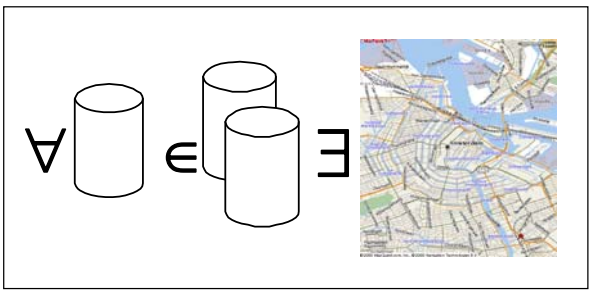
Een kartografisch expert systeem ten behoeve van presentatie van gedistribueerde geografische informatie

Afstudeerscriptie

Auteur:	Ilmar Kotte
Studie:	Technische Informatica, TU Delft
Afstudeer hoogleraar	prof. dr. ir. F.W. Jansen
begeleider IF:	drs. P.R. van Nieuwenhuizen
begeleider GDMC:	drs. M. de Vries
datum:	17 juli 2002

Een kartografisch expert systeem ten behoeve van presentatie van gedistribueerde geografische informatie



Auteur:	Ilmar Kotte
Studie:	Technische Informatica, TU Delft
begeleider IF:	Peter van Nieuwenhuizen
begeleider GDMC:	Marian de Vries
datum:	17 juli 2002

I. Voorwoord

Dit verslag is geschreven in het kader van de afstudeeropdracht horend bij het curriculum van de 4-jarige opleiding van de studie Technische Informatica van de TU Delft.

Het onderwerp van deze scriptie is Kartografische Expertsystemen. Dit onderwerp maakt onderdeel uit van het project "Internet GIS" van het Geo Database Management Centre, het research en development centre voor Geo-informatie technologie van de TU Delft.

Voorbereiding voor uitvoering van deze opdracht bestond onder anderen uit een literatuuronderzoek. Onderzocht is wat de mogelijkheden zijn voor het ontwerp van een kartografisch expertsysteem, en er is een aanzet gegeven tot de bouw van een KES.

Voor het "Internet GIS" project wordt samengewerkt met de Basiseenheid Computer graphics en CAD/CAM (CG) binnen de afdeling Mediamatica (MM) van de faculteit Information Technology and Systems van de TU Delft.

Tot slot een opmerking betreffende de spelling van het woord kartografie en daarvan afgeleide woorden. In het "van Dale Groot Woordenboek Hedendaags Nederlands" worden deze woorden met "c" in plaats van "k" gespeld. Ondanks deze officiële spelling wordt in dit verslag de 'k' spelling aangehouden omdat deze is ingeburgerd binnen het vakgebied van de kartografie, en deze in Nederlandstalige literatuur nog steeds de voorkeur geniet.

Ilmar Kotte, juni 2002

II. Samenvatting

Al sinds het moment dat afbeeldingen in webpagina's kunnen worden opgenomen, is internet een geschikt medium om geografische informatie mee te distribueren. Het is toegankelijk voor veel mensen, de hoeveelheid informatie is eindeloos en de informatie kan gemakkelijk up-to-date worden gehouden.

GIS via internet beperkt zich niet meer tot het oversturen van raster of bitmap plaatjes, maar ook de geografische data zelf kan worden opgevraagd. Dit schept nieuwe mogelijkheden; het is nu bijvoorbeeld mogelijk om gedistribueerde geografische data te integreren tot een enkele kaart. Een topografische ondergrond van een overheidsinstantie kan zo gecombineerd worden met een leidingnetwerk van het energiebedrijf en het kabelnetwerk van een telecommunicatiebedrijf.

In de praktijk verloopt die informatie-integratie niet vlekkeloos. Afgezien van technische moeilijkheden kunnen bij gelijktijdige afbeelding van gegevens uit meerdere onafhankelijke bronnen afbeeldingsproblemen ontstaan. Zo kunnen kleuren conflicteren, de informatiedichtheid in de kaart kan te groot worden en de kaartlaagvolgorde kan verkeerd zijn. Om een consistente en correcte afbeelding te genereren is een Kartografisch Expertsysteem (KES) nodig.

Bestaande systemen die assistantie bij kartografische problemen kunnen geven bleken allen slechts delen van het probleem te bestrijken. Daarom is een ontwerp voor een KES gemaakt. Binnen dit ontwerp is een framework opgesteld om het mogelijk te maken het KES in combinatie met verschillende softwarecomponenten zoals GIS viewers te gebruiken.

Het KES framework is geïmplementeerd bovenop de Java GIS tool GeoTools. Daarna zijn twee versies van een KES engine (het component dat de kartografische kennis bevat) geïmplementeerd. De eerste versie maakt gebruik van een expert system shell voor het toepassen van kartografische kennis. De voornaamste reden om in eerste instantie voor een expert system shell te kiezen was omdat de manier van informatie verwerken goed leek aan te sluiten bij de slecht te formaliseren kartografische kennis die erin opgeslagen moet worden. Deze implementatie bleek tekortkomingen te hebben, daarom werd een tweede KES engine geimplementeerd met een procedurele toepassing van kartografische regels.

De twee implementaties zijn vervolgens geëvalueerd, getest en met elkaar vergeleken. Met de procedurele implementatie bleek de meeste sturing aan het optimalisatieproces te kunnen worden gegeven. De expert system gebaseerde implementatie kan werken met meer impliciet gestelde regels, maar is trager dan de procedurele engine. Conclusies en aanbevelingen zijn naar aanleiding van de bevindingen in dit project geformuleerd en opgenomen in het laatste hoofdstuk.

III. Abstract

Ever since images can be integrated into web pages, the Internet has been a suitable medium for accessing and distributing geographical information. It is accessible to many people, the amount of information is vast and information can be kept current easily.

GIS via internet is no longer limited to transferring bitmap images, the geographical data itself can be retrieved as well. This scenario creates new possibilities to integrate geographical data from different sources into a single map. A topographical background supplied by a governmental department could be instantly combined with a cable network layer from a telecom company.

Integration of information from independent data sources incurs some issues. More fundamental than technical difficulties that may arise, there may be difficulties to produce a consistent map image from the sources. Color conflicts may arise, as well as issues relating to information density and layer-ordering. A Cartographic Expert System (KES) may assist to make a consistent map from different sources.

The functionality of a KES overlaps partly with the design process that a Cartographer follows to create a map. Research shows that attempts to capture and model this process have not succeeded. Despite this, parts of the design process are suitable to include in a KES.

Within this research project an attempt has been made to identify existing systems with KES-like features. Some systems were found, which addressed part of the problems. After this research a requirements analysis and a design was made to create a KES. A framework was designed to assemble a KES from different software components, such as GIS viewer software.

The framework was implemented using the Java GIS tool GeoTools. Using the framework, two versions of a KES engine have been implemented. The first implementation uses an expert system shell to apply the cartographical knowledge; the second one uses a procedural cartographical engine.

The functionality of the KES was tested, evaluated and the two implementations were compared side by side. Conclusions and reccomendations are based on this evaluation.

IV. Inhoudsopgave

1 Inleiding

Al sinds het moment dat afbeeldingen in webpagina's kunnen worden opgenomen, is internet een geschikt medium om geografische informatie mee te distribueren. Het is toegankelijk voor veel mensen, de hoeveelheid informatie is eindeloos en de informatie kan gemakkelijk up-to-date worden gehouden.

GIS via internet beperkt zich niet meer tot het oversturen van raster of bitmap plaatjes, maar ook de geografische data zelf kan worden opgevraagd. Dit schept nieuwe mogelijkheden; het is nu bijvoorbeeld mogelijk om gedistribueerde geografische data uit diverse onafhankelijke bronnen te integreren tot een enkele kaart. Een topografische ondergrond van een overheidsinstantie kan zo online gecombineerd worden met een leidingnetwerk van het energiebedrijf en het kabelnetwerk van een telecommunicatiebedrijf.

In de praktijk verloopt die informatie-integratie niet vlekkeloos. Afgezien van technische moeilijkheden kunnen bij gelijktijdige afbeelding van gegevens uit meerdere onafhankelijke bronnen afbeeldingsproblemen ontstaan. Zo kunnen kleuren conflicteren, de informatiedichtheid in de kaart kan te groot worden en de kaartlaagvolgorde kan verkeerd zijn. Om automatisch een consistente en correcte afbeelding te genereren is een Kartografisch expert systeem (KES) nodig. In deze scriptie wordt aandacht besteed aan de eisen en wensen aan zo'n KES, er wordt bekeken of er software bestaat die aan die eisen tegemoet komt, en er wordt een nieuw KES ontworpen en geimplementeerd.

1.1 Leeswijzer

Hoofdstuk 2 is een introductie in een notendop op het gebied van GIS, kartografie en het gebruik van beide op internet. De scheiding tussen gegevens en weergave bij GIS toepassingen wordt toegelicht, alsmede open standaarden die daarop betrekking hebben. Voor lezers met uitgebreide kennis op deze gebieden is dit hoofdstuk optioneel.

In Hoofdstuk 3 gaan we nader in op het Kartografisch Expertsysteem concept. Er wordt een definitie gegeven, er worden mogelijke toepassingen genoemd, en er wordt onderzocht wat er op dit gebied al beschikbaar is aan KES achtige systemen.

