere documentatie was gemaakt van de huisstijl was ik genoodzaakt aan de hand van het eindproduct deze te achterhalen. Het documenteren van de specificaties van een eindproduct door de code ervan en de werking uit te pluizen heet reverse engineering en dat heb ik in dit geval toegepast met als doel een gedocumenteerde versie van de huisstijl van het CMS.
Dit was de eerste keer dat ik te maken had met een vorm van reverse enginering en ik moet toegeven dat het inderdaad lastiger is om op deze manier een document op te stellen dan het zelf opstellen in de ontwerpfase van een product.

In het grafisch ontwerp zijn de huisstijl van het CMS, gedigitaliseerde schetsen van de mailingapplicatie met uitleg en de informatie om deze twee aan elkaar te koppelen geplaatst. Ik heb bewust deze tweedeling gemaakt zodat het document ook als gedocumenteerde huisstijl voor het CMS kon dienen.

4.4 Bedrijfsbezoek van de Haagse Hogeschool

In de 5^{e} week van de afstudeerstage is door de examinatoren contact opgenomen om afspraken te maken voor het bedrijfsbezoek, wat naderhand schriftelijke is bevestigd.
In de 6^{e} week heeft het bedrijfsbezoek plaatsgevonden waarin de student is geïnformeerd over de gang van zaken en de examinatoren op hun beurt een indruk konden krijgen over het verloop van de stage en van het stagebedrijf zelf. Tijdens het gesprek is er een afspraak gemaakt voor het inleveren van het voortgangsverslag een week later, hierna is er tot de

bespreking van het bouwplan geen contact meer geweest tussen de student en de examinatoren.

4.5 Schneiderman's 8 regels voor Interfase ontwerp

De 8 "gouden" regels van Schneiderman voor interface ontwerp zijn bruikbare richtlijnen om de gebruikersinterface zo aan te passen dat deze minder verkeerd geïnterpreteerd kan worden en minder inspanning vergt om te begrijpen voor de gebruiker. Ze zijn geformuleerd door de heer Schneiderman aan de hand van opgedane ervaring als een expert op het gebied UI-design (GUIDE).

Bij het nadenken over een grafisch ontwerp had ik al onbewust sporadisch de regels van Schneiderman toegepast, toen ik me hier bewust van werd heb ik besloten consistent de regels in het ontwerp door te voeren. Voor het gemak zijn hier de 8 regels:

1. Strive for consistency
2. Enable frequent users to use shortcuts
3. Offer informative feedback
4. Design dialog to yield closure
5. Offer simple error handling
6. Permit easy reversal of actions
7. Support internal locus of control
8. Reduce short-term memory load

De ideeën die ik voor de applicatie had door het gebruik van deze regels zijn doorgevoerd in het grafisch waar toepasselijk, de overige zijn opgenomen in de aantekeningen en ingevoerd tijdens de bouwfase. Voorbeelden van de toegepaste regels zijn hieronder te vinden.

1. Strive for consistency.
- Plaatsing navigatie-objecten op dezelfde locatie
- Consistent gebruik iconen conform het CMS
- Consistente vormgeving (kleurgebruik, tekst)
- Consistente weergave data, bv. in een gridview
- Consistent taalgebruik

2. Enable frequent users to use shortcuts

- Het apart beheren van mailinglists en afzendergegevens om toekomstige mailing te versnellen
- Het kunnen selecteren van voorgemaakte mailinglists en afzendergegevens tijdens het verzorgen van een mailing

3. Offer informative feedback.

- Bij elke handeling van de gebruiker een reactie van de pagina, bv. het laden van
- gegevens of een andere pagina en het opkomen van een pop-up venster
- Wanneer de juiste handelingen zijn uitgevoerd kan de gebruiker navigeren naar de volgende stap in het proces
- De objecten van een pagina krijgen waar hun doel onduidelijk kan zijn een tekstveld met meer informatie die getoond word wanneer de cursor erover hangt

4. Design dialogues to yield closure

- Bij het maken van een mailing zijn de stappen genummerd en word er aangegeven waar binnen het proces de gebruiker zit (bv. Stap 3/11 mailinglist selecteren)
- Objecten op pagina's worden beschikbaar gesteld afhankelijk van de handelingen van de gebruiker of deze aan de voorwaarde voldoen
- Foute invoer of het niet invullen van vereiste velden stopt de handeling en geeft foutmeldingen

5. Offer simple error handling.

- Bij het verkeerd of niet invoeren van velden krijgt de gebruiker indicatieve foutmeldingen
- Wanneer gegevens niet kunnen worden opgeslagen wordt aangegeven waarom dit niet kan.

6. Permit easy reversal of actions.

- De gebruiker kan door het mailingproces heen navigeren en aanpassingen aanbrengen
- Bij het opslaan van gegevens wordt de bevestiging van de gebruiker gevraagd alvorens op te slaan
- De gebruiker kan apart de opgeslagen mailinglists en afzendergegevens beheren

7. Support internal locus of control.

- Bij handelingen met permanente gevolgen zoals het opslaan, aanpassen of verwijderen van gegevens of het verzenden van een mailing wordt de toestemming van de gebruiker gevraagd alvorens de handeling uit te voeren

8. Reduce short-term memory load.

- De gebruiker moet per pagina niet meer als 2 tot 3 handelingen uitvoeren voordat deze door kan gaan
- Op elke pagina wordt aangegeven wat de gebruiker dient te doen om door te kunnen gaan

Hoofdstuk 5. Fase 4, Maken interactief ontwerp

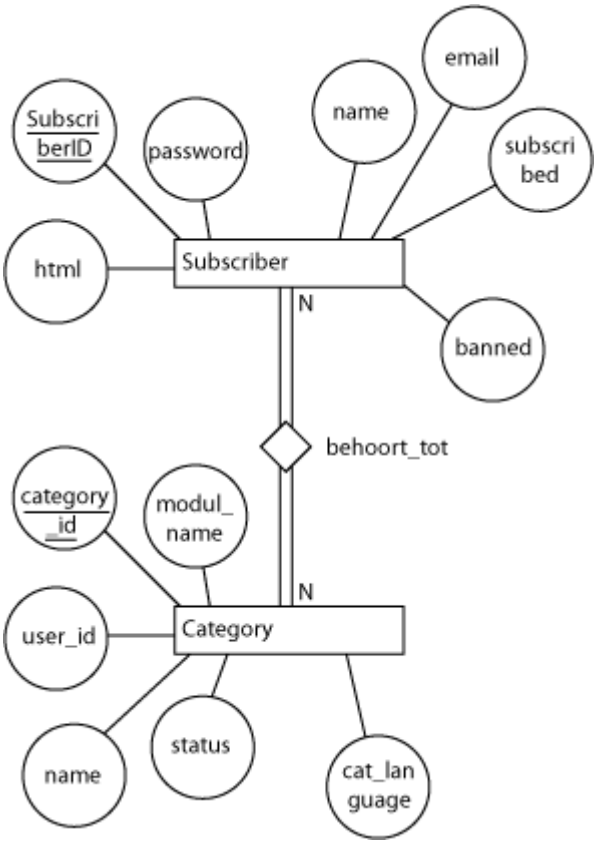
Ik heb niet lang bij het interactief ontwerp stil willen staan. Het functioneel en grafisch ontwerp hadden het beeld van de applicatie geschetst en een (uitgebreid) interactief ontwerp zou weinig hier aan toevoegen als je keek naar de opdrachtgever. Hij had ervaring met de ontwikkelmethode en met het bouwen van websites dus een aantal HTML pagina's zouden enkel bevestigen wat hij eerder al gelezen en besproken had en ik verwachtte ook weinig feedback zonder van het interactief ontwerp een prototype te maken. De ingevulde functionaliteit zou in tegenstelling tot de statische pagina's wel reacties uitlokken.

Ik heb een paar pagina's gebouwd alvorens door te gaan met functies in te voeren en ik kwam al gauw tot de conclusie dat ik niet verder kon zonder een databasemodel te maken voor de applicatie. Het leek me namelijk nutteloos om een kladdatabase te maken om er daarna aanpassingen in aan te brengen en bij het ingaan van de volgende fase een database met een goed ontwerp te maken. Om deze reden ben ik begonnen met het schrijven van een databasemodel

5.1 Databasemodel maken

Ik heb de databasemodel gemaakt op basis van de UML – diagrammen uit het functioneel ontwerp en de eisen die de applicatie aan de database zou stellen
(bv. Inloginformatie, gegevens betreffende de abonneedatabase). Ik heb de externe database apart de applicatiedatabase beschreven om het databasemodel overzichtelijk te houden.
Het model bestond uit een conceptueel datamodel met meerdere Entity Relationship - diagrammen (het origineel was te groot), een relationeel representatiemodel en een implementatiemodel. Elk model binnen de databasemodel was een stap van het abstracte beeld richting de vertaling die direct omgezet kan worden naar de database.
In hoofdstuk 7 is het conceptueel databasemodel opgenomen die de database beschrijven.

Ik heb gebruik gemaakt SQL Server Enterprise Manager voor de implementatie en voor de invoer van proefgegevens. Het model is een aantal keren aangepast naarmate er meer onderdelen van de applicatie gebouwd of bestaande delen aangepast werden aan de hand van feedback van de opdrachtgever.



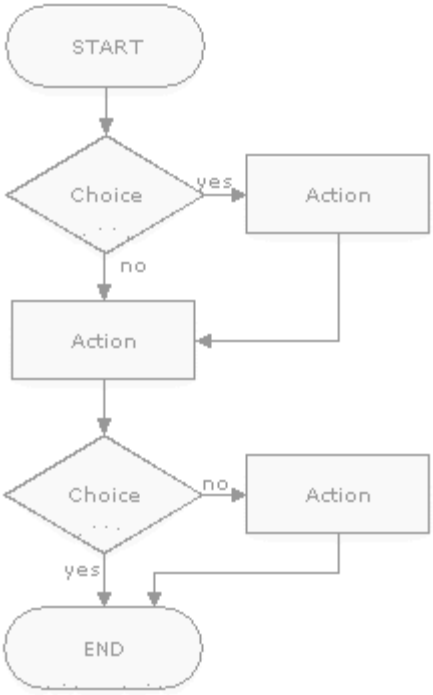
Figuur 5.1 ER-diagram Externe database

5.2 Flowcharts

Na wat aanmodderen om het functionele ontwerp in te voeren vond ik dat er een tussenvorm moest komen tussen het redelijk abstracte functionele ontwerp en de applicatie zelf om deze te kunnen realiseren. Om deze reden ben ik weer naar de tekentafel gegaan om de processen van de applicatie uit te denken op papier.

In eerste instantie kwam ik na het uiteenzetten van de stappen van de processen met een checklist waarin per applicatiepagina werd aangegeven wat er gedaan moest worden volgens het functionele ontwerp. Het idee werkte maar ik was genoodzaakt een tweede checklist in de code van de pagina toe te voegen om alle sub-functies en –acties in op te nemen.
Deze aanpak werkte dus maar ten dele, niet lang na het gebruiken van de checklist raadde een collega me aan om een flowchart te proberen. Na het gesprek ben ik op het internet gaan zoeken informatie over flowcharts. Het bleek een diagram te zijn waar met behulp van symbolen stap voor stap een proces wordt doorlopen, binnen UML komt het nog het meest overeen met een activiteitendiagram. Tijdens het zoeken naar informatie had ik een reader gevonden die het gebruik van flowcharts uitlegde.

De informatie in de reader kwam overeen met de algemene informatie die ik op het internet had gevonden en kwam ook overeen met de symbolen die in het programma Microsoft VISIO, een programma gericht op het maken van modellen, dus heb ik het gebruikt een flowchart van één van de processen te maken. Ik was tevreden met het resultaat, op één A4 kon een geheel proces op overzichtelijke worden beschreven.



Figuur 5.2 Basisprincipe van een flowchart

Doordat elk symbool een actie, een keuze of een opgeslagen waarde vertegenwoordigde kon een proces vrij snel gestructureerd worden en als controlemiddel gebruikt worden. Ik heb toen besloten om de processen binnen het functioneel ontwerp op deze manier te beschrijven en om de gemaakte checklist te vervangen met de flowcharts.

5.3 Prototype

Met het invoeren van functies in het interactief ontwerp ging deze van een paar losse HTML – pagina's naar een prototype, voornamelijk omdat ik had besloten om zoveel mogelijk van de functionaliteit in te voeren.

Ik wilde op met een werkend voorbeeld het ontwerp toetsen en de opdrachtgever de gelegenheid te geven door te denken over de applicatie die hij in gedachte had en op deze manier door zijn feedback het ontwerp afstemmen.

Met het maken van de prototype is er technisch gesproken een de start gemaakt met de bouwfase, het prototype was meer als het eindproduct van de fase interactief ontwerp en dus besloot ik dat het prototype afhankelijk van de kwaliteit de applicatie kon worden. Ik heb hierna met het tonen van de applicatie pagina's in ieder geval interactieve ontwerp afgesloten zodat ik ongestoord aan de applicatie kon bouwen.

5.4 Module keuze: CuteEditor

Bij het nagaan van de processen stond ik wederom stil bij het feit dat de mogelijkheden om de email de juiste vormgeving te verlenen karig waren. Omdat het bouwdeel waarin het aanpassen van het uiterlijk van de email zat, pas tegen het einde de bouwfase gerealiseerd zou worden was een kans aanwezig dat er weinig tijd beschikbaar voor zou zijn. Om die reden ben ik gaan kijken naar een manier om er eventueel tijd op te besparen.

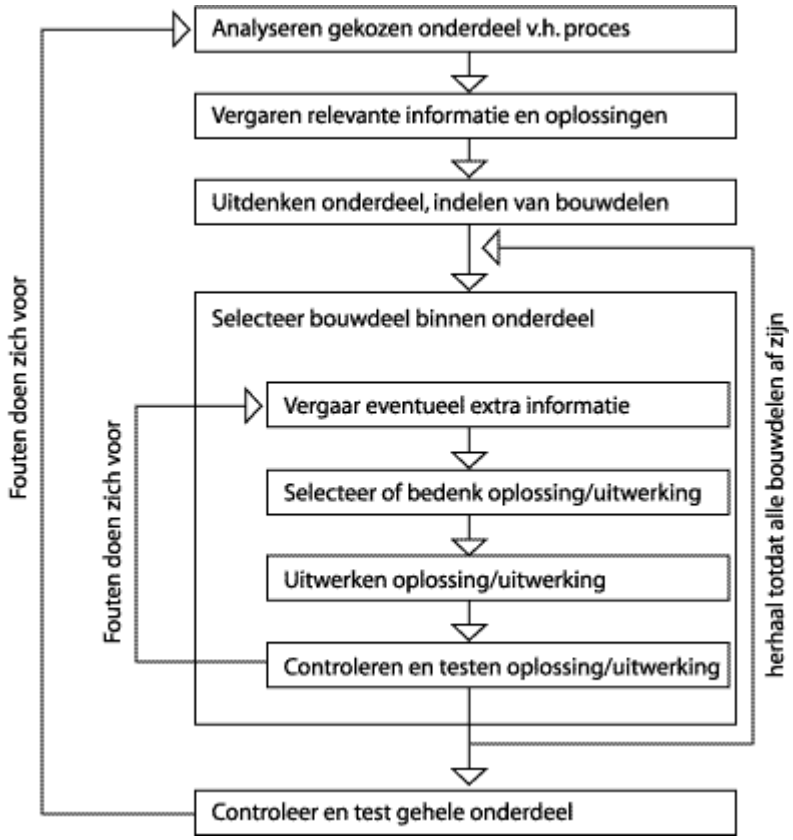
Een van de mogelijkheden voor de opmaak van de email is de module CuteEditor. Het is een web teksteditor die de ingevoerde tekst aanpast zodat deze in HTML-syntax staat. De module is aan te passen zodat de bestaande functies wel of niet beschikbaar worden gesteld bij het inladen van de module.
Naast het opmaken van de direct ingevoerde tekst en afbeeldingen kan de module Microsof Word bestanden importeren, ondersteund het relatieve en absolute URL-adressen en kan het de gemaakte tekst opslaan in een aantal type bestanden zoals pdf, word en rtf.

Mediaweb bezit de licentie van deze module en de module voegt veel toe wanneer ik het zou gebruiken binnen de applicatie, het bouwen van vergelijkbare functies zou (te) veel tijd en inspanning kosten. Gezien CuteEditor meer dan voldoende bood en reeds beschikbaar was besloot ik niet verder te zoeken naar andere web editor modules.

Hoofdstuk 6. Fase 5, Bouwen applicatie(s)

In de bouwfase werd er weinig anders gedaan dan informatie vergaren voor het realiseren van het ontwerp en het bouwen van de applicatie. Het bouwtraject werd per pagina en per onderdeel uitgevoerd en bestond uit het vergaren van relatieve informatie of oplossingen, het uitdenken en uitwerken van een werkbare oplossing en het testen van het gemaakte deel.

Op deze manier en in deze volgorde zijn de processen het mailinglistbeheer, het Afzendergegevensbeheer, het Inlogsysteem en het Gebruikersbeheer gemaakt. Vaak moesten de bouwdelen meerdere keren doorlopen worden voordat deze naar behoren werkte.



Figuur 6.1 aanpak bouwen applicatie

6.1 Behouden van informatie tussen de pagina's

Het doorgeven en behouden van de gegevens tussen de pagina's in ASP.NET moet net als bij statische HTML pagina's gebeuren ia een omweg. ASP.NET heeft een aantal manieren om data over te dragen, in de hieronder staands tabel zijn ze aangegeven.

Mogelijkheden vanuit de serverkant	**Voordelen**	**Nadelen**
Application state	Snel. Wordt gedeeld door alle gebruikers	Wordt eenmaal opgeslagen
Session state: in proces	Per gebruiker, sneller als out of proces en database-baked	Gebruiken geheugen totdat sessie eindigt
Session state: out of proces	Robuuster als in proces	Problemen met reboots van server en recycling van website
Session state: database-baked	Balans tussen snelheid en robuustheid Kan worden benadert door meerdere servers	Vereis sql database server Vereist configuratie / toevoeging van database
Session state: cookieless	Vereist geen cookies	Zeer onveilig, informatie gaat via URL
Database	Kan worden benadert door meerdere servers	Query voor elke

Mogelijkheden vanuit de client/ bezoeker kant	**Voordelen**	**Nadelen**
Cookies	Kan ge-encrypt worden	Belastend door dubbel gebruik bij elke transactie
Hidden field	Vereist geen actie van de server	Onveilig, kan in broncode pagina gelezen worden
View state	kan ge-encrypt worden	Zelfde als een hidden field
Postback/crosspostback	Wordt geactiveerd bij interactie met een object van de pagina	Kan geen data bewaren tussen verschillende postbacks
Query strings	Navigatiespecifieke data	Zeer onveilig, informatie gaat via URL

Nu heb ik voor het behouden van de gegevens verschillende methoden toegepast afhankelijk van het soort data dat betrokken was. In het kort zullen de beslissingen doorgenomen worden.

Omdat zowel mailinglistbeheer als afzendergegevensbeheer als onderdeel van het maken van een mailing of losstaand gebruikt konden worden moest aangegeven dit worden bij de selectie van het proces. Hiervoor heb ik gekozen voor het gebruik van querystrings omdat het geschikt was voor navigatie tussen pagina's hier geen gevoelige informatie betrof en afgeschermd kon worden voor verkeerde invoer. Ik heb querystrings tevens gebruikt om onderscheid aan te maken bij mailinglistbeheer en afzendergegevensbeheer om aan te geven of er een nieuw record aangemaakt werd of dat een bestaande record benaderd werd.

Bij het maken en verzenden van een mailing moesten een aantal gegevens tijdelijk opgeslagen worden aan het einde van het proces verwerkt te worden bij het verzenden van de mailing. Omdat de data één bezoek sessie mee moest gaan en bij voorkeur snel moest verwerkt worden met een klein aantal gebruikers heb ik gekozen voor het gebruiken van in proces session state om de (tijdelijke) mailinglist, de (tijdelijke) en de inhoud van de email op te slaan.

Voor data afhandeling binnen een pagina, zoals het controleren van waarden in velden, heb ik gebruik gemaakt van de postback mogelijkheid. De data wordt terug verzonden naar de server die vervolgens hiermee iets kan doen, omdat het enkel op dezelfde pagina werkt en via een omweg naar een aangegeven pagina heb ik postbacks enkel gebruikt bij verwerking van de data voordat de gebruiker naar de volgende pagina moet kunnen.

Verder waren er een aantal gegevens die beschikbaar moesten zijn zodat de gebruiker kon inloggen op het systeem en zijn/haar abonneedatabase kon gebruiken. Samen met alle andere gebruikersafhankelijke informatie die niet van tijdelijke aard waren zoals gedane mailings, mailinglists en afzendergegevens die op een later tijdstip beschikbaar moesten zijn heb ik deze ondergebracht in de database van de mailingapplicatie. Het betrof hier altijd data die meerdere sessies mee moesten gaan en veilig opgeslagen moesten worden.

6.2 Database conversie voor toevoeging functionaliteit

Binnen ASP.NET worden de componenten voor een inlogsysteem aangeboden waarmee bezoekers geverifieerd kunnen worden en de status van de gebruiker samen met voorkeursinstellingen bewaard kunnen worden. De componenten maken gebruik van de membership en profile klassen om de gebruikersgegevens te bewaren en om gebruik te kunnen maken van deze klassen moest de database aangepast worden.
Om gebruik te kunnen maken van de ingebouwde membership en profile mogelijkheden van ASP.NET zonder soortgelijke functies te bouwen moest ik of een database, lokaal of op de server, aanmaken of de bestaande database aanpassen.

Om de structuur van de mailingapplicatie overzichtelijk te houden wilde ik buiten de abonneedatabases van de gebruikers één database de gebruikersprofielen en andere data van de gebruikers zoals mailinglists laten afhandelen. Het aanpassen van de bestaande database van de mailingapplicatie hield het toevoegen van een aantal tabellen met secundaire indexen en onderlinge relaties in, de reeds bestaande data en metadata zou onaangeroerd blijven.

Wanneer de membership functionaliteit wordt gebruikt zonder zelf een database klaar te maken aan te brengen zal Visual Web Developer zelf een geschikte lokale database aanmaken. Om een bestaande database aan te kunnen passen moest een wizard gebruikt worden die standaard bij Visual Basic 2005 geleverd werd, de andere optie was om in de geïnstalleerde Framework twee SQL scripts, InstallPersonalization.sql en UnInstall - Personalization.sql, te zoeken en te draaien. Ik heb omdat Visual Basic al geïnstalleerd stond de wizard gebruikt om de aanpassingen door te voeren.
De uiteindelijke database is terug te vinden in hoofdstuk 7 en in de bijlagen.

