Auteur: W. G. van den Eijkel Datum: Mei 2001

Smart Proxy

Afstudeerscriptie

KPN Research

Informatieblad bij Rapport

Titel:	Smart Proxy
Excerpt:	
Auteurs:	W. G. van den Eijkel
Reviewers:	
Afdeling:	Services Development, Service Prototyping
Project:	KPN Homeservices
Projectleider:	Jans Aasman
Projectnummer:	
Programma:	
Programma-manager:	
Opdrachtgever:	
Datum:	mei 2001

Uitgegeven onder verantwoordelijkheid van:

Trefwoorden: Family Proxy, recommendation engine, aanbevelingen, Webpad, labelen, labelserver, TFIDF, matching, filteren, Filtertechnieken

Voorwoord

Dit rapport is geschreven in het kader van mijn afstudeerproject van mijn studie Technische Informatica aan de Technische Universiteit Delft.

Dit rapport is in eerste instantie bedoeld als afstudeerscriptie voor mijn studie. Ten tweede is het bedoeld voor de projectmedewerkers van het project KPN Homeservices dat uitgevoerd wordt binnen KPN Research.

Bij deze wil ik de mensen bedanken die mij hebben geholpen bij het tot stand brengen en uitvoeren van dit onderzoek. In het bijzonder wil ik Leon Roos van Raadshooven, Alan Verberne, Andries Hekstra, Jans Aasman, Arjen Vollebregt bedanken die mij tijdens mijn onderzoeksperiode bij KPN Research enorm geholpen hebben.
Ook wil ik mijn docent Leon Rothkrantz bedanken voor zijn opbouwende kritiek op mijn onderzoek.
Last but not least wil ik mijn verloofde Drs. Linda van der Toorn bedanken die mij geholpen heeft om van mijn scriptie een duidelijk en vooral leesbaar verhaal te maken.

Wouter van den Eijkel, mei 2001

Inhoud

Samenvatting

De aanleiding van het onderzoek is het project KPN Homeservices van KPN Research. In dit project wordt een Family Proxy ontwikkeld om het internetten voor een gebruiker sneller te maken dan normaal. Om dit te bereiken worden er webpagina's in de Family Proxy ge-precached. Het probleem is het bepalen van welke webpagina's er in de Family Proxy moeten worden ge-precached. In dit onderzoek wordt er een methode voorgesteld en uitgewerkt om de cache van de Family Proxy te vullen. De methode die wordt gebruikt zal aanbevelingen genereren voor gebruikers op basis van hun interesses.

Hiervoor is uitgegaan van de volgende probleemstelling:

Welke methoden en technieken kunnen er worden gebruikt om webpagina's aan te bevelen aan Internetgebruikers (om ze uiteindelijk te kunnen pre-cachen) en hoe worden de aanbevelingen van de methoden ten opzichte van elkaar beoordeeld

De methoden en technieken op basis waarvan het prototype is gebouwd zijn de volgende:

- Filtertechnieken
- Labeltechnieken
- Verfijningmethoden
- Matchingtechnieken

Allereerst is hiervoor een keuze gemaakt tussen een drietal methoden van filteren, te weten Time Series Analysis, Collaborative Filtering en Filteren op basis van inhoud.
Hierbij is gebleken dat filteren op basis van inhoud het meest geschikt is. Aan de hand van deze methode kunnen aanbevelingen worden gegenereerd op basis van de inhoud van de webpagina's. Er wordt gekeken naar het 'onderwerp' van de webpagina, door de voorkomens van woorden op de webpagina te tellen. Het onderwerp van de webpagina wordt vervolgens vergeleken met een profiel van een gebruiker om te kijken of deze aansluit bij de interesses van de gebruiker.

Vervolgens zijn een aantal technieken besproken om van de inhoud van een webpagina een beschrijving te maken op basis waarvan op een later tijdstip matching plaats kan vinden. Hiervoor is de volgende methoden beschreven, te weten labelen op basis van tekstinhoud, labelen op basis van HTML-opmaak, labelen op basis van context en labelen op basis van Webpadcategorieën.
Voor het prototype is ervoor gekozen om geen gebruik te maken van het labelen op basis van context. Dit omdat er voor het labelen, een beschrijvingen gemaakt moeten worden van een bepaalde webpagina. In het geval van labelen op basis van context is het noodzakelijk om links te hebben op andere webpagina's die naar de betreffende webpagina wijzen. Deze waren in het onderzoek niet voorhanden.
De overige methoden bleken binnen het kader van dit onderzoek goed bruikbaar voor het labelen van webpagina's. Bij labelen op basis van tekstinhoud worden er op basis van platte tekst, woorden geteld. Aan de hand hiervan wordt een vector gecreëerd die aangeeft hoe vaak een woord op de webpagina voorkomt. Op deze manier wordt een eenvoudige handreiking geboden in het vergelijken van webpagina's met gebruikersprofielen. Met behulp van de HTML opmaak van een webpagina is gekeken naar woorden die een bepaalde nadruk hebben. Hierbij gaat het bijvoorbeeld om woorden die visueel opvallen doordat ze bijvoorbeeld vet zijn, cursief gedrukt zijn of van een groter lettertype zijn. Met behulp van de Webpadcategorieën is gekeken naar de

beschrijvingen die de gebruikers aan hun bookmarks hebben gegeven. Op deze manier hebben de gebruikers zelf als het ware al een label meegegeven aan een bepaalde categorie van webpagina's. Er is voor gekozen om in het prototype deze drie technieken te combineren.

Er zijn ook een aantal verfijningtechnieken toegepast. Uit de beschreven verfijningtechnieken zijn er drie gekozen die zijn toegepast in het prototype, namelijk het filteren van stopwoorden, het toepassen van stemming en het uitvoeren van TFIDF.
Het filteren van stopwoorden zorgt ervoor dat woorden zoals 'de, het, een', woorden die niets zeggen over het onderwerp van de pagina, uit de beschrijving van de webpagina worden gefilterd. Stemming zorgt ervoor dat meervouden en enkelvouden van een woord als hetzelfde woord worden gezien. Hierbij wordt ook rekening gehouden met vervoegingen van zelfstandige naamwoorden. Hiervoor er gebruik gemaakt van de CELEX-database. TFIDF tot past de gewichten van de termen in de beschrijving van een webpagina aan, door te kijken naar het globaal voorkomen van de woorden in alle (voor het onderzoek) beschikbare documenten. Een woord wat in de beschrijving van veel webpagina's voorkomt, zegt minder over die webpagina dan een woord dat maar op een enkele webpagina voorkomt.

Om de inhoud van een pagina te vergelijken met een gebruikersprofiel is matching toegepast. Hiervoor is uitgegaan van de methode welke de hoekafstand van de vectoren berekent. Voor het resultaat hiervan geldt dat hoe kleiner de hoek, hoe groter de overeenkomst tussen de vector(en) en de interesses van de gebruiker.

Op basis van deze methoden is en prototype ontwikkeld dat geschreven is in LISP. Normaliter spelen Webpadcategorieën een belangrijke rol binnen het Family Proxy project voor het afleiden van gebruikersinteresses. Bij het gebrek aan gebruikers die beschikking hadden over een Webpad ten tijde van het onderzoek, zijn de Webpadcategorieën van een gebruiker gesimuleerd voor het gebruikersonderzoek.

Het gebruikersonderzoek betrof zo'n 600 personen binnen een redelijk homogene groep mensen. Hiervan hebben er uiteindelijk 62 meegewerkt aan het uitvoeren van het onderzoek. Gebruikers hebben hiervoor webpagina's opgegeven voorzien van steekwoorden. De webpagina's zijn gebruikt als input voor het onderzoek. Op basis van deze webpagina's zijn een vijftal sets van aanbevelingen gegenereerd, te weten:

- Random aanbevelingen (compleet willekeurig uit de set van aanwezige pagina's)
- Aanbevelingen op basis van gebruikerslabels (op basis van Webpadcategorieën)
- Aanbevelingen op basis van computergegenereerde labels (tekstinhoud en HTML-opmaak)
- Aanbevelingen op basis van zowel gebruikerslabels als computergegenereerde labels
- Aanbevelingen op basis van de Internet zoekmachine AltaVista (zoeken op Internet aan de hand van de interesseprofielen van gebruikers)

Uiteindelijk werd de set, welke gebaseerd is op het doen van aanbevelingen op basis van het gebruikerslabel, door de testpersonen het beste gewaardeerd.

Met het onderzoek is echter niet te achterhalen hoe de gebruikers de verschillende aanbevelingen binnen een set hebben gewaardeerd. Tevens bleek de taal waarin de webpagina is geschreven een probleem voor sommige gebruikers. Ook blijkt dat de aanbevelingen die gegenereerd zijn op basis van een kleine set URL's (de door gebruikers bezochte webpagina's) beter scoren dan aanbevelingen die gebaseerd zijn op een groot deel van het Internet. Hierbij dient rekening te worden gehouden met het optreden van het 'Garbage in, Garbage out' effect, waardoor de kwaliteit van de aanbevelingen die het systeem genereert zou kunnen afnemen.

Tot slot is het gebleken dat het moeilijk is om de interesses van de gebruiker te achterhalen zonder dat de gebruiker hier direct om gevraagd wordt. De in het onderzoek gebruikte methode, waarbij gebruikers URL's met bijbehorende steekwoorden opgeven, blijkt goed te werken en vergt minimale inspanning van de gebruikers. Toch is het belangrijk dat men inspanning aan de kant van de gebruiker probeert te vermijden.

Veel gebruikers zullen geen benul hebben van het nut van de bookmarks die ze aanmaken en de omschrijving die ze hieraan meegeven. Beide zijn uiteindelijk van groot belang voor de kwaliteit van de aanbevelingen die ze terug zullen krijgen. Het is daarom van groot belang dat, indien een dergelijk project op grote schaal gaat worden uitgevoerd, er een goede communicatie hierover naar de eindgebruiker moet plaatsvinden. Met het besef van het belang van zijn eigen rol en invloed in dit proces, staat of valt het succes van het doen van aanbevelingen.

Lijst van afkortingen

ADSL	Asymmetric Digital Subscriber Line
DF	Document Frequency
DFD	Dataflow Diagram
ERD	Entiteit Relatie Diagram
FTP	File Transfer Protocol
HTML	HyperText Markup Language
IDF	Inverse Document Frequency
IP	Internet Protocol
ISP	Internet Service Provider
KPN	Koninklijke PTT Nederland
LISP	List Processor
LSI	Latent Semantic Indexing
Mbps	Megabit per seconde
PDA	Portable Digital Agents
PDDP	Principal Direction Divisive Partitioning
PLOB	Persistent LISP Objects
SVD	Singular Value Decomposition
SQL	Structured Query Language
TFIDF	Term Frequency Inverse Document Frequency
URL	Unified Resource Locator
WAP	Wireless Application Protocol

1 Inleiding

De geschiedenis van het Internet gaat een flinke dertig jaar terug. In 1969 werden de eerste testen gedaan met ARPANET, de voorloper van het Internet zoals wij dat nu kennen. Hieronder staat een transcriptie van één van de eerste testen van computer communicatie.

> "We set up a telephone connection between us and the guys at **SRI**...," Kleinrock ... said in an interview: "We typed the **L** and we asked on the phone,
> "Do you see the **L**?"
> "Yes, we see the **L**," came the response.
> "We typed the **O**, and we asked, "Do you see the **O**."
> "Yes, we see the **O**."
> "Then we typed the **G**, and the **system crashed**"...
> Yet a **revolution had begun**"...
>
> **Source**: *Sacramento Bee*, May 1, 1996, p.D1

In de afgelopen dertig jaar is er een heleboel veranderd. Het netwerk is gegroeid van een paar computers naar miljoenen computers. Daardoor is de hoeveelheid informatie die op het Internet is op te vragen, enorm in omvang gestegen. Ook is het aantal gebruikers in de loop der jaren enorm toegenomen. Al deze mensen vragen informatie op van het Internet waardoor het dataverkeer toe is genomen. Vooral de laatste jaren heeft de groei van het dataverkeer een enorme vlucht genomen. Door deze enorme groei is een aantal problemen ontstaan.

1.1 Aanleiding tot het onderzoek

Het Internet is, zoals in de inleiding van dit hoofdstuk al is aangegeven, enorm gegroeid. Daardoor heeft het Internet te kampen met een aantal problemen. Twee van deze problemen zijn:

- Informatie op Internet is moeilijk te vinden
- Het Internet is vaak langzaam

De oorzaak van het eerste probleem ligt in het feit dat het Internet zo enorm groot is.

Het Internet is de afgelopen jaren sterk gegroeid en de verwachting is dat Internet de komende jaren alleen maar meer zal groeien. Volgens onderzoek groeit het Internet op dit moment met bijna 2 miljoen webpagina's per dag [19]. Er wordt zelfs verwacht dat in 2002 het aantal webpagina's de 8 miljard passeert.

De informatie die de gebruiker zoekt, zal hoogstwaarschijnlijk wel ergens op Internet staan. Het probleem voor de gebruiker is echter: hoe vindt hij deze informatie?

Op dit moment maken gebruikers vaak gebruik van verschillende methoden om informatie te zoeken op het Internet. Veel mensen gebruiken hiervoor één van de vele zoekmachines op Internet. Een bepaalde zoekmachine heeft echter maximaal 30% [25] van alle aanwezige webpagina's op het Internet in zijn database staan, waardoor een

gebruiker slechts het topje van de ijsberg zal kunnen doorzoeken met de betreffende zoekmachine.
Ook de hedendaagse media speelt handig in op de Internettrends. In verschillende media (tijdschriften, radio, tv) worden Internetsites aan gebruikers aanbevolen. Er zijn zelfs speciale bladen met daarin alleen maar beschrijvingen en adressen van Internetsites.
Tevens proberen veel mensen via nieuwsgroepen aan de benodigde informatie te komen. Zoeken in een bepaalde categorie levert dan vaak relevante informatie op.
Veel interessante sites worden daarentegen in veel gevallen nog steeds gevonden door aanbevelingen van familie, vrienden of collega's. Zij kennen in veel gevallen de voorkeuren en interesses van de betreffende persoon. De aanbevelingen die zij doen, zijn meestal veel meer gericht op de individuele wensen van de persoon dan aanbevelingen die via de diverse hiervoor genoemde media worden gedaan.
Uit al deze voorbeelden blijkt dat er een toenemende behoefte is aan informatie over informatie op Internet.

Het tweede genoemde probleem, de traagheid van het Internet, heeft een aantal oorzaken.
Ten eerste is de verbinding van de gebruiker naar het Internet vaak heel traag. Hierdoor duurt het lang voordat informatie is binnengehaald. Wanneer de verbinding van de gebruiker naar het Internet wel snel is, hoeft dit echter nog niet te betekenen dat de pagina’s snel binnen gehaald zullen worden. Dit heeft te maken met beperkingen in de capaciteit van de verbindingen tussen de Internet Servers. Als de Server, waar een bepaalde webpagina vandaan moet komen, een langzame verbinding naar het Internet heeft, zal deze verbinding de bottleneck vormen.
Hierin ligt tevens een tweede oorzaak van de traagheid van het Internet. Door de beperkte capaciteit van verbindingen tussen Internetservers raken deze verbindingen verstopt. Hierdoor duurt het lang voordat de opgevraagde informatie bij een gebruiker is.
Ook de snelheid van de Internet Servers zelf is een factor die van invloed is op de snelheid van het Internet of in dit geval de traagheid. Wanneer de Server niet snel genoeg is om de aanvragen van de gebruikers af te handelen, zal dit resulteren in een trage verbinding.
Tot slot is het enorme aantal mensen dat gebruik maakt van het Internet een vertragende factor. Dit zorgt er in combinatie met de beperkte bandbreedte voor dat het Internet traag wordt.

1.2 Probleemstelling en doelstelling

Het Internet is een enorm netwerk met een schat aan informatie. Het probleem van het Internet is dat het over het algemeen langzaam is. Het duurt lang voordat een webpagina van het Internet is binnengehaald. Dit komt vaak door een beperkte bandbreedte en het enorme aantal mensen dat gebruik maakt van het Internet. Ook is het voor veel mensen moeilijk om de gewenste informatie te vinden. Het is voor een gebruiker niet altijd duidelijk waar hij moet zoeken en hoe. Voor veel mensen die nog niet zo bekend zijn met Internet is dit vaak één van de redenen dat ze Internet niet willen gebruiken. Ze kunnen gewoon niet vinden wat ze zoeken.

De Family Proxy in het project van KPN Homeservices biedt de gebruiker de mogelijkheid om het Internet gebruikersvriendelijker, sneller en meer gepersonaliseerd te maken.
Eén van de manieren om het Internet te personaliseren is door de gebruiker te helpen met zoeken naar informatie op Internet in de vorm van aanbevelingen.

Dit leidt tot de volgende probleemstelling:

Welke methoden en technieken kunnen er worden gebruikt om webpagina’s aan te bevelen aan Internetgebruikers (om ze uiteindelijk te kunnen pre-cachen) en hoe worden de aanbevelingen van de methoden ten opzichte van elkaar beoordeeld

De volgende doelstelling is gedefinieerd:

Het bouwen van een prototype waarmee aanbevelingen voor gebruikers kunnen worden gegenereerd.

Het doel van dit onderzoek is het bedenken en uitvoeren van een methode die het mogelijk maakt om webpagina's te pre-cachen zodat het Internet voor de gebruiker sneller lijkt.

Voor het onderzoek zijn een aantal onderzoeksvragen opgesteld waarin in dit onderzoek antwoord op wordt gegeven. Deze onderzoeksvragen luiden als volgt:

- Welke filtermethoden zijn geschikt om te bepalen welke informatie in een Family Proxy moet komen?
- Welke methoden zijn er om een goede beschrijving van een webpagina te maken?
- Welke informatie is nodig om deze beschrijvingen te maken?
- Welke methoden zijn er om voorspellingen voor webpagina's te doen?
- Hoe kunnen aanbevelingen van webpagina's voor gebruikers worden gegenereerd?
- Hoe goed zijn deze aanbevelingen?
- Hoe zijn de resultaten te verklaren?

Het onderzoek heeft als hoofddoel het uitdenken en testen van een systeem om aanbevelingen van webpagina's te doen aan gebruikers. Er moet rekening gehouden worden met het feit dat deze techniek toegepast moeten kunnen worden om de proxycache van de Family Proxy te vullen. Om de aanbevelingen te genereren zal er ook worden gekeken naar het afleiden van gebruikersprofielen. Er is hierbij niet gekeken naar de beste methode om interactie met de gebruiker te realiseren of de beste methode om de aanbevelingen te presenteren. Er is alleen gekeken naar een goede methode om de interesses van de gebruiker impliciet af te leiden.

In het onderzoek is de volgende oplossingsmethodiek gekozen:

- Selecteer eerst op basis van literatuur en vergelijkend onderzoek veelbelovende methoden
- Analyseer en ontwerp vervolgens een systeem dat gebruik maakt van de geselecteerde methoden
- Implementeer een prototype van het systeem
- Test het prototype met behulp van een gebruikerstest

1.3 Opbouw scriptie

Deze scriptie is in vier delen opgedeeld.

- De introductie (Hoofdstuk 1 tot en met 3)
- Het theoretische deel (Hoofdstuk 4 tot en met 8)
- Het praktijk deel (Hoofdstuk 9 tot en met 11)
- De conclusie en aanbevelingen (Hoofdstuk 12)

In het eerste deel van deze scriptie wordt het doel van het onderzoek duidelijk gemaakt. De probleemstelling en doelstelling worden in hoofdstuk 1 (dit hoofdstuk) uiteengezet. Vervolgens wordt in hoofdstuk 2 kort ingegaan op het bedrijf en de afdeling waar dit onderzoek is uitgevoerd. Daarna wordt in hoofdstuk 3 nog een plaatje geschetst van het project waar dit onderzoek deel van uit maakt.
Het tweede deel van de scriptie beschrijft de theorie van de technieken en methoden die dienen als onderbouwing voor het onderzoek. In hoofdstuk 4 wordt het principe van een Proxy Server uitgelegd. In het daaropvolgende hoofdstuk wordt een aantal filtertechnieken besproken. In hoofdstuk 6 wordt de filtermethode die voor dit onderzoek is gebruikt, verder uitgewerkt. Hoofdstuk 7 gaat over matching en hoe dit in dit onderzoek

is toegepast. Als afsluiting van het theoriedeel wordt in hoofdstuk 8 clustering behandeld. Het derde deel gaat in op het prototype dat gebouwd is. In hoofdstuk 9 wordt de analyse van het prototype behandeld. Daarna wordt in hoofdstuk 10 het ontwerp uitgewerkt. In hoofdstuk 11 wordt het gebruikersonderzoek dat uitgevoerd is om het prototype te testen beschreven.
Het laatste deel behandelt de conclusies en aanbevelingen. Deze worden in hoofdstuk 12 uitgewerkt.

2 KPN

Dit hoofdstuk zal een korte introductie geven over het bedrijf KPN en KPN Research waarbinnen dit onderzoek is uitgevoerd.

2.1 Het bedrijf KPN

KPN is één van de grootste aanbieders van telecommunicatie en van op telecommunicatie gebaseerde diensten en producten in Nederland. KPN wil zich in de toekomst richten op de terreinen waarop KPN kan uitblinken en waarin grote groei te verwachten is. KPN heeft daarom voor vier speerpunten gekozen. Deze speerpunten zijn: diensten via het vaste netwerk, diensten via het mobiele netwerk, datacommunicatie op basis van het Internet Protocol (data/IP) en Internet, call centra en mediadiensten.

2.1.1 Diensten via het vaste netwerk

Het vaste netwerk wordt steeds meer gebruikt voor het verzenden van Internet data. Bij diensten over het vaste netwerk moet vooral gedacht worden aan Internet aansluitingen. Voor deze aansluiting biedt KPN een aantal mogelijkheden waaronder de vertrouwde analoge telefoonlijn, ISDN en binnenkort ook ADSL.
ADSL staat voor Asymmetric Digital Subscriber Line. Asymmetric houdt in dat de upstream- en downstream snelheid niet gelijk zijn. De downstream snelheid (naar de gebruiker toe) is maximaal 9 Megabit per seconde (Mbps), de upstream (het internet op) is maximaal 2Mbps. Voorlopig worden de snelheden beperkt tot 1Mbps downstream en 256Kbps upstream. Dit is veel sneller dan de huidige analoge telefoonlijnen (56.6Kbps) of ISDN (128Kbps).

2.1.2 Diensten via het mobiele netwerk

KPN Mobiel is in Nederland met 3,5 miljoen klanten marktleider op het gebied van telecommunicatie. KPN mobile levert diensten op basis van mobiele netwerken. Hieronder valt de mobiele telefonie en alle diensten die daar bij horen zoals bijvoorbeeld voicemail.

2.1.3 Datacommunicatie op basis van het Internet Protocol (Data/IP)

Met het kerngebied Data/IP richt KPN zich op de technieken die nodig zijn voor het versturen van data over het Internet. De behoefte aan datacommunicatie is de laatste jaren zeer sterk toegenomen. Om opstoppingen in de infrastructuur door het groeiende IP-verkeer te voorkomen, is een zeer grote capaciteit nodig voor het transport van al deze gegevens. KPN ontwikkeld en exploiteert in samenwerking met KPN Quest een groot glasvezelnetwerk in Europa om aan de enorme vraag naar bandbreedte te kunnen voldoen.

2.1.4 Internet, call center en mediadiensten

Voor het grootste gedeelte bestaat deze speerpunt uit Internet dienstverlening. Hieronder wordt onder andere verstaan het bieden van Internet toegang (Internet Service Providers,

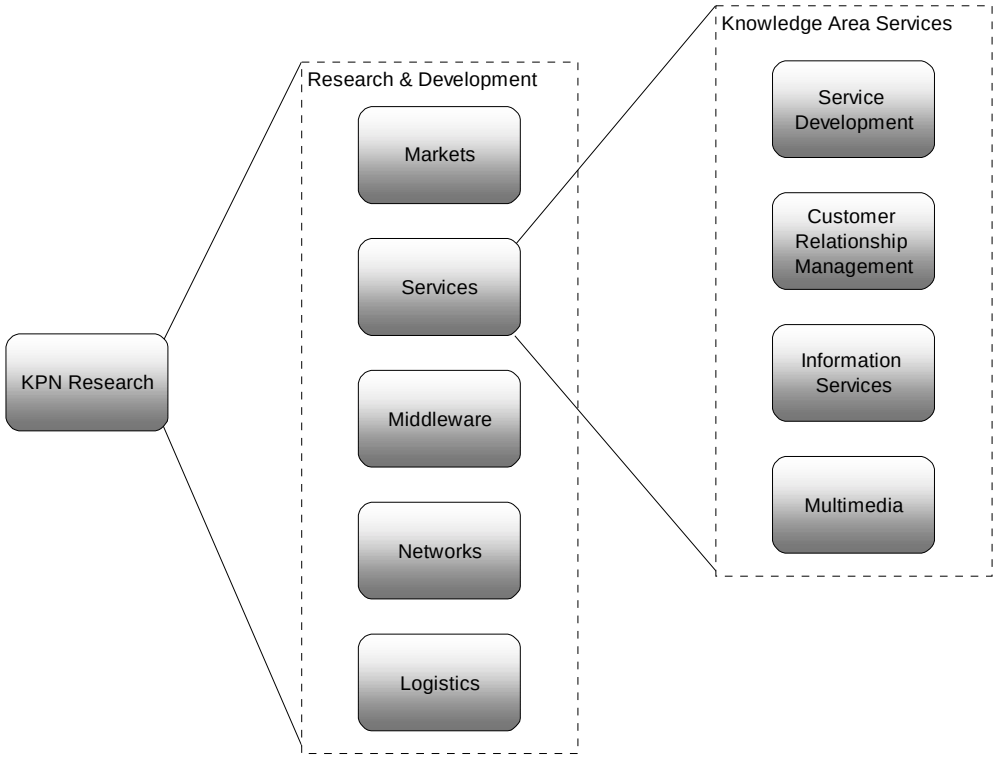
ISP's) en het leveren van specifieke diensten voor het elektronisch zaken doen (e-commerce).

Om in elk van deze disciplines de beste te zijn, te blijven of te worden moet er veel geld en tijd in onderzoek gestoken worden. KPN heeft een aparte faciliteit die deze researchtaken kan uitvoeren, KPN Research.

2.2 KPN Research

KPN Research is de onderzoeks- en ontwikkelingsorganisatie van KPN. KPN Research richt zich met name op de gebieden Internet, mobiele communicatie, bedrijfsnetwerken, vernieuwing in het vaste net en de internationale kansen van KPN. Met de ondersteuning van KPN Research kunnen de belangrijkste klanten KPN Telecom en TNT Post Groep, tegen de laagst mogelijke kosten, snel nieuwe diensten en producten op de markt brengen.

KPN Research bestaat uit een aantal afdelingen die zich elk op een bepaald onderdeel richten. Het organisatieschema van KPN Research met de verschillende afdelingen die hierbinnen vallen, is weergegeven in Figuur 1. In de figuur is de afdeling Services verder uitgelicht omdat dit de afdeling is waarbinnen dit onderzoek heeft plaatsgevonden. In de volgende subparagrafen zullen de verschillende afdelingen kort worden besproken.



Figuur 1. Organisatieschema KPN Research

2.2.1 Markets

De afdeling Markets is opgesplitst in vier kennis gebieden. Het eerste kennisgebied, Society Understanding, richt zich op het uitwerken van toekomstscenario's, het analyseren van lange termijn ontwikkelingen.

Het kennisgebied Consumer Understanding doet onderzoek naar individuele behoeften van de klanten. Op basis hiervan worden nieuwe concepten bedacht.

Het kennisgebied Organisation Understanding richt zich met name op de informatie- en communicatiediensten voor groepen. Met deze kennis worden innovatieve concepten en tools voor groepscommunicatie ontwikkeld.

Het laatste kennisgebied binnen markets is Business Modelling. Dit kennisgebied houdt zich bezig met de ontwikkeling en toepassing van kwantitatieve modellen die de effecten van in- en externe ontwikkelingen in kaart kunnen brengen.

2.2.2 **Services**

Het kennisgebied Service Development richt zich op het ontwikkelen van diensten in de brede zin van het woord, met een focus op bedrijfscommunicatie, prototyping en systeem kennis. Dit houdt in dat men zich bezig houdt met het vertalen van wensen van het bedrijfsleven naar technische mogelijkheden en andersom. Tevens het vertalen van de dienstenwens naar een concreet product.

Binnen de afdeling Services bevindt zich ook het kennisgebied Customer Relationship Management. CRM innoveert bij en voor KPN op het gebied van klantrelaties. Binnen CRM worden nieuwe veelal Internet georiënteerde diensten bedachten en vaak ook ontwikkeld van concept tot prototype.

Het kennisgebied Information Services richt zich om het ontwikkelen van nieuwe diensten en dienstideeën en het aanleveren van onderdelen voor het beter functioneren van bestaande diensten.

Het kennisgebied Multimedia richt zich op drie gebieden. Ten eerste de spraak- en taaltechnologieën. Dit gebied probeert kennis en ervaring op met spraak- en taaltechnologie. Ten tweede de Multimedia Technologie. Het doel van dit gebied is het in de markt zetten van nieuwe multimedia diensten en het verbeteren van estaande diensten. Ten slotte is er het gebied Multimedia kwaliteit. Het doel van dit gebied is het plannen en meten van de perceptieve (zoals door de gebruiker waargenomen) kwaliteit voor spraak, audio en video en de relatie van de perceptieve kwaliteit tot het onderliggende netwerk.

2.2.3 **Middleware**

De afdeling Middleware is ook weer onderverdeeld in een aantal kennisgebieden. Het kennisgebeid Telecommunication Management houdt zich bezig met het mogelijk maken van diensten over telecommunicatie netwerken met een zo hoog mogelijke kwaliteit en tegen zo laag mogelijke exploitatiekosten en toekomstige investeringskosten, waarbij een snelle Time To Marken van nieuwe diensten als een kritische succesfactor wordt gezien.

Het kennisgebied Security houdt zich bezig met de beveiliging van het netwerk en de diensten die hierop werken.

Het kennisgebied Internet Technologie houdt zich bezig met het bouwen van generieke Service platformen. Op deze platformen kunnen de klanten (KPN Telecom) eenvoudig diensten bouwen.

Het laatste kennisgebied binnen Middleware is Information Technologies. Dit kennisgebied richt zich op het configureren, onderhouden, integreren en schalen van applicatie servers.

2.2.4 **Networks**

De afdeling Networks richt zich op de verschillende aspecten en soorten netwerken die door KPN gebruikt worden om diensten aan te bieden. Binnen de afdeling Networks bevinden zich een aantal kennisgebieden.

Het kennisgebied Fixed Acces Networks richt zich op breedband technologieën zoals ADSL en glasvezel. Binnen het kennisgebied wordt ook gekeken aan welke randvoorwaarden moet worden voldaan om een breedband in-huis netwerk met verschillende elementen en transmissietechnieken tot een werkend geheel te kunnen maken.

In het kennisgebied Fixed Core Networks wordt gewerkt aan innovaties in het nationale en internationale core netwerk (het netwerk tussen de centrales) van KPN en KPNQuest. Het kennisgebied Network Planning adviseert over de wijzigingen van bestaande netwerken en hoe de beschikbare capaciteit zo efficiënt mogelijk benut kan worden.

Ten slotte is er binnen de afdeling Networks nog het kennisgebied Mobile Networks welke zich richt op de aspecten van GSM, UMTS en mogelijke nieuwe generaties mobiele netwerken.

2.2.5 Logistics

Binnen de afdeling Logistics Technologies wordt er gewerkt aan methoden en technieken om logische processen binnen en tussen organisaties vloeiend te laten verlopen. Het gaat daarbij zowel om het verbeteren van werkmethode als het ontwerpen en realiseren van de concrete productiemiddelen.

Vaak wordt een project niet door één afdeling uitgevoerd, maar werken er verschillende mensen van verschillende kennisgebieden samen om een project succesvol te voltooien. Een project waarbij een aantal afdelingen betrokken zijn, is het project van KPN Homeservices. Aan dit project werken de afdelingen Markets en Services nauw samen. Het onderzoek dat in deze scriptie staat beschreven is uitgevoerd binnen de afdeling Services, specifiek het kennisgebied Service Development.

3 KPN Homeservices

Dit onderzoek is uitgevoerd binnen het project KPN Homeservices. KPN Homeservices is een project van KPN Research waarbij het automatiseren van processen in en om het huis centraal staat. Het project wordt uitgevoerd door een aantal verschillende afdelingen binnen KPN Research.

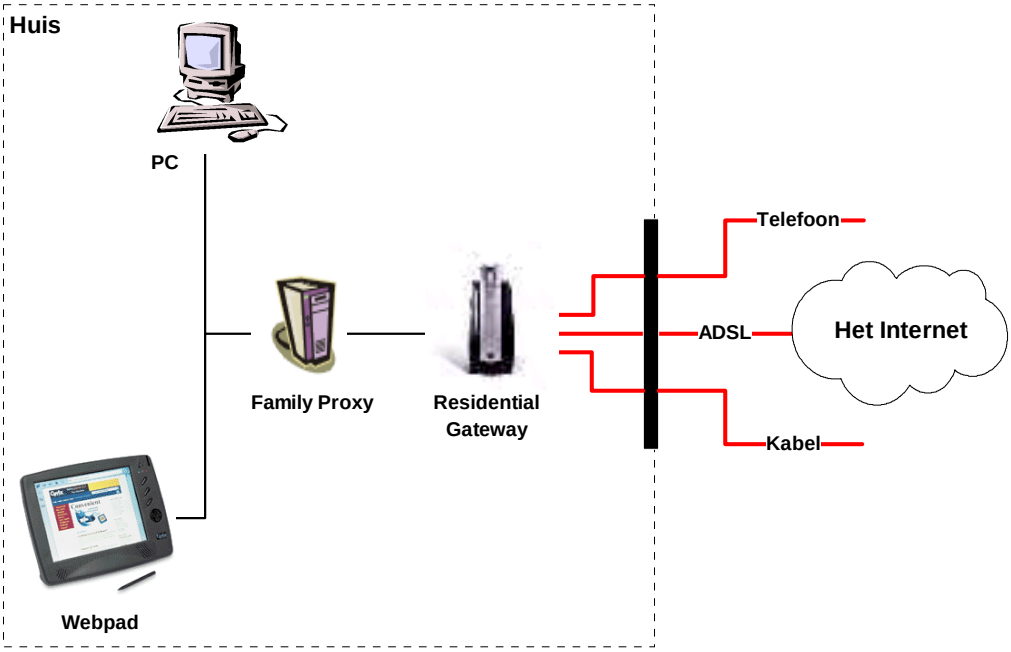
Binnen dit project staat het "in-huis" netwerk centraal. Dit netwerk zorgt ervoor dat veel apparatuur in het huis met elkaar kan communiceren. Niet alleen de computers, maar ook de televisie, videorecorder, radio, thermostaat enzovoorts, worden op dit netwerk aangesloten. Hierdoor wordt het mogelijk om deze apparaten via het netwerk aan te sturen. De televisie kan zo bijvoorbeeld met de computer bediend worden.
De bedoeling is dat de gebruiker een breedband internetaansluiting krijgt (ADSL) zodat alle aangesloten apparatuur ook via het Internet aangestuurd kan worden.

Binnen het project KPN Homeservices zijn drie deelprojecten te onderscheiden, te weten:

- Residential Gateway
- Webpad
- Family Proxy

In de onderstaande figuur is te zien hoe het in-huis netwerk eruit ziet. In dit plaatje is te zien hoe de verschillende componenten in het netwerk aangesloten zitten.

In de volgende paragrafen zullen de Residential Gateway, de Webpad en de Family Proxy kort worden toegelicht.



Figuur 2. Het In-huis netwerk

3.1 Residential Gateway

De Residential Gateway bevindt zich in het huis van de gebruiker en vormt de verbinding tussen het “in-huis” netwerk en het Internet. De Residential Gateway is naar buiten toe op verschillende netwerken aangesloten, zoals Internet, kabel en telefoon. Binnen het huis moet het in de toekomst mogelijk worden om alle diensten, zoals tv, telefoon en Internet, over één netwerk aan te bieden. De Residential Gateway is voor de gebruiker een computer waar hij verder niets zelf mee hoef te doen. Het is voor de gebruiker niet belangrijk hoe de Residential Gateway werkt, zolang hij maar werkt. Het onderhoud op deze machine moet daarom volledig op afstand kunnen plaatsvinden zonder de gebruiker daarbij lastig te vallen.

3.2 Webpad

De Webpad is een draadloze computer zonder muis of toetsenbord. De bediening van de Webpad gebeurt door middel van het "Touchscreen". De gebruiker kan door middel van zijn vingers of een pen, knoppen op het scherm aanklikken (zoals hij dit normaal gesproken met zijn muis zou doen). Ook is het mogelijk om een virtueel toetsenbord op het scherm te tonen. Op het scherm wordt dan een plaatje van een toetsenbord getoond waarbij de gebruiker de toetsen op het scherm kan indrukken (aanraken) met zijn vingers of met een speciale pen zodat bijvoorbeeld tekst kan worden ingetypt.
De Webpad is specifiek bedoeld om mee te kunnen internetten. De Webpad is niet bedoeld als gewone PC om bijvoorbeeld mee te kunnen tekstverwerken. In de toekomst zou het ook mogelijk moeten worden om televisie te kijken op de Webpad.

Voor de Webpad is een speciale browser ontwikkeld. Met deze browser kunnen de gebruikers niet alleen op het Internet surfen maar kunnen zij via deze browser ook apparaten in het huis bedienen. Op dit moment is het al mogelijk om door middel van de Webpad de televisie te bedienen.
De speciale browser moet het mogelijk maken dat mensen, met weinig tot geen ervaring met computers, kunnen internetten. In bijna alle bestaande browsers kan een gebruiker pagina’s die hij of zij vaak bezoekt als ‘Favoriet’ of ‘Bookmark’ aanmerken. Eén van de belangrijkste verschillen van de Webpad ten opzichte van andere browsers, zoals Netscape en Internet Explorer, is dat de ‘Favorieten’ van de gebruiker altijd in beeld worden getoond. Bij de browser op de Webpad zijn de ‘Favorieten’ altijd aanwezig door een reeks icoontjes onderaan in het scherm. Dit in tegenstelling tot andere browsers waar de ‘Favorieten’ vaak achter een menu verstopt zitten.

Figuur 3. Webpad

De Favorieten zijn onderverdeeld in een aantal categorieën welke door de gebruiker kunnen worden aangepast en benoemd. In Figuur 3 is een schermafdruk te zien van de browser die op de Webpad draait. De categorieën die in dit voorbeeld staan zijn: ‘Dagelijks’, ‘Handig’, ‘Vermaak’, enz. Onder elke categorie staat een aantal icoontjes die de gebruiker rechtstreeks naar de betreffende pagina leidt. Deze links hebben, evenals de categorieën, een naam die door de gebruiker zelf is opgegeven. De links onder de categorie Dagelijks zijn ‘AD’, ‘NRC’, enz. De namen van de links en de categorieën kunnen door de gebruiker zelf aangepast worden.

3.3 Family Proxy

De Family Proxy dient hoofdzakelijk voor het sneller maken (of doen lijken) van het Internet voor de gebruiker. Dit wordt bereikt door webdata (HTML- pagina’s, multimedia, enz) te cachen. In de toekomst zal de cache van de Family Proxy niet alleen gebruikt worden om deze webdata op te slaan, maar zal deze ook gebruikt worden om data van FTP sites en ‘news’ van Newsservers op te slaan.

