Multi-languagesupport voor het Virtueel Kantoor

Afstudeerverslag

Student: Jeroen Timmermans
Studentnr: 98001525
Bedrijf: Webbeat BV
Afstudeerperiode: 2003-2.1

Colofoon

Naam: Jeroen Timmermans
Studentnr.: 98001525
Opleiding: Vormgeving en ontwerp van interactie (VIA)
Project: Multi-languagesupport voor het virtueel kantoor
Bedrijf: Webbeat BV
Bedrijfsmentor: ing G. Merkelbach
Examinator 1: P. Deters
Examinator 2: M. van Vliet

Haagse Hogeschool
Bezoekadres: Johanna Westerdijkplein 75
2521 EN Den Haag

Postadres: Postbus 19
2520 CB Den Haag

Telefoon: 070- 455 84 00
Website: www.hhs.nl

Webbeat BV
Adres: Treubstraat 27
2288 EH Rijswijk ZH
Telefoon: 070 352 2664
Fax: 070 319 2381
Website: www.webbeat.nl

Referaat

Timmermans, J.A., afstudeerverslag bij Webbeat BV, Rijswijk ZH.

Dit is het afstudeerverslag van J.A. Timmermans en beschrijft de activiteiten die uitgevoerd zijn in het kader van de afstudeerperiode 1 september 2003 tot en met 9 januari 2004. Het afstudeerproject bestond uit het ontwerpen en ontwikkelen van een hulpmiddel die het mogelijk zou maken om een applicatie in meer dan één taal aan te bieden.

Het afstudeerproject als afsluiting van de opleidingsvariant Vormgeving en ontwerp van Interactie van de opleiding Informatica & Informatiekunde aan de Haagse Hogeschool, te Den Haag.

Descriptoren

Afstuderen
Webbeat BV
Rapid Prototyping
Multilanguage
Multilingual solution
XML

Inhoud

1 Inleiding

Dit document is geschreven in het kader van de afstudeerstage die als afsluiting van de vierjarige HBO-opleiding VIA (Vormgeving en ontwerp van Interactie) aan de Haagse Hogeschool, sector Informatica & Informatiekunde. Deze afstudeerstage is uitgevoerd bij Webbeat BV.

De indeling van het verslag is als volgt:

Hoofdstuk 2: Dit hoofdstuk schetst een beeld van de omgeving waarin ik mijn afstudeerstage heb uitgevoerd.

Hoofdstuk 3: Hier leest U de opdracht zoals deze bij aanvang van de stage was opgesteld.

Hoofdstuk 4: In dit hoofdstuk staan alle activiteiten beschreven die ik heb uitgevoerd tijdens mijn afstudeerstage.

Hoofdstuk 5: De evaluatie van het proces en het product staan ik in dit hoofdstuk beschreven.

Deel 1
Organisatie en opdracht

2 Organisatiebeschrijving

In dit hoofdstuk zal ik beschrijven in welke omgeving ik mijn afstudeerstage heb doorlopen.

2.1 Webbeat BV

Webbeat is een klein bedrijf dat zich bezighoudt met internettechnologieën, en deze toepast in diverse applicaties. De belangrijkste producten op dit moment zijn het Virtueel Kantoor en Speedtest.
Webbeat telt momenteel vijf medewerkers, en er kan daardoor ook niet gesproken worden van afdelingen. Iedere medewerker krijgt opdrachten toebedeeld aan de hand van zijn of haar specialisme.

2.2 Geschiedenis

Webbeat is in december 2000 gestart met als eerste product het Virtueel Kantoor. De eerste medewerkers van Webbeat waren oud CMG collega's die in zeer uiteenlopende omgevingen al geruime ervaring hadden met Internet en Intranet toepassingen. Webbeat streeft vanaf het begin naar een evenwichtige verdeling tussen klantprojecten en eigen produkten. In eerste instantie lag de nadruk meer op de klantprojecten maar de produkten zijn inmiddels begonnen aan een inhaalslag.

2.3 Belangrijkste producten

Zoals gezegd zijn de belangrijkste producten het Virtueel Kantoor en Speedtest. Het Virtueel Kantoor is een digitale omgeving die veel functies bevat die normaliter op papier uitgevoerd zouden worden. Voorbeelden hiervan zijn een digitale agenda, urenregistratie en documentbeheer.
Ieder bedrijf of afdeling kan een eigen Virtueel Kantoor aanvragen, en hier zelf gebruikers aan toevoegen. Er bestaan twee verschillende vormen; de Basis- en de Pro-uitvoering. Het Basiskantoor is gratis, maar in aantal gebruikers en mogelijkheden beperkt. Voor het Pro-kantoor moet een maandelijks bedrag betaald worden, en zijn alle functionaliteiten beschikbaar.
Het Basiskantoor telt momenteel ruim 12.000 gebruikers, het Pro-kantoor 856.

Speedtest is een website die de snelheid van de ISP's (Internet Service Providers) meet. Speedtest wordt momenteel onder andere gebruikt door de Consumentenbond in hun onderzoeken.
Ook consumenten kunnen hun eigen verbinding hier testen, of kijken wat de beste of goedkoopste internetverbinding is in hun postcodegebied. Dagelijks bezoeken ongeveer 30.000 mensen Speedtest.nl.

2.4 Plaats van de afstudeerder

Doordat er weinig medewerkers zijn, is er ook geen sprake van een strakke hiërarchie. De afstudeerder staat daarom binnen de organisatie gelijk aan de overige medewerkers.

3 De opdracht

Het volgende hoofdstuk zal de opdracht beschrijven zoals die aan het begin van de afstudeerperiode is opgesteld. In dit hoofdstuk staan ondere andere de probleem- en doelstelling, net zoals de planning.

3.1 Zoeken naar een opdracht

Tijdens mijn zoektocht naar een stageplaats werd mij al duidelijk dat de arbeidsmarkt in de ICT-branche aan het veranderen was. Er kwamen steeds meer arbeidskrachten, terwijl de vraag afnam of hetzelfde bleef. Wat als logisch gevolg had dat de bedrijven keus kregen, waar zij eerst blij mochten zijn als er iemand uit de ICT bij hen wilde werken. Gezien de economische ontwikkelingen die voor mijn stage al waren ingezet, en zich erna hebben doorgezet, vermoedde ik dat het vinden van een geschikte afstudeerplaats nog veel moeilijker zou gaan worden.
Het aantal aangeboden afstudeerplaatsen bleek nog zeer hoog, maar de kwaliteit van deze afstudeeropdrachten liet te wensen over. Daarom besloot ik om eerst te beslissen waar de opdracht aan moest voldoen voor mij persoonlijk en welke eisen de opleiding aan een opdracht stelde.
Ik wilde geen simpele opdracht, ik wilde een opdracht waarmee ik iets anders deed dan de gemiddelde afstudeerder. Door een dergelijke eis te stellen vielen een groot aantal opdrachten meteen af. Bij Webbeat waren een aantal afstudeeropdrachten beschikbaar die stuk voor stuk binnen mijn interessegebied vielen, niet de gemiddelde afstudeeropdracht waren en voldeden aan de eisen vanuit de opleiding.

Door de ervaring van Webbeat met eerdere afstudeerders bestond de mogelijkheid om samen met hen de opdracht dusdanig te formuleren dat deze als afstudeeropdracht aan de opleiding gepresenteerd kon worden.

3.2 Probleemstelling

De producten van Webbeat zijn momenteel allemaal Nederlandstalig. Het probleem hierbij is dat alleen mensen die de Nederlandse taal machtig zijn deze producten kunnen gebruiken. Webbeat wil deze producten in meerdere talen aan kunnen bieden, zonder daarvoor het product opnieuw te hoeven programmeren.

3.3 Doelstelling

De doelstelling is het ontwerpen en implementeren van een XML-omgeving die multi-taal ondersteuning biedt aan de huidige en toekomstige applicaties.