6.3 Afronding

Normaal gezien zou het project nog de 6e fase, het testen van de eindproducten, hebben doorlopen maar gezien het eindproduct nog niet afgerond is kon deze ook nog niet getest worden. Fase 7, het opleveren en invoeren van de producten, zou informeel doorlopen worden en fase 8, de nazorg, was aan het begin van de afstudeerstage besproken hoefde niet gedaan te worden.

Deze 3 fases kunnen en zullen dus niet in de scriptie behandelt worden.

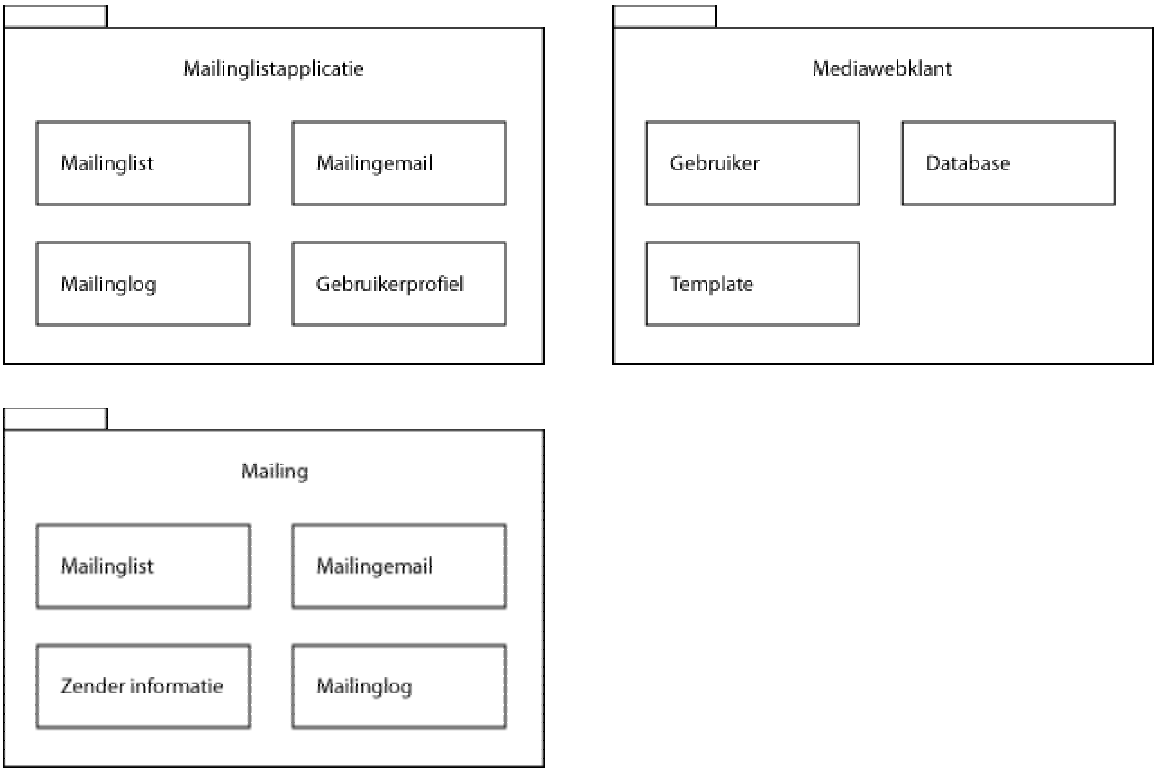
Hoofdstuk 7. Aparte stukken

In dit hoofdstuk worden kleine stukken geplaatst die om de leesbaarheid te bewaren zijn verplaatst naar dit hoofdstuk. Het gaat om stukken die (extra) informatie bevatten of gedetailleerde (technische) delen van het project van andere hoofdstukken Met een kleine verwijzing bij deze gedeeltes wordt aangegeven welk stuk in dit hoofdstuk gelezen moet worden wil de lezer meer informatie hebben.

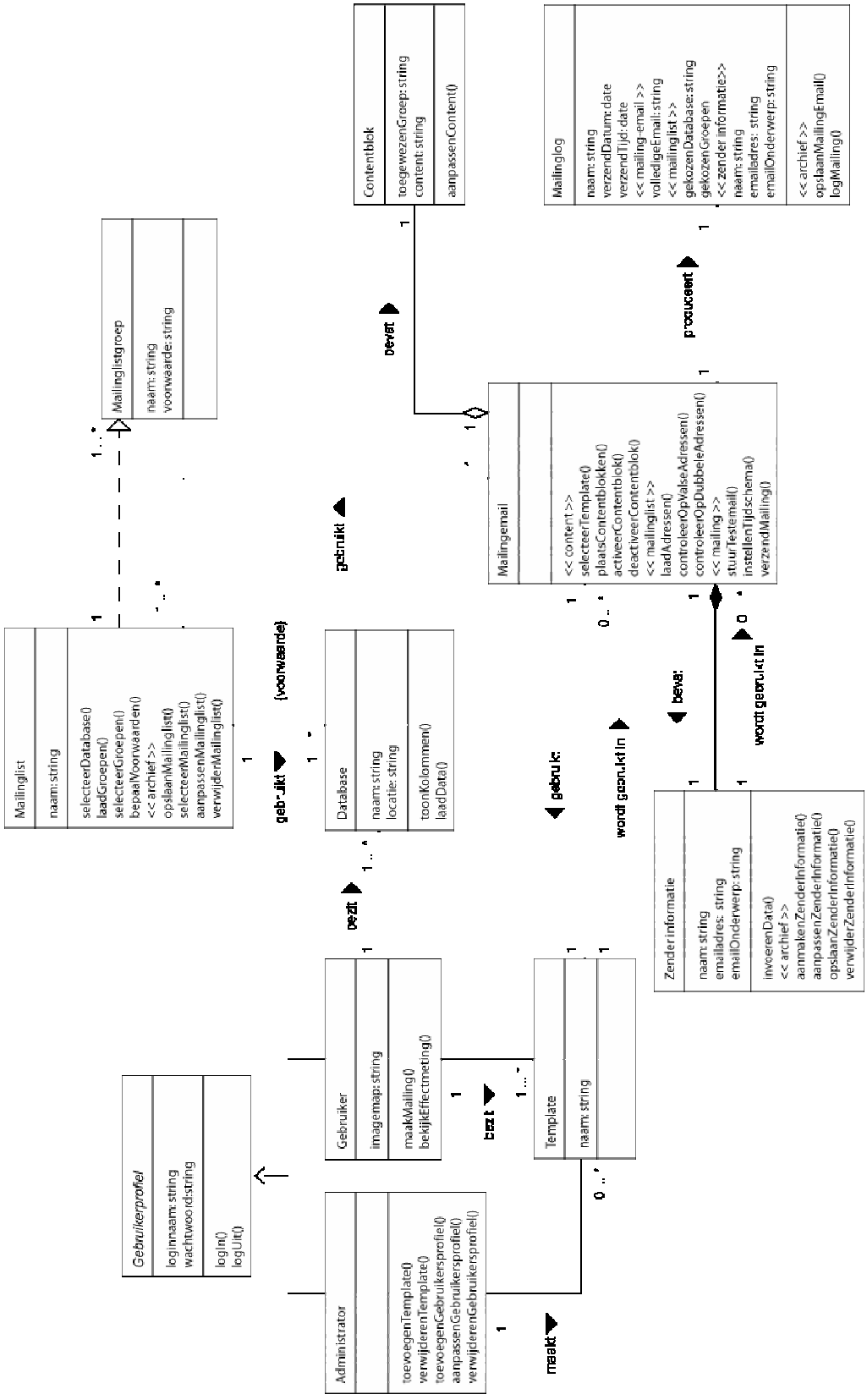
7.1 UML diagrammen

Dit is een selectie van de diagrammen die zijn toegevoegd in het functioneel ontwerp. De toestandsdiagrammen zijn achterwege gelaten omdat deze na implementatie maar beperkt weer konden geven hoe de applicatie functioneerde, de gaten zijn veroorzaakt door de kenniskloof ten tijde van het opstellen van de diagrammen.

7.1.1 Klassendiagram

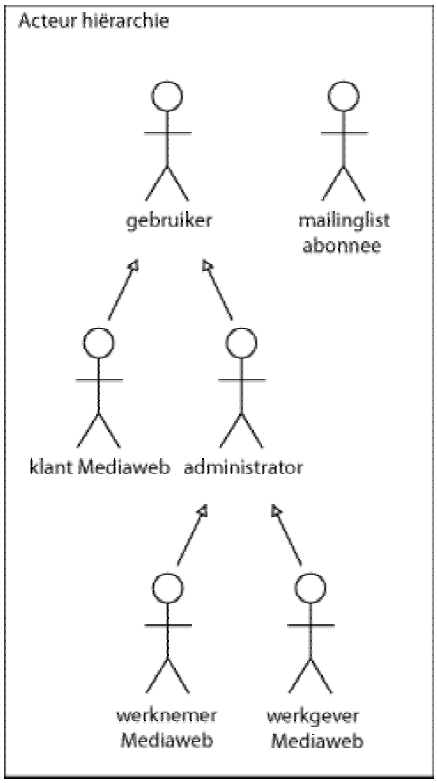


Figuur 7.1 klassendiagram packages

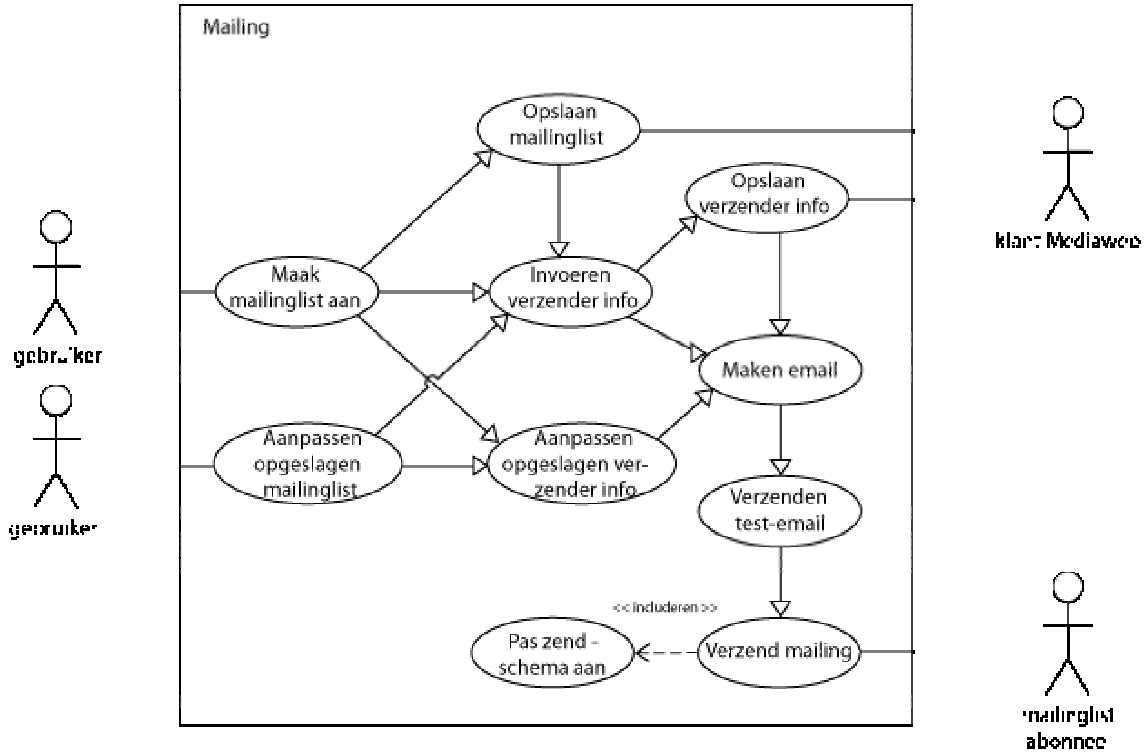


Figuur 7.2 Klassendiagram mailingapplicatie

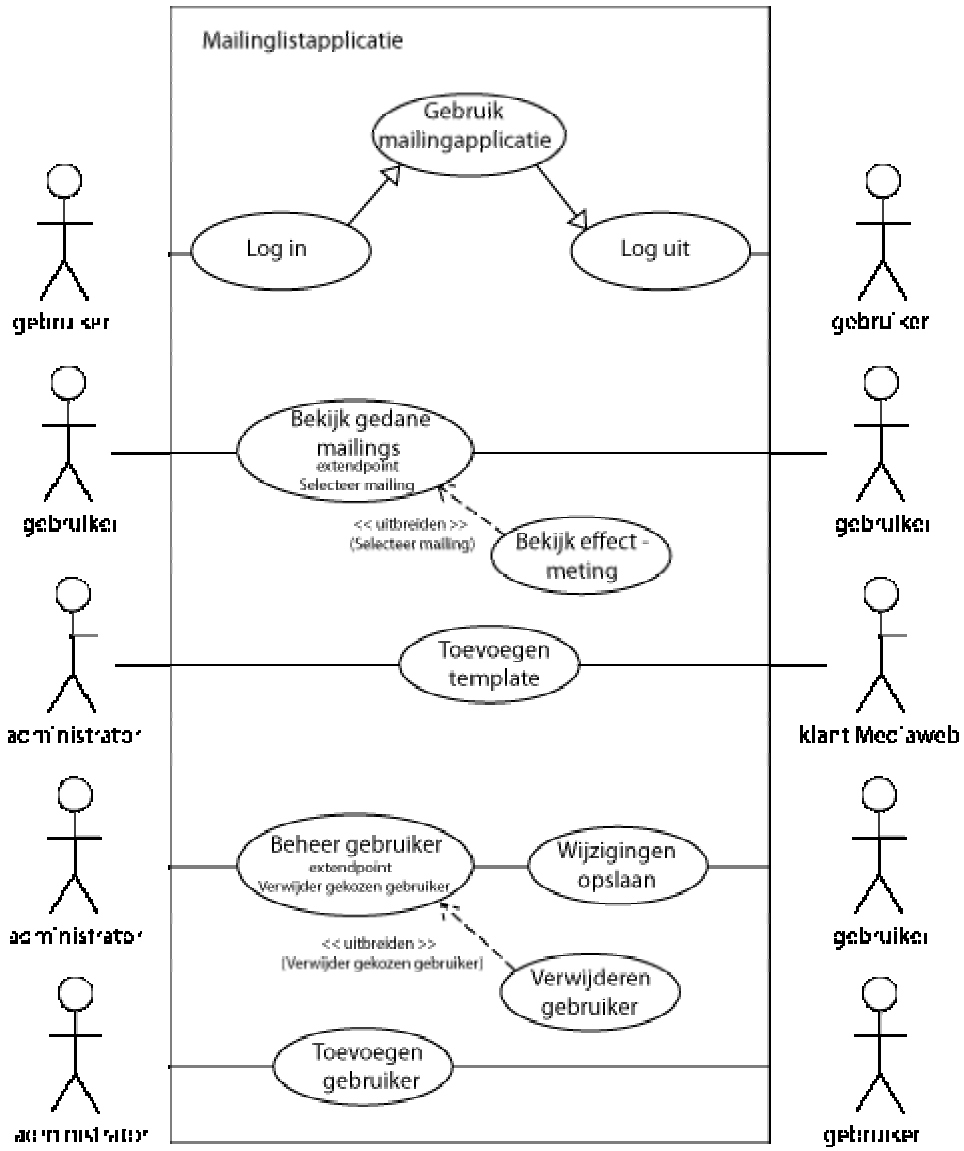
7.1.2 Use case - diagrammen



Figuur 7.3 use case - actor hiërarchie

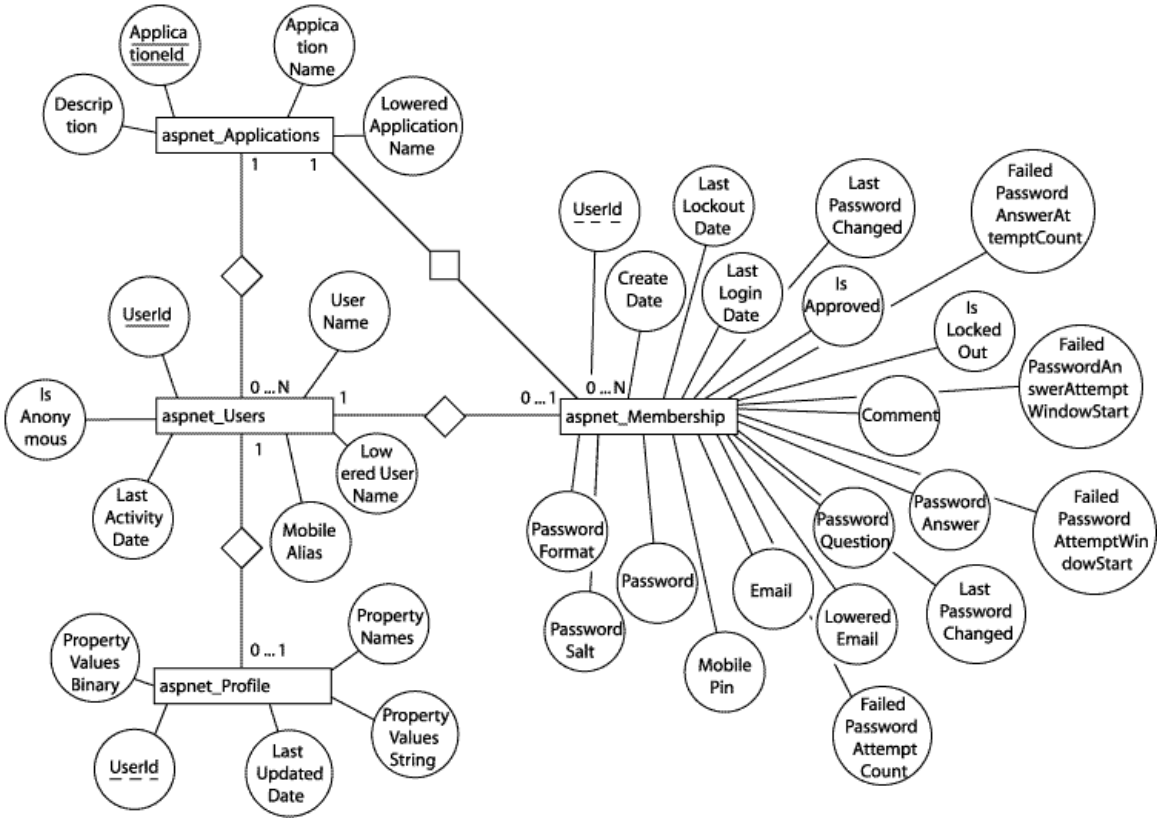


Figuur 7.4 use case - mailing verzenden

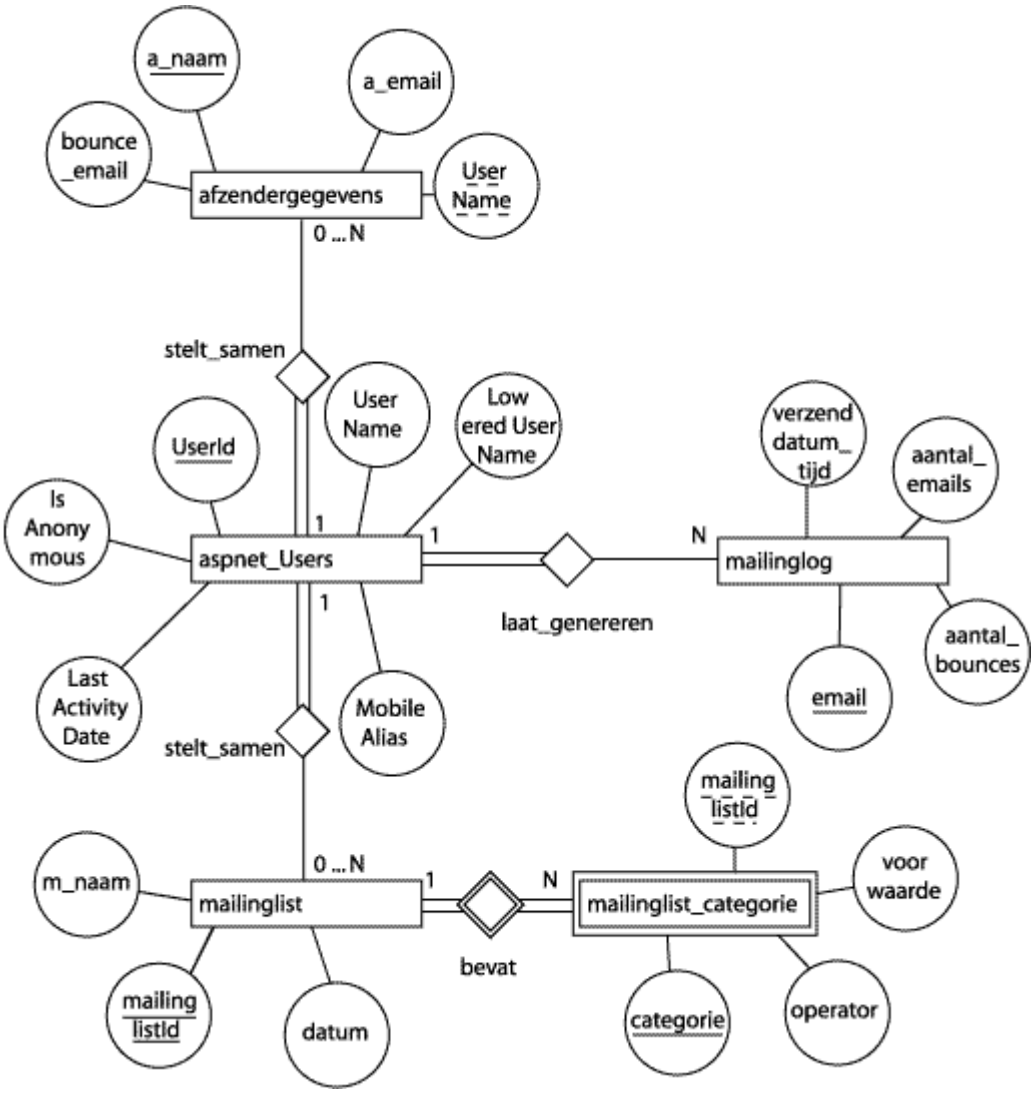


Figuur 7.5 use case - mailingapplicatie

7.2 ERD van het databasemodel



Figuur 7.6 ERD - profile en membership sectie



Figuur 7.7 ERD - mailingapplicatie

Hoofdstuk 8. Evaluatie procesgang en producten

In dit hoofdstuk wordt eerst het proces en daarna de producten door de afstudeerder onder de loep genomen en gemotiveerd beoordeelt. Dit wordt zoveel mogelijk in chronologische volgorde en objectief gedaan.

