Centraal Bureau voor de Statistiek

Postbus 4000
2270 JM Voorburg

Visualisatie Statistische gegevens

Afstudeerscriptie Master Software Engineering

Universiteit van Amsterdam

Chris Houwing

Samenvatting: Het ontwikkelen van een grafische module voor het visualiseren van statistische gegevens.

Trefwoorden: SVG, Data Visualisatie

Afstudeerdocent: P. Klint
Stagebegeleider: E. de Jonge / O. ten Bosch
Opdrachtgever: Centraal Bureau voor de Statistiek
Datum: *21 juni '04*

Inhoudsopgave

1 Samenvatting

De mogelijke nieuwe standaard, Scalable Vector Graphics (SVG) verdrijft nu al de raster images bij het Centraal Bureau voor de Statistiek (CBS). StatWeb 4.0, het digitale loket voor het opvragen van statische gegevens, is momenteel nog in de ontwikkelingsfase. De nieuwe versie van StatWeb is geheel .NET gebaseerd, een mooi moment om eens na te denken over een bijpassende visualisatie methode voor het tonen van data.

Tijdens een iteratief proces is gekeken: welk ontwerp het beste paste bij het nieuwe StatWeb, rekening houdend met de toekomstige functionaliteiten die het StatWeb zal gaan bieden. Er is in C# een prototype ontwikkeld welke een grafische visualisatie kan genereren uit data, al dan niet statistisch. Er wordt via JavaScript gereageerd op de gebruiker, om zo het geheel een dynamische uitstraling te geven.

Zoals na elk traject is verbetering zonder evaluatie niet mogelijk, er wordt dan ook gekeken wat er goed is gegaan en wat er tijdens een volgend traject anders zou moeten. Wat waren onvoorziene problemen en hoe zouden deze in de toekomst voorkomen kunnen worden.

2 Achtergrond en onderzoeksvraag

2.1 Introductie

Wat doet het Centraal Bureau voor de Statistiek

Het Centraal Bureau voor de Statistiek (CBS) heeft tot taak het verzamelen, bewerken en publiceren van statistieken ten behoeve van praktijk, beleid en wetenschap. Naast de verantwoordelijkheid voor de nationale (officiële) statistieken is het CBS ook belast met de productie van Europese (communautaire) statistieken.[1]

Het CBS publiceert deze gegevens op verschillende manieren, waarvan er maar één van belang is voor dit onderzoek: publicatie via StatLine[2]. Via StatLine kan een ieder met toegang tot het Internet de aangeboden publicaties opvragen en inzien. We moeten er dus rekening mee houden dat de gebruikers variëren van studenten, journalisten tot huisvrouwen en onderzoeksbureaus.

2.2 Context

Wat gaat er gebeuren

Anno juni ’04 maakt het CBS gebruik van drie nauw samenwerkende systemen voor het bouwen, publiceren en onderhouden van publicaties. Voor het bouwen van publicaties wordt StatBuild gebruikt. Deze is geschreven in Delphi 2, waarvan de code al minimaal 4 jaar in de ijskast staat. Het publiceren gebeurt door middel van StatWeb 3.1 welke geschreven is in een verzameling van: ASP, VB, C++, Java, JavaScript en Com. Dit systeem is nog het beste onderhouden, maar er zijn drastische maatregelen nodig om het volgende te bereiken:

- verbeteren van de gebruiksvriendelijkheid;
- mee schalen met de groei van de data en metadata die gepubliceerd wordt;
- verhogen van samenhang in de data;
- toevoegen moderne visualisatietechnieken;
- toevoegen van archieffunctie.

Als laatste hebben we het over StatLineBeheer, een pakket dat het administratieve proces dat aan publicatie in StatLine ten grondslag ligt ondersteunt. Dit laatste systeem moet meegroeien met de overige systemen.

Er is besloten om het geheel opnieuw te ontwikkelen met behulp van Microsoft .NET, omdat individuele aanpassing van de pakketten minder efficiënt zou zijn. Daarbij wordt ook getracht het StatLine up-to-date te houden met nieuwe technieken. Een van die technieken is visualisatie in grafiekvorm.

Het opvragen van een publicatie

Om een beeld te krijgen bij het selecteren en inzien van publicaties, zijn de volgende twee figuren toegevoegd. Figuur 2.2-1 stelt te StatLine Webselector voor, een gebruiker kan hier onderwerpen uit de verschillende groepen selecteren. Vervolgens kan men de publicatie verder specialiseren door enkel de gewenste categorieën te selecteren.

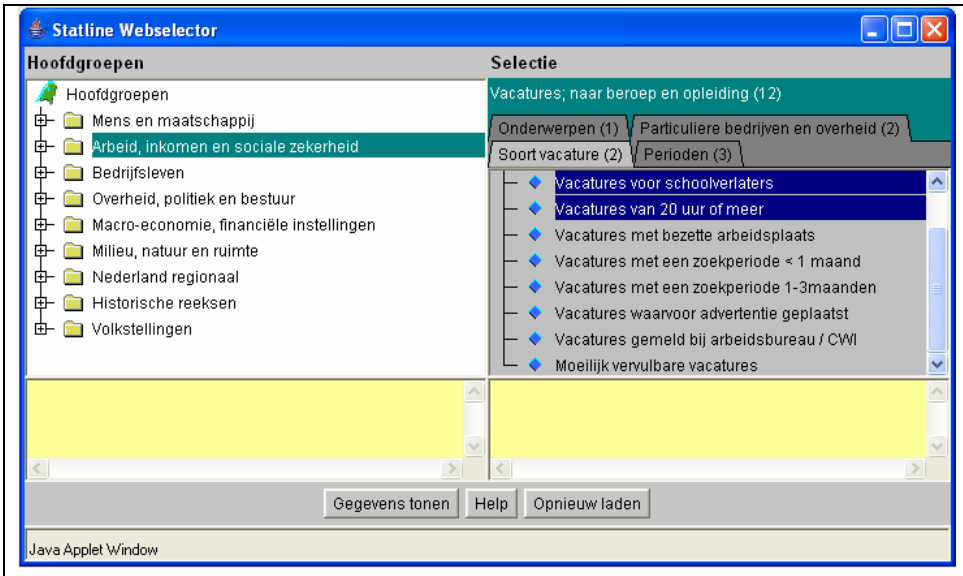


Fig. 2.2-1 Het selecteren van een publicatie.

Wanneer de gebruiker op ‘Gegevens tonen’ druk, wordt er een grid gevuld met de geselecteerde gegevens. Figuur 2.2-1 toont hiervan een voorbeeld.