3.4 Uitgangssituatie

Software

- Microsoft Visual Studio
- Microsoft Visual Interdev
- Internet Information Services 6
- SQL Server 2000

Hardware

- Testserver; AMD Athlon XP 2600+, 1 GB DDR Ram

Aanwezige ideeën

Binnen het bedrijf zijn de volgende ideeen met betrekking tot dit project ontstaan:

- Op de XML resource sets dient automatisch versiebeheer plaats te vinden
- De tool dient op een intelligente manier gebruik te maken van een set met algemene termen die in verschillende applicaties voorkomen
- De tool dient de XML resource sets zelf te publiceren op een per applicatie te speciferen locatie, of te emailen aan een per applicatie te specificeren beheerder

3.5 Werkzaamheden

- Onderzoek projectmethodiek Rapid Prototyping
- Ontwerpfase
    - Onderzoek Virtueel Kantoor
    - Onderzoek naar eerdere Multi-taal oplossingen
    - Opstellen adviesrapport Multilingual Solutions
    - Beschrijven eisen aan de Multilanguage tool
    - Bestuderen mogelijkheden XML
    - Technisch ontwerp
- Publish-, Client-, Debugmodule
    - Ontwikkelen
    - Prototyping
    - Testen
- Beheer
    - Grafisch ontwerp
    - Ontwikkelen
    - Prototyping
    - Testen

3.6 Op te leveren producten

- Adviesrapport Multilingual Solutions
- Eisenomschrijving
- Ontwerprapport
- Prototypes van de modules
- XML-omgeving die multitaal ondersteuning biedt aan applicaties

3.7 Globale planning en fasering

De globale planning is opgesteld aan het begin van het project. Het geeft een overzicht wanneer de diverse activiteiten uitgevoerd zullen worden. Om het geheel overzichtelijk te maken, is de opdracht opgedeeld in 4 onderdelen; Beheer, Publish, Client en Debug.
Beheer bevat de GUI waarmee de resources beheert worden, en alleen binnen Webbeat toegepast wordt. Vanuit deze GUI wordt ook de Publishmodule geactiveerd, die de resources op de diverse servers plaatst.

De Clientmodule is hetgene wat uiteindelijk in het Virtueel Kantoor geïmplementeerd wordt (en later ook in andere toepassingen). Deze module zal voor het daadwerkelijke vertalen zorgen.
Tot slot is er de Debugmodule, dit is een hulpmiddel voor programmeurs. De debugmodule loopt de ASP-code door en controleert de MLT-code op fouten.

Week -->	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
OZ RP																				
OZ VK																				
OZ MTO																				
AR MTS																				
Eisenpakket																				
XML studie																				
Tech. ontw.																				
Beheer																				
Graf. ontw.																				
Ontwikkelen																				
Prototyping																				
Publish																				
Ontwikkelen																				
Prototyping																				
Client																				
Ontwikkelen																				
Prototyping																				
Debug																				
Ontwikkelen																				
Prototyping																				
Uitloop																				
Verslag																				

3.8 Methodiek

De afstudeerder zal gebruik gaan maken van de projectmethodiek Rapid Prototyping. Met deze methodiek wordt een product iteratief ontwikkeld. Het traject wordtgestart door het opstellen van een lijst met eisen en functies, waar het uiteindelijke product aan moet voldoen. De cyclus begint met de ontwerpfase, waarin de eisen uitgewerkt worden in een ontwerprapport. Na de ontwerpfase wordt er begonnen met de ontwikkelfase. Deze fase beslaat het ontwikkelen van het prototype aan de hand van het ontwerprapport.
In de prototyping fase die hierop volgt wordt het ontwikkelde (sub)product getest op functionaliteit en correcte werking. Het vierde en laatste deel in deze cyclus is de testfase, waarin het product wordt beoordeeld aan de hand van de gestelde eisen.

Als de eisen en functies niet terugkomen in het opgeleverde product, wordt de cyclus opnieuw doorlopen. Indien het product wel voldoet aan de op voorhand gestelde eisen, kan het product geïmplementeerd worden.

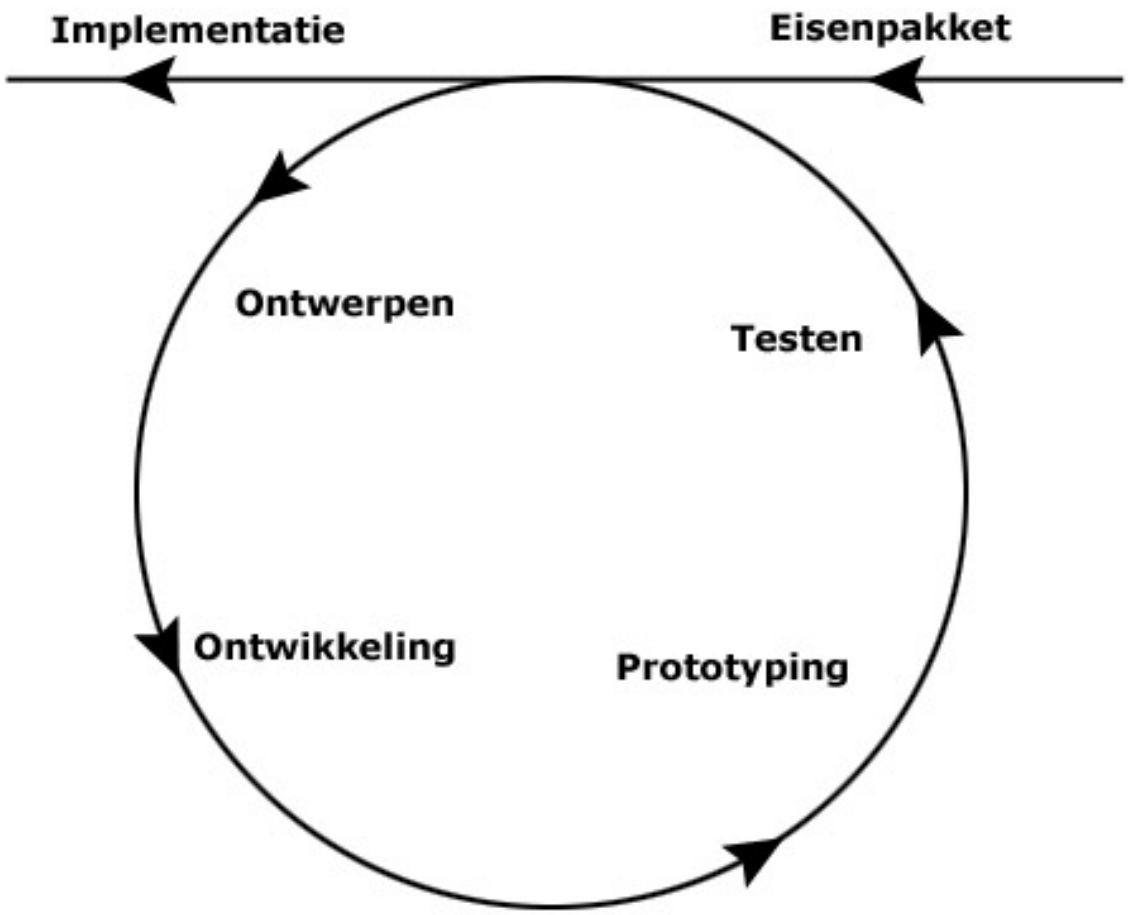


Diagram 1 Rapid Prototyping

3.9 Technieken

Tijdens de afstudeerstage zal er gebruik gemaakt worden van de techniek 'bronnenonderzoek' voor het onderzoek Multi-taal Oplossingen. Om de functionaliteit van de tool te testen zal ik gebruik maken van de testtechniek 'Workshop'.

Deel 2
Werkzaamheden

4 Uitgevoerde werkzaamheden

In dit hoofdstuk zullen naast de werkzaamheden, ook de tegengekomen obstakels beschreven worden. Tevens worden alle keuzes vermeld, alsmede de beargumentatie.

4.1 Onderzoek Rapid Prototyping

Aangezien Webbeat gebruik maakt van RP (Rapid Prototyping) als methodiek bij het ontwikkelen van al haar applicaties, leek het mij interessant om eens te weten te komen in hoeverre deze methodiek verschilt van IAD.
Ik ben begonnen door in Google op Rapid Prototyping te zoeken. Hier kwamen wel de voordelen van RP naar voren, maar een goed beeld van de methode kreeg ik hier niet van.
Een vriend van mij kwam met een zeer informatieve website die een beschrijving gaf van Rapid Prototyping, en de verschillende fases die binnen deze methode gebruikt werden. Voor een beschrijving van Rapid Prototyping verwijs ik U graag naar paragraaf 3.8; Methodieken.
Het bleek dat RP goed toe te passen is in kleinere projecten, aangezien er wordt uitgegaan van voortschrijdend inzicht. Het iteratieproces is zeer kort, en ook de ontwerpfase zit in dit proces opgesloten.
Ik heb tijdens de diverse projecten binnen de opleiding meegemaakt dat de eerste 10 à 11 weken vooral besteed werden aan het schrijven van documentatie, vooronderzoeken en rapportages, die later in een aantal gevallen weer teniet werden gedaan door het voortschrijdend inzicht.