8.1 Evaluatie verloop van het project

In dit gedeelte van het hoofdstuk worden de belangrijkste delen van het proces geëvalueerd, door de scriptie heen heb ik al mijn mening gegeven over het verloop van de zaken dus ik beperk me hier de kernpunten van het proces.

Het doel van de opdracht was het vervaardigen van een goed ontwerp voor een mailingapplicatie om deze daarna te realiseren. Tijdens de stage heb ik ook dit doel nagestreefd.

8.1.1 Het maken van een goed ontwerp

De ontwerpfases zijn succesvol doorlopen. Het maken van het ontwerp van de applicatie was een geleidelijk proces waarbij in de gesprekken met de opdrachtgever en in de ontwerpen steeds meer informatie beschikbaar kwam. Bij de gesprekken met de opdrachtgever werd de applicatie herhaaldelijk onder de loep genomen om meer informatie over de werking ervan te verkrijgen, soms voor het ontwerp en soms om specialistische kennis te verkrijgen zoals het verzenden van een mailing. De ontwerpen waren doelbewust bij het begin algemeen gehouden en pas bij de latere ontwerpen werd er de interface van de applicatie detail beschreven.

8.1.2 Het implementeren van het ontwerp

Bij de voorbereidingen realisatie van het ontwerp ging het verkeerd en toen ik aan het bouwen sloeg was ik niet voldoende voorbereid of ingewerkt. Ik had in de voorgaande fases kennis en ervaring op kunnen doen maar er zaten nog aardig wat gaten in, hoewel de aanpak bij de realisatie geschikt was werkte het niet.

Ik ging per bouwdeel zoeken naar mogelijk oplossingen, eerst in MSDN en op de ASP.NET website en daarna op het internet, en met behulp van de gevonden informatie probeerde ik het bouwdeel te realiseren. Opkomende fouten werden op dezelfde manier opgelost door te zoeken naar informatie gevolgd door beslissingen maken en vervolgens het oplossen, de stappen herhalend wanneer het probleem niet werd opgelost of een andere zich voordeed na het oplossen. Het probleem zat hem in dat het soms lang duurde voordat er een oplossing gevonden was en dat ik soms met moeite had om de nieuwe informatie te bevatten, vaak was het teveel en gevarieerd of mistte ik de voorkennis om het helemaal te snappen.

Wanneer ik meer tijd had besteed aan het leren van de basis van ASP.NET voor de aanvang van het bouwen dan had ik minder moeite gehad met deze fase en was de applicatie waarschijnlijk afgeweest. Hoewel ik er in de planning rekening mee gehouden had door het

merendeel van de projecttijd vrij te houden om te (leren) programmeren had ik er beter aan gedaan in plaats daarvan het leren uit te spreiden en goed voorbereid de bouwfase in te gaan.

8.1.3 Planning

Een overenthousiaste (en onrealistische) planning maakte in het begin dat ik vrij snel achter liep, hoewel deze planning genoeg ruimte liet voor het programmeren was er te weinig ruimte besteed aan het ontwerp van de applicatie. Dit heb ik in de planning rechtgetrokken en in plaats van de eerste 4 weken voor het ontwerp te gebruiken werden het er ongeveer 10 waardoor de ontwerpen de diepte konden krijgen die nodig was voor het realiseren van het ontwerp.

Mijn idee om het leren van ASP.NET buiten de werkuren te doen bleek al gauw niet werkbaar. Na een werkdag achter de PC was dit niet doenbaar waardoor mijn leertijd met 2/3 werd gereduceerd. Het heeft ook lang geduurd voordat ik een geschikt boek over ASP.NET had gevonden en verkregen met voldoende diepgang maar niet te ingewikkeld verwoord. Tot die tijd was het voornamelijk aanmodderen met de tutorials die beschikbaar waren via de ASP.NET pagina. Tutorials leren je één manier om iets uit te voeren maar geven vaak niet aan waar de verworven kennis past in het totaalbeeld.
De gebrekkige kennis die ik op deze wijze heb vergaard hebben tijdens de realisatie veel parten gespeeld, het kostte mij meer tijd en inspanning om dezelfde hoeveelheid werk uit te voeren waardoor de applicatie ten tijd van het schrijven van de scriptie niet af is.

8.2 Wensen en eisenlijst

De wensen en eisen - lijst is essentieel voor het slagen van een project. Er wordt in heldere taal aangegeven waaraan de één of meerdere eindproducten moeten voldoen waar de opdrachtomschrijving dat in globale zin doet. Voor zowel de opdrachtgever als de opdrachtuitvoerende is het een controlemiddel maar voor de opdrachtuitvoerende bevat de lijst ook de richtlijnen voor het ontwerp.

In eerste instantie vond ik de lijst aan de kleine kant met een nadruk die teveel lag op de werking van de processen van de applicatie zonder aandacht te besteden aan de interface of de performance. Naderhand stond ik stil bij het feit dat de wensen en eisen –lijst de verzameling van eisen van de opdrachtgever betrof en dat deze op dat punt verschilde van een lijst van systeemeisen waarvan ik dacht dat de wensen en eisen – lijst er één was. In plaats van de wensen en eisen aan te vullen en zo verder te gaan als dat bedoeld was met dit document besloot ik de lijst aan te vullen door de ontwerpen in de daaropvolgende fases te gebruiken om het ontwerp de stevigheid te geven, wat ik vervolgens gedaan heb.

8.3 Evaluatie Functioneel ontwerp

Het originele functionele ontwerp, het navigatieschema en de beschrijvingen van de verschillende onderdelen van de applicatie, was een mooi begin om de applicatie uiteen te zetten. Er is puur gekeken naar wat er tijdens de gesprekken over de applicatie naar boven is gekomen en dat is op papier gezet. De lezers met geen kennis van UML hebben dit

gedeelte kunnen gebruiken om een beeld van de applicatie te krijgen en dit gedeelte was een goede opstap voor het maken van de diagrammen.
De UML diagrammen zijn later toegevoegd en hebben mij geholpen om een beter en scherper beeld van de mailingapplicatie te krijgen. Door de eerder verkregen informatie om te zetten in een model kon ik ook meer gerichte vragen stellen om het beeld uit te breiden.

Het klassendiagram gaf me de gelegenheid om de betrokken objecten vanuit ontwerperoogpunt op een rij te zetten en bekende eigenschappen toe te kennen. Aan de hand van de eerste gesprekken heb ik die eigenschappen kunnen uitdiepen evenals de relaties tussen de objecten.
De use cases waren nuttig om voor het weergeven van de te doorlopen processen binnen de applicatie en om aan te geven welke gebruikersgroepen van welke processen gebruik konden.
Toestandsdiagrammen net als later de flowcharts waren het meest gevoelig voor veranderingen omdat ze op het niveau van de werking van de applicatie beschreven. Omdat mijn kennis op dat punt nog te globaal was moeten deze diagrammen met een korreltje zout genomen worden, ze zijn een indicatie geweest van wat er gedaan moet worden maar het kon gemakkelijk zo zijn dat er een paar stappen overgeslagen waren of dat er een verkeerde handeling gekozen is.

Het functionele ontwerp en met name de UML diagrammen heb ik later kunnen gebruiken om de database van de applicatie mee te maken, wat anders zeker meer tijd had vereist.

8.4 Evaluatie Grafisch ontwerp

Omdat het functionele proces enkel de werking en de daarbij benodigde onderdelen beschreef was er toevoeging nodig die de overbrugging maakte tussen de werking en de uitwerking. Het grafische ontwerp heeft het ontwerp diepte gegeven door de gebruikerinterface, op voornamelijk grafisch niveau, vast te leggen.

De schetsen hebben zowel mij als de opdrachtgever aan het denken gezet en punten die eerder niet aan bod waren gekomen, hiermee doel ik voornamelijk op invoeging bij het CMS, zijn mede hierdoor boven water gekomen.

De huisstijl van het content management system was middel om de eigen huisstijl te definiëren. Door het CMS onder de loep te nemen en uit te werken heb ik daarna met wat aanvullingen de eigen huisstijl voor de applicatie mee kunnen maken.

8.5 Evaluatie Interactief ontwerp

Het maken van het interactief ontwerp heeft weinig toegevoegd aan het ontwerp van de applicatie voor de ontwerper. Het was duidelijk voor een opdrachtgever bedoeld, voor mij had met uitproberen of een vroege start aan de applicatie evenveel resultaten geworpen maar dan was de overstap voor de opdrachtgever waarschijnlijk te groot geweest. Bij het van een ontwerp naar een volwaardige prototype of een eindproduct gaan zijn de controlepunten te ver uit elkaar en kan de opdrachtgever het gevoel geven dat deze geen invloed heeft en dat het project een eigen leven lijdt.

8.6 Evaluatie Databasemodel

Het databasemodel was nodig om de passende database te kunnen maken voor de applicatie. Het model is een aantal keren aangepast vanwege wensen van de opdrachtgever, inconsistenties in het model en om gebruik te kunnen maken van functies binnen ASP.NET. Ik zou graag nog een keer het model hebben nagekeken omdat het in delen aanpassen van een database(model) de structuur aan kan tasten.

8.7 Evaluatie Mailingapplicatie

Het grootste minpunt aan de applicatie is dat deze ten tijde van het schrijven van de scriptie nog niet is afgerond en ik niet weet of het afgemaakt kan worden binnen de tijd voor de zitting. Omdat de applicatie niet af is wordt het moeilijk er een oordeel over te maken, het gaat de goed kant op.
Gebruikersbeheer, inlogsysteem, mailinglistbeheer en afzendergegevensbeheer zijn op een paar bugs na klaar. Ik ben van plan nu eerst de het mailingproces doorloopbaar te maken en vanuit daar de functionaliteiten binnen het proces uit te breiden.

Woordenlijst

CMS
: Content Management System

Conceptueel datamodel
: Een weergave van de entiteiten, hun beperkingen en onderlinge relaties

Content Management System
: Een applicatie die de gebruiker in staat stelt een website of applicatie te beheren. Het beperkt zich veelal tot de inhoud of vormgeving.

Database
: Een virtueel opslagmedium.

Database Management System
: Een applicatie die één of meerder databases en haar data te beheren.

DBMS
: Database Management System

ER –diagram of ERD
: Entity Relationship – diagram

Entiteit
: Een cluster van attributen die (een deel van) een object weergeeft

Entity Relationship – diagram
: Een diagram die de relaties en de attributen van entiteiten (in de database vertegenwoordigt in tabellen) weergeeft

HTML-tags
: De syntax van HTML om normale tekst te onderscheiden van HTML objecten. Is te herkennen aan de volgende opbouw: < … /> of < … > *eventuele tekst* < … />

IDE
: Integrated Development Environment

Implementatiemodel
: Binnen een databasemodel is het de volledige SQL-code om de database te kunnen bouwen

Mailing
: Het verzenden van een email naar één of meer ontvangers

Normaliseren

Normaliseren is een formeel proces om te bepalen welke attributen in een relatie opgenomen kan worden,

Object

1. Een weergave van iets uit de realiteit geplaatst in een model
2. Een instantie van een klasse (geclusterde groep attributen en functies)

Relationeel representatiemodel

Genormaliseerde conceptueel datamodel, het is de vertaling van het conceptueel databasemodel voor het implementatiemodel

XUAN-model

Een bouwkaart van een programma waar per venster/scherm elk object van de interface in kaart is gebracht en waar per venster/scherm elke reactie en handeling van het systeem samen met manipulatie van de data opslagmedia in stappen is genoteerd.

Referentielijst

Boeken

Application note: Guidelines for proper mailing list management
MAPS, zie referentielijst: websites

Het handboek voor Internet- & Intranettechnologie, Jeroen Vanheste
Pearson Education Uitgeverij b.v., 2e druk, ISBN 90 430 0491 X

IBM Data Processing Techniques, Flowcharting Techniques
International Business Machines Corporation, 2e druk, zie referentielijst: websites

ICT-zakboekje, informatie- en communicatietechnologie
Koninklijke PBNA b.v., 2e druk, ISBN 90 6228 303 9

Professional Asp.net 2.0, B. Evjen S. Hanselman F. Muhammad S. Sivakumar D. Rader
Wiley Publishing inc., 1e druk, ISBN 0 7645 7610 0

UML in 24 uur, Joseph Schmuller
Academic service, 1e druk, ISBN 90 395 1344 9

Websites

Application note: Guidelines for proper mailing list management
http://www.mail-abuse.com/pdf/AN_ListMgmtGuidelines_052604.pdf

AspEmail, Persits Software Inc.
http://www.aspemail.com

AspNetEmail, Advanced Intellect LLC
http://www.aspnetemail.com

IBM Data Processing Techniques - Flowcharting Techniques
http://www.fh-jena.de/~kleine/history/software/IBM-FlowchartingTechniques-GC20-8152-1.pdf

Mail Abuse Protection System - MAPS, Trend micro inc.
http://www.mail-abuse.com

The web matrix, Microsoft Corperation
http://asp.net/webmatrix/default.aspx?tabindex=4&tabid=46

Bijlagen

Bijlage 1: Plan van Aanpak

Plan van Aanpak: Mailingapllicatie Mediaweb

Versie 2.0
Datum: 15-6-2006

Schrijver/student: Jeroen Cramer HHS 20021998
Opdrachtgever/mentor: Peter van Marwijk

1. Inleiding

Dit document is geschreven naar aanleiding van het leerplan voor afstuderen uit het VIA studentenstatuut deel 2 en om als leidraad te dienen voor het project.

Het doel van dit document is het afbakenen van het project door het op papier zetten van afspraken, richtlijnen en een planning. Het document is bedoeld voor de projectleden en de opdrachtgever, in dit geval de student en de bedrijfsmentor, en eventueel voor de examinatoren ter beoordeling van het project.

In dit document worden achtereenvolgens behandeld:

- De opdrachtomschrijving, de opdracht in detail
- De aanpak, de te gebruiken methoden en technieken evenals de planning
- De projectinrichting, de rolbeschrijving en afspraken binnen het project
- De kwaliteitsborging, voorwaarden voor goedgekeurde producten

1.1 Achtergrond van het probleem

Een van de veel geleverde diensten van Mediaweb is het maken en hosten van een website waarbij er extra mogelijkheden zijn. Een andere dienst die vaak ernaast wordt uitgevoerd is het uitvoeren een mailing. Bij de website kunnen bezoekers zich dan inschrijven om daarna (periodiek) emails te ontvangen.
Een deel van de klanten die door Mediaweb een website voor hen hebben laten ontwikkelen, maken gebruik van de bovengenoemde mogelijkheid om bij Mediaweb mailings (nieuwsbrieven, mailinglists e.d.) te versturen aan hun cliëntèle. Aan deze mailings worden steeds meer eisen gesteld, zowel in volume als in effectmeting (het nagaan van het effect van de mailing).
Volume is nog geen echt probleem, de grotere mailings worden buiten het gebruikelijke programma gemaakt en verzonden. Effectmeting van de mailing is nu te arbeidsintensief voor het bedrijf. Voorts moet het effect van een mailing kunnen worden gemeten, groot of klein, zonder dat een medewerker van mediaweb daar lang mee bezig is.

1.2 Aanleiding van de opdracht

Voor het verzenden van standaard mailings gebruikt Mediaweb al geruime tijd de zelf ontwikkelde Newspublisher waarmee klanten zelf een nieuwsartikel kunnen opstellen en deze als email kunnen verzenden. Mediaweb wil haar mailingdienst vernieuwen zodat deze beter aansluit op eisen en wensen van de klant.

2. Opdrachtomschrijving

In dit hoofdstuk wordt de opdrachtsomschrijving nogmaals beknopt beschreven en samen met de overige hoofdstukken moet het een beeld geven van hoe het project zal verlopen. sommige punten zijn toegevoegd zoals randvoorwaarden en risicofactoren, maar het merendeel is terug te vinden in de voorlopige opdrachtsomschrijving.

2.1 Opdrachtgever

De opdracht wordt in naam van Mediaweb Internet Integrators uitgvoerd met Peter van Marwijk als de persoon die het project overziet en als opdrachtgever en bedrijfsmentor optreedt.

2.2 Probleemstelling

De applicatie die de mailings verzorgt is niet toereikend om in te kunnen spelen op de wensen v.d. klant en het zou meer tijd kosten om de applicatie aan te passen dan deze te vervangen door een nieuwe applicatie.

Klanten willen grote hoeveelheden emails per mailing kunnen verzenden. Voor volumineuze mailings wordt nu nog een alternatieve oplossing gebruikt omdat de Newspublisher deze niet aankan. Deze handmatige uitgevoerde mailings kosten verhoudingsgewijs veel tijd en inspanning.
Deze oplossing wordt ook gebruikt voor het op maat maken van een email voor abonnees die zich bij verschillende categorieën hebben ingeschreven. Ook dit is iets wat Mediaweb graag standaard aan de klanten wil aanbieden, maar omdat newspublisher niet catogoriegebonden inhoud voor de mailing ondersteunt moet dit buiten het programma gedaan worden.

Verder willen klanten inzicht hebben in de effecten van een mailing. Op het moment wordt een effectmeting van een mailing door Mediaweb zelf uitgevoerd, een proces dat arbeidsintensief is.

2.3 Doelstelling opdracht

Het doel van de opdracht is de ontwikkeling van een mailinglist-applicatie die met de middelen binnen het bedrijf gerealiseerd kan worden. De applicatie dient door klanten gebruikt te gaan worden.

Van deze applicatie wordt verwacht dat deze beschikt over:

- Een functionaliteit om een op een template gebaseerde (volumineuze) mailing stapsgewijs te verzorgen
- Een koppeling met de bestaande abbonnee databases
- Een functionaliteit die inzicht biedt in de effectiviteit van een mailing d.m.v. statistieken, in het bijzonder:
    - De weergave van het aantal verzonden emails
    - De verwerking en weergave van bounces en errors
    - De weergave van het aantal gebruikte doorverwijzingen vanuit de mailing
- Een ontwerp dat aansluiting vindt op bestaande ontwikkeltechnieken binnen Mediaweb
- Een mogelijkheid om in de toekomst uitbreidbaar te zijn door Mediaweb zelf

2.4 Opdrachtformulering

Het ontwerpen en vervaardigen van een mailinglistapplicatie die te benaderen is via het internet en te gebruiken door zowel klanten als medewerkers van Mediaweb. De belangrijkste functies zijn mailings maken en het meten van hun effecten. Dit is een interne opdracht van het bedrijf.

2.5 Op te leveren producten en diensten

Het 8 fasen model dat wordt aangehouden vereist een aantal producten, hierbuiten zijn er geen extra producten aangewezen. (exclusief documentatie vereist van de Haagse Hogeschool) Voor meer informatie over het 8 fasenmodel verwijs ik u door naar hoofdstuk 3.3 werkzaamheden.

- Een mailinglistapplicatie
- Een functioneel ontwerp
- Een grafisch ontwerp
- Een interactief ontwerp
- Een (kort) rapport van de testresultaten of een wijzigingsrapport

Naast het uitvoeren van de gegeven opdracht zijn er geen afspraken over verder te leveren diensten.

2.6 Randvoorwaarden

- De opdracht dient volgens het 8 fasen model uitgevoerd te worden, afwijkingen mogen mits deze verantwoord worden
- De opdracht en het project dienen in 17 weken in zijn geheel uitgevoerd te kunnen worden
- De mailinglistapplicatie moet in de ASP.NET omgeving geimplementeerd worden

2.7 Risicofactoren

De nieuwe server waar de applicatie op zal draaien moet bij de start van het project nog geinstalleerd worden. Het kan zo zijn dat de installatie vertraging of uitstel oploopt door onvoorziene omstandigheden. In dit geval zal er overlegd worden met de serverbeheerder en de bedrijfsmentor over de te ondernemen stappen.

3. Aanpak

In dit hoofdstuk worden onderwerpen besproken die het functioneren van het project ten goede komen. Naast een planning en geplande activiteiten worden de methoden en technieken aangegeven evenals andere nuttige richtlijnen en standaards binnen het project. De in dit hoofdstuk besproken zaken zijn uitgangspunten en kunnen dus tijdens het project aangeult of aangepast worden naar wat de situatie vereist.

3.1 Methoden en technieken

Methoden:
- 8 fasen model

Technieken:
- UML, vnl. usecases en toestanddiagrammen

Voor het ontwikkelen van de mailinglistapplicatie wordt er gebruikt van het 8 fasen model die het bedrijf gebruikt voor haar ontwikkeling van websites en ander webgerelateerde applicaties. De mailinglistapplicatie is in feite een applicatie die via internet benadert zal worden en het verzoek was om dit model aan te houden bij de ontwikkeling van de applicatie, om deze redenen wordt het 8 fasen model gebruikt.

3.2 Standaards en richtlijnen

Voor documentatie wordt de volgende standaard aangehouden:

	Font	Size
Titel	Tahoma	16
Kop	Tahoma	14
Subkop	Tahoma	13
Tekst	Tahoma	11
Verwijzing	Tahoma, underline	11

Voor het ontwerpen van de mailinglistapplicatie wordt er gebruik gemaakt van de richtlijnen die staan op de website van MAPS, Mail Abuse Prevention System, over mailinglist management. (http://www.mail-abuse.com/an_listmgntgdlines.html)

Voor de implementatie van het ontwerp wordt Visual Studio 2005 gebruikt. De (programeer) talen die hierbij gebruikt worden zijn VisualBasic.NET, HTML, SQL en de ASP server controls.

3.3 Werkzaamheden (fasering en activiteiten)

Dit zijn in het algemeen de activiteiten die uitgevoerd worden binnen het project en geplaatst binnen de verschillende fasen van het 8 fase model.

Fase 1: Wensen en eisen

In deze fase worden in overleg de wensen van de opdrachtgever in kaart gebracht evenals het doel van de website/applicatie. Na het in kaart brengen van de wensen volgt een goedkeuringsmoment met de opdrachtgever.