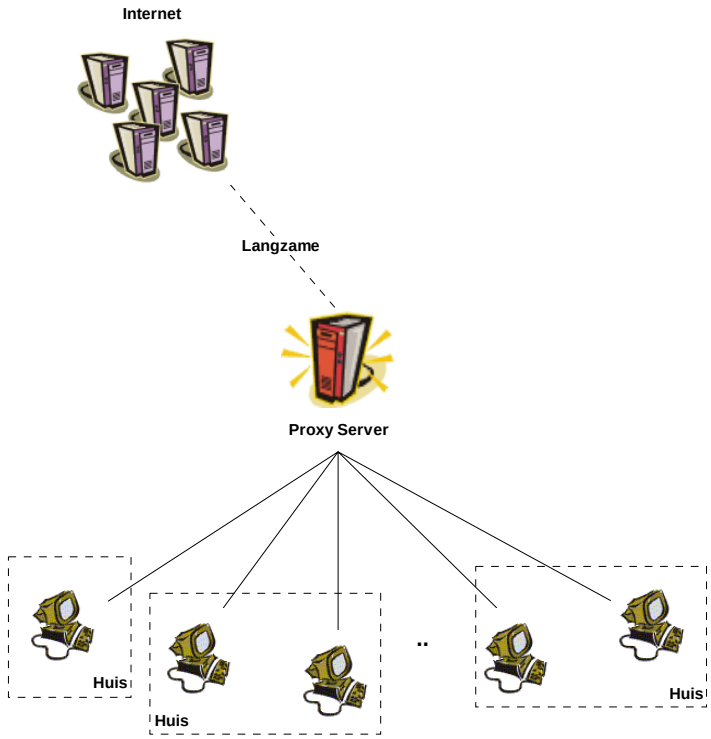
De Family Proxy komt uitgebreid aan de orde in het volgende hoofdstuk.

4 Proxy Server

In het vorige hoofdstuk is de Family Proxy genoemd als onderdeel van het KPN Homeservices project. In dit hoofdstuk wordt uitgelegd wat een Proxy Server is en hoe deze werkt. Nadat het principe van een Proxy Server is uitgelegd, wordt uitgelegd welke problemen opgelost moeten worden om een Family Proxy goed te laten functioneren.

4.1 Principe van een Proxy Server

Een Proxy Server is een computer die onderdeel uitmaakt van het Internet. Deze computer is uitgerust met een verbinding naar de gebruiker en een verbinding naar de rest van het Internet. Tevens heeft deze computer een grote hoeveelheid geheugen om pagina's te cachen. Het doel van een Proxy Server is het ontlasten van het Internet door veelbezochte pagina's in de cache van de Proxy Server op te slaan. De cache van de Proxy is een stukje geheugen waar internetpagina's tijdelijk kunnen worden opgeslagen. De verbinding van de Proxy Server naar de gebruiker thuis is vaak veel sneller dan de verbinding naar de gevraagde site op het Internet. Een Proxy Server zorgt er dus niet alleen voor dat de belasting van het Internet wordt verminderd, maar ook dat informatie van het Internet sneller bij de gebruiker thuis is. Een schematische weergave van een Proxy Server is te zien in Figuur 4.



Figuur 4. Schematische weergave van een Proxy Server

Een Proxy Server is een soort geheugenbuffer, met aan de ene kant een verbinding naar de gebruiker en aan de andere kant een verbinding naar de Internetservers. De Proxy Server is voor de gebruiker onzichtbaar. Het lijkt voor de gebruiker alsof hij direct met een Internetserver verbonden is.

De Proxy Server houdt veelbezochte pagina's in zijn cache vast. De eerste keer dat een pagina wordt opgevraagd, wordt deze niet alleen van het internet opgehaald en naar de gebruiker gestuurd, maar ook opgeslagen in de cache van de Proxy. Wanneer de pagina voor een tweede keer wordt opgevraagd, wordt deze pagina niet opnieuw van het Internet opgehaald, maar rechtstreeks uit de cache van de Proxy Server gelezen. Dit geeft uiteraard problemen wanneer de betreffende pagina tussentijds is veranderd. Hiervoor biedt een normale Proxy Server helaas geen oplossing.

De verhouding tussen bestanden die via de cache naar de gebruiker worden gestuurd en bestanden die niet in de cache staan, en dus van het Internet moeten worden gehaald, wordt de ‘hitratio’ genoemd. Wanneer er bijvoorbeeld door de gebruikers in totaal 100 webpagina’s worden opgevraagd en de Proxy Server heeft in 50 van deze gevallen de pagina rechtstreeks uit de cache naar de gebruiker kunnen sturen, is de hitratio 50%.

De hitratio is afhankelijk van een aantal zaken. Allereerst is de hitratio afhankelijk van het aantal gebruikers dat via deze Proxy Server het Internet bezoekt. Hoe meer gebruikers er zijn, hoe groter de kans is dat iemand de pagina die een bepaalde gebruiker opvraagt al eerder bezocht heeft. Wanneer die pagina al eerder bezocht is staat deze namelijk al in de cache van de Proxy Server.

Een tweede factor die invloed heeft op de hitratio, is de hoeveelheid geheugen in de Proxy Server. De cache van de Proxy Server is natuurlijk lang niet groot genoeg om alle informatie op het Internet in op te slaan. Pagina’s in de cache van de Proxy Server moeten dus worden verwijderd wanneer de cache (bijna) vol is. Hiervoor zijn verschillende criteria bedacht. Een simpel criterium zou de volgende kunnen zijn: “verwijder telkens die informatie uit de cache die het langst geleden voor het laatst is opgevraagd, totdat er voldoende vrije ruimte in de cache aanwezig is om nieuwe pagina’s in de cache op te kunnen slaan.

De hitratio is een belangrijke factor voor de snelheid waarmee pagina’s bij de gebruiker (op het beeldscherm) verschijnen. Een hoge hitratio betekent dat de meeste pagina’s die de gebruiker opvraagt al in de cache van de Proxy Server staan. In dit geval is de snelheid van de verbinding tussen de Proxy Server en de computer van de gebruiker bepalend voor de snelheid van het Internet. Of, om het anders te zeggen, dit is bepalend voor de perceptie van de gebruiker over de snelheid van het Internet. Om de snelheid van het oversturen van pagina’s (die reeds in de cache van de Proxy Server staan) nog verder te vergroten, dient de snelheid van de verbinding tussen de Proxy Server en de computer nog verder te worden verhoogd. Een mogelijkheid om dit te bereiken is om de Proxy Server bij de mensen thuis neer te zetten, vandaar de naam ‘Family’ Proxy.

De Family Proxy staat bij de mensen thuis (bijvoorbeeld in de meterkast) en is door middel van een netwerkverbinding met de overige computers in het huis aangesloten. Deze verbinding zal altijd sneller zijn dan een verbinding naar een Proxy Server via het Internet.

4.2 Family Proxy

De Family Proxy moet het internetten voor de gebruiker aangenamer maken. De informatie op Internet moet sneller toegankelijk en meer overzichtelijk worden voor de gebruiker.

Het sneller maken wordt bereikt door te zorgen dat de webpagina’s die de gebruiker (waarschijnlijk) van het Internet zal gaan opvragen, al in het geheugen van de Family Proxy staan. Wanneer de gebruiker een webpagina, die al in de cache van de Family Proxy staat opvraagt, zal deze veel sneller binnen gehaald worden dan wanneer de pagina van het Internet binnengehaald moet worden.

Het overzichtelijker maken van het Internet wordt bereikt door delen van webpagina’s weg te filteren. Dit onderdeel valt echter buiten het bestek van dit onderzoek en zal om deze reden in dit onderzoek niet verder worden behandeld.

De Family Proxy heeft een aantal taken.

- Vullen van de cache van de Family Proxy
- Up-to-date houden van de cache
- Wegfilteren van delen van webpagina's

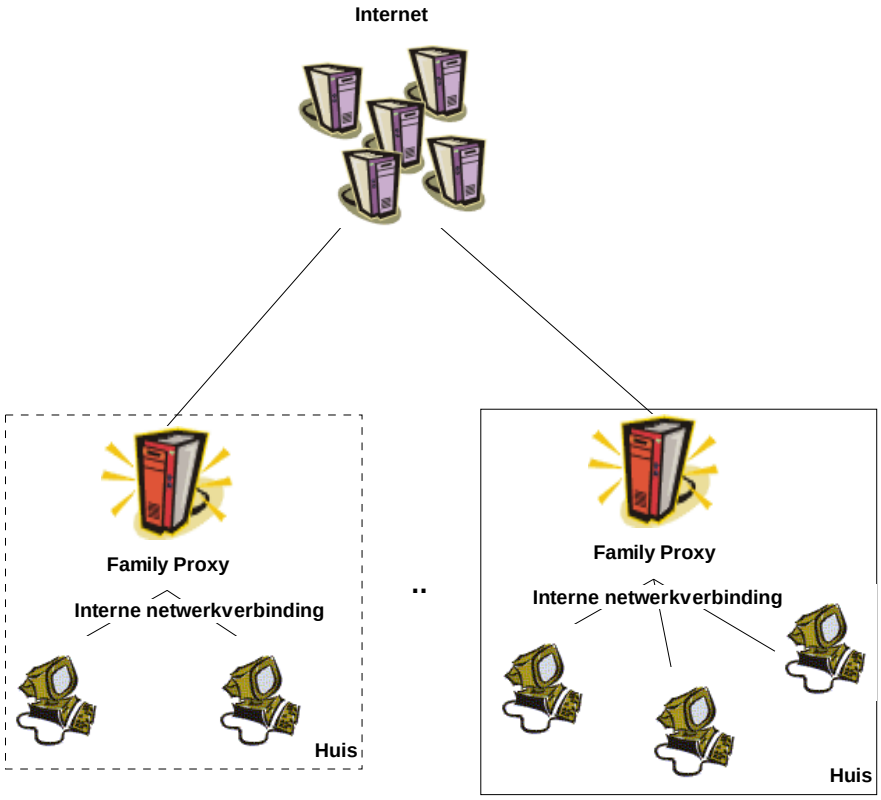
Allereerst moet de Family Proxy ervoor zorgen dat zijn cache gevuld is met pagina's. Een lege cache zal niet voor een verhoogde snelheid van het Internet zorgen. Een cache waar pagina's in staan die niet door een gebruiker worden bezocht, is eveneens nutteloos. Het vullen van de cache moet dus zodanig gebeuren dat er een goede hitratio wordt gehaald.

Ten tweede moet de Family Proxy ervoor zorgen dat de pagina's in de cache up-to-date zijn. De krant van gisteren is uiteraard niet interessant voor een gebruiker die de krant van vandaag wil lezen.

De derde taak van de Family Proxy is het (weg) filteren van delen van pagina's, bijvoorbeeld reclamebanners. Hierdoor zullen pagina's makkelijker leesbaar worden voor de gebruiker.

Dit onderzoek richt zich uitsluitend op de eerste taak van de Family Proxy, het vullen van de cache.

Om de cache van de Family Proxy te vullen is een mechanisme nodig. De reden dat het mechanisme om de cache van een normale Proxy Server te vullen goed werkt, is dat er veel gebruikers zijn die gebruik maken van deze Server (enkele honderden tot duizenden). De kans dat een webpagina in de cache van de Proxy Server staat, is afhankelijk van het aantal gebruikers en de grootte van de cache. Hoe meer gebruikers er zijn, hoe groter de kans is dat een andere gebruiker de webpagina die bezocht wordt, al eens eerder heeft opgevraagd. De webpagina staat dan dus al in de cache. Het probleem bij de Family Proxy is dat er niet veel gebruikers zijn. De gebruikers bij een Family Proxy bestaan uit de gezinsleden. Gemiddeld zal een Family Proxy ongeveer 3 tot 4 gebruikers hebben. Hierdoor is de kans dat een pagina in de cache zit, doordat iemand anders deze al eerder heeft opgevraagd, veel kleiner. In Figuur 5 staat een schematische weergave van een Family Proxy.



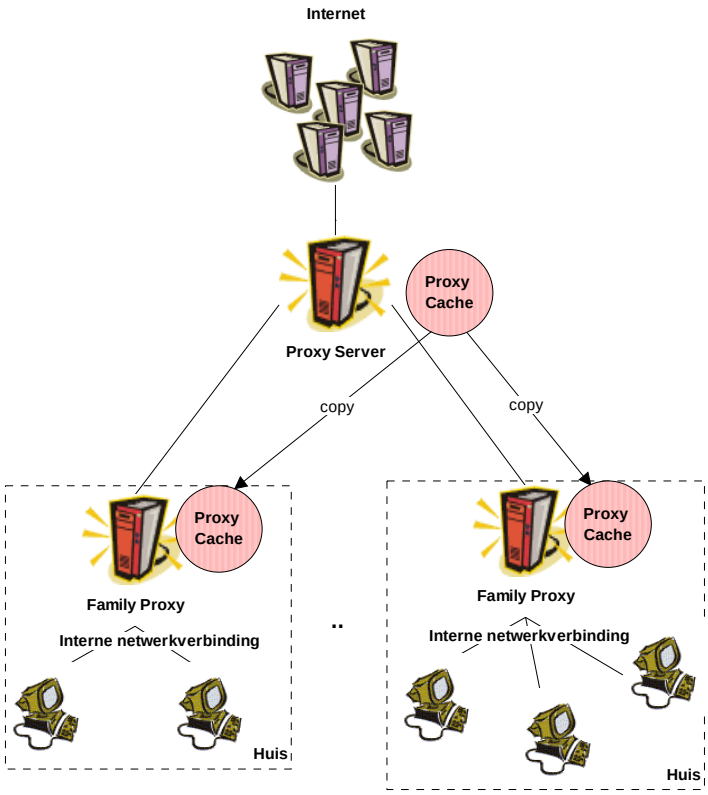
Figuur 5. Schematische weergave Family Proxy

De Family Proxy moet daarom gebruik maken van andere methoden om een hoge hitratio te bereiken. De Family Proxy moet dus zodanig 'slim' gemaakt worden dat hij van tevoren kan voorspellen welke webpagina's een gebruiker zal gaan bezoeken of zou willen bezoeken. Deze webpagina's zullen dan in het cachegeheugen worden geladen. Wanneer de gebruiker naar één van deze webpagina's 'surft', wordt deze uit de cache geladen en staat deze vervolgens binnen een fractie van een seconde op het beeldscherm van de computer van de gebruiker.

4.3 Principe Gedistribueerde Proxy

De Family Proxy moet op één of andere manier gevuld worden met webpagina's. Een simpele manier om dit te doen, zou zijn om de inhoud van de cache van de normale Proxy Server één op één naar de cache van de Family Proxy te kopiëren. Dit is geïllustreerd in Figuur 6.

Omdat de inhoud van de cache van de Family Proxy nu gelijk is aan de inhoud van de cache van de normale Proxy Server zal een gebruiker dezelfde hitratio ervaren bij de normale Proxy Server als bij de Family Proxy. De snelheid van de verbinding tussen de Family Proxy en de gebruiker is echter veel sneller dan die tussen de normale Proxy Server en de gebruiker. Er van uitgaande dat de hitratio van de normale Proxy Server goed is, zorgt deze methode voor de twee positieve effecten die men wilde bereiken met de Family Proxy (goede hitratio en een snelle verbinding tussen de gebruiker en de Proxy Server).



Figuur 6. De cache kopiëren

Behalve deze voordelen zijn er ook twee belangrijke nadelen te noemen voor deze oplossing. Ten eerste moet de cache van de Family Proxy even groot zijn als die van de normale Proxy Server. Dit houdt in dat er een behoorlijke hoeveelheid harddiskruimte en geheugen in de Family Proxy moet zitten. Een normale Proxy Server heeft vele tientallen gigabytes aan cache geheugen. Voor een gebruiker van een Family Proxy is enkele gigabytes aan webdata al ruim voldoende. De Family Proxy zou, indien men de Family Proxy net zo wil inrichten als een normale Proxy Server, te duur worden.
Een tweede nadeel is dat alle informatie die in de cache van de normale Proxy Server staat, naar alle Family Proxy's gestuurd zou moeten worden. Dit zorgt voor een zeer hoge belasting van het netwerk en dat is niet gewenst.

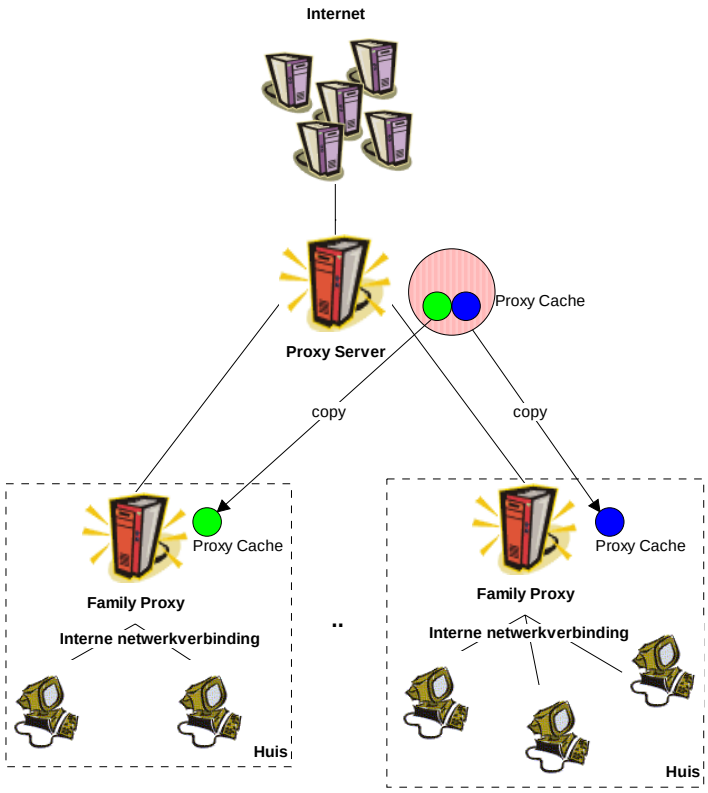
De hitratio mag dan wel goed zijn bij de voorgestelde oplossing, maar het aantal pagina's dat door de gebruikers wordt opgevraagd, is veel kleiner dan het totaal aantal pagina's dat in de cache van de Family Proxy is opgeslagen. Met andere woorden: de effectiviteit van de Family Proxy is, wanneer gekozen wordt voor deze manier om de cache van de Family Proxy te vullen, heel klein. Er moet daarom gezocht worden naar een andere oplossing.

Om de effectiviteit van de cache van de Family Proxy te verbeteren, moeten alleen die pagina's die de gebruiker **waarschijnlijk** zal bekijken, van de cache van de normale Proxy Server naar de cache van de Family Proxy gekopieerd worden. Dit is geïllustreerd in Figuur 7.

Als dus van elke pagina in de normale Proxy Server voorspeld kan worden of een gebruiker deze wel of niet zal bezoeken, worden de nadelen van de eerste genoemde oplossing (het kopiëren van de volledige cache van de normale Proxy Server) opgeheven.

De cache van de Family Proxy kan in dit geval dus een stuk kleiner zijn dan de cache van een normale Proxy Server. Tevens blijft de netwerkbelasting welke ontstaat door het vullen van de cache binnen de perken. De voordelen die genoemd werden bij de eerste oplossing, blijven bij deze oplossing echter wel behouden: de hitratio blijft even goed als

die van een normale Proxy Server en de snelheid van de verbinding tussen de gebruiker en de Family Proxy neemt toe.



Figuur 7. Gedeelte van de cache kopiëren

De informatie die in de normale Proxy Server zat, wordt nu gedistribueerd opgeslagen over de verschillende Family Proxy's. In Figuur 7 is dit geïllustreerd door middel van het kleine lichte en het kleine donkere cirkeltje 'Proxy Cache' bij de Family Proxy's. Elke Family Proxy heeft slechts de webpagina's in zijn cache staan die interessant zijn voor zijn specifieke gebruikers. Merk hierbij op dat de inhoud van de cache van verschillende Family Proxy's elkaar kan overlappen. Er kunnen uiteraard meerdere gebruikers geïnteresseerd zijn in een bepaald onderwerp en zullen daarom dus waarschijnlijk dezelfde pagina('s) in de cache van hun Family Proxy hebben staan.

5 Filtertechnieken

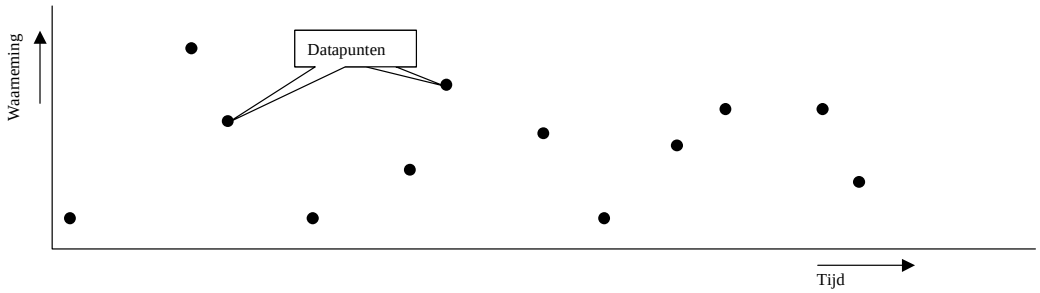
Zoals in het vorige hoofdstuk al is aangegeven, is het geen goede methode om alle webpagina's die in de cache van een normale Proxy Server staan, in de cache van de Family Proxy te zetten. Dit is bovendien onzinnig omdat deze pagina's nooit allemaal bezocht zullen worden door de gebruikers van een Family Proxy. Om de Family Proxy te vullen met webpagina's moet er een selectie worden gemaakt uit een grote hoeveelheid webpagina's die normaal in de normale Proxy Server zouden staan. In dit hoofdstuk wordt een aantal technieken besproken waarmee een dergelijke selectie kan worden gemaakt. Het gaat hier om een drietal technieken, te weten:

- Time Series Analysis
- Collaborative Filtering
- Filtering op basis van inhoud

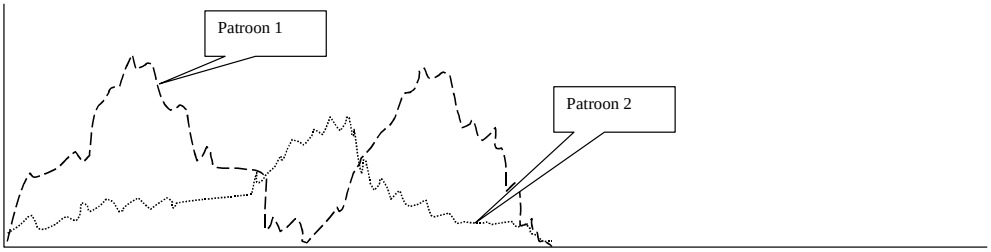
In de volgende drie paragrafen worden deze technieken besproken.

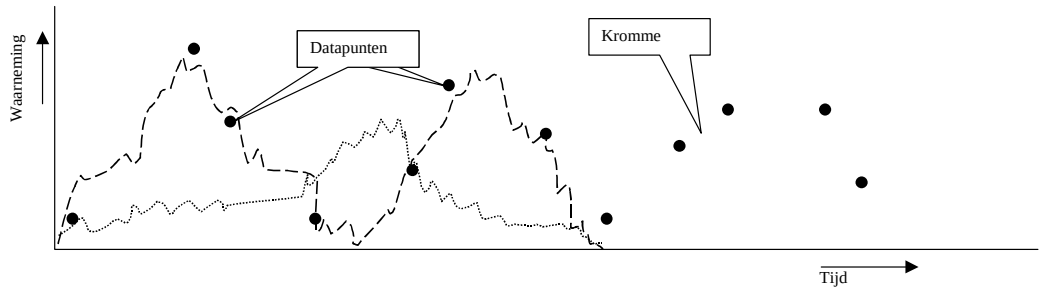
5.1 Time Series Analysis

Een tijdreeks (time serie) is een verzameling van opeenvolgende waarnemingen. Omdat de waarnemingen opvolgend in tijd zijn verzameld, kan de tijdreeks worden gezien als een functie van de tijd. Deze functie heeft op een bepaald tijdstip een bepaalde waarde, namelijk de waarde van de waarneming.

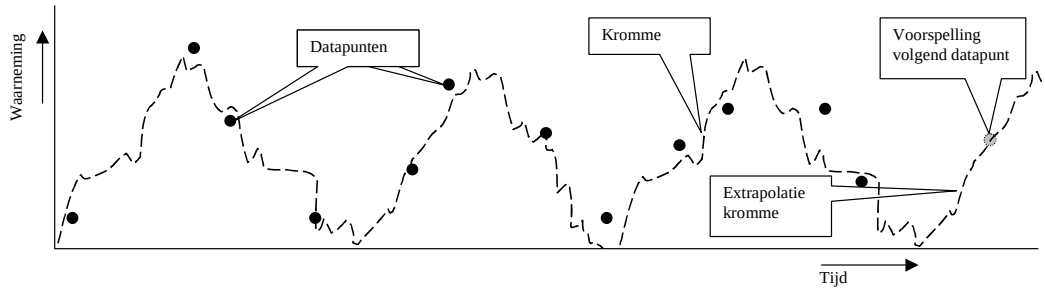


De punten in het bovenstaande plaatje zijn de waarnemingen die zijn gedaan. Deze punten kunnen vervolgens worden gebruikt om een eerder opgeslagen patroon te herkennen. In dit geval is er, om het voorbeeld eenvoudig te houden, een tweetal patronen waarmee gematcht kan worden. De patronen kunnen iedere vorm hebben. Het feit dat ze misschien sinus-vormig zijn is toeval.





De datapunten blijken het beste te matchen patroon 1. Door deze lijn te extrapoleren kan een waarneming in de toekomst worden voorspeld.



Time Series Analysis zou ook gebruikt kunnen worden om te voorspellen welke webpagina een gebruiker zal gaan bezoeken. Elke waarneming is dan een webpagina die bezocht wordt op een bepaald tijdstip door een bepaalde gebruiker. Door het surfgedrag van vele gebruikers te analyseren kan worden voorspeld waar een gebruiker heen zal surfen bij de volgende ‘klik’.

Neem het volgende voorbeeld:

Gebruiker X bezoekt achtereenvolgens de pagina’s **A** B D **C** B G E **H** F G H D

Gebruiker Y bezoekt achtereenvolgens de pagina’s **A C D** G **C H** D A B C D G

Als gebruiker Z nu achtereenvolgens de pagina’s A C D F C H bezoekt, is de kans groot dat de volgende pagina die gebruiker Z zal bezoeken pagina D is. Dit blijkt uit het feit dat het surfpatroon van gebruiker Z het meest overeenkomt met die van gebruiker Y (pattern matching). De pagina die Y na pagina H bezoekt is pagina D. Het surfgedrag van gebruiker Z lijkt veel minder op die van gebruiker X, de kans dat gebruiker Z na het bezoeken van pagina H naar pagina F zal gaan is daarom veel kleiner.

Het voordeel van Time Series Analysis is dat er vrij exacte voorspellingen gedaan kunnen worden met behulp van een eenvoudig principe.
Het nadeel van deze methode voor het voorspellen van webpagina’s is dat er enorm veel webpagina’s zijn. De kans dat er een overlap tussen het surfgedrag van twee gebruikers aanwezig is, is daarom vrij klein.

5.2 Collaborative Filtering

Een andere methode waarmee men kan voorspellen welke pagina’s voor een gebruiker interessant zijn, is Collaborative Filtering.
Collaborative Filtering werkt met een *rating* (beoordeling) van informatie. De informatie die wordt beoordeeld, bestaat in dit geval uit informatie op webpagina’s. De rating geeft aan hoe goed een gebruiker een bepaalde webpagina beoordeeld. Een webpagina die interessant is voor een gebruiker scoort beter dan een pagina die hij niet interessant vindt.
Een rating kan op twee verschillende manieren tot stand komen. Ten eerste aan de hand van een impliciete rating. Dit houdt in dat er op basis van gegevens, bijvoorbeeld hoe vaak een bepaalde webpagina wordt bezocht, een rating wordt bepaald. Een hoge rating

wordt toegekend aan een pagina die een gebruiker vaak bezoekt. Een pagina die slechts een enkele keer bezocht wordt, krijgt een lage rating.
De tweede methode om een rating vast te stellen, is expliciete rating. Hierbij wordt aan de gebruiker gevraagd om een cijfer te geven aan bijvoorbeeld een bepaalde webpagina. Hoe hoger het cijfer, hoe beter de betreffende gebruiker de webpagina beoordeeld.

Het resultaat van de rating speelt een grote rol in het doen van aanbevelingen. Hoe hoger de rating, hoe eerder een bepaalde webpagina wordt aanbevolen aan andere mensen. Collaborative Filtering houdt bij het doen van aanbevelingen, rekening met de interesses van de gebruikers. Er wordt bij het doen van aanbevelingen voor een webpagina bijvoorbeeld gekeken of de interesses van de gebruiker aansluiten bij die van de overige mensen die de betreffende pagina bezocht hebben. Dit principe zal in het onderstaande voorbeeld nader worden toegelicht.

Het principe van Collaborative Filtering werkt als volgt: stel bijvoorbeeld dat het bekend is dat veel mensen die webpagina A hebben bezocht ook pagina B hebben bezocht. Wanneer een persoon pagina A bezoekt, en zijn interesses sluiten aan bij die van de groep mensen die pagina A en B hebben bezocht, wordt hem tevens pagina B onder de aandacht gebracht met de vraag of hij wellicht ook interesse heeft in pagina B. De Family Proxy kan hier gebruik van maken door alle pagina's die worden aanbevolen in de cache te zetten. Het onderstaande voorbeeld illustreert dit.

Er zijn in dit voorbeeld 5 gebruikers en de webpagina's A, B, C, D, E en F. Van elke gebruiker is bijgehouden welke webpagina een gebruiker interessant vindt. Wanneer voor gebruiker 5 een aanbeveling moet worden gedaan, wordt er gekeken naar de overeenkomst met de andere gebruikers. Op basis hiervan wordt pagina E aanbevolen.

Gebruiker	Webpagina					
	A	B	C	D	E	F
1	X	X			X	
2	X	X	X	X		X
3			X	X		
4	X	X			X	
5	X	X				

Figuur 8. Webpagina voorbeeld Collaborative Filtering

De formules om Collaborative Filtering uit te rekenen luiden als volgt:

Om de rating te voorspellen voor gebruiker i:

$$\bar{v}_i = \frac{1}{|I_i|}\sum_{j \in I_i} v_{i,j} \qquad (1)$$

waarbij I_i de set van ratings is die de gebruiker heeft opgegeven en $v_{i,j}$ de rating is van gebruiker i voor item j.

De voorspelde rating ($p_{a,j}$) van de actieve gebruiker (a) voor item j kan berekent worden dor het gewogen gewicht te nemen van de ratings van alle andere gebruikers.

$$p_{a,j} = \bar{v}_a + \kappa \sum_{i=1}^{n} w(a,i)(v_{i,j} - \bar{v}_i) \qquad (2)$$

Waarbij n het aantal gebruikers in de database is. De gewichten w(a,i) stelt een afstandsmaat of correlatie voor tussen de gebruiker i en de actieve gebruiker. Voor meer informatie over Collaborative Filtering zie [9].

Aan de methode van Collaborative Filtering kleven echter drie grote nadelen waardoor het moeilijk is om deze techniek te gebruiken om informatie voor de Family Proxy te filteren.
Een eerste nadeel is dat Collaborative Filtering positieve en negatieve ratings nodig heeft om goed te functioneren. Men wil deze ratings niet expliciet aan de gebruiker vragen. De Family Proxy moet het internetten tenslotte plezieriger maken, niet belasten. Wanneer bij elke pagina gevraagd wordt om deze een cijfer te geven, zal de gebruiker hier niet blij mee zijn.
Om door middel van impliciete rating een positieve rating af te leiden is mogelijk. Een pagina die door een gebruiker vaak bezocht wordt, scoort blijkbaar hoog voor deze gebruiker. Om nu te stellen dat de pagina's die een gebruiker niet bezoekt een negatieve rating moeten krijgen is problematisch. Dit zou betekenen dat de rest van het Internet negatief beoordeeld moet worden. Dit is extreem. Het maken van impliciete negatieve ratings is dus een probleem. Met enkel en alleen positieve ratings werkt Collaborative Filtering niet goed. Collaborative Filtering moet daarnaast ook weten wat een gebruiker niet goed vindt. Wanneer alleen bekend is wat een gebruiker wel goed vindt, kan Collaborative Filtering geen goede aanbevelingen doen.
Ten tweede kan Collaborative Filtering voor een verrassingseffect zorgen. Een mooi voorbeeld hiervan is een proef die enige tijd geleden is uitgevoerd bij een supermarkt. Hier wilde men meten welke producten vaak samen werden gekocht om te bepalen welke producten bij elkaar zouden moeten staan om de verkoop van deze producten te bevorderen. Het bleek dat luiers en six-packs bier heel vaak samen werden gekocht. Het bleek dat mannen die luiers halen hun imago willen opkrikken door er een six-pack bier bij te kopen. Wanneer iemand dus een six-pack bier haalt, kan hem gevraagd worden of hij er ook een doos luiers bij wil hebben. Dit is het verrassingseffect wat ook kan optreden bij het doen van aanbevelingen voor webpagina's.
Webpagina's die door veel mensen vaak samen worden bezocht, gaan niet noodzakelijkerwijs over hetzelfde onderwerp. Aanbevelingen die gedaan worden op basis van Collaborative Filtering, hoeven daarom niet aan te sluiten bij de directe interesses van de gebruiker. Dit hoeft niet noodzakelijkerwijs een nadeel te zijn, maar kan wel voor verrassingen zorgen.
Tenslotte werkt Collaborative Filtering alleen wanneer er veel mensen zijn die gebruik maken van het systeem. Als er te weinig mensen gebruik maken van het systeem, kunnen er ook geen patronen worden gevonden. Het Family Proxy project wordt in eerste instantie getest bij tien gezinnen. Dit zijn dus veel te weinig gebruikers om goede voorspellingen te kunnen doen op basis van Collaborative Filtering.

Een betere methode om aanbevelingen te doen, zou zijn om te kijken naar de inhoud van de webpagina's zelf. De volgende paragraaf behandelt deze methode.

5.3 Filteren op basis van inhoud

Wanneer iemand een webpagina opvraagt, zal hij beginnen met het lezen hiervan en vervolgens bepalen of hij de pagina interessant vindt (en dus door zal lezen) of dat hij deze niet interessant vindt (en niet verder leest). De gebruiker bepaalt dus aan de hand van de inhoud of hij deze pagina interessant vindt.

Webpagina's beoordelen op basis van inhoud kan gebruikt worden om aanbevelingen voor webpagina's aan een gebruiker te doen. Handmatig bepalen waar een webpagina over gaat, is heel goed mogelijk maar is ondoenlijk als het gaat om grote hoeveelheden webpagina's.
Dit proces kan ook automatisch worden uitgevoerd door een computerprogramma de inhoud van een webpagina te laten lezen. Wanneer het computerprogramma weet wat de interesses van de gebruiker zijn, dan kan het programma bepalen of een bepaalde pagina wel of niet interessant is voor de betreffende gebruiker. Dit is filteren op basis van inhoud. Wanneer dus bekend is wat de interesses zijn van een bepaald persoon en wanneer bekend is waar een webpagina over gaat, kan op basis van deze informatie voorspeld worden of een gebruiker deze pagina wellicht interessant zal vinden. Voor dit proces is het dus belangrijk om een goede beschrijving te hebben van de inhoud van een

pagina. Hoe deze beschrijving gemaakt wordt, zal in het volgende hoofdstuk aan de orde komen.

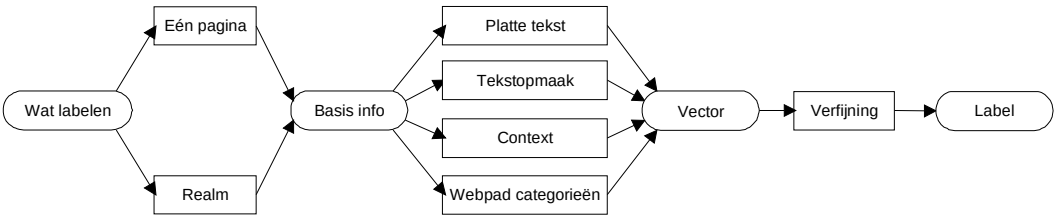
5.4 Conclusies

Voor het onderzoek is gekozen om één techniek te gaan gebruiken voor het filteren van informatie. Time Series Analysis is een techniek waarbij heel veel data nodig is om deze goed te kunnen laten werken. Deze grote hoeveelheid data is in dit onderzoek niet aanwezig. Daarom is ervoor gekozen om deze techniek niet in het onderzoek te gebruiken.
Collaborative Filtering is al eens eerder in een project van KPN Research toegepast. Daar is gebleken dat het algoritme alleen goed werkt wanneer zowel positieve als negatieve ratings worden gebruikt. Omdat het heel ingewikkeld is om impliciet goede ratings vast te stellen, en het uitgesloten is om deze ratings aan de gebruikers te vragen, is er niet gekozen voor Collaborative Filtering.
Er is gekozen voor de derde techniek, filteren op basis van inhoud. Deze techniek biedt voor de Family Proxy de beste manier om webpagina's goed te kunnen filteren. Ten eerste omdat de techniek kan worden toegepast op een kleine groep testpersonen. Ten tweede omdat men de gebruiker bij deze techniek niet lastig hoeft te vallen met vragen over ratings.

6 Labelen van webpagina's

Zoals in het vorige hoofdstuk al is beschreven, zal dit onderzoek zich richten op het filteren door middel van de inhoud van een webpagina. Het labelen van webpagina's houdt in dat er van een webpagina automatisch een beschrijving wordt gemaakt. Deze beschrijving is het label waarmee de pagina gekarakteriseerd wordt.

Er zijn al methoden bekend [20] om teksten automatisch te labelen of om automatisch samenvattingen te maken van artikelen. Deze technieken kunnen niet zonder meer toegepast worden op HTML pagina's.
Ten eerste bevatten webpagina's vaak veel minder tekst dan artikelen. Hierdoor kan het lastiger zijn om te bepalen waar een webpagina over gaat. Ook wordt er meer gebruik gemaakt van grafische objecten, zoals plaatjes en knoppen. Deze plaatjes bevatten vaak veel informatie, maar automatisch bepalen wat er in een plaatje staat is lastig.
Tenslotte zit ook de structuur van een webdocument anders in elkaar dan de structuur van een artikel. Een artikel is meestal één document. Een webdocument kan over meerdere HTML pagina's verspreid staan, welke door middel van hyperlinks met elkaar verbonden zijn. Om deze redenen moeten de technieken om teksten te labelen uitgebreid worden voor het labelen van webpagina's.



Figuur 9. Globaal labelproces

In Figuur 9 is te zien hoe het labelen globaal in zijn werk gaat. Eerst moet er gekozen worden of een webpagina apart gelabeld wordt, of dat een aantal webpagina's samen (een Realm) gelabeld worden. Dit wordt besproken in paragraaf 6.1. Als deze keuze gemaakt is, is er een basis voor het labelen. Deze basis is de tekst van één webpagina of van een aantal webpagina's bij elkaar. Van deze tekst wordt vervolgens een vector gemaakt in de tweede stap.

Voor het maken van deze beschrijvingsvector zijn een aantal oplossingen mogelijk. Er kan hier gekeken worden naar de tekst op de webpagina, de tekstopmaak op een webpagina, de context of de Webpadcategorieën. Deze methoden worden besproken in paragraaf 6.2.