Bij de keuze tussen IAD en RP heb ik gekeken naar de aard van het project. Aangezien aan het begin van het traject de opdracht nog abstract en onbekend was, was voortschrijdend inzicht een belangrijke factor. Ook de communicatie met Webbeat was van groot belang. Deze twee argumenten pleitten voor een keuze voor RP. Anderzijds was en is IAD een methode waar ik negen maal mee gewerkt heb, en dus bekend mee ben. Dit neemt het risico weg dat ik een methode kies die eigenlijk niet bij de opdracht past.
In het onderzoek heb ik niet alleen de voor- en nadelen van RP bekeken, maar ook de verschillen met IAD. Het verbaasde me hoeveel raakvlakken IAD en RP eigenlijk hebben. Beiden zijn iteratief, en zeer goed aan te passen aan de behoeften van het

project. Het belangrijkste verschil zit hem in de documentatie. Waar bij IAD een gedetailleerde definitiestudie opgeleverd wordt, is het opstellen van een lijst met eisen voldoende bij het gebruik van RP.
IAD gaat vooral uit van een van te voren gedefinieerd idee dat uitgewerkt moet worden, en biedt een uitstekende projectbeheersing.
RP gaat op het iteratieve vlak verder dan IAD, aangezien het gehele ontwerp (wat in IAD al in de definitiestudie vastligt) in de iteratie meegenomen wordt. Dit heeft als voordeel dat voortschrijdend inzicht geen grote gevolgen heeft, het nadeel is dat er een groot risico is van oneindige iteratie door datzelfde voortschrijdend inzicht.
De betrokken partijen moeten ervoor waken dat dit niet gebeurd.
Op basis van de flexibiliteit heb ik uiteindelijk gekozen voor RP. Omdat het project nog niet concreet genoeg was voor mij om een helder beeld te kunnen vormen, was de flexibiliteit het belangrijkste aspect voor mij.

4.2 Onderzoek Multilingual Solutions

In samenspraak met de opdrachtgever is besloten om een onderzoek te doen naar bestaande middelen om applicaties in meerdere talen aan te bieden. Het leek de opdrachtgever en mij namelijk verstandig om te leren van de ervaring die anderen al hadden opgedaan.
In eerste instantie heb ik bekeken wat voor soort oplossingen er beschikbaar zijn. Dit vond ik nodig om gerichter te kunnen zoeken. De twee belangrijkste methoden die ik tegenkwam waren Hardcoded Multi-Language en Dynamic Translation.
By Hardcoded Multi-Language wordt de aplicatie in meerdere talen geprogrammeerd, wat dus fysiek verschillende applicaties oplevert. Als de gebruiker een andere taal kiest, gaat hij/zij in feite naar een compleet andere applicatie.
Dynamic Translation is een onderhoudsvriendelijkere methode. Er wordt slechts één programma geschreven, in één taal, en de tekst wordt door een woordenboek-systeem vertaald. Hier zag ik al gauw de nadelen van, aangezien grammatica iets is wat nog (bijna) geen enkel programma perfect lukt. Ik zag hier ook geen heilzame oplossing in vanwege het prijskaartje wat aan de commerciële producten hangt; vanaf €50.000 heb je een systeem dat dynamisch een website vertaald voor de gebruiker.

Het bleek lastig om degelijke, technische informatie te krijgen over de bestaande oplossingen. Dit kwam vooral omdat taaltechnologieën nog in de kinderschoenen

staan, en bedrijven er veel geld in geïnvesteerd hebben. Ze zullen dus voorzichtig zijn met het vrijgeven van technische informatie.
Er was echter één artikel waar ik redelijk veel informatie uitgehaald heb. Dit artikel is geschreven voor een website die zich richt op webdevelopers, waardoor het artikel uit heldere en objectieve informatie bestond. Het artikel bevatte naast de twee eerder genoemde methodes, nog twee methodes; handmatige vertaling en vertaling vanuit een XML-bestand. Bij handmatige vertaling worden alle mogelijke vertalingen in een database opgeslagen, en door het systeem opgehaald. Het XML-gebaseerde systeem is een afgeleide hiervan, met als groot voordeel dat het ophalen uit XML-bestanden vele malen efficiënter is.

Aangezien er een grote groep mensen gebruik gaat maken van de Clientmodule, is efficiëntie van zeer hoog belang. Één van de eisen die bij het opstellen van de opdracht al naar voren kwam was namelijk dat de gebruikers niet mogen merken dat er woorden uit een extern bestand gehaald worden.
Op basis van deze bevindingen heb ik uiteindelijk het adviesrapport Onderzoek Multilingual Solutions geschreven, welke U terug kunt vinden in de bijlagen. Dit adviesrapport is uitgegaan naar de bedrijfsmentoren.
Nadat zij dit rapport gelezen hadden, hebben we in een bespreking besloten welke oplossing er gebruikt zou worden. De oplossing die Webbeat voor ogen had, kwam overeen met de door mij geadviseerde methode. Tijdens deze bespreking hebben we ook de eerste eisen omschreven.

4.3 Onderzoek XML

Aangezien XML een belangrijke rol zou gaan spelen binnen de opdracht, was het belangrijk dat ik mij de basis van XML eigen zou maken. Hiervoor heb ik gebruik gemaakt van bronnen op Internet, die ik vond via Google, de pagina van het W3C en xml.pagina.nl.
Wat mij vooral opviel tijdens deze zoekgangen, was dat de meeste pagina's XML nog vooral zien als tweede generatie HTML, en niet als een nieuwe manier van data aanbieden. De meeste site concentreerden zich dan ook op hoe je een pagina met XML kon tonen in een browser alsof het HTML was.
Na het ontdekken van de mogelijkheden van XML, vind ik dit erg kortzichtig, en doet het de mogelijkheden van XML tekort. Op de sites die zich wel serieus concentreren op de mogelijkheden van XML (onder andere

http://wdvl.com/Authoring/Languages/XML/Intro/ en http://www.xml101.com), leerde ik hoe een XML-document is opgebouwd, aan welke eisen deze moet voldoen en hoe je zelf ook eisen kunt stellen. Verder werd er ook aandacht geschonken aan het opmaken van deze gegevens in XSL, maar hier heb ik geen of weinig aandacht aan geschonken, aangezien XML binnen dit project alleen gebruikt wordt als datasource, en de opmaak door ASP geregeld wordt.
Ik kwam bij het onderzoek ook de zoekmethode Xpath tegen. Dit is een onderdeel van het Microsoft XMLDOM-object, dat door ASP gebruikt wordt om XML-bestanden te lezen. Met Xpath wordt een XML-tag opgezocht door middel van een query. In deze query wordt de waarde van een attribuut van een tag opgevraagd. Zo'n query ziet er als onderstaand uit.

Code:

```
XML.selectSingleNode(/resourceset/resource/tra_translation[@res='forum'])
```

Na het afronden van dit onderzoek, heb ik een afspraak gemaakt met de twee betrokken personen van dit project, en mijn bevindingen aan hen voorgelegd. De methode die ik voorstelde om te volgen, bleek ook de methode te zijn die zij in grove lijnen voor ogen hadden.

4.4 Opstellen eisenpakket

Na het kiezen van een methode om de sites in meerdere talen aan te bieden, is het opstellen van de eisen direct begonnen. Dit is in de vorm van brainstormsessies gegaan, die direct aansloot op de bespreking. Voor de eerste sessie kon ik mij niet voorbereiden, aangezien de opdracht nog niet helder was voor mij. Daardoor had ik een passieve rol; ik deed niet actief mee in het bijdragen van ideeën, ik bekeek kritisch de ideeën van de twee anderen, ik noteerde de functies van de MLT (Multi-Language Tool), en baseerde hierop mijn ideeën voor de volgende sessie. Deze sessie bevatte geen structuur; er werden ideeën over de MLT geöpperd, en daar werd verder op doorgegaan. Tijdens deze sessie is wel besloten dat het project in vier delen opgedeeld zou worden; Beheer, Publish, Client en Debug. Beheer omvat het onderhoud op de database door middel van een Graphical User Interface (GUI). Publish zet de database-gegevens om in XML-bestanden. Client zorgt voor het ophalen van de data uit de XML naar de client toe en tot slot is de Debug een module die controleert of de uitvoer die door de Client gegenereert wordt, ook daadwerkelijk

de uitvoer is die de programmeur wil. Na afloop van deze sessie werd er direct een afspraak gemaakt voor de volgende sessie.
In die sessie werden er concretere ideeën geleverd, omdat er voor iedereen een beter beeld was ontstaan van wat er binnen mijn afstudeerstage mogelijk was. Niet alleen qua tijd, maar zeker ook qua kennis van mijn kant.
Tijdens deze sessie hebben we de ideeën uit de eerste sessie beoordeeld op haalbaarheid, en besloten om eerst de kern-functies van de MLT te ontwikkelen en als er tijd over was na te gaan denken over verdere functies.
Toen eenmaal besloten was welke functies ontwikkeld zouden worden tijdens de afstudeerstage, zijn we prioriteiten vast gaan stellen, op basis van drie prioriteitsniveaus; basis, luxe en wens. Basisfuncties zijn de functies die de hoogste prioriteit hebben, luxe zijn één niveau lager, en wensen zijn functionaliteiten die ontwikkeld worden indien er tijd en middelen beschikbaar voor zijn.
U kunt het uiteindelijke eisenpakket terug vinden in de bijlagen.