- Inventarisatie wensen en eisen mailingapplicatie
- Maken Beschrijving gebruikers
- Terugkoppel-/goedkeuringsmoment

Fase 2: Functioneel ontwerp

Aan de hand van de wensen en eisen die aan het product worden gesteld wordt er een functioneel ontwerp gemaakt. Dit bestaat uit een navigatieschema en/of een beschrijving van de verschillende onderdelen van de website/applicatie. In deze fase wordt er een planning opgesteld. Na het maken van het functioneel ontwerp volgt een goedkeuringsmoment met de opdrachtgever.

- Verwerken wensen en eisen in ontwerp
- Aanvullen v.h. ontwerp
- Maken navigatieschema
- Maken planning
- Terugkoppel-/goedkeuringsmoment

Fase 3: Grafisch ontwerp

Aan de hand van het functioneel ontwerp wordt er een grafisch ontwerp gemaakt, dit blijft beperkt tot een aantal kernpagina's. Na het maken van het grafisch ontwerp volgt een goedkeuringsmoment met de opdrachtgever.

- Maken schetsen a.d.h.v. functioneel ontwerp
- Maken grafisch ontwerp
- Terugkoppel-/goedkeuringsmoment

Fase 4: Interactief ontwerp

Aan de hand van het grafisch ontwerp wordt er een interactief ontwerp gemaakt, de in fase 3 gemaakte pagina's worden hier gebouwd zonder veel van de functies. Oplossing op problemen die uit de vormgeving voort zijn gekomen worden op dit punt voorgedragen. Na het maken van het interactief ontwerp volgt een goedkeuringsmoment met de opdrachtgever.

- Uitwerken voorbeeldpagina's a.d.h.v. grafisch ontwerp
- Terugkoppel-/goedkeuringsmoment

Fase 5: Bouwen

Conform het goedkgekeurde functionele, grafsiche en interactieve ontwerp de website/applicatie volgens de planning opgebouwd.

- Per bouwdeel opbouwen van applicatie
- Testen van het bouwdeel
- Invoegen bouwdeel in applicatie

Fase 6: Testfase

In een korte periode wordt de website/applicatie getest op functionele en technische aspecten, er kunnen op dit punt nog kleine wijzigingen worden doorgevoerd.

- Testplan opzetten
- Afgeronde applicatie testen
- Kort rapport over de ontdekte fouten en verbetervoorstellen
- Fouten en verbeteringen afhandelen

Fase 7: Oplevering

Na goedkeuring wordt de website/applicatie ingevoerd en vindt er een laatste controle plaats.

- Terugkoppel-/goedkeuringsmoment
- Invoering applicatie
- Wijzigingsrapport schrijven

Fase 8: Nazorg

Na een periode wordt er nagegaan of de doelen zijn bereikt en hoe de bezoekers/gebruikers reageren op het product. Er wordt ook nagedacht hoe het Product in de toekomst verbeterd kan worden.

- Wordt door het bedrijf zelf uitgevoerd

3.4 Planning

Wanneer een bouwdeel van de applicatie voltooid is, zal deze nagelopen worden om de werking te controleren en eventuele fouten te achterhalen. Herstelwerkzaamheden en verbeterpunten worden genoteerd en zullen afhankelijk van hun prioriteit en de beschikbare tijd behandelt worden of voldoende gedocumenteerd aan het eind van een cyclus of fase geplaatst worden om dan verwerkt te worden. Het kan dus zo zijn dat één of meerdere punten niet verwerkt worden, in dit geval zullen ze in een wijzigingsrapport worden opgenomen als onbehandelt.

Bij de 1e week is de uitvoering ook toegevoegd, *Cursieve* tekst is schoolgerelateerd.

Week	Data	Activiteiten	Producten
1	15/5 - 19/5	Kennismaking Mediaweb Installatie van werkplek Uitdiepen opdracht Maken wensen en eisen-lijst Onderzoek bedrijfsdocumentatie Uitbreiden kennis (spam) Uitbreiden kennis (mailinglist)	W & E -lijst (voorl)
2	22/5 - 26/5	Bespreking bedrijfsmentor: • Wensen en eisen-lijst (voorl) • Functioneel ontwerp (voorl) Maken functioneel ontwerp Evt. Verbeteren wensen en eisen-lijst Goedkeuringsmoment W & E-lijst Terugkoppelmoment F. Ontwerp *Maken Plan van Aanpak*	Wensen en eisen-lijst F. ontwerp (voorl) *Plan van Aanpak*
3	29/5 - 02/6	Evt. Verbeteren functioneel ontwerp Goedkeuringsmoment F. Ontwerp	
4	05/6 - 09/6	Evt. Verbeteren functioneel ontwerp Maken grafisch ontwerp Goedkeuringsmoment G. ontwerp Doornemen en evt. al verbeteren voorlopige opdrachtsomscrhijving *Bekendmaking 1e & 2e examinator*	Functioneel ontwerp Grafisch ontwerp

5	12/6 - 16/6	Goedkeuringsmoment I. Ontwerp *Maken def. opdrachtsomschrijving in overleg met de 2 examinatoren*	
6	19/6 - 23/6	Maken Interactief ontwerp *Maken def. opdrachtsomschrijving in overleg met de 2 examinatoren* *19/6 Bezoek examinatoren aan Mediaweb*	
7	26/6 - 30/6	Bouwen applicatie *Maken definitieve opdrachtsomschrijving in overleg*	*Definitieve opdrachtsomschrijving*
8	03/7 – 07/7	Bouwen applicatie *Inleveren def. opdrachtsomschrijving (*.txt) bij examinatoren 7/6* *Maken voortgangsverslag*	*Voortgangsverslag*
9	10/7 – 14/7	Bouwen applicatie *10/6 of 11/6 Inleveren voortgangsverslag bij examinatoren*	
10	17/7 – 21/7	Bouwen applicatie	
16/7 -16-8		OV-weekkaart niet geldig (alleen korting), het gat in de planning. Controle voortgang Bouwen applicatie	
11	12/8 – 18/8	Bouwen applicatie	
12	21/8 – 25/8	Bouwen applicatie	
13	28/8 – 01/9	Testen applicatie Herstellen van fouten Aanbrengen verbeteringen *Maken bouwplan afstudeerscriptie*	Wijzigingsrapport *Bouwplan afstudeerscriptie*
14	04/9 - 08/9	Terugkoppelmoment Oplevering applicatie Schrijven wijzigingsrapport *Inleveren gedet. bouwplan van het afstudeerverslag uiterlijk 08/9*	

15	11/9 – 15/9	Schrijven afstudeerscriptie *Bespreking met examinatoren voortgang, bouwplan en opdracht* *Gesprek examinatoren met bedrijfsmentor(zie leerplan)*	
16	18/9 – 22/9	Schrijven afstudeerscriptie *Bespreking met examinatoren voortgang, bouwplan en opdracht* *Gesprek examinatoren met bedrijfsmentor(zie leerplan)*	
17	25/9 – 29/9	Schrijven afstudeerscriptie	
18	2/10 - 6/10	Afmaken aan afstudeerscriptie Inleveren 4 x afstudeerscriptie, **6 oktober voor 10:30 - 13:00**	*Afstudeerscriptie (x4!)*
19	9/10–13/10	*Maken presentatie* *Voorbereiden op hoorzitting*	
20	16 - 20/10	*Afstudeerbespreking, precieze tijd komt nog*	*Presentatie + hoorzitting*
22	30/10–3/11	*Afstudeerbespreking, precieze tijd komt nog*	*Presentatie + hoorzitting*
24	14/11	*getuigschrift ontvangen*	*Voldaan gevoel*

4. Projectinrichting

In dit hoofdstuk worden de informatievoorziening, de organisatie en de faciliteiten van de organisatie besproken.

4.1 Projectorganisatie

Het project wordt uitgevoerd door de student, er zijn geen andere medewerkers van het bedrijf of externen die er aan meewerken.

De bedrijfsmentor, tevens de opdrachtgever, voorziet de student van informatie, geeft advies en beoordeelt het werk van de student.

De examinatoren beoordelen de student op zijn werkwijze, de geleverde documentatie en de hoorzitting aan het einde van het traject. In de tussentijd zullen zij een bezoek leveren aan Mediaweb en de student adviseren op de tussentijdse voortgang.

4.2 Informatie

Tijdens de afronding van een fase bij de 1^e 4 fasen van het 8 fasen model is er goedkeuringsmoment waarin er samen met de opdrachtgever naar de gepresenteerde voortgang wordt gekeken. Op dit punt krijgt de student feedback en wanneer de opdrachtgever tevreden is een groen licht.
Tussentijds zijn er besprekingen met de opdrachtgever om benodigde informatie te verkrijgen of op met elkaar te stemmen. Geen van de afspraken heeft een periodiek karakter dit omdat de opdrachtgever een drukbezet schema heeft.

De eerste 3 fase van het 8 fase model vereisen documentatie, in de overige fasen gebeurd dit per discretie van de gebruiker van het model. Het is de bedoeling van de student om een duidelijk ontwerp af te leveren alvorens te beginnen aan de realisatie. Lukt dit niet met de beschikbare documentatie van het model dan zullen er extra documenten aan toe gevoegd worden om dit gat te vullen.

4.3 Faciliteiten

Hier worden kort de middelen besproken die de student tot zijn beschikking heeft voor het ontwerp en de realisatie van de mailinglistapplicatie.

Benodigde software

De software voor het creëren van het programma is reeds aanwezig binnen het bedrijf. De lijst is niet definitief, wanneer de gebruikte software niet toereikend is kan er voor gekozen worden om andere software te zoeken die dat wel is.

Te gebruiken software:

- Visual Studio 2005
- Visual Web Developer 2005 Express Edition
- SQL Server 2003
- FrontPage 2003
- ASP Email

Benodigde hardware

Een werkruimte met de benodigde software en hardware om het ontwerp en de applicatie te realiseren

De mailinglist-applicatie zal of draaien op de huidige mailserver van het bedrijf of er zal een nieuwe (virtuele) server voor ingericht worden, hierbij is er ondersteuning vanuit het bedrijf door de serverbeheerder die verantwoordelijk is voor de mailserver.

5. Kwaliteitsborging

In dit hoofdstuk worden de algemene voorwaarden besproken waaraan een product of bouwdeel moet voldoen om als voltooid te worden beschouwd.

Bouwdelen en applicatie

Wanneer er (systeem)eisen aan het product zijn gesteld moet nagegaan worden of het product aan die eisen voldoet. Dit kan door te testen of door een contole uit te voeren maar in alle gevallen moet er een geschikte aanpak geformuleerd worden en eventueel de resultaten gedocumenteerd.

Wanneer een bouwdeel van de applicatie voltooid is wordt deze door de ontwerpers voor zover mogelijk getoetst a.d.h.v. systeemeisen. Fouten en/of verbeterpunten worden na notatie (inclusief motivatie) verwerkt in het bouwdeel, ook dit wordt genoteerd. Deze wijzigingslijsten worden gebruikt tijdens besprekingen en om een wijzigingsrapport op te stellen.

Documentatie

Wanneer het product een document betrefd dient deze leesbaar en begrijpelijk geschreven te zijn, dit hangt af van het doel en de doelgroep van het document. Ongeacht het doel of de doelgroep moet het document uit afgeronde en elkaar logisch opvolgende delen bestaan en dienen er geen openstaande vragen of ontbrekende zaken te zijn na het lezen van het document.

Bijlage 2: Wensen en eisen - lijst

Fase 1: Inventarisatie eisen en wensen

Het doel van de applicatie is het kunnen verzorgen en verzenden van nieuwsbrieven en andere mailings en daarbij de effecten te kunnen meten van een verzonden email en deze in een overzichtelijke vorm weer te geven.

De doelgroep van de applicatie zijn werknemers van Mediaweb en klanten van Mediaweb. De opleiding en ICT-gerelateerde ervaring wisselt sterk binnen deze groep, bij het ontwerp zal hier rekening mee gehouden moeten worden.

De verwachtingen van de opdrachtgever zijn ingedeelt in basis, comfort en luxe. De basiseisen dienen zonder uitzondering verwerkt te worden. Comfort eisen en luxe eisen kunnen, wanneer er niet genoeg tijd of middelen zijn om alle eisen in te willigen, maar gedeeltelijk verwerkt worden waarbij de voorrang wordt gegeven aan comfort eisen.

BASIS

- Elke klant die gebruiker is moet een gebruikersprofiel hebben
- Het gebruikersprofiel moet beheerbaar zijn door de medewerkers van Mediaweb
- Werknemers van Mediaweb moeten in staat zijn gemaakte templates voor klanten beschikbaar te maken in hun gebruikersprofiel
- De applicatie moet het aantal verzonden emails bijhouden per gebruiker voor administratieve doeleinden
- De applicatie moet via het internet benaderbaar zijn
- De applicatie moet onderscheid maken tussen de gebruikers onderling

- Mailinglists moeten ingevoerd kunnen worden door gebruikers
- De applicatie moet een koppeling kunnen maken met bestaande databases
- De applicatie moet in staat zijn 10.000 of meer emails in één mailing te verzenden
- De applicatie moet uit een mailinglist dubbele inschrijvingen kunnen halen

- Het maken van een email voor een mailing moet in stappen doorlopen worden
- Onafhankelijk van de technische achtergrond van een gebruiker moet een email voor een mailing gemaakt en verzonden kunnen worden
- Een mailing moet naar alle abonnees van de mailinglist of een selectie van de subgroepen verzonden kunnen worden
- Errors en bounces van een mailing moeten verwerkt kunnen worden door de applicatie en weergegeven worden in de effectmeting

- De gebruiker moet de email van een mailing kunnen zien zoals deze verzonden zou worden en aanpassingen kunnen aanbrengen alvorens deze te verzenden
- De email van een mailing moet naast een voor alle groepen gelijk gedeelte, een voor elke groep apart gedeelte kunnen bevatten
- Een nieuwsbrief moet opgeslagen kunnen worden
- De opmaak van een email voor een mailing wordt met behulp van een template verzorgt

- Een opgeslagen nieuwsbrief moet opgeroepen kunnen worden en vervolgens verzonden

COMFORT

- De applicatie moet uit een mailinglist foute email adressen kunnen halen
- De applicatie moet de kosten van een mailing kunnen weergegeven

LUXE

- De effecten van emails verzonden naar een mailinglist moeten gemeten kunnen worden
- Bij de effectmeting moeten de verkregen gegevens in zowel tabellen als in grafieken kunnen worden weergegeven
- Mailinglists moeten geimporteerd kunnen worden door gebruikers
- De mailing moet kunnen worden verzonden volgens een bepaald tijdschema
- Er moet een controle uitgevoerd worden op de mailingemail of deze als spam gezien zal worden

De volgende voorwaarden worden niet expliciet door de student uitgevoerd maar zijn wel van belang:

- De applicatie moet op hetzelfde netwerk draaien als de mailserver
- De voorkeur gaat uit naar een wisselbaar IP-adres voor de mailserver die de emails voor de mailinglist verzend

Verwachtingen van de gebruikers

Alle verwachtingen van de gebruikers worden als basiseisen behandelt:

- De gebruiker moet zonder voorkennis van de applicatie inzicht kunnen krijgen in de werking ervan
- De gebruiker moet met een zo min mogelijk aantal handelingen de processen binnen de applicatie kunnen doorlopen
- De handelingen van een proces moeten voor de gebruiker elkaar logisch opvolgen of de logica ervan moet aangegeven worden

Bijlage 3: Functioneel ontwerp

Functioneel ontwerp:
Mailingapllicatie Mediaweb

Versie 2.3
Datum: 20-6-2006

Schrijver/student: Jeroen Cramer HHS 20021998
Opdrachtgever/mentor: Peter van Marwijk

Inleiding

Dit document heeft dezelfde functie als het functioneel ontwerp maar is niet bedoeld voor de klant maar voor projectleden die geen direct contact hebben gehad met de opdrachtgever of projectleden die meer informatie wensen om hun verdere ontwerp op te baseren.

Het document bevat naast een navigatieschema en een beschrijving van de verschillende onderdelen van het product een aantal beschrijvende stukken die de relaties en eigenschappen van de betrokken objecten nader beschrijven.

1. Beschrijving mailingapplicatie

In dit hoofdstuk zal de mailingapplicatie zoals deze tijdens het proces van het functioneel ontwerp maken werd beschreven in kaart worden gebracht voor het ontwerpende team en voor anderen die de opdracht later zullen oppakken.
Binnen het project hoort dit voorafgaand aan het functioneel ontwerp te geschieden voor het beste effect. Onduidelijkheden en tegenstrijdigheden in het model zijn snel en makkelijk op te sporen evenals het generen van eventuele oplossingen vanwege de beschrijvende aard van de gebruikte methode.

Naast deze inleidende paragraaf bestaat dit hoofdstuk uit beschrijvende afbeeldingen die het model van de mailingapplicatie voorstellen. Hiervoor is er gebruik gemaakt van de beschrijvingstaal Unified Modelling Language (UML).
Hoewel de diagrammen en afbeeldingen redelijk te begrijpen zijn zonder kennis van UML zullen veel nuances gemist worden, daarom wordt er aangeraden (algemene) kennis op te doen van UML alvorens dit hoofdstuk te lezen.

Achtereenvolgens worden de volgende punten behandelt:

- Een klassendiagram van de mailingapplicatie
- Use cases van het gebruik van de applicatie
- Toestandsdiagrammen van de mailingapplicatie

1.1 Klassendiagram Mailingapplicatie

Het klassendiagram beschrijft de mailingapplicatie door het indelen van de onderdelen van de mailingapplicatie in klassen aan de hand van hun overeenkomende kenmerken en gedragingen.

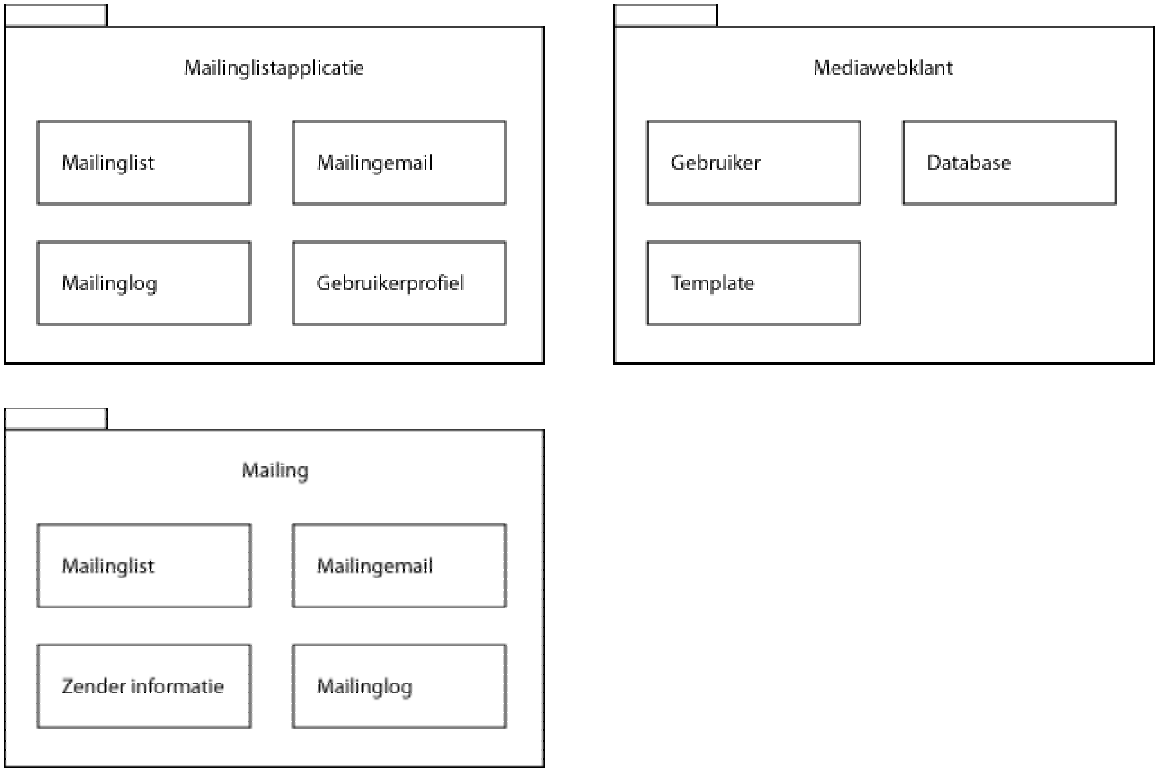


Fig. 1.1 packages van het klassendiagram mailingapplicatie

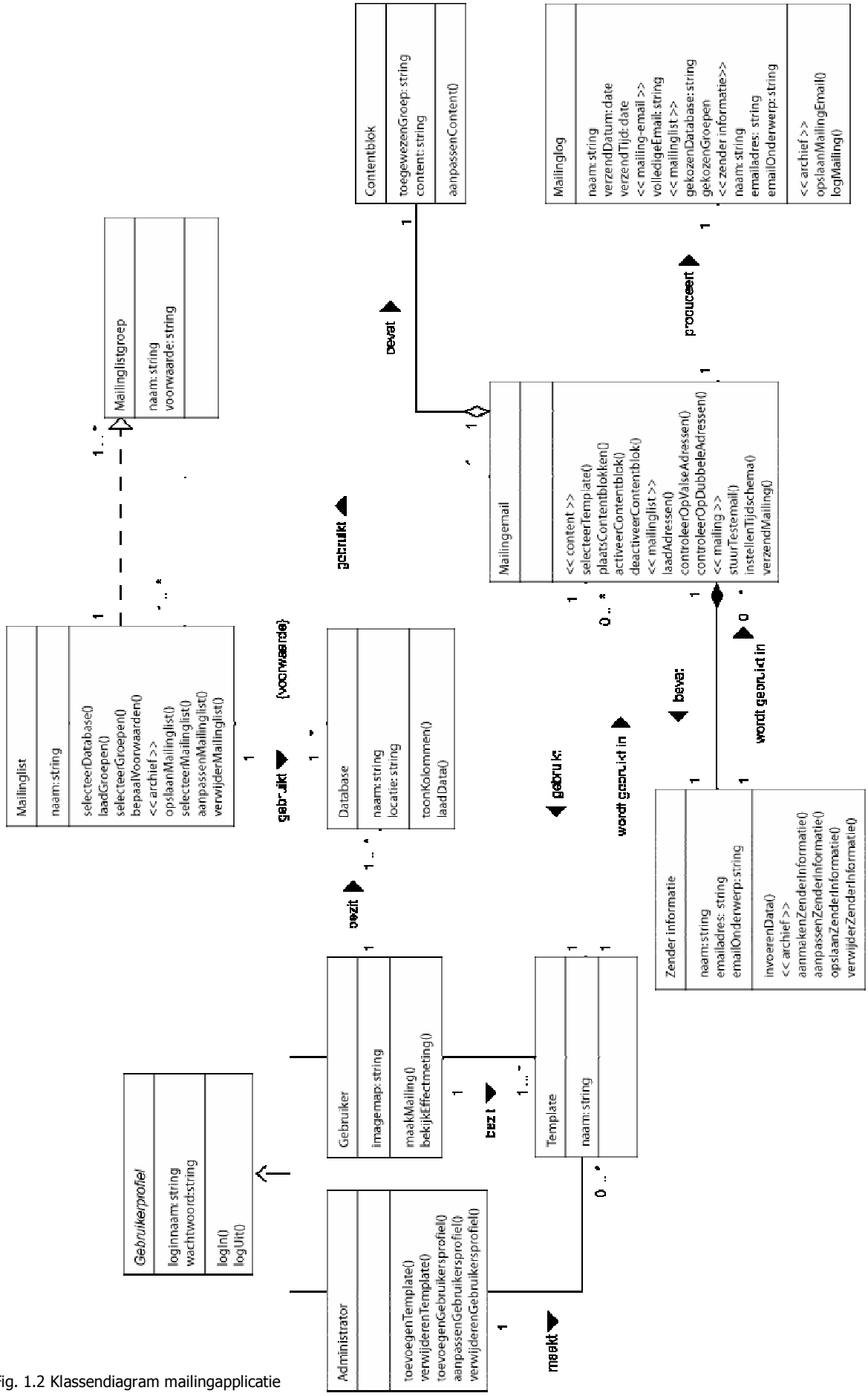


Fig. 1.2 Klassendiagram mailingapplicatie

1.2 Use Cases

In dit gedeelte van het hoofdstuk worden het klassendiagram uitgebreid met de use cases. Use cases beschrijven de handelingen tussen een acteur en het systeem, in dit geval de mailingapplicatie. Een acteur is veelal een gebruiker maar kan ook een ander systeem of een stuk hardware zijn.