Tenslotte kunnen nog een aantal verfijningtechnieken toegepast worden om van een vector een label te maken. In totaal worden er hiervoor een vijftal technieken besproken in paragraaf 6.3.

6.1 Keuze voor labelen

Voor de keuze wat er gelabeld moet worden, zijn er een tweetal mogelijkheden. Ten eerste kan worden uitgegaan van het labelen van iedere webpagina apart of van het labelen van een verzameling webpagina's in één keer.

Zoals al eerder genoemd is staan webdocumenten vaak verspreid over een aantal HTML-pagina's die door links met elkaar verbonden zijn. Een dergelijke set van webpagina's wordt een Realm genoemd. Om een webdocument goed te kunnen labelen zou een webdocument, dat uit verschillende webpagina's bestaat, dus niet als afzonderlijke pagina's moeten worden beschouwd, maar als één document. De moeilijkheid zit hierbij in het bepalen wat één document is.

Om te bepalen wat nog tot een bepaalde Realm hoort, kan er gekeken worden naar de hyperlinks op de betreffende pagina. Alle links op een webpagina worden dan beschouwd als behorend tot het eerste document. Helaas is deze linkstructuur meestal niet beperkt tot één laag. Op de pagina's waarnaar wordt verwezen vanaf de eerste pagina kunnen ook weer hyperlinks staan. Horen de pagina's waar deze links naar wijzen dan ook tot het eerste document of niet? Om te bepalen welke webpagina's nog wel, en welke niet meer tot een Realm behoren, kunnen er een tweetal restricties worden opgelegd.

Een eerste restrictie die kan worden opgelegd, is de eis dat een link niet naar een document op een andere server mag wijzen. Over het algemeen zullen de pagina's van één webdocument niet verspreid staan over verschillende servers (alhoewel dit natuurlijk wel zou kunnen en ongetwijfeld ook voorkomt).

Een tweede restrictie is dat webpagina's in dezelfde directory op de server moeten staan om te behoren tot dezelfde Realm. Elke server heeft de mogelijkheid om directories aan te maken. Veelal staan de pagina's van een webdocument in dezelfde directory. Ook hier geldt weer dat dit niet 'verplicht' is. Er is geen regel die vastlegt dat pagina's die bij elkaar horen in dezelfde directory moeten staan.

Een derde restrictie die kan worden opgelegd is dat de webpagina's binnen een Realm in hetzelfde cluster moeten vallen. Omdat de webpagina's binnen een Realm over hetzelfde onderwerp moet gaan, kan met behulp van clustering op basis van inhoud bepaald worden of een webpagina wel of niet bij een Realm hoort.

Bij het labelen van een Realm worden alle pagina's in deze Realm 'aan elkaar geplakt' en gezien als één pagina. Het voordeel van het labelen van een Realm ten opzichte van het labelen van een afzonderlijke webpagina, is dat een label wordt gemaakt op basis van meer tekst. De kans dat dit label een goede beschrijving van de webpagina's is, is daardoor ook groter.

Een nadeel van labelen met behulp van Realms is dat de labels meer termen bevatten dan een label van een afzonderlijke webpagina. Ook bestaat het gevaar nog steeds dat een Realm webpagina's bevat die er niet in moeten zitten en dat er webpagina's niet in zitten die er wel in hadden moeten zitten.

6.2 Basis voor labeling

Om een label te maken, is er tekst nodig op basis waarvan dit label gemaakt wordt. Hiervoor zijn een aantal keuzes mogelijk. Ten eerste kan er gekeken worden naar de tekst op de pagina zelf. Er kan dan wel of niet rekening gehouden worden met de opmaak van de pagina. Deze twee methoden worden in paragraaf 6.2.1 en 6.2.2 behandeld. Een andere methode is op basis van de context van hyperlinks. Dit wordt uitgewerkt in paragraaf 6.2.3. Tenslotte wordt in paragraaf 6.2.4 nog een methode besproken om informatie van gebruikers te gebruiken om webpagina's te labelen.

6.2.1 Labelen op basis van tekst inhoud

De meest eenvoudige manier om labels te maken is door alleen te kijken naar de tekst op een pagina. Webdocumenten zijn geschreven in een speciale opmaaktaal, namelijk HTML. Deze taal bevat behalve de inhoudelijke tekst ook opmaakcodes. Deze opmaakcodes moeten niet worden gebruikt om een pagina mee te labelen. De

opmaakcodes zeggen namelijk alleen iets over hoe de tekst er op het beeldscherm uit gaat zien, de opmaakcodes zeggen niets over de inhoud van de tekst zelf.

Als alle opmaakcodes uit een HTML pagina worden verwijderd, blijft slechts de ruwe tekst over. Van de losse woorden in deze tekst kan dan een vector gemaakt worden die bestaat uit de woorden die in de tekst voorkomen. Daarbij geeft een waarde aan hoe vaak het betreffende woord in de tekst voorkomt. De vectoren die zo ontstaan, bevatten over het algemeen veel verschillende woorden. De moeilijkheid hierbij is dat niet precies bepaald kan worden welke woorden wel, en welke woorden niet iets zeggen over de inhoud van de webpagina.

Het nadeel van labelen op basis van tekst inhoud is dat niet exact vastgesteld kan worden waar een webpagina precies over gaat. Het voordeel is dat de techniek vrij simpel is. Een ander voordeel is dat de ‘vindkans’ groot is. Als er gezocht wordt op een term die het document classificeert, is de kans, dat de webpagina ook daadwerkelijk gevonden wordt, erg groot. Het label van de webpagina bevat immers veel verschillende woorden.

6.2.2 Labelen op basis van HTML opmaak

Woorden die opvallen in een tekst geven vaak aan waar het stuk tekst over gaat. In HTML is het mogelijk om met opmaakcodes teksten meer te laten opvallen. Bijvoorbeeld kopteksten vallen, visueel gezien, op doordat deze groter en dikker worden afgebeeld. Vaak zeggen deze teksten veel over de inhoud van de tekst waarin ze voorkomen. Woorden die op deze manier opvallen zouden een groter gewicht kunnen krijgen dan de overige woorden in de tekst. De waarde van een woord in een vector hangt dan niet alleen af van hoe vaak een woord voorkomt in een tekst, maar hangt dan tevens af van de manier waarop het woord is opgemaakt (vet, cursief, koptekst, etc).

Een ‘probleem’ met deze strategie is dat het niet altijd goed werkt. Pagina’s met bijvoorbeeld als titel ‘page 1’ of ‘no title’, welke standaard door html-editors worden ingevuld wanneer de maker zelf geen titel opgeeft, zeggen niet erg veel over de inhoud. Omdat deze standaard titels bekend zijn, kunnen deze temen daarentegen toch eenvoudig worden weggefilterd.

Een andere opmaakcode die gebruikt kan worden voor het labelen van een webdocument is ‘Meta-tags’. Meta-tags zijn tags die door de maker zelf gedefinieerd kunnen worden. De tag ‘keyword’ wordt bijvoorbeeld vaak gebruikt door de auteur om aan te geven waar de betreffende pagina over gaat. Verschillende searchengines maken gebruik van deze ‘keyword Meta-tag’. De Meta-tags zijn overigens niet direct zichtbaar voor een bezoeker van de pagina.
In Figuur 10 is een voorbeeld gegeven van een Meta-tag. Dit voorbeeld laat de keywords zien van de hoofdpagina van AltaVista.

```
<META
content="search, searches, search engine, directory, directories, category,
categories, help, multi media, maps, business finder, yellow pages, white pages,
people search, find people, searching, searchers, advanced search, search help,
search guide, search tips, search tools"
name=keywords>
```

Figuur 10. Keyword Meta-tag van AltaVista

Op het eerste gezicht lijkt dit ideaal om te bepalen waar een webpagina over gaat. Veel zoekmachines maken hier dan ook gebruik van. Er zijn zelfs zoekmachines die uitsluitend van deze Meta-tags gebruik maken.

Er kleeft echter een groot nadeel aan het gebruik van deze Meta-tags voor het labelen van een pagina. De termen worden door de maker van de pagina opgegeven en omdat er geen controle is, kunnen zij hier van alles in zetten. Veel makers van webpagina's willen zo veel mogelijk bezoekers op hun pagina krijgen en kunnen deze Meta-tags misbruiken om hun pagina zo hoog mogelijk in de ranglijst van zoekmachines te krijgen. Dit wordt dan ook vaak gedaan.

Het voordeel van deze techniek is dat woorden die visueel opvallen zwaarder wegen in de beschrijving van de webpagina. Er moet wel goed worden gekeken welke tags gebruikt worden. Zoals in deze paragraaf al gezegd is, kunnen sommige tags handig misbruikt worden. Ook zal dit gewicht gerelativeerd moeten worden, bij een webpagina waar alles 'bold' is, heeft het niet veel zin meer om de woorden in 'bold' een groter gewicht te geven.

6.2.3 Labelen op basis van context [1]

Eén van de dingen waarmee webdocumenten zich onderscheiden van platte tekstdocumenten zijn de hyperlinks. Hyperlinks zijn doorverwijzingen naar andere pagina's waar een gebruiker op kan klikken. Deze links zijn vaak verwijzingen naar vervolg pagina's of webpagina's die gerelateerd zijn aan de huidige pagina. In de beschrijving van een hyperlink staat vaak, maar niet altijd, een korte omschrijving van de pagina waar naar wordt verwezen. Deze omschrijving geeft vaak heel kort en krachtig weer waar een pagina over gaat.

```
<a href="http://www.altavista.com/">
<img src = "altavista.gif"
alt = "My Favorite Searchengine">
</a>
```

```
<a
href="http://www.altavista.com/">
My Favorite Searchengine
</a>
```

My Favorite Searchengine

Figuur 11. Voorbeelden van hyperlinks

In Figuur 11 staan twee voorbeelden van hyperlinks. In het eerste voorbeeld is gebruik gemaakt van een plaatje waarop geklikt kan worden. In het tweede voorbeeld is gebruik gemaakt van tekst waarop geklikt kan worden.

Een hyperlink begint altijd met de tag <a en eindigt met de tag . De website waar naar wordt verwezen, is in beide voorbeelden www.altavista.com. De link wordt gerepresenteerd als een plaatje (altavista.gif) in het eerste voorbeeld en een tekst (My Favorite Searchengine) in het tweede voorbeeld. Binnen de tag `img` (image) staat nog de variabele `alt` (alternative) die de waarde `"My Favorite searchengine"` heeft. Deze alternatieve tekst wordt getoond wanneer het plaatje niet geladen wordt. Internet Explorer toont deze tekst ook wanneer met de muis over het plaatje wordt bewogen.

Voor het labelen met behulp van context kan zowel de alternatieve tekst uit het eerste voorbeeld als de tekst van de link uit het tweede voorbeeld, worden gebruikt.

Een effect van labelen met context is dat er over het algemeen weinig termen voor een pagina worden gevonden. Het aantal termen is afhankelijk van de beschrijving in een hyperlink en het aantal links naar de betreffende pagina. Dit kan zowel voor- als nadelig zijn. Voordelig omdat de pagina goed kan worden beschreven met een beperkt aantal termen. Het is ook nadelig dat de webpagina niet wordt gevonden wanneer er gezocht wordt met een term die hetzelfde betekent of verwant is aan het label van de pagina. In dit laatste geval wordt de term namelijk niet letterlijk teruggevonden in het label.

Een ander probleem wat bij deze techniek speelt, is dat het niet eenvoudig is om webpagina's te vinden die naar een bepaalde webpagina verwijzen. Het is dus moeilijk om een gegeven webpagina met behulp van deze techniek te labelen.

6.2.4 Labelen op basis van de Webpadcategorieën

Binnen het project KPN Homeservices wordt gebruik gemaakt van een Webpad. Zoals al eerder in paragraaf 3.2 is beschreven, heeft de Webpad een navigatiebalk waar de gebruiker links naar webpagina's in heeft staan. De gebruiker kan de links onderverdelen in categorieën welke hij zelf een naam kan geven. Ook moet de gebruiker een link een naam geven. De naam van de categorie en de naam van de link zijn dus een indicatie voor de inhoud van de webpagina's die door de gebruiker in deze categorie is geplaatst. Deze naam van de categorie en naam van de link kan dan gebruikt worden als label voor de betreffende pagina.

Eigenlijk is deze vorm van labelen hetzelfde als labelen op basis van context zoals besproken in de vorige paragraaf. De voor- en nadelen van labelen op basis van context zijn hier ook van toepassing. Met uitzondering van het laatst genoemde probleem, omdat hier wel eenvoudig kan worden nagegaan welke links van een gebruiker naar een bepaalde webpagina verwijzen. Deze informatie wordt namelijk door de gebruiker beschikbaar gesteld.

6.3 Technieken voor verfijning

Uit de methoden die in de vorige paragraaf besproken zijn, kan een beschrijving in de vorm van een termfrequentievector worden afgeleid. Een termfrequentievector is een reeks van termen (woorden) met een bijbehorende gewichtsfactor. Het gewicht van een term wordt bepaald door de frequentie waarmee deze term op de bepaalde pagina voorkomt. Hoe vaker een term voorkomt, hoe hoger het gewicht.

Het maken van een goede termfrequentievector is essentieel voor het matchen van interessante pagina's bij de interesses van een gebruiker. Van de termfrequentievector wordt vervolgens een label gemaakt welke aangeeft waar de pagina over gaat. Met dit label kunnen pagina's onderling en met gebruikersprofielen worden vergeleken. Wanneer een label niet juist is, kan het gebeuren dat en pagina wel matcht met de interesses van een gebruiker, terwijl dit niet zo zou moeten zijn. Het is dus belangrijk dat de termfrequentievector zorgvuldig wordt gemaakt. Wanneer hier fouten worden gemaakt, zal dat resulteren in een verkeerde match en daarmee een verkeerde voorspelling. Om de kwaliteit van voorspellingen te kunnen verbeteren, moet het aantal verschillende woorden worden verkleind. Hoe groter het aantal verschillende woorden in alle labels, hoe kleiner de kans op een mismatch. Het matchen, zoals hier besproken is, komt in hoofdstuk 7 uitgebreid aan de orde.

In de volgende paragrafen zullen een vijftal technieken besproken worden die de kwaliteit van de vectoren verbeteren.

- Stopwoorden
- Stemming
- Synoniemen / alias
- TFIDF
- Significante woorden

6.3.1 Stopwoorden

Stopwoorden zijn niet informatiedragende woorden. Ze dragen niets bij aan de inhoud van een pagina en moeten er om die reden uit worden gefilterd. Woorden zoals "de",

“het” en “een” zijn voorbeelden van stopwoorden. Deze woorden zullen over het algemeen niets zeggen over de inhoud van een pagina. Een manier om de stopwoorden te verwijderen is door gebruik te maken van een stopwoordenlijst. De woorden die in deze lijst voorkomen, worden uit het te labelen document gehaald.

Over het algemeen geldt echter, dat wanneer deze stopwoorden weggefilterd worden, de overeenkomst tussen documenten, of tussen een gebruikersprofiel en een document, wordt vergroot. Neem als voorbeeld een document waarin staat “de fiets” en een tweede document waarin staat “een fiets”. Intuïtief zou men verwachten dat deze twee documenten voor 100% met elkaar overeenkomen. In de praktijk zal dit niet het geval zijn omdat ‘fiets’ in het eerste document wel overeenkomt met ‘fiets’ in het tweede document, maar de woorden ‘de’ en ‘een’ (gezien vanuit het perspectief van de computer) niet met elkaar overeen komen. Wanneer de woorden ‘de’ en ‘een’ weg worden gefilterd, wordt wel het beoogde resultaat behaald.

Het voordeel van het verwijderen van stopwoorden uit de termfrequentievector is dat het totaal aan verschillende termen afneemt en het zal in het algemeen de overeenkomst tussen documenten, of tussen een document en een gebruikersprofiel, verbeteren.

Een nadeel van deze methode is dat er soms te veel wordt weggehaald. Vaak worden in een stopwoordenlijst ook de woorden meegenomen die bestaan uit één letter, de woorden ‘a’ tot en met ‘z’. Een document dat over de programmeertaal ‘C’ gaat, verliest op deze manier zijn belangrijkste kernwoord!

6.3.2 Stemming

Stemming is een methode waarbij woorden tot hun stam worden teruggebracht. Bijvoorbeeld ‘lopen’ wordt teruggebracht tot ‘loop’. Ook meervouden verdwijnen door toepassing van stemming, ‘voeten’ wordt vervangen door ‘voet’. Op deze manier zal een document dat gaat over ‘fietsen’ en een document dat gaat over ‘gefietst’, beide worden weergegeven met het woord fiets. Merk op dat wanneer dit **niet** gedaan wordt en het eerste document uitsluitend het woord ‘fietsen’ bevat en het tweede uitsluitend ‘gefietst’, de documenten voor een computer niet overeenkomen. Wanneer beide termen echter worden vervangen door fiets, komen de documenten voor de computer wel overeen.

Voor het stemmen kan gebruik worden gemaakt van woordenlijsten. In deze woordenlijsten staat dan welke woorden vervoegingen van elkaar zijn. Een voorbeeld van een dergelijke woordenlijst is gemaakt door CELEX.

```
…
17194\computer\8865\1113\0\62\1.7924\978\59\1.7709\135\105\2.0212
17195\computerize\8866\1\1\0\0\0\0\1\1\0
17196\computerized\8866\10\28\1\0\9\1\0\1\1\0
17197\computerizes\8866\0\0\0\0\0\0\0\0\0
17198\computerizing\8866\3\0\0\0\3\0\0\0\0
17199\computers\8865\570\0\32\1.5051\478\29\1.4624\92\71\1.8513
17200\computes\8864\4\0\0\0\4\0\0\0\0
17201\computing\8864\79\0\4\.6021\55\3\.4771\24\19\1.2788
…
```

Figuur 12. CELEX database

In de CELEX database staat van elk woord bij welke categorie ze horen. Het eerste nummer in de figuur is het nummer van een woord. Het woord ‘computer’ is dus het 17.194ste woord in de database. Het eerste getal na het woord is het nummer van de categorie. Het woord ‘computer’ hoort dus thuis in categorie 8865 evenals het woord ‘computers’. Door woorden die behoren tot dezelfde categorie te vervangen door hetzelfde woord, wordt het totaal aantal woorden verkleind.

Een andere methode die kan worden toegepast voor het stemmen van woorden zijn technieken die gebruik maken van regels. Een voorbeeld van een regel is het verwijderen van suffixen. Een suffix is dat deel van een woord waar het woord op eindigt. In het Engels zijn dit bijvoorbeeld –ual, -ly of –ing. Deze suffixen worden weggehaald om het aantal verschillende woorden te reduceren. Het woord 'searching' bijvoorbeeld, zal worden vervangen door 'search', zodat documenten waarin 'search' voorkomt, overeenkomen met documenten waar het woord 'searching' in voorkomt.

Een probleem wat zich hier kan voordoen, is overstemming ofwel het te ver doorvoeren van stemming. Hierdoor zullen woorden die niet dezelfde betekenis hebben ook worden teruggebracht tot hetzelfde woord. Een voorbeeld hiervan is 'generic' en 'general'. Wanneer bij de woorden de suffix –ic en –al worden verwijderd, blijft er van beide woorden het woord 'gener' over. Hiermee zou dus foutief geconcludeerd kunnen worden dat de twee woorden dezelfde betekenis hebben.

6.3.3 Synoniemen

Een andere methode om het aantal verschillende woorden in een termfrequentievector te reduceren, is het opsporen van synoniemen. Een document dat uitsluitend gaat over rijwielen zal niet overeenkomen met de een documenten waar alleen het woord fiets in voorkomt, terwijl rijwielen en fietsen toch redelijk hetzelfde zijn. Het opsporen van synoniemen is over het algemeen veel lastiger omdat de context vaak bepaalt wat de betekenis van een woord is. Met een 'automaat' kan bijvoorbeeld een auto met automatische versnellingsbak worden bedoeld, of een automaat waar men blikjes fris uit kan halen.

6.3.4 TFIDF

De termfrequentie vectoren die nu gecreëerd zijn, hebben nog één nadeel. De gewichten van de termen zijn gebaseerd op het voorkomen van de term in één document. Een term die veel in een bepaald document voorkomt zou intuïtief aangeven dat dit een belangrijke term is voor dit document. Maar wanneer blijkt dat deze term in de meeste documenten veel voorkomt, is deze term plotseling niet meer geschikt om het betreffende document te beschrijven. Er moet dus een mechanisme ingebouwd worden wat dit probleem ondervangt. TFIDF (Term Frequency times Invers Document Frequency) is een techniek waarmee de gewichten van de termen aangepast kunnen worden aan het globaal voorkomen van de woorden in alle documenten. Een woord dat in veel documenten voorkomt, zegt minder over de inhoud van een document waarin dit woord voorkomt dan een woord dat in weinig documenten voorkomt. Een woord in een document dat maar in heel weinig andere documenten voorkomt is blijkbaar heel specifiek voor het betreffende document.
Bijvoorbeeld documenten die gaan over AI conferenties. Wanneer wordt gekeken naar alle beschikbare documenten over allerlei onderwerpen, zullen de woorden 'AI' en 'conferenties' weinig voorkomen. In dit geval zijn deze termen dus uitermate geschikt om de betreffende documenten mee te labelen. Wanneer alleen gekeken wordt naar de verzameling van documenten die gaan over conferenties met in het bijzonder AI conferenties, zijn de woorden 'AI' en 'conferenties' veel minder geschikt om te labelen. Immers, alle documenten in de verzameling gaan over AI conferenties. In dat geval moet gekeken worden naar andere termen die specifieker zijn voor de betreffende documenten.
De methode TFIDF is bedoeld om deze specifieke termen op te sporen.

De Inverse Document Frequency (IDF) wordt als volgt berekend:

$$IDF(w_i) = \log\left(\frac{|D|}{DF(w_i)}\right)$$

Waarbij |D| het totale aantal documenten is en DF(w_i) het aantal documenten is waarin de term w_i minimaal één keer voorkomt.

Het gewicht van een term w_i in document d kan nu als volgt worden berekend:

$$d^{(w_i)} = TF(w_i, d) \cdot IDF(w_i)$$

Waarbij TF(w_i,d) de frequentie is waarmee de term w_i voorkomt in document d.

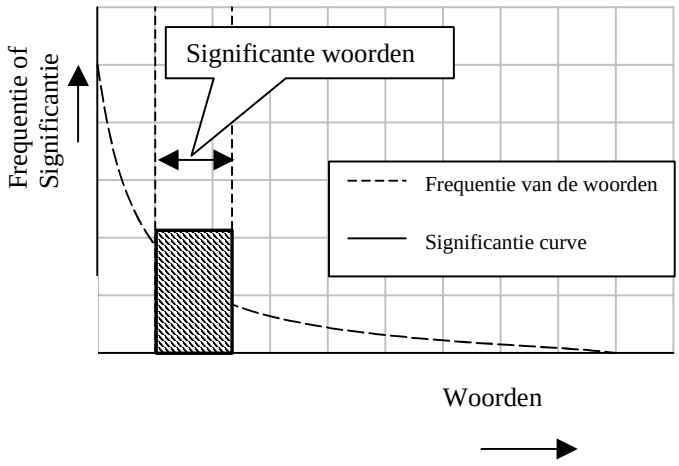
Tot slot moet opgemerkt worden dat TFIDF niet hetzelfde effect zal hebben als de stopwoorden die in paragraaf 6.3.1 zijn besproken. Woorden, zoals ‘de’, ‘het’ en ‘een’, zullen in heel veel documenten voorkomen. Wanneer deze woorden met behulp van een stopwoordenlijst weggefilterd worden, komen ze in het geheel niet meer voor in het label van de pagina. Bij TFIDF verdwijnen deze woorden niet, maar krijgen ze slechts een heel klein gewicht.

In theorie zou TFIDF de stopwoorden een gewicht van 0 moeten geven. Neem als voorbeeld het woord ‘de’. Dit woord zal waarschijnlijk in elk (Nederlands) document voorkomen. Het totaal aantal documenten is dus gelijk aan het aantal documenten waar ‘de’ in voor komt, dus |D| / DF(‘de’) is gelijk aan 1. De IDF is de log hiervan en is gelijk aan 0. Uit de praktijk blijkt het woord ‘de’ niet op alle Nederlandse pagina’s voor te komen. In dit geval is |D| / DF(‘de’) niet gelijk aan 1 en dus ook de IDF niet 0. Het werkt dan dus beter om een stopwoordenlijst te gebruiken om woorden als ‘de’, ‘het’ en ‘een’ weg te filteren in plaats van dit over te laten aan TFIDF.

Door het gebruik van TFIDF worden alle gewichten in de termfrequentievector aangepast. Belangrijke termen in een document krijgen een hogere minder belangrijke krijgen een lage waarde. De spreiding tussen de gewichten wordt daardoor over het algemeen groter.

6.3.5 Significante woorden

Niet alle woorden die in de (aangepaste) termfrequentievector staan, zijn relevant voor de inhoud van de webpagina. De frequentie waarmee woorden voorkomen in een tekst, is een indicator voor de significantie van een woord. Significant wil zeggen dat het woord een bijdrage levert aan de beschrijving van de pagina. Het is niet zo dat naarmate een woord vaker voorkomt op een pagina de significantie ervan ook toeneemt. Woorden die heel vaak op een pagina voorkomen zullen minder significant zijn dan woorden die minder vaak voorkomen. Dit bleek reeds uit voorgaande paragrafen. Denk hierbij bijvoorbeeld aan woorden als ‘de’ ‘het’ en ‘een’. Deze woorden kunnen zeer vaak voorkomen (frequentie) maar zeggen niets bijzonders over de tekst (significantie). In Figuur 13 is de frequentie-significantie curve volgens Luhn [23] getekend. Het gearceerde blokje is toegevoegd en zal kort worden toegelicht.



Figuur 13. Frequentie significantie curve

Uit deze curve blijkt dat de inhoud van een pagina over het algemeen niet goed gekarakteriseerd kan worden wanneer alleen die woorden worden meegenomen die zeer frequent in het document voorkomen. Woorden die heel weinig voorkomen in een

document zeggen ook niet veel over de inhoud van het document. Woorden die niet weinig, maar ook niet heel vaak ten opzichte van de andere woorden voorkomen in een pagina, zeggen het meest over de inhoud van een pagina. Op deze manier worden de meest significante woorden gebruikt om de webpagina te labelen. Dit is in de figuur aangegeven met het gearceerde blokje.

6.4 Conclusies

Dit onderzoek is gericht op de vraag of aanbevelingen van webpagina's gedaan kunnen worden op basis van de interesseprofielen van de gebruikers. In een gebruikerstest, waarin in hoofdstuk 11 wordt teruggekomen, geven gebruikers webpagina's op die zij interessant vinden. Deze webpagina's worden vervolgens gelabeld met een aantal van de in dit hoofdstuk beschreven technieken.

Het labelen op basis van context is niet gebruikt in dit onderzoek. Het zoeken naar webpagina's die verwijzen naar een bepaalde webpagina is niet eenvoudig. Op een webpagina zelf staat geen informatie die aangeeft welke webpagina's verwijzingen hebben naar de betreffende webpagina. Deze informatie is nodig wanneer men een bepaalde webpagina met behulp van labelen door context wil labelen. Het labelen op basis van context is om deze reden in dit onderzoek niet meegenomen.

Omdat niet iedereen een bepaalde webpagina hetzelfde zou labelen wanneer hij of zij dit handmatig zou moeten doen, is het te verwachten dat alleen op basis van labelen met Webpadcategorieën niet alle webpagina's gevonden worden die interessant zouden kunnen zijn voor een gebruiker. Wanneer bijvoorbeeld één persoon een webpagina over de sport voetbal als 'voetbal' labelt, en een andere persoon een andere webpagina die ook over de sport voetbal gaat als 'balsport' labelt, zal geen van beide gebruikers de webpagina van de ander als aanbeveling krijgen omdat de termen niet overeenkomen. Wanneer de termen 'balspel' en 'voetbal' op de webpagina zelf voorkomen, zullen de beide gebruikers de webpagina's wel als aanbeveling krijgen met behulp van het labelen op basis van tekst inhoud. Daarom is in dit onderzoek ook gebruik gemaakt van het labelen op basis van tekst inhoud aangevuld met labelen op basis van HTML opmaak.

Een aantal technieken voor verfijning die besproken zijn in dit hoofdstuk zijn toegepast op de verschillende labels. Synoniemen wordt niet gebruikt omdat er niet de beschikking was over een synoniemenlijst. Stemming is gedaan met behulp van de CELEX database zoals besproken in paragraaf 6.3.2. Er is ook gebruik gemaakt van een stopwoordenlijst die al in eerdere projecten bij KPN Research succesvol is toegepast.

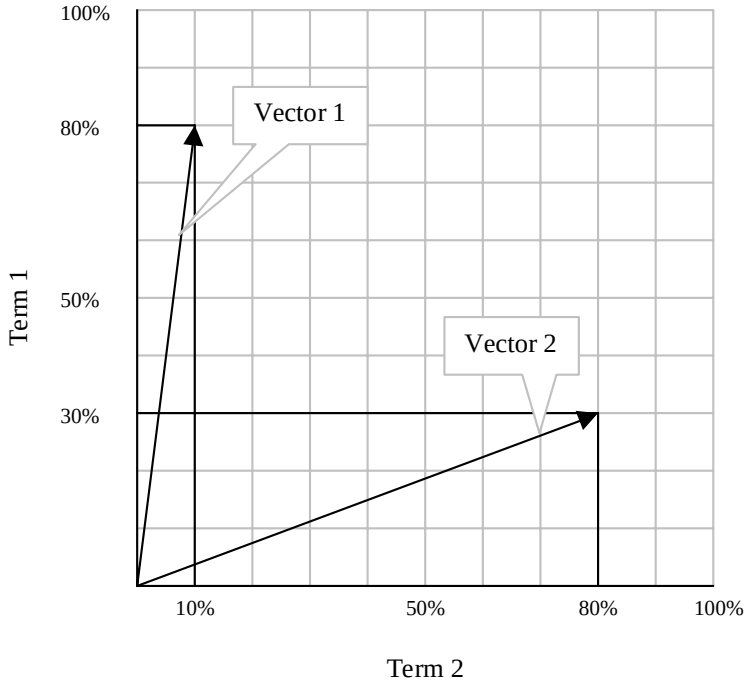
7 Matching

In hoofdstuk 6 is besproken hoe labels worden gemaakt. Om deze labels te kunnen gebruiken voor het doen van aanbevelingen, moet er een methode bedacht worden om labels met gebruikersprofielen te kunnen vergelijken. Een gebruikersprofiel bevat de interesses van een bepaalde gebruiker. Zonder een goede vergelijkingsmethode kan niet worden voorspeld of een pagina bij de interesses van een gebruiker past.

In de eerste paragraaf van dit hoofdstuk wordt eerst besproken wat de betekenis van een vectorrepresentatie van een document is. Daarna wordt een drietal methoden besproken hoe twee documenten met elkaar vergeleken kunnen worden. Vervolgens wordt in de tweede paragraaf van dit hoofdstuk één van deze drie methoden verder uitgelicht. Deze methode zal ook in de rest het onderzoek gebruikt worden.

7.1 Webpagina's vergelijken

De labels van de webpagina's bestaan uit woorden (termen). Bij elk woord staat een gewicht dat aangeeft in welke mate dit woord kenmerkend is voor de webpagina. Ook een gebruikersprofiel bestaat uit woorden met gewichten. Zowel de labels als de gebruikersprofielen kunnen gerepresenteerd worden als vectoren. Een voorbeeld van een vector is weergegeven in Figuur 14. Elke term in de vector wordt gerepresenteerd als een aparte dimensie in de figuur. Om de figuur overzichtelijk te houden zijn de vectoren 2-dimensionaal. Dit houdt in dat de vectoren maar uit 2 woorden bestaan met een bijbehorend gewicht.
Uit de plaats van de punt van de vector kan worden bepaald wat het gewicht is van elk van de termen voor het betreffende document. Wanneer term 1 bijvoorbeeld 'computers' is en term 2 'Internet', gaat document 1, gerepresenteerd door vector 1, voor het grootste gedeelte over 'computers' (80%), en slechts een klein gedeelte over 'Internet' (10%). De percentage worden bijvoorbeeld bepaald door de frequentie waarmee een term op een pagina voorkomt.



Figuur 14. Voorbeeld van twee vectoren

Om de overeenkomst tussen twee vectoren te bepalen wordt een bepaalde afstandsmaat gedefinieerd. Er zijn verschillende afstandsmaten:

- De hoek tussen twee vectoren
  De grootte van de hoek bepaalt de mate waarin twee documenten van elkaar verschillen. Hoe groter de hoek, hoe meer de vectoren van elkaar verschillen. Wanneer de hoek 0 is, zijn de twee documenten qua inhoud aan elkaar gelijk. De lengte van de vectoren is hier niet van belang. Bij de overige twee methoden is dit wel van belang.

- De Euclidische afstandsmaat
  Deze afstand is de lengte van de vector die de twee vectoren met elkaar verbindt. Hoe langer deze vector is hoe meer de twee documenten van elkaar verschillen. De lengte wordt berekend met behulp van de volgende formule:

  $$D = \sqrt{(x_2 - x_1)^2 + (y_1 - y_2)^2}$$

  Om de berekening van de Euclidische afstandsmaat wat te versnellen wordt het wordteltrekken vaak achterwege gelaten. Het vergelijken van afstanden is immers hetzelfde als het vergelijken van gekwadrateerde afstanden.

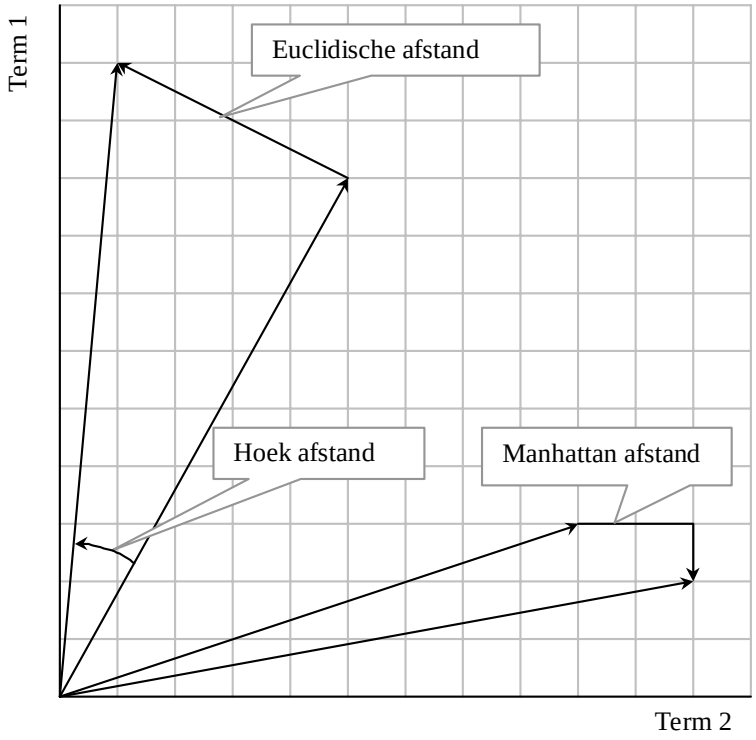
- De Manhattan- of Cityblock- afstandsmaat
  Deze afstandmaat wordt ook gedefinieerd door de lengte van de vector die de twee vectoren met elkaar verbindt net als bij de Euclidische afstandsmaat het geval is. Het verschil met de Euclidische afstand is echter, dat de verbindingsvector alleen uit loodrechte lijnen mag bestaan. Ook hier geldt, hoe langer de vector hoe meer de twee documenten van elkaar verschillen.

  De afstand wordt berekend met de volgende formule:

  $$D = | x_2 - x_1 | + | y_1 - y_2 |$$

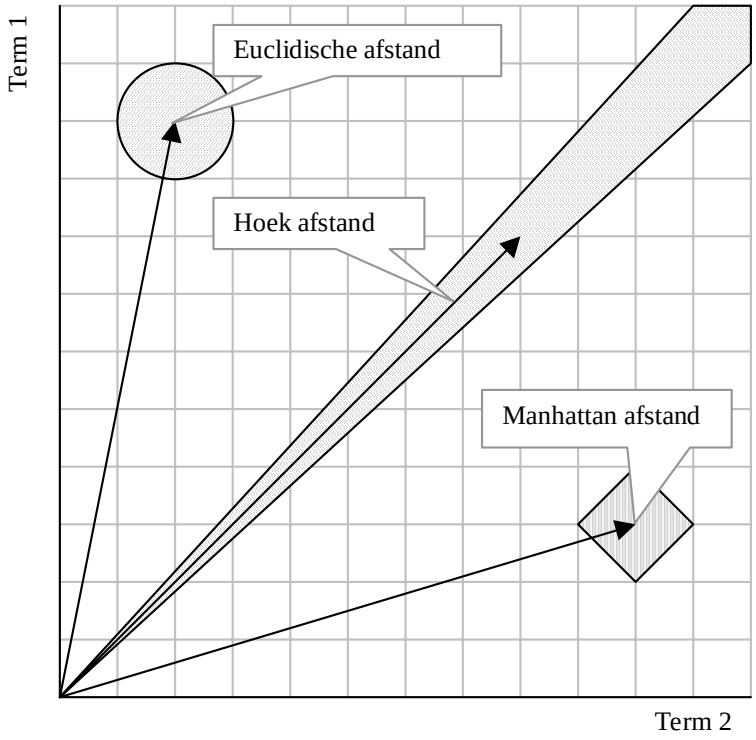
  De berekening van de Manhattan afstandsmaat neemt minder tijd in beslag dan de Euclidische afstandsmaat. Voor de berekening van de Manhattan afstand hoeven geen kwadraten berekend te worden.

In Figuur 15 staan de verschillende afstandsmaten afgebeeld.



Figuur 15. Afstandsmaten

Om te bepalen of een vector ‘lijkt’ op een andere vector kan één van de bovengenoemde afstandsmaten worden gebruikt. Elke afstandsmaat zal een andere mate van overeenkomst geven. Om de verschillen tussen de afstandsmaten duidelijk te maken, wordt gekeken naar het gebied om een vector heen. De afstand tussen de vector en een willekeurige andere vector in dit gebied is hoogstens 1. In Figuur 16 is dit aangegeven.

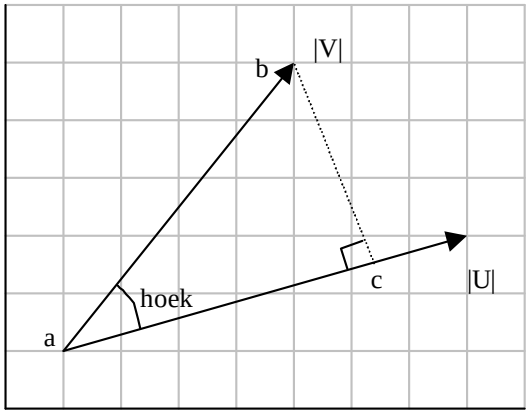


Figuur 16. Gebieden waar de afstand 1 is

Het is duidelijk te zien dat de mate van gelijkheid verschilt per afstandsmaat. De gebieden zijn niet allemaal even groot. Vooral de hoekafstand vertoont een afwijkend gedrag. Bij deze afstandsmaat wordt niet gekeken naar de lengte van de vector. Slechts de hoek is bepalend voor de mate van overeenkomst.