Hoofdstuk 4 bestaat uit een requirements analyse, een functioneel ontwerp en een technisch ontwerp voor het KES. Een objectmodel wordt voorgesteld en een GIS component wordt geselecteerd.

Hoofdstuk 5 gaat over de daadwerkelijke implementatie van een KES. Er wordt uiteengezet hoe het KES is opgebouwd rond een framework, uit welke componenten het bestaat en hoe ze samenwerken. Twee alternatieve redenatie mechanismen worden geïmplementeerd.

In Hoofdstuk 6 testen en evalueren we het KES. Nagegaan wordt of het voldoet aan de verwachtingen. Er wordt ingegaan op de voor en nadelen van de twee KES engine implementaties.

Met hoofdstuk 7 wordt het verslag afgesloten; hier worden conclusies en aanbevelingen besproken.

In de bijlagen is aanvullende informatie te vinden die te gedetailleerd was om in de tekst op te nemen.

2 Internet GIS

2.1 Introductie GIS

Kartografie en GIS (Geografische Informatie Systemen) zijn tegenwoordig onlosmakelijk met elkaar verbonden. Een GIS systeem is een geautomatiseerd systeem voor het opslaan, controleren, integreren, manipuleren, analyseren en weergeven van geografische informatie. Kartografie omvat het toegankelijk en hanteerbaar maken van ruimtelijke informatie door middel van kaarten [Schans 99].

Een kartograaf gebruikt zijn kennis en ervaring om een afbeelding van de (hypothetische) werkelijkheid te maken, die eenmaal geïnterpreteerd door een waarnemer uit de beoogde doelgroep, de juiste informatie oplevert voor de waarnemer.

De opkomst van GIS systemen heeft ervoor gezorgd dat de scheiding tussen gegevens en presentatie van die gegevens mogelijk werd. Meetgegevens zoals bijvoorbeeld bodemsoort of perceelafmetingen zijn opgeslagen in een database in een GIS, en kunnen met behulp van de GIS software op diverse manieren tot afbeelding worden verwerkt. Zo kan worden gekozen om de grenzen tussen bodemlagen te tekenen, of om slechts een enkele bodemsoort weer te geven met behulp van gekleurde vlakken.

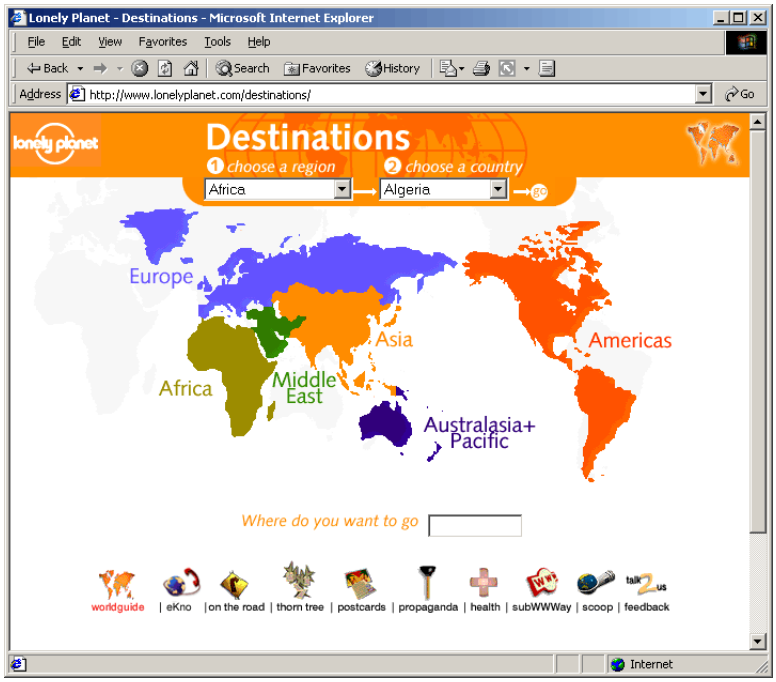
Deze scheiding tussen objectieve, intrinsieke gegevens en de weergave ervan is bij GIS via internet belangrijk, omdat door deze scheiding een gebruiker zelf de wijze van afbeelden kan kiezen, afhankelijk van het doel dat hij met de kaart heeft. Dezelfde dataset kan zo gebruikt worden voor meerdere doelen (Zie paragraaf 2.4).

2.2 Geografische informatie op internet

Internet als medium is zeer geschikt om Geografische informatie mee te distribueren. Het is toegankelijk voor veel mensen, de hoeveelheid informatie is eindeloos en de informatie kan gemakkelijk up-to-date worden gehouden. Het kan een grote rol spelen in de geo-informatievoorziening van een organisatie. In de volgende paragraaf gaan we kort in op de historie van Geo-informatie op internet.

2.3 Historie

Afbeeldingen op webpagina's, waaronder afbeeldingen van kaartmateriaal, zijn pas gemeengoed geworden sinds Marc Andreessen in 1993 de `<IMG>` tag introduceerde in de door hem ontwikkelde Mosaic internet browser. Hoewel deze uitbreiding van de tot op dat moment op tekst gebaseerde HTML pagina's mogelijkheden schiep voor het publiceren van Geografische Informatie, waren de mogelijkheden zeer beperkt. De plaatjes moesten door geringe beschikbare bandbreedte vaak beperkt worden in omvang, er moesten concessies met betrekking tot kleurpalet worden gedaan, en interactie met de plaatjes door middel van scripttalen of applets was nog niet mogelijk. Bezoekers van pagina's met dergelijke informatie konden bekijken, niet manipuleren. De eigenschappen (schaal, kleur, detailnivo etc.) zijn eenmalig bepaald. Als de gebruiker er al iets mee kan doen, dan is het in- of uitzoomen en/of pannen. Voor sommige toepassingen is dat overigens meer dan genoeg of zelfs teveel, zoals bijvoorbeeld als het kaartje wordt gebruikt als index naar informatie.



Figuur 2.1 Gebruik van een kaart als informatie-index

Tegenwoordig zijn er voor veel beperkingen oplossingen voorhanden. Alle moderne internetbrowsers ondersteunen ECMAscript (javascript), diverse afbeeldingsformaten, en hebben ondersteuning voor de inbedding van objecten zoals plugins of applets in pagina's. De kleurenbeperking van 8-bits kleuren die veel personal computers van begin jaren negentig hadden, is tegenwoordig ook verdwenen.

Toch zijn er weer nieuwe uitdagingen op het gebied van het afbeelden van kaartmateriaal afkomstig van internet, omdat er steeds meer verschillende soorten apparaten (afbeeldingsplatforms) toegang krijgen tot internet. PC's, GSM telefoon met WAP of I-mode functionaliteit en Personal Digital Assistants (organizers) zijn in principe geschikt.

Aan de server kant is ook grote diversiteit. Software van verschillende fabrikanten, elk met eigen standaarden, beletten eenvoudige integratie van kaartmateriaal uit verschillende bronnen.

2.4 Scheiding DLM en DKM

In het pre-digitale tijdperk bestond geospatiële informatie voornamelijk uit kaartmateriaal. Het kaartmateriaal *was* de informatie over een stuk landschap. Deze manier van informatieopslag had als nadeel dat de informatie die in een kaart was afgebeeld, niet meer eenvoudig voor andere soorten kaarten te gebruiken was. De afbeelding wordt tenslotte vaak gemaakt met een bepaald doel voor ogen; een bepaald publiek met een bepaalde informatiebehoefte.

Geografische informatiesystemen brachten hier verandering in. De geospatiële informatie kon onafhankelijk van afbeelding worden opgeslagen. Semanticus Alfred Korzybski gebruikte een analogie, voor mentale beelden van de werkelijkheid, die in dit verband letterlijk genomen kan worden: "The map is not the territory it represents, but, if correct, it has a similar structure to the territory, which accounts for its usefulness....".

De scheiding tussen inhoud en weergave van geospatiële informatie levert voordelen op.

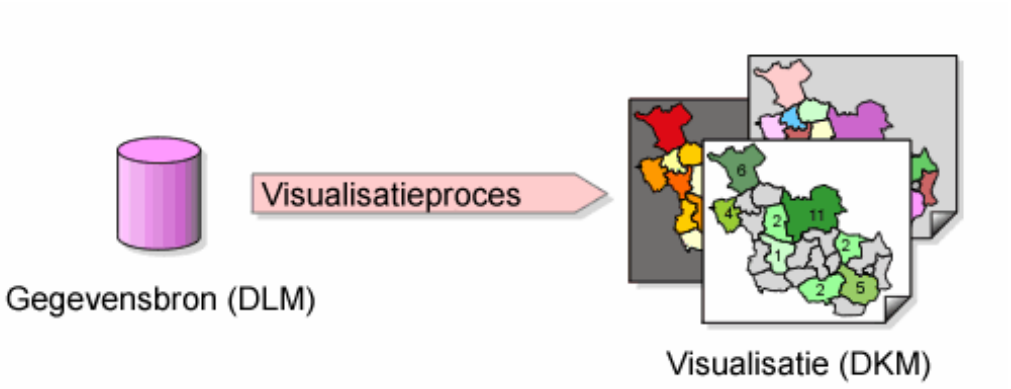
- Uit een gegevensbron kunnen verschillende afbeeldingen voor verschillende toepassingen worden gemaakt.
- De gegevensbron kan veel meer informatie bevatten (meer detail, een derde en zelfs vierde dimensie, etc) dan daadwerkelijk wordt afgebeeld.