Vacatures; bedrijfsgrootte, beroep, opleiding

sleep hierheen om in laag te brengen

		Onderwerpen	Vacatures naar opleiding 3-digit
			Totaal aantal vacatures
Particuliere bedrijven en overheid	Soort vacature	Perioden	x 1000
Particuliere bedrijven	Vacatures voor schoolverlaters	2000	77,1
		2001	69,2
		2002	28,0
	Vacatures van 20 uur of meer	2000	137,2
		2001	126,3
		2002	76,8
Overheid	Vacatures voor schoolverlaters	2000	3,9
		2001	4,9
		2002	2,1
	Vacatures van 20 uur of meer	2000	10,5
		2001	12,4
		2002	9,3

Fig. 2.2-2 Voorbeeld weergave, behorende bij figuur 2.2-1

Huidige grafiekmodule

In de StatWeb 3.1 omgeving wordt gebruik gemaakt van de open source component GDChart, om deze aan te passen aan de eisen van het CBS is er een wrapper ontwikkeld. Figuur 2.2-3 is toont een voorbeeld van de huidige weergave. Enkele beperkingen van de gebruikte GDChart zijn:

- geen ondersteuning voor andere lettertypen;
- geen mogelijkheid voor het gebruik van stylesheets;
- niet interactief;
- geen antialias.

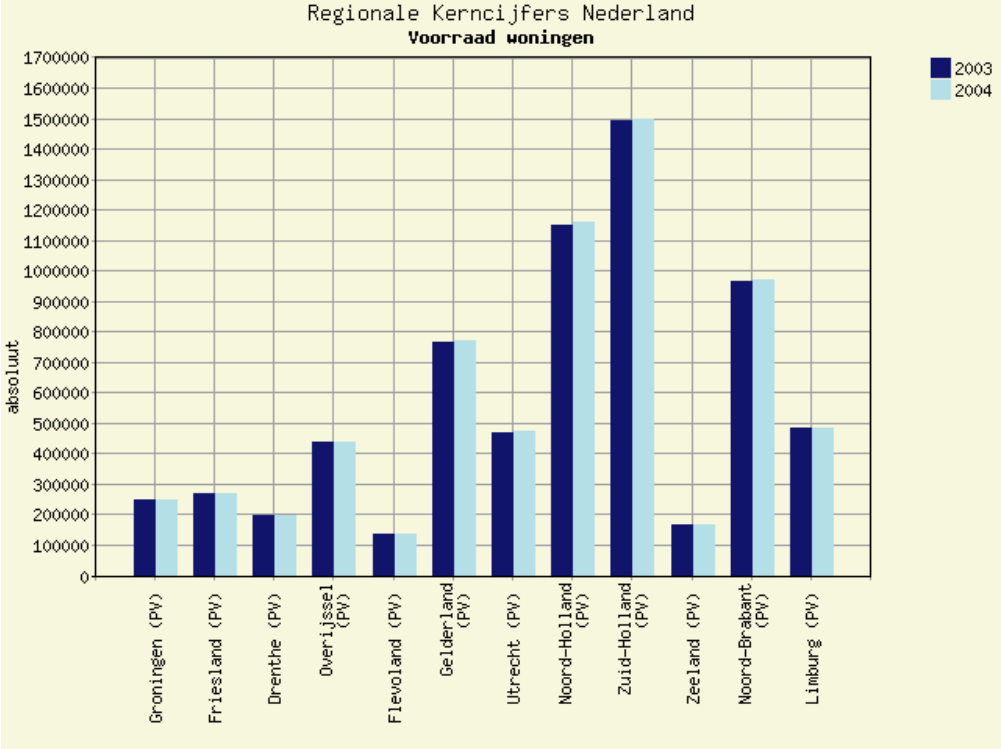


Fig. 2.2-3 Voorbeeld huidige grafiek weergave.

Met de vernieuwingen voor de deur is er gekeken naar een nieuwe vorm van visualisatie, waarbij het gebruik van Scalable Vector Graphics (SVG) onder andere om de volgende reden een goed alternatief leek:

- interactief;
- schaalbaar;
- consequent met de kaartjesmodule (binnen StatLine) die al SVG gebruikt,
- toekomst perspectief van SVG als technologie;
- SVG is open standaard.

In hoofdstuk **Error! Reference source not found.** worden de voor- en nadelen, mede de keuze waarom er voor SVG is gekozen verder uitgewerkt.

2.3 Afstudeeropdracht

Er bestaan drie soorten grafieken:

1. Statische grafieken, dit zijn grafieken zoals ze nu worden gebuikt binnen StatLine. Er wordt een afbeelding gegenereerd en er is geen enkele vorm van interactiviteit.
2. Interactieve grafieken, hierbij kan de gebruiker op afzonderlijke elementen klikken voor extra informatie.
3. Interactieve grafieken, waarbij de gebruiker door de data heen kan navigeren. Hierbij kan bijvoorbeeld worden gedacht aan een grafiek met het aantal inwoners van de verschillende provinciën, waar na het selecteren van een provincie de bijbehorende gemeentes met het aantal inwoners wordt getoond.

Gezien de korte tijdsduur richt ik mij op de grafieken van het tweede soort. Mocht het allemaal erg snel gaan dan kan altijd nog naar het derde soort gekeken worden.

Voor de opdracht moet er gekeken worden naar commerciële producten, deze mogen niet gebruikt worden, maar dienen enkel ter vergelijking. Open source pakketen mogen wel gebruikt worden, indien ze stabiel zijn en gebruik maken van de afgesproken technieken.

De uiteindelijke component dient geschreven te worden in C# en moet Scalable Vector Graphics (SVG) genereren.

2.4 Verwachte resultaten van het project

Als eindresultaat wordt een prototype component verwacht, geschreven in C#. Dit component moet in staat zijn om een grafiek, van het tweede type, te genereren. Het product moet uitbreidbaar zijn zodat deze later de functionaliteit van het derde type bevat. Het ontwerp moet voorbereid zijn op het toevoegen van nieuwe grafiektypes.

3 Plan van aanpak

Gezien de korte tijd (een afstudeerstage van 3 maanden) en het gebruik van onbekende technieken en ontwikkelomgeving, zijn de eisen voor het eindproduct flexibel opgesteld. In eerste instantie richt ik mij op het genereren van een enkele statische grafiek en vanuit daar wordt er uitgebreid naar extra functionaliteiten.

3.1 Beschikbare producten

Als eerst ben ik gaan kijken naar de beschikbare (niet) commerciële producten voor het genereren van grafieken. Belangrijke kernvragen waren hierbij: welke functionaliteiten worden er aangeboden en wat zijn de verschillende weergaven. Er is echter geen gebruik gemaakt van een bestaand pakket, onder andere om de volgende redenen:

- Aanbeveling om gebruik te maken van open standaarden (hoofdstuk **Error! Reference source not found.**);
- Keuze was zeer beperkt als er gekeken wordt naar een combinatie van technieken als C# en SVG;
- De eisen qua aanpasbaarheid waren zo specifiek, dat een te bouwen wrapper zodanig complex zou worden dat zelfbouw een gelijke hoeveelheid werk zou zijn.

3.2 Scalable Vector Graphics

Het gebruik van Scalable Vector Graphics stond vanaf de startdatum al vast. Waarom is er voor SVG gekozen, wat zijn de voor- en nadelen, en is de gewenste interactiviteit te gebruiken in samenwerking met SVG.