1.2.1 Beschrijving omgeving mailingapplicatie

Hieronder worden de onderlinge relaties tussen de acteurs beschreven in de acteur hiërarchie, alle acteurs binnen de use cases zijn erin vertegenwoordigt. Ernaast is de use case van de mailingapplicatie in zijn meest algemene vorm te vinden, enkel de belangrijkste handelingen worden in het hoog-niveau use case behandelt. De meer gedetailleerde use cases zijn in het volgende stuk te vinden.

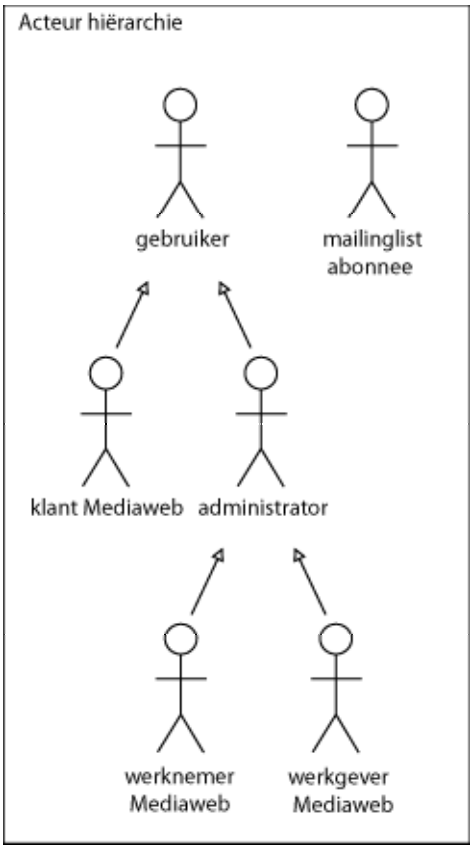


Fig. 1.3 Relaties tussen de betrokkenen

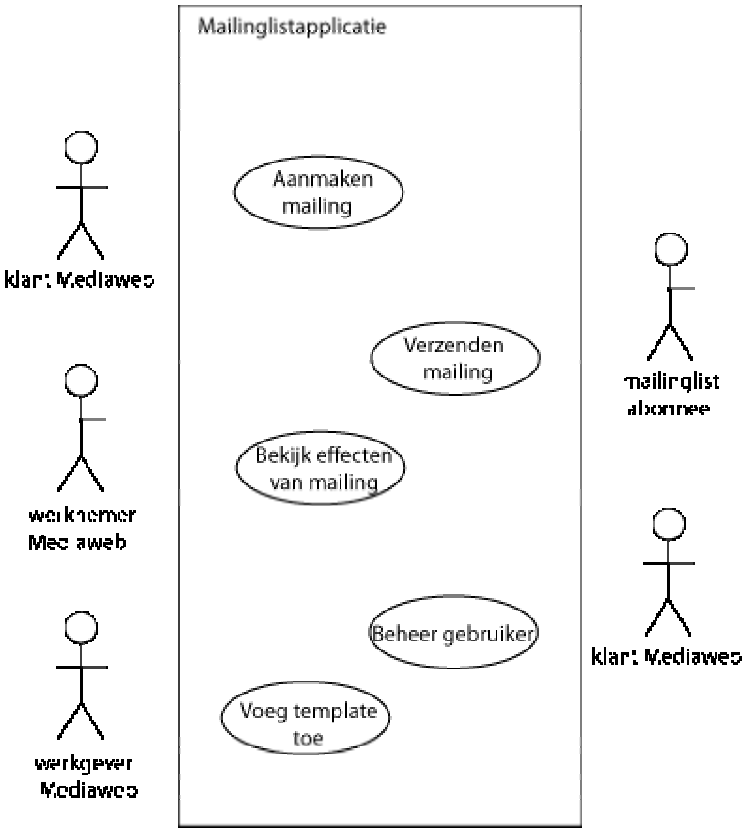


Fig 1.4 hoog-niveau use case, mailingapplicatie

1.2.2 laag niveau use cases

Zoals in het vorige stuk al vermeld is zijn hier de meer gedetailleerde use cases te vinden die de interactie met de mailingapplicatie beschrijven. De packegeklassen met relevante interactie uit 1.1. hebben hier hun acties toegewezen gekregen.

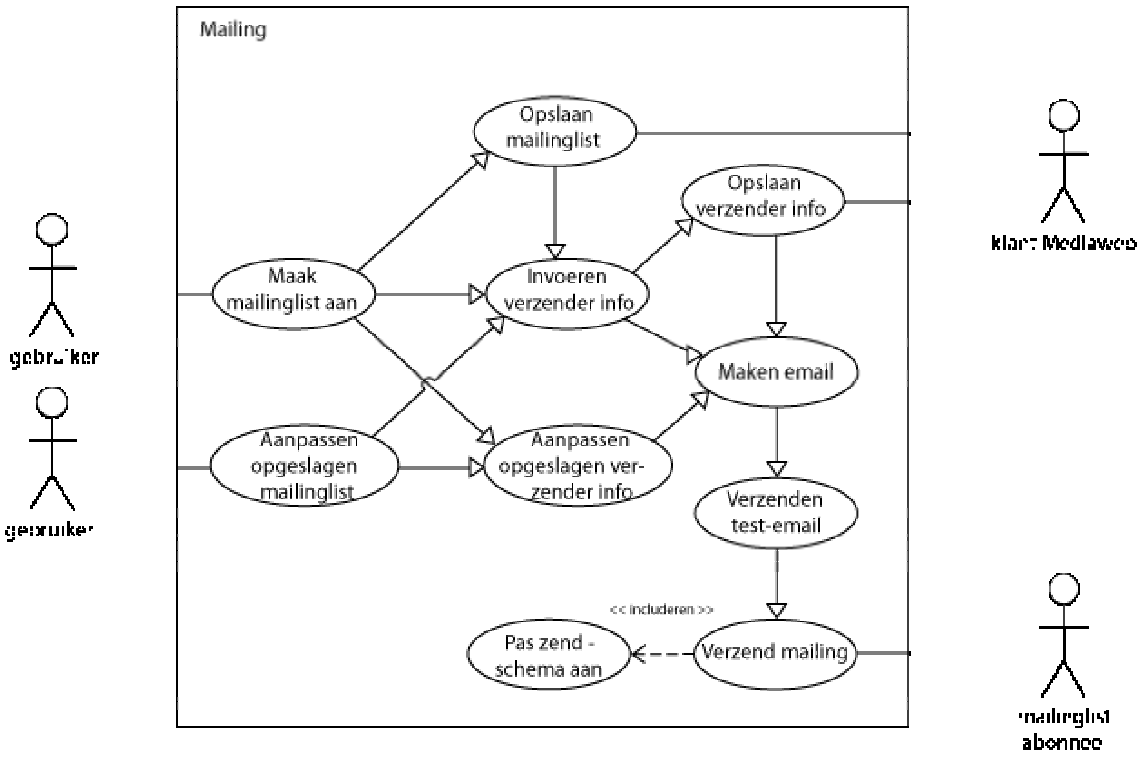


Fig. 1.5 use case mailing

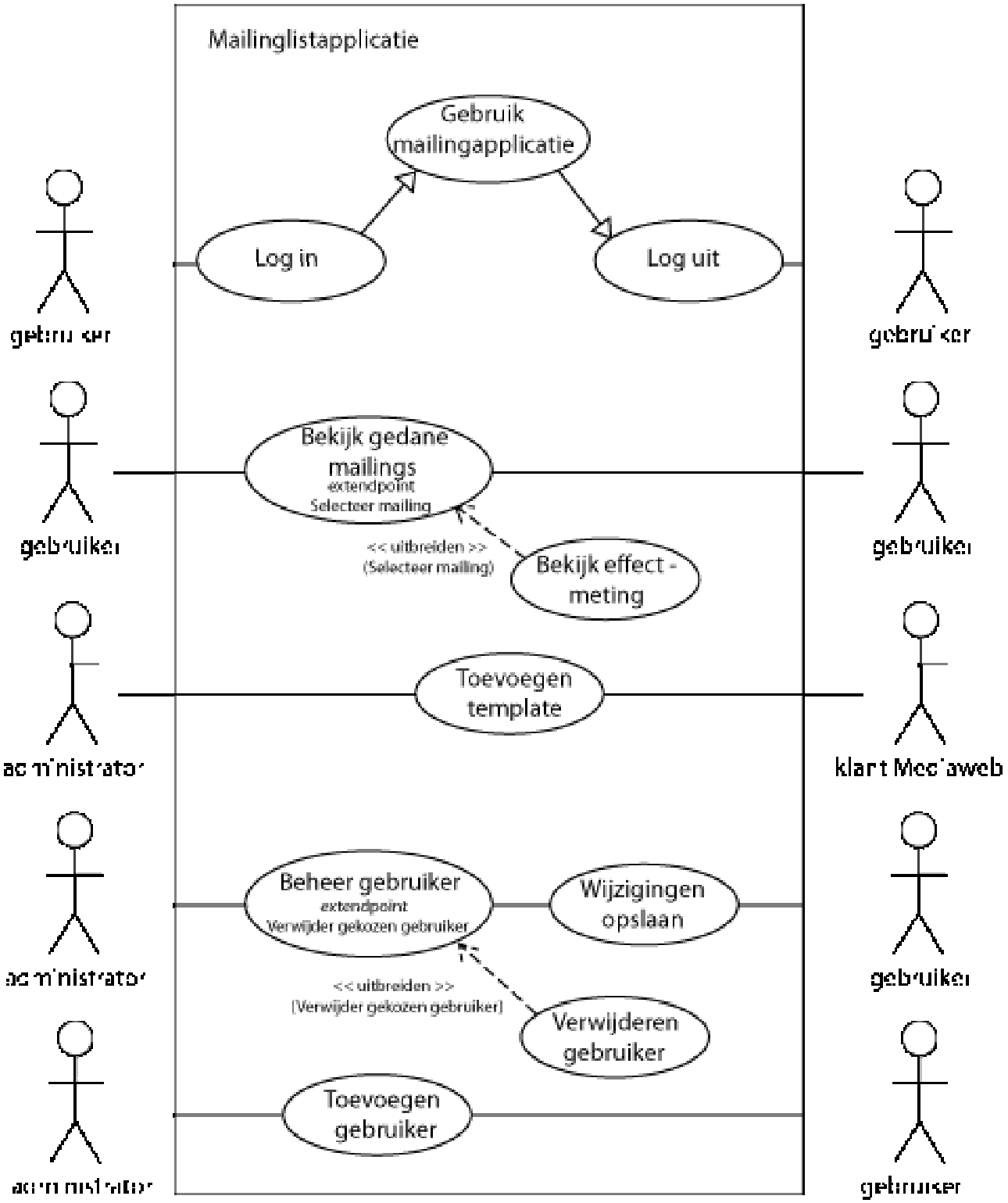


Fig 1.6 use cases mailingapplicatie

1.3 Toestandsdiagrammen

In de voorgaande hoofdstukken zijn de applicatie en de interactie met de applicatie beschreven, voor een compleet beeld dient nog de mechaniek binnen de applicatie beschreven te worden. De toestandsdiagrammen beschrijven de toestanden waarin een systeem zich kunnen bevinden, deze kunnen zonder stimulans van een acteur plaatsvinden en zijn daarom een goede aanvulling.

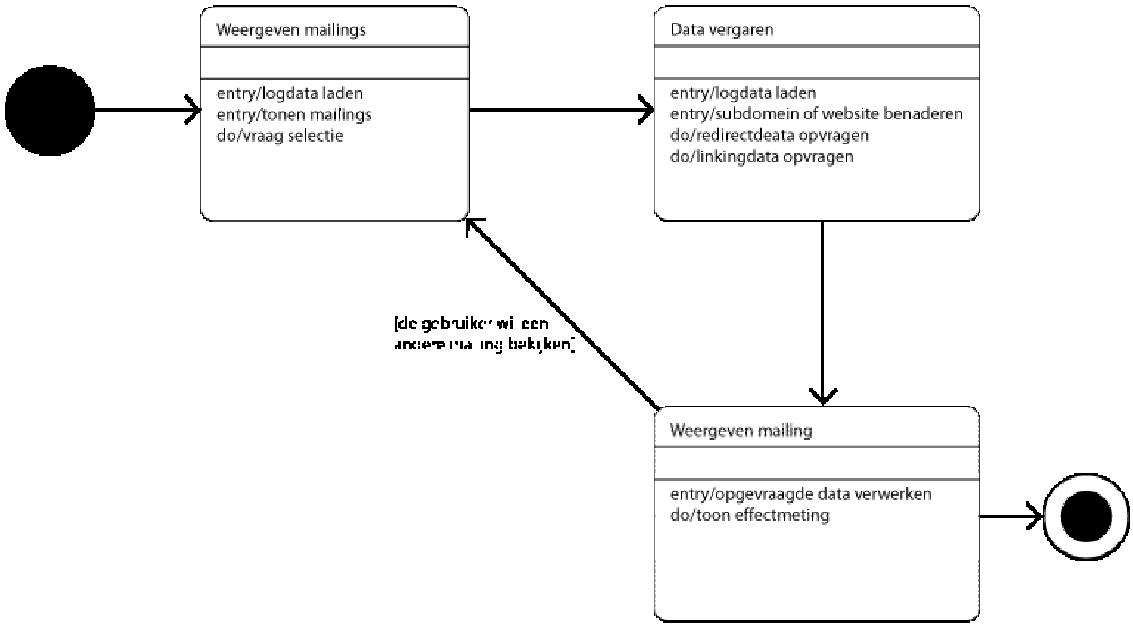


Fig. 1.7 toestandsdiagram effectmeting

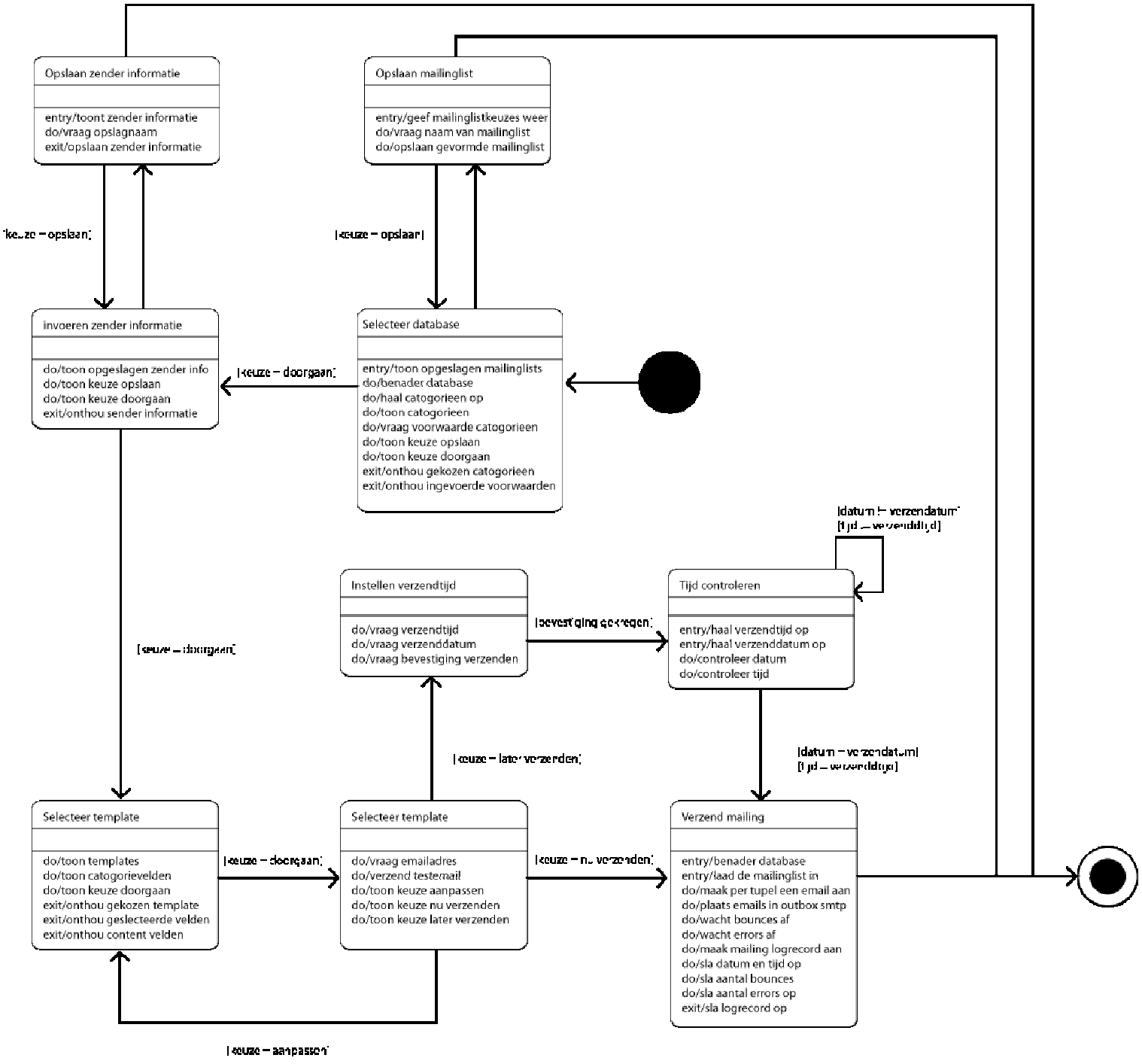


Fig. 1.8 Toestandsdiagram mailing

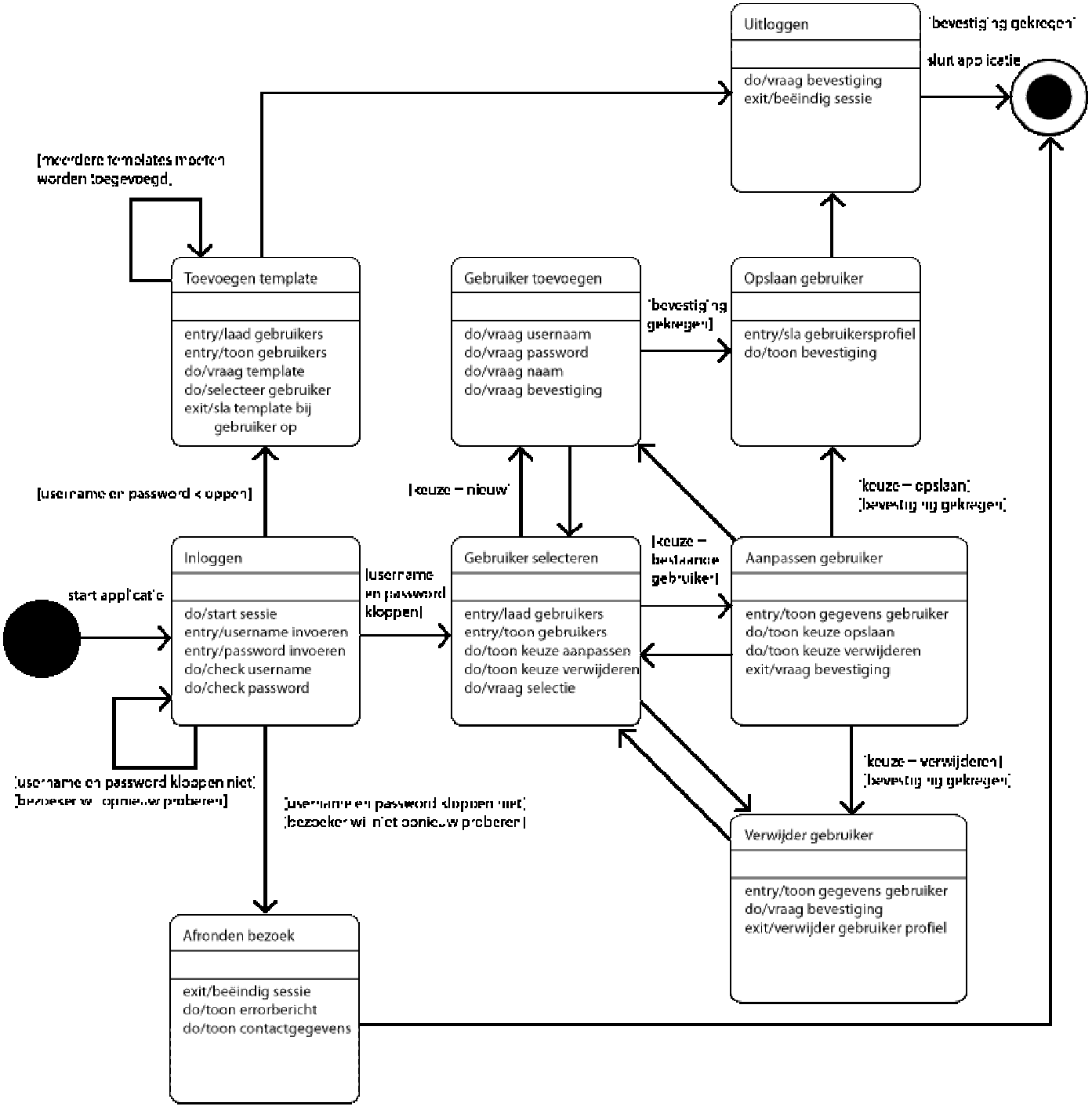


Fig. 1.9 toestandsdiagram gebruikersbeheer

2. Functioneel Ontwerp

Dit is het functioneel ontwerp zoals deze voor een klant zou aangeleverd worden. In dit geval is het een interne opdracht maar dient het nog steeds hetzelfde doel: Het schetsen van een beeld van de uitwerking van de opdracht en het dienen als basis voor het daadwerkelijke ontwerp van het product.