Deze zienswijze op de scheiding tussen inhoud en weergave wordt ook wel het LM/VM paradigma genoemd [Schans 1999], waarbij LM voor landschapsmodel en VM voor visualisatiemodel staat.

> "Een landschaps- of terreinmodel (LM) beschrijft
> een landschap en wat daarin voorkomt en gebeurt
> als (dynamische) terreinobjecten, met
> terreingeometrie en terreinattributen (thematiek),
> geheel ongeacht hun eventuele grafische weergave.
> Het vormt de gedachteninhoud."

> "Een visualisatie- of kaartmodel (VM) beschrijft
> een kaart of ander visualisatiemedium als (dynamische)
> grafische beeldobjecten (stippen, strepen,
> vlekken, letters), met beeldgeometrie en
> kleuren of grijswaarden, ongeacht wat het allemaal
> voorstelt. Het is de uitdrukkingsvorm."

In het vervolg van deze scriptie zullen we in plaats van LM/VM de termen DLM en DKM gebruiken, die staan voor Digitaal Landschaps Model en Digitaal Kartografisch Model. Het DLM legt de geometrische eigenschappen van het landschap vast. Het DKM specificeert de grafische elementen waarmee een stuk landschap wordt gevisualiseerd.



Figuur 2.2 Uit een enkele bron kunnen meerdere visualisaties worden gemaakt (aangepast van [Kraak 2001])

2.5 DLM: geografische data

Het Digitale Landschaps Model is de benaming van de intrinsieke gegevens van een terreinbeschrijving, losstaand van afbeelding. Middels het DLM worden bijvoorbeeld afmetingen van een perceel vastgelegd, of de locatie van een terreinobject.

Het DLM bestaat uit "rauwe" data die moet worden afgebeeld om door mensen geïnterpreteerd te kunnen worden.

2.5.1 DLM standaarden

Het OpenGIS consortium heeft zich tot doel gesteld om open standaarden te ontwikkelen voor GIS toepassingen. Een van die standaarden is GML (Geography Markup Language), die bedoeld is om DLM-gegevens zoals features, attribuutinformatie en meta-gegevens vast te leggen en te communiceren. Het is een XML-grammatica, hetgeen ervoor zorgt dat data gecodeerd in GML gemakkelijk, in tekstvorm, zijn te transporteren over een internetverbinding. Een nadeel hiervan is de grotere omvang van gegevens die getransporteerd moet worden, door de extra tekstuele GML-tags en door het versturen van numerieke data in tekstuele vorm.

Uit de GML 2.0 recommendation [Cox 2001] over scheiding tussen data en presentatie:

> "GML has been designed to uphold the principle of separating content from presentation. GML provides mechanisms for the encoding of geographic feature data without regard to how the data may be presented to a human reader. Since GML is an XML application, it can be readily styled into a variety of presentation formats including vector and raster graphics, tekst, sound and voice. Generation of graphical output such as maps is one of the most common presentations of GML."

Op het moment is versie 2.0 van de standaard nog een recommendation. Desondanks is er al serversoftware die voldoet aan de OpenGIS Web Feature Server specification, en die dus features kan leveren in GML 2 formaat (Ionic software, zie [URL 2]).

Aangezien de meeste GIS fabrikanten zijn aangesloten bij OpenGIS, is de verwachting dat de standaard in de nabije toekomst door deze fabrikanten geïmplementeerd wordt.

2.6 DKM: afbeelding

Het afbeelden van gegevens uit het DLM wordt gedaan middels het DKM. Entiteiten binnen dit model zijn bijvoorbeeld lijnstukken en polygonen, als afbeelding van DLM entiteiten als wegen en percelen.

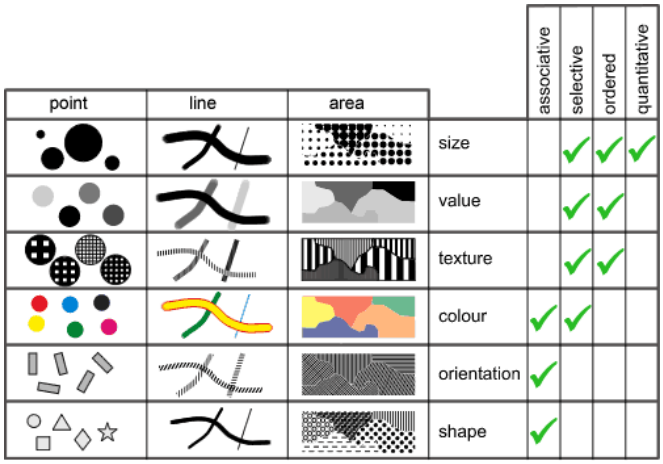
In de kartografie worden de begrippen *kaartsymbool* en *grafische variabele* gebruikt als het over kaartafbeeldingen gaat. Hieronder de definities uit het Kartografisch Woordenboek:

> **grafische variabelen – KW 4.1.12**
> "Elementaire aspekten op grond waarvan symbolen van elkaar worden onderscheiden.
> Opm.: Door de Franse kartograaf Bertin worden de volgende grafische variabelen onderscheiden: positie, vorm, richting, kleur, grein, grijswaarde en grootte."
> **(kaart)symbool – KW 4.5.1**
> Teken ter aanduiding of onderscheiding van een object of verschijnsel in de kaart. Te onderscheiden in (1) puntsymbolen, (2) lijnsymbolen en vlaksymbolen.
> Opm.: Voor dit begrip gebruikt men ook de termen (kaart)teken of (kaart)signatuur.

Een voorbeeld van een kaartsymbool is een polygoon dat een rivier afbeeldt. De grafische variabele 'kleur' zou in dit geval de waarde 'blauw' kunnen hebben.

2.6.1 Grafische variabelen

In 1967 heeft de kartograaf Jaques Bertin acht variabelen beschreven die toegepast kunnen worden om kaartsymbolen van elkaar te onderscheiden. Twee van de acht zijn voor locatie (x,y), de overige zes staan in onderstaande figuur. Dit zijn: grootte, lichtheid (van kleur), tekstuur, kleur, oriëntatie en vorm.



Figuur 2.3 Bertin's grafische variabelen

Grafische variabelen zijn in meer of mindere mate geschikt om kwantitatieve gegevens, geordende gegevens of nominale (kwalitatieve) gegevens af te beelden. Ze zijn stuk voor stuk koppelbaar aan een of meerdere representatieschalen.

Representatie-schalen:

Nominaal	gegevens op deze schaalsoort zijn onderscheidbaar van elkaar, maar niet geordend.
Ordinaal	Gegevens zijn geordend
Interval	Gegevens zijn geordend, en hebben een bepaalde afstand tot elkaar
Ratio	Gegevens zijn proportioneel ten opzichte van een nulpunt.

Gebruik van de grafische variabelen induceert een bepaalde reactie, of gedrag bij de waarnemer. Ze kunnen associatief, selectief, ordenend of kwantitatief werken.

Associatief
In hoeverre bewerkstelligt een variabele een binding of associatie tussen elementen met dezelfde eigenschappen? Vorm, oriëntatie, kleur en tekstuur zijn associatieve variabelen. Grootte en lichtheid vallen hier niet onder.

Selectief
De mogelijkheid om onderscheid tussen groepen te maken. Waar ligt een bepaalde categorie? Welke groepering onderscheidt zich van de rest?

Ordenend
De variabelen lichtheid, grootte en tekstuur kunnen een visuele, relatieve ordening aangeven.

Kwantitatief
Alleen grootte is een kwantitatieve variabele.

Er zijn sinds de introductie van Bertin's variabelen een aantal aanvullingen gedaan. Zo stelde Morrison (zie [MacEachren 1995:272]) in1974 voor om de variabele *kleur* te scheiden in *kleurwaarde* (hue) en *verzadiging* (saturation). Hij voegde ook de variabele *rangschikking* toe. MacEachren voegt daar nog drie variabelen voor verschillende soorten "helderheid" (clarity) aan toe: *hardheid* (crispness) voor harde grenzen of zachte overgangen tussen vlakken, *resolutie*

(resolution) of mate van gedetailleerdheid van raster of vectorafbeeldingen, en *transparantie* (transparancy).

Met het dynamische afbeeldingsmedia zoals een computerscherm kan *animatie* als dertiende variabele worden onderscheiden.

Grafische variabelen bepalen de ruimte waarbinnen gemanoeuvreerd kan worden om een DLM object af te beelden in een DKM, en zijn derhalve relevant om te beschouwen bij het ontwerpen van een KES.

2.6.2 DKM Standaarden

Voor het DKM zijn er een aantal mogelijke open standaarden, naast de proprietary (fabrikanteigen) standaarden. Het doel van het DKM is vastleggen van de grafische primitieven die de kaartafbeelding vormen.

Een van de standaarden om het DKM in vast te leggen is VRML (Virtual reality modelling language). Deze standaard is vooral geschikt voor 3D gegevens. Hoewel GIS gegevens in drie dimensies gerepresenteerd kunnen worden, kijken we bij dit onderzoek naar mogelijkheden om 2D afbeeldingen te maken. In dit varband is een meer geschikte kandidaat SVG (Scalable Vector Graphics), een standaards die is toegesneden op vastlegging van 2D gegevens, tekstuele informatie en allerlei grafische attributen zoals transparantie en animatie.