3.3 Ontwikkeling prototype

Om mij de te gebruiken technieken eigen te maken heb ik een prototype ontwikkeld. Op deze manier leerde ik de mogelijkheden en beperkingen van SVG en C# kennen zodat ik hier in mijn uiteindelijke ontwerp rekening mee kon houden. Tevens was er tijd om te experimenteren met JavaScript om de verschillende interacties te proberen. Het uiteindelijke product dat hieruit is gekomen was half automatisch gegenereerd, half handmatig. Het voordeel hiervan was wel dat er een zichtbaar product was om over te praten.

3.4 Ontwikkeling eindproduct

Het prototype was statische data weergave waarbij door het gebruik van JavaScript interactie werd gerealiseerd. Het was nu mijn taak om een soortgelijke weergave geheel automatisch te laten genereren door een in C# geschreven module. In eerste instantie zou de module één grafiektype ondersteunen, maar voor het ontwerp moest wel rekening worden gehouden met de uitbreidbaarheid.

3.4.1 Ontwikkeling datamodule

Als eerste onderdeel van het eindproduct is er een datamodule bedacht welke dynamisch kan groeien / krimpen en meerdere dimensies aan kan.

3.4.2 Ontwikkeling engine

Vervolgens is er gekeken hoe deze datamodule het beste uitgelezen kon worden en hoe ik ervoor kon zorgen dat het geheel later gemakkelijk uitbreidbaar zou zijn. Werkzaamheden die hieruit naar voren kwamen zijn onder andere het ontwikkelen van een interface waardoor de verschillende grafiektypes later op een uniforme manier zijn aan te spreken. Tevens zorgt het engine voor het maken, vullen en retourneren van het SVG document. De bijbehorende interactie komt door middel van JavaScript tot stand. Dit werd pas gedaan nadat het tekenen van de grafieken mogelijk was.

3.4.3 Ontwikkeling chart module(s)

Als laatste is er invulling gegeven aan de gedefinieerde interface, genaamd IChart. In eerste instantie zou alleen een BarChart (histogram) worden ontwikkeld, maar om de uitbreidbaarheid aan te tonen is de LineChart (lijndiagram) toegevoegd.

3.4.4 Toevoegen van interactiviteit

Als het mogelijk is om een statische grafiek te genereren, kan worden begonnen aan het toevoegen van een JavaScript bestand. Het JavaScript bestand moet ervoor zorgen dat de gebruiker bij het selecteren van een grafiekonderdeel de bijbehorende data krijgt te zien.

3.5 Ontwikkeling wrapper svgChart

De grafiek module is in eerste instantie zo gebouwd dat deze niet afhankelijk was van de door het CBS gebruikte database/datatype. Uiteindelijk moest de ontwikkelde datamodule wel gevuld worden met deze gegevens en hiervoor moest een wrapper ontwikkeld worden.

4 Uitvoering

4.1 Beschikbare producten

Als we kijken naar beschikbare producten dan kunnen we deze verdelen in Open Source en commerciële producten. Commerciële producten hebben gemiddeld genomen de volgende voordelen:

- meer functionaliteiten;
- support;
- meer grafiektypes;
- ondersteuning .NET.

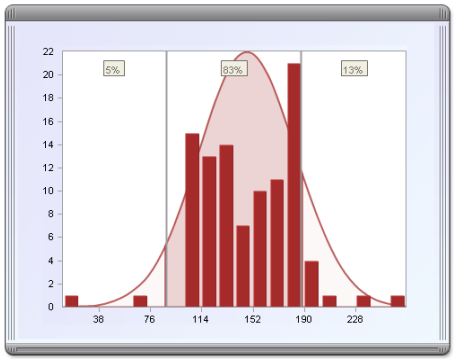
Om verschillende reden welke worden besproken in hoofdstuk **Error! Reference source not found.**, is het geen optie om gebruik te maken van een commercieel product, voor de visualisatie van statistische gegevens. Toch vormen commerciële producten een goede bron van informatie, om te zien wat er mogelijk is en als metafoor om ideeën uit te wisselen met anderen. Commerciële producten die gebruikmaken van .NET en als uitvoer SVG kunnen genereren zijn onder andere:

Software FX Statical

Bedrijf Software FX

Soort Commercieel

http://www.softwarefx.com/

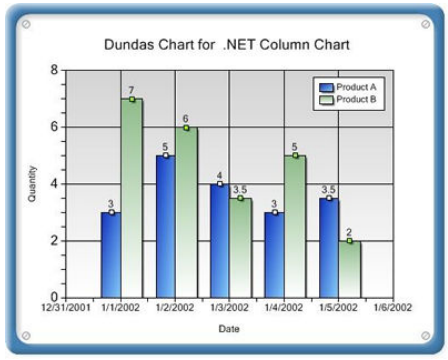


Dundas ChartFX .NET

Bedrijf Dundas DataVisualization

Soort Commercieel

http://www.dundas.com

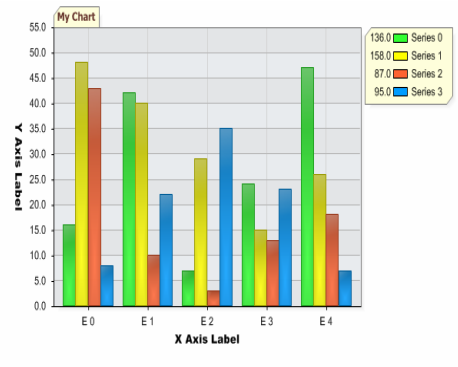


.netCHARTING Enterprice Edition

Bedrijf Corporate Web Solutions Ltd.

Soort Commercieel

http://www.dotnetcharting.com



Open source producten richten zich meestal op het bieden van kern functionaliteiten, terwijl de commerciële tegenhangers vaak doorschieten in het aanbieden van verschillende grafiektypes. De prijs die je hiervoor betaalt varieert van 699 tot 2999 dollar exclusief de benodigde licenties wanneer het product wordt gebruikt op een webserver, zoals voor het CBS van toepassing zou zijn.

4.1.1 Wat is het voordeel van interactiviteit?

We proberen de grafiek zo helder en minimalistisch mogelijk te houden, waarom wordt er dan wel gekeken naar interactiviteit? Interactiviteit kan de gebruiker helpen bij het lezen en/of begrijpen van een grafiek. In de meeste gevallen worden er kleuren gebruikt, om het verband tussen verschillende onderdelen te benadrukken. Interactiviteit kan dit effect nog eens vergroten, tevens kan extra informatie worden getoond, zoals de waarde van een item.

4.2 Scalable Vector Graphics

Als je denkt aan vernieuwing, dan is het nog niet vanzelfsprekend dat je enkel aan het gebruik van SVG denkt. Is het visualiseren met behulp van SVG wel de meest voor de hand liggende oplossing? De keuze voor SVG heeft verschillende oorzaken, welke hierop volgend duidelijk worden gemaakt.