Dit hoofdstuk bevat als eerste een navigatieschema met daaropvolgend een beschrijving van het navigatieschema aan de hand van de verschillende onderdelen. Als laatste zijn de twee type gebruikers nader toegelicht om het beeld compleet te maken.

2.1 Navigatieschema mailingapplicatie

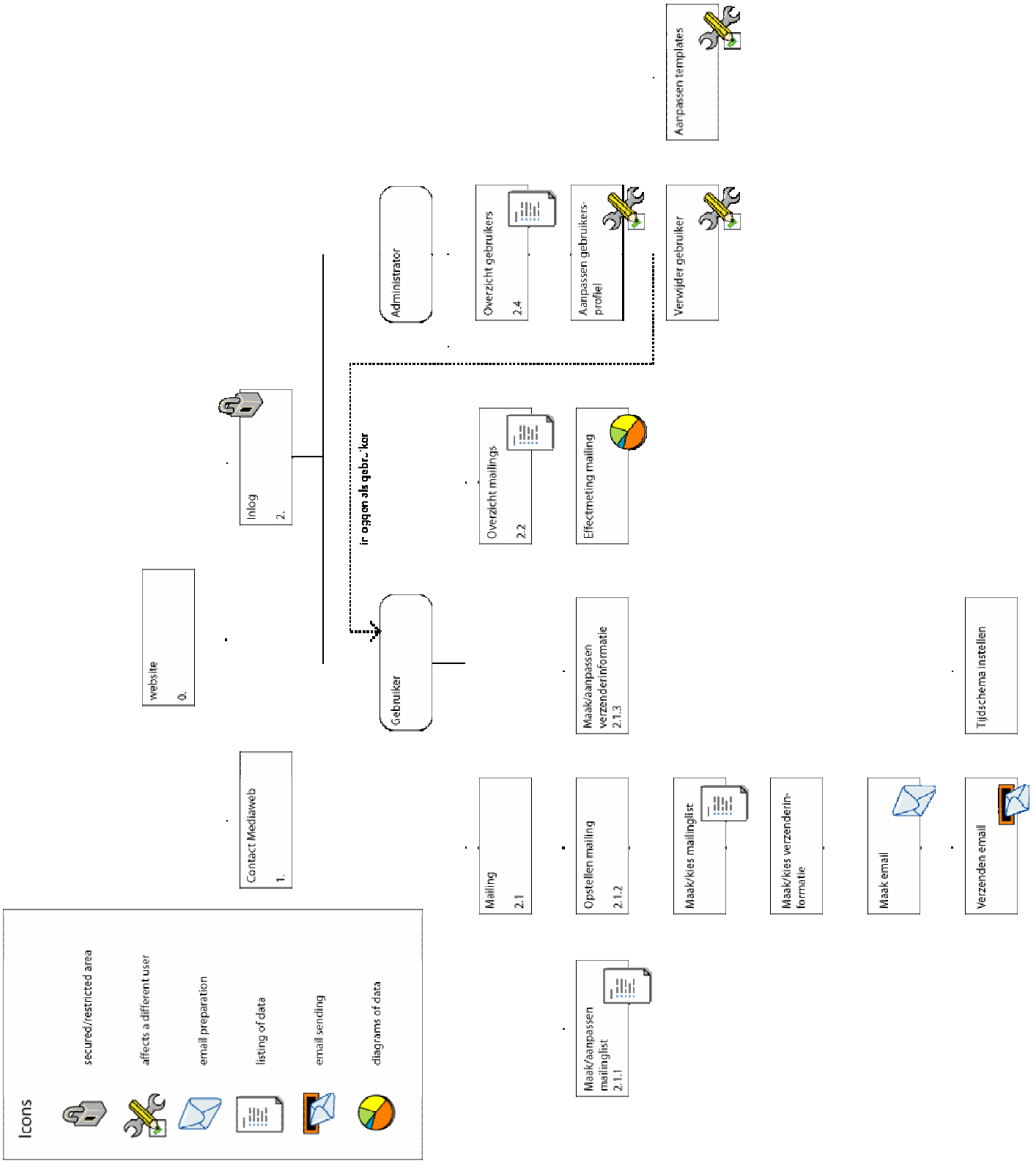


Fig. 2.1. Navigatieschema mailinglistapplicatie

2.2 Beschrijving mailingapplicatie onderdelen

0. Website

De website is de ingang van de applicatie, door in te loggen kan de mailingapplicatie worden gebruikt. Lukt dat niet dan kan er via de contactpagina contact opgenomen worden met Mediaweb.

1. Contact Mediaweb

Om de mogelijkheid tot snel en eenvoudig contact opnemen met Mediaweb is er een contactpagina toegevoegd. Wanneer het niet lukt om in te loggen of wanneer een gebruiker vragen heeft dan kan deze door middel van de contactgegevens op deze pagina contact opnemen met Mediaweb.

2. Log in

Via het internet kan de applicatie met een webbrowser benaderd en gebruikt worden, een gebruiker dient in te loggen met behulp van de gebruikersnaam en het bijbehorende wachtwoord om gebruik te kunnen maken van de applicatie.

2.1 Mailing

Door de mailingpagina te selecteren kan de gebruiker de keuze maken om een mailing te maken of om voorbereidingen te treffen voor mailings die in de toekomst zullen plaatsvinden. Voorbereidingen treffen kan door het maken of aanpassen van een mailinglist of de verzenderinformatie.

2.1.1 Maak mailinglist

Via de mailingpagina kan de gebruiker op deze pagina de mailinglist die voor een mailing wordt gebruikt aanmaken of een bestaande mailinglist aanpassen. Door de database te benaderen van de website waar de inschrijvingen voor de mailinglist plaatsvindt wordt de data voor een mailinglist opgehaald.
Uit deze database worden de categorieën gehaald die in aanmerking komen om als mailinglist of als subgroep in de mailinglist te dienen. Na de selectie van categorieën en de daaraan verbonden voorwaarden kan de mailinglist worden opgeslagen. De mailinglist kan zo op een later tijdstip herhaaldelijk gebruikt worden voor het doen van een mailing.

2.1.2 Maak mailing

Deze pagina start het proces om een email naar een gekozen mailinglist te verzenden. Het proces is te onderbreken na het aanmaken van een mailinglist en na het aanmaken van de verzenderinformatie. Hierna moet de gebruiker doorgaan totdat de email gemaakt en verzonden is of de gebruiker moet ervoor kiezen af te zien van het maken van een mailing. Het proces is op te delen in een aantal stappen, ze worden de chronologische volgorde behandelt:

- Maak/kies mailinglist, hetzelfde als bij *2.1.1 Maak mailinglist.* Na het opslaan kan het proces worden gestopt.
- Maak/kies verzenderinformatie, hetzelfde als bij *2.1.3 Maak verzenderinformatie.* Na het opslaan kan het proces worden gestopt.
- Maak email, deze stap bevat een aantal acties. Als eerste het kiezen van een template uit de verzameling van templates van de gebruiker. Als tweede het selecteren en invullen van de contentgroepen, de beschikbare contentgroepen zijn de gekozen categorieën van de mailinglist en een standaardveld voor alle lezers. Als laatste kan een testemail worden verzonden ter controle van de opmaak.
- Verzenden email, na bevestiging te hebben gevraagd zal de email aangemaakt en verzonden worden naar de adressen die in de mailinglist aanwezig zijn. Bij voltooiing van de mailing wordt deze gelogd.
- Tijdschema instellen, de gebruiker heeft de mogelijkheid om in plaats van direct de mailing te verzenden deze op een ander tijdstip te verzenden. Na het invoeren van de datum en tijd wordt er wederom gevraagd om bevestiging waarna de email en mailing in de wacht komen te staan.

2.1.3 Maak verzenderinformatie

De gebruiker kan de verzenderinformatie die elke email zal bevatten op deze pagina reeds prepareren of bestaande verzenderinformatie aanpassen. De verzenderinformatie bevat de relevante gegevens over de afzender, een correspondentie-emailadres en het onderwerp van de email. Na het invoeren van de gegevens kunnen deze opgeslagen worden onder een naam om op een later tijdstip te gebruiken in een mailing.

2.2 Overzicht mailings

Van alle mailings die hebben plaatsgevonden wordt een log bijgehouden en die zijn op deze pagina te zien en te selecteren. Bij een selectie is het mogelijk om een effectmeting te tonen, dit is afhankelijk van het feit of de klant dit wil en of de mogelijkheid tot een effectmeting zijn geïmplementeerd (denk hierbij aan redirects).

2.3 Overzicht gebruikers

Vanuit deze pagina zijn alle gebruikers te beheren, de meest voorkomende handelingen zullen het toevoegen van een template bij een gebruiker en het inloggen als een gebruiker om een mailing uit te voeren zijn.
Dit gedeelte begint met een lijst van de gebruikers op alfabetische volgorde waarna een selectie kan plaatsvinden. Wanneer een individuele gebruiker is geselecteerd wordt een de pagina met de gegevens van het gekozen gebruikersprofiel geladen. Hier kunnen de gegevens worden aangepast, een template worden toegevoegd, de gebruiker worden verwijderd of als een gebruiker worden ingelogd.

Het toevoegen een template wordt gedaan door de template te selecteren en up te loaden naar een daarvoor bestemde map van het gebruikersprofiel. Omdat de templates enkel door Mediaweb gemaakt mogen worden is deze functie opgenomen onder de administrator rol, een rol die alleen leden van Media zullen bezitten.

Het inloggen als een specifieke gebruiker geeft de administrator dezelfde rechten als de gebruiker zou hebben. Omdat niet alle gebruikers de tijd, de behoefte of de technische vaardigheid bezitten om een mailing te kunnen verzenden is de mogelijkheid ingebouwd om dit door leden van Mediaweb te laten doen.

2.3 Gebruikersprofielen

Administrator

Het administratorprofiel bevat:

- Inloggegevens van de administrator
- Gebruikersprofielen van de gebruikers

Het doel van de administrator is het beheren van de gebruikers en de applicatie zelf, alleen medewerkers van Mediaweb kunnen deze positie bezitten. Het beheer van de gebruikers wordt binnen de applicatie gedaan, het beheer van de applicatie zal buiten de applicatie gebeuren door het aanpassen en toevoegen van modules en eventueel de code van individuele pagina's.

Gebruiker

Het gebruikersprofiel bevat:

- Inloggegevens van de gebruiker
- Mailinglist(s)
- Informatie v.d. verzender v.e. mailing
- Templates (sjablonen voor emails)
- Gegevens over oude mailings

De mailingapplicatie wordt een extra service die aan klanten van mediaweb wordt geleverd. De klant kan als gebruiker de mailingapplicatie benutten om mailings te verzenden groepen van mensen die zich bij de gebruiker hebben aangemeld. Dit dient gedaan te zijn zoals beschreven in de richtlijnen van MAPS (Mail Abuse Prevention System) http://www.mail-abuse.com/an_listmgntgdlines.html

De gebruiker kan door middel van een aantal stappen te doorlopen een mailing samenstellen en verzenden naar de (sub)groep(en) van een mailinglist. De mailinglists worden gemaakt aan de hand van de beschikbaar gestelde database met abbonnees. De emails van een mailing worden gemaakt aan de hand van templates. Templates worden voor de klant gemaakt en een gebruiker is niet in staat zelf nieuwe templates in te voeren.

Bijlage 4: Grafisch ontwerp

Grafisch ontwerp:
Mailingapllicatie Mediaweb

Versie 1.0
Datum: 6-7-2006

Schrijver/student: Jeroen Cramer HHS 20021998
Opdrachtgever/mentor: Peter van Marwijk

Inleiding

Dit is het grafisch ontwerp van de mailingapplicatie, het product van de derde fase van het 8 fasenmodel dat binnen het bedrijf Mediaweb Internet Integrators b.v. wordt gebruikt.

De mailingapplicatie zal eventueel als onderdeel van het huidige Content Management System gebruikt gaan worden. Om hiermee rekening te houden wordt de huisstijl overgenomen. Van de huisstijl van het CMS is er geen documentatie en om deze reden zal dit document als Grafisch Ontwerp dienen voor de mailingapplicatie en als een gedocumenteerde versie van de huisstijl van het CMS.

Binnen dit document vindt u de volgende zaken:

- Huisstijl van het CMS, hier worden de vormgeving en vormgegeven trends beschreven van het Content Management System
- Grafisch Ontwerp, dit zijn schetsen met uitleg van de belangrijkste onderdelen van de mailingapplicatie

1. Huisstijl Content Management System

Omdat de Mailingapplicatie binnen het bestaande Content Management System (CMS) van Mediaweb moet geplaatst kunnen worden zal het uiterlijk van de Mailingapplicatie dus overeen moeten komen met de huisstijl van het CMS. Om deze reden wordt de bestaande huisstijl zoveel mogelijk overgenomen.
Om dit te kunnen moet de huisstijl eerst gedocumenteerd worden, wat in dit hoofdstuk wordt gedaan.

Lettertype

Binnen het gehele CMS wordt het lettertype Tahoma gebruikt met de lettertypes Helvetica en Arial om op terug te vallen. In de tabel wordt uitgegaan van het lettertype Tahoma, door binnen de kolom Lettertype Tahoma te vervangen met Helvetica of Arial kunnen de andere configuraties verkregen worden.

	Lettertype	Grootte	Kleur	Kleurcode	Opmerking
Titel	Tahoma bold	12	Zwart	#000000	
Kop	Tahoma bold	9	Zwart	#000000	
Kop (Navigeerbaar)	Tahoma underline	9	Zwart	#000000	lowercase
Text	Tahoma	9	Zwart	#000000	
CMS Navigatiemenu	Verzorgd navigatie tussen de onderdelen van het CMS				
CMS Navigatiemenu	Tahoma bold	9	Zandkleurig	#C5BCB4	
structuurmenu	Verzorgt navigatie tussen de pagina's van de website				
structuurmenu onbezocht	Tahoma bold	9	Zwart	#000000	
structuurmenu bezocht	Tahoma bold	9	Zwart	#000000	
structuurmenu cursor erover	Tahoma bold underline	9	Zwart	#000000	
structuurmenu huidige pagina	Tahoma bold underline	9	Zwart	#000000	
structuursub-menu onbezocht	Tahoma	9	Zwart	#000000	
structuursub-menu bezocht	Tahoma	9	Zwart	#000000	
structuursub-menu cursor erover	Tahoma underline	9	Zwart	#000000	wijkt rechts uit
structuursub-menu huidige pagina	Tahoma underline	9	Zwart	#000000	blijft rechts
structuursub-submenu	Hetzelfde als structuursubmenu enkel is de grootte 8 en is er geen uitwijking bij cursor erover en huidige pagina				

Navigatielink	Wordt buiten de navigatiestructuur van het structuurmenu en het navigatiemenu gebruikt				
Navigatielink onbezocht	Tahomu underline	9	Blauw	#0076C1	lowercase
Navigatielink bezocht	Tahomu underline	9	Lichtblauw	#1EA7DB	lowercase
Navigatielink cursor erover	Tahomu underline	9	Zwart	#000000	lowercase
Navigatielink huidige pagina	Tahomu underline	9	Zwart	#000000	lowercase
Websitepagina functies	Hetzelfde als Navigatielink maar dan UPPERCASE				

Kenmerkend woord- en taalgebruik

Binnen het CMS is de gebruikte taal Nederlands, wanneer er geen geschikte woorden in het Nederlands beschikbaar zijn wordt de Engelse taal gebruikt.

Het Content Management System vermijdt het gebruik van veel tekst. Door korte zinnen, handelingsgerichte woorden en iconen moet de gebruiker het CMS besturen. Enkel bij handelingen met betrekking tot websitepagina's zijn beschrijvingen geplaatst, de gespecialiseerde onderdelen van het CMS navigatiemenu hebben geen beschrijvingen.

Beschrijvingen zijn zakelijk en enigszins informeel, ze zijn gericht op het uitleggen van de handeling van de link waaronder de beschrijving geplaatst is. Er wordt geen gebruik gemaakt van persoonlijke voornaamwoorden.
De opbouw is één enkele zin die handeling beschrijft of een korte paragraaf met uitleg en informatie. Bij een uitgebreide uitleg worden meer zinnen besteed aan het uitleggen van de handeling waarna de gevolgen van de handeling worden beschreven.

Een titel wordt bovenaan de pagina getoond met ervoor een / - teken (forward slash) en heeft dezelfde naam als het bijbehorende CMS navigatie menu-item. In het geval van een website menu-item wordt eerst de naam van de hoofdcategorie getoond gevolgd door de pagina zelf, gescheiden door een forward slash.
Koppen in de pagina's zijn zelfstandig naamwoorden die uitvoerbare acties en/of brokken informatie samenvoegen (voorbeelden zijn "functies" en "informatie").

Bij het uitvoeren van acties worden klikbare links gebruikt, deze zijn werkwoorden in meervoudsvorm die geassocieerd worden met de actie. Bijvoorbeeld "verwijderen" voor het verwijderen van een webshop- of nieuwsartikel. Met betrekking tot websitepagina's wordt de eerste persoonsvorm enkelvoud van een werkwoord gebruikt voor het uitvoeren van acties. Bijvoorbeeld "hernoem" voor het hernoemen van de websitepagina.

Indeling CMS

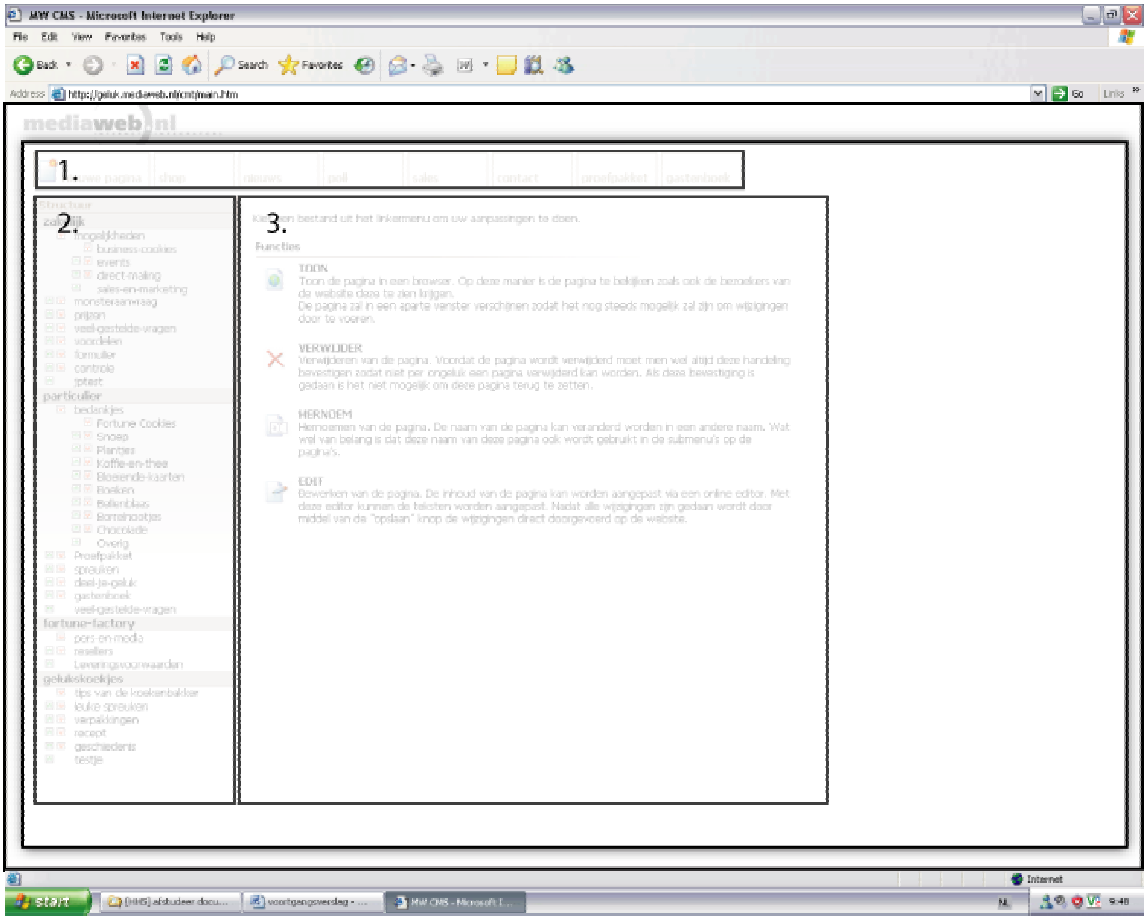


Fig. 1.1 indeling CMS

1. Navigatiemenu van het CMS
2. Navigatiemenu van de website van het CMS
3. Pagina gerelateerde inhoud

Het Content Management System bestaat uit navigatiemenu's voor het CMS (1) en de voor de pagina's van de te beheren website (2). Het website-navigatiemenu opent een tussenmenu in de paginaruimte (3) waarmee de pagina's van de website te beheren zijn.
De CMS pagina's van deze applicatie zijn via het CMS-navigatiemenu (1) bovenin te benaderen en laden een pagina rechts in het midden (3). Het betreft hier aparte onderdelen/pagina's van de website die meer aandacht of functies vereisen zoals een webshop, een enquête of nieuwsberichten dan de gemiddelde pagina van de website.