7.2 Hoek-matching

Voor het bepalen van de hoek tussen twee vectoren is een snel algoritme beschikbaar. Dit algoritme bepaald de hoek tussen elke twee vectoren met een ongelimiteerde dimensie. Om het algoritme uit te leggen wordt gebruik gemaakt van een 2-dimensionaal voorbeeld



Figuur 17. Hoekafstand tussen twee vectoren

In Figuur 17 zijn twee vectoren U en V te zien. Wanneer de hoek bepaald moet worden tussen deze twee vectoren, is de lengte niet meer van belang. Voor het bepalen van de hoek kan dus uitgegaan worden van de driehoek a, b, c. De cosinus van de hoek is dan gelijk aan de lengte van zijde ac gedeeld door de lengte van de zijde ab.

$$\cos(\alpha) = \frac{ac}{ab} \tag{1}$$

Omdat voor de hoek de lengte van de vectoren niet uitmaakt, kunnen deze genormaliseerd worden. Een genormaliseerde vector is een vector waarvan de lengte precies 1 is. Zijde ab is daarmee precies 1. De lengte van ac is nog onbekend.
Om de lengte van ac te bepalen zijn de coördinaten van het punt c nodig. Om deze te vinden, kan gebruik worden gemaakt van een *orthogonale projectie*.

$$c = \frac{V \cdot U}{U \cdot U} U \tag{2}$$

Omdat de lengte van de vector U gelijk is aan 1 (U is genormaliseerd) is het dot-product van U gelijk aan 1.
In formule 2 wordt het punt c uitgerekend in absolute coördinaten. Hiervoor wordt vermenigvuldigd met de vector U. Omdat alleen de lengte van ac berekend moet worden, hoeft deze laatste vermenigvuldiging niet uitgevoerd te worden. Formule 2 kan dus vereenvoudigd worden tot de volgende formule:

$$c = V \cdot U = \sum_i v_i \cdot u_i \tag{3}$$

Het vergelijken van twee hoeken met elkaar, is hetzelfde als het vergelijken van de cosinus van de twee hoeken. De verhouding tussen de cosinussen van de hoeken of de hoeken zelf blijft gelijk. Het omrekenen van een cosinus van een hoek naar de werkelijke hoek kost ook nog eens extra rekentijd. Het omrekenen wordt daarom achterwege gelaten.

Het voorbeeld is gegeven voor een 2-dimensionaal voorbeeld maar de formule werk ook voor meerdimensionale vectoren.

7.3 Conclusies

Hoekmatching is de meest gangbare techniek om vectoren met elkaar te vergelijken om te bepalen of documenten over hetzelfde onderwerp gaan. Door gebruik te maken van de berekenmethode zoals in de vorige paragraaf is beschreven, is de rekentijd tot een minimum te beperken. Voor dit onderzoek is daarom gebruik gemaakt van het hoekvergelijking algoritme zoals beschreven in de vorige paragraaf. Dit algoritme is snel, ook wanneer het aantal dimensies van de vectoren toeneemt.

8 Clusteren

Zoals al eerder is besproken, wordt het gebruikersprofiel vergeleken met de labels van de verschillende webpagina's om te bepalen of deze interessant genoeg zijn voor de gebruiker om in de cache van de Family Proxy op te slaan. Er moet dus gezocht worden op labels van webpagina's waar de woorden uit een gebruikersprofiel in voorkomen. Dit kan een zeer tijdrovend proces worden, zeker wanneer er veel labels van webpagina's zijn. Om sneller te kunnen matchen, kunnen de labels van tevoren worden geclusterd. Labels die veel op elkaar lijken, dat wil zeggen webpagina's die qua inhoud over hetzelfde gaan, komen op deze manier in één cluster terecht.
Van een cluster met webpagina's kan ook een label worden gemaakt. Dit label stelt het middelpunt van het cluster voor. Bij het zoeken naar interessante webpagina's worden niet de labels van alle webpagina's apart vergeleken met het gebruikersprofiel, maar wordt het label van een clusters vergeleken met het gebruikersprofiel. Wanneer een gebruikersprofiel overeenkomt met het label van een cluster, is er een grote kans dat het gebruikersprofiel ook overeenkomt met de labels van de afzonderlijke webpagina's in het betreffende cluster.

Om het matching proces sneller te maken, kan een gebruikersprofiel dus vergeleken worden met alleen de verschillende clusterlabels. Wanneer een gebruikersprofiel binnen een cluster valt, worden de webpagina's die hier binnen vallen als aanbevelingen gedaan.

In paragraaf 8.1 wordt een overzicht gegeven van een aantal verschillende clusteringmethoden. Daarna wordt in paragraaf 8.2 een hele andere vorm van clusteren behandeld, namelijk Latent Semantic Indexing (LSI). Tot slot worden in paragraaf 8.3 de problemen met clusteren en de keuzes voor het onderzoek besproken.

8.1 Overzicht clusteringprincipes

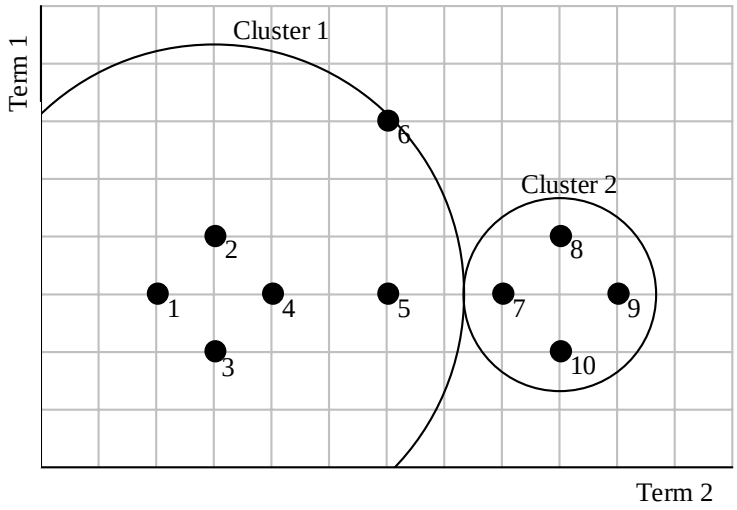
Voor het clusteren van de labels van de webpagina's kan gekozen worden voor verschillende clusteralgoritmen. In deze paragraaf zullen een aantal verschillende soorten clustertechnieken worden besproken.

Clustertechnieken kunnen worden onderverdeeld in een aantal categorieën.

- Hard clustering versus Fuzzy clustering
- Supervised versus Unsupervised clustering
- Verdelend versus Agglomeratief

Hieronder zullen deze verschillende categorieën worden beschreven.

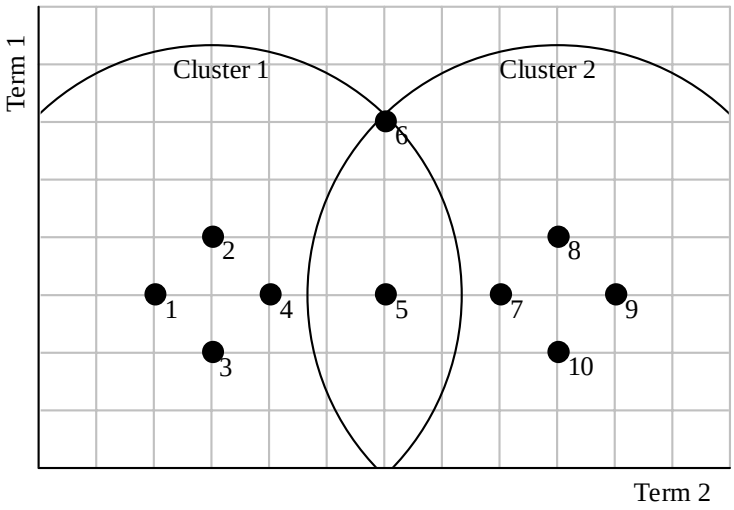
De eerste onderverdeling die kan worden gemaakt is Fuzzy Clustering of Hard Clustering. Bij Hard Clustering hoort een element of wel, of niet tot een cluster. Bij Fuzzy Clustering daarentegen hoort een element met een bepaald gewicht tot een cluster. Dit gewicht varieert hierbij tussen de 0 en 1. Een element dat zeker tot een cluster behoort krijgt een waarde van 1. Een element dat iets buiten het cluster valt, maar er toch nog wel bij hoort, kan een lagere waarde krijgen, bijvoorbeeld 0.8. Een element kan ook, in tegenstelling tot Hard Clustering, tot meerdere clusters behoren. In Figuur 18 is een voorbeeld gegeven van Hard Clustering. Hier is duidelijk te zien dat ondanks dat punt 5 en 6 aan de rand van cluster 1 vallen deze toch geheel tot dit cluster worden gerekend.



	1	2	3	4	5	6	7	8	9	10
Cluster 1	1	1	1	1	1	1	0	0	0	0
Cluster 2	0	0	0	0	0	0	1	1	1	1

Figuur 18. Hard Clustering

In het voorbeeld in Figuur 19 zijn dezelfde punten gegeven, maar dan geclusterd met behulp van Fuzzy Clustering. In dit voorbeeld is te zien dat de punten 5 en 6 niet voor 100% bij cluster 1 horen zoals in het geval van Hard Clustering, maar een klein beetje bij cluster 1 en een klein beetje bij cluster 2. In het voorbeeld van Figuur 19 is de som van de gewichten van de punten 5 en 6 gelijk aan 1. In dit voorbeeld betekent dit dat hoewel punt 6 op de rand van cluster 1 en op de rand van cluster 2 ligt, deze wel dezelfde gewichten heeft als punt 5 welke veel meer binnen de twee clusterranden valt. Om het onderscheid tussen punt 5 en 6 duidelijk te maken zou punt 6 voor cluster 1 en 2 een lager gewicht kunnen krijgen (bijvoorbeeld 0.25) zodat de gewichten van punt 6 niet meer tot 1 sommeren. Er kan op deze manier aangegeven worden dat het niet duidelijk is in welk cluster punt 6 valt.



	1	2	3	4	5	6	7	8	9	10
Cluster 1	1	1	1	1	0.5	0.5	0	0	0	0
Cluster 2	0	0	0	0	0.5	0.5	1	1	1	1

Figuur 19. Fuzzy Clustering

De tweede onderverdeling die gemaakt kan worden bij clusteralgoritmen is tussen een agglomeratief algoritme of een verdelend algoritme. Een agglomeratief clusteralgoritme begint met een heleboel kleine clusters waarbij ieder element in zijn eigen cluster valt. Daarna worden clusters samengenomen om tot een optimale clusterverdeling te komen. De verdelende algoritmen beginnen met één cluster waar alle elementen in zitten. Vervolgens wordt dit cluster net zo lang verder onderverdeeld totdat een optimale clustering is gevonden.

De laatste onderverdeling die kan worden gemaakt is die naar Supervised Clustering of Unsupervised Clustering. Bij Supervised Clustering worden er randvoorwaarden opgegeven voordat met het clusteren wordt begonnen. Bijvoorbeeld hoeveel clusters er gemaakt moeten worden of wat de maximale afstand is van een punt tot het centrum van een cluster. Bij Unsupervised Clustering worden deze beslissingen genomen door het clusteralgoritme. Bij Unsupervised Clustering is er geen a-priori kennis nodig van de structuur van de data die moet worden geclusterd om een afstandsmaat tussen de te clusteren elementen te kunnen definiëren. Er hoeft geen vooronderzoek te worden gedaan naar de eigenschappen van de data die moet worden geclusterd. Dit is wel het geval bij Supervised Clustering.

8.2 Latent Semantic Indexing

In paragraaf 7.1 is al uitgelegd dat hoe kleiner het aantal woorden in het label, hoe beter het vergelijken gaat. Bij heel veel verschillende termen wordt de overlap tussen verschillende documenten kleiner. Er zijn in dit geval weinig woorden die met elkaar overeenkomen. De kunst is dus om het aantal termen zo klein mogelijk te houden, zodat de kans op een match wordt vergroot. In paragraaf 6.3 is al uitgelegd hoe het aantal termen gereduceerd kan worden door gebruik te maken van bijvoorbeeld woordenlijsten.

Een andere manier om het aantal verschillende termen te reduceren is Latent Semantic Indexing, kortweg LSI. LSI is een statistische methode. Er wordt geen gebruik gemaakt van betekenissen van woorden maar puur van statistische gegevens om de data te reduceren.
Door het toepassen van LSI worden niet de documenten zelf, maar de termen die de documenten beschrijven geclusterd. Dit komt neer op het afbeelden van een hoogdimensionale ruimte (veel verschillende termen) waar de documenten zich in bevinden, naar een ruimte met een lagere dimensie (geclusterde termen). Elke dimensie is een term (of begrip). Hoe minder termen (of begrippen) er in totaal zijn, hoe beter het classificeren van documenten in deze ruimte is.

Om relevante webdocumenten te vinden, worden deze op inhoud met elkaar vergeleken. Er wordt vaak gezocht naar documenten waarin bepaalde woorden voorkomen. Hoe vaker deze woorden in een document voorkomen, hoe hoger het document in de resultatenlijst komt te staan.
Een probleem hierbij is dat er meerdere woorden zijn die dezelfde betekenis hebben. Woorden als ‘computer’ en ‘PC’ lijken niet op elkaar qua letters, maar betekenen wel hetzelfde. Wanneer gezocht wordt op ‘computer’, zullen documenten waar alleen ‘PC’ in wordt gebruikt niet worden gevonden.
LSI maakt het mogelijk om relevante documenten te vinden waarin de zoekterm, bijvoorbeeld ‘computer’, niet voorkomt. Wanneer bijvoorbeeld op ‘computer’ wordt gezocht, zullen ook de documenten waar het woord ‘PC’ in zit, worden gevonden. Een denkfout die veel gemaakt wordt, is dat LSI synoniemen kan opsporen. Dit is echter onjuist. Woorden die vaak samen in één document voorkomen worden door LSI in dezelfde deelruimte geplaatst. Wanneer ‘Computer’ en ‘PC’ vaak samen in één document voorkomen, legt LSI een relatie tussen beide woorden. De woorden ‘Computer’ en ‘PC’ worden in hetzelfde cluster gestopt. Het is helemaal niet gezegd dat deze woorden ook hetzelfde betekenen. Zo zullen de woorden ‘software’ en ‘computer’ ook veel samen in één document voorkomen en dus waarschijnlijk in hetzelfde cluster terecht komen. Deze woorden zijn echter beslist geen synoniemen. Dit houdt echter wel in dat, indien er wordt gezocht op de term ‘computer’, er documenten kunnen worden gevonden die enkel het woord ‘software’ bevatten. Hier dient dus rekening mee te worden gehouden.

Om te laten zien wat LSI voor effect heeft, wordt er in deze paragraaf een voorbeeld van LSI gegeven. Dit voorbeeld is overgenomen uit het rapport van Nice News Now [22]. Als dataset is gebruik gemaakt van een verzameling titels van rapporten die bij Bellcore verschenen zijn. De dataset van titels is te zien in Tabel 20. De sleutelwoorden zijn cursief gedrukt. Wanneer een woord in tenminste twee documenten voorkomt, wordt dit woord tot een sleutelwoord gerekend. De sleutelwoorden worden gebruikt voor het labelen van de teksten.

1	*Human* Machine *Interface* for Lab ABC *Computer* Application
2	A *Survey* of *User* Opinion of *Computer System Response Time*
3	The *EPS User Interface* Management *System*
4	*System* and *Human System* Engineering Testing of *EPS*
5	Relation of *User*-Perceived *Response Time* to Error Measurement
6	The Generation of Random, Binary, Unordered *Trees*
7	The Intersection *Graph* of Paths in *Trees*
8	*Graph Minors* IV: Widths of *Trees* and Well-Quasi-Ordering
9	*Graph Minors*: A *Survey*

Tabel 20. Titels van rapporten bij Bellcore

In Tabel 21 is de term-document matrix afgebeeld. Hierin staat aangegeven welke termen hoe vaak in welk document voorkomen. Het is duidelijk te zien dat de documenten zich in twee groepen verdelen. De documenten 1 tot en met 5 vormen één groep, en de documenten 6, 7, 8 en 9 vormen de tweede groep.

	Termen	**Documenten**								
		1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*
	Human	1	0	0	1	0	0	0	0	0
	Interface	1	0	1	0	0	0	0	0	0
	Computer	1	1	0	0	0	0	0	0	0
	User	0	1	1	0	1	0	0	0	0
	System	0	1	1	2	0	0	0	0	0
A =	Response	0	1	0	0	1	0	0	0	0
	Time	0	1	0	0	1	0	0	0	0
	EPS	0	0	1	1	0	0	0	0	0
	Survey	0	1	0	0	0	0	0	0	1
	Trees	0	0	0	0	0	1	1	1	0
	Graph	0	0	0	0	0	0	1	1	1
	Minors	0	0	0	0	0	0	0	1	1

Tabel 21. Term-document matrix

De documenten kunnen worden gerepresenteerd in een ruimte. Deze ruimte bestaat dan uit evenveel dimensies als er verschillende termen zijn. De matrix A bestaat uit 12 verschillende temen. In dit geval zijn er dus 12 dimensies. Om het aantal dimensies te verkleinen wordt gebruik gemaakt van Singular Value Decomposition.

Singular Value Decomposition (SVD) is een techniek om een matrix M te splitsen in drie aparte matrices U, E en V. Waarbij E een diagonaalmatrix is waarbij de elementen de eigenwaarden zijn. Door vervolgens in deze matrix de grootste paar eigenwaarden te laten staan en de andere 0 te maken, kan de ruimte waarin de documenten zich bevinden worden verkleind. De dimensie wordt teruggebracht tot het aantal eigenwaarden die blijven staan.

De matrices die door de SVD berekend zijn voor matrix A, zijn in de onderstaande tabellen weergegeven.

	0.22	-0.11	0.29	-0.41	-0.11	-0.34	0.52	-0.06	-0.41
	0.20	-0.07	0.14	-0.55	0.28	0.50	-0.07	-0.01	-0.11
	0.24	0.04	-0.16	-0.60	-0.11	-0.26	-0.30	0.06	0.49
	0.40	0.06	-0.34	0.10	0.33	0.38	0.00	0.00	0.01
	0.64	-0.17	0.36	0.33	-0.16	-0.21	-0.17	0.03	0.27
U =	0.27	0.11	-0.43	0.07	0.08	-0.17	0.28	-0.02	-0.05
	0.27	0.11	-0.43	0.07	0.08	-0.17	0.28	-0.02	-0.05
	0.30	-0.14	0.33	0.19	0.11	0.27	0.03	-0.02	-0.17
	0.21	0.27	-0.18	-0.03	-0.54	0.08	-0.47	-0.04	-0.58
	0.01	0.49	0.23	0.02	0.59	-0.39	-0.29	0.25	-0.23
	0.04	0.62	0.22	0.00	-0.07	0.11	0.16	-0.68	0.23
	0.03	0.45	0.14	-0.01	-0.30	0.28	0.34	0.68	0.18

Tabel 22. Matrix U

	3.34	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	0.00	**2.54**	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	0.00	0.00	2.35	0.00	0.00	0.00	0.00	0.00	0.00
	0.00	0.00	0.00	1.64	0.00	0.00	0.00	0.00	0.00
E =	0.00	0.00	0.00	0.00	1.50	0.00	0.00	0.00	0.00
	0.00	0.00	0.00	0.00	0.00	1.31	0.00	0.00	0.00
	0.00	0.00	0.00	0.00	0.00	0.00	0.85	0.00	0.00
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.56	0.00
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.36

Tabel 23. Diagonaalmatrix E

	0.20	-0.06	0.11	-0.95	0.05	-0.08	0.18	-0.01	-0.06
	0.61	0.17	-0.50	-0.03	-0.21	-0.26	-0.43	0.05	0.24
	0.46	-0.13	0.21	0.04	0.38	0.72	-0.24	0.01	0.02
	0.54	-0.23	0.57	0.27	-0.21	-0.37	0.26	-0.02	-0.08
V =	0.28	0.11	-0.51	0.15	0.33	0.03	0.67	-0.06	-0.26
	0.00	0.19	0.10	0.02	0.39	-0.30	-0.34	0.45	-0.62
	0.01	0.44	0.19	0.02	0.35	-0.21	-0.15	-0.76	0.02
	0.02	0.62	0.25	0.01	0.15	0.00	0.25	0.45	0.52
	0.08	0.53	0.08	-0.02	-0.60	0.36	0.04	-0.07	-0.45

Tabel 24. Matrix V

Om de samenhang tussen de documenten straks grafisch weer te kunnen geven, wordt het aantal dimensies terug gebracht tot 2. Dit wordt bereikt door de eerste 2 getallen in de diagonaalmatrix E ongewijzigd te laten, en de andere getallen 0 te maken. Vervolgens kan de matrix A opnieuw berekend worden door matrix U, E en V met elkaar te vermenigvuldigen. Wat nu ontstaat is een aangepaste term-document matrix. Hierin zijn de gewichten van de termen in de verschillende matrices aangepast. De aangepaste matrix A, hierna aangeduid als A', is te zien in onderstaande figuur.

	Termen	**Documenten**								
		1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*
	Human	0.16	0.40	0.38	0.47	0.18	-0.05	-0.12	-0.16	-0.09
	Interface	0.14	0.37	0.33	0.40	0.17	-0.03	-0.07	-0.10	-0.04
	Computer	0.15	0.51	0.36	0.41	0.24	0.02	0.06	0.09	0.12
	User	0.26	0.84	0.61	0.70	0.39	0.03	0.08	0.12	0.19
	System	0.45	1.23	1.05	1.27	0.56	-0.07	-0.15	-0.21	-0.05
A' =	Response	0.16	0.58	0.38	0.42	0.28	0.06	0.13	0.19	0.22
	Time	0.16	0.58	0.38	0.42	0.28	0.06	0.13	0.19	0.22
	EPS	0.22	0.55	0.51	0.63	0.24	-0.07	-0.14	-0.20	-0.11
	Survey	0.10	0.53	0.23	0.21	0.27	0.14	0.31	0.44	0.43
	Trees	-0.06	0.23	-0.14	-0.27	0.14	0.24	0.55	0.77	0.66
	Graph	-0.06	0.34	-0.15	-0.30	0.20	0.31	0.69	0.98	0.85
	Minors	-0.04	0.25	-0.10	-0.21	0.15	0.22	0.50	0.71	0.62

Tabel 25. Matrix A' na SVD

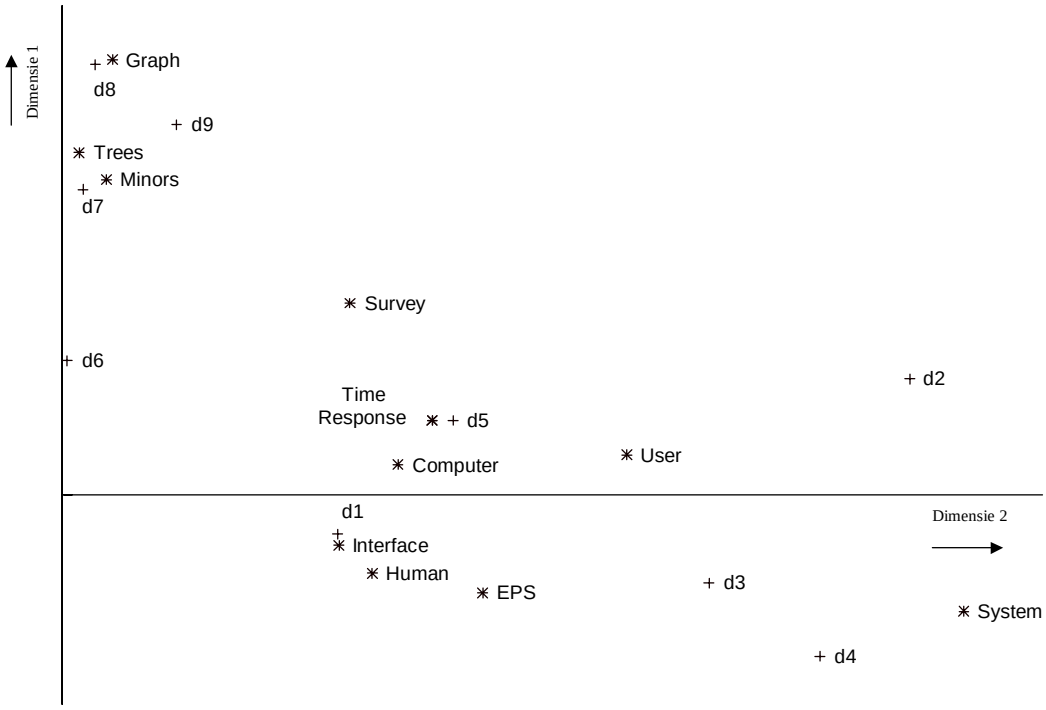
Ook hier is de verdeling in twee groepen weer heel duidelijk te zien. De termen *human* tot en met *EPS* horen heel duidelijk bij de eerste 5 documenten. In dit deel van de matrix staan alleen maar positieve getallen. De laatste 4 documenten hebben bij deze termen

hoofdzakelijk negatieve of heel kleine gewichten, wat erop duidt dat deze termen niet op de laatste 4 documenten slaan.

Het eerste wat opvalt, is dat er nergens meer nullen staan in de matrix. Elke term zegt iets over een document. Een negatief getal houdt in dat de term niet op het document slaat, een positief getal dat de term wel op dat document slaat. Hoe groter de waarde hoe belangrijker deze term is voor het document, wanneer de waarde positief is. Een grote negatieve waarde houdt in dat het document helemaal niet door deze term kan worden gerepresenteerd. Het feit dat er nergens meer nullen staan, heeft effect op het aantal documenten dat wordt gevonden bij het zoeken. Wanneer gezocht wordt op *computer*, worden niet alleen de documenten waar het woord *computer* in voorkomt (document 1 en 2) gevonden, maar alle documenten die in dezelfde LSI dimensie vallen (documenten 3, 4 en 5). LSI berekent dus een schatting van hoe vaak het woord *computer* voor zou komen in een oneindig grote steekproef van documenten waar de overige woorden van deze cluster ook in staan.

Het aantal LSI dimensies is bepalend voor de scheiding tussen de documenten. Algemeen geldt dat hoe groter het aantal dimensies hoe duidelijker de scheiding. Wanneer in dit voorbeeld voor 9 LSI dimensies wordt gekozen (het aantal documenten), zal elk document in zijn eigen cluster vallen. Oftewel het generaliseren wordt moeilijker. Hoe groter het aantal dimensies, hoe minder er gegeneraliseerd wordt. Wanneer er echter voor te weinig dimensies gekozen wordt, zal de generalisatie te groot worden. Documenten die niets met elkaar te maken hebben gaan in de LSI dimensie op elkaar lijken. Wat de optimale grootte zou moeten zijn van het aantal LSI dimensies is alleen maar te bepalen door ‘trial and error’.

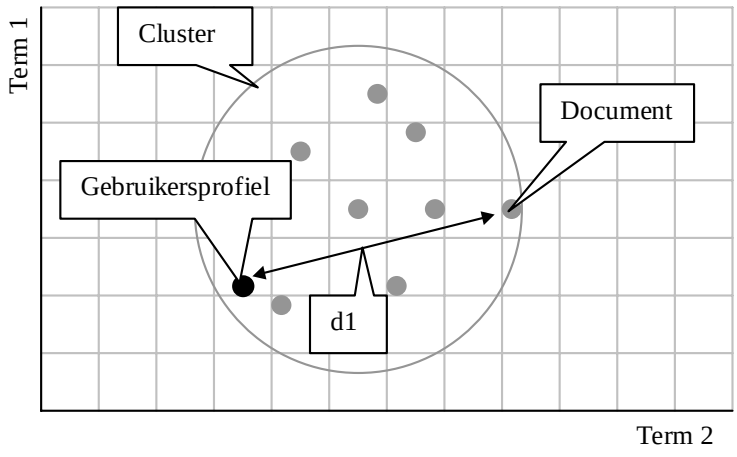
In Figuur 26 is de data uit Tabel 20 geplot in een 2-dimensionale ruimte. De plusjes stellen de documenten voor, de sterretjes de kernwoorden. De documenten zijn gelabeld met d1 tot en met d9. Deze figuur geeft voor elk van de termen en documenten aan hoe belangrijk de dimensies 1 en 2 zijn voor de karakterisering ervan. Duidelijk is te zien dat de documenten d6, d7, d8 en d9 hoofdzakelijk gekarakteriseerd worden door de tweede dimensie. De overige documenten worden meer gekarakteriseerd door de eerste dimensie. De term *Survey* valt precies tussen beide dimensies in. Deze term is dan ook de enige term die in beide clusters van documenten wordt gebruikt (namelijk d2 en d9).



Figuur 26. Documenten en termen geplot in een 2-dimensionale ruimte

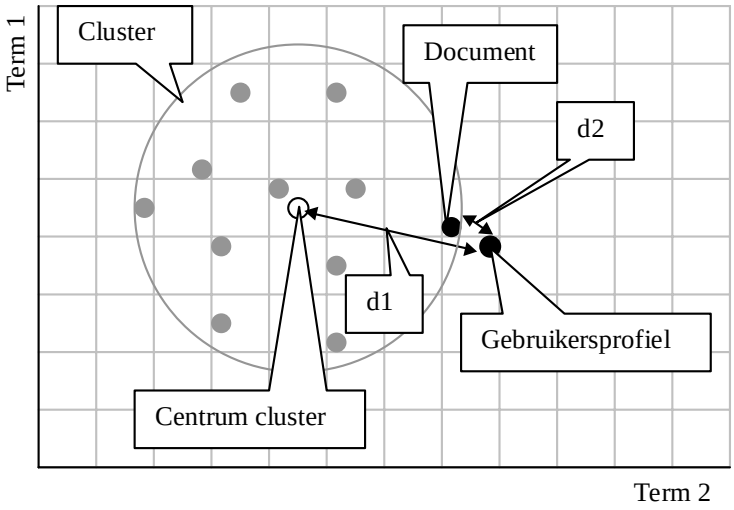
8.3 Conclusies

In het begin van dit hoofdstuk werd voorgesteld dat wanneer een label van een cluster matcht met het profiel waarop gezocht werd, alle URL's in dit cluster als aanbevelingen gedaan kunnen worden. Andersom wordt ook gesteld dat wanneer een profiel niet matcht met een label van een cluster, dit profiel ook niet matcht met de afzonderlijke URL's in dit cluster. Uit de volgende twee voorbeelden, die zijn weergegeven in Figuur 27 en Figuur 28 is te zien dat dit niet altijd helemaal waar is.



Figuur 27. Probleem met grote clusters (1)

In de bovenstaande figuur is het duidelijk te zien dat het gebruikersprofiel wel in het cluster valt. Of het gebruikersprofiel overeenkomt met het aangegeven document wat aan de andere kant van het cluster ligt, is nog maar de vraag. Zeker bij grote clusters zal dit vaak niet het geval zijn.



Figuur 28. Probleem met grote clusters (2)

In de bovenstaande figuur valt het gebruikersprofiel niet binnen het aangegeven cluster. De afstand tussen het gebruikersprofiel en het aangegeven document dat wel binnen het cluster valt is heel klein. Op basis van de afstand tussen het gebruikersprofiel en het document, komt het document wel overeen met het gebruikersprofiel. Wanneer alleen wordt gekeken naar het cluster waar het gebruikersprofiel in valt, wordt in dit geval foutief geconcludeerd dat het document niet overeenkomt met het gebruikersprofiel.

Uit bovenstaande twee voorbeelden blijkt dat er voor een betrouwbaar matchingproces nog aan een aantal extra voorwaarden moet worden voldaan. Ten eerste kan, wanneer een gebruikersprofiel in een cluster valt, niet direct gezegd worden dat alle documenten binnen dit cluster matchen met het gebruikersprofiel. Dit moet eerst nog gecontroleerd worden door de documenten binnen het cluster afzonderlijk te vergelijken met het gebruikersprofiel.
Ten tweede moet de eis dat een gebruikersprofiel binnen een cluster moet vallen, iets versoepeld worden. Een gebruikersprofiel moet binnen een bepaalde afstand van de clustergrens vallen om nog tot het cluster te worden gerekend.

Er is voor het prototype gekeken naar implementatie mogelijkheden voor LSI. Het is gebleken dat LSI heel rekenintensief is en niet schaalbaar. Omdat het prototype schaalbaar moet zijn, is ervoor gekozen om LSI niet toe te passen.
Andere clustermethoden kunnen wel gebruikt worden. In dit onderzoek is daar geen gebruik van gemaakt omdat de meerwaarde daarvan te klein is. Clusteren zou in vervolgonderzoek van belang kunnen zijn en is daarom gedocumenteerd in dit hoofdstuk.

9 Systeem Analyse

In de voorgaande hoofdstukken is de theorie uitgelegd die nodig is om te bepalen of een webpagina interessant zou kunnen zijn voor een bepaald persoon. In de verschillende hoofdstukken is al aangegeven welke methoden gebruikt zullen worden en waarom. Om te kunnen bepalen of de gekozen methoden ook in de praktijk werken, is een prototype gebouwd.

Dit hoofdstuk beschrijft de analysestap die uitgevoerd is voor het ontwikkelen van het prototype. In de eerste paragraaf worden de eisen aan het prototype uiteengezet, daarna worden de processen in het systeem geanalyseerd. Vervolgens worden de gegevensstromen in kaart gebracht. Tot slot wordt in dit hoofdstuk de architectuur van het prototype geanalyseerd. In het volgende hoofdstuk wordt verder ingegaan op het ontwerp van het prototype.

9.1 Specificaties

Het doel van het prototype is het uittesten van technieken zodat deze technieken in de Family Proxy kunnen worden gebruikt. De Family Proxy is daarom als uitgangspunt genomen bij het maken van de specificatie.

Globaal moet het prototype een aantal fundamentele eisen voldoen, te weten:

- Ten eerste moeten er gegevens van gebruikers kunnen worden verzameld.
- Van deze gebruikers moet automatisch de interesses kunnen worden bepaald.
- Van de webpagina's die de gebruikers bezoeken moet automatisch een beschrijving worden gemaakt.
- Met deze beschrijvingen en de interesses van de gebruiker moeten aanbevelingen kunnen worden gegenereerd.
- Deze aanbevelingen moeten tot slot worden gepresenteerd aan de gebruikers.

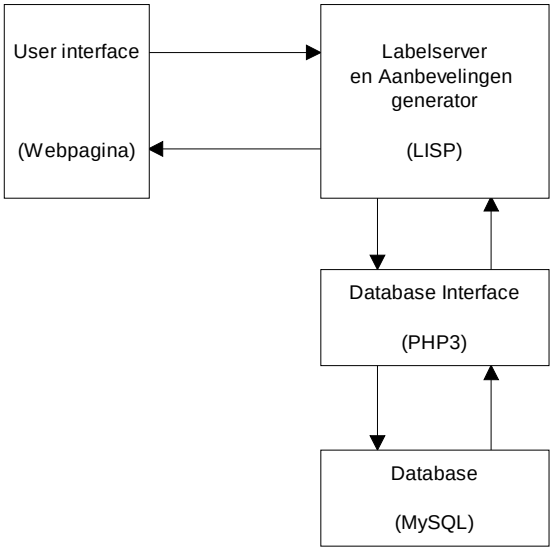
De eis dat de interesses van de gebruikers automatisch bepaald moeten worden, hangt samen met de eis dat de Family Proxy het Internetten voor de gebruiker gemakkelijk moet maken. Als gebruikers zelf hun interesseprofiel moeten opstellen, gaat dit voorbij aan het basisidee van de Family Proxy.

De hoofdeis is dat het systeem aanbevelingen voor een gebruiker moet genereren die op basis van de interesses van de gebruiker zijn gebaseerd. Om de gebruiker zoveel mogelijk te ontlasten moeten de interesses van de gebruiker impliciet uit het surfgedrag van de gebruiker worden afgeleid.
Daarnaast moet het systeem schaalbaar zijn. Het systeem dient op kleine schaal getest worden, maar moet wel schaalbaar zijn zodat het ook werkt in de situatie dat er sprake is van veel gebruikers met veel informatie.

9.2 Systeem architectuur

Het systeem bestaat uit vier delen. Het eerste deel is de User Interface. Deze zorgt ervoor dat informatie van gebruikers, zoals bezochte webpagina's en dergelijke, bij het systeem bekend is. Daarnaast is er een Labelserver en een Aanbevelingen generator

welke webpagina's voorziet van een label en aanbevelingen genereert voor de gebruikers. Er is een database nodig om de labels van de webpagina's en informatie van gebruikers in op te slaan. Om deze database te kunnen benaderen is een Interface nodig tussen de Labelserver en de Labeldatabase. In Figuur 29 is de systeemarchitectuur weergegeven.

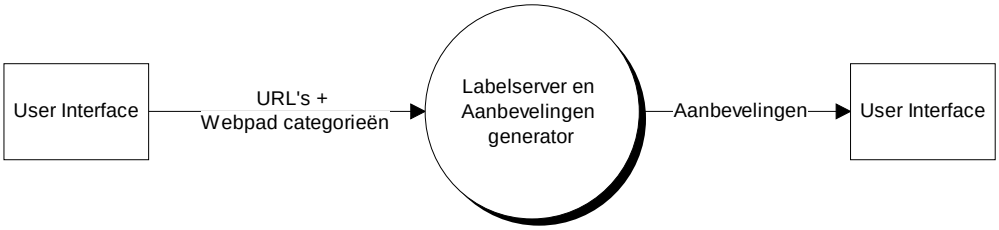


Figuur 29. Systeem Architectuur

9.3 Procesanalyse en Dataflow

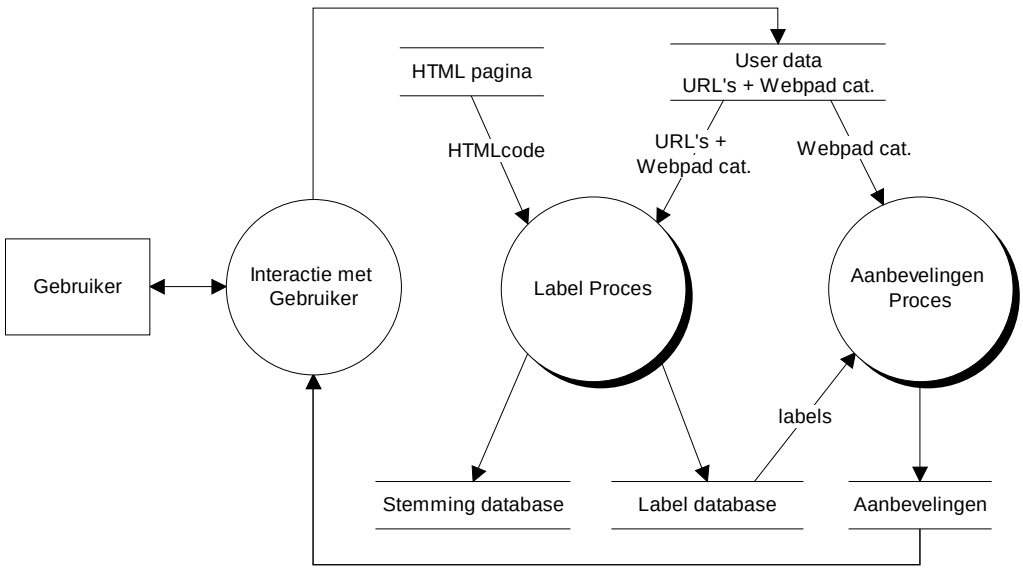
Een dataflowdiagram (DFD) geeft alle processen en de informatiestromen tussen deze processen weer die in een systeem aanwezig zijn. In deze paragraaf worden de dataflow diagrammen van het prototype getoond en uitgelegd.