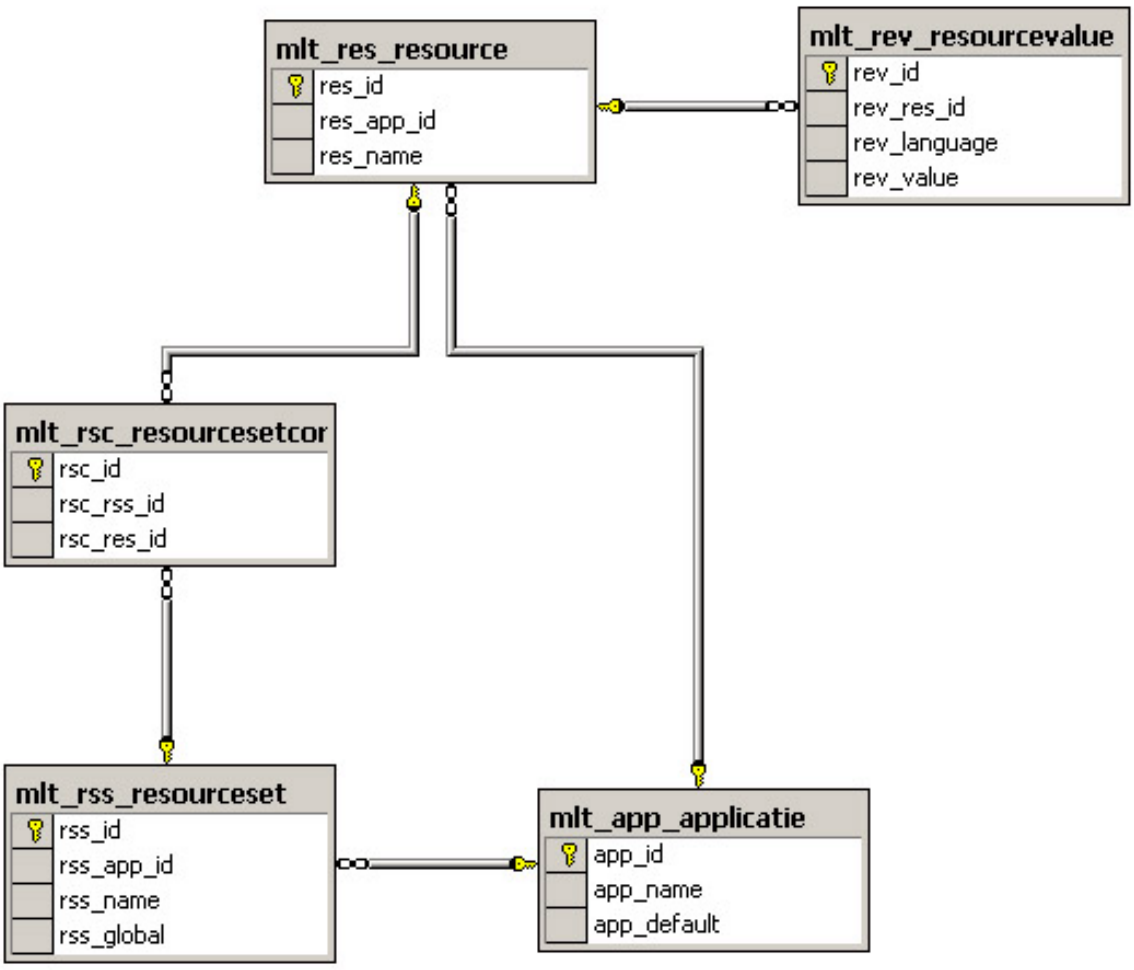
4.5 Ontwerprapport

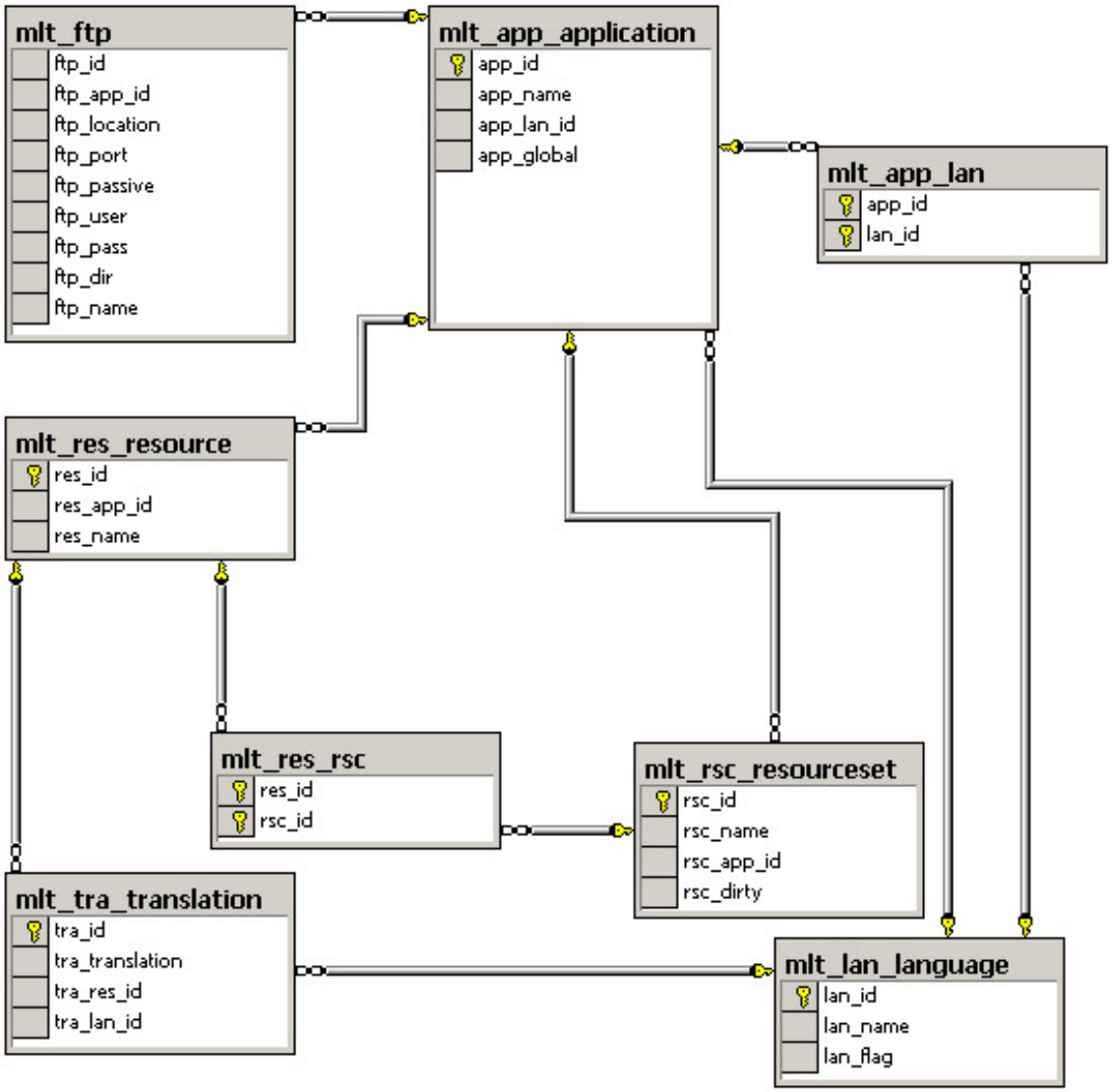
Het technisch ontwerp is gebaseerd op de functies zoals die beschreven stonden in het eisenpakket en het algmene beeld wat ik kreeg uit de diverse besprekingen. Omdat de MLT voornamelijk een onzichtbare tool is, heb ik gekozen om met behulp van SDL-diagrammen de programmastructuur globaal op papier te krijgen. Een SDL-diagram geeft de globale structuur en verloop van een applicatie schematisch weer. Door het gebruik van een dergelijk diagram was ik in staat om de toepassing te ontwerpen zonder over code na te hoeven denken. Dit bespaarde veel tijd, aangezien het aanpassen van een ontwerp minder tijd kost dan programma-code aanpassen.
Tijdens het schrijven van het technisch ontwerp kwamen nog meer ideeën naar voren, die ook na overleg in het ontwerp geïmplementeerd werden. Één van deze ideeën was het versiebeheer op de bestanden. Automatisch versiebeheer bleek een haalbaar idee te zijn, maar werd naar de achtergrond geschoven omdat het niet een hoge prioriteit had. De lage prioriteit is het gevolg van de toepassing van de MLT. De MLT zal namelijk in eerste instantie alleen in het Virtueel Kantoor gebruikt worden, waardoor het handmatig beheersen van de versies nog overzichtelijk is. Pas als er er meerdere applicaties gebruik gaan maken van de MLT, heeft automatisch versiebeheer een toegevoegde waarde en dus een hogere prioriteit.