4.2.1 Waarom wordt er voor Scalable Vector Graphics gekozen?

Nederlandse overheidsinstellingen worden aangemoedigd om gebruik te maken van open standaarden. Er is een instelling in het leven geroepen welke open standaarden en open source software keurt en aanbeveelt. Deze instelling gaat door het leven onder de naam Open Standaarden en Open Source Software (OSOSS) [3]. Op de website van het OSOSS is een lijst te vinden met open standaarden die worden aanbevolen voor gebruik. Voor het visualiseren wordt het gebruik van SVG aanbevolen.

Om te voldoen aan de definitie open standaard moet worden voldaan aan de volgende eisen[4]:

- De standaarden worden op basis van een open beslissingsprocedure (consensus of meerderheidsbeslissing, etc.) vastgesteld;
- Het beheer van de standaard ligt bij een not-for-profit organisatie die een volledig vrij toetredingsbeleid kent;
- De standaarden zijn gepubliceerd;
- De kosten voor het gebruik van de standaarden zijn laag en vormen geen drempel voor toegang tot de standaard. Eventueel aanwezig intellectueel eigendom dat aan een open standaard ten grondslag ligt, wordt royalty-free ter beschikking gesteld;
- Er zijn geen beperkende voorwaarden omtrent het hergebruik van een standaard.

Het wordt Nederlandse overheidsinstellingen aanbevolen om open standaarden te gebruiken, maar dit is niet verplicht. De tweede reden heeft te maken met mogelijke toekomstplannen van het StatWeb. Er bestaan plannen om het gehele StatWeb open source te maken. Wanneer gekozen zou worden voor een commercieel product als Macromedia Flash, welke een groot deel van de functionaliteiten van SVG deelt, dan zou dit in strijd zijn met de eisen van een opensource project. Punt negen van de Open Source definitie luid:

9. License Must Not Restrict Other Software

The license must not place restrictions on other software that is distributed along with the licensed software. For example, the license must not insist that all other programs distributed on the same medium must be open-source software. [5]

4.2.2 Scalable Vector Graphics vs. Raster Images

Vector images are based on mathematical formulae and a coordinate system. Therefore, they are not limited according to pixel size, as are bitmapped graphics. Roughly, a vector image takes a series of coordinate points plotted on a graph and contains instructions to the browser on how to render those points as an image, using curves, straight lines, etc. [6]

Voordelen

Raster afbeeldingen verliezen hun resolutie bij vergroting, het gevolg hiervan is een hoekige afbeelding. Willen we de resolutie behouden dan zal de grote van het bestand bijna evenwijdig toenemen met de vergrotingfactor. Omdat vector afbeeldingen zijn opgeslagen als wiskundige formules, zijn ze gemakkelijk te vergroten, zonder enige vorm van kwaliteitsverlies. Vector afbeeldingen zijn daarbij ook nog eens herbruikbaar, wat inhoud dat een object meerdere malen op verschillende plaatsen kan worden geprojecteerd, zonder dat deze noemenswaardige invloed op de bestandsgrote heeft.

Nadelen

Het is nu niet zo dat het gebruik van SVG enkel voordelen heeft en overal voor inzetbaar is. Wanneer men een fotografische afbeelding in SVG probeert weer te geven, dan zal de bestandsgrote groter zijn dan dat van een raster afbeelding welke dezelfde afbeelding weergeeft. Dit omdat er geen tot weinig samenhang bestaat tussen de verschillende vlakken waaruit een fotografische afbeelding is opgebouwd. Dit hoeft overigens geen belemmering te zijn, SVG biedt namelijk de mogelijkheid om raster afbeeldingen te importeren.

Copy-paste

Als we ons inleven in de toekomstige gebruikers van een eventuele SVG visualisatie van statistische gegevens stuiten we op het copy-paste probleem. Het is momenteel nog niet op eenvoudige wijze toegankelijk voor een doorsnee gebruiker om een SVG visualisatie te exporteren naar een afbeelding, om deze vervolgens te gebruiken in een tekst document. Er wordt wel aan gewerkt en er zijn al enkele open source projecten welke een dergelijke export mogelijk maken.

Ondersteuning

Een ander nadeel dat momenteel nog speelt is het gebrek aan browsers welke SVG ondersteunen. Er dient dan ook een plugin te worden geïnstalleerd om SVG syntax te vertalen naar afbeeldingen. De meest bekende is de Adobe SVG Viewer, welke gratis is te downloaden. De meeste bekende open source variant is de Batik SVG Toolkit. CBS heeft gekozen voor het gebruik van de Adobe SVG Viewer, deze is het gemakkelijkst te installeren voor de eindgebruikers. De Batik SVG Toolkit is tevens geschreven in Java. De combinatie van Java en de bestaande componenten van het CBS werken niet optimaal samen en deze combinatie heeft in het verleden dan ook voor verschillenden problemen gezorgd.

4.3 Ontwikkeling prototype

Voor het prototype ontwerp heb ik gekeken wat de twee grafieken (histogram, lijndiagram) voor gezamenlijke eigenschappen hadden. Beide grafieken kennen:

- een background;
- een x-as, y-as en titels;
- een grid, gridlijnen en gridverdeling;
- een chart welke uit items bestaat (balken of cirkels);
- en een legenda.

Door gebruik te maken van overerving , dacht ik een zo efficiënt mogelijk ontwerp te verkrijgen, ‘dacht ik’. In het prototype was het startpunt de Line2DChart of de Bar2DChart, en vanuit hier werd er omhoog gewerkt waarbij de ChartSVG gedurende dit proces een SVG document opbouwt. Boven aangekomen vraagt de ChartBuilder het SVG document van ChartSVG en geeft deze terug. Als laatste zorgt de ASP.NET pagina ervoor dat het SVG document aan de gebruiker wordt getoond.

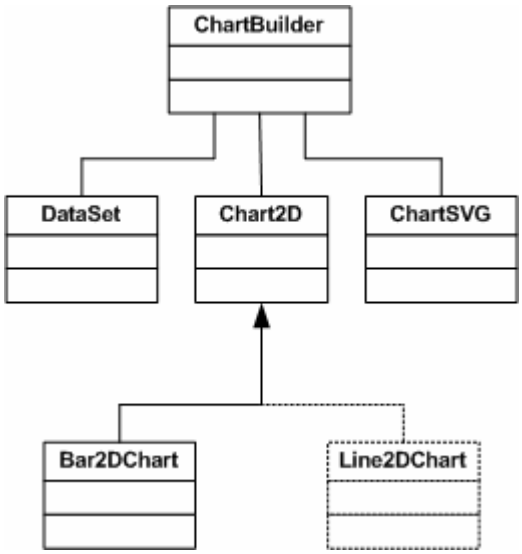


Fig. 4.3-1 prototype ontwerp

Het prototype bleek een goede metafoor om de eisen van het te ontwikkelen product verder te specificeren, maar toonde ook direct de zwakke punten van het ontwerp aan. Zo beperkte het gebruik van meerdere overervingen het aantal te doorlopen paden aanzienlijk, anders gezegd: het kwam de flexibiliteit niet ten goede.