Wanneer de Mailingapplicatie als module wordt toegevoegd zal in het navigatiemenu van het CMS (1) een extra menu item komen, bij selectie van het menu-item wordt in de paginaruimte (3) de mailingapplicatie gestart.

Uiterlijk CMS

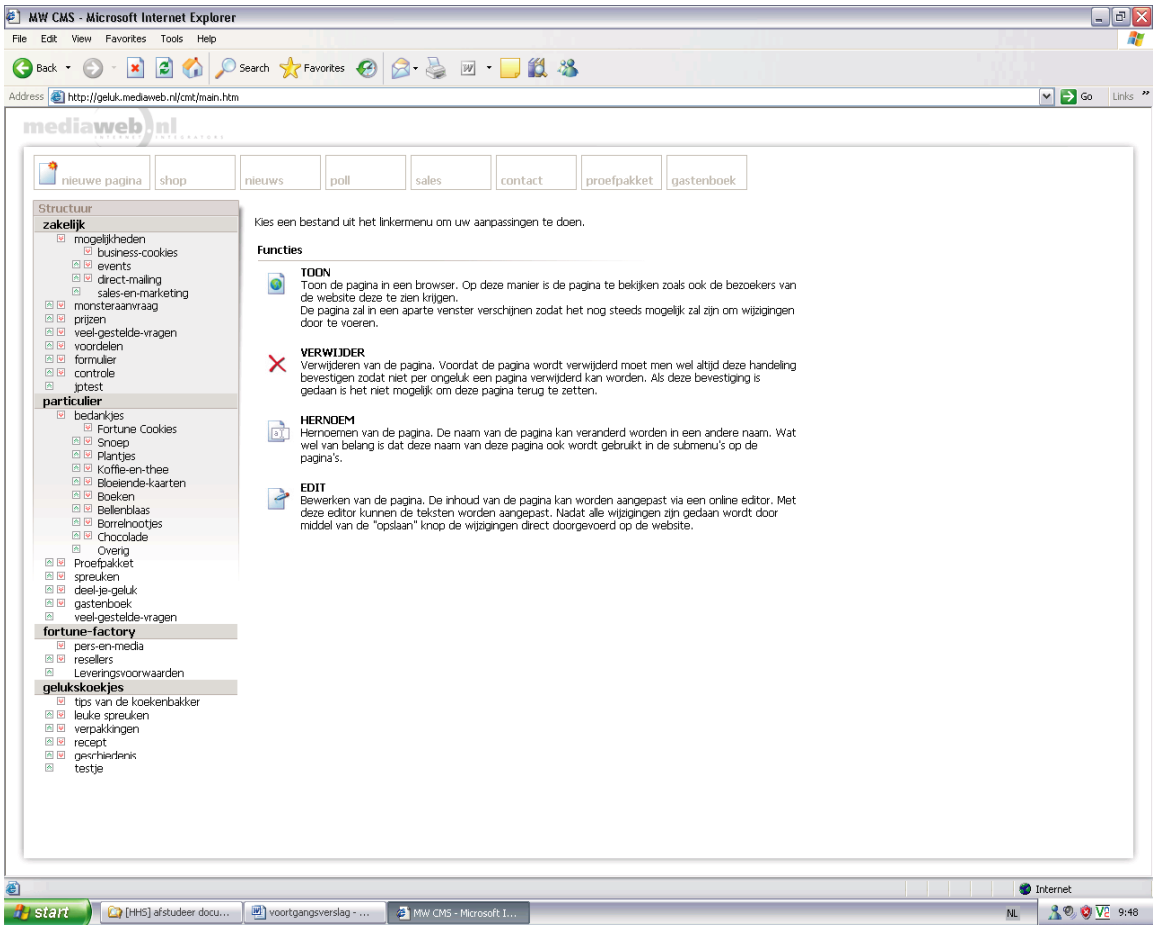


Fig. 1.2 Content Management Systeem

Er zijn vale aarden kleuren gebruikt om de twee menu's weer te geven. Tekst binnen het CMS is voornamelijk zwart en klikbare elementen zijn herkenbaar door de onderstreepte tekst of door in een menu te zijn opgenomen. Klikbare tekstelementen buiten de menu's zijn altijd blauw van kleur, koppen van teksten uitgesloten.

Functies worden vaak ondersteund door een afbeelding, ter uitleg en ter herkenning. Iconen worden altijd rechts ter hoogte van de eerste regel geplaatst of links naast de kop. Ten opzichte van de site zelf worden heldere kleuren gebruikt om de iconen weer te geven.

Gebruikte Iconen:

Een nieuwe pagina toevoegen aan de website, afhankelijk van waar de gebruiker zich bevindt in het websitenavigatiemenu

De bijbehorende pagina een plaats verhogen binnen het websitemenu

De bijbehorende pagina een plaats verlagen binnen het websitemenu

De websitepagina aanpassen

De websitepagina hernoemen

De websitepagina tonen zoals deze op het moment is

De website verwijderen

Het winkelartikel heeft een afbeelding bij de beschrijving

Grafisch Ontwerp

In dit hoofdstuk wordt het uiterlijk van de Mailingapplicatie binnen en buiten het Content Management System van Mediaweb bepaald. Om binnen het ontwerp van het Content Management System te passen moet de mailinglistapplicatie zowel het uiterlijk als de logische opbouw overnemen van het Content Management System, voor zover deze niet tegenstrijdig is met de gestelde wensen en eisen aan de mailingapplicatie.

Veel van wat er in het voorgaande hoofdstuk is besproken zal hier worden toegepast op basis van wat er in het Functioneel ontwerp is besproken. De schetsen in dit hoofdstuk beperken zich tot de belangrijkste onderdelen van de mailingapplicatie.

Algemene indeling mailingapplicatie

Omdat bij de schetsen enkel individuele pagina's verwerkt zullen worden wordt om het overzicht te behouden eerst de algemene indeling van de mailingapplicaties behandelt.

De applicatie moet in twee situaties werken, namelijk losstaand en binnen een CMS gemaakt door Mediaweb. De indeling verschilt niet veel van elkaar, consistentie maakt een programma nu eenmaal navigeerbaar, maar er is een klein verschil.
Beide situaties hebben een menubalk waarmee de functies die bij het verzorgen van een mailing en het bekijken van de resultaten kunnen worden aangeroepen (onderdeel 2 van fig. 2.1 en 2.2). De items in de menubalk van de mailingapplicatie worden weergegeven in de volgorde waarin die het meest gebruikt zullen worden. Eenmaal aangeroepen wordt in het centrale gedeelte de betreffende pagina geladen (onderdeel 3 van fig. 2.1 en 2.2).
Het verschil zit hem in onderdeel 1, waar bij het Content Management dit beperkt blijft tot een knop waarmee er naar de mailingapplicatie genavigeerd wordt zijn bij de mailingapplicatie daar functies te vinden voor de applicatie en is er ruimte om extra modules toe te voegen.

Fig 2.1 mailingapplicatie

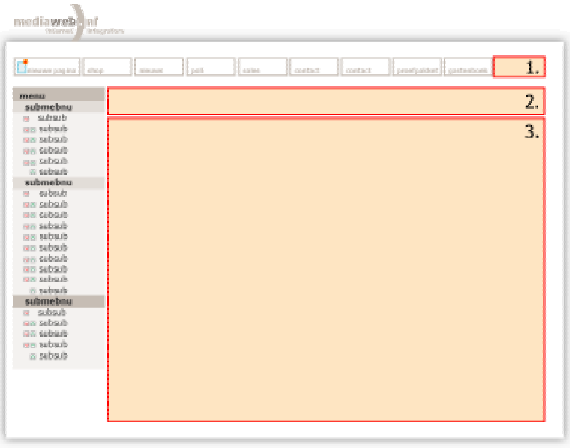


Fig. 2.2 mailingapplicatie binnen het CMS

Schetsen mailingapplicatie

De schetsen die hier gevonden worden zijn enkel van de mailingapplicatie, hoewel er rekening is gehouden met het Content Management System worden daar geen aparte schetsen voor gemaakt. Door de huisstijl van het CMS over te nemen en de indeling af te stemmen (zie indeling mailingapplicatie) is er voldoende rekening gehouden met de mogelijkheid dat de mailingapplicatie in het CMS geplaatst wordt.

Buiten de getoonde schetsen worden in dit hoofdstuk een aantal punten kort besproken die met de vormgeving van de applicatie te maken hebben.

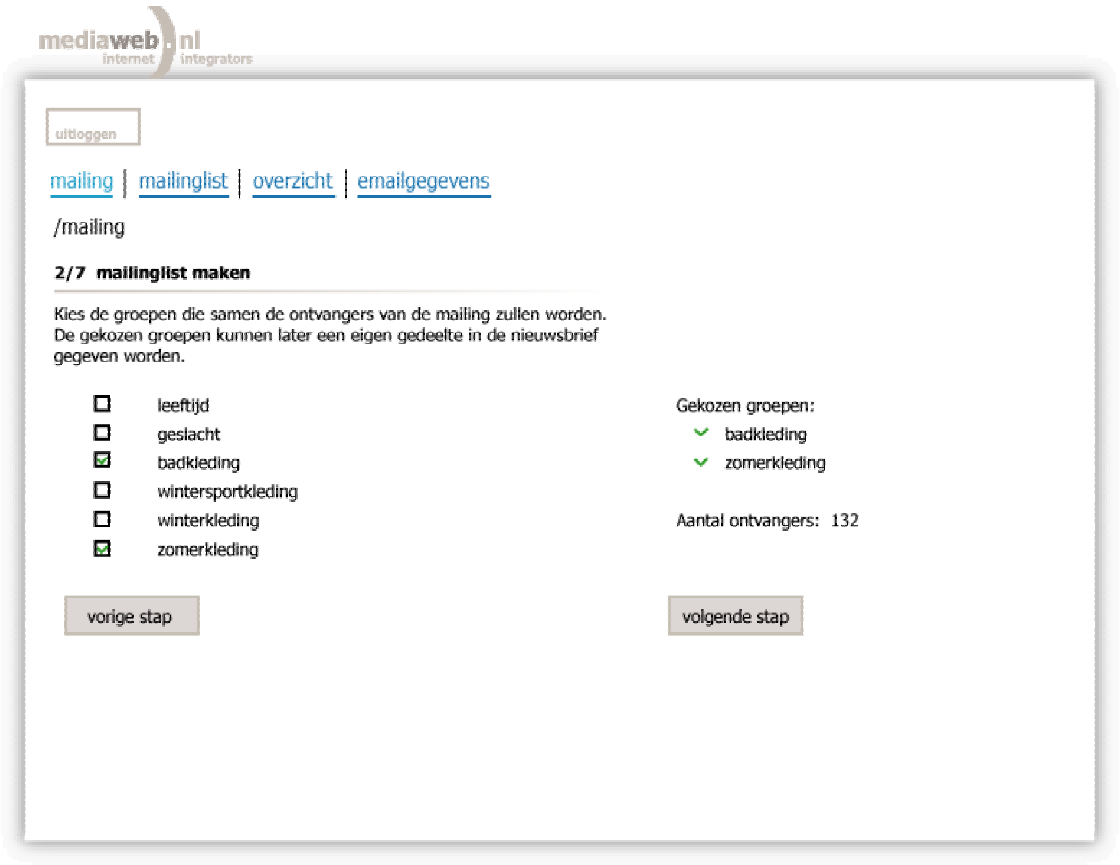


Fig. 2.3 mailingapplicatie, aanmaken v.e. mailinglist

De titel van de pagina is voor de gehele applicatie "/ mailing", dit is voor wanneer de applicatie in het CMS wordt geplaatst of zelf andere modules toegevoegd krijgt. De kop van de pagina verteld wat er op de pagina gedaan zal worden. De positie binnen een proces worden in de kop opgenomen zodat de gebruiker weet hoever hij/zij met het proces is.

Bij elke handeling vanuit de gebruiker is er een reactie of feedback van het systeem om aan te geven dat de gebruiker iets heeft gedaan, dit wisselt per actie. Bij het maken van een keuze wordt een samenvatting of de effecten van de keuze zichtbaar gemaakt aan de linkerzijde van de pagina.

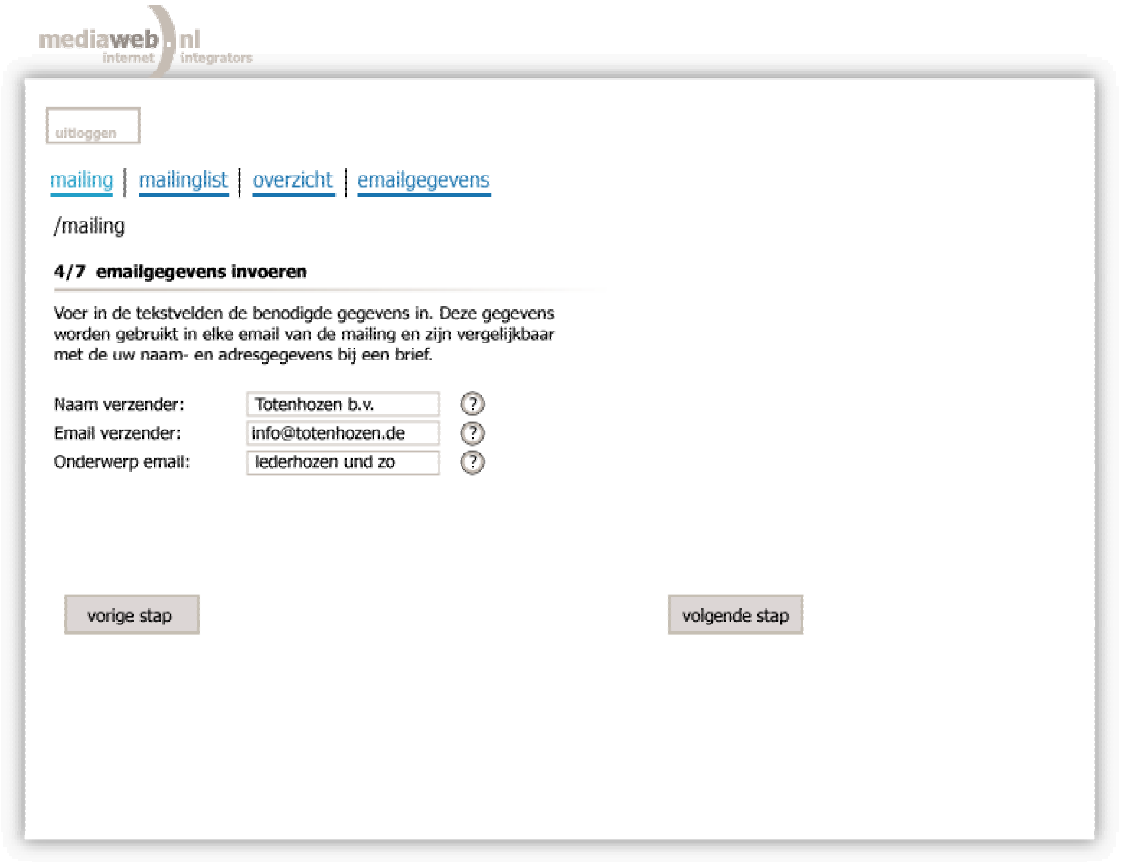


Fig. 2.3 mailingapplicatie, invoeren van de zender informatie

Hoewel gestreefd wordt het te voorkomen kunnen er binnen de applicatie objecten zijn die verwarring kunnen veroorzaken of niet even duidelijk zullen zijn voor iedere gebruiker. Deze objecten krijgen een informatie icoon erbij geplaatst. De gebruiker kan deze gebruiken om een uitleg te zien die bij het object hoort.

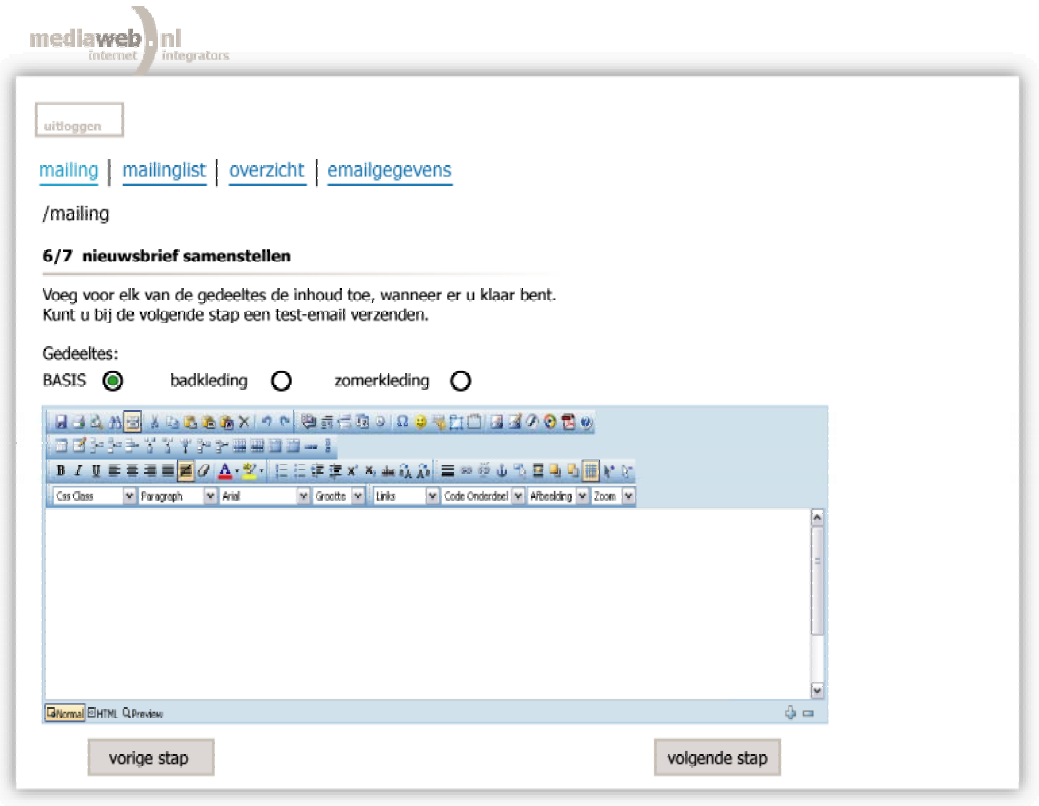


Fig. 2.4 mailingapplicatie, invoeren van inhoud van de nieuwsbrief

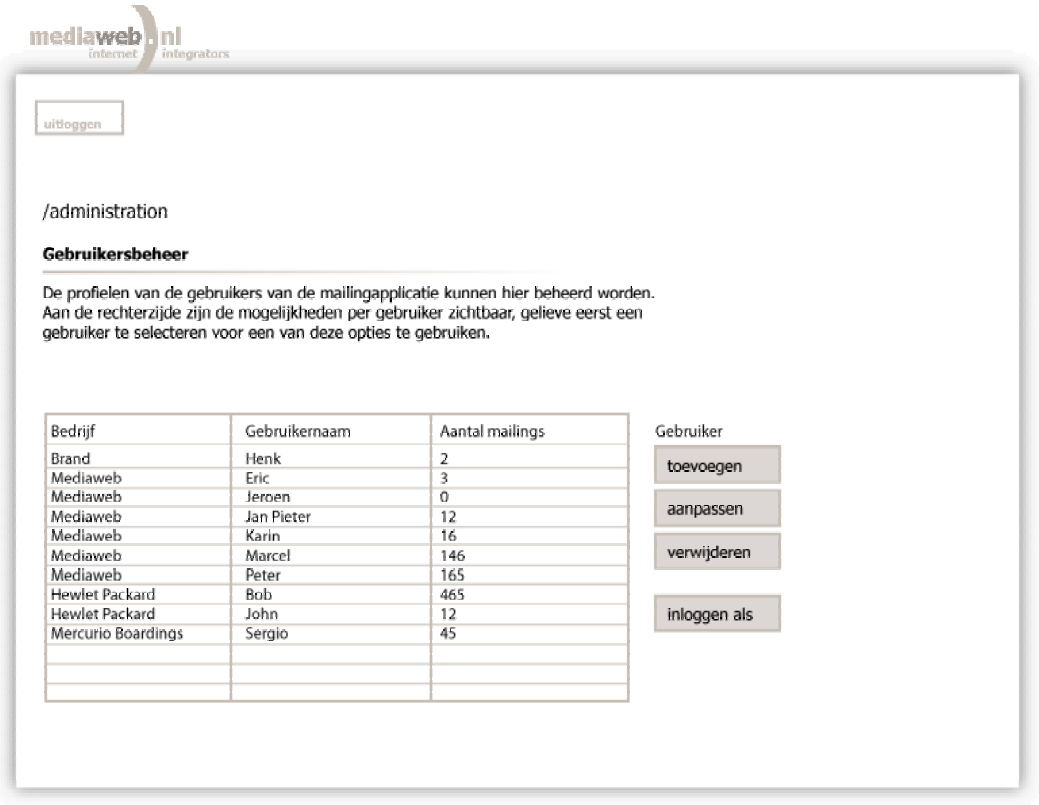


Fig. 2.5 Mailingapplicatie, gebruikersbeheer door administrator

Bijlage 5: Databasemodel

Databasemodel:
Mailingapllicatie Mediaweb

Versie 1.2
Datum: 28-9-2006

Schrijver/student: Jeroen Cramer HHS 20021998
Opdrachtgever/mentor: Peter van Marwijk

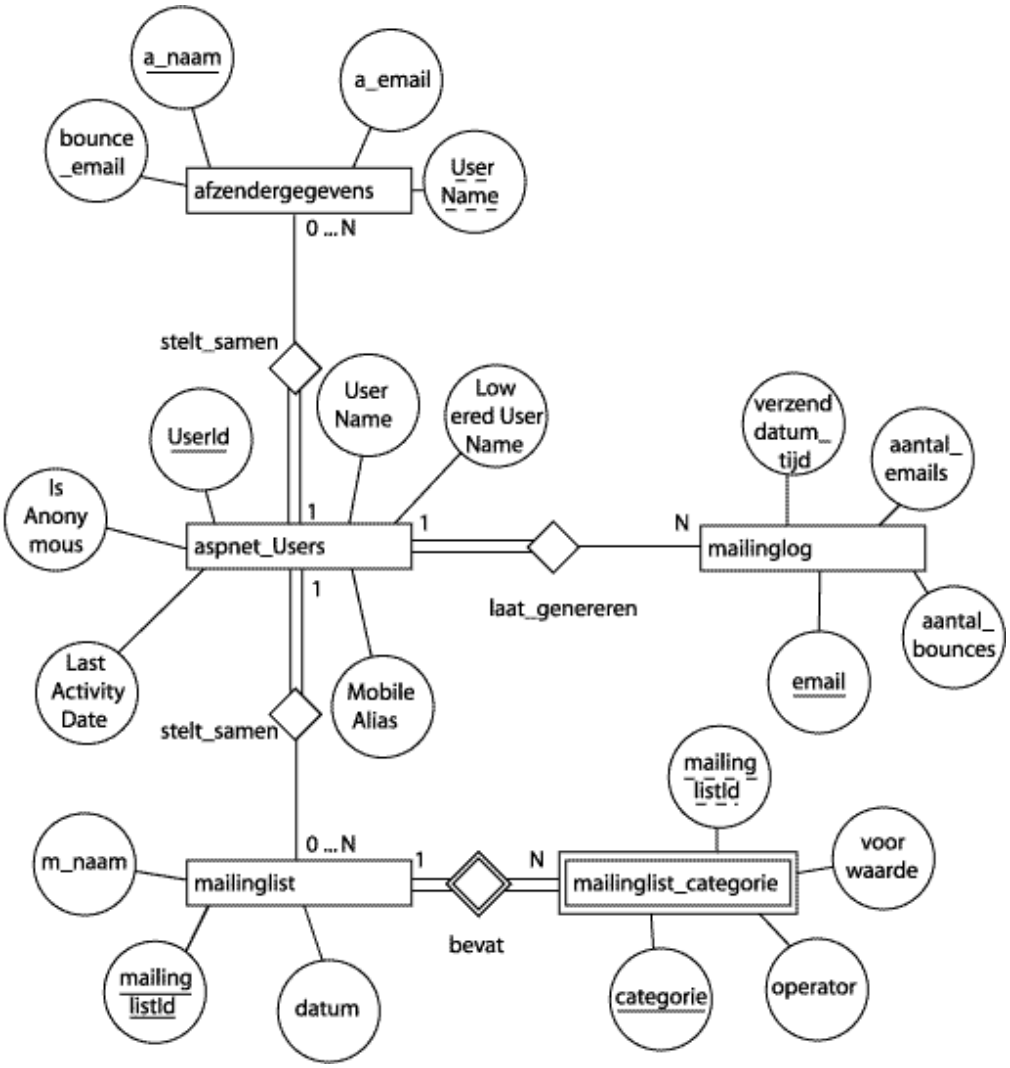
Applicatie database

Buiten de gegevens van de abbonnee's wordt alle informatie die niet van tijdelijke aard is opgeslagen in deze database. De gegevens van de abbonnee's van de mailings komen van externe databases die verschilt per gebruiker.