Om het automatisch beheer mogelijk te maken, moet er een systeem komen dat van afstand de diverse XML-bestanden kan inlezen, en dan kan zien aan de hand van de datum (die al in de XML staat opgenomen) of dit de nieuwste versie is.
We hebben besloten om het versiebeheer te beperken tot een zogenaamd 'dirty'-veld in de resourceset-databasetabel. Indien er een fout optreedt tijdens het uploaden, wordt dit veld gemarkeerd, en komt er een uitroepteken achter de desbetreffende resourceset te staan; dit is slechts een grafisch geheugensteuntje aan de beheerder dat de resourceset op de server niet de nieuwste versie is.
Een ander idee was om in de Debugmodule een automatische functie maken, die geen zichtbare feedback geeft, maar direct alle missende resources in de database invoert. Dit heeft als voordeel dat er weinig handmatig werk gedaan moet worden (alleen het invoeren van de vertalingen), maar heeft als groot nadeel dat als een resource verkeerd gespeld is, niet herkent wordt en als nieuwe resource ingevoerd wordt door het systeem. Deze methode kan nog verder geperfectioneerd worden, maar zal in de huidige vorm al opgenomen worden in het ontwerp, zodat het later makkelijker is om de uitgewerkte vorm te implementeren.
Bij het technisch ontwerp is er ook gekeken naar toekomstige toepassingen van de MLT. Zo is er alvast een default-taal opgenomen, waar men aan kan geven in welke taal de toepassing getoond moet worden, als er geen voorkeur-taal is aangegeven. Ook de mogelijke talen waarin een applicatie bekeken kan worden is alvast in het ontwerp meegenomen, mocht dit in een later stadium noodzakelijk blijken.

In het technisch ontwerp is ook het databaseontwerp opgenomen. Er is gekozen voor een relationele database, omdat Webbeat een licentie heeft op SQL Server 2000, deze stabiel en goed loopt, en de aanschaf van een object geörienteerde database daarom bijna niet te verantwoorden is. Daarnaast heeft een OO-database bijna geen toegevoegde waarde binnen een web-applicatie, omdat de voordelen die een OO-database heeft ten opzichte van een relationele database niet tot uiting komen in een relatief simpele database-omgeving zoals deze. Echter, dat Webbeat geen interesse had in het migreren van alle gegevens van een relationele naar een object geörienteerde database, en de daarbij komende pijn en moeite om alles te installeren, was de belangrijkste reden om voor een relationele database te kiezen.
Om tot dit relationeel model te komen ben ik begonnen met een objectmodel te schetsen, en alle noodzakelijke velden te beschrijven. Deze velden zijn deels gebaseerd op de bestaande MLT-database, en deels toevoegingen.

De volgende twee afbeeldingen zijn de relationele modellen van de oude en de nieuwe database.





Bij het eerste ontwerp had ik de database volledig uitgenormaliseerd. Dit leverde een zeer net en overzichtelijk databasemodel op. Alleen op het gebied van prestaties zou dit model problemen opleveren. Ik had namelijk geen relatie tussen de tabel mlt_res_resource en mlt_app_application aangemaakt in het eerste ontwerp, wat als gevolg had dat als ik resources per applicatie wilde selecteren, dat via de mlt_rsc_resourceset-tabel moest gebeuren. Bij vijf gebruikers vormt dit geen probleem, maar aangezien de databaseserver voor meerdere databases gebruikt wordt, is het van belang dat er zo min mogelijk belasting plaats vindt.
Ik heb de mlt_lan_language-tabel toegevoegd om zo een centrale plaats te hebben om de diverse talen op te slaan. Zo kunnen er geen inconsistentie-fouten ontstaan

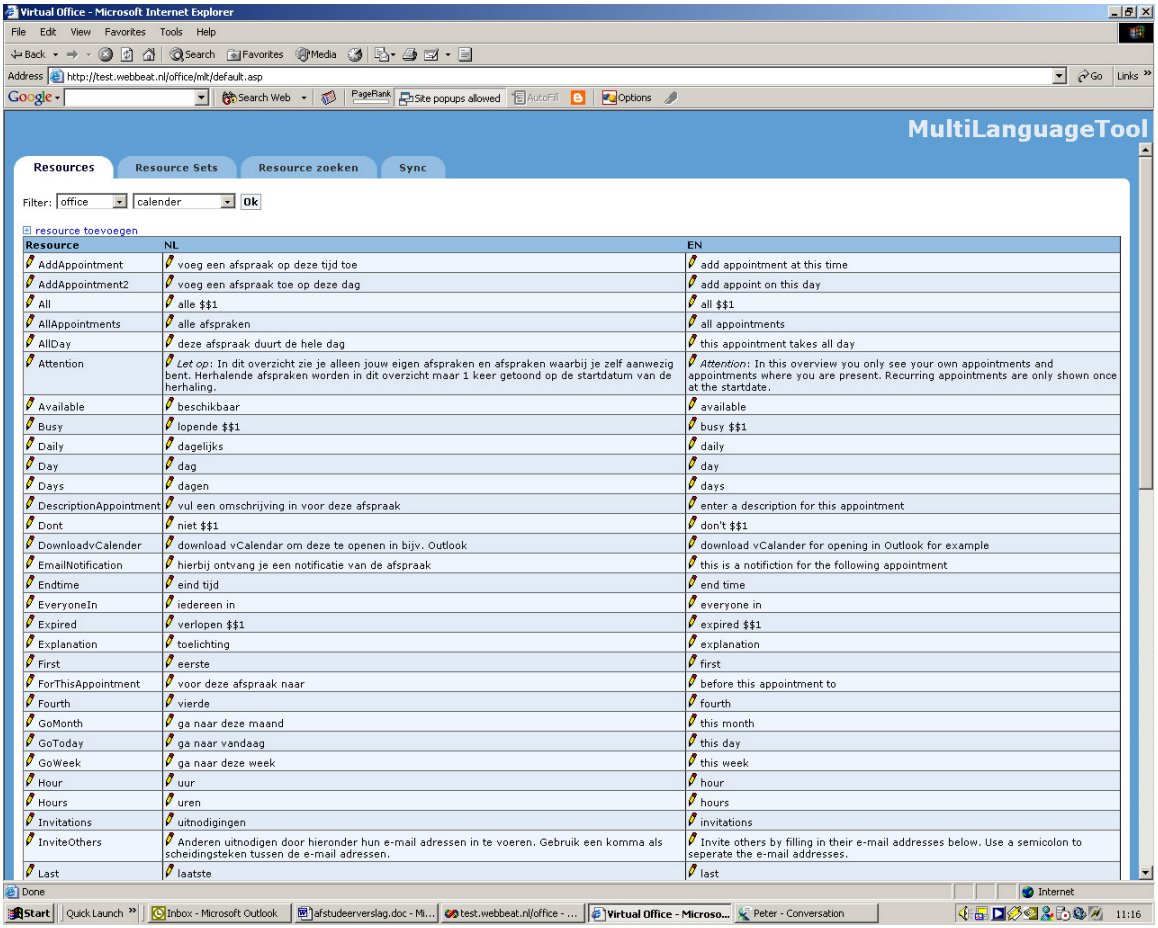
bij het definiëren van de standaardtaal, het aangeven van de beschikbare talen, en het instellen van de taal van een vertaling in mlt_tra_translation.
De tabel mlt_ftp is toegevoegd voor de functie van de Publishmodule. Deze zal door middel van FTP de XML-bestanden uploaden naar externe servers. Deze servers worden in deze tabel vastgelegd. Ik heb gekozen voor een aparte tabel, omdat het mogelijk is dat een applicatie op meerdere locaties staat.