4.4 Ontwikkeling eindproduct

Het semi-automatische gegenereerde prototype moet vervolgens geheel automatische worden gegenereerd. Verder moet het nieuwe ontwerp ervoor zorgen dat het opbouwen van een SVG document flexibeler wordt. De ChartEngine zorgt voor een centrale aansturing, met als input een DataView en als output een SVG document. Dit SVG document wordt geretourneerd naar de webpagina.

Figuur 4.4-1 laat het eindontwerp zien en figuur 4.4-2 is toegevoegd om een beeld te krijgen bij de omvang van de verschillende componenten. In de volgende paragrafen worden de verschillende componenten behandeld.

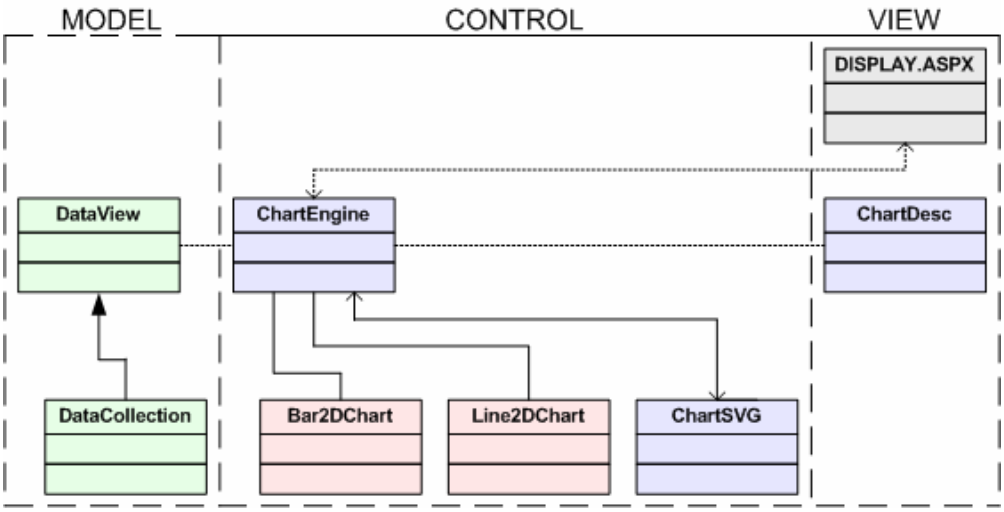


Fig. 4.4-1 Ontwerp eindproduct

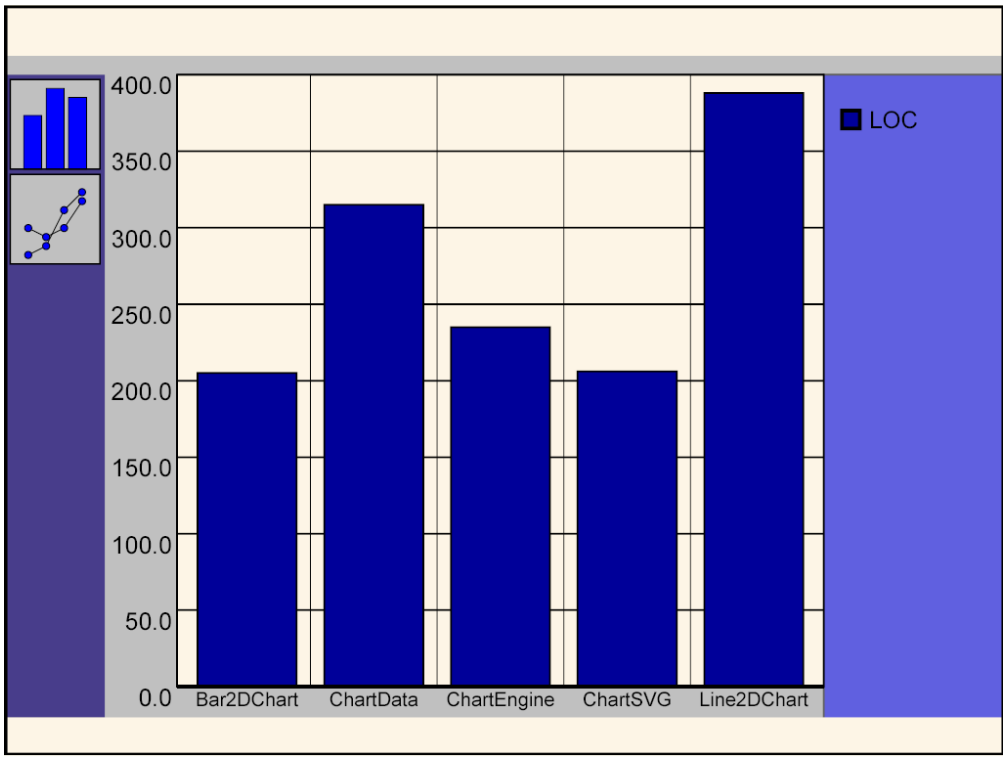


Fig. 4.4-2 Lines of Code per component

4.4.1 Ontwikkeling datamodel

Het ontwikkelen van een eigen datamodel heeft de volgende voordelen:

- Onafhankelijk van type database;
- Onafhankelijk van een database (bv. Statisch data);
- Specifieke functionaliteiten (aantallen, min, max, find, sort, enz.);
- Uitbreidbaar, om in de toekomst navigatie mogelijk te maken.

DataCollection

Een grafiek bestaat uit een verzameling onderwerpen, die op hun beurt weer een waarde bevatten. In een verzameling hebben alle waardes een gemeenschappelijke eenheid. Het laagste niveau van het datamodel wordt DataCollection genoemd. Een DataCollection bevat een tweetal lijsten:

- CaptionList, bedoeld om onderwerp op te slaan;
- ValueList, de bijbehorende waardes van de onderwerpen.

Naast de CaptionList en ValueList heeft elke DataCollection een naam, waarop later gegroepeerd kan worden. Figuur 4.4.1-1 betreft een voorbeeld van een DataCollection.

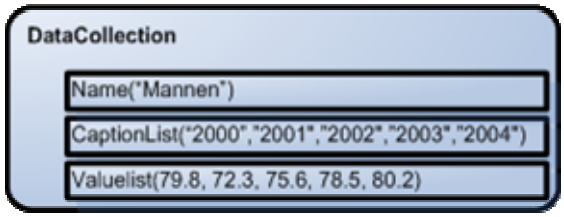


Fig. 4.4.1-1 Voorbeeld van een DataCollection

DataView

Om het mogelijk te maken om meerdere DataCollections weer te geven wordt er een extra laag over de DataCollection gelegd. Een DataView bevat nul of meerdere DataCollections, zoals in figuur 4.4.1-2 is te zien. Tevens zijn er verschillende methodes toegevoegd voor het: toevoegen, verwijderen, sorteren en zoeken van data.