Het dataflowdiagram in Figuur 30 laat de informatiestroom zien op het hoogste niveau. Dit diagram laat nog weinig van de interne werking van het systeem zien. Het systeem heeft als invoer de URL's die de gebruikers bezoeken en de Webpadcategorieën van de webpagina's die de gebruikers in hun *navigation tree* hebben staan. Deze informatie wordt in het systeem gebruikt en verwerkt. Uiteindelijk komen er aanbevelingen uit voor elke gebruiker. Dit dataflow diagram wordt verder uitgewerkt in Figuur 31.



Figuur 30. Level 0 DFD

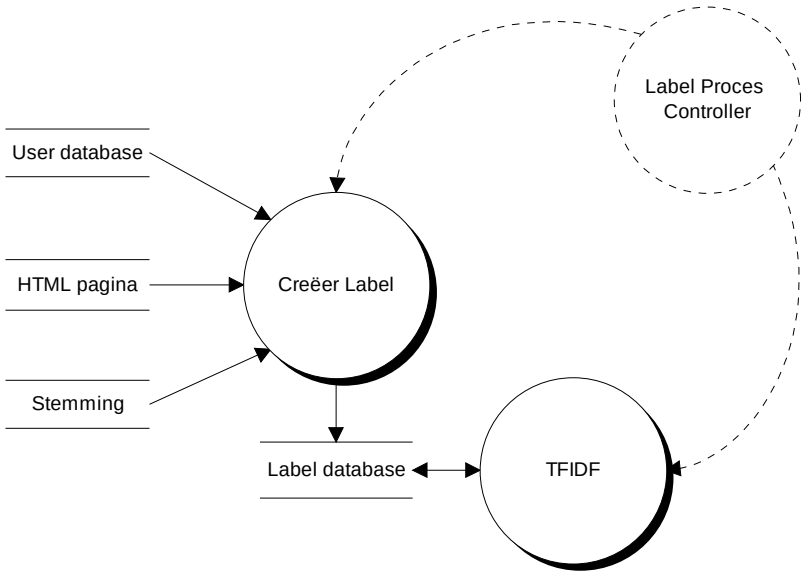
In Figuur 31 is duidelijk te zien welke informatie er binnen het systeem wordt gebruikt en opgeslagen. In de figuur zijn de twee hoofdprocessen in het systeem, het Aanbevelingen Proces en het Label Proces, aangegeven. Deze lezen de informatie uit de verschillende databases en verwerken dit (uiteindelijk) tot een set met aanbevelingen voor iedere gebruiker. De processen die een schaduwrand hebben worden verderop in deze paragraaf verder uitgewerkt.



Figuur 31. Level 1 DFD

De informatie van de gebruikers (bezochte URL's en Webpadcategorieën) worden in een database gestopt. Het Label Proces leest uit deze database de URL's die gelabeld moeten worden. Het labelen wordt gedaan zowel op basis van de inhoud van de webpagina (de HTML-code) als de beschrijvingen van de gebruikers (de Webpadcategorieën). De labels van de webpagina worden vervolgens opgeslagen in een database.

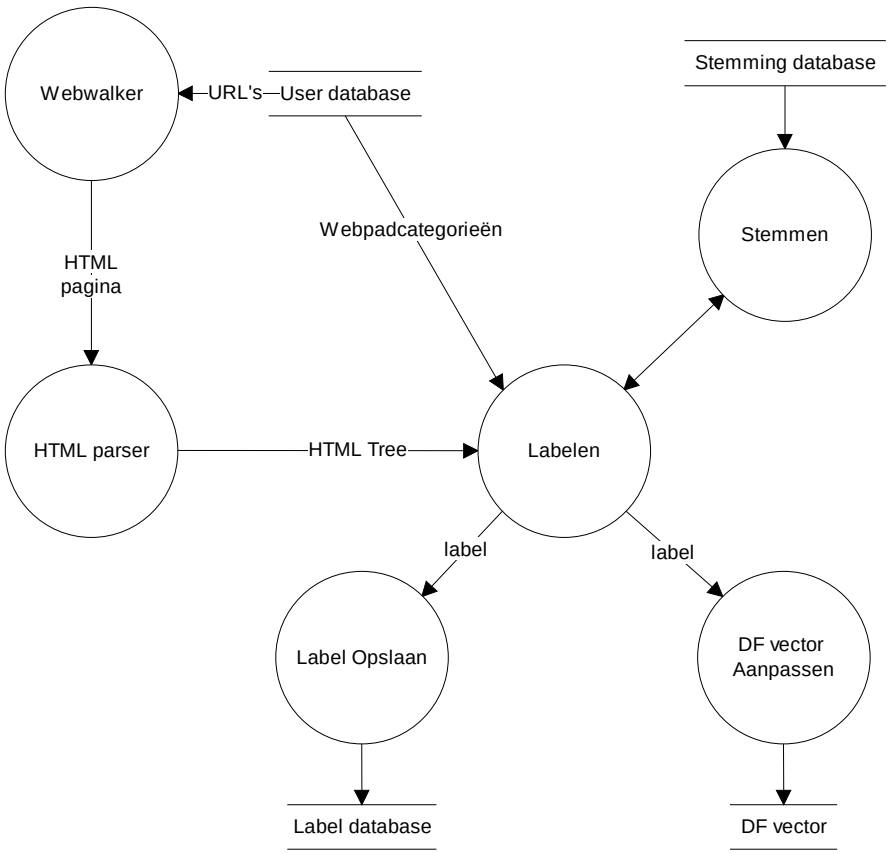
Het Aanbevelingen Proces genereert, op basis van de beschrijvingen die gebruikers aan door hun bezochte webpagina's hebben gegeven, een aantal sets van aanbevelingen. Deze aanbevelingen worden ook opgeslagen in een database.



Figuur 32. DFD Label Proces

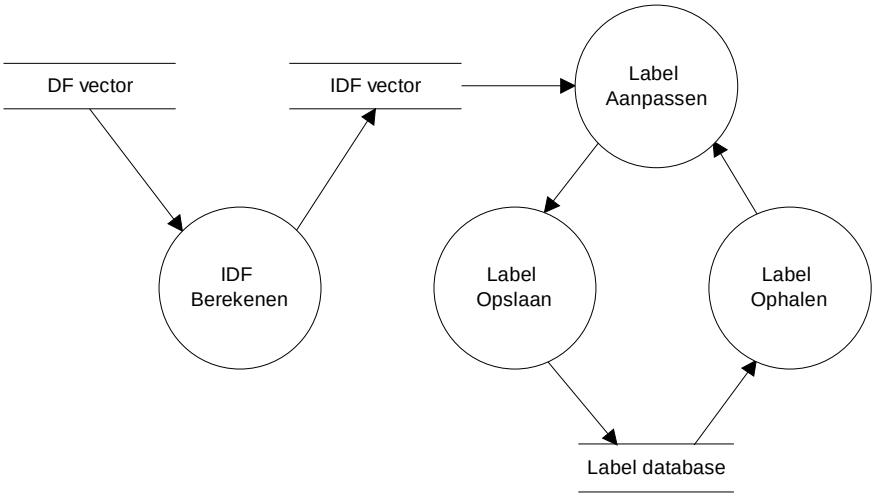
De controller stuurt eerst het 'Creëer Label' proces aan. Met behulp van dit 'Creëer Labels' proces worden er twee labels gemaakt. Eén label op basis van de inhoud van een webpagina en éénop basis van de Webpadcategorieën uit de database. Nadat alle labels van de URL's die in de database staan, zijn gemaakt, kan TFIDF worden toegepast om de labels te verfijnen.
Het dataflow diagram van het proces 'Creëer Label' is weergegeven in Figuur 33. Het

dataflow diagram van het aanpassen van de labels op basis van TFIDF is weergegeven in Figuur 34.



Figuur 33. DFD Creëer label proces

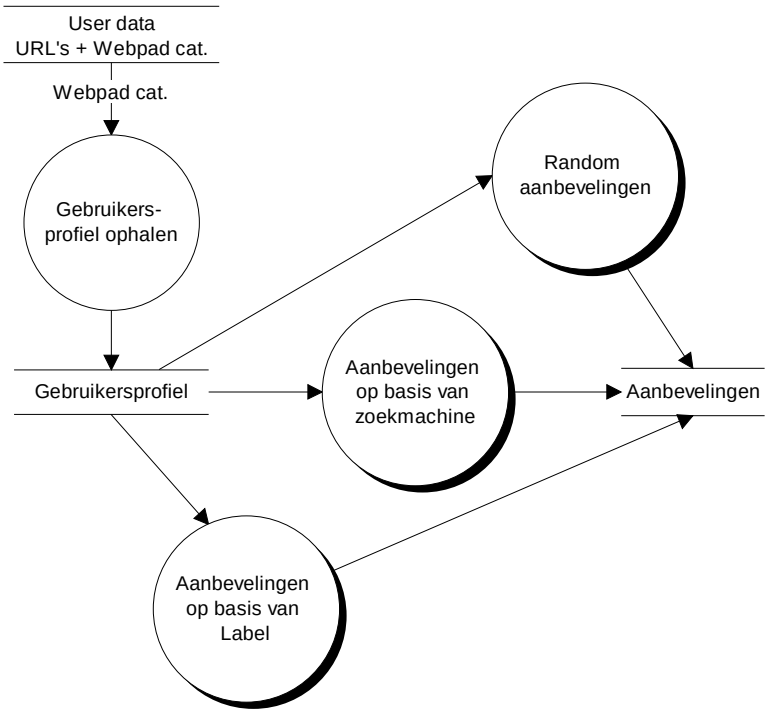
Het dataflow diagram van het proces ‘Creëer Label’ begint linksboven in Figuur 33 met de Webwalker. De Webwalker leest de URL van de te labelen webpagina uit een database. De HTML-pagina wordt door de Webwalker van het Internet opgehaald en naar de HTML-parser gestuurd. De HTML-parser geconverteerd de HTML-pagina tot een HTML-boom. Deze HTML-boom wordt gebruikt in het proces ‘Labelen’. In het proces ‘Labelen’ wordt het daadwerkelijke label van een webpagina gemaakt. Het proces ‘Label Opslaan’ ontvangt de labels van het proces ‘Labelen’ en slaat deze op in de Label database. Het proces ‘DF vector Aanpassen’ krijgt ook de labels van het proces ‘Labelen’ en maakt hiermee een DF vector welke later nodig is om TFIDF te kunnen toepassen.



Figuur 34. DFD TFIDF

Het dataflow diagram van het toepassen van TFIDF is te zien in Figuur 34. In het proces TFIDF worden de labels van de webpagina's aangepast volgens de principes die zijn uitgelegd in paragraaf 6.3.4.

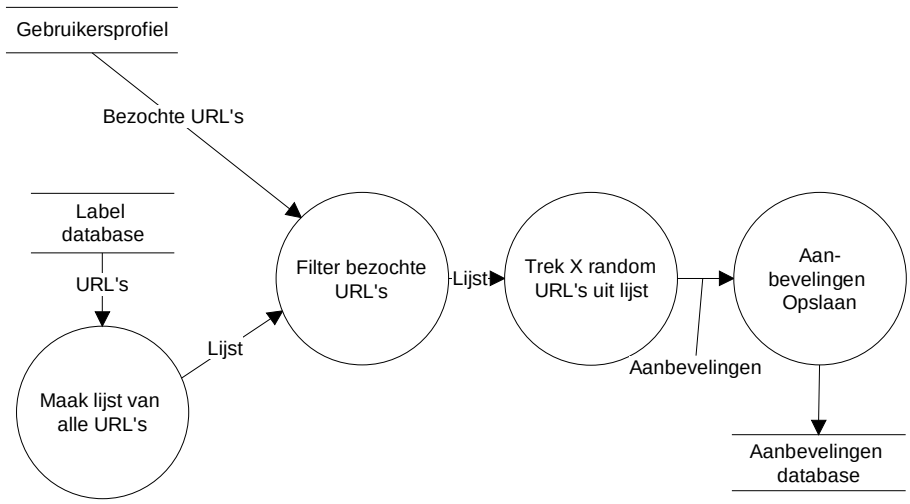
Eerst wordt er een IDF vector berekend op basis van de DF vector en het totaal aantal gelabeld webpagina's. Deze IDF vector wordt gebruikt voor het aanpassen van de labels van de webpagina's. De labels worden opgehaald uit de Label database via het proces 'Label Ophalen'. De labels worden met het proces 'Label Aanpassen' genormaliseerd met de IDF vector. Het proces 'Label Opslaan' schrijft het aangepaste label weer terug in de Label database.



Figuur 35. DFD aanbevelingen genereren

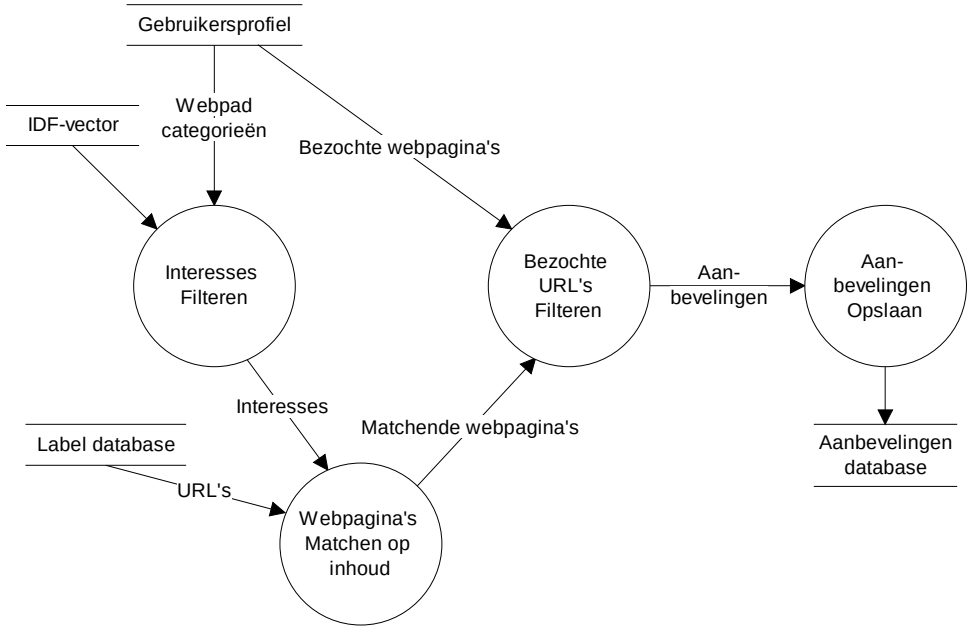
Het genereren van aanbevelingen is een proces waarbij vijf verschillende aanbevelingslijsten worden gegenereerd. Deze aanbevelingslijsten zijn onder te verdelen in drie soorten. De willekeurige (of random) aanbevelingen, de aanbevelingen die gegenereerd zijn op basis van de labels en aanbevelingen die gegenereerd zijn met

behulp van een zoekmachine. In Figuur 35 is het toplevel dataflowdiagram van het genereren aanbevelingen te zien.



Figuur 36. Random aanbevelingen

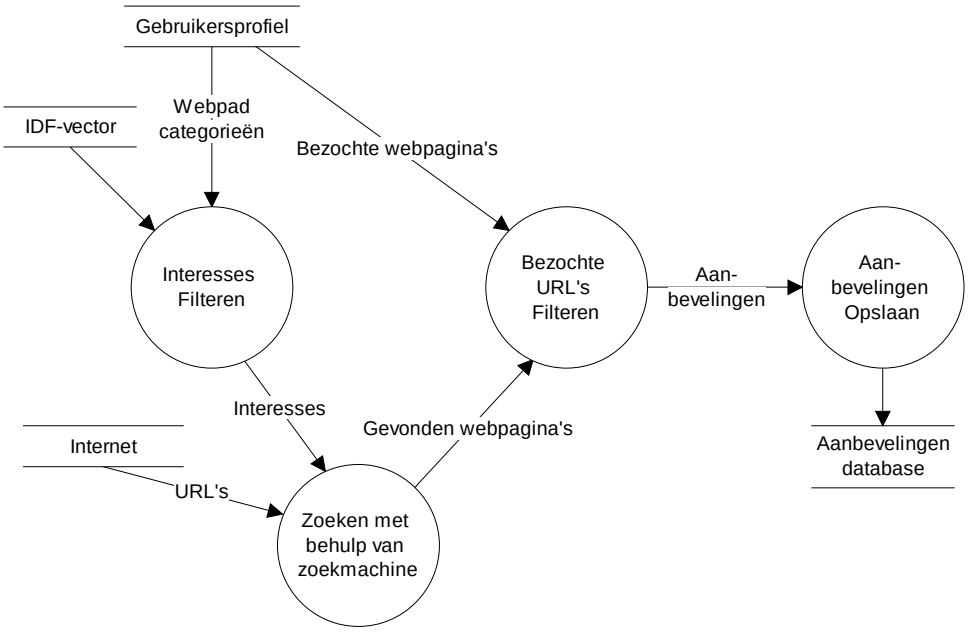
Het proces van het random genereren van aanbevelingen is weergegeven in Figuur 36. Bij het genereren van random aanbevelingen wordt er van alle in het systeem aanwezige URL's een lijst gemaakt. Uit deze lijst worden de URL's die de betreffende persoon al heeft bezocht, weggefilterd. Uit deze lijst van URL's wordt vervolgens een willekeurig aantal URL's getrokken en als aanbeveling aan de gebruiker gedaan.



Figuur 37. Aanbevelingen op basis van label

In Figuur 37 is het proces van het genereren van aanbevelingen op basis van de labels in de Labeldatabase weergegeven. Deze methode is ingewikkelder dan het genereren van random aanbevelingen. Bij het genereren van random aanbevelingen kon volstaan worden met willekeurig URL's kiezen. Bij het genereren van aanbevelingen op basis van de labels moet naar de inhoud van een webpagina worden gekeken. De interesses van een gebruiker moeten overeenkomen met de inhoud van de webpagina. Een interesseprofiel van een gebruiker wordt aan de hand van de Webpadcategorieën, die de gebruiker bij zijn favoriete webpagina's heeft staan, gemaakt. Met dit interesseprofiel wordt in de labeldatabase gezocht naar webpagina's die overeenkomen met dit profiel.

Ook hier worden de webpagina's die de gebruiker al kende weggefilterd. De overgebleven webpagina's worden als aanbeveling gemarkeerd en opgeslagen in De Aanbevelingen database.

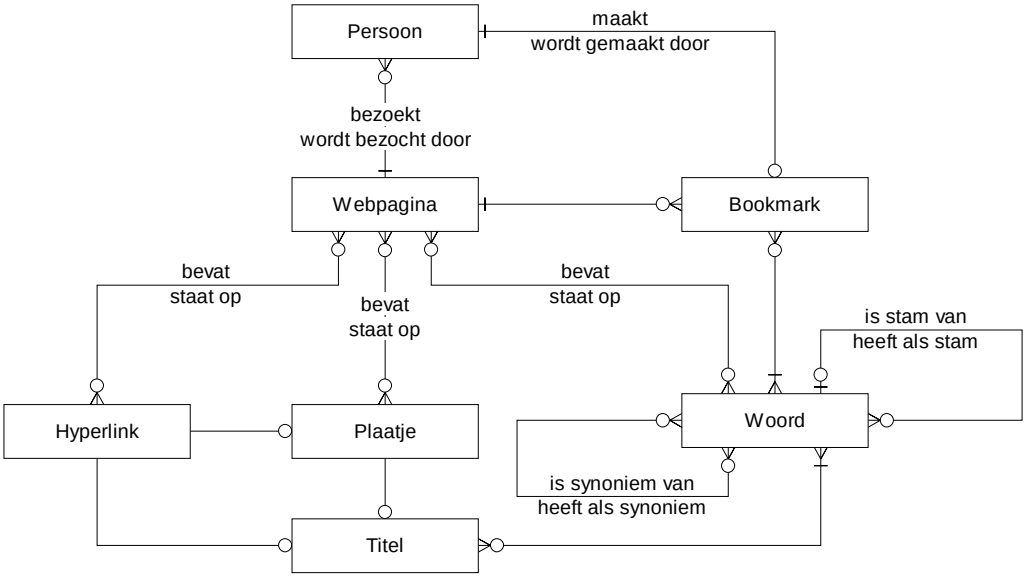


Figuur 38. Aanbevelingen op basis van zoekmachine

Het genereren van aanbevelingen op basis van de labels en het genereren van aanbevelingen met behulp van een zoekmachine lijken erg sterk op elkaar. Het enige verschil is het zoek en match algoritme. Het dataflow diagram van het genereren van aanbevelingen op basis van een zoekmachine is weergegeven in Figuur 38. Bij het genereren van aanbevelingen op basis van de labels wordt er in de Label database gezocht naar webpagina's die matchen met een bepaald interesseprofiel van een gebruiker. In het geval van het genereren van aanbevelingen op basis van een zoekmachine worden de termen uit een interesseprofiel aan een Internet zoekmachine aangeboden. De resultaten hiervan worden gebruikt als aanbeveling.

9.4 Datamodel

Het datamodel geeft aan welke entiteiten zich bevinden in het probleemdomein en wat de relaties zijn tussen deze entiteiten. Om dit alles duidelijk weer te geven is een plaatje gemaakt van het entiteit relatie diagram. Dit is te zien in Figuur 39.



Figuur 39. Entiteit relatie model (ERD)

Een persoon bezoekt een webpagina. Deze webpagina kan nul of meerdere hyperlinks bevatten, hij kan nul of meerder laatjes hebben en hij kan bestaan uit nul of meerdere woorden.
Een hyperlink kan op zijn beurt weer bestaan uit een plaatje of uit een tekstuele ‘link’. Aan de hand van deze hyperlink kan een gebruiker doorklikken naar een andere webpagina.
Het plaatje van de hyperlink kan tekst bevatten. Met deze tekst is niet de voor de gebruiker zichtbare tekst in het plaatje bedoeld, maar de onderliggende titel van het plaatje. Deze titel is ook weer opgebouwd uit woorden. Deze woorden kunnen diverse (of geen) synoniemen hebben en kunnen tot een stam worden herleid aan de hand waarvan bijvoorbeeld stemming plaats kan vinden. Woorden kunnen ook dienen om de naam van een bookmark of een categorie van bookmarks te omschrijven. Deze bookmarks worden door een gebruiker aangemaakt.

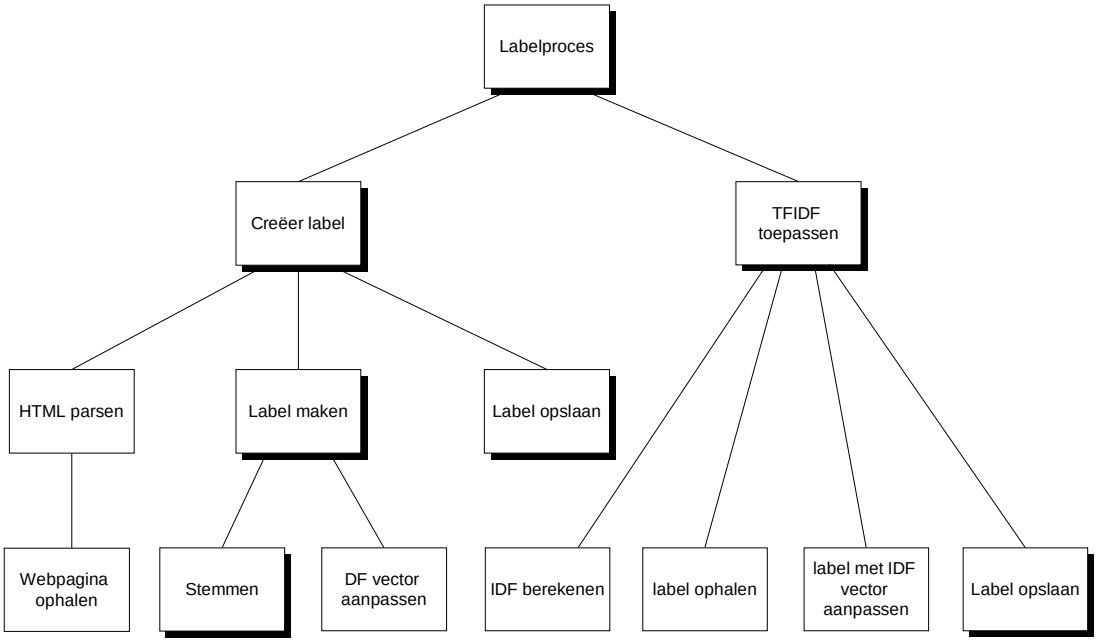
10 Systeem Ontwerp en Implementatie

In dit hoofdstuk wordt ingegaan op het ontwerp en de implementatie van het prototype van het aanbevelingensysteem.

In de eerste paragraaf van dit hoofdstuk worden de dataflow diagrammen uit het vorige hoofdstuk geconverteerd naar programma-structuurdiagrammen. In de tweede paragraaf worden de afzonderlijke processen uit deze structuurdiagrammen verder gespecificeerd en wordt nader ingaan op de details en implementatie van de processen. Tot slot wordt van het ERD uit het vorige hoofdstuk een ontwerp van de database gemaakt. In deze paragraaf worden ook de keuzes voor de database uitgewerkt.

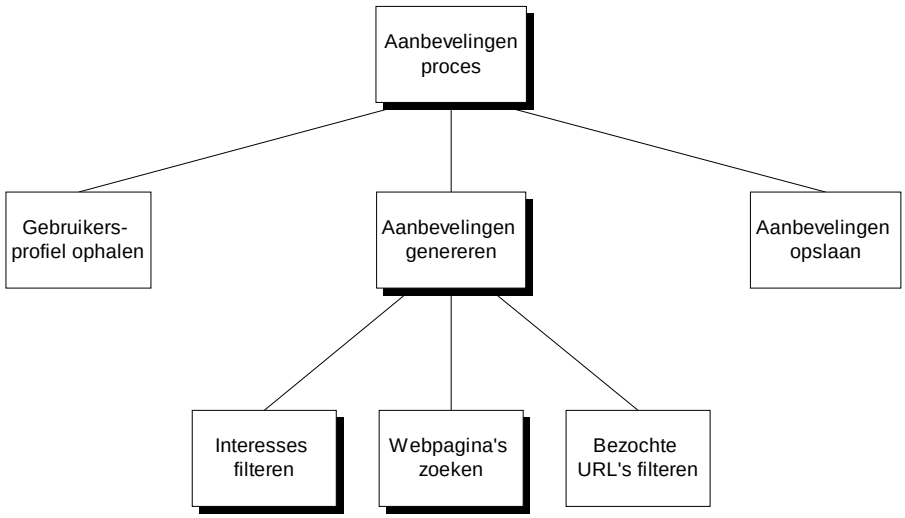
10.1 Programmastructuur

In deze paragraaf worden de dataflow diagrammen uit paragraaf 9.3 geconverteerd naar programma-structuurdiagrammen. Het prototype is opgesplitst in twee delen, het Labelproces en het Aanbevelingenproces. Dit komt ook weer terug in de programma-structuurdiagrammen naar voren. De programma-structuurdiagrammen geven aan in welke volgorde de processen worden doorlopen. De processen in het diagram worden van links naar rechts en van onder naar boven doorlopen (depth-first).
In het Labelproces worden de labels gecreëerd van de webpagina's. Wanneer alle labels zijn gecreëerd worden de labels aangepast met behulp van TFIDF. In Figuur 40 is de programmastructuur van het labelproces weergegeven. Van de processen die in deze figuur zijn getekend met een schaduwrand wordt in de volgende paragraaf een proces flowchart uitgewerkt. De overige processen zijn eenvoudig en hebben geen verdere uitleg nodig.



Figuur 40. Programmastructuur voor Labelproces

Het tweede deel van het programma bestaat uit het genereren van aanbevelingen, het Aanbevelingenproces. In Figuur 41 is het programma-structuurdiagram weergegeven van het Aanbevelingenproces.



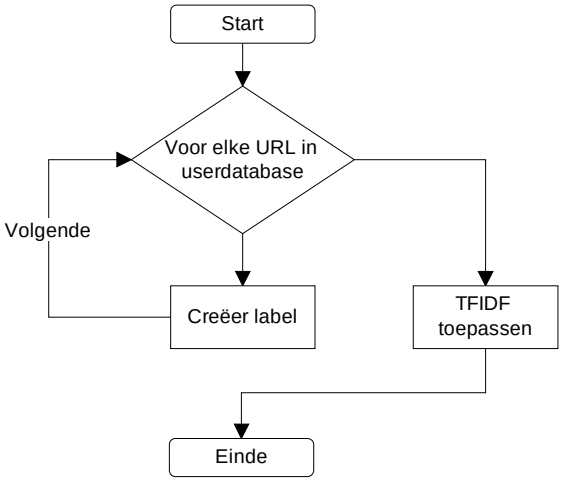
Figuur 41. Programmastructuur voor Aanbevelingenproces

10.2 Proces Flowcharts

De programma-structuurdiagrammen geven de structuur en volgorde van het prototype aan. De proces flowcharts geven een gedetailleerd ontwerp weer van de processen zodat deze naar code omgezet kunnen worden. In deze paragraaf wordt eerst het Labelproces gedetailleerd besproken, vervolgens wordt het Aanbevelingenproces onder de loep genomen.

10.2.1 Labelproces

Voor elke URL die een gebruiker heeft bezocht, wordt een label gecreëerd. Wanneer alle URL's uit de userdatabase zijn gelabeld, worden de labels aangepast met TFIDF (zie paragraaf 6.3.4). De TFIDF-vector is een genormaliseerde vector die rekening houdt met het feit dat sommige woorden op heel veel pagina's voorkomen, en dus niet veel zeggen over de inhoud van een pagina. De mate waarin een woord de inhoud van een pagina beschrijft is omgekeerd evenredig met het aantal documenten waar het woord op voorkomt. Een woord dat op alle pagina's voorkomt, zal niet iets specifieks zeggen over de inhoud van een pagina. Deze term krijgt dus een lage waarde.
Het is ook mogelijk om TFIDF tijdens het creëren van een label uit te voeren. Voor elke volgende webpagina wordt de TFIDF berekening nauwkeuriger. Pas wanneer voor elke webpagina's een label gecreëerd is, is de TFIDF vector het meest nauwkeurig.
Wanneer de labels achteraf worden aangepast door toepassing van TFIDF, moet er een extra keer door de hele Label database worden gelopen. Voor kleine aantallen webpagina's is de overhead om de labels achteraf aan te passen met TFIDF niet groot. De TFIDF berekening is wel het meest nauwkeurig wanneer de berekening achteraf wordt gedaan. Voor het prototype worden geen grote hoeveelheden webpagina's gelabeld. Er is daarom gekozen voor het loskoppelen van het creëren van de labels en het aanpassen van de labels door toepassing van TFIDF.



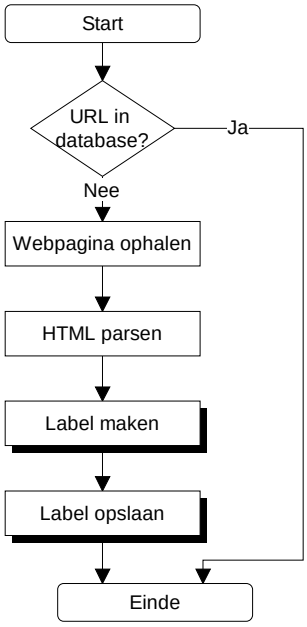
Figuur 42. Procesflowchart Labelproces

De implementatie van het Labelproces is weergegeven in Figuur 43. Eerst wordt er een lijst gemaakt van alle te labelen URL's. Vervolgens worden met behulp van een Webwalker de webpagina's van het Internet gehaald, gelabeld en in de database opgeslagen. Dit proces wordt op de volgende bladzijde beschreven.
Wanneer alle webpagina's gelabeld zijn, wordt de Inverse Document Vector berekend. Vervolgens worden alle labels één voor één weer uit de database gelezen, aangepast met de IDF vector en weer opgeslagen in de database.

```
;; this function automatically labels all url's in the testdata database
(defmethod label-testdata()
  (let
   ((url-list
     (read
      (make-string-input-stream
       (database-interface::send-sql-query
         labels
         "query=select distinct url from testdata where url like '|%/|'")))))
    (dolist (item url-list)
      (w4::label-url-agent (symbol-name (first item)) 0))
    (labeldatabase::calculate-idf-vector)
   (dolist (item url-list)
      (labeldatabase::calculate-label (symbol-name (first item))))))
```

Figuur 43. Implementatie van het Labelproces

Het creëren van een label is weergegeven in Figuur 44. Voor het creëren van een label van een webpagina wordt eerst gekeken of de betreffende webpagina al in de database zit. Wanneer dit het geval is, zal deze webpagina overgeslagen worden. Wanneer de webpagina nog niet in de labeldatabase zit, wordt de webpagina van het Internet opgehaald. De HTML-code die de webpagina beschrijft, wordt door een HTML-parser gehaald, die van de HTML-code een nette datastructuur maakt die in de opvolgende processen eenvoudig is te gebruiken. De datastructuur die uit de HTML-parser komt wordt vervolgens gebruikt voor het maken van een label. Tot slot wordt dit label opgeslagen in de database.



Figuur 44. Procesflowchart Creëer label

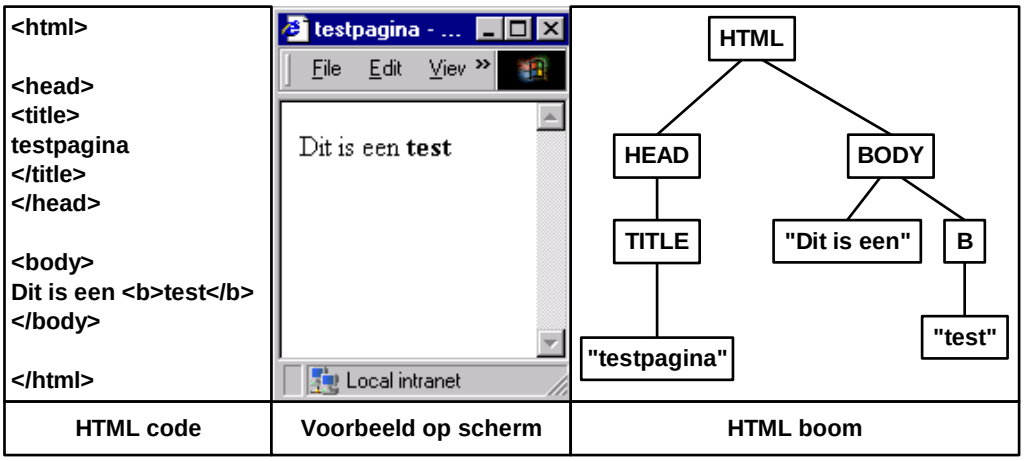
Het ophalen van een webpagina wordt gedaan met behulp van een zogenaamde Webwalker. Dit is een programma dat, gegeven een aantal constraints, webpagina's van het Internet haalt. Eén van de constraints hierbij is de vraag hoeveel recursielagen diep de Webwalker moet gaan met het ophalen van webpagina's. Voor het prototype is het niet nodig om dieper te gaan dan één laag diep, namelijk de webpagina zelf waar de Webwalker mee gestart is.
De Webwalker roept voor elke webpagina die hij van het Internet haalt de functie 'label-url' aan. Deze functie bevat de functionaliteit om de webpagina te labelen.
De implementatie van de webwalker functies zijn weergegeven in Figuur 45.

```
(defun label-url-agent (url depth &key hosts last-modification content-type
                                  directory-p (respect-robot-protocol-p nil))
  (with-activity ("label-url-agent"
                  (:report-stream '*standard-output*)
                  :constraints `((depth ,depth)
                                 (no-cycles)
                                 ,.(when directory-p
                                     `((url-parent-subsumed-by-directory-path
,(url:path url))))
                                 ,.(when hosts
                                     `((url-referrer-host ,hosts)))
                                 ,.(when last-modification
                                     `((header-last-modification #'< ,last-
modification)))
                                 ,.(when content-type
                                     `((header-content-type ,content-type)))
                                 ,.(when respect-robot-protocol-p
                                    (list '(header-robots-allowed))))
                  :actions `((label-url) (generate-inferiors)))
    (walk url activity)))

(define-action-type
    label-url
    (:standard
     :documentation
        "An action that labels an url and stores this label in the database")
    (action activity url)
  (declare (ignore action))
  (let* ((report-stream (report-stream activity))
         (headers (get-resource-headers activity url))
         (content (get-resource-content activity url))
         (content-type (second (get-header :content-type headers)))
         (date (get-header :date headers))
         (remote-host (host-string url))
         (depth (depth))
         (url-string ""))
    (if (equal content-type ':HTML)
        (progn
          (setf url-string
            (concatenate
              'string (scheme url)
              "://"
              (host-string url)
              "/"
              (if (path url)
              (let ((str ""))
               (dolist (item (path url) str)
                 (setf str (concatenate 'string str item "/")))))
              (if (object url) (object url))
              (if (extension url) (concatenate 'string "." (extension url)))))
          (if (label-database::is-url-in-database url-string)
              (format t "url ~a is already in the database~%" url-string)
            ;create and store label
            (progn
             (let ((url-label (make-instance 'label-database::url-class
                                             :url url-string
                                             :timestamp (get-universal-time))))
              (label-database::make-label url-label content)
              (label-database::store-label url-label)
                ;update df vector
              (label-database::update-document-frequency-vector
               (label-database::url-class-tf-vector url-label))
              (format t "url ~a labeled and stored~%" url-string)))))
      (format t "no html page~%"))))
```

Figuur 45. Webwalker functies

Het label dat door wordt gemaakt, is gebaseerd op de inhoud van de webpagina. De HTML-code van een webpagina bevat niet alleen de tekst op de pagina zelf, maar ook de opmaakcodes. Om eenvoudig onderscheid te kunnen maken tussen de opmaak en de tekst zelf, en om duidelijk te zien welke opmaak bij welke tekst hoort, kan een HTML-parser de HTML-code van een webpagina parsen. Een parser is een programma dat als invoer een reeks instructies krijgt, bijvoorbeeld HTML tags, deze in stukken verdeeld en er een eenvoudig te doorlopen datastructuur van maakt. Deze datastructuur kan later door een ander programma eenvoudig worden gebruikt, Het programma kan met deze datastructuur makkelijker overweg dan de originele HTML-code.

De parser die voor het prototype gebruikt is, is afkomstig uit het pakket CL-HTTP. Dit is de Common Lisp HTTP server die gebruikt wordt voor het Family Proxy project. De gebruikte HTML-parser maakt van een webpagina een hiërarchische gestructureerde boom. In Figuur 46 is een voorbeeld weergegeven van een HTML-boom met de bijbehorende HTML pagina.



Figuur 46. Een voorbeeld van een HTML-boom

Met behulp van deze HTML-boom is het heel eenvoudig om de tekst die op de wegpagina staat, uit de HTML-code te halen.

Het probleem met de gebruikte HTML-parser is dat deze wel perfect werkt met correcte HTML pagina's, maar wanneer er een fout zit in de HTML-code crasht deze parser. In de praktijk blijkt helaas dat er heel veel webpagina's zijn waarvan de HTML-code fouten bevat. De parser kan daar niet goed tegen en geeft in plaats van een HTML-boom een fout terug. Een tweede probleem met deze HTML-parser is dat deze is gemaakt volgens de officiële W3C standaard. Er zijn vele uitbereidingen op deze standaard, waardoor veel webpagina's niet correct geparsed kunnen worden met deze HTML-parser. Opgemerkt moet worden dat de bekende browsers, zoals Internet Explorer en Netscape Navigator, niet volledig correcte HTML-code wel goed kunnen laten zien. Ook de uitbereidingen die op de W3C standaard zijn gemaakt worden door de bekende browsers wel goed weergegeven.