2.6.3 SVG

SVG staat voor Scalable Vector Graphics, en is een World Widw Web Consortium (W3C) standaard, die is gedefinieerd als XML-grammatica. SVG heeft de mogelijkheid om 2D grafische objecten te specificeren, zoals lijnstukken en vlakken. Verder is het mogelijk om animatie van deze objecten te bewerkstelligen, interactiviteit toe te voegen en om tekst op te nemen. In een SVG bestand kan zowel vector als bitmap informatie worden opgenomen.

Voor het bekijken van SVG-documenten is een SVG-viewer nodig. Dit kan een losstaande applicatie zijn, of in het geval van internet GIS een browser -plugin. Omdat SVG een open standaard is, is software van diverse fabrikanten beschikbaar, soms zelfs in open-source vorm, dus inclusief broncode.

Een nadeel van SVG is dat, zoals bij andere XML documenten, de omvang van een SVG document door de tekstuele vorm vrij groot kan worden. Als een SVG document via een HTTP 1.1 connectie wordt verstuurd, zoals gebeurd bij het opvragen van een SVG document met een moderne webbrowser, dan bestaat de mogelijkheid om het SVG document met het Gzip algoritme transparant voor de ontvanger te comprimeren. Dit kan SVG documenten met een compressieratio van 75% tot 85% kleiner maken [SVG 2001].

2.7 Metadata

Metadata is een term die gebruikt wordt voor data die andere data beschrijft. Het kan dienen voor het zoeken naar gegevensbronnen, identificatie, beheer, administratie, beschrijving en locatie van datasets. Voorbeelden van metadata elementen zijn:

- de eigenaar van de dataset
- de kwaliteit (o.a. nauwkeurigheid) van de dataset
- prijsinformatie voor gebruik van de dataset

Metadata kan een potentiele bron voor styling informatie zijn, omdat het semantische informatie over de dataset kan bevatten. Een metadata tag zou bijvoorbeeld kunnen aangeven dat een dataset voor een kaartlaag een collectie waterwegen bevat. Als voor de kaartlaag geen styling informatie beschikbaar is, kan interpretatie van de metadata door een rendering engine een 'educated guess' voor afbeelding in een blauwtint kunnen doen.

Voor het vastleggen van metadata zijn diverse standaarden (bijv. ISO/TC211, en FDGC), die echter nog niet wijdverbreid worden ingezet.

2.8 Afbeelding van DLM op DKM

Het KES onderzoeksproject concentreert zich op een correcte afbeelding van kaartlagen uit verschillende DLM's in een enkel DKM. De transformatie van het DLM tot een kaartafbeelding kan op vele manieren gebeuren.

Factoren die bij het afbeeldingsproces (ook "styling" genoemd) een rol spelen zijn:

- beschikbare mogelijkheden tot afbeelding van hardware en/of software (aantal kleuren, resolutie etc.)
- doel van de kaart (toeristisch, specialistisch etc)
- formaat waarin DLM beschikbaar is, en het formaat waarin het DKM beschikbaar moet komen
- afbeeldmedium (printen/plotten, beeldscherm etc)
- conventies horend bij een bepaald toepassingsgebied

Afhankelijk van de implementatie van het KES en de gebruikte standaarden, zijn de volgende mechanismen voor afbeelding denkbaar:

- Als het DLM in een XML grammatica (zie paragraaf 2.9 Open standaarden) wordt aangeleverd, en het DKM ook in XML formaat kan worden gemaakt, dan is tranformatie door middel van XSL (eXtensible Stylesheet Language) en XSLT een optie. In deze taal is het mogelijk om een transformatie te definieren tussen verschillende typen XML documenten. Een voorbeeld hiervan zou zijn de transformatie van features in GML naar grafische elementen in SVG. Een nadeel is dat XSL geen bewerkingen zoals berekeningen op de features in het DLM toestaat, en dat er dus beperkingen zijn bij bepalen van een optimale afbeelding.
- Als gebruik wordt gemaakt van een GIS server die de WMS (Web map server) standaard implementeert, dan kan met behulp van de nog in ontwikkeling zijnde SLD (Styled Layer Descriptor) standaard een styling-voorschrift worden meegegeven met een Request aan de server. Zonder gebruik van SLD voorschiften kan een WMS slechts voorgedefinieerde (op de server) styles toepassen.

```
http://yourfavoritesite.com/WMS?
VERSION=1.0.5&
REQUEST=GetMap&
SRS=EPSG%3A4326&
BBOX=0.0,0.0,1.0,1.0&
SLD=http%3A%2F%2Fmyclientsite.com%2FmySLD.xml&
WIDTH=400&
HEIGHT=400&
FORMAT=PNG
```

Figuur 2.4 Voorbeeld van een WMS request met SLD styling

- Ontwikkelen van speciale software voor de DLM naar DKM transformatie. Voor optimale controle over het 'symboliseren' van kaartfeatures, en in het geval dat gegevens in verschillende bronformaten worden aangeleverd geeft deze optie de meeste mogelijkheden.

2.9 Open standaarden

In het recente verleden domineerden de fabrikant-eigen formaten de GIS wereld. Voor de fabrikanten was dit een manier van klantenbinding, voor gebruikers vaak een bron van problemen. Bestanden moesten geconverteerd worden, een handeling die vaak met verlies aan informatie, precisie en met introductie van fouten gepaard ging. Ook voor de distributie van geospatiële informatie zijn deze gesloten standaarden niet behulpzaam.

Het OpenGIS Consortium (OGC) is opgericht in 1994 om te zorgen voor ontwikkeling van open standaarden die zorgen voor een betere toegankelijkheid van geospatiële informatie. Deelnemers in het consortium zijn onder anderen organisaties met een sleutelrol in de GIS wereld, zoals de grote GIS fabrikanten, hetgeen de daadwerkelijke implementatie van de standaarden garandeert. Hoewel het OpenGIS Consortium al langer bestaat dan de XML standaard, heeft de ontwikkeling van OpenGIS standaarden een groei gezien door de XML specificatie. Hoewel XML niet een specifieke GIS standaard is, is het zeker een relevante standaard, omdat het de basis vormt voor een aantal andere standaarden zoals GML en SVG.

2.9.1 XML: de basis

EXtensible Markup Language is een taaldefinitie waarmee gegevens op gestructureerde wijze vastgelegd kunnen worden. Doelen bij de ontwikkeling van XML waren (vrij naar [XML 2000]):

- XML moet gemakkelijk bruikbaar zijn op internet
- XML moet een breed scala aan applicaties ondersteunen
- XML moet gemakkelijk te gebruiken zijn in applicaties
- XML moet een formele, bondige specificatie worden

Een XML document is enerzijds een doodgewoon tekstbestand, dat te lezen en bewerken is met een teksteditor. Middels zogenaamde markup in de vorm van tags (labels), wordt in het tekstdocument de informatie gestructureerd.

Anderzijds, en veel belangrijker, is een XML document een representatie van een hiërarchie van objecten met attributen. De klassen van de objecten en van de hiërarchie als geheel kunnen worden vastgelegd in een los document waardoor eenduidig kan worden vastgesteld of een XML document voldoet aan de specificatie van die klassen.

XML is een meta-taal, waarmee, afhankelijk van de beoogde toepassing, allerhande grammatica kunnen worden opgesteld. Het vastleggen van een grammatica gebeurt door het gebruik van een Document Type Declaration (DTD), of met behulp van de XML Schema taal. Deze laatste mogelijkheid is de krachtiger en meest flexibeler, dan het gebruik van DTD's. Met XML Schema kunnen complexe datatypes worden gedefinieerd, en restricties (constraints) worden opgelegd aan elementen en attributen.

```
<xsd:schema xmlns:xsd="http://www.w3.org/2000/10/XMLSchema">
	<xsd:annotation>
		<xsd:documentation xml:lang="en">
		Voorbeeld Schema voor een kaart klasse
   De klasse Kaart kan nul of meerdere themas bevatten
   auteur: Ilmar Kotte
</xsd:documentation>
	</xsd:annotation>
	<xsd:element name="Kaart" type="KaartType"/>
	<xsd:element name="comment" type="xsd:string"/>
	<xsd:complexType name="KaartType">
		<xsd:sequence>
			<xsd:element ref="comment" minOccurs="0"/>
			<xsd:element name="themas" type="ThemasType"/>
		</xsd:sequence>
		<xsd:attribute name="LastModified" type="xsd:date"/>
	</xsd:complexType>
	<xsd:complexType name="ThemasType">
		<xsd:sequence>
			<xsd:element name="Thema" minOccurs="0" maxOccurs="unbounded">
				<xsd:complexType>
					<xsd:sequence>
						<xsd:element name="ThemaId" type="xsd:decimal"/>
						<xsd:element name="ThemaNaam" type="xsd:string"/>
						<xsd:element ref="comment" minOccurs="0"/>
					</xsd:sequence>
				</xsd:complexType>
			</xsd:element>
		</xsd:sequence>
	</xsd:complexType>
</xsd:schema>
```

Figuur 2.5 Voorbeeld van een schema *definitie voor een kaartstructuur*

```
<?xml version="1.0" encoding="UTF-8"?>
<Kaart xmlns:xsi="http://www.w3.org/2000/10/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="D:\My documents\Studie\Afstuderen\Werkmap\kaart.xsd">
	<comment>ThemaSet Delft</comment>
	<themas>
		<Thema>
			<ThemaId>1</ThemaId>
			<ThemaNaam>Topografische ondergrond Delft</ThemaNaam>
		</Thema>
		<Thema>
			<ThemaId>2</ThemaId>
			<ThemaNaam>Wegennet Delft</ThemaNaam>
		</Thema>
	</themas>
</Kaart>
```

Figuur 2.6 Voorbeeld van een XML document gebaseerd op bovenstaand schema

De Geography Markup Language (GML) is een XML grammatica (XML applicatie) waarin geografische features kunnen worden opgeslagen. Bovenstaande voorbeelden hebben overigens niets met GML te maken, hoewel beide grammatica betrekking hebben op het vastleggen van geografische informatie.