4.6 Beheer

Het beheergedeelte is de module waarmee de programmeurs van Webbeat de applicaties, resourcesets, resources, vertalingen en talen kunnen toevoegen, bewerken en verwijderen. Alleen medewerkers van Webbeat kunnen hierbij komen, waardoor bij het ontwerpen rekening gehouden moest worden met een groep van vijf mensen.

Oude situatie

Bij het ontwerp van het nieuwe beheerscherm heb ik het oorspronkelijke beheerscherm dat door Webbeat ontwikkeld is als uitgangspunt genomen. Deze interface bleek zeer gebruikersonvriendelijk. Figuur 1 toont de interface na drie klikken. Om daadwerkelijk nog actief beheer uit te voeren, moet er minstens nog twee maal geklikt worden. Dat is vijf klikken om onderhoud aan een resource te plegen. Volgens de bedrijfsmentor was dit te veel.



Figuur 1 Oude resourcebeheerscherm

Ontwerp van nieuwe GUI

Naar mijn mening kon dit ook sneller. Ik ben begonnen om dit op te lossen door naar de huidige structuur te kijken. Deze had als voordeel dat alle elementen netjes gescheiden en dus overzichtelijk geplaatst waren. Het nadeel was dat het niet lekker werkte volgens de gebruikers; er miste een bepaalde ergonomische logica in.
Omdat het kleurgebruik mij wel aansprak, en voor de gebruikers ook bekend is (hetzelfde kleurenschema wordt in het Virtueel Kantoor toegepast), heb ik ervoor gekozen om zoveel mogelijk bij deze kleuren te blijven. Ik ben alleen afgestapt van het tabbladen systeem dat zij hanteren, omdat volgens mij de belangrijkste oorzaak was van het onprettig werken met de beheerstool.
De iconen die Webbeat gebruikt heb ik overgenomen, aangezien dit iets is wat de medewerkers gewend zijn. Het icoontje om een item weg te gooien is een

vuilnisbakje (), om een item te bewerken heb ik een potloodje () als icoon gebruikt.

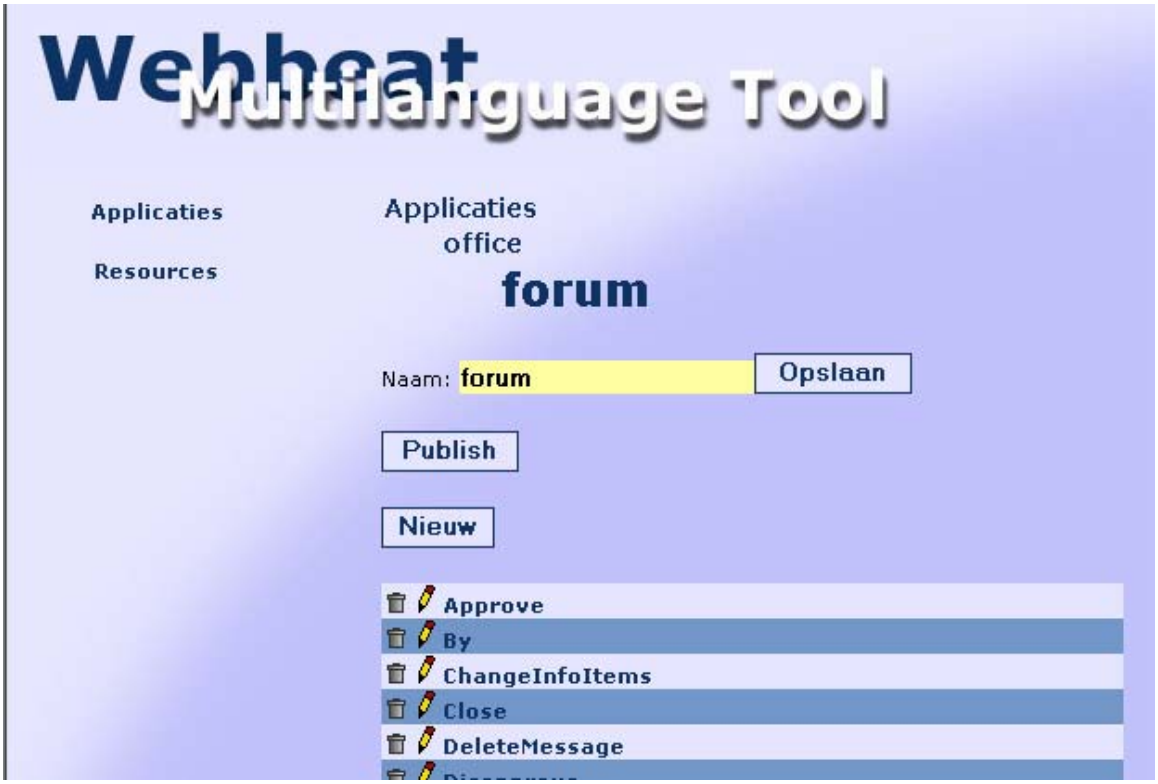
Om de informatie op een juiste manier in te kunnen delen, ben ik in Photoshop begonnen met het plaatsen van de juiste informatie in het juiste scherm. Het voordeel van deze methode is dat kleuren, letertypes, en eventuele plaatswijzigingen. Het nadeel van het gebruik van Photoshop is dat het moeilijk is om de exacte plaatsing en kleuren van items over te nemen in de uiteindelijke webpagina. Ik heb dit risico erkend en alleen kleuren gebruikt die gemarkeerd werden in Photoshop als web-safe; kleuren die door iedere browser hetzelfde getoond worden. Alleen wat betreft de plaatsing van de diverse objecten en teksten was het niet mogelijk om het dupliceren bij het ontwikkelen zo exact mogelijk te laten verlopen. De vraag is echter; hoe ernstig is dit probleem? Ik beoordeelde dat het geen enkel probleem was, aangezien ik de objecten en tekstblokken in Photoshop alleen gebruikte als grove indicatie van de invulling van het scherm.
Een ander aspect wat ik in Photoshop ontwierp was de structuur. Ik wilde een trappensysteem ontwikkelen, waarbij een gebruiker alle direct gerelateerde sub-items bij een gekozen item te zien krijgt. Door een logische hiërarchische structuur te bedenken, gebaseerd op de gesprekken die ik voerde met de twee voornaamste gebruikers van de MLT, werd dit trappensysteem helder.
Ik ben uitgegaan van de programmeur, die werkt in een applicatie, dus om bij de resourcesets te komen die voor hem van toepassing zijn, is het van belang om eerst aan te geven in welke applicatie hij aan het werk is. Vanuit deze beredenering besloot ik om een overzicht van alle applicaties in het openingsscherm te tonen (Figuur 2).
In het daarop volgende scherm (figuur 3) krijgt de programmeur een overzicht van de eigenschappen van de applicatie die hij koos, bijvoorbeeld ‘office’. Om de moeite van het bewerken van deze eigenschappen zoveel mogelijk te beperken, heb ik in dit scherm ook de mogelijkheid ingebouwd om direct velden te bewerken.

Figuur 2 Applicatieoverzicht

Tijdens het ontwikkelen kwam ik een logisch probleem tegen; wat als de gebruiker een default-taal kiest die niet voorkomt in de lijst met beschikbare talen? Hierop besloot ik om een controle hierop in te bouwen. Als de gebruiker de veranderingen wil opslaan, wordt er eerst gecontroleerd of de default-taal wel beschikbaar is.

Figuur 3 Applicatiedetailscherm

Door een resourceset aan te klikken, krijgt de programmeur een overzicht van de resources die bij de gekozen resourceset horen, in dit geval 'Forum'. In dit scherm gaat verder dezelfde logica schuil achter de indeling en keuzes als in het vorige scherm.



Figuur 4 Resourceset-detailscherm

Figuur 5 Resource-detailscherm

Als de programmeur vervolgens gekozen heeft welke resource hij wil bewerken, wordt hem het scherm dat in figuur 5 is afgebeeld getoond. In dit geval gaat het om de resource 'Close'. Iedere resource kan in meerdere resourcesets geplaatst worden, daarom heb ik ervoor gekozen om onder de vertalingen een lijst met alle resourcesets van de gekozen applicatie te plaatsen. Door de bijbehorende hokjes aan te vinken, geeft de gebruiker aan bij welke resourcesets de resource nog meer hoort. Ik heb voor Checkboxes gekozen, omdat deze meerdere keuzes toelaten, terwijl

radiobuttons slechts één item toestaan. Ook een multi-selectbox was geen optie, omdat dit onoverzichtelijk is bij meer resourcesets dan er in de box passen.