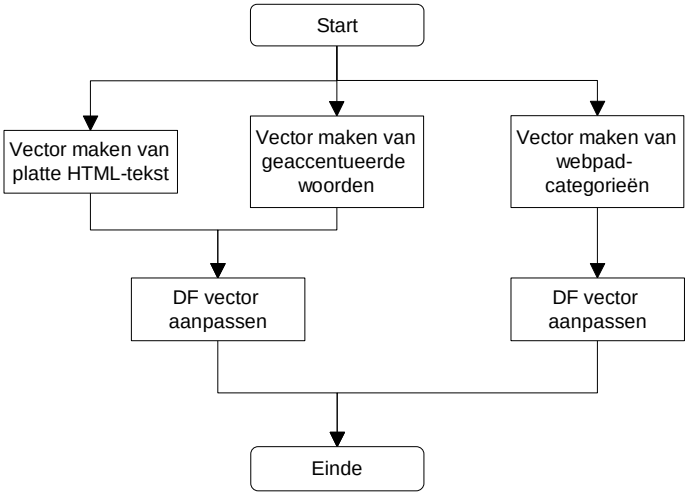
De reden waarom, ondanks de aangegeven problemen, toch is gekozen voor de HTML-parser van CL-HTTP is omdat deze parser op dat moment de enig beschikbare parser was. Het alternatief om er zelf een te bouwen ging niet omdat er de tijd niet voor was om een goede, robuuste parser te schrijven. Tevens was er aangekondigd dat er een verbeterde versie van de parser beschikbaar zou komen. Deze update kwam inderdaad, de parser was beter maar nog steeds niet perfect.
Na het onderzoek kwam er wel een andere goede parser beschikbaar die gebruik maakt van een Internet browser om de informatie van de webpagina af te halen.

```
;;; this function makes a string from a html-parse-tree
(defun remove-html-tags (html-tree)
  (cond
   ((or (typep html-tree 'html-parser:HTML-ENTITY-TOKEN)
        (typep html-tree 'HTML-ENTITY-TOKEN)) "")
   ((or (typep html-tree 'html-parser:html-tag-instance)
        (typep html-tree 'html-tag-instance))
    (let ((string " "))
      (dolist (item (html-parser::parts html-tree) string)
        (setf string (concatenate 'string string (remove-html-tags item))))))
   ((typep html-tree 'STRING)  html-tree)
   (t  " ")
   ))

;;; This function returns a string with all accentuated words fromn a html-page
(defmethod accentuated-text (html-tree)
  (if (eql (type-of html-tree) 'html-tag-instance)
      (cond
       ;if the text is accentuated, return the text
       ((is-title html-tree) (remove-html-tags html-tree))
       ((is-bold html-tree) (remove-html-tags html-tree))
       ((is-italic html-tree) (remove-html-tags html-tree))
       ((is-underlined html-tree) (remove-html-tags html-tree))
       ((is-big html-tree) (remove-html-tags html-tree))
       ((is-strong html-tree) (remove-html-tags html-tree))

       ;parse the rest of the tree
       (t (let ((string ""))
            (dolist (item (html-parser::parts html-tree) string)
              (setf string (concatenate 'string string (accentuated-text item)))))))))
```

Figuur 47. Code om platte tekst en geaccentueerde woorden uit HTML-boom te halen

Nadat de webpagina geparsed is kan een label van de webpagina gemaakt worden. Een label van een webpagina wordt opgebouwd aan de hand van een aantal vectoren. In totaal worden er drie vectoren gebruikt, een vector met de allen de platte HTML-tekst, een vector met de door specifieke HTML-tags geaccentueerde woorden en een vector van de termen die gebruikt zijn als Webpadcategorie voor de betreffende webpagina. Om deze eerste twee vectoren te maken moet de HTML-boom worden doorlopen om de termen uit de boom te kunnen abstraheren. De code die hiervoor gebruikt is, is weergegeven in Figuur 47. De code laat zien dat het doorlopen van een HTML-boom redelijk eenvoudig is.

Met behulp van de drie bovengenoemde vectoren worden de DF-vectoren (Document Frequency) gemaakt. Deze vectoren geven aan in hoeveel webpagina's (of Webpadcategorieën) een term voorkomt. De DF-vectoren zijn noodzakelijk voor uitvoeren van het TFIDF proces. Het maken van een label is grafisch weergegeven in Figuur 48.



Figuur 48. Label maken

De implementatie van het maken van een label is weergegeven in Figuur 49. Deze figuur laat zien dat de code drie vectoren maakt en opslaat in een URL-object. Het maken van een vector op zich komt in Figuur 52uitgebreid aan de orde.

```
;; make-label creates a term-frequency vector from an html-page
;; html-page is a sting
;; All html-tags are removed
(defmethod make-label ((url url-class) html-page)
  (progn
    (let ((html-tree (html-parser::simple-parser html-page)))
      (setf (url-class-tf-vector url)
            (html-filtering::make-document-tf-vector (list (concatenate
'string (html-parser::remove-html-tags html-tree) " ")) "efw"))
      (setf (url-class-tag-vector url)
            (html-filtering::make-document-tf-vector (list (concatenate
'string (html-parser::accentuated-text html-tree) " ")) "efw")))))

  ;read wpc description from testdata
  (let ((description ;read webpadcategories
         (read
          (make-string-input-stream
           (database-interface::postpage
            hostname interface-page
            (format nil "database=labels&query=select
concat(':|',description,'|') from testdata where url='|~A|'"
                    (searchengine::encode-search (url-class-url url))))))))
    (if description
        (setf (url-class-wpc-vector url)
              (html-filtering::make-document-tf-vector
               (mapcar #'(lambda (cons)(concatenate 'string (symbol-name
(car cons)) " ")) description) "efw"))))
  url)
```

Figuur 49. Vectoren van platte tekst, geaccentueerde tekst en de Webpadcategorieën maken

Voor een webpagina wordt een URL-object aangemaakt. Elke vector wordt opgeslagen in dit URL object. De code van het URL object is weergegeven in Figuur 50

```
(defclass url-class ()
;the URL-string
  ((url :initarg :url :accessor url-class-url)
;the vectors...
   (tf-vector       :accessor url-class-tf-vector)
   (tfidf-vector    :accessor url-class-tfidf-vector)
   (tag-vector      :accessor url-class-tag-vector)
   (wpc-vector      :accessor url-class-wpc-vector)
   (label1          :accessor url-class-label1)    ;user-label
   (label2          :accessor url-class-label2)    ;computer-label
   (label3          :accessor url-class-label3)))  ;combi
```

Figuur 50. URL object

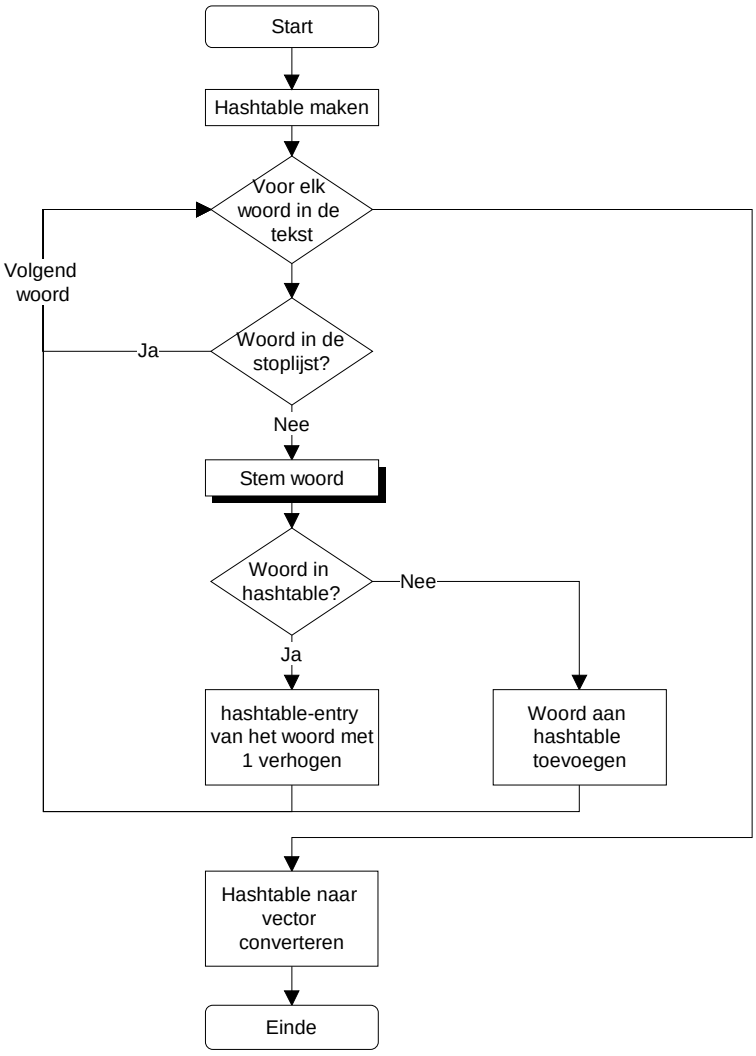
```
(defmethod update-document-frequency-vector ((vec cons))
  ;check if df exists
  (if (not *df-vector*)
      (setf *df-vector* (make-instance 'document-frequency-vector :id '1 )))
  ;update *df-vector*
  (setf (df-vector-vec *df-vector*)
        (vector::map-double-alist #'(lambda (df-vec tf-vec)
                                      (+ df-vec
                                         (if (> tf-vec 0) 1 0)))
                                  (df-vector-vec *df-vector*) vec))
  (incf (df-vector-nr-docs  *df-vector*)))
```

Figuur 51. Implementatie van updaten van DF vector

Voor het maken van een label moet een aantal vectoren worden gemaakt. Een vector wordt gemaakt aan de hand van een stuk tekst. In Figuur 52 is weergegeven hoe een vector gemaakt wordt. Voor het maken van een vector van een tekst als tussenstap een conversie gemaakt van en naar een hashtable. In de hashtabel staat voor elke term het aantal keren dat een woord voorkomt in de tekst. Het voordeel van een hashtable ten

opzichte van een vector is dat er heel snel gezocht kan worden of een bepaald woord al eerder is voorgekomen. Dit is nodig bij het doorlopen en tellen van de woorden in de tekst. Elk woord in de tekst wat nog niet in de hashtable voorkomt wordt toegevoegd. Voor elk woord in de tekst welke al wel in de hashtable staat, wordt alleen de bijbehorende teller opgehoogd.

Elk woord in de tekst wordt eerst vergeleken met een stopwoordenlijst. Deze stopwoordenlijst bevat woorden die geen betekenis aan de inhoud geven. Bijvoorbeeld lidwoorden (bijvoorbeeld ‘de’, ‘het’ en ‘een’) of voegwoorden (bijvoorbeeld ‘en’). De stopwoorden staan ook in een hashtable zodat ook snel gezocht kan worden.
De woorden die niet in de stopwoordenlijst staan worden tot de stam teruggebracht in de functie Stem woord (zie voor details Figuur 54) en toegevoegd aan de hashtable wanneer het woord er nog niet in zit. Wanneer een woord al wel in de hashtable zit, wordt het gewicht van dit woord met 1 opgehoogd. Tot slot wordt deze hashtable omgezet naar een vector voor verdere verwerking door het systeem.



Figuur 52. Vector maken

```
(defmethod make-document-tf-vector ((text list) stem-db)
  (let (;(count-table (make-hash-table :size 2000 :test #'equal))
        (hline "")
         (hword "")
         (tf-vector '()))
    (block parse-loop
      (dolist (line text)
       (nstring-downcase line)
       (multiple-value-setq
            (hline hword) (end-hyphen (concatenate 'string hword line)))
       (let ((words (parse-line hline)))
         (dolist (word words)
            (let ((stemmed-word (stem-word word stem-db)))
             (unless (gethash stemmed-word *stop-word-table*)
               (incf (gethash stemmed-word *count-table* 0))))))))
    (setf tf-vector
          (vector:make-alist-vector *count-table*))
    ;; free the table as soon as possible
    (clrhash *count-table*)
    (values tf-vector)))
```

Figuur 53. Implementatie het maken van een vector

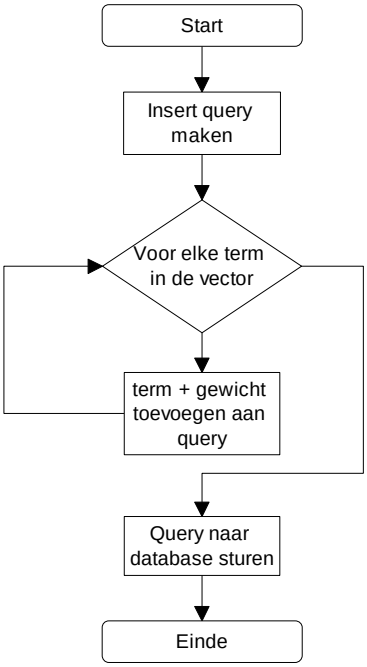
Het stemmen gebeurt aan de hand van de database van CELEX (zie paragraaf 6.3.2). In deze database staat van elk woord het nummer van een categorie waar dit woord bij hoort. Verschillende woorden kunnen hetzelfde categorienummer hebben. Deze woorden zijn, wat betreft de inhoudelijke betekenis, hetzelfde. Verschillende woorden met hetzelfde categorienummer worden bij het stemmen vervangen door de ‘stam’ van het woord. De stam van een woord staat niet expliciet vermeld in de CELEX database. Daarom wordt als stam voor de woorden met hetzelfde categorienummer het eerste woord in de database genomen dat dit categorienummer heeft. Dit woord is niet de stam, maar, wanneer consequent het eerste woord van een categorie als stam gekozen wordt, heeft dit wel hetzelfde effect als stemmen. In Figuur 54 is het proces van het stemmen van een woord, zoals dit gedaan wordt in het prototype, weergegeven.

Figuur 54. Stem woord (het stemmen van een woord)

```
(defmethod stem-word (word stem-db)
 (if stem-db
   (let
    ((catnr (read (make-string-input-stream
      (database-interface::postpage
        label-database::hostname
        label-database::interface-page
        (format nil "database=stem&query=select catnr from ~A where word='~A' limit 1;"
           stem-db word))))))
        (if catnr
          (string-downcase (format nil "~A" (first (first (read (make-string-input-stream
            (database-interface::postpage
             label-database::hostname label-database::interface-page
             (format nil "database=stem&query=select word from ~A where catnr=~A limit 1;"
               stem-db (first (first catnr))))))))))
    word))
  word))
```

Figuur 55. Implementatie van het stemmen van een woord

Tot slot moet het label opgeslagen worden. Dit opslaan wordt gedaan door een SQL statement naar de database te sturen. Hoe dit proces in zijn werk gaat is weergegeven in Figuur 56. De implementatie van het opslaan van een label in de database is weergegeven in Figuur 57.

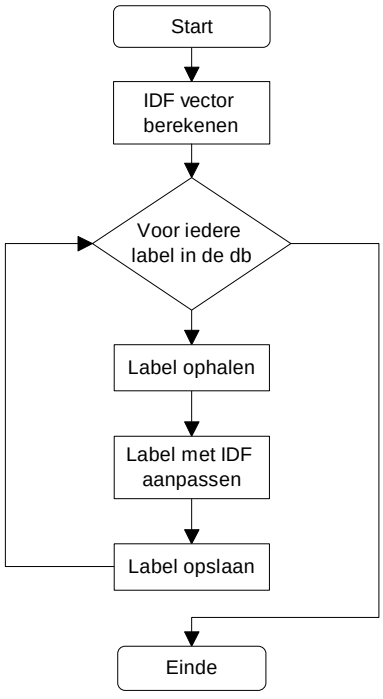


Figuur 56. Label opslaan in database

```
(defmethod store-label ((url url-class))
  (let ((sep #\Space)
        (query "database=labels&query=insert into label values"))
    (dotimes (itemnr (length (url-class-tf-vector url))) query
      (setf query
            (concatenate 'string query
              (format nil "~c('~A','|~A|',~A,~A,~A,~A,~A,~A,~A)"
                          sep
                          (car (nth itemnr (url-class-tf-vector url)))        ;term
                          (searchengine::encode-search (url-class-url url))   ;url
                          (if (slot-boundp url 'label1)
                            (cdr (nth itemnr (url-class-label1 url)))         ;label1
                            0)
                          (if (slot-boundp url 'label2)
                            (cdr (nth itemnr (url-class-label2 url)))         ;label2
                            0)
                          (if (slot-boundp url 'tf-vector)
                            (cdr (nth itemnr (url-class-tf-vector url)))      ;tf
                            0)
                          (if (slot-boundp url 'tfidf-vector)
                            (cdr (nth itemnr (url-class-tfidf-vector url)))   ;tfidf
                            0)
                          (if (slot-boundp url 'tag-vector)
                            (if (cdr (nth itemnr (url-class-tag-vector url))) ;tag
                                 (cdr (nth itemnr (url-class-tag-vector url)))
                                0)
                            0)
                          (if (slot-boundp url 'wpc-vector)
                            (cdr (nth itemnr (url-class-wpc-vector url)))     ;wpc
                            0)
                          (if (slot-boundp url 'label3)
                            (cdr (nth itemnr (url-class-label3 url)))         ;label3
                            0))))
      (setq sep #\,))
    (database-interface::postpage hostname interface-page query )))
```

Figuur 57. Implementatie van het opslaan van een label

Wanneer de labels van alle webpagina's zijn bepaald, kan TFIDF worden toegepast. De TFIDF normalisatie bestaat uit het vermenigvuldigen van de vectoren waaruit een label is opgebouwd met een zogenaamde IDF vector. De IDF vector wordt berekend uit de DF vector welke tijdens het labelproces incrementeel is bepaald. Het berekenen van de IDF vector gaat volgens de formule beschreven in 6.3.4. In Figuur 58 is het TFIDF proces weergegeven.



Figuur 58. TFIDF

De implementatie van de verschillende functies uit het TFIDF proces Figuur 58 zijn in de volgende drie figuren weergegeven. Het proces ‘Label met IDF vector aanpassen’ is opgenomen in het maken van de labels van de vectoren in Figuur 61.

```
(defmethod calculate-idf-vector ()
  ;log ( |Docs| / W)
  ;IDF = '((log |D|/W)(log |D|/W) ... (log |D|/W))
  (let ((docs (df-vector-nr-docs *df-vector*)))
    (setf (df-vector-idf *df-vector*)
          (mapcar #'(lambda (df) (cons (car df) (log (/ docs (cdr df)))))
                  (df-vector-vec *df-vector*)))))
```

Figuur 59. Implementatie van het berekenen van een IDF vector

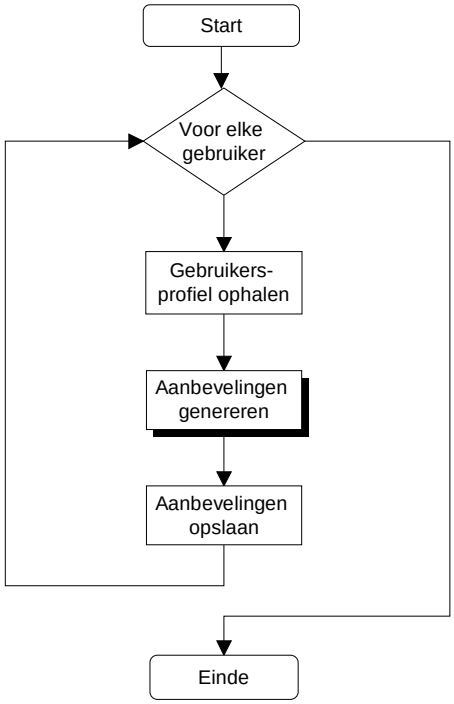
```
;;; This function returns a url-class with label.
;;;If the url is not in the database, the function returns nil
(defmethod load-label ( url-string)
  (let* ((label (make-instance 'url-class :url=url-string))
         (query (format nil "select
concat(':|',lcase(term),'|'),label1,label2,tf,tfidf,tag,wpc,label3 from label where url='|~A|'
order by term asc" (searchengine::encode-search url-string)))
         (dbresult (read (make-string-input-stream
                          (database-interface::send-sql-query "labels" query)))))
    (if dbresult
        (progn
          (setf (url-class-url label) url-string)
          (setf (url-class-label1       label)
             (list (cons (first (first dbresult))(second  (first dbresult)))))
          (setf (url-class-label2       label)
             (list (cons (first (first dbresult))(third   (first dbresult)))))
          (setf (url-class-tf-vector    label)
             (list (cons (first (first dbresult))(fourth  (first dbresult)))))
          (setf (url-class-tfidf-vector label)
             (list (cons (first (first dbresult))(fifth   (first dbresult)))))
          (setf (url-class-tag-vector   label)
             (list (cons (first (first dbresult))(sixth   (first dbresult)))))
          (setf (url-class-wpc-vector   label)
             (list (cons (first (first dbresult))(seventh (first dbresult)))))
          (setf (url-class-label3       label)
             (list (cons (first (first dbresult))(eighth  (first dbresult)))))
          (dolist (item (cdr dbresult))
            (setf (cdr (last (url-class-label1       label)))
              (list (cons (first item)(second  item))))
            (setf (cdr (last (url-class-label2       label)))
              (list (cons (first item)(third   item))))
            (setf (cdr (last (url-class-tf-vector    label)))
              (list (cons (first item)(fourth  item))))
            (setf (cdr (last (url-class-tfidf-vector label)))
              (list (cons (first item)(fifth   item))))
            (setf (cdr (last (url-class-tag-vector   label)))
              (list (cons (first item)(sixth   item))))
            (setf (cdr (last (url-class-wpc-vector   label)))
              (list (cons (first item)(seventh item))))
            (setf (cdr (last (url-class-label3       label)))
              (list (cons (first item)(eighth  item))))
            ) label)
      nil)))
```

Figuur 60. Code voor het ophalen van een label

```
;; this function calculates label1, label2 and label3 for a specific url
(defmethod calculate-label (url-string)
  ;load label url-string
  (let ((label (load-label url-string)))
    (if label
      ;calculate tfidf and normalise
      (progn
        (setf (url-class-tfidf-vector label)
              ;subtract and add needed to correct the lenth of the tfidfvector
              (vector:subtract-vector
               (vector:add-vector      ;length(tf)=length(tfidf)
                (vector:normalise-vector
                 (vector:multiply-vector (url-class-tf-vector label)
                                         (df-vector-idf *df-vector*)
                                         0.0))
                (url-class-tf-vector label))
               (url-class-tf-vector label)))
        ;create and normalise label
        ;label1 = norm ( 1 * norm (wpc) + 0 * (norm ( norm (tag) + norm (tfidf))))
        (setf (url-class-label1 label)
              (vector:normalise-vector
               (vector:add-vector
                 (vector:multiply-vector
                  (vector:subtract-vector
                   (vector:add-vector
                    (vector:normalise-vector
                     (vector:multiply-vector (url-class-wpc-vector label)
                                             (df-vector-idf *wpc-df-vector*)
                                             0.0))
                    (url-class-wpc-vector label))
                   (url-class-wpc-vector label)) 5)
                 (vector:multiply-vector
                  (vector:normalise-vector
                   (vector:add-vector
                    (vector:normalise-vector (url-class-tag-vector label))
                    (vector:normalise-vector (url-class-tfidf-vector label)))) 0))))
        ;label2 = norm ( 0 * norm (wpc) + 1 * (norm ( norm (tag) + norm (tfidf))))
        (setf (url-class-label2 label)
              (vector:normalise-vector
               (vector:add-vector
                 (vector:multiply-vector
                  (vector:normalise-vector (url-class-wpc-vector label)) 0)
                 (vector:multiply-vector
                  (vector:normalise-vector
                   (vector:add-vector
                    (vector:normalise-vector (url-class-tag-vector label))
                    (vector:normalise-vector (url-class-tfidf-vector label)))) 1))))
        ;label3 = norm ( 1 * norm (wpc) + 1 * (norm ( norm (tag) + norm (tfidf))))
        (setf (url-class-label3 label)
              (vector:normalise-vector
               (vector:add-vector
                 (vector:multiply-vector
                  (vector:subtract-vector
                   (vector:add-vector
                    (vector:normalise-vector
                     (vector:multiply-vector (url-class-wpc-vector label)
                                             (df-vector-idf *wpc-df-vector*)
                                             0.0))
                    (url-class-wpc-vector label))
                   (url-class-wpc-vector label)) 1)
                 (vector:multiply-vector
                  (vector:normalise-vector
                   (vector:add-vector
                    (vector:normalise-vector (url-class-tag-vector label))
                    (vector:normalise-vector (url-class-tfidf-vector label)))) 1))))
        ;store url-class
        (update-label label))
```

10.2.2 Figuur 61. Label van de vectoren maken Aanbevelingenproces

Het aanbevelingenproces bestaat uit drie processen. Het eerst proces bestaat uit het ophalen van het gebruikersprofiel. Daarna volgt het genereren van de aanbevelingen op basis van het gebruikersprofiel. Tot slot worden deze aanbevelingen opgeslagen in een aanbevelingen database. Het ophalen van het gebruikersprofiel bestaat uit het inlezen van de bezochte URL's met de bijbehorende Webpadcategorieën uit de userdatabase. In Figuur 62 is het aanbevelingenproces grafisch weergegeven.



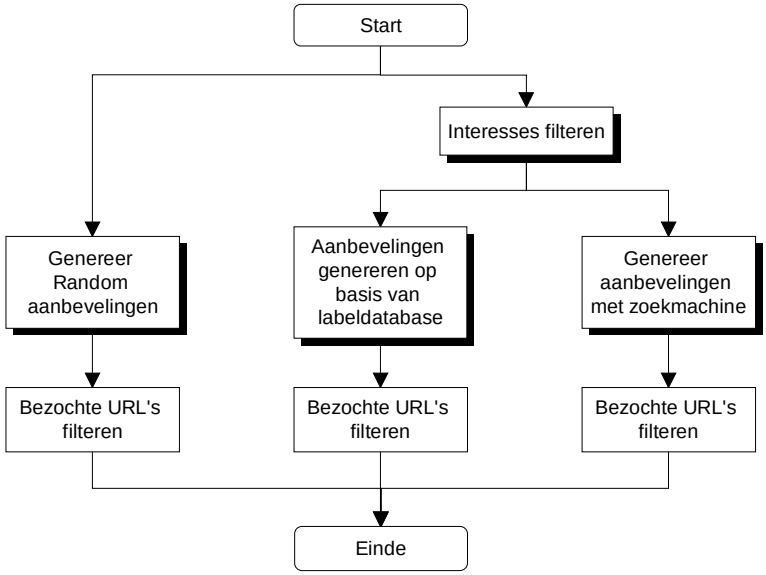
Figuur 62. Aanbevelingenproces

De implementatie van het Aanbevelingenproces volgens bovenstaande figuur is weergegeven in Figuur 63.

```
;;; generate recommendations for each user and store these in the database
(defmethod generate-recommendation-database ((nr-of-recs number))
  ;; do for all users
  (dolist
    (email
     (read (make-string-input-stream
            (database-interface::send-sql-query "labels"
             (format nil "select distinct email from testdata "))))
    (generate-recommendations-user (symbol-name (car email)) nr-of-recs)))

;;;generate recommendations for user
(defmethod generate-recommendations-user ((email string)(nr-of-recs number))
  (let ((terms
         (read (make-string-input-stream
                (database-interface::send-sql-query "labels"
                 (format nil
                    "select concat(':|',description,' |') from testdata where email='~A'"
                         email)))))
        (recs nr-of-recs)
        (visited-urls (read (make-string-input-stream
         (database-interface::send-sql-query "labels"
          (format nil "select concat(':', url) from testdata where email='~A'" email))))))
    (store-recommendations
     (mapcar #'car (make-recommendationlist-1 terms recs visited-urls)) email 1)
    (store-recommendations
     (mapcar #'car (make-recommendationlist-2 terms recs visited-urls)) email 2)
    (store-recommendations
     (mapcar #'car (make-recommendationlist-3 terms recs visited-urls)) email 3)
    (store-recommendations
     (mapcar #'car (make-recommendationlist-4 terms recs visited-urls)) email 4)
    (store-recommendations
     (mapcar #'car (make-recommendationlist-5 terms recs visited-urls)) email 5)))

(defmethod store-recommendations ((rec-list list)(email string)(rec-method number))
  (let ((query (format nil "database=labels&query=insert into recommendations values"))
        (sep #\Space))
    (dolist (item rec-list)
      (setf query (concatenate 'string query (format nil "~c('~A','~A',~A,~A)" sep email
item rec-method 'NULL)))
      (setf sep #\,))
    (database-interface::postpage hostname interface-page
                                  (format nil "database=labels&query ~A" query))))
```

Figuur 63. Implementatie van Aanbevelingenproces

Het genereren van aanbevelingen wordt gedaan met behulp van een drietal verschillende methoden.



Figuur 64. Aanbevelingen genereren

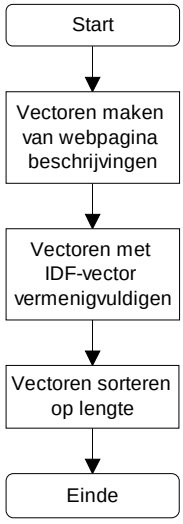
De random aanbevelingen worden gegenereerd door een lijst te maken van alle URL's die door de gebruikers zijn bezocht en hier de door de betreffende gebruiker bezochte URL's vanaf te halen. Uit deze lijst kunnen dan willekeurig een aantal URL's getrokken worden en als aanbeveling aan de gebruiker worden gedaan. In Figuur 65 is weergegeven hoe het genereren van Random aanbevelingen is geïmplementeerd.

```
;=============Random generator==============

;;; generate recommendations with randomizer
(defmethod make-recommendationlist-1 ((terms list)(rec-length number)
                                      (visited-urls list))
  (let ((all-urls (remove-list-from-list visited-urls
         (label-database::show-database))))
           (make-random-recommendationlist all-urls rec-length)))

;;; pick at random rec-length items from the urllist
;;; return the list
(defmethod make-random-recommendationlist ((urllist list) (rec-length number))
  (let ((url-list (remove-duplicates urllist))) ;remove duplicates
    ;if more recommendations needed than url's given return the whole list
    (if (>= rec-length (length url-list))
        url-list
      (let* ((length (length url-list))
             (rec-list (list (nth (random length) url-list))))
        (do ((item (nth (random length) url-list)
               (nth (random length) url-list)))      ;select element
            ((>= (length rec-list) rec-length))      ;until length reached
          (if (not (find item rec-list))
            (setf (cdr (last rec-list)) (list item)))) ;if not in list add
        rec-list))))
```

Figuur 65. Implementatie van genereren van Random aanbevelingen

De interesses van de gebruikers worden afgeleid uit de termen die ze opgeven bij de verschillende bookmarks of favorieten. Deze termen kunnen vaak vage termen zoals bijvoorbeeld 'handig' bevatten. Deze termen zijn voor de gebruiker wel nuttig, maar om op basis van 'handig' aanbevelingen te genereren is niet handig. Het filteren van de interesses is nodig om gerichter aanbevelingen te kunnen doen. Door algemene termen een lager gewicht te geven, kunnen termen als 'handig' worden weggefilterd. Dit is bereikt worden door TFIDF toe te passen op de termen die de gebruikers opgeven. In de

onderstaande figuur is het proces weergegeven dat de interesses van een gebruiker filtert.



Figuur 66. Interesses filteren

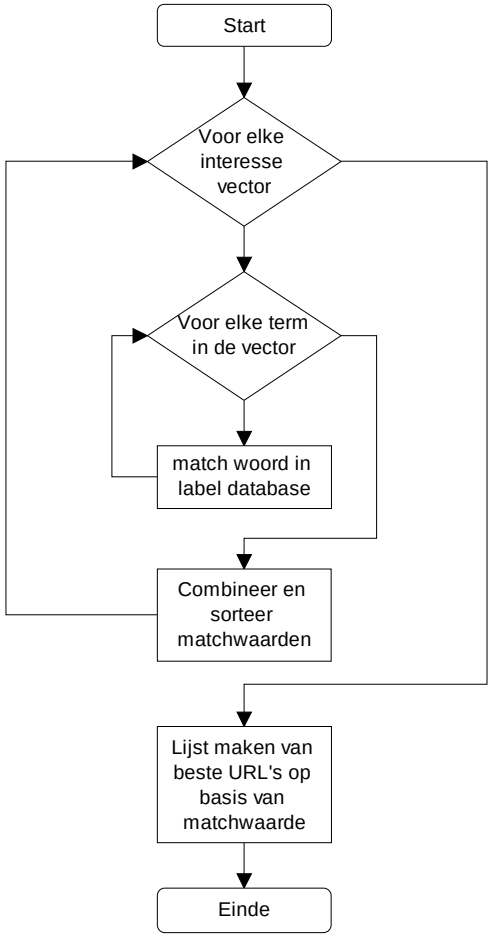
De interesses van de gebruiker zijn nodig voor het genereren van aanbevelingen met de labeldatabase. De interesse van een gebruiker bestaat uit een vector waarbij elk element bestaat uit een aantal termen die de gebruiker bij een bepaalde URL heeft opgegeven. In de onderstaande code worden de interesses van de gebruiker aan de hand van de Webpadcategorieën opgesteld. De interesses van een gebruiker worden met behulp van TFIDF genormaliseerd om zo veel mogelijk specifieke termen naar boven te halen. Van de interesses wordt een vector gemaakt die wordt gesorteerd, zodat de termen die voor de betreffende gebruiker het zwaarst wegen als eerste in de vector staan. De volgorde is belangrijk voor het genereren van aanbevelingen. Het genereren van aanbevelingen zal beginnen met de interesse met de hoogste waarde, gevolgd door de interesse met de daaropvolgende waarde, enzovoorts. Wanneer er dus in totaal 5 aanbevelingen worden gegenereerd voor een gebruiker, zullen er op basis van de eerste 5 interesses van de gebruiker aanbevelingen worden gegenereerd.

```
;;;returns a sorted list of lists of terms with weights.
;;;words that occur a lot (in terms of tfidf) are at the bottom of the list
(defmethod sort-search-terms ((list-of-termlists list))
 (let ((lst '()))
   (dolist (item list-of-termlists)
     (let
      ((temp
        (vector:multiply-vector
         (html-filtering::make-document-tf-vector
            (list (symbol-name (car item))) nil)
         (label-database::df-vector-idf label-database::*wpc-df-vector*) 0.0)))
      (if temp
       (progn
        (setf temp
         (cons
          (sqrt
           (reduce #'+
            (mapcar #'cdr (vector:multiply-vector temp temp)))) temp))
        (setf lst (cons temp lst))))))
  (mapcar #'cdr (sort (reverse lst) #'> :key #'car))))
```

Figuur 67. Implementatie van het filteren van de interesses

Het genereren van aanbevelingen met behulp van de labeldatabase is weergegeven in Figuur 68. Voor elke term in een interesse-element wordt in de database gezocht naar overeenkomstige webpagina's. Als een webpagina op meer dan één woord matcht, worden de matchwaarden bij elkaar opgeteld. Er wordt hiervan een lijst van URL's gemaakt, zodanig dat wanneer de beste X pagina's als aanbeveling teruggegeven

worden, er webpagina's over X interesses worden gekozen, mits X kleiner is dan de grootte van de interessevector van de gebruiker.



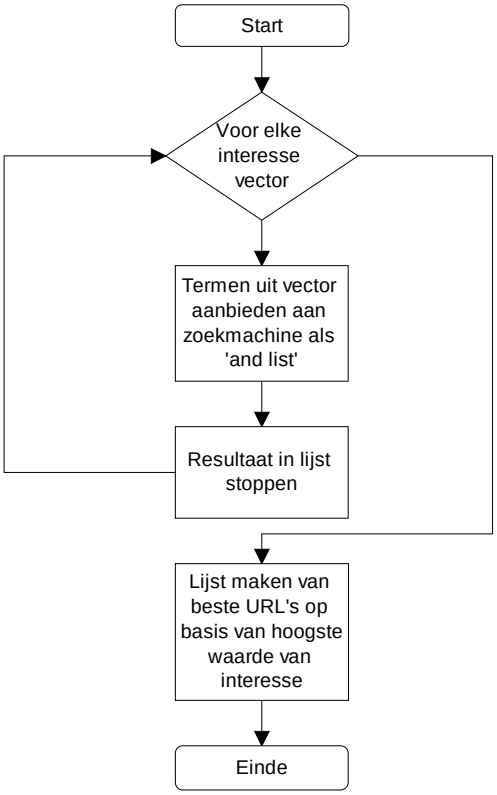
Figuur 68. Aanbevelingen genereren met labeldatabase

De functies die nodig zijn voor het uitvoeren van het genereren van aanbevelingen met behulp van de labeldatabase zijn opgenomen in Figuur 69.

```
;================Recommendations with labels==================

;;; generate recommendations with label1
(defmethod make-recommendationlist-2 ((list-of-termlists list)(rec-length number)
                                      (visited-urls list))
 (let*
  ((list-of-url-lists        ;;make a list of url-list with term-lists
    (make-list-of-url-lists
     (sort-search-terms list-of-termlists) "label1" visited-urls));label1
   (rec-list '())
    ;the length of the longest list
   (max-length (reduce #'max (mapcar #'length list-of-url-lists))))
  (do ((index 0 (1+ index)))
      ((> index max-length))
    (dolist (url-list list-of-url-lists)
      (if (and
           (< index (length url-list))   ;there is an element in the list
           (not (find (nth index url-list) visited-urls))
            ;element is in rec-list
           (not (find (nth index url-list) rec-list :test #'equal )))
            (setf rec-list (cons (nth index url-list) rec-list)))))
    (if (> rec-length (length rec-list))
        (sort rec-list #'> :key #'cdr)       ;return the whole list
      (subseq (reverse rec-list) 0 rec-length))))

;;; term is dottedlist of 1 term (symbol) and its weigth
(defmethod match-term ((term list)(label string))
  (sort (vector:multiply-vector
         (read (make-string-input-stream
            (database-interface::send-sql-query "labels"
             (format nil
                 "select concat(':',url),' . ',~A from label where term='~A' and ~A>0.0"
                   label (symbol-name (car term)) label))))
         (cdr term))
        #'> :key #'cdr))

;;vectormatch
;;terms is a list of multiple terms
(defmethod find-urls ((terms list)(label string))
  (let ((result-vec '()))
    (dolist (item terms)
      (setf result-vec
            (vector:add-vector
             result-vec
             (match-term
              (cons (car (car (html-filtering::make-document-tf-vector
                       (list (concatenate 'string (symbol-name (car item)) " ")) "efw")))
                    (cdr item)) label))))
    (sort result-vec #'> :key #'cdr)))
```

Figuur 69. Implementatie van aanbevelingen met labels

Het genereren van aanbevelingen met behulp van een zoekmachine is iets eenvoudiger. Het proces is weergegeven in Figuur 70 en de implementatie in Figuur 71.
Bij het genereren van aanbevelingen met een zoekmachine wordt net zoals bij het genereren van aanbevelingen op basis van de labeldatabase gebruik gemaakt van het interesseprofiel van een gebruiker. De termen die in één element staan worden als ‘and-list’ aan een zoekmachine gegeven. Uit de resultaten die de zoekmachine teruggeeft worden weer webpagina’s gekozen die als aanbeveling worden teruggegeven. Ook hier wordt weer rekening gehouden dat de aanbevelingen gelijkmatig worden verdeeld over de interesses zodat het niet kan voorkomen dat een gebruiker aanbevelingen krijgt over één onderwerp (tenzij zijn of haar profiel maar uit één onderwerp bestaat).