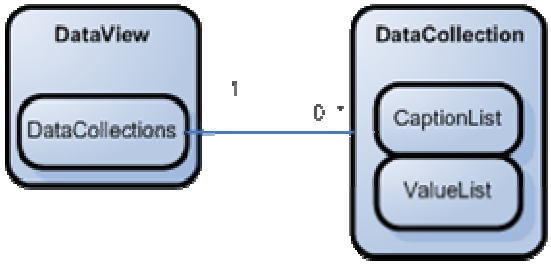


Fig. 4.4.1-2 datamodel

Toekomstige uitbreiding

De gegevens welke door het CBS worden gebruikt kunnen een zekere hiërarchie bezitten. Als voorbeeld nemen we het totaal aantal inwoners van Nederland:

- Totaal Nederland [niveau 0]
    - provincie Groningen [niveau 1]
        - gemeente Appingedam [niveau 2]
        - gemeente Delfzijl [niveau 2]
        - … [niveau 2]
    - provincie Drenthe [niveau 1]
    - ... [niveau 1]

Als eerste wordt een grafiek getoond op het laagste (niveau[0]) waarna er desgewenst kan worden ingezoomd op bijvoorbeeld een provincie (niveau[1]) of bepaalde gemeente (niveau[2]). Om dit te bereiken hoeft men enkel een Child / Parent methode toe te voegen aan de bestaande DataView classe. Als we nog verder gaan dan zou er zelfs een Next / Prev methode toegevoegd kunnen worden voor het scrollen door grafieken op hetzelfde niveau. Bijvoorbeeld het achtereenvolgens tonen van de verschillende gemeentes per provincie.

4.4.2 Ontwikkeling engine

ChartEngine

De ChartEngine is het centrale aanspreekpunt en coördineert het bouwen van het SVG document van create tot return. Doordat er runtime nog kan beslist kan worden wel type grafiek er gebouwd moet worden, wordt er gebruik gemaakt van een factory pattern. Een factory pattern houdt in dat er een interface wordt gedefinieerd, in mijn geval IChart. Er kan voor worden gekozen om een gehele weergave te genereren, of een deel hiervan, waardoor het mogelijk is de weergave te modificeren. Hierdoor kan er bijvoorbeeld voor worden gekozen om de legenda achterwege te laten of verschillende grafieken in een enkele weergave te tonen. Extra informatie dan wel onderbouwing waarom er voor een gemeenschappelijk gedeelde interface is gekozen is hieronder te vinden onder het kopje interface IChart.

Interface IChart

Het prototype ontwerp bevatte een groot aantal overervingen. De verschillende grafieken bevatten veel gedeelde eigenschappen en leek het dus efficiënt om deze door overerving te hergebruiken. Na het ontwikkelen van een prototype bleek dat elke grafiek toch net weer even anders is. Zo kent een LineBar bijvoorbeeld geen verplichte nullijn en moet het bereik (y-as) zo klein mogelijk worden weergegeven. Een BarChart heeft daarentegen weer een verplichte nullijn, meer hierover in de bespreking van de betreffende componenten.

Hoewel het geen vereiste was, zou het via de bovenstaande constructie lastig worden om verschillende grafiektypes in eenzelfde weergave te tonen, omdat er door de overerving iedere keer een nieuwe weergave wordt aangemaakt. Het zelfde probleem zou men ondervinden als men bij een eventuele toekomstige uitbreiding, meerdere weergaven schuin achterelkaar zou willen plaatsen, om zo een weergave met diepte-effect te realiseren.

Om een grote mate van flexibiliteit te verkrijgen moeten de individuele onderdelen van de grafieken afzonderlijk getekend kunnen worden. Hieruit volgend moet het mogelijk worden om run-time te beslissen welk grafiektype er getekend moet worden. Door het gebruik van een Factory Pattern wordt dit mogelijk.

Het Interface genaamd IChart wordt geïmplementeerd in de verschillende grafieken. Door het gebruik van een interface wordt het mogelijk om alle grafieken op een gelijke manier aan te spreken en hoeven we pas run-time te beslissen welke grafiek moet worden gecreëerd.

ChartDesc

Om ervoor te zorgen dat de vormgeving achteraf gemakkelijk is te wijzigen, is er een klasse gemaakt waarin een default vormgeving kan worden ingesteld. Deze klasse genaamd ChartDesc bevat de coördinaten en groottes van de verschillende onderdelen waaruit de weergave is opgebouwd. Daarnaast is de output aan te passen doormiddel van stylesheets, hierdoor zijn de kleuren bijvoorbeeld aan te passen voor slechtzienden.

ChartSVG

Zoals te zien is in figuur 4.4.1-3 bestaat SVG slechts uit een beperkt aantal figuren. ChartSVG is een hulpbestand welke de verschillende elementen en attributen aan de XML boom hangt. Hieronder een kort code fragment om een beeld te geven van de SVG syntax.

```
<g id="buttonLineChartBarChart">
        <rect x="0.0" y="0.0" width="500.0" height="500.0" />
        <line x1="100.0" y1="450.0" x2="200.0" y2="400.0" />
        <circle cx="100.0" cy="450.0" r="20.0" />
</g>
```

Fig. 4.4.1-3 SVG code fragment

4.4.3 Ontwikkeling chart module(s)

Er is begonnen met het ontwikkelen van een histogram, genaamd Bar2DChart. Gegeven dat dit, in vergelijking tot een gestapelde histogram of lijndiagram, de meest eenvoudige weergave is. Bij het ontwerp is wel rekening gehouden met het ondersteunen van andere grafiektypes.

ForcedZero

Een van de verschillen tussen een lijndiagram en een histogram is het bereik. Bij een histogram is dit van de nullijn tot minimaal de grootste waarde, bij een lijndiagram is dit minimaal van de kleinste tot de grootste waarde. Om de functie voor het berekenen van bijvoorbeeld de gridinterval (de afstand tussen de gridlijnen) wordt er een zogehete ForcedZero boolean meegegeven. Afhankelijk hiervan weet de functie of er rekening moet worden gehouden met een nullijn of niet.

4.4.4 Toevoegen van interactiviteit

De code die gebruikt wordt om te reageren op de gebruiker (interactiviteit) staat niet in het SVG document zelf, alhoewel dit wel mogelijk is. Er is een apart JavaScript bestand gemaakt omdat:

- dit gemakkelijker is om te testen;
- het een statische tekst betreft en dus is het onnodig om dit steeds opnieuw te genereren;
- om het geheel overzichtelijk te houden.

Er wordt bij het opbouwen van het SVG document wel rekening gehouden met de interactiviteit, zeker omdat er een enkel JavaScript bestand is voor de verschillende grafiek types. Door bijvoorbeeld de naamgeving en de opbouw van de items binnen een grafiek te standaardiseren. Een voorbeeld hiervan is te zien in figuur 4.4.4-1

<groep + nummer>

  <groep + nummer + item + nummer>

Fig. 4.4.4-1

Het gebruik van een Document Type Definition (DTD) is niet mogelijk door het verschil in elementen en opbouw van het document per grafiektype.

4.5 Ontwikkeling wrapper svgChart

In hoofdstuk **Error! Reference source not found.** wordt het DataModel besproken. Er is gekozen voor een eigen datamodel, maar om het product geschikt te maken voor het CBS is er een wrapper ontwikkeld. In figuur 4.5-1 is te zien wat de plaats van deze wrapper is.