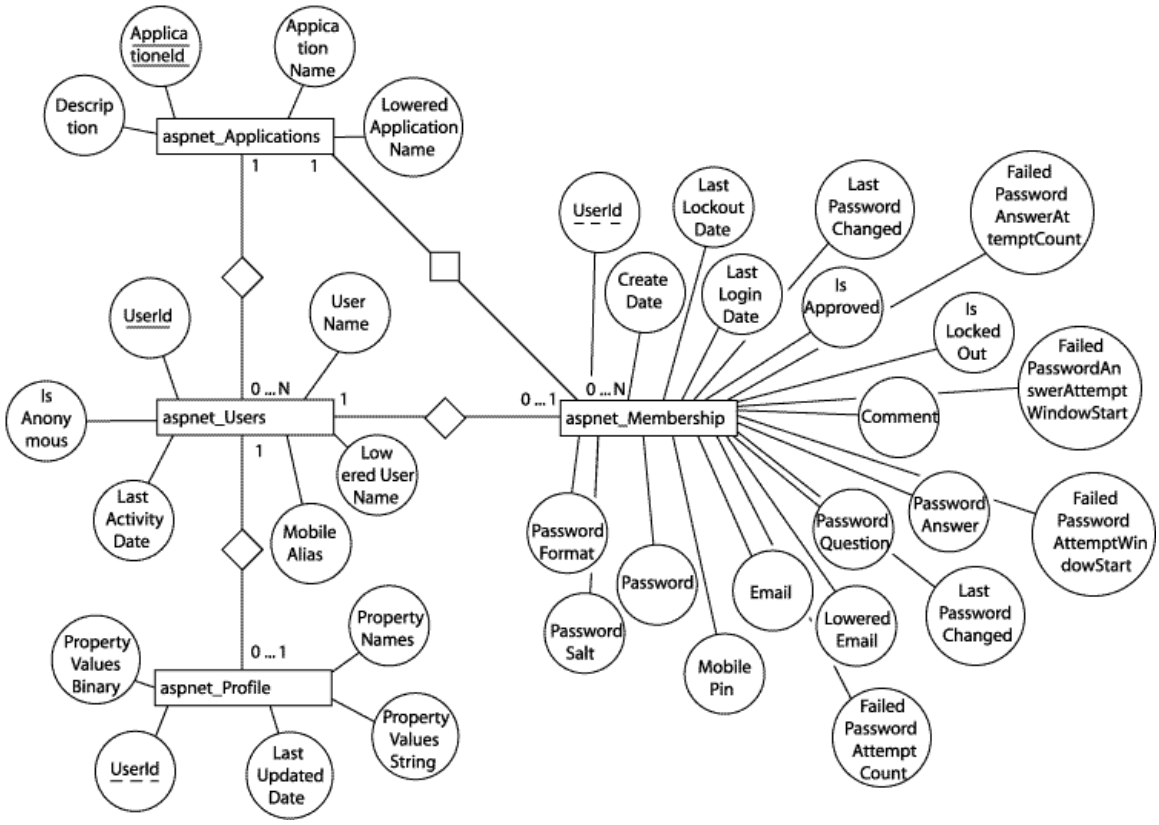
De applicatiedatabase bestaat uit twee delen: de eigen samengestelde tabellen en de tabellen die nodig zijn voor het gebruiken van de membership en profile functionaliteiten binnen de applicatie.
Als eerste wordt het eigen gemaakte ontwerp behandelt en daarna het aanvullende deel dat vanuit databaseconversie is toegevoegd waarvan enkel de tabellen die betrekking hadden op de membership en profile functies binnen ASP.NET zijn gebruikt.
De ongebruikte tabellen zijn in het model voor het compleet maken van het beeld toegevoegd en hebben de kleur grijs. Binnen de applicatie zijn deze tabellen niet verwijderd om het gebruik van de bijbehorende functies bij latere uitbreiding van de applicatie mogelijk te laten.

De verbinding tussen de twee databases (applicatie en externen) gebeurd via de kolom categorie van de tabel mailinglist_categorie van de applicatiedatabase, deze is de Foreign Key die refereert aan de tabel Category van de externe database.

Conceptueel datamodel



Figuur 8. Applicatiedatabase: mailingapplicatie - deel



Figuur 9. Applicatiedatabase: membership & profile - deel

Relationeel Representatiemodel

Mailingapplicatie

mailinglist (mailinglistId, m_naam, *UserName,* datum)
UserName is een foreign key die refereert aan aspnet_Users, null niet toegestaan

mailinglist_categorie (*mailinglistId,* categorie, operator, voorwaarde)
mailinglistId is een foreign key die refereert aan mailinglist, null niet toegestaan

afzendergegevens (a_naam, *UserName*, a_email, bounce_email)
UserName is een foreign key die refereert aan aspnet_Users, null niet toegestaan

mailinglog (*UserName*, email, verzenddatum_tijd, aantal_emails, aantal_bounces)
UserName is een foreign key die refereert aan aspnet_Users, null niet toegestaan

ASP.NET profile en membership

aspnet_Applications (ApplicationeId, AppicationName, LoweredApplicationName, Description)

aspnet_Membership (*ApplicationeId*, *UserId*, Password, PasswordFormat, PasswordSalt, MobilePin, Email, LoweredEmail, PasswordQuestion, PasswordAnswer, IsApproved, IsLockedOut, CreateDate, LastLoginDate, LastPasswordChanged, LastLockoutDate, FailedPasswordAttemptCount, FailedPasswordAttemptWindowStart, FailedPasswordAnswerAttemptCount, FailedPasswordAnswerAttemptWindowStart, Comment)
UserId is een foreign key die refereert aan aspnet_Users, null niet toegestaan
ApplicationeId is een foreign key die refereert aan aspnet_Applications, null niet toegestaan

aspnet_Paths (*ApplicationId*, PathId, Path, LowerPath)
ApplicationeId is een foreign key die refereert aan aspnet_Applications, null niet toegestaan

aspnet_PersonalizationAllUsers (*PathId,* PageSettings, LastUpdatedDate)
PathId is een foreign key die refereert aan aspnet_ Paths, null niet toegestaan

aspnet_PersonalizationPerUser (Id, *PathId*, *UserId*, PageSettings, LastUpdatedDate)
PathId is een foreign key die refereert aan aspnet_ Paths, null toegestaan
UserId is een foreign key die refereert aan aspnet_ Users, null toegestaan

aspnet_Profile (UserId, PropertyNames, PropertyValuesString, PropertyValuesBinary, LastUpdatedDate)
UserId is een foreign key die refereert aan aspnet_Users, null niet toegestaan

aspnet_Roles (*ApplicationId*, RoleId, RoleName, LoweredRoleName, Description)
ApplicationeId is een foreign key die refereert aan aspnet_Applications, null niet toegestaan

aspnet_SchemaVersions (Feature, CompatibleSchemaVersion, IsCurrentVersion)

aspnet_Users (*ApplicationId*, UserId, UserName, LoweredUserName, MobileAlias, IsAnonymous, LastActivityDate,
ApplicationeId is een foreign key die refereert aan aspnet_Applications, null niet toegestaan

aspnet_UsersInRoles (*UserId, RoleId*)
UserId is een foreign key die refereert aan aspnet_ Users, null niet toegestaan
RoleId is een foreign key die refereert aan aspnet_ Roles, null niet toegestaan

aspnet_WebEvent_Events (EventId, EventTimeUtc, EventTime, EventType, EventSequence, EventOccurence, EventCode, EventDetailCode, Message, ApplicationPath, ApplicationVirtualPath, MachineName, RequestUrl, ExceptionType, Details)

Implementatiemodel

Mailingapplicatie

```
create table mailinglist (
        mailinglistId   uniqueidentifier        not null,
        m_naam          varchar(30)             not null,
        UserName        nvarchar(256)           not null,
        datum           datetime                        ,
        primary key (mailinglistId),
        Foreign key(UserName) references aspnet_Users
                on update cascade on delete cascade);

create table mailinglist_categorie (
        mailinglistId   uniqueidentifier        not null,
        categorie       varchar(38)             not null,
        operator        varchar(6)                      ,
        voorwaarde      varchar(30)                     ,
        primary key (mailinglistId, categorie),
        foreign key (mailinglistId) references mailinglist
                on update cascade on delete cascade);

create table afzendergegevens (
        a_naam          varchar(30)             not null,
        UserName        nvarchar(256)           not null,
        a_email         nvarchar(255)           not null,
```

```
        bounce_email nvarchar(255)                ,
        primary key(a_naam, UserName),
        Foreign key(UserName) references aspnet_Users
                on update cascade on delete cascade);

create table mailinglog (
        email               varchar(30)    not null,
        UserName            nvarchar(256) not null,
        verzenddatum_tijd   datetime               ,
        aantal_emails       int                    ,
        aantal_bounces      int                    ,
        primary key (email),
        Foreign key(UserName) references aspnet_Users
                on update cascade on delete cascade);
```

ASP.NET profile en membership

```
create table aspnet_Applications (
        ApplicationeId              uniqueidentifier     not null,
        AppicationName              nvarchar(256)        not null,
        LoweredApplicationName      nvarchar(256)        not null,
        Description                 nvarchar(256)                ,
        primary key (ApplicationeId));

create table aspnet_Membership (
        ApplicationeId                          uniqueidentifier     not null,
        UserId                                  uniqueidentifier     not null,
        Password                                nvarchar(128)        not null,
        PasswordFormat                          int                  not null,
        PasswordSalt                            nvarchar(128)        not null,
        MobilePin                               nvarchar(16)                 ,
        Email                                   nvarchar(256)                ,
        LoweredEmail                            nvarchar(256)                ,
        PasswordQuestion                        nvarchar(256)                ,
        PasswordAnswer                          nvarchar(128)                ,
        IsApproved                              bit                  not null,
        IsLockedOut                             bit                  not null,
        CreateDate                              datetime             not null,
        LastLoginDate                           datetime             not null,
        LastPasswordChanged                     datetime             not null,
        LastLockoutDate                         datetime             not null,
        FailedPasswordAttemptCount              int                  not null,
        FailedPasswordAttemptWindowStart        datetime             not null,
        FailedPasswordAnswerAttemptCount        int                  not null,
        FailedPasswordAnswerAttemptWindowStart  datetime             not null,
        Comment                                 ntext                        ,
        Primary key(UserId),
```

```
        Foreign key(UserId) references aspnet_Users
                on update no action on delete no asction,
        Foreign key(ApplicationeId) references aspnet_Applications
                on update no action on delete no asction);

create table aspnet_Paths (
        ApplicationId           uniqueidentifier        not null,
        PathId                  uniqueidentifier        not null,
        Path                    nvarchar(256)           not null,
        LowerPath               nvarchar(256)           not null,
        Primary key(PathId),
        Foreign key(ApplicationeId) references aspnet_Applications
                on update no action on delete no action);

create table aspnet_PersonalizationAllUsers (
        PathId                  uniqueidentifier        not null,
        PageSettings            image                   not null,
        LastUpdatedDate         datetime                not null,
        Primary key(PathId),
        Foreign key(PathId) references aspnet_Paths
                on update no action on delete no action);

create tableaspnet_PersonalizationPerUser (
        Id                      uniqueidentifier        not null,
        PathId                  uniqueidentifier                ,
        UserId                  uniqueidentifier                ,
        PageSettings            image                   not null,
        LastUpdatedDate         datetime                not null,
        Primary key(Id),
        Foreign key(PathId) references aspnet_Paths
                on update no action on delete no action,
        Foreign key(UserId) references aspnet_Users
                on update no action on delete no action);

create table aspnet_Profile (
        UserId                  uniqueidentifier        not null,
        PropertyNames           ntext(6000)             not null,
        PropertyValuesString    ntext(6000)             not null,
        PropertyValuesBinary    image                   not null,
        LastUpdatedDate         datetime                not null,
        Primary key(UserId),
        Foreign key(UserId) references aspnet_Users
                on update no action on delete no action);

create table aspnet_Roles (
        ApplicationId           uniqueidentifier        not null,
        RoleId                  uniqueidentifier        not null,
        RoleName                nvarchar(256)           not null,
```

```
        LoweredRoleName     nvarchar(256)       not null,
        Description         nvarchar(256)               ,
        Primary key(RoleId),
        Foreign key(ApplicationeId) references aspnet_Applications
                on update no action on delete no action);

create table aspnet_SchemaVersions (
        Feature                     nvarchar(128)       not null,
        CompatibleSchemaVersion     nvarchar(128)       not null,
        IsCurrentVersion            bit                 not null,
        Primary key(Feature, CompatibleSchemaVersion));

create table aspnet_Users (
        ApplicationId       uniqueidentifier    not null,
        UserId              uniqueidentifier    not null,
        UserName            nvarchar(256)       not null,
        LoweredUserName     nvarchar(256)       not null,
        MobileAlias         nvarchar(16)                ,
        IsAnonymous         bit                 not null,
        LastActivityDate    datetime            not null,
        Primary key(UserId),
        Foreign key(ApplicationeId) references aspnet_Applications
                on update no action on delete no action);

create table aspnet_UsersInRoles (
        UserId              uniqueidentifier    not null,
        RoleId              uniqueidentifier    not null,
        Primary key (UserId, RoleId),
        Foreign key(UserId) references aspnet_Users
                on update no action on delete no action,
        Foreign key(RoleId) references aspnet_Roles
                on update no action on delete no action);

create table aspnet_WebEvent_Events (
        EventId             char(32)            not null,
        EventTimeUtc        datetime            not null,
        EventTime           datetime            not null,
        EventType           nvarchar(256)       not null,
        EventSequence       decimal(19,0)       not null,
        EventOccurence      decimal(19,0)       not null,
        EventCode           int                 not null,
        EventDetailCode     int                 not null,
        Message             nvarchar(1024)              ,
        ApplicationPath     nvarchar(256)               ,
        ApplicationVirtualPath nvarchar(256)            ,
        MachineName         nvarchar(256)       not null,
        RequestUrl          nvarchar(1024)              ,
        ExceptionType       nvarchar(256)               ,
```

```
Details                    ntext                    ,
Primary key (EventId));
```

Kanttekeningen

Voor het gebruik van de membership en profile functies waren de attributen van de gebruiker ondergebracht in de tabel "gebruiker" samen met de gebruikersnaam en wachtwoord maar met de databaseconversie zijn deze geplaatst in het gebruikersprofiel in de tabel aspnet_profile.
In de kolommen PropertyNames en PropertyValuesString zijn de namen en waardes van deze attributen opgenomen. Voor het gemak worden ze hier genoemd: bedrijfsnaam, closed_account, abonneedatabase server, naam abonneedatabase, abonneedatabase gebruikersnaam en wachtwoord, redirect map, image map en bounce adres Abonneedatabase server, naam abonneedatabase, abonneedatabase gebruikersnaam en wachtwoord zijn de verbindingsgegevens voor de externe database van de desbetreffende gebruiker. Redirect map en image map zijn URL-adressen die verwijzen naar de plek waar links en afbeeldingen langs genavigeerd worden voor het doen van een effectmeting.

Omdat de unieke UserName altijd beschikbaar is nadat de gebruiker is ingelogd heb ik voor deze gekozen kolom als primaire sleutel in plaats van de UserId. De UserId moet namelijk geladen worden en daarna tijdelijk bewaard worden voor de sessie.

Microsoft SQL Server 2000 bezit niet het datatype boolean, in plaats daarvan is er gebruikt gemaakt van het datatype bit die de waardes 0 en 1 kan bevatten.
Het datatype uniqueidentifier bestaat uit een unieke hexidecimale waarde van 128-bits, bijvoorbeeld e3e60518-3f81-42bb-9461-582cdbf7769f of b319ce8f-61b8-4c2f-a220-919da58f66e1.

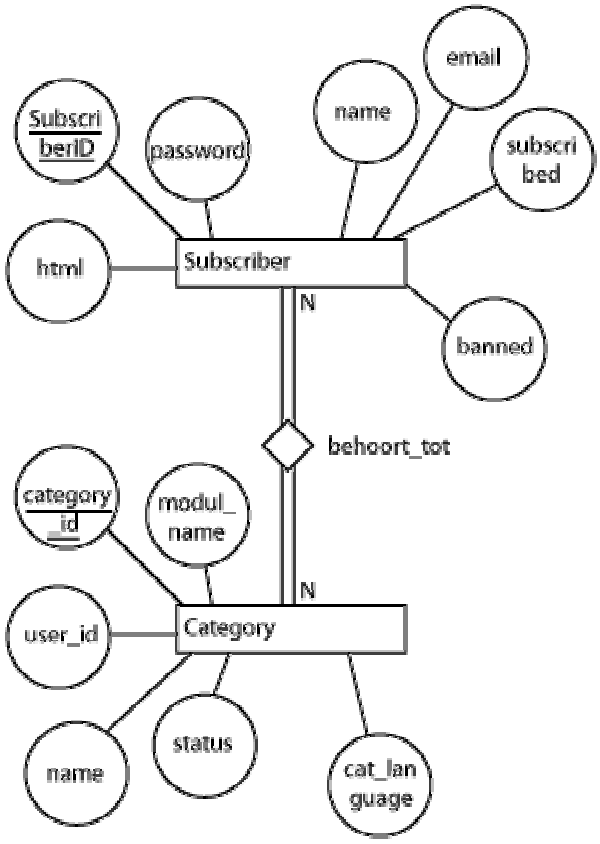
De datatypes nchar en nvarchar bevatten unicode-gecodeerde data die gelijk is binnen de verschillende grote platformen en programma's, op deze manier worden vertaalproblemen met de opgeslagen data tegengegaan.
Omdat een nchar-datatype twee keer zoveel ruimte inneemt als een char-datatype met dezelfde inhoud worden alleen belangrijke gegevens en gegevens van buiten de server (logingegegevens, URL's en emailadressen) opgeslagen in een unicoded kolom om de invloed op de performance te beperken.

In de tabel mailinglog refereert de kolom email tevens naar een html- of textbestand die de gehele versie van de verzonden email bevat.

Externe database

Het gedeelte dat van deze database zal gebruikt worden in de applicatie is ter verduidelijking opgenomen in dit document. Samen met de applicatiedatabase omvat deze de opslagplaatsen van de gegevens van de applicatie.

Conceptueel datamodel



Figuur 10. conceptueel datamodel externe database

Relationeel representatiemodel

Category (category_id, modul_name, user_id, name, status, cat_language)

Subscriber_Category (SubscriberID, CategoryID)

CategoryID is een foreign key die refereert naar Category, null niet toegestaan
SubscriberID is een foreign key die refereert naar Subscriber, null niet toegestaan

Subscriber (SubscriberID, password, name, email, subscribed, html, banned)

Implementatiemodel

```
create table Category (
        category_id             varchar(38)     not null,
        modul_name              varchar(10)     not null,
        user_id                 varchar(38)     not null,
        name                    varchar(255)    not null,
        status                  int             not null,
        cat_language            varchar(10)     not null,
        primary key (category_id));

Subscriber_Category (
        SubscriberID    char(40)        not null,
        CategoryID      char(40)        not null,
        primary key (SubscriberID, CategoryID)
        foreign key (SubscriberID) references Subscriber
                on delete cascades on update cascades,
        foreign key (CategoryID) references Category
                on delete cascades on update cascades);

Subscriber (
        SubscriberID    char(40)        not null,
        Password        char(8)         not null,
        Name            char(255)       not null,
        Email           char(255)       not null,
        Subscribed      datetime        not null,
        Html            bit                     ,
        Banned          int             not null,
                primary key(SubscriberID));
```