Figuur 70. Aanbevelingen genereren met zoekmachine

In de onderstaande figuur is de implementatie van het aanbevelingenalgoritme weergegeven.

```
;====================Recommendations with searchengine(s)=======================

;;; generate recommendations with altavista
(defmethod make-recommendationlist-5 ((terms list)(rec-length number)(visited-urls list))
 (common-prolog:with-prolog (searchengine::forget-urls-keywords ?))
 (let ((orandlist (make-or-and-list terms)))
  (dolist (item orandlist)
    (searchengine::search-site item default-searchengine :all :values))
  (let
   ((rec-list
     (remove-list-from-list
      visited-urls
      (mapcar #'(lambda (str) (list (intern str :keyword)))
       (searchengine::generate-urls orandlist (list default-searchengine :all :list))))))
      (if (<= (length rec-list) rec-length)
          rec-list
        (subseq rec-list 0 rec-length)))))
```

Figuur 71. Implementatie van aanbevelingen met zoekmachine genereren

10.3 Database model

De Label Server moet een grote hoeveelheid aan informatie opslaan, waaronder de labels van een groot aantal URL's. Om deze informatie op te slaan is gebruik gemaakt van een database. Het voordeel hiervan is dat niet alle informatie in het geheugen hoeft te staan om er snel in te kunnen zoeken. Dit is met name belangrijk omdat de labels van grote hoeveelheden pagina's opgeslagen moeten worden en er regelmatig gezocht moet worden naar labels van een bepaalde webpagina.

Voor de database is gekeken naar twee verschillende types. De eerste database waar naar gekeken is, is PLOB. PLOB staat voor Persistent Lisp Objects. Met PLOB kunnen

variabelen in een Lisp programmeeromgeving persistent gemaakt worden. Elke persistent variabele die aangemaakt wordt, wordt ook op disk vastgelegd. Met behulp van een indexeersysteem kan eenvoudig gezocht worden in de verzameling objecten. PLOB is geïntegreerd in LISP en is daardoor heel eenvoudig te gebruiken.
Het probleem met PLOB is echter dat dit systeem niet bedoeld is om grote hoeveelheden data persistent te maken. Voor het opslaan van grote hoeveelheden labels is PLOB dus niet geschikt. Het zoeken is vrij traag en de hoeveelheid werkgeheugen dat gebruikt wordt, is ondoenlijk groot.

Daarom is gekeken naar een andere manier om de labels van de webpagina's op te slaan. Met MySQL kunnen wel grote hoeveelheden data efficiënt opgeslagen worden. Het probleem met MySQL is dat er geen standaard interface is om vanuit Lisp een MySQL database aan te spreken. Om dit probleem te omzeilen, is er gebruik gemaakt van een PHP interface. PHP is een scripttaal die standaard functionaliteit biedt om een MySQL database aan te spreken. Lisp kan met PHP communiceren door deze PHP scripts aan te roepen.

Het PHP script dat gebruikt is om met de SQL database te communiceren is afgebeeld in Figuur 72. Hierin wordt eerst een connectie gemaakt met de database. Wanneer dit lukt wordt de query, die als argument bij het ophalen van de webpagina wordt meegegeven, uitgevoerd op de database. Met het query-argument kunnen willekeurige SQL queries naar de database gestuurd worden. Bijvoorbeeld een 'insert' statement waarbij de database alleen aangeeft of de query gelukt is of niet. Het PHP script geeft in dit geval een 1 terug als indicatie dat de query gelukt is. Een ander voorbeeld is een 'select' statement waarbij de database de geselecteerde data teruggeeft. In dit geval formatteert het script de data zodanig dat deze door een LISP programma eenvoudig kan worden gebruikt.

```
<?php
  $link = mysql_pconnect ("localhost", "wouter", "cl-http")
         or die ("Could not connect");

  mysql_select_db ($database)
         or die ("Could not select database");
  parse_str($QUERY_STRING);

  $query2= stripslashes($query);
  $result = mysql_query ($query2) or die ("Query failed");
  if ($result > 1){
     echo "(";
     $numfields = mysql_num_fields ($result);
     while ($row = mysql_fetch_array ($result)) {
       echo "(";
       for ($index = 0; $index < $numfields; $index ++) {
          echo $row[$index]."\t";
       }
       echo ")";
     }
     echo ")";
     mysql_free_result ($result);
  }
  else
     echo"1";
?>
```

Figuur 72. PHP interfacepagina met SQL database

De functie om vanuit LISP een SQL database te benaderen is weergegeven in Figuur 73. De functie heeft als argumenten de querystring en de naam van de database. De locatie van de database wordt uit een globaal gedefinieerde variabele gehaald. De output bestaat uit een string. De 'send-sql-query' maakt gebruik van de functie 'post-page'. Deze functie haalt een webpagina op waarbij input data wordt meegegeven. In dit geval wordt de query en de database als input meegegeven. De implementatie van de functie 'post-page' staat in Figuur 74.

```
;; lobal variables
(defvar hostname "http://pc4413.research.kpn.com")
(defvar interface-page "/~wouter/sql_query.php3")

;; send a sql query to the database
(defmethod send-sql-query ((query string)(database string))
  (postpage hostname interface-page
    (format nil "database=~A&query=~A" database query)))
```

Figuur 73. Interface om SQL queries op een database te doen

```
;;; do a postrequest to a http server
(defun postpage (hostname data postdata &key (port 80)
                 (proxy-hostname nil) (body-func #'identity))
  (let ((header (make-array 1 :element-type 'character :adjustable t :fill-pointer 0))
        (body (make-array 1 :element-type 'character :adjustable t :fill-pointer 0))
        (do-header t)
        (encoded-postdata postdata))
    (with-open-stream (http (comm:open-tcp-stream
                             hostname port :timeout 30))
      (sendline http "POST ~@[http://~a~]~a HTTP/1.0" proxy-hostname data)
      (sendline http "Content-type: application/x-www-form-urlencoded")
      (sendline http "Content-length: ~A" (length encoded-postdata))
      (sendline http "")
      (sendline http "~a" encoded-postdata)
      (when proxy-hostname
        (sendline http "Host: ~a" proxy-hostname))
      (sendline http "")
      (force-output http)

      (when *progress-indication*
        (write-string "Waiting for reply...")
        (loop for ch = (read-char-no-hang http nil :eof)
              until ch
              do (write-char #\.)
              (sleep 0.25)
              finally (unless (eq ch :eof)
                        (unread-char ch http)))
        (terpri))

      (loop for line = (remove-returns (read-line http nil nil))
            while line
            do (if do-header
                   (if (= (length line) 0)
                       (setq do-header nil)
                     (format header "~a~%" line))
                 (format body "~a~%" line))))
    (funcall body-func body)))
```

Figuur 74. Implementatie van de functie ‘post-page’

Tot slot volgen nog een tweetal voorbeelden die het gebruik van de PHP interface illustreren. In Figuur 75 is de functie weergegeven die nagaat of een webpagina met een bepaalde URL al in de database zit.

```
;;; this function returns t if the url is already in the database
;;; else it retruns nil
(defmethod is-url-in-database (url-string)
  (read (make-string-input-stream
         (database-interface::send-sql-query "labels"
          (format nil "select term from label where url='|~A|'"
                  (searchengine::encode-search url-string))))))
```

Figuur 75. Voorbeeld gebruik database interface

In Figuur 76 is een voorbeeld output gegeven die wordt getoond wanneer een label wordt opgevraagd uit de database. De code die gebruikt wordt om dit label op te vragen is eerder in Figuur 60 weergegeven. De data die wordt teruggeven door de functie ‘send-sql-query’ wordt door haakjes bij elkaar gehouden en door tabs gescheiden

```
((term1   1    1    1    1    1    1    1)
(term2    2    2    2    2    2    2    2)
(term3    1    1    1    1    1    1    1))
```

Figuur 76. Voorbeeld output

De volgende gegevens moeten worden opgeslagen in de database:

- URL labels
- Stemmingslijsten(en)
- Gebruikersdata
- Aanbevelingen
- Scores van aanbevelingen

Nr.	Naam	Ver-plicht?	Type	Lengte	Beschrijving
1 🔑	Term	J	String	30	Dit is een woord uit een webpagina
2 🔑	URL	J	String	256	De naam van de locatie van de gelabelde webpagina
3	Label1	N	Real		Gewicht van label op basis van tekst genormaliseerd met TFIDF
4	Label2	N	Real		Gewicht van label op basis van Webpad gewicht, genormaliseerd met TFIDF
5	Label3	N	Real		Gewicht van label, is en combinatie van label1 en label2
6	TFIDF gewicht	N	Real		Met TFIDF genormaliseerd TF gewicht
7	TF gewicht	N	Int		Geeft aan hoe vaak het woord op deze webpagina staat
8	Tag gewicht	N	Int		Dit gewicht geeft aan of het woord een bepaalde accentuering heeft
9	Webpad gewicht	N	Int		Geeft aan hoe vaak de term voor deze webpagina door gebruikers is opgegeven

Figuur 77. Tabel Label

Nr.	Naam	Ver-plicht?	Type	Lengte	Beschrijving
1 🔑	Term	J	String	30	Een woord
2	Categorie nummer	J	Int		De groep waar deze term bij hoort

Figuur 78. Tabel Stemming

Nr.	Naam	Ver-plicht?	Type	Lengte	Beschrijving
1 🔑	e-mail	J	String	256	Het e-mail adres van een gebruiker
2	URL	N	String	256	De naam van de locatie van de favoriete webpagina
3	Beschrijving	N	String	256	De beschrijving van de webpagina die de gebruiker heeft opgegeven

Figuur 79. Tabel Testdata

Nr.	Naam	Ver-plicht?	Type	Lengte	Beschrijving
1 🔑	e-mail	J	String	256	Het e-mail adres van een gebruiker
2 🔑	URL	J	String	256	De naam van de locatie van de aanbevolen webpagina
3	Set	J	Int		Het nummer van de set van aanbevelingen

Figuur 80. Tabel Recommendations

Nr.	Naam	Ver-plicht?	Type	Lengte	Beschrijving
1 🔑	e-mail	J	String	256	Het e-mail adres van een gebruiker
2 🔑	Set	J	Int		Het nummer van de set van aanbevelingen
3	Score	J	Int		De score die de gebruiker gegeven heeft aan deze set van aanbevelingen
4	Known	J	Int		Aantal URL's die gebruiker al kende in deze set van aanbevelingen

Figuur 81. Tabel Scores

11 Gebruikersonderzoek

Het prototype, zoals beschreven in de vorige twee hoofdstukken, labelt automatisch webpagina's en genereert aanbevelingen van webpagina's voor gebruikers. Het aanbevelen van webpagina's is gebaseerd op filteren op basis van content, zoals besproken in paragraaf 5.3. Deze methode is erop gebaseerd dat op basis van de inhoud van webpagina's en de interesses van een gebruiker, aanbevelingen gegenereerd worden. Om de kwaliteit van deze aanbevelingen te meten is een gebruikersonderzoek gedaan.

11.1 Het onderzoek

Het onderzoek is uitgevoerd onder medewerkers binnen KPN Research. Naar alle medewerkers van KPN Research is een e-mail gestuurd met de vraag om mee te werken aan dit gebruikersonderzoek. Deze e-mail is in Bijlage A opgenomen. In de e-mail werd gevraagd om een formulier op een webpagina in te vullen. Op deze webpagina moesten de gebruikers een aantal URL's invullen die zij als favoriet bestempelen. Deze webpagina's zijn pagina's die zij vaak bezoeken en die zij dus ook hoogstwaarschijnlijk in hun *bookmarks* of *favorieten* hebben staan. Dit simuleert de *navigationtree* van de Webpad.

Voor elke URL werd aan de gebruiker gevraagd ook een aantal steekwoorden te geven die deze pagina beschrijven. Het formulier is te vinden in Bijlage B. Met de termen geeft de gebruiker aan waar een bepaalde webpagina over gaat. Deze termen worden ten eerste gebruikt om de pagina te labelen en ten tweede om de interesses van de gebruiker af te leiden.

In totaal zijn er vijf verschillende methoden gebruikt om aanbevelingen te genereren. De eerste methode is een randomgenerator. Uit de set van alle door de gebruikers opgegeven URL's zijn er vijf URL's uitgekozen en als aanbeveling aan de gebruiker gedaan. Deze random aanbevelingen dienen als ijkpunt voor de overige aanbevelingen. Wanneer de random aanbevelingen hoger scoren dan de andere aanbevelingen, kan geconcludeerd worden dat de andere, complexere aanbevelingenmethoden niet goed functioneren.

De tweede aanbevelingenmethode is gedaan met behulp van de steekwoorden die de testpersonen hebben opgegeven voor hun favoriete webpagina's. Elke webpagina heeft een label gekregen op basis van de termen die de testpersonen er aan gegeven hebben. Bij het genereren van de aanbevelingen zijn de gebruikersprofielen vergeleken met deze labels.

De derde methode is op basis van de inhoud van de webpagina. Van de woorden op de webpagina is een label gemaakt waarbij rekening is gehouden met de speciale opmaak van de tekst zoals 'bold', 'title' enz. De termen uit het gebruikersprofiel worden gematched met deze labels.

Bij de derde methode worden de labels van de twee voorgaande methoden gecombineerd tot één label. Het gewicht van de termen die de gebruikers aan de verschillende pagina's hebben toegekend, tellen even zwaar mee voor het label als de woorden die op de webpagina zelf voorkomen. Ook hier is weer het gebruikersprofiel vergeleken met het label.

Voor de vijfde aanbevelingenmethode is gebruik gemaakt van de zoekmachine AltaVista. De termen uit het gebruikersprofiel worden aan de zoekmachine aangeboden. De webpagina's met de hoogste score die door de zoekmachine worden teruggegeven, worden als aanbeveling gedaan aan de gebruiker.

Om te meten of de aanbevelingen goed zijn, is er aan de mensen die meegedaan hebben aan het onderzoek gevraagd om de aanbevelingen een cijfer van 1 tot 10 te geven. Elke testpersoon heeft een e-mail gekregen (zie Bijlage C) met daarin een link naar een webpagina waar 5 sets met elk 5 aanbevelingen voor de testpersoon op staan. De testpersoon moest aan elke set van aanbevelingen een cijfer geven. Door de testpersonen een cijfer te laten geven voor een set van aanbevelingen, kan eenvoudig de kwaliteit van een aanbevelingenmethode worden gewaardeerd. Het nadeel hiervan is dat niet nagegaan kan worden hoe goed de afzonderlijke webpagina's zijn als aanbeveling voor een gebruiker. Ook is het niet helemaal duidelijk hoe het cijfer tot stand is gekomen. Heeft een bepaalde persoon de set een matig cijfer gegeven omdat alle vijf de aanbevelingen matig waren, of omdat er één goede tussen zat en de rest slecht was. Het vergelijken van de cijfers wordt op deze manier wat onduidelijk. Het cijfer dat de testpersonen hebben gegeven aan de aanbevelingensets is een rapportcijfer met een schaalverdeling van 1 tot en met 10. Een afdruk van de webpagina waar de gebruikers hun scores hebben ingevuld, is opgenomen in Bijlage D.
Tevens is aan elke testpersoon gevraagd hoeveel van de aanbevolen webpagina's hij al kende. Dit is gedaan om de cijfers deels te kunnen verklaren. Sommige testpersonen geven een hoog cijfer omdat de pagina die zij als aanbeveling krijgen goed aansluit bij hun interesses. Andere geven deze pagina's juist een slecht cijfer omdat ze hem al kende.

11.2 Problemen bij het onderzoek

Tijdens het doen van het onderzoek en na afloop van het onderzoek is een aantal problemen geconstateerd die de resultaten kunnen vertroebelen. Naar deze punten zal dan ook zeker gekeken moeten worden bij het doen van vervolgonderzoek.

Een eerste probleem is dat veel gebruikers webpagina's opgeven die ze vaak bezoeken maar die niet direct gekoppeld zijn aan hun interesses. Veel gebruikers hebben zoekmachines opgegeven als favoriete pagina's. Deze gebruikers zullen vaak ook als aanbeveling zoekmachines terugkrijgen. Dit is het *garbage in, garbage out* principe. Het is niet mogelijk om goede aanbevelingen te genereren als de kwaliteit van de inputdata niet goed is.

Een tweede probleem is dat de gebruikersgroep erg klein is. In totaal hebben 62 mensen meegedaan aan het onderzoek. Van deze 62 mensen hebben er 49 ook de scores ingevuld.

Tot slot moet nog opgemerkt worden dat de groep van testpersonen redelijk homogeen is. De groep van testpersonen is werkzaam bij hetzelfde bedrijfsonderdeel, in dezelfde branche, behoort tot ongeveer dezelfde leeftijdscategorie (gemiddeld 30 - 40 jaar) en bestaat voornamelijk uit mannen. Dit zijn een aantal factoren die wellicht van invloed zijn geweest op de resultaten van het onderzoek.

11.3 Onderzoeksresultaten

Voordat begonnen is met de gebruikerstest is een tweetal hypotheses opgesteld. Met de gebruikerstest kunnen deze hypotheses worden gecontroleerd.

De twee hypotheses luiden als volgt:

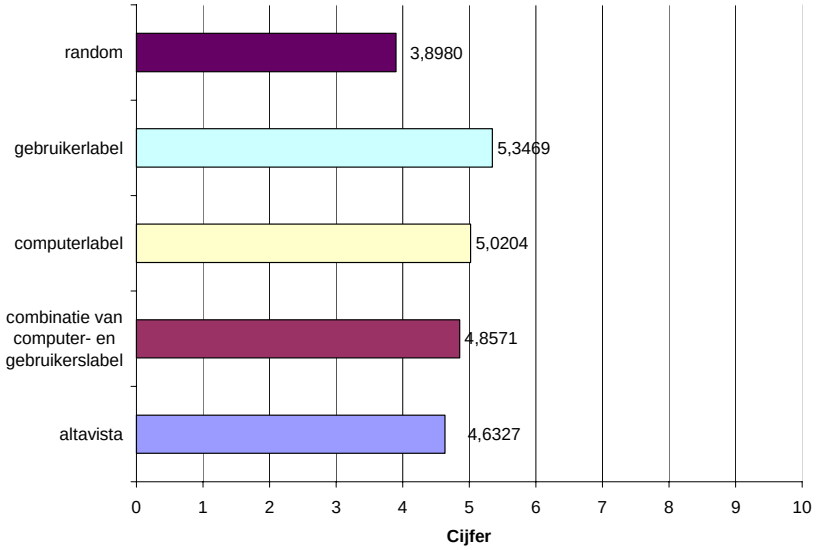
- De termen die gebruikers aan webpagina's geven zijn beter om mee te matchen dan de termen op de webpagina zelf.
- Aanbevelingen doen uit een beperkte set geeft betere aanbevelingen dan wanneer de set het gehele (of een heel groot deel) van het Internet bevat

In totaal zijn ongeveer 600 personen aangeschreven om mee te doen aan het onderzoek. Van deze 600 mensen hebben er 62 hun favoriete pagina's met beschrijvende termen ingevuld. In totaal werden er 612 webpagina's ingevuld waarvan er 427 uniek waren. Van de 62 testpersonen hebben er 49 de aanbevelingensets ook beoordeeld. Deze resultaten zijn kort samengevat in Tabel 82.

Aantal personen aangeschreven	600
Aantal reacties	62
Aantal webpagina's	612
Aantal verschillende webpagina's	427
Aantal scores	49

Tabel 82. Samenvatting onderzoeksresultaten

Voor het gebruikersonderzoek zijn een vijftal methoden gebruikt om aanbevelingen te genereren. Elke aanbevelingenset is door de testpersonen beoordeeld met een cijfer. Een overzicht van de scores voor de verschillende aanbevelingensets is weergegeven in Figuur 83.



Figuur 83. Resultaten van de aanbevelingensets

In de bovenstaande figuur is duidelijk te zien dat de random gegenereerde aanbevelingen het laagst scoren. Het commentaar van gebruikers dat veel bij deze aanbevelingenset naar voren komt, is dat de aanbevolen webpagina's niet interessant zijn en dat ze zich afvragen waarom ze deze pagina als aanbeveling krijgen. Het doen van aanbevelingen met behulp van een randomgenerator zal dus niet tot een hoge mate van tevredenheid leiden bij de gebruikers. Dit resultaat was uiteraard van te voren te voorspellen. De aanbevelingen zijn niet gebaseerd op enige kennis van de interesses van de gebruiker. De reden dat deze methode meegenomen is in het onderzoek, is dat het als ijkpunt kan dienen voor de andere methoden.

De tweede aanbevelingenset, die is gebaseerd op de match tussen gebruikerslabels van webpagina's en de uit de steekwoorden afgeleide interesses van de gebruikers, scoort het hoogst. De meeste gebruikers waren tevreden met deze aanbevelingen. Een oorzaak hiervoor zou bijvoorbeeld kunnen liggen in het feit dat meerdere gebruikers eenzelfde omschrijving (in de vorm van een steekwoord) gebruiken voor het beschrijven van een pagina over een bepaald onderwerp. Doordat meerdere mensen bijvoorbeeld de term 'voetbal' gebruiken als steekwoord voor een pagina over voetbal, zal een sterke match ontstaan tussen een gebruikersprofiel en een steekwoord. Deze sterke match zorgt er uiteindelijk voor dat mensen ook inderdaad de betreffende pagina over voetbal aanbevolen krijgen en ze (wellicht meer) tevreden zijn met de set van aanbevelingen. Dit zou bijvoorbeeld niet het geval zijn indien mensen bijvoorbeeld een ander taalgebruik

hanteren waardoor de een de term ‘voetbal’ opgeeft en de ander te term ‘balsport’. In dit geval zal de match minder sterk zijn. Het kan in het laatste geval zelfs zo zijn dat, gezien de ‘zwakke match’ er niet eens pagina’s over voetbal worden aanbevolen.Het is in dit laatste geval ook meer voor de hand liggend dat de gebruiker minder tevreden is met de set van aanbevelingen. Uit het onderzoek is gebleken dat verschillende pagina’s die over hetzelfde onderwerp gaan, met dezelfde termen zijn omschreven. Tevens zijn dezelfde pagina’s door verschillende testpersonen beschreven met dezelfde termen. Dit kan bijvoorbeeld weer het gevolg zijn van de homogeniteit van de groep van testpersonen. Dit bevestigt de eerste hypothese.

De derde set van aanbevelingen zijn gegenereerd met behulp van de computergegenereerde labels. Het interesseprofiel wordt bij deze aanbevelingenmethode vergeleken met de computergegenereerde labels in de labeldatabase. Deze labels zijn gegenereerd op basis van de tekst op een webpagina. Hierbij is in beschouwing genomen dat woorden die visueel opvallen zwaarder meewegen in het label. Deze set van aanbevelingen wordt net iets lager gescoord door de testpersonen dan de tweede aanbevelingenset. Dit zou kunnen komen doordat de tekst op een webpagina niet dezelfde woorden bevat als de woorden die in het interesseprofiel van de gebruiker staan.

De vierde set aanbevelingen is gegenereerd met een combinatie van de gebruikerslabels en de computergegenereerde labels. Het label dat is gegenereerd voor de webpagina’s is een combinatie van de tekst op de webpagina, en de woorden uit de Webpadcategorieën. Verwacht zou worden dat deze aanbevelingenmethode het hoogst gewaardeerd wordt omdat in theorie de beste aanbevelingen van de vorige twee sets worden gecombineerd tot één set. Deze set scoort echter lager dan de aanbevelingen die zijn gegenereerd op basis van de gebruikerslabels en lager dan de aanbevelingen die gegenereerd zijn op basis van de computergegenereerde labels. Hieruit blijkt dat de som van het geheel niet noodzakelijkerwijs groter of gelijk hoeft te zijn dan de som van de delen.

De vijfde aanbevelingenset is gegenereerd met behulp van de zoekmachine AltaVista. Met deze methode is op het Internet gezocht naar webpagina’s die matchen met termen uit een gebruikersprofiel. Deze set scoort weer iets lager dan de vierde methode. Deze aanbevelingenmethode genereert als enige methode aanbevelingen op basis van een veel grotere set van webpagina’s. In dit geval blijkt dat de kwaliteit van de aanbevelingen die gegenereerd zijn op basis van alleen de door de gebruikers bezochte URL’s, beter is dan de kwaliteit van de aanbevelingen die op basis van een groot deel van het Internet zijn gebaseerd.
Dit is te verklaren door het feit dat een favoriete webpagina van een gebruiker over het algemeen iets nuttigs of iets interessants bevatten (minstens nuttig of interessant voor één gebruiker). Het Internet bevat vele webpagina’s die niets nuttigs of niets interessants bevatten. Webpagina’s die zijn uitgezocht door mensen zijn over het algemeen beter dan webpagina’s die zijn uitgezocht met behulp van een machine. Door alleen te zoeken in de set met webpagina’s waarvan gebruikers hebben aangegeven dat deze interessant of nuttig zijn, kan al een verbetering van de kwaliteit van de aanbevelingen verkregen worden.
Een andere mogelijke verklaring kan zijn dat de groep van testpersonen erg homogeen is. Door de gedeelde interesses tussen de testpersonen is de kans al groter dat een aanbeveling aansluit bij de interesses van een gebruiker. Dit effect kan ook waargenomen worden in de score van de random gegenereerde aanbevelingen. Deze set scoort wel slechter dan de andere sets, maar niet voor iedereen.
In ieder geval bevestigt de score van deze vijfde aanbevelingenset de hypothese dat de aanbevelingen die gegenereerd zijn op basis van een beperkte informatieset, beter zijn dan de aanbevelingen die gegenereerd zijn op basis van een veel grotere informatieset.

De scores van de aanbevelingensets zijn te vinden in Bijlage F. Om de verschillen tussen de aanbevelingensets wat duidelijker te maken is normalisatie toegepast. Het gemiddelde van alle aanbevelingensets is hierbij van het gemiddelde van de afzonderlijke sets afgetrokken. Deze genormaliseerde score is opgenomen in Bijlage G.

Een moeilijkheid blijft het achterhalen van de interesses van de gebruiker. Met de gebruikte methode blijkt het goed te werken maar vergt wel wat inspanning van de gebruikers.

Er is ook gevraagd aan de testpersonen om het aantal bekende sites per aanbevelingenset op te geven. Op basis hiervan is gekeken naar het gemiddelde aantal bekende sites. Het resultaat hiervan staat in Bijlage H. Er lijkt een verband te bestaan tussen het aantal al bekende sites per aanbevelingenset en de score van de set. De aanbevelingen op basis van de gebruikerslabels scoren het hoogst, daarna de aanbevelingen op basis van de computerlabels, dan de aanbevelingen op basis van een combinatie van deze twee en tenslotte de aanbevelingen op basis van AltaVista. Wanneer wordt gekeken naar het aantal bekende sites in de aanbevelingen dan blijkt dat de aanbevelingenset op basis van de gebruikerslabels de meeste bekende sites bevat, daarna de aanbevelingen op basis van de computerlabels, dan de aanbevelingen op basis van een combinatie van deze twee en tenslotte de aanbevelingen op basis van AltaVista. Dit is dus exact dezelfde volgorde als de score van de aanbevelingen. Of dit verband ook daadwerkelijk aanwezig is zal moeten blijken uit vervolgonderzoek.

11.4 Aandachtspunten voor vervolg onderzoek

Wanneer de aanbevelingen goed gewaardeerd worden door de testpersonen, kan deze test worden uitgebreid door er een Proxy Server aan te koppelen. De aanbevelingen die de personen krijgen, worden in de cache van de Proxy Server gezet. Hiermee wordt bereikt dat het Internet sneller lijkt voor de gebruikers.

Zoals al in paragraaf 11.2 naar voren kwam, is het aantal gebruikers dat meegedaan heeft met dit onderzoek betrekkelijk klein. Ook is de groep redelijk homogeen. Een vervolgonderzoek zal dan ook gehouden moeten worden onder een veel grotere groep gebruikers.

Tevens is het aantal webpagina's dat gebruikt is als basis redelijk klein. In totaal hebben de gebruikers 437 verschillende webpagina's opgegeven. In veel gevallen bleek dat de gebruikers al veel webpagina's die zij als aanbeveling kregen al kenden. Het gebruiken van de logfiles van alle testpersonen zou een veel grotere basis bieden.

Ook zou gekeken moeten worden naar een dynamisch systeem. De aanbevelingen die gegenereerd zijn met het prototype, zijn gegenereerd op basis van éénmalige gegevens. Wanneer dit systeem in de praktijk zal gaan werken, zal er veel vaker een aanbevelingen gegenereerd moeten worden om aan te tonen dat de aanbevelingenmethoden werken.

Bij de uitgevoerde gebruikerstest is gekozen om elke aanbevelingenset te laten beoordelen door de gebruiker omdat op deze manier een mate van tevredenheid kan worden gemeten voor een bepaalde aanbevelingenmethode. In de praktijk is het lastig om te bepalen of een matig cijfer nu komt doordat alle aanbevelingen matig zijn, of omdat er maar één of twee heel goed zijn en de andere slecht doordat ze op basis van een verkeerde interesse zijn gebaseerd. Om een nog duidelijker beeld te krijgen van de kwaliteit van de aanbevelingen, zou elke aanbevolen webpagina apart een cijfer moeten krijgen van een testpersoon. Ook is het belangrijk dat er een goede instructie voor de gebruikers beschikbaar is zodat ze exact weten hoe ze de scores moeten invullen en waar ze op moeten letten.

12 Conclusies & Aanbevelingen

Dit hoofdstuk behandelt de conclusies en aanbevelingen die met betrekking tot dit onderzoek gedaan kunnen worden.

12.1 Conclusies

In dit onderzoek is aan de hand van een aantal geselecteerde methoden een prototype gebouwd voor een aanbevelingengenerator. Dit prototype dient ertoe om op basis van voorkeuren van gebruikers, Internetpagina's aan te bevelen waarin de gebruiker wellicht interesse heeft.

Hiervoor is uitgegaan van de volgende probleemstelling:

Welke methoden en technieken kunnen er worden gebruikt om webpagina's aan te bevelen aan Internetgebruikers (om ze uiteindelijk te kunnen pre-cachen) en hoe worden de aanbevelingen van de methoden ten opzichte van elkaar beoordeeld

De methoden en technieken op basis waarvan het prototype is gebouwd zijn de volgende:

- Filtertechnieken
- Labeltechnieken
- Verfijningmethoden
- Matchingtechnieken

Allereerst is hiervoor een keuze gemaakt tussen een drietal, in hoofdstuk 5 beschreven methoden van filteren, te weten Time Series Analysis, Collaborative Filtering en Filteren op basis van inhoud.
Hierbij is gebleken dat filteren op basis van inhoud het meest geschikt is. Aan de hand van deze methode kunnen aanbevelingen worden gegenereerd op basis van de inhoud van de webpagina's. Er wordt gekeken naar het 'onderwerp' van de webpagina, door simpelweg de voorkomens van woorden op de webpagina te tellen. Het onderwerp van de webpagina wordt vervolgens vergeleken met een profiel van een gebruiker om te kijken of deze aansluit bij de interesses van de gebruiker. In tegenstelling tot de andere genoemde technieken van filtering, is het filteren op basis van inhoud geschikt in situaties waarin er geen grote hoeveelheden data beschikbaar zijn (maar bijvoorbeeld slechts een paar webpagina's). Deze methode vraagt tevens weinig inspanning en input aan de kant van de gebruiker. De informatie die nodig is voor het uitvoeren van deze techniek wordt uit de bookmarktree van de gebruiker afgeleid.

Vervolgens zijn een aantal technieken besproken om van de inhoud van een webpagina een beschrijving te maken op basis waarvan op een later tijdstip matching plaats kan vinden. Hiervoor is de volgende methoden beschreven, te weten labelen op basis van tekstinhoud, labelen op basis van HTML-opmaak, labelen op basis van context en labelen op basis van Webpadcategorieën.
Voor het prototype is ervoor gekozen om geen gebruik te maken van het labelen op basis van context. Dit omdat er voor het labelen, een beschrijvingen gemaakt moeten worden van een bepaalde webpagina. In het geval van labelen op basis van context is het noodzakelijk om links te hebben op andere webpagina's die naar de betreffende webpagina wijzen. In dit onderzoek is uitgegaan van een verzameling webpagina's

waarvan in principe geen informatie beschikbaar was in de vorm van links naar de webpagina's in de verzameling. Labelen op basis van context is daarom niet toegepast. De overige methoden bleken binnen het kader van dit onderzoek goed bruikbaar voor het labelen van webpagina's.
Bij labelen op basis van tekstinhoud worden er op basis van platte tekst, woorden geteld. Aan de hand hiervan wordt een vector gecreëerd die aangeeft hoe vaak een woord op de webpagina voorkomt. Een vector van een webpagina is dus een lijst van woorden (op de webpagina) met een bijbehorende waarde (welke afhankelijk is van het aantal keer dat een woord op de webpagina voorkomt). Op deze manier wordt een eenvoudige handreiking geboden in het vergelijken van webpagina's met gebruikersprofielen (welke natuurlijk ook als een vector kunnen worden weergegeven). Met behulp van de HTML opmaak van een webpagina wordt gekeken naar woorden die een bepaalde nadruk hebben. Hierbij gaat het bijvoorbeeld om woorden die visueel opvallen doordat ze bijvoorbeeld vet zijn, cursief gedrukt zijn of van een groter lettertype zijn. Met behulp van de Webpadcategorieën wordt gekeken naar de beschrijvingen die de gebruikers aan hun bookmarks hebben gegeven. Op deze manier hebben de gebruikers zelf als het ware al een label meegegeven aan een bepaalde categorie van webpagina's. Er is voor gekozen om in het prototype deze drie technieken te combineren.

Bovengenoemde drie methoden zorgen ervoor dat bepaalde woorden op een webpagina een waarde meekrijgen die meeweegt in het creëren van een vector. Het meegeven van een waarde aan een woord kan echter nog beter gebeuren. Om deze reden zijn er een aantal verfijningtechnieken toegepast. Uit de beschreven verfijningtechnieken zijn er drie gekozen die zijn toegepast in het prototype, namelijk het filteren van stopwoorden, het toepassen van stemming en het uitvoeren van TFIDF.
Het filteren van stopwoorden zorgt ervoor dat woorden zoals 'de, het, een', woorden die niets zeggen over het onderwerp van de pagina, uit de beschrijving van de webpagina worden gefilterd. Op deze manier blijven de meer relevante woorden over in de 'beschrijvingsvector'. Stemming zorgt ervoor dat meervouden en enkelvouden van een woord als hetzelfde woord worden gezien. Ook vervoegingen van werkwoorden worden herleid tot de stam waardoor ze ook als een en hetzelfde woord worden beschouwd. Voor het toepassen van stemming is er gebruik gemaakt van de CELEX-database waarin de vervoegingen van vele werkwoorden en de meervouden en enkelvouden van veel zelfstandige naamwoorden zijn opgenomen. Er is gekozen om van een database gebruik te maken omdat dit eenvoudiger en betrouwbaarder is dan op regels gebaseerde technieken. TFIDF tot slot, past de gewichten van de termen in de beschrijving van een webpagina aan, door te kijken naar het globaal voorkomen van de woorden in alle (voor het onderzoek) beschikbare documenten. Een woord wat in de beschrijving van veel webpagina's voorkomt, zegt minder over die webpagina dan een woord dat maar op een enkele webpagina voorkomt.

Om te kijken of een webpagina op basis van de inhoud bij de interesses van een gebruiker aansluit, moet de inhoud vergeleken worden met een gebruikersprofiel. Hiervoor is een matchingmethode (het berekenen van de hoekafstand) gebruikt welke beschreven is in hoofdstuk 7. Dit is in het geval van dit onderzoek de meest 'passende' methode van vectormatching. De methode berekent de hoek tussen twee vectoren (welke zijn opgebouwd volgens de hierboven genoemde principes). Hoe kleiner de hoek, hoe groter de overeenkomst tussen de vectoren en dus tussen het document en de interesses van de gebruiker.

De methoden die hierboven zijn behandeld, vormen de basis voor het gebouwde prototype. Voor het prototype is tevens een keuze gemaakt voor de taal waarin het prototype is geschreven en de context waarin het prototype moet functioneren. De gehele Family Proxy is geschreven in LISP. Vanwege het feit dat delen van het prototype wellicht gebruikt gaan worden voor de Family Proxy is ervoor gekozen (en werd er vanuit het project gezien sterk benadrukt) om ook het prototype in LISP te schrijven. Dit om op een later tijdstip mogelijke compatibiliteitsproblemen te voorkomen en de integratie zo soepel mogelijk te laten verlopen. Om de reden dat er in de Family Proxy reeds gebruik werd gemaakt van CL-http is er tevens voor gekozen om hiermee ook in het prototype aan de slag te gaan. Voor de parser is er gebruik gemaakt van de HTML-parser van CL-http omdat dit binnen het project de enige beschikbare parser was.

Met betrekking tot de context is met name de Webpad van belang gebleken. De Webpad wordt in het Family Proxy project gebruikt. Bij de beschrijving van de webpagina's is de Webpad reeds genoemd. Het voordeel van de Webpad voor het prototype is dat het informatie, over zowel de gebruiker als over webpagina's bevat. Aan de hand van de informatie die aanwezig is (in de vorm van Webpadcategorieën) in de Webpad kan een interesseprofiel van een gebruiker worden opgesteld. Deze interesseprofielen zullen uiteindelijk nodig zijn voor het kunnen genereren van aanbevelingen. Bij het gebrek aan gebruikers die beschikking hadden over een Webpad ten tijde van het onderzoek, zijn de Webpadcategorieën van een gebruiker gesimuleerd voor het gebruikersonderzoek. De gebruikers hebben hiervoor expliciet steekwoorden opgegeven.

Voor het gebruikersonderzoek is heel KPN research aangeschreven met de vraag om mee te doen aan het onderzoek. Het betrof hier een groep van zo'n 600 personen, waarvan er uiteindelijk 62 hebben gereageerd. Hiervoor is door elke gebruiker een tiental pagina's opgegeven met daarbij een aantal steekwoorden per pagina die volgens hen het beste de lading van de pagina dekken. Een nadeel van de groep testpersonen is dat het gaat om een redelijk homogene groep mensen. De groep van testpersonen is werkzaam bij hetzelfde bedrijfsonderdeel, in dezelfde branche, behoort tot ongeveer dezelfde leeftijdscategorie (gemiddeld 30 - 40 jaar) en bestaat voornamelijk uit mannen. Dit zijn een aantal factoren die wellicht van invloed zijn geweest op de resultaten van het onderzoek. Dit kan met het onderzoek echter niet worden aangetoond, maar dient zeker in het achterhoofd te worden gehouden bij het verwerken van de resultaten en het uitvoeren van een vervolgonderzoek.
De door de gebruikers opgegeven pagina's zijn gebruikt als input voor het onderzoek en daarmee voor het testen van het prototype. Op basis van deze pagina's zijn een vijftal sets van aanbevelingen gegenereerd, te weten:

- Random aanbevelingen (compleet willekeurig uit de set van aanwezige pagina's)
- Aanbevelingen op basis van gebruikerslabels (op basis van Webpadcategorieën)
- Aanbevelingen op basis van computergegenereerde labels (tekstinhoud en HTML-opmaak)
- Aanbevelingen op basis van zowel gebruikerslabels als computergegenereerde labels
- Aanbevelingen op basis van de Internet zoekmachine AltaVista (zoeken op Internet aan de hand van de interesseprofielen van gebruikers)

De resultaten van de vijf sets van aanbevelingen zijn teruggekoppeld naar de gebruikers die vervolgens per set een waardering hebben gegeven. Uit deze waardering is gebleken dat **aanbevelingen op basis van de gebruikerslabel** het beste werd gewaardeerd ten opzichte van de andere sets. Hierna volgden in aflopende volgorde van waardering de computergegenereerde labels, de combinatie van beide methoden, aanbevelingen door AltaVista en Random aanbevelingen. Uit het onderzoek is gebleken dat verschillende pagina's die over hetzelfde onderwerp gaan, met dezelfde termen zijn omschreven. Tevens zijn dezelfde pagina's door verschillende testpersonen beschreven met dezelfde termen.
Een kanttekening die gemaakt moet worden bij het gebruikersonderzoek is dat de gebruikers de aanbevelingenset hebben gewaardeerd en niet de afzonderlijke aanbevelingen. Er lijkt namelijk een verband te bestaan tussen het aantal bekende sites en de waardering van de aanbevelingen. Dit kan aan de hand van dit onderzoek echter niet worden bewezen. Hier is dus niet te achterhalen welke criteria er bij het waarderen van de sets door de gebruiker zijn gehanteerd.