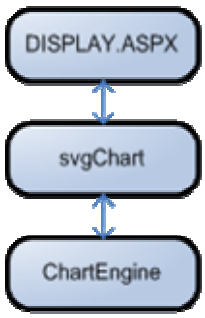


Fig. 4.5-1 Wrapper: svgChart

Hoe gaat het in zijn werk?

Een gebruiker vraagt een standaard of zelf gespecificeerde publicatie, een voorbeeld hiervan is figuur 4.5-2. zo'n publicatie is opgeslagen in een Cube wat te vergelijken is met een multi-dimensionale array. Het aantal dimensies die een Cube bevat, verschilt per publicatie. In dit geval is het aantal dimensies zes, waarbij de afzonderlijke dimensies zijn: Regio's, Motieven, Geslacht, Populatie, Persoonskenmerken en Perioden. Elke dimensie bevat één of meerdere categorieën, bij Populatie is dit er bijvoorbeeld één, genaamd 'Hele bevolking'.

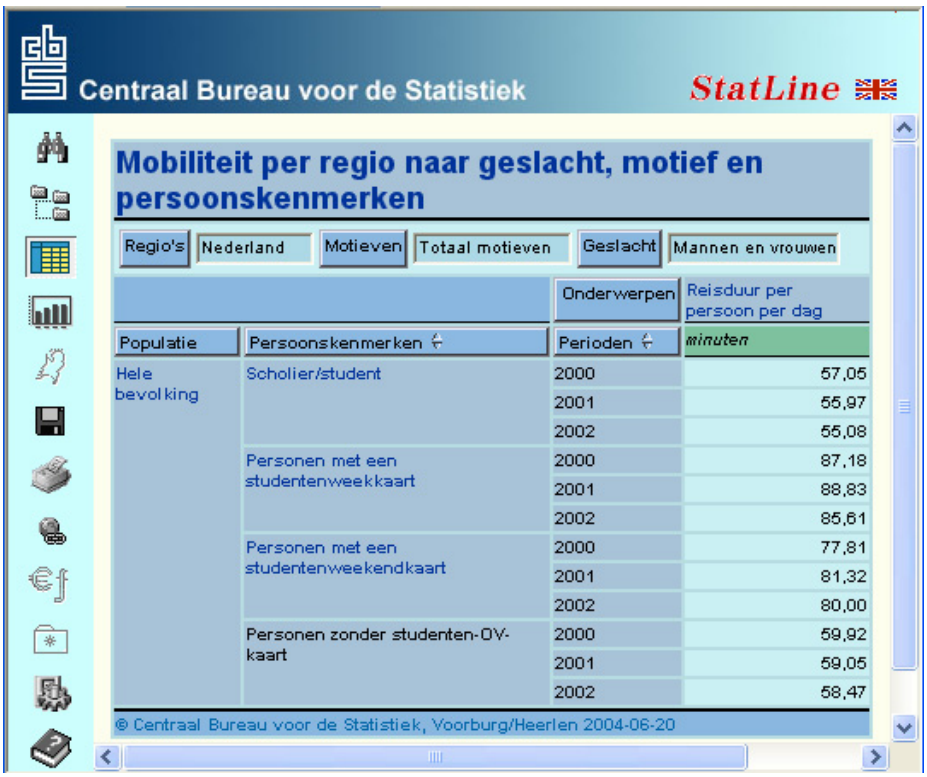


Fig. 4.5-2 Voorbeeld StatLine publicatie

Een tweedimensionale weergave is slechts geschikt om drie dimensies weer te geven. Twee verdeeld over de assen en de derde door middel van het aanbrengen van groepen. Zijn er meer dan drie dimensies, maar bevatten slechts twee dimensies hiervan meer dan één categorie, dan is een weergave nog steeds mogelijk. Door de dimensies met één categorie op te sommen in het onderwerp (zoals ook in het voorbeeld is te zien) is een visualisatie alsnog mogelijk.

De wrapper svgChart, haalt de bruikbare dimensies uit de Cube en vertaald deze naar een DataView. Vervolgens wordt de DataView gebruikt om de ChartEngine een SVG document te laten genereren, welke via svgChart wordt geretourneerd aan de webpagina.

5 Resultaten

Beschikbare producten

Er is onderzoek gedaan naar beschikbare producten op het gebied van datavisualisatie. Niet alleen om het product te hergebruiken maar ook om te zien welke visualisaties populair zijn en de bijbehorende functionaliteiten die worden aangeboden. Uit het onderzoek is geen product gekomen dat geschikt was voor hergebruik, mede doordat de eisen qua techniek te specifiek waren of omdat het product nog niet stabiel was.

Mogelijkheden Scalable Vector Graphics

De mogelijkheden van Scalable Vector Graphics (SVG) zijn onderzocht en vooral of SVG wel de oplossing is voor het visualiseren van data. Of het succes van SVG zo groot wordt als men hoopt kan ik niet zeggen. Doordat het een openstandaard is en de leercurve heel laag is, heeft het zeker potentieel.

Prototype

Om de te gebruiken technieken te verkennen is een prototype ontworpen, deze deed tevens dienst als metafoor. Het prototype werd half automatisch gegenereerd waarna deze handmatig is aangepast om het uiteindelijk doel na te bootsen. Ook is er gekeken in hoeverre en op welke manier de structuur van het document voorbereid moet worden op de toe te voegen interactiviteit.

Eindproduct

Er is een eindproduct welke zowel een histogram als een lijndiagram kan genereren. Ook is het gelukt om interactiviteit toe te voegen om de grafieken leesbaarder te maken. De figuren 5-1 en 5-2 tonen een voorbeeld weergave zoals deze wordt gegenereerd. Het eerste figuur toont tevens een wat er gebeurd als de gebruiker een item selecteert.