Belangrijk kenmerk van XML is dat het niet een enkel bestandsformaat is, maar een meta-formaat, waarin zeer diverse gegevens kunnen worden vastgelegd.

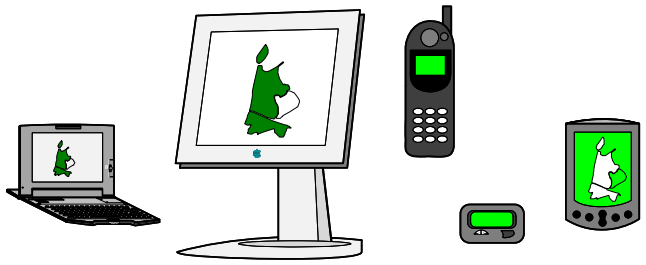
2.9.2 Server standaarden

In het kader van het Web Mapping Testbed project zijn er door het OpenGIS Consortium een aantal standaarden opgesteld die betrekking hebben op het opvragen van geografische informatie via internet. Web Map Server (WMS) is een standaard die een interface beschrijft tussen een GIS client en een WMS-compliant GIS server. Een dergelijke server levert kaarten in de vorm van bitmaps, dat wil zeggen dat de gegevens op de server zijn gerenderd. Deze standaard is niet geschikt voor gebruik aan de invoerkant van een KES, omdat het DLM niet los beschikbaar is.

Web Feature Server (WFS) is een standaard die geschikt is voor het opvragen van de kaartfeatures in intrinsieke vorm, in plaats van in de vorm van een gegenereerde afbeelding. Een GIS server die de WFS specificatie implementeert kan dus DLM informatie van kaartelementen retourneren. Deze standaard is bruikbaar aan de invoerkant van een KES.

2.10 Toekomst Internet GIS

Door het te verwachten grotere aanbod van gestandaardiseerde DLM informatie op internet wordt een scenario waarin internetgebruikers zelf kaarten samenstellen met data uit diverse bronnen steeds realistischer. Hoewel deze data waarschijnlijk qua dekkingsgebied gerelateerd is, (de gebruiker wil het immers in 1 kaart tonen) zijn de afbeeldingsattributen waarschijnlijk ongerelateerd en misschien zelfs conflicterend. Voeg daarbij het toenemend aantal platforms dat toegang biedt tot internet, elk met eigen afbeeldingsmogelijkheden en beperkingen, en het wordt duidelijk dat gespecialiseerde kartografische software in dit geval zal kunnen assisteren bij het maken van kaartafbeeldingen.



Figuur 2.7 Nieuwe toegangsplatformen tot (Geografische) informatie op internet

De mogelijkheden om geografische data online op commerciele basis aan te bieden zijn met het volwassen worden van internet toegenomen. De mogelijkheden tot authenticatie van gebruikers en aanbieders, het tot stand kunnen brengen van goed beveiligde verbindingen en de toegenomen datatransfer snelheden zijn instrumenten die het direct online beschikbaar stellen van datasets mogelijk maken.

3 Kartografisch Expert Systeem

3.1 Introductie

Na de inleiding over GIS, Internet GIS en open standaarden gaan we in de volgende hoofdstukken kijken naar het kernbegrip binnen dit project: het Kartografisch Expert Systeem oftewel KES.

In dit hoofdstuk worden achtereenvolgens besproken:

- Definitie KES: *wat is een KES? Voor wie heeft een KES nut?*
- Toepassingen: *Waarvoor kan een KES worden ingezet?*
- Literatuuronderzoek naar Kartografische Expertsystemen: *Is er eerder onderzoek gedaan?*

3.2 Definitie en Doel

De definitie van een kartografisch expertsysteem zoals hier wordt bedoeld is als volgt: [Rengelink 1999]:

> Een systeem dat opgebouwd is uit regels en functies die een gebruiker (liefst) ongemerkt begeleidt bij het maken van bruikbare kaarten. Bruikbaar wil hier zeggen dat de kaarten op juiste wijze kunnen worden geïnterpreteerd, zodat ze beslissingsondersteunend kunnen werken.

Het hoofddoel van een KES onder deze definitie is het behoeden van gebruikers voor een kaartweergave die door foutieve grafische attributen niet juist te interpreteren is.

De voornaamste funkties van een KES zoals hier wordt besproken zijn dan ook:

- Automatische assistentie bij het samenstellen van kartografisch correcte kaarten.
- Optimalisatie van de weergave in het geval dat gegevens afkomstig van ongerelateerde (wellicht gedistribueerde) databronnen tot afbeeldingsconflicten leiden.

3.3 Doelgroep

Door het assisterende karakter van een KES horen ervaren GIS gebruikers niet tot de primaire doelgroep voor een KES. Een internet gebruiker met ervaring met GIS systemen en/of kartografische kennis zal in het algemeen weinig problemen hebben om de weergave van geselecteerde kaartlagen aan te passen tot een correcte kaart. Toch kunnen ook zij baat hebben bij een optimalisatie door een KES, bijvoorbeeld als uitgangspunt voor het maken van een kaart. Er zullen ook ervaren GIS gebruikers zijn die interventie van het KES bij het afbeelden van kaartlagen eerder als hinderlijk dan als behulpzaam ondervinden. Tot de doelgroep voor een KES zijn te rekenen:

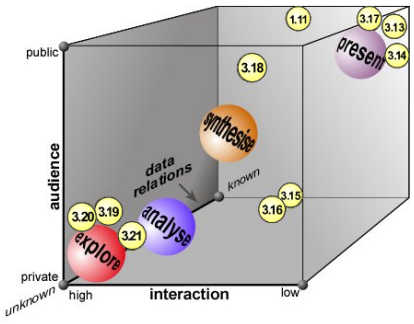
- Gebruikers met minder ervaring op het gebied van (internet) GIS, die zelfstandig een kaart willen samenstellen.
- Ervaren GIS gebruikers die snel een 'uitgangssituatie' voor hun werk willen maken, op basis waarvan ze aanpassingen kunnen maken.

Er is in [Kraak 2001] onderzoek gedaan naar de doelen en doelgroepen van kaartgebruik. Door Kraak wordt in dit verband de "map use cube" besproken, die de ruimte representeert van de verschillende gebruiken van GIS en kaarten. De ruimte wordt opgespannen door de assen:

- Interactiviteit (laag-hoog)
- Relatie tussen data (onbekend-bekend)
- Het doel van de kaart (prive gebruik of publicatie)

Vier hoofdgebieden worden binnen de kubus onderscheiden en benoemd:

1. verkennen/ontdekken (explore)
2. analyseren (analyse)
3. samenstellen (synthesise)
4. presentatie (present)



Een KES zal vooral behulpzaam kunnen zijn bij het samenstellen (synthesise) van een kaart uit lagen die van verschillende bronnen afkomstig zijn. Vaak zal zo'n kaartje voor privé gebruik worden gemaakt, met weinig interactie en met bekende datarelaties.

3.4 Mogelijke toepassingen

Het voornaamste toepassingsgebied van een KES is op Internet, door de grote beschikbaarheid van geografische data van diverse bronnen. De volgende mogelijkheden zijn denkbaar voor toepassing van een KES:

Hulp bij het snel samenstellen van een (preview) kaartje
Bijvoorbeeld als tool op de website van een cataloging service [URL 11] waar het KES zou kunnen dienen als preview instrument om data uit verschillende bronnen bij elkaar te zoeken.

Aanvulling op de functionaliteit van GIS software
Een andere toepassing zou kunnen zijn om KES functionaliteit in te bouwen in een GIS webbrowser plugin, of zelfs in GIS desktop pakketten. Een plugin zou op verzoek een geoptimaliseerd kaartbeeld kunnen maken.

Automatische generatie van kaartbeelden
Bij het automatische genereren van grote hoeveelheden kaarten met weinig of geen menselijke interactie, bijvoorbeeld om een set kant-en-klare bitmaps voor een website te genereren.

Het maken van platform-specifieke afbeeldingen
Door de toegenomen mogelijkheden om informatie te benaderen is het denkbaar dat een kaartafbeelding die goed interpreteerbaar moet zijn op een kleine PDA (Personal Digital Assistant) minder details kan weergeven dan een 21 inch desktop monitor. Een KES zou kunnen helpen om ondanks de beperkingen een zo goed mogelijk kaartje voor het platform te maken.

3.5 Bestaande Kartografische Expertsystemen

Uit het literatuuronderzoek dat vooraf ging aan het maken van dit verslag kwam naar voren dat er nog geen KES bestaat in de zin van de hier gehanteerde definitie. Er waren wel een aantal systemen die raakvlakken hebben met de omschrijving van een KES zoals in dit hoofdstuk is voorgesteld. In de volgende paragrafen bespreken we ze kort.