De verschillende talen waarin webpagina's gepubliceerd worden op Internet blijkt wel een probleem te zijn wat bij de waardering van de sets naar boven komt. Veel gebruikers geven bij de aanbevelingen als opmerking aan dat de sites ongetwijfeld interessant zouden zijn, maar dat ze de taal waarin de webpagina is geschreven niet kunnen lezen.

Uit het onderzoek is gebleken dat het mogelijk is om aanbevelingen voor gebruikers te genereren met behulp van filteren op basis van inhoud. Tevens blijkt dat de aanbevelingen die gegenereerd zijn op basis van een kleine set URL's (de door gebruikers bezochte webpagina's) beter scoren dan aanbevelingen die gebaseerd zijn op een groot deel van het Internet. Dit blijkt uit het feit dat de aanbevelingen die gegenereerd zijn met de zoekmachine AltaVista niet erg goed werden gewaardeerd. Er is in het kader van het onderzoek bewust gekozen om gebruik te maken van een selecte set van URL's die door de testpersonen zijn aangedragen. Dit omdat de kwaliteit van pagina's die door personen specifiek worden aangegeven als interessant, waarschijnlijk meer toegevoegde waarde hebben voor een andere gebruiker, dan willekeurige pagina's op Internet welke gerelateerd zijn aan de interesses van de gebruiker. Met betrekking tot het al eerder genoemde 'voetbal' voorbeeld betekent dit bijvoorbeeld dat een pagina aanbevolen door een andere gebruiker (kleine set), daadwerkelijk over het onderwerp voetbal gaat terwijl een pagina op Internet (grote set) een pagina oplevert waarop bijvoorbeeld een voetbal te koop aan wordt geboden.
Een conclusie die hieruit tevens volgt, is dat er bij aanwezigheid van grotere hoeveelheden pagina's, de kwaliteit van de aanbevelingen die het systeem genereert zou kunnen afnemen door het optreden van het zogenaamde 'Garbage in, Garbage out' effect. Dit principe geeft onder meer aan dat de kwaliteit van de aanbevelingen afhankelijk is van de kwaliteit van de gegevens die zich in het systeem bevindt. In het geval van het onderzoek is dit de input die door de gebruikers is aangeleverd op basis waarvan aan de ene kant het interesseprofiel van de gebruiker wordt bepaald en aan de andere kant de steekwoorden (nodig voor het maken van een label voor de pagina) voor een pagina worden gegenereerd. Als de gebruikers vervuilde gegevens in het systeem invoeren zal dit resulteren in slechtere aanbevelingen.

Tot slot is het gebleken dat het moeilijk is om de interesses van de gebruiker te achterhalen zonder dat de gebruiker hier direct om gevraagd wordt. Men heeft wel de interesses van de gebruiker nodig om uiteindelijk aanbevelingen te kunnen doen, maar wil de gebruiker hier niet mee lastig vallen. Om die reden wordt er geprobeerd om de interesses zo goed mogelijk af te leiden uit informatie die al beschikbaar is, zoals in dit onderzoek is gebeurd aan de hand van Webpadcategorieën. Hierbij loopt men nog steeds het risico dat men indirect 'verkeerde' conclusies trekt omtrent de interesses van de gebruiker. De in het onderzoek gebruikte methode, waarbij gebruikers URL's met bijbehorende steekwoorden opgeven, blijkt goed te werken en vergt minimale inspanning van de gebruikers. Toch is het belangrijk dat men inspanning aan de kant van de gebruiker probeert te vermijden.

Veel gebruikers zullen geen benul hebben van het nut van de bookmarks die ze aanmaken en de omschrijving die ze hieraan meegeven. Beide zijn uiteindelijk van groot belang voor de kwaliteit van de aanbevelingen die ze terug zullen krijgen. Het is daarom van groot belang dat, indien een dergelijk project op grote schaal gaat worden uitgevoerd, er een goede communicatie hierover naar de eindgebruiker moet plaatsvinden. Met het besef van het belang van zijn eigen rol en invloed in dit proces, staat of valt het succes van het doen van aanbevelingen.

12.2 Aanbevelingen

In deze paragraaf zullen aanbevelingen worden gedaan voor verbetering van (het testen van) allereerst het labelsysteem aan de hand van genoemde knelpunten. Vervolgens zullen aanbevelingen worden gedaan voor het verbeteren van het genereren van aanbevelingen en zullen tot slot aanbevelingen worden gedaan voor het doen van vervolgonderzoek.

Verbeteringen voor het labelssysteem

Het systeem houdt op dit moment geen rekening met de verschillende talen waarin webpagina's worden gemaakt. Omdat webpagina's op inhoud worden vergeleken met de interesseprofielen van de gebruikers, is de taal waarin een webpagina is geschreven, belangrijk. Wanneer een document en een interesseprofiel die in een verschillende taal geschreven zijn, worden vergeleken, zal dit niet het gewenste resultaat opleveren omdat

de taal van het interesseprofiel niet matcht met de taal van het document. Engelse documenten zoeken op basis van Nederlandse kernwoorden is zonder een vertaalslag niet mogelijk.
Een oplossing zou zijn om pagina's die zijn geschreven in een andere taal, automatische te laten vertalen. Op Internet bestaan al verschillende (gratis) vertaalpagina's. Deze geven een redelijke vertaling van een webpagina. Uit de vertaalde woorden kan begrepen worden waar een webpagina over gaat. De vertaling zelf is vaak niet goed leesbaar, maar om te bepalen of de webpagina aansluit bij de interesses van een gebruiker, is een slechte vertaling vaak al voldoende, het gaat tenslotte om de steekwoorden. Een eis is natuurlijk wel dat de persoon voor wie de aanbeveling bedoeld is, de taal kan lezen.
Een ander probleem met de verschillende talen is dat sommige verfijningtechnieken zoals stemming en synoniemen zoeken ook taalafhankelijk zijn. De software zou met een language awareness uitgerust kunnen worden zodat stemming en synoniemen effectief toegepast kunnen worden. Het taalbewust maken zou gedaan kunnen worden met de woordenlijsten in de CELEX database. CELEX heeft voor iedere taal een aparte database. Van een webpagina kunnen de woorden van de eerste paar regels worden vergeleken met de verschillende databases van CELEX. Aan de hand hiervan kan worden bepaald in welke taal de webpagina is geschreven.
Ook zouden op een andere manier aanbevelingen kunnen worden gegenereerd door bijvoorbeeld gebruikers te clusteren naar interessegebied. Op dat moment zijn de gebruiker met overeenstemmende interesses al bij elkaar gebracht. Aanbevelingen kunnen nu gegenereerd worden door voor een gebruiker webpagina's aan te bevelen van een andere gebruiker die zicht binnen hetzelfde gebruikerscluster bevindt.

Verbeteren van het genereren van aanbevelingen

Een manier tegenovergesteld aan degene die in de vorige alinea is genoemd, is het clusteren van labels. Het zoeken van matchende webpagina's wordt hierdoor versneld. Dit omdat er clusters van pagina's worden vergeleken met een interesseprofiel van en gebruiker, waarna het hele cluster (indien er en match is) kan worden aanbevolen. Deze methode heeft echter alleen zin indien er veel aanbevelingen gegenereerd moeten worden en/of de set van labels redelijk constant is. Er dient hierbij echter rekening te worden gehouden met het feit dat het telkens opnieuw uitvoeren van het clustering algoritme rekenintensief is.

Aanbevelingen voor vervolgonderzoek

Allereerst is het van belang om de groep met gebruikers verder uit te breiden en ervoor te zorgen dat de samenstelling van de groep testpersonen redelijk heterogeen is. Denk hierbij aan verschillende leeftijdscategorieën, geslacht, beroep etc.
Tevens is het van belang om meer informatie over de gebruikers ter beschikking te hebben. Denk hierbij bijvoorbeeld aan log files waarin de reeds bezochte Internetpagina's over bijvoorbeeld de afgelopen 2 weken vastligt. Aan de hand hiervan kunnen deze bezochte pagina's (die dus al voor de gebruiker bekend zijn) uit de aanbevelingen voor een gebruiker worden gefilterd. Op dat moment kan men gaan waarderen of de gegenereerde aanbevelingen een toegevoegde waarde hebben voor de gebruiker. Hierbij dient rekening gehouden te worden dat niet iedereen evenveel 'suft'. Dit heeft invloed op de hoeveelheid informatie er beschikbaar is van een persoon en zal doorwerken in de kwaliteit van de aanbevelingen.
Het is hierbij tevens de bedoeling dat er een meer dynamische omgeving voor het genereren van aanbevelingen ontstaat. Dit in tegenstelling tot het werken vanuit een momentopname.

13 Referenties

[1] *G. Attardi, S. Di Marco, D. Salvi,* Categorization by context, Università di Pisa,

[2] *Rakesh Agrawal, Johannes Gehrke, Dimitrios Gunopulos, Prabhakar Raghavan,* Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications, IBM Almaden Research Center

[3] All about the Internet: A Brief History of the Internet, Internet Society (ISOC), http://www.isoc.org/internet-history/brief.html, augustus 2000.

[4] *Babuska, Robert.,* Knowledge-Based Control Systems, november 1999.

[5] *Daniel Boley, Maria Gini, Kyle Hastings, Bamshad Mobasher, J. Moore,* A Client-Side Web Agent for Document Categorization, University of Minesota

[6] *Daniel Boley, Maria Gini, Robert Gross, Eui-Hong (Sam) Han, Kyle Hastings, George Karypis, Vipin Kumar, Bamshad Mobasher, Jerome Moore,* Partition –Based Clustering for Web Document Categorization, University of Minesota - Departement of Computer Science and Engineering, Minneapolis

[7] *Daniel Boley, Vivian Borst,* Unsupervised Clustering: A Fast Scalable Method for Large Datasets, University of Minesota

[8] *George E. P. Box, Gwilym M. Jenkins,* Time Series Analysis – forecasting and control, 1970

[9] *Breese, J. S., Heckernab, D., Kadie, C.,* Emperical Analysis of Predictive Algorithms for Collaborative Filtering, juli 1998.

[10] *Enting, Henk., Oldengarm, Petra., Peper, Christian.,* Indexeren, classificeren en clusteren voor personalisatie, KPN Research, januari 2000.

[11] *Van de Nieuwe Giessen, Dick,* Onderzoek en ontwikkeling bij KPN – Een geschiedenis van de eerste honderd jaar, december 1995.

[12] *Paul Graham,* Ansi Common Lisp, Prentise Hall, 1996

[13] *Jane L. Harvill,* An Introduction to Time Series Analysis, http://cba/bgsu.edu/asor/facstaff/jarvil/416/lectures/lecture1/lecture1.html, januari 1997

[14] History of the Internet and WWW: Road 1 – USA to Europe, http://www.internetvalley.com/intval.html

[15] Homeservices Newsletter, nummer 2, oktober 2000.

[16] Jaarverslag KPN 1999

[17] *Sonya E. Keene,* Object Oriented Programming in Common Lisp: A Programmers Guide to the Common Lisp Object System, Addison Wesley, 19

[18] *Kirschke, Heiko.,* Persistent Lisp Objects User Guide, mei 2000.

[19] *Lake, David.,* The Web: Growing by 2 million Pages a Day, http://www.thestandard.com/research/metrics/display/0,2799,12329,00.html, februari 2000.

[20] *Ah-Hwee Tan,* Text Mining: The state of the art and the challenges, Kent Ridge Digital Labs, 1999

[21] *Thomas K. Landauer, Peter W. Foltz, Darrell Laham,* An Introduction to Latent Semantic Analysis, Discource Processes, 25, 259-284

[22] *Liempd, G. van., Oudshoff, Sandra.,* Nice News Now, KPN Research, december 1999.

[23] *H. P. Luhn,* The automatic creation of literature abstracts, IBM Journal of Research and Development, 2, 159-156, 1958

[24] *Roger S. Pressman,* Software Engineering, A Practitioner's Approach, McGrawHill, 1997

[25] *Dany Sullivan,* Search Engine Sizes, Searchenginewatch.com, april 2001

Bijlage A. E-mail voor medewerking onderzoek

Sent: Monday, December 11, 2000 5:49 PM
To: KPN Research Groningen; KPN Research Leidschendam; KPN Research Twente
Subject: research on website recommendation / onderzoek naar anbevelingen van websites

(***English version below)***

Hallo Researcher,

Wij doen een onderzoek naar het geven van aanbevelingen van webpagina's. Voor dit onderzoek vragen wij jouw hulp.

Beschrijving onderzoek
Voor dit onderzoek willen we graag weten welke sites en onderwerpen je interesseren. Wij doen je een aantal aanbevelingen van webpagina's die op verschillende manieren tot stand zijn gekomen. Vervolgens vragen wij je om een oordeel te geven over deze aanbevelingen. Aan de hand van deze uitkomsten zullen wij onze hypothesen kunnen toetsen.

Achtergrond: vanuit Personal Proxy interesse in methoden voor doen van aanbevelingen
Vanuit het project Personal Proxy is de vraag gerezen hoe we zo goed mogelijk aanbevelingen van webpagina's kunnen geven aan een groep gebruikers. Voor het bepalen van een verzameling aanbevelingen voor een gebruiker kunnen tal van methoden gebruikt worden. Onze veronderstelling is dat we ons daarbij niet moeten richten op hoe een computer zo goed mogelijk deze aanbevelingen kan ophoesten (de kwaliteit van dit soort aanbevelingen laat ernstig te wensen over), maar dat we proberen zoveel mogelijk andere gebruikers als 'tipgevers' te gebruiken. Daarnaast veronderstellen wij dat het beter is aanbevelingen te selecteren uit een deel van het WWW - het deel dat de gebruikers van het systeem bezoeken - in plaats van het hele WWW. Om deze hypothesen te toetsen, hebben wij een onderzoekje opgezet.

Help je ons?
Wij zouden het heel erg op prijs stellen als je ons wilt helpen. **Dit kost je ongeveer 10 en daarna nog eens 5 minuten**. Wij hebben een webpagina gemaakt: http://pc4413.research.kpn.com/~wouter . Wij verzoeken je hierop 10 URL's in te vullen die je erg interessant vindt en in twee of drie woorden aan te geven waar deze pagina over gaat. Bijvoorbeeld: www.ns.nl, treintijden, openbaar vervoer. Vervolgens wordt deze informatie door ons verwerkt en krijg je aan de hand van deze beschrijvingen 4 maal een paar webpagina's aanbevolen. Wij verzoeken je deze te bekijken (*als het goed is, zijn ze interessant!)* en vervolgens aan te geven wat je ervan vindt.

In het kort
1) Vul in de webpagina http://pc4413.research.kpn.com/~wouter
voor 15 december
2) In de loop van volgende week krijg je van ons vier setjes van aanbevelingen
volgende week
2) Geef voor elk van deze lijstjes een score (de website om dit te doen komt binnenkort)
volgende week

Wij hopen dat je meedoet,

Wouter van den Eijkel en Alan Verberne

Als je vragen of opmerkingen hebt kun je ons bellen of e-mailen.

English version

Hi Researcher,

We are doing a project on how to make recommendations of webpages. We would like your help.

Experiment description
For this experiment we would like to know which sites and subjects you're interested in. We will give you a number of recommendations of websites that have been generated in a number of different ways. Than we will ask you to rate these sets of recommendations. Using the outcome of this experiment we will be able to test our hypotheses.

Background: Interest in methods for generating recommendations in project Personal Proxy
We would like to know which methods are most suited for generating recommendations of webpages. We assume that in order to do so, we should not use (fancy) artificial intelligence techniques (quality of computer generated recommendations is often very poor) , but we should try and use the users of the system themselves as recommenders as much as possible. Next to that, we assume it's better to select recommendations not from the entire WWW but only from a part of it, that is, the part that the system's users actually visit. To test these hypotheses, we are conducting an experiment.

Will you help us?
We would very much appreciate your help. **It will cost you about 10 minutes and then 5 minutes** more. We have made a webpage: http://pc4413.research.kpn.com/~wouter . We request you to fill in 10 URLs you find very interesting and to describe each of them in two or three words. For example www.ns.nl, departure times, public transport. Your information will be processed by us and you will be given 4 sets of recommendations. Finally we ask you to look at these recommendations (*it should be interesting!*) and rate them.

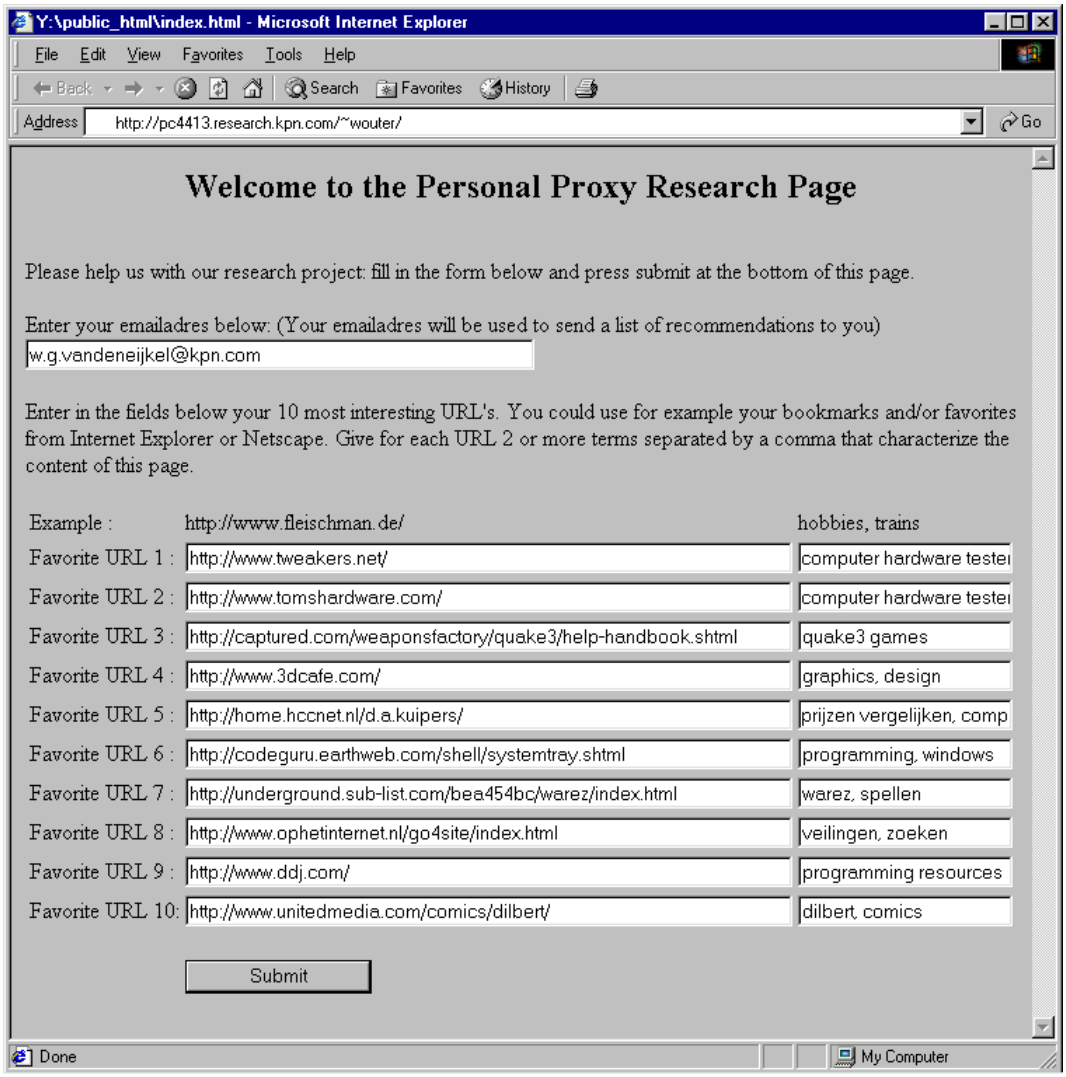
In short
1) Fill in the webpage http://pc4413.research.kpn.com/~wouter
   before december 15th
2) Next week you will receive your recommendations
   next week
3) Rate the sets of recommendations (website will be available shortly)
   next week

We are counting on you,

Wouter van den Eijkel, Alan Verberne

If you have any questions, please call or email us

Bijlage B. Formulier gebruikersonderzoek



Bijlage C. E-mail aanbevelingen naar testpersonen

Sent: Tuesday, January 16, 2001 2:13 PM
To: w.g.vandeneijkel@kpn.com
Subject: your recommendations, uw aanbevelingen

(English version below)

Hallo,

Een aantal weken geleden heeft u uw medewerking verleend aan ons onderzoek naar aanbeveling van webpagina's. Hiervoor zijn wij u zeer dankbaar. Wij hebben voor u aantal webpagina-aanbevelingen klaarliggen. Om precies te zijn, zijn dit 5*5 aanbevelingen (enkele uitzonderingen daargelaten). Dit zijn 5 aanbevelingen per experimentele conditie. Wij verzoeken u de aanbevolen pagina's te bekijken en voor elk van de 5 setjes een totaalscore te geven die aangeeft hoe interessant of misschien wel, hoe leuk u de aanbevelingen vond. Als wij uw scores binnenhebben, zult u van ons te horen krijgen op welke manier elk van de setjes is gegenereerd en zult u uitleg krijgen over het precieze doel van ons onderzoek. (Aanvankelijk was het de bedoeling nog einde vorig jaar u de aanbevelingen te geven. Enkele technische problemen hebben dit vertraagd, onze excuses hiervoor.)

wat te doen?

- ga naar de pagina http://pc4413/~wouter/recommendation.php3?email=w.g.vandeneijkel@kpn.com U krijgt nu 5 'recommendation lists' te zien.
- Bezoek alle aanbevolen sites (kort) en geef per lijstje van 5 een totaalscore
- Geef aan per lijstje hoeveel van de aanbevolen pagina's u al kende
- druk onderaan de pagina op 'submit scores'

Klaar!

Binnenkort ontvangt u achtergrondinformatie over ons onderzoek. Bedankt voor uw medewerking,

Wouter van den Eijkel en Alan Verberne

English version

Hello,

A few weeks ago you helped us with our research on webpage recommendation. We are very grateful for your help. We have generated some webpage recommendations for you, 5 sets of 5 recommendations to be precise (except for a few cases). These are 5 recommendations per tested condition. We request you to visit each of the pages that are recommended and give each set of recommendations an overall score indicating how interesting (or fun!) the pages of that set are. When we have received your scores, you will be told which conditions were tested and you will receive more background information on our research. (At first we had planned to give you these recommendations at the end of last year. Unfortunately, we have ran into some technical problems. We

apologise for the delay.)

What to do?

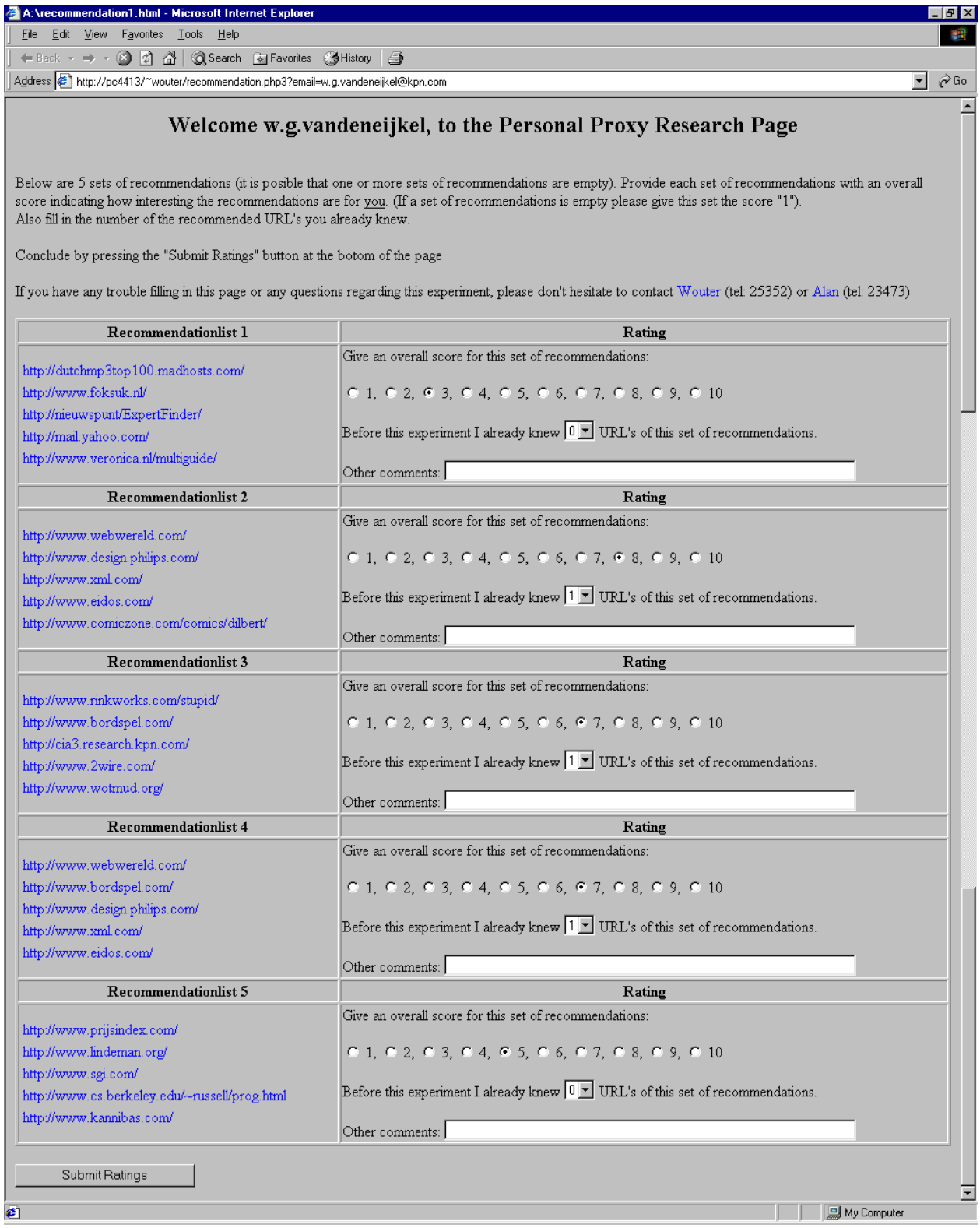
- Visit http://pc4413/~wouter/recommendation.php3?email=w.g.vandeneijkel@kpn.com Now you will see your 5 recommendation lists
- Visit the recommended sites (shortly) and give an overall score for each list of recommendations
- Tell us how many of the recommended pages you already knew
- press 'Submit scores' at the bottom of the page.

You're finished!

You will soon receive some background information on our research. Thank you for your cooperation,

Wouter van den Eijkel en Alan Verberne

Bijlage D. Aanbevelingenpagina



Bijlage E. E-mail met resultaten naar testpersonen

Sent: dinsdag 30 januari 2001 16:54
(English version below)

Hallo,

Bedankt voor je deelname aan ons onderzoek naar webpagina-aanbeveling. Zoals toegezegd enige achtergrondinformatie over ons onderzoek.

Methoden voor webpagina-aanbeveling
Om aanbevelingen van webpagina's aan gebruikers te doen, gebruiken we de methode *content matching*. Bij deze methode wordt de inhoud van pagina's bekeken. Hieruit komt een inhoudsbeschrijving welke wordt vergeleken met de interesses van de gebruiker. Komt de inhoudsbeschrijving (het *label*) en de beschrijving van de interesses genoeg overeen, dan wordt deze pagina aan de gebruiker aanbevolen.

Onderzoeksvraag

In ons onderzoek hebben we ons gericht op content matching en ons afgevraagd *op welke manier het genoemde label van een Internetpagina het beste kan worden gemaakt*. Dit kan door
1) automatisch de tekst (en evt. opmaak) van het document door te lopen, zoals zoekmachines dit doen.
2) gebruikers een label te laten geven (zoals u dat heeft gedaan door de pagina in enkele woorden te beschrijven)
3) een combinatie te maken van labels 1 en 2.

Onze hypothese was (o.a) dat labels gemaakt door gebruikers een betere beschrijving bieden dan automatisch gegenereerde labels en dat labels die voortkomen uit een combinatie het beste zijn.

Test

Om de bovengenoemde vraag te beantwoorden hebben we de volgende condities getest:
1) willekeurige aanbevelingen (als benchmark)
2) aanbeveling van pagina's door vergelijken interesses en *door gebruikers gemaakte* labels
3) aanbeveling van pagina's door vergelijken interesses en *automatisch aangemaakte* labels
4) aanbeveling van pagina's door vergelijken interesses en *combinatie* van gebruikers- en automatisch aangemaakt label
5) zoeken met Altavista (tweede benchmark)

Uitkomst

De uitkomst was de volgende rangorde:

1 aanbeveling van pagina's door vergelijken interesses en *door gebruikers gemaakte* labels
2 aanbeveling van pagina's door vergelijken interesses en *automatisch aangemaakte* labels

3 aanbeveling van pagina's door vergelijken interesses en *combinatie* van gebruikers- en automatisch aangemaakt label
4 zoeken met Altavista (tweede benchmark)
5 willekeurige aanbevelingen (als benchmark)

Voor meer informatie over het onderzoek, kun je natuurlijk bellen met Alan Verberne, tel 23473

Nogmaals bedankt!

Hello,

Thank you for participating in our research on webpage recommendation. As promised, we conclude by giving you some background on our research:

Techniques for webpage recommendation

To give users recommendations of webpages, we apply a method called *content matching*. This method examines the content of Internet pages. Their content is to construct a description of the page's contents, this description is compared with the users' interests. If a page's content description and the description of his interests match enough., then the page is recommended to this user.

Topic of the research

Our research has addresses content matching. We posed ourselves the question of *how to construct the description of an Internet page in the best possible way.*
This can be done in the following ways:
1) by automatically examining the text (and lay-out) of the page - just like search engines do
2) by having the system's users label Internet pages (you gave labels to the sites that were your favorites)
3) By combining labels 1) and 2)

Our hypothesis is that man-made labels (by actual users) offer a better description of a page than automatically generated labels and that labels constructed by combining the two types can be better than either one of them.

Test

To answer the above question, we have tested the following conditions:
1) random recommendations (benchmark)
2) pages recommended by comparing users' interests and *human made* page descriptions
3) pages recommended by comparing users' interests and *automatically generated* page descriptions
4) pages recommended by comparing users' interests and page descriptions based on *a combination* of man- and computer made descriptions.
5) recommendations by searching with AltaVista on the users' interests.

Outcome

The outcome of the research was the following order in the recommendations' quality

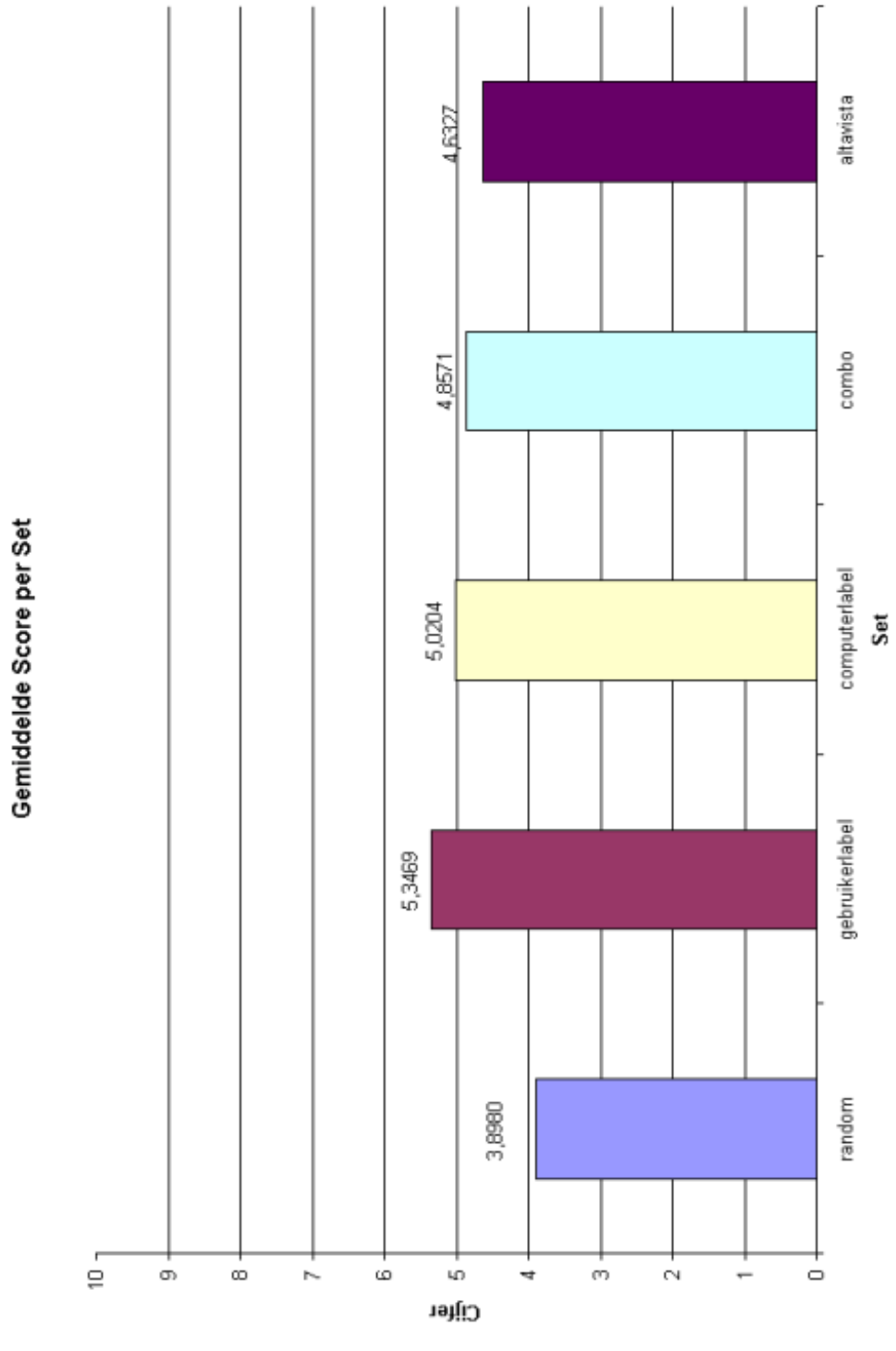
1 pages recommended by comparing users' interests and *human made* page descriptions
2 pages recommended by comparing users' interests and *automatically generated* page descriptions
3 pages recommended by comparing users' interests and page descriptions based on *a*

combination of man- and computer made descriptions.
4 recommendations by searching with AltaVista on the users' interests.
5 random recommendations (benchmark)

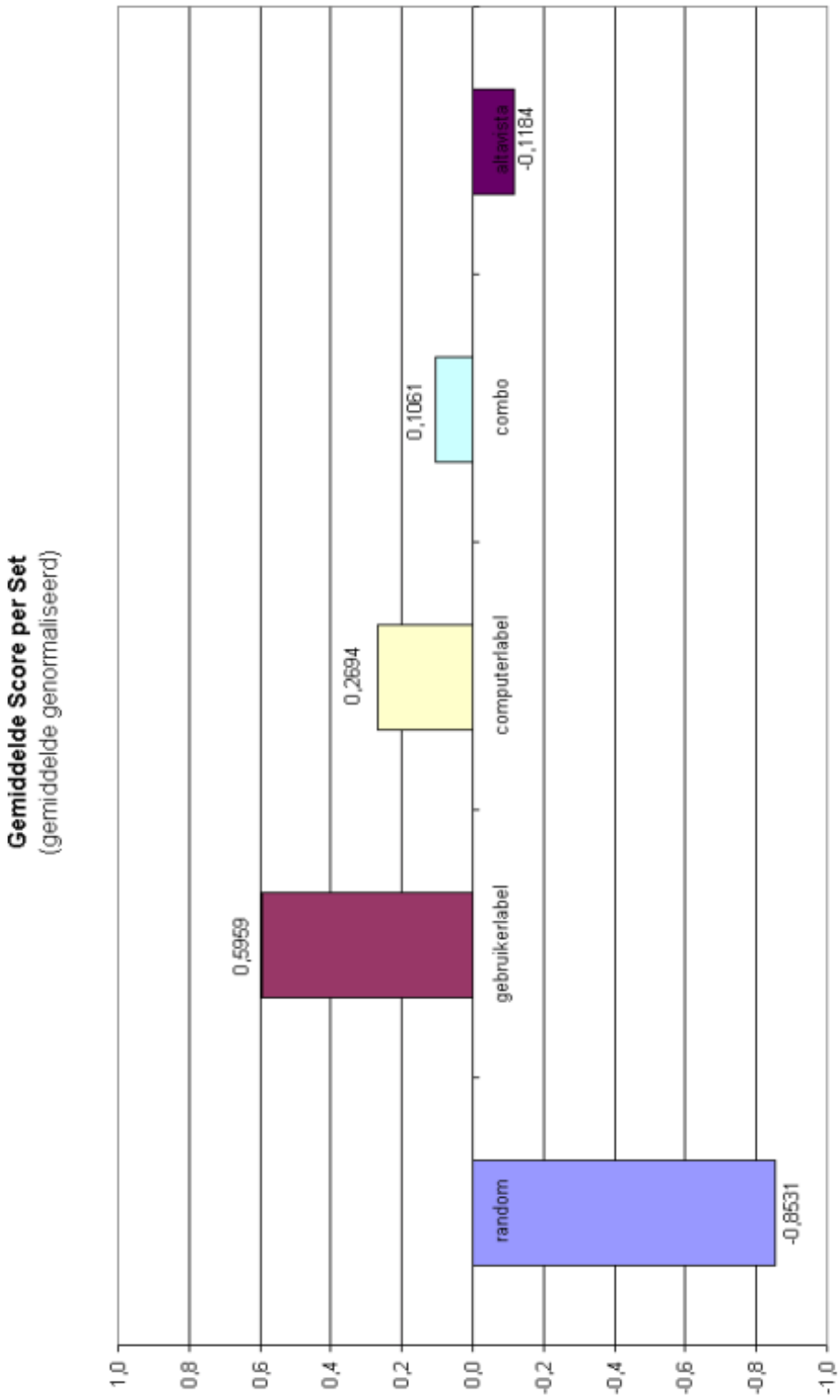
For more information, don't hesitate to call Alan Verberne (tel. 23473)

Thanks again for your co-operation.

Bijlage F. Gemiddelden per set



Bijlage G. Genormaliseerde Gemiddelden



Bijlage H. Gemiddeld aantal bekende sites