Figuur 6 Resourcesoverzicht

Nadat ik de Beheertool had opgeleverd zoals hiervoor beschreven stond, heb ik deze laten testen door de twee gebruikers binnen Webbeat. Deze vonden het prettig werken, maar misten een mogelijkheid om direct bij de resources te komen. Ik ben uitgegaan van een nieuwe applicatie die in het ontwikkelstadium is bij mijn ontwerp, de insteek die zij misten was vanuit het onderhoud. Als er onderhoud gepleegd moet worden aan resources binnen een bestaande applicatie, is het sneller om direct alle resources te tonen en vanuit dat scherm het noodzakelijke onderhoud te plegen.
Na zelf ook een aantal keren met de Beheermodule gewerkt te hebben, kon ik niets anders zeggen dan dat ze gelijk hadden. Ik heb wel bewust ervoor gekozen om de

resources te scheiden per applicatie en resourceset, om op deze manier nog enige structuur in het geheel aan te brengen.
Het filter heb ik ontwikkeld om zo het aantal getoonde resources kleiner te maken en het geheel overzichtelijk te houden voor de gebruiker.

4.7 Publish

Het doel van de publishtool is tweeledig; enerzijds het ophalen van de relevante data uit de database, en deze om te zetten in een XML-bestand, anderzijds het versturen van deze bestanden naar de diverse FTP-locaties.
Doordat een XML-bestand zeer gestructureerd is, is het omzetten van databasegegevens naar XML gemakkelijk. Ik heb dit gedaan door een ASP-script de databasegegevens op te laten halen en datzelfde ASP-script het XML-bestand aan te laten maken, en dat te vullen met de opgehaalde database gegevens volgens onderstaande structuur.