3.5.1 Symboolselectie

In [Zhan 1995] wordt een frame-based kennismodel voorgesteld ten behoeve van kartografische symboolselectie. Met symbool wordt hier bedoeld een symbool uit het domein van de kartografie, zoals een punt, lijn of polygoon. Het systeem is in staat om, met name voor thematische kaarten en statistische kaarten, aanbevelingen te doen voor de weergave ervan. De gebruiker dient door middel van beantwoording van vragen zijn intentie aan het systeem op te geven.

Voordelen van dit frame based schema zijn, vrij naar [Zhan 1995]:

- Overerving
  - Een symbool kan van een superclass eigenschappen erven
- Flexibiliteit
  - in het modelleren van relaties tussen objecten en variabelen door is-a en has-a relaties.
- Herbruikbaarheid
  - Uitbreiding van het model kan modulair gebeuren

Het systeem is vrij academisch in de zin dat de gebruiker eerst een lijst vragen moet beantwoorden, waarna het systeem in tekstuele vorm aanbevelingen doet voor weergave.

3.5.2 Label plaatsing

MapInfo heeft onder de naam MapLex een product op de markt dat tekstuele informatie in kaarten helpt plaatsen. Het handmatig plaatsen van labels bij kaartsymbolen is arbeidsintensief werk. De MapLex software automatiseert deze handelingen voor een deel, en zorgt er onder anderen voor dat:

- labels niet overlappen
- labels eenduidig bij een feature of symbool horen
- label tekst automatisch in grootte wordt aangepast aan het symbool waarbij het hoort. Zo kan een label dat in een polygon wordt geplaatst zodanig worden geschaald dat het er geheel binnen valt.

Binnen MapLex kunnen regels worden gedefinieerd die de automatische label plaatsing sturen. MapLex is niet zozeer als hulp bedoeld bij interactief bekijken of samenstellen van kaarten, maar meer als kartografische hulp bij productie van (hardcopy) kaarten.

3.5.3 Kleurconflicten

Er zijn twee voorbeelden van Kartografische Expertsystemen gevonden die kleurconflicten als probleemgebied hebben.

In [Rengelink 1999] wordt een eerste stap gezet tot een KES. Dit systeem kan kleurconflicten detecteren binnen een desktop GIS pakket. Het programma detecteert het conflict, het oplossen ervan moet handmatig gebeuren.

Een aanzet tot een KES wordt ook gegeven in [Alkemade 2000]. Zij paste een bestaande GIS viewer aan zodat deze kaartlagen met gelijke kleuren kan detecteren, en van een nieuwe kleur kan voorzien. Dit KES is beperkt tot het oplossen van kleurconflicten tussen kaartlagen met lijnsymbolen en is hard-coded in een GIS-viewer.

3.5.4 Descartes

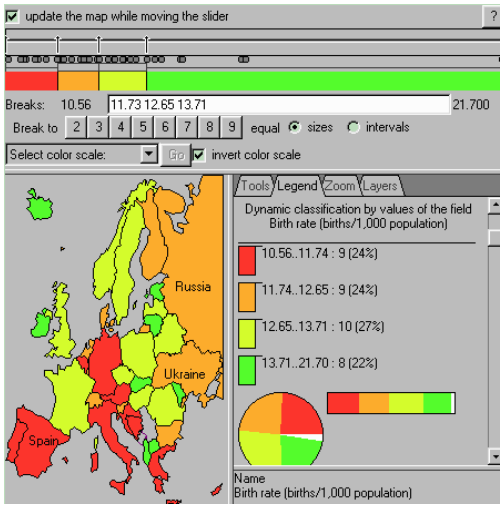
Het Descartes-systeem is ontwikkeld door het Duitse instituut GDM. Het is een systeem dat in staat is een gebruiker te begeleiden bij het afbeelden van statistische gegevens in een kaart, met name in grafiekvorm.

De gebruiker selecteert uit tabellen de attributen die hij gevisualiseerd wil zien, en Descartes kan vervolgens op basis van die selectie een of meerdere suggesties doen voor de presentatie ervan. Zo kan Descartes suggereren om een taartdiagram te gebruiken voor de weergave van het aantal mannen en vrouwen in een regio, waarbij de oppervlakte van het diagram het totaal aantal inwoners voorstelt [Voss 1999].

De presentatie-voorstellen die uit het systeem komen worden afgeleid uit een tweetal bronnen:

- een verzameling domein-onafhankelijke regels die kennis over thematische kartografie representeren. Deze regels zijn niet aanpasbaar voor een gebruiker.
- metadata van de betreffende gegevenssets, die met de bij Descartes horende "Application Builder" software kan worden aangepast.

Het interessante aan Descartes is dat het er in slaagt om tot op zekere hoogte generieke kennis over thematische kartografie toe te passen op door de gebruiker gekozen datasets. Het feit dat Descartes in verschillende grote projecten (o.a. CommonGIS, zie [Andrienko 1999]) is ingezet bewijst de toepasbaarheid van kennisgebaseerde ondersteuning in een GIS omgeving.



Figuur 3.1Descartes gebruikt voor statistische analyse

4 KES ontwerp

4.1 Introductie

Omdat gebleken is dat er nog geen compleet Kartografisch Expert Systeem bestaat, is de doelstelling om een KES te ontwikkelen. Door middel van het uitvoeren van testscenarios kunnen we dan onderzoeken of een KES in de praktijk nuttig kan zijn.

We doorlopen bij de ontwikkeling de volgende stappen:

- Requirements analyse: *duidelijk vastleggen wat er verwacht wordt van een KES*
- Functioneel ontwerp: *hoe wordt voldaan aan de requirements*
- Technisch ontwerp: *hoe wordt het functioneel ontwerp gerealiseerd*
- Technologie selectie *wat te gebruiken voor constructie van een KES, welke standaarden te kiezen?*
- Componenten selectie *zijn er bruikbare standaardcomponenten die aan de eisen voldoen?*
- Eerste test met de componenten *zijn de componenten in principe geschikt?*
- Implementatie: *het ontwikkelen van de software*
- Testen en evaluatie: *voldoet het aan de eisen en verwachtingen? Hoe kan het verbeterd worden?*

De eerste drie punten worden in de volgende paragrafen nader uitgewerkt.

4.2 KES Requirements

De hoofddoelgroep voor inzet van een KES bestaat uit onervaren GIS gebruikers die snel een interpreteerbare kaart willen genereren uit ongerelateerde informatiebronnen.

Een Kartografisch expertsysteem levert een dienst aan een eindgebruiker. Deze dienst dient erop gericht te zijn dat de gebruiker voordeel heeft van de werking van het systeem, bijvoorbeeld door snelheidswinst, of door het scheppen van extra mogelijkheden door beschikbaar komen van expertise die de gebruiker zelf niet heeft.

De requirements voor een KES zijn:

1. Het moet kartografische kennis kunnen toepassen om een correct kaartbeeld te maken.
2. Het moet regels kunnen uitvoeren met betrekking tot het genereren van een kartografisch correcte kaartafbeelding.
3. De werking moet zoveel mogelijk automatisch en buiten het zicht van de eindgebruiker plaatsvinden.
4. Functionaliteit die gemakkelijk uit te breiden is. Gedacht kan worden aan het toevoegen van regels aan het KES.
5. Functionaliteit die gemakkelijk is te koppelen aan andere software, bijvoorbeeld een andere GIS viewer.
6. Inbedding van het KES in een testomgeving die experimenteren met de functionaliteit mogelijk maakt.

Aanvullende wensen voor een KES zijn:

1. Het moet gebruikers kunnen waarschuwen voor problemen in het kaartbeeld die het KES niet kan oplossen.
2. Het KES zou rekening kunnen houden met de beperkingen en mogelijkheden van het platform waarvoor een afbeelding wordt samengesteld.

4.2.1 Kartografische Probleemklassen

Een KES moet kartografische kennis gaan toepassen bij het maken van een kaartafbeelding. Hieronder volgt een overzicht van mogelijke problemen die zouden kunnen optreden bij het combineren van kaartlagen, en een indicatie van de actie die het KES zou kunnen nemen. De algoritmes worden in paragraaf 4.3.2 inhoudelijk toegelicht.

Kleurconflicten

Probleem:	Er is sprake van een kleurconflict als symbolen (bijv. polygonen) die van elkaar te onderscheiden moeten zijn, kleuren hebben die gelijk zijn, of te dicht bijelkaar liggen.
Oplossingen:	Als er alternatieve (voorkeurs)kleuren beschikbaar zijn, kies een niet-conflicterende kleur. Pas anders een van de conflicterende kleuren aan.

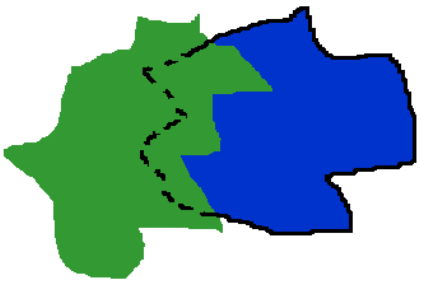
Afdekking

Probleem: Kaartlagen met een kleine dekking kunnen worden afgedekt door kaartlagen met een grotere dekking. In topologische termen: de features van laag A *contain* de features van laag B.