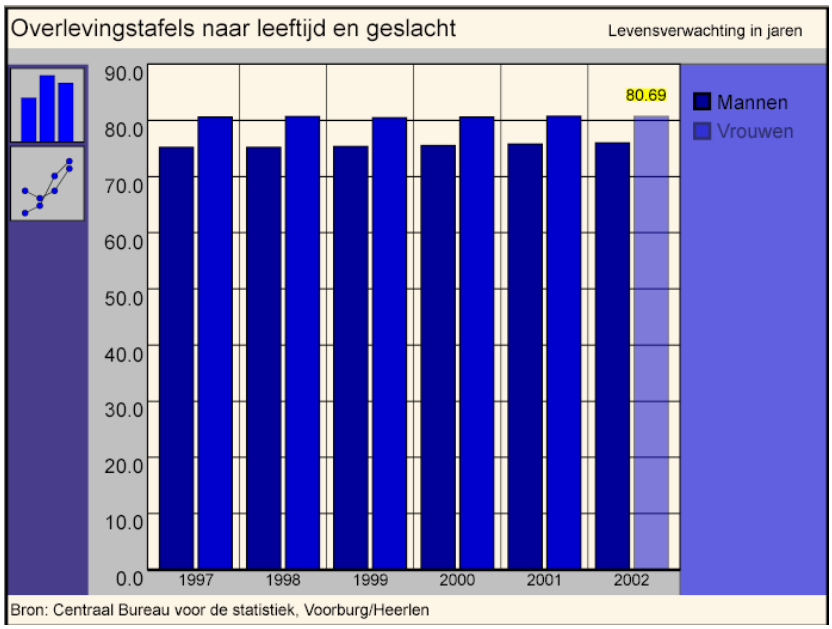


Fig. 5-1 Voorbeeld Bar2DChart

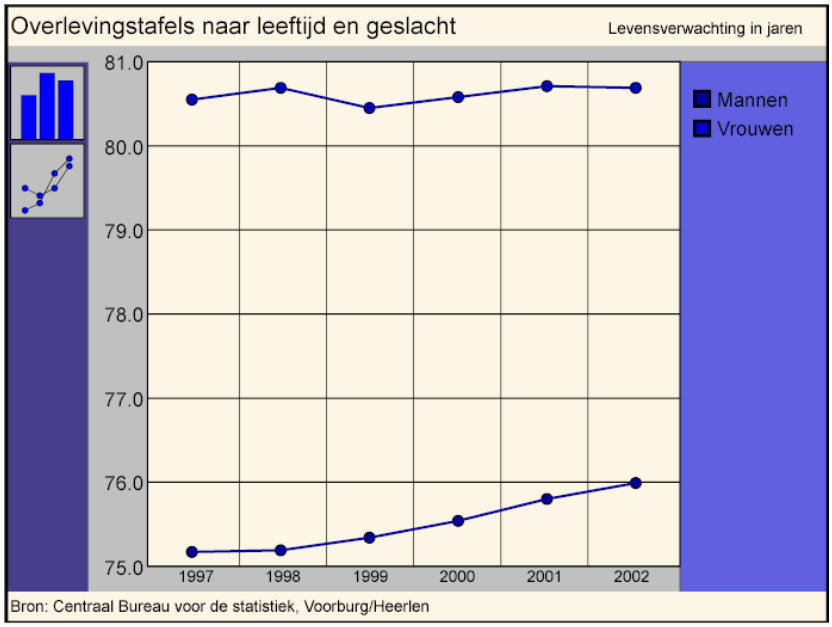


Fig. 5-2 Voorbeeld Line2DChart

Waar staat er nog open

Op het moment van schrijven bieden de grafieken nog niet de functionaliteit om door de data te navigeren. Dit was geen ‘must have’ maar werd wel gezien als een ‘could have’. Tevens is het nog niet mogelijk om de knoppen aan de zijkant te gebruiken, welke ervoor moeten zorgen dat de gebruiker gemakkelijk kan kiezen tussen de verschillende grafiektypen. Dit laatste zal de komende dagen nog worden aangepakt.

6 Evaluatie

6.1 Wat is er bereikt

In de korte tijd van drie maanden ben ik instaat geweest meerdere nieuwe technieken onder de knie te krijgen. Zo had ik nog geen ervaring met: C#, ASP.NET, XML, SVG en JavaScript. Voeg daar aan toe een onbekende ontwikkelomgeving en je hebt een uitdaging. Door de source code van bestaande projecten te inspecteren en het bouwen van een prototype wist ik de leercurve toch nog aanzienlijk te verkleinen en was ik instaat om een werkend product te ontwikkelen.

Er is stabiele module ontwikkeld welke een SVG document kan genereren uit data. Tevens is er een wrapper ontwikkeld voor het converteren van de door het CBS gebruikte Cube naar het datamodel van de module, DataView. Er is gekeken naar de mogelijkheden voor het navigeren door data en hier is in het ontwerp rekening mee gehouden.

Verder zijn de voor en nadelen van SVG onderzocht en is er gekeken naar eventuele alternatieven.

6.2 Wat zou ik anders doen

Als eerste zou ik mijn ontwerp aanpassen en in het bijzonder de datamodule. Achteraf zou het gemakkelijke zijn geweest om als input een XML document te gebruiken, dit zou de volgende voordelen opleveren:

- gebruik van de grafiek generator door derden zou eenvoudiger zijn;
- het tonen van waarden in verband met het ondersteunen van interactiviteit zou worden vereenvoudigd, omdat in het XML document zijn terug te vinden. Momenteel moet er bij het opbouwen van het SVG document rekening meer worden gehouden;
- XML bied meer mogelijkheden.

De kunst van het ontwikkelen zit hem in het nemen van afstand om het ontwerp met een zekere abstractie opnieuw te evalueren. Geregeld heb de neiging iets uit te breiden terwijl een nieuwe aanpak misschien wel beter is voor het ontwerp. Een voorbeeld hiervan is het gebruik van een XML document voor de opslag van gegevens, je beveelt het een medeafstudeerder aan en ziet dan pas dat het voor je eigen ontwerp ook beter zou zijn.

6.3 Hoe is het product ontvangen

Zoals eerder vermeld is het CBS nog druk bezig met het ontwikkelen van StatLine 4.0. Het is de bedoeling dat de door mij geschreven module hierin wordt geïmplementeerd. Momenteel draait mijn module in de test omgeving en na enkele kinderziektes heeft de module een stabiele status verkregen.

6.4 Onderzoeksaanpak

Zoals eerder genoemd in dit hoofdstuk, had ik mijzelf vaker moeten dwingen om afstand te nemen van het ontwikkelen, om het ontwerp te evalueren. Bij aanvang van het project had ik geen gedegen kennis van XML, waardoor ik het ontwerp vertaalde in de door mij beheerste technieken.

Wel zou ik dezelfde iteratieve ontwikkelmethode handhaven, maar de iteraties zouden hierbij kleiner worden. Als het aan mij de keuze was geweest had ik bij een werknemer van het CBS op de kamer willen werken, zodat ook de kleine soms simpele vragen gemakkelijk gesteld konden worden. Daarnaast had ik hierdoor nog meer kennis qua C# en andere technieken kunnen opdoen in een kortere tijd. Al heeft het zelfstandig vergaren van kennis ook zijn voordelen.

Niet geheel ontevreden kijk ik dan ook terug op drie prettige productieve maanden bij het CBS.

Literatuur

1. Centraal Bureau voor de Statistiek

   http://www.cbs.nl/

2. StatLine

   http://statline.cbs.nl

3. Open Standard and Open Source Software

   http://www.ososs.nl

4. Definitie Open Standard

   http://www.ososs.nl/index.jsp?alias=watisos

5. Definition Open Source

   http://opensource.org/docs/def_print.php

6. Introduction to SVG, House, Pearlman, Prenitice Hall PTR, 2003

   http://www.informit.com/articles/printerfriendly.asp?p=99036

Boeken

1. [Eisenberg, 2002] J.D. Eisenberg, *SVG Essentials*, 2002,335p
2. [Robinson, et al. ,2002] Robinson, et al. , *Professional C#, 2nd Edition*, 2002, 1264p.