Code:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<resourceset date="12/11/2003" app="office" set="forum" lang="NL">
      <resource>
            <tra_translation res="Close">sluit $$1</tra_translation>
      </resource>
</resourceset>
```

De eerste regel identificeert de XML voor de parser. Op de tweede regel wordt aangegeven op welke datum het bestand aangemaakt is, bij welke set hij hoort, en welke taal. Deze drie attributen worden op het moment niet gebruikt, maar kunnnen van pas komen in toekomstige toepassingen.
De resourcetag geeft aan de parser aan dat er een nieuwe resource staat. In de tra_translation-tag staat de naam van de resource opgenomen als attribuut, en de vertaling tussen de tags. De reden voor deze structuur is de manier waarop het zoeksysteem (Xpath van het XMLDOM-object) zoekt naar resources. Deze zoekt namelijk op een tagnaam, gefilterd op attribuut. Ik heb de resourcetags eromheen laten staan om het systeem flexibel te houden voor toekomstig gebruik. Als er later andere clientfunctionaliteiten toegevoegd worden, zijn deze vrij gemakkelijk in te passen in het huidige systeem.

Het versturen van de bestanden door middel van FTP gebeurd automatisch na het genereren van de XML-bestanden. Hiervoor heb ik gebruik gemaakt van een serverside plugin genaamd Majodio FTP, een plugin die Webbeat al enige tijd in gebruik heeft voor dit soort doeleinden.
Mocht er tijdens het uploaden iets fout gaan, dan wordt dit weggeschreven naar het bestand Errorlog.txt en weergegeven in het scherm door middel van een waarschuwingstekst (bv. "Fout bij het uploaden van bestand office_forum_nl.xml naar omgeving Pro"). Ook wordt er een uitroepteken achter de set geplaatst als visueel geheugensteuntje voor de beheerder dat de set niet goed geupload is.

4.8 Client

De clientmodule bestaat uit het ophalen van de vertalingen uit de XML-bestanden. Dit gebeurd door middel van Xpath, een zoeksysteem van het XMLDOM-object. Door de simpele structuur van het XML-document is de zoektijd van Xpath zeer laag (bij 2000 zoekopdrachten is de totale tijd nog geen 1,5 seconden, terwijl de gemiddelde ASP-pagina van het Virtueel Kantoor slechts 40 resources opvraagt), waarmee dit dus een uitstekend systeem is voor de toepassing binnen de Webbeat-applicaties.
Er worden feitelijk twee verschillende XML-bestanden geladen. De eerste is een global XML-bestand, die de meest voorkomende resources bevat, resources die door de hele applicatie gebruikt worden. De tweede is de set-specifieke XML, met daarin de overige resources die noodzakelijk zijn op deze pagina.
Om deze XML-file te kunnen laden, moet de toepassing weten welke applicatie en welke set er momenteel in gebruik dienen te zijn, aangezien de naam van de XML-file opgebouwd is volgens het stramien `applicatie_resourceset_taal.xml`. Ik had eerst voor de meest makkelijke en voor de hand liggende manier gekozen; sla de applicatie- en resourcesetnaam op in een sessie, en haal deze sessies op wanneer het nodig is. Dat leek goed te gaan, maar sessies vergen vrij veel geheugen, en kunnen na verloop van tijd een zware belasting vormen op de prestaties van de pagina, zeker wanneer deze niet netjes gesloten worden.
Daarom besloot ik om gebruik te maken van strings, die te vullen door middel van een initialisatiefunctie die aangeeft welke applicatie en resourceset gebruikt gaat worden.

De programmeur moet in de code de volgende code opnemen om een resource op te laten halen: `GetResource("resourcenaam")` of, als de resource aan het begin van een zin staat, met `GetResourceCaps("resourcenaam")`.

4.9 Debug

Deze tool was gedefinieerd als luxe-eis, maar gezien de grote overeenkomsten tussen de clienttool en de debugger, was er voldoende tijd om deze te ontwikkelen. De functie van de debugger is het opsporen van onbekende resources, en dit presenteren aan de programmeur danwel beheerder.

De eerste versie van de debugtool bestond uit een aparte tool die alle bestanden uit een bepaalde folder doorzocht naar resources en deze vergeleek met waarden in de database. Kwam deze niet overeen, werd de resource in het rood op het scherm getoond. Later kwam er nog bij dat de resource wel bestond maar in een andere resourceset voor kwam, deze resource werd in het blauw getoond.

Bij het overleg kwam naar voren dat Webbeat liever zag dat de debugger de uitvoer op het scherm toonde in de uiteindelijke omgeving. Daarom heb ik ervoor gekozen om de debugger te integreren in de Clientmodule. Door een string in te stellen wordt de Clientmodule verandert in de Debugmodule.
Dit heeft als voordeel dat de functionaliteiten die overeenkomen met de Clientmodule niet opnieuw geschreven te hoeven worden, en daarmee het onderhoud te bemoeilijken.
De werking verandert dan van het ophalen van de resources in XML-bestanden naar het opzoeken van resources in de database. Als de resource gevonden wordt, dan wordt de vertaling gewoon in het zwart getoond aan de gebruiker. In ieder ander geval wordt de resourcenaam getoond met een kleurcode. De kleurcodes hebben de volgende betekenissen:

Kleur	Kleurcode	Betekenis
Grijsblauw	777D9D	Vertaling staat op {ignore}
Lichtblauw	B1ECFF	Vertaling staat op {use-(taal)
Donkergroen	32941B	Geen vertaling
Roze	FFB1B1	Resource bestaat niet in deze applicatie
Donkerblauw	215781	Resource staat in een andere set

Automatische modus

Vanuit Webbeat was er de wens om bij het aanmaken van een nieuwe applicatie de Debugmodule zelf direct alle resources in de database op te laten nemen. Ik heb hiervoor de oorspronkelijke vorm genomen, en alle javascripts om een resource met de hand toe te voegen ertussenuit gehaald. Vervolgens heb ik de commando's toegevoegd die ervoor zorgen dat de resource in de database geplaatst wordt.
Aan het gebruik van de automatische modus zit één risico vast, namelijk dat programmeurs fouten kunnen maken. Als een programmeur twee dezelfde resources wil aanroepen, maar in één van de twee gevallen een spelfout maakt, zal dit systeem het herkennen als twee verschillende resources.
Dit probleem is theoretisch gezien op te lossen, maar door de beperkte tijd ben ik hier niet aan toegekomen. Het is mogelijk om gevonden resource te vergelijken met resources in de database, en mocht de resource op één van de resources in de database lijken, dan kan er een melding hiervan op het scherm getoond worden. De programmeur kan dan eventueel de code aanpassen.
In een verder uitgebreide versie kan het zelfs zo zijn dat de Debugmodule de sourcecode aanpast, indien de programmeur dit wenst.

4.10 Testen

Ik wilde naast de tesfases die in de Rapid Prototyping methode zijn opgenomen, nog een algehele test uitvoeren. Maar doordat Webbeat tijdens het opstellen van de eisen aangaf dat de Beheermodule een luxe-eis was, en omdat de gebruikers bij de testfase aangaven dat het product prettig werkte, en zij eigenlijk geen verbeteringen aan konden geven, heb ik besloten om deze algemene test niet uit te voeren.
Doordat de Beheermodule klein en overzichtelijk is, is de kans op verborgen fouten klein. Mochten er nog fouten gevonden worden, dan vormen zij geen probleem in het werken, aangezien iedere mogelijke gebruiker verstand van computers heeft, en kennis van SQL-server bezit. Tevens bezit Webbeat de kennis om deze fouten snel op te kunnen lossen.

Deel 3
Evaluatie

5 Evaluatie

In dit hoofdstuk zal een reflectie van de gehele afstudeerperiode opgenomen worden. De werkzaamheden zullen worden afgezet tegen de opdracht, en bekeken in hoeverre deze twee overeenkomen. Verder zijn er nog aandachtspunten en eventuele verbeterpunten opgenomen die in de toekomst meegenomen kunnen worden.

5.1 Productevaluatie

Adviesrapport Multilingual Solutions

Het enige probleem wat ik tegenkwam bij het schrijven van dit adviesrapport was het gebrek aan bruikbare informatie. Ik had wel verwacht dat het moeilijk zou zijn om bruikbare informatie te vinden, maar dat er zo weinig zou zijn viel me erg tegen. Dit was een leerzame ervaring voor mij, tijdens de opleiding heb ik eigenlijk alleen gezocht naar onderwerpen waar veel concrete informatie te vinden is. Ik heb nu de andere kant gezien, en geleerd hoe ik uit productbeschrijvingen af kan leiden hoe een product technisch in elkaar steekt.
Als naar het product kijk, vind ik dat ik de gevonden bronnen optimaal gebruikt heb om een goed onderbouwd advies aan Webbeat te kunnen geven.

Eisenomschrijving

Zoals gezegd, bij het gebruik van Rapid Prototyping is het heel goed mogelijk dat een project eindeloos door blijft gaan, doordat bij iedere iteratie het product uitgebreider wordt. Door een lijst op te stellen met eisen, wordt het project afgebakend, en makkelijker te beheersen.
Ik denk dat dit product in dat opzicht geslaagd is. Het eindproduct is niet groter geworden dan oorspronkelijk bedacht, en pas nadat alle eisen ingebouwd waren werd er gekeken naar wat er nog meer mogelijk was.
Het opstellen van de eisen was eenvoudig, doordat Webbeat vrij helder voor ogen had wat de feitelijke functie van het eindproduct zou moeten zijn. Ik hoefde deze beschrijvingen alleen maar samen te vatten in een puntsgewijs lijstje.
Ik denk dat de brainstormsessies gestructureerder hadden kunnen verlopen, maar doordat de opdracht voor Webbeat vrij helder was, denk ik dat dit ten koste zou zijn gegaan van het creatieve proces wat doorlopen wordt tijdens een brainstormsessie.

Ontwerprapport

Bij het opstellen van het ontwerprapport was het eisenpakket de leidraad. Doordat het eisenpakket helder was, was het opstellen van een ontwerprapport eenvoudiger. De enige moeilijkheden kwamen pas naar voren bij het opstellen van het database-ontwerp. Dit was te verwachten, aangezien databases niet mijn sterkste kant is. Dit was een heel bewust proces, terwijl bijvoorbeeld het ontwerp van de GUI heel vlot en intuïtief ging. Toch denk ik dat het database-ontwerp goed is, aangezien de database momenteel in gebruik is door de MLT en er geen problemen zijn ontstaan.

Prototypes van de modules

De prototypes waren over het algemeen niet moeilijk om te ontwikkelen, omdat het eigenlijk kleine elementen die goed te overzien waren. De ontwikkeling van elke module begon heel simpel, en als dat functioneerde, werd de volgende functie erin gevoegd. Dit gaf ook snel toonbare resultaten.
De beheermodule was de enige module waarvan ik het prototype eerst in Photoshop gemaakt had. Ik vind het prettiger om een GUI eerst in Photoshop te ontwerpen, je kan direct aanpassingen maken, zonder met code te hoeven rommelen.
Ik ben tevreden over de prototypes, zij lieten goed zien hoe de MLT zou gaan werken. Ook op functioneel vlak zijn ze dusdanig werkzaam, dat zij direct in gebruik genomen konden worden.

XML-omgeving die multitaal ondersteuning biedt aan applicaties

Deze XML-omgeving is tijdens de opdracht hernoemd tot de Client-module. Deze Client-module bleek eenvoudiger te programmeren dan ik verwachtte. Ik dacht dat het ophalen van gegevens uit een XML-bestand veel moeilijker zou zijn dan dat gebleken is.
Als ik bekijk hoe deze Client-module functioneert, denk ik dat deze aan alle vooraf gestelde eisen voldoet. De verloren snelheid bij het starten van een pagina ten opzichte van diezelfde pagina waar geen MLT geïmplementeerd is, is minimaal. De gebruikers zullen hier dus geen problemen van ondervinden.
Minder tevreden ben ik over de module die de XML-bestanden genereert op basis van de database; de Publish-module. Op zich functioneert deze prima, maar bij het publiceren worden de XML-bestanden naar alle locaties verzonden, maar zijn deze niet direct beschikbaar in de ontwikkelomgeving. Om deze beschikbaar te maken is een menselijke actie noodzakelijk. Dit zou verholpen kunnen worden door de locatie

waar de XML-bestanden aangemaakt worden te veranderen, maar dit is vrijwel onmogelijk, aangezien het systeem niet weet in welke folder een applicatie staat. Momenteel is de MLT geïmplementeerd in het virtueel kantoor, dit werkt allemaal naar behoren, dus ik denk dat dit product geslaagd is.

5.2 Procesevaluatie

Methodiek en techniek

Voor ik aan dit project begon, wist ik niet waar ik aan ging beginnen. De opdracht was nog niet echt concreet voor mij, en de projectmethode was mij totaal onbekend. Aan het begin van de opdracht heb ik dan ook eerst onderzoek gedaan naar de mogelijkheden van XML, Rapid Prototyping en bestaande methodes van multi-lingual solutions. Ik denk dat dit een goed en logisch begin was. Zo kreeg ik grip op het project en daarmee ook zicht op het eindproduct.

Dit was het eerste project dat ik doorlopen heb met de Rapid Prototyping methode en ik denk dat dit een prima methode is om toe te passen in kleinere projecten. Als ik terugkijk op dit project, denk ik dat ik in tijdnood was gekomen als ik de IAD-methode gebruikt zou hebben, door het schrijven van de vele en veelal gedetailleerde rapporten.

Het enige waar ik wel moeite mee had, was de afweging die bij iedere iteratie gemaakt moest worden; is een idee een verbetering van de huidige functie of een compleet nieuwe functie die niet in de eisenomschrijving staat. Als het om een nieuwe functie ging, kwam ook nog eens de afweging of het wel of niet tijdens de afstudeerstage in de MLT geïmplementeerd zou worden, of niet.

Door alleen de kleine functies erbij te nemen, zoals het opnemen van de creatiedatum in het XML-bestand, bleef het project hanteerbaar, en werd de werking van de MLT net even beter. Ideeën die een grote impact hadden op het ontwerp, werden wel genoteerd, maar werden niet doorgevoerd in het ontwikkelproces. Ik denk dat dit er aan bijgedragen heeft dat het product op tijd klaar was en dat het goed functioneert.

Tijdens het proces had ik het gevoel dat ik een volwaardig lid van het team van Webbeat was, omdat er niet wekelijks een controle was op m'n vorderingen. Er werd op vertrouwd dat ik mijn werk zou doen, en dat ik dat naar mijn best op zou doen. Dit deed mij erg goed, waardoor ik ook gemotiveerd werd.

Planning

Het opstellen van de planning was in het begin zeer moelijk, omdat ik nog geen overzicht over het project had. Ik wist zelfs nog niet eens zeker of het te doen was binnen de gestelde tijd. Nadat ik meer overzicht over het project had gekregen, was het opstellen van de planning nog steeds vrij lastig, omdat ik in moest schatten hoeveel tijd ik nodig zou hebben voor de specifieke onderdelen. Aangezien ik anderhalf jaar niet intensief bezig ben geweest op het gebied van programmeren, was het inschatten van de benodigde tijd een kleine gok.

Achteraf gezien moet ik zeggen dat de mate waarin ik programmeer- en ontwerptechnieken hanteer me nog behoorlijk meegevallen is. Hiedoor liep ik ook licht voor op de planning, wat natuurlijk een goed gevoel geeft, zeker bij een project waarvan je aan het begin nog niet concreet weet wat je nu precies gaat doen.

Literatuurlijst

Internet:

Google	www.google.nl
XML-pagina	xml.pagina.nl
ASP 101	www.asp101.com