Oplossingen: Door het aanpassen van de kaartlaagvolgorde kan een kaartlaag met laag dekkingspercentage weer bovenop komen. Ook is een oplossing mogelijk door andere grafische variabelen te kiezen, zoals bij overlapping (zie beneden).

Overlapping

Probleem: Twee of meer kaartlagen met grote dekking kunnen elkaar geheel of gedeeltelijk overlappen

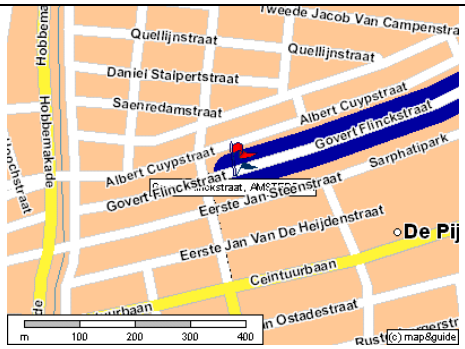


Oplossingen: Aanpassen van kaartlaagvolgorde helpt niet, de lagen blijven elkaar gedeeltelijk overlappen. Een oplossing kan zijn om andere grafische variabelen te kiezen om onderscheid te maken tussen de lagen. Zo kan de bovenste laag transparant of gearceerd gemaakt worden, zodat de onderste erdoorheen te zien is.

Labelplaatsing

Probleem:

Tekstuele labels bij kaartsymbolen kunnen verkeerd geplaatst zijn, waardoor features of andere labels onzichtbaar worden. Ook kunnen labels door verkeerde plaatsing hun associatie met de feature die ze beschrijven verliezen.



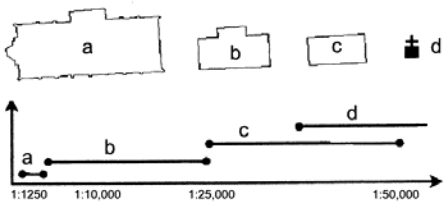
Oplossingen:

Voor dit probleemgebied is geen eenvoudige oplossing. Het probleem kan worden opgelost met een complexe verzameling algoritmes. Een aanwijzing dat het in elk geval gedeeltelijk mogelijk is om software bij dit proces te laten assisteren is het softwarepakket MapLex van MapInfo (zie paragraaf 3.5.2), dat speciaal ontwikkeld werd voor assistentie bij labelplaatsing.

Generalisatie

Probleem:

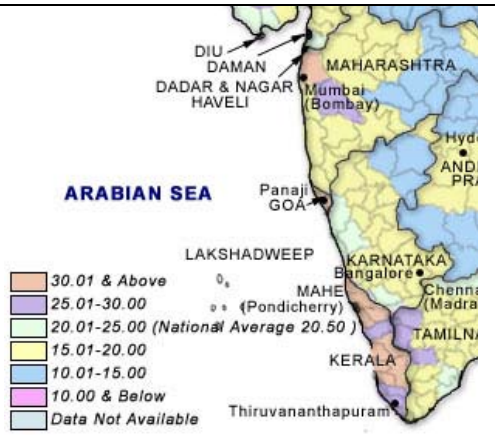
Bij verschillende kaartschalen is het nuttig om verschillende symbolisatie toe te passen. De exacte contouren van een kerkgebouw op een wijkkaart van Zwolle zijn op landelijke schaal niet meer relevant.



Oplossingen:

Kaartlagen kunnen een minimale en maximale schaal meekrijgen waarop ze relevantie hebben. Buiten dit bereik kan dan bijvoorbeeld dezelfde (kant en klare) kaartlaag in een ander detailniveau worden gekozen. Dit wordt ook wel passieve generalisatie genoemd. Er is sprake van actieve generalisatie als een systeem in staat is om uit een enkele basis dataset nieuwe datasets af te leiden. Een complexe vlakfeature kan door actieve generalisatie worden gegeneraliseerd tot een eenvoudiger vlakfeature, of een punt, of zelfs helemaal worden weggelaten.

Grafische attribuut keuze	
Probleem:	Bij thematische kaarten moeten gekozen grafische attributen (zie 2.6.1) passen bij de grootheid en de schaalsoort die ze afbeelden. De kaart hiernaast illustreert een verkeerde attribuutkeuze gedaan door een grootheid met een geordende, ordinale schaal af te beelden op een ongerelateerde verzameling kleuren, waardoor kleur voor kleur moet worden opgezocht in de legenda om de kaart te interpreteren.
Oplossingen:	Binnen een GIS moet specifiek worden aangegeven of de attribuutdata waar een thematische kaartlaag op is gebaseerd, moet worden onderverdeeld in categorieën of klassen. De aanwezigheid van zo'n classificatie kan door een KES worden gedetecteerd, en als er een ordering in de classificatie aanwezig is, maar niet in de kleurentoewijzing, kan een waarschuwing worden gegeven, of een aanpassing.worden gedaan aan de kleuren.



Niet alle genoemde problemen kunnen door een KES worden opgelost. In het functioneel ontwerp concentreren we ons op een subset die voor een eerste opzet haalbaar lijkt.

4.3 Functioneel ontwerp

De requirements bepalen dat een KES kartografische kennis moet kunnen toepassen. Hiervoor moeten stukken kartografische kennis worden geformaliseerd. In dit onderzoek spitst zich dat toe op het oplossen van de volgende probleemgebieden:

- Kleurconflicten tussen kaartlagen oplossen
- Meest logische volgorde van kaartlagen bepalen
- Voorkomen van zichtbaarheidproblemen door afdekking van lagen door andere lagen

4.3.1 Model definitie en aannamen

Ten behoeve van het ontwerpen en implementeren wordt hier een afbakening van het domein waarbinnen een KES gaat functioneren gegeven.

De volgende aannamen en definities gelden:

- Een kaartthema, of kaartlaag, bestaat uit features van één enkel type. Een laag bestaat dus slechts uit vlakfeatures of uit lijnfeatures of uit puntfeatures. Een feature kan bijvoorbeeld een weg representeren, of een perceel.
- Alle features in een kaartlaag hebben dezelfde afbeeldingsattributen, zoals kleur.

- het KES kan vooralsnog niet beschikken over semantische betekenis van gegevens, slechts over de inhoud. Meta-informatie is niet beschikbaar.
- Topologische operatoren zijn niet of nauwelijks voorhanden. Met topologische relaties worden bedoeld *intersectie*, *overlap*, *grenst aan*, etc. Het KES heeft tot doel een goede afbeelding te maken van data uit verschillende bronnen, waartussen geen topologische relaties gedefinieerd zijn. Het (automatisch) leggen van topologische relaties is een complexe operatie, en wordt in het kader van dit project niet beschouwd.

4.3.2 Regels en Algoritmen

In deze paragraaf laten we zien welke regels en algoritmen door een KES zouden kunnen worden geimplementeerd. De regels zijn gedeeltelijk gebaseerd op stappen uit het kartografisch proces dat wordt gevolgd bij het maken van kaartmateriaal. Ze hebben betrekking op de probleemsoorten zoals beschreven in paragraaf 4.2.1. In de paragrafen hierna worden de algoritmes verder toegelicht.

Het resultaat van uitvoering van de kartografische regels levert niet per definitie een optimaal kaartbeeld, om de volgende redenen:

- De informatie die als invoer voor een KES wordt gebruikt is beperkt en incompleet. Er wordt hier vanuit gegaan dat het KES geen weet heeft van meta-informatie, of semantische betekenis van gegevens.
- De regels zijn gebaseerd op heuristieken ('vuistregels') die in veel gevallen kloppen, maar niet in alle.
- Voor veel regels is het werken met grenswaarden of drempelwaarden nodig. Wanneer lijken bijvoorbeeld twee kleuren teveel op elkaar om nog onderscheiden te kunnen worden, of wanneer is de dekking van een kaartlaag 'groot'? Voor deze drempelwaarden moet een keuze worden gemaakt, die de werking van de regels beïnvloedt.
- De hier gegeven regels zijn niet domein-afhankelijk. Het zou kunnen zijn dat voor een bepaald soort kaartinformatie deze 'algemene' regels niet allemaal relevant zijn. Aan de andere kant kunnen er regels worden opgesteld die specifiek voor een bepaald doel gelden. Voor bijvoorbeeld wegenkaarten kan een regel bijvoorbeeld de wegen nadrukkelijk weergeven en de achtergrond zachtere kleuren toebedelen.

4.3.2.1 Kleurconflicten

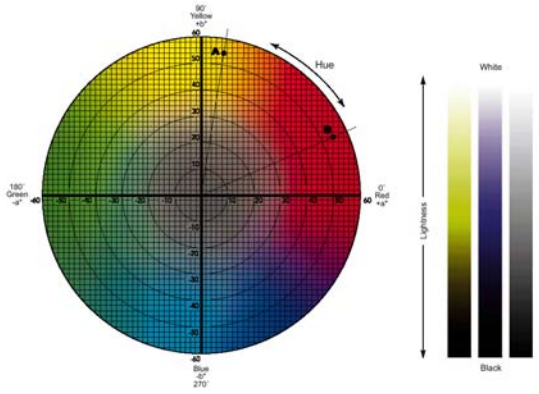
Kleurconflicten kunnen ontstaan doordat kleuren te veel op elkaar lijken om ze te kunnen onderscheiden van elkaar. Hierdoor kan interpretatie van kaartinformatie geheel of gedeeltelijk onmogelijk worden. Denk aan een groene topografische ondergrond met groene wegen eroverheen getekend.

Een KES moet kunnen meten aan kleuren. Het moet bijvoorbeeld een maat kunnen geven aan hoe ver kleuren van elkaar af liggen. Om te kunnen werken met het begrip "kleurverschil" is een device-independent kleurmodel nodig. In het gangbare RGB kleurmodel levert een verandering van een van de drie parameters geen lineaire perceptie-verandering. Daarom wordt voor het bepalen van de perceptuele kleurafstand gewerkt met het device-independent CIELAB kleurmodel.

De CIELAB kleurcomponenten van de RGB kleuren worden bepaald, waarna er met de volgende formule een afstandmaat wordt gedefinieerd [Kotte, 2001]:

$$\Delta E = \sqrt{(\Delta L)^2 + (\Delta a^*)^2 + (\Delta b^*)^2}$$

Hierin representeren parameters L de intensiteit, a* de rood/groen as en b* de geel/blauw as:



Figuur 4.1 CIELAB kleurmodel parameters

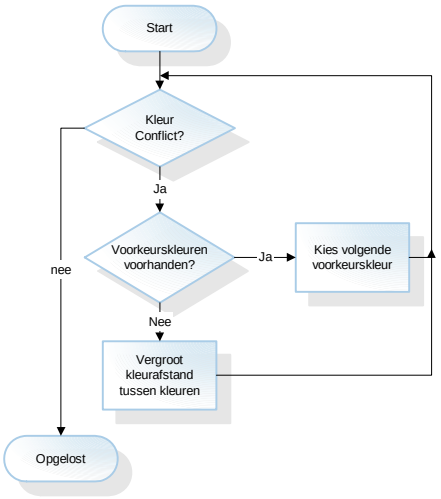
Nu moet er nog een minimale afstand worden bepaald waarbinnen kleuren teveel op elkaar lijken. Deze grens is afhankelijk van vele zaken zoals de eigenschappen van computer hardware, monitor, omgevingslicht en waarnemer. Hierdoor moet de grens ruim worden genomen: kleuren moeten redelijk ver van elkaar afliggen. De parameter ΔE representeert geen interpreteerbare grootheid, ze is slechts een maat voor een afstand.

Nu we kleurafstand kunnen bepalen, kunnen we een oplossing bedenken voor conflictsituaties tussen twee kaartlagen. Twee mogelijkheden worden besproken:

- We kunnen uitgaan van een model voor de kaartlagen dat de specificatie van een aantal voorkeurskleuren toestaat. Het idee is hier dat de kaartlagen in een kaart in principe hun eerste voorkeurskleur krijgen toegewezen. Als dit kleurconflicten geeft, kan voor een aantal lagen de tweede of zelfs derde voorkeurskleur worden gekozen.

- Voor het geval er geen voorkeurskleuren voor handen zijn, (bijvoorbeeld omdat alleen het DLM wordt gecommuniceerd), of omdat de beschikbare voorkeureskleuren allen conflicten geven, is een andere strategie nodig. De conflicterende kleuren kunnen worden aangepast zodanig dat er zo min mogelijk verandert aan het kleurkarakter. De 'hue' (basiskleur of dominante golflengte) van een kleur wordt gelijk gehouden, en de intensiteit wordt juist zoveel aangepast dat het kleurconflict verdwijnt.

Beide situaties kunnen met de volgende regel worden afgedekt:

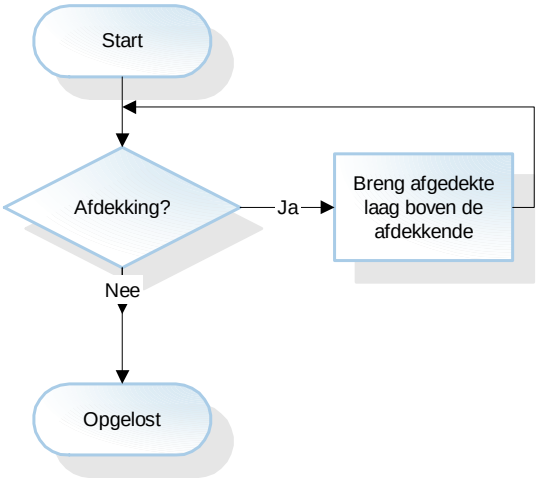


Figuur 4.2 Kleurconflict regel

4.3.2.2 Afdekking

Het probleem van afdekking doet zich voor als alle features van kaartlaag A onzichtbaar zijn doordat features van kaartlaag B eroverheen worden getekend (zie paragraaf 4.2.1). Anders gezegd met behulp van een topologische relatie betekent dit: kaartlaag B *contains* kaartlaag A. Of nog anders gezegd, de *union* van alle features van kaartlaag B *contains* de *union* van features van laag A.

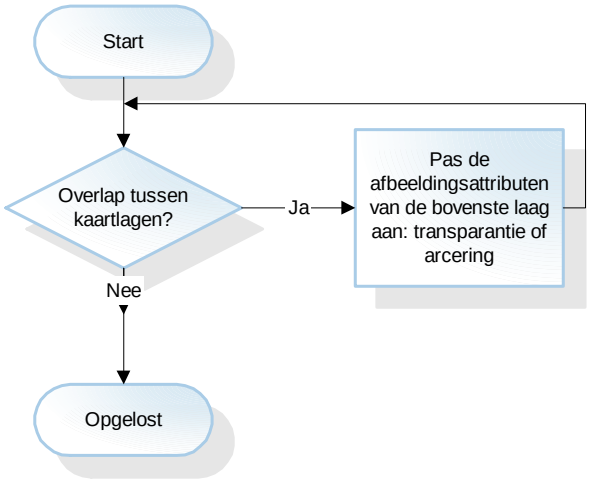
Afdekking is moeilijk te detecteren met de meeste GIS viewer software. Containment is een relatie die kan worden vastgesteld door het bevragen van de ruimtelijke gegevens van het DLM. Er zijn echter wel vuistregels op te stellen die de kans op afdekking zo klein mogelijk maken. Zo kan een regel die de kaartlagen sorteert op dekkingspercentage ervoor zorgen dat lagen met weinig informatie een minder grote kans hebben om te worden afgedekt door een andere laag.



Figuur 4.3 Afdekkings regel

4.3.2.3 Overlapping

Gerelateerd aan afdekking is overlapping; afdekking is te definieren als 100% overlapping. Een reden om het hier apart te bespreken is dat er andere oplossingen voor zijn.



Figuur 4.4 Overlappings regel

4.4 Technisch Ontwerp

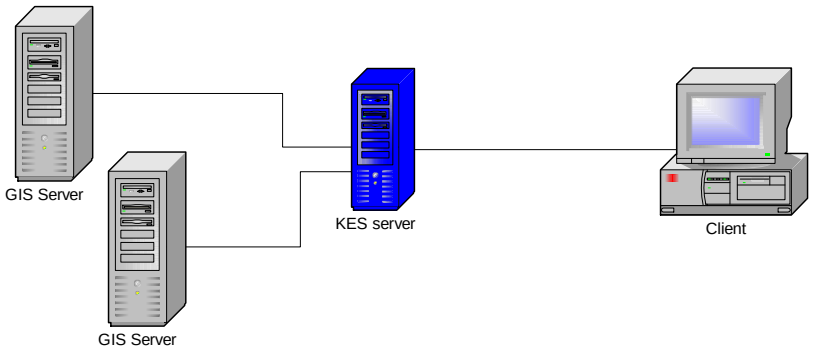
4.4.1 Introductie

Na het opstellen van het functioneel ontwerp gaan we bij het maken van het technisch ontwerp kijken hoe we het KES technisch kunnen realiseren. Als eerste wordt de architectuur besproken, daarna het objectmodel, en als laatste worden software componenten geselecteerd die gebruikt kunnen worden in het systeem.

4.4.2 Architectuur

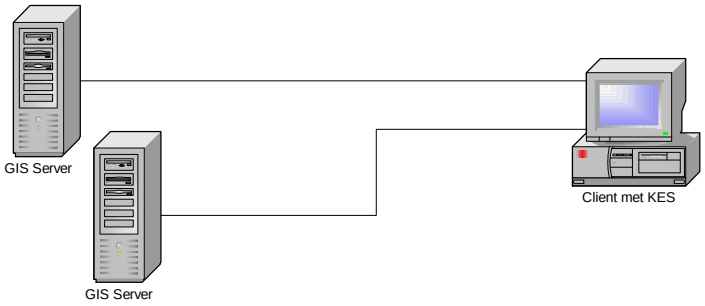
Bij het bepalen van de architectuur van het KES is rekening gehouden met het primaire doel van het ontwikkelen van een KES: het evalueren van de functionaliteit en het praktisch nut van een KES.

Een KES kan op verschillende niveaus in de keten van gegevensbron naar afbeelding worden ingezet. Een voor de hand liggende mogelijkheid is als middleware tussen gegevensbron en afbeelding, om op deze wijze een integratie ‘service’ aan verschillende clients te kunnen bieden.



Figuur 4.5 KES als middleware oplossing

Een andere mogelijkheid is plaatsing aan de client zijde, bijvoorbeeld als plugin of applet in de internetbrowser, zodat specifieke toepassingen gerealiseerd kunnen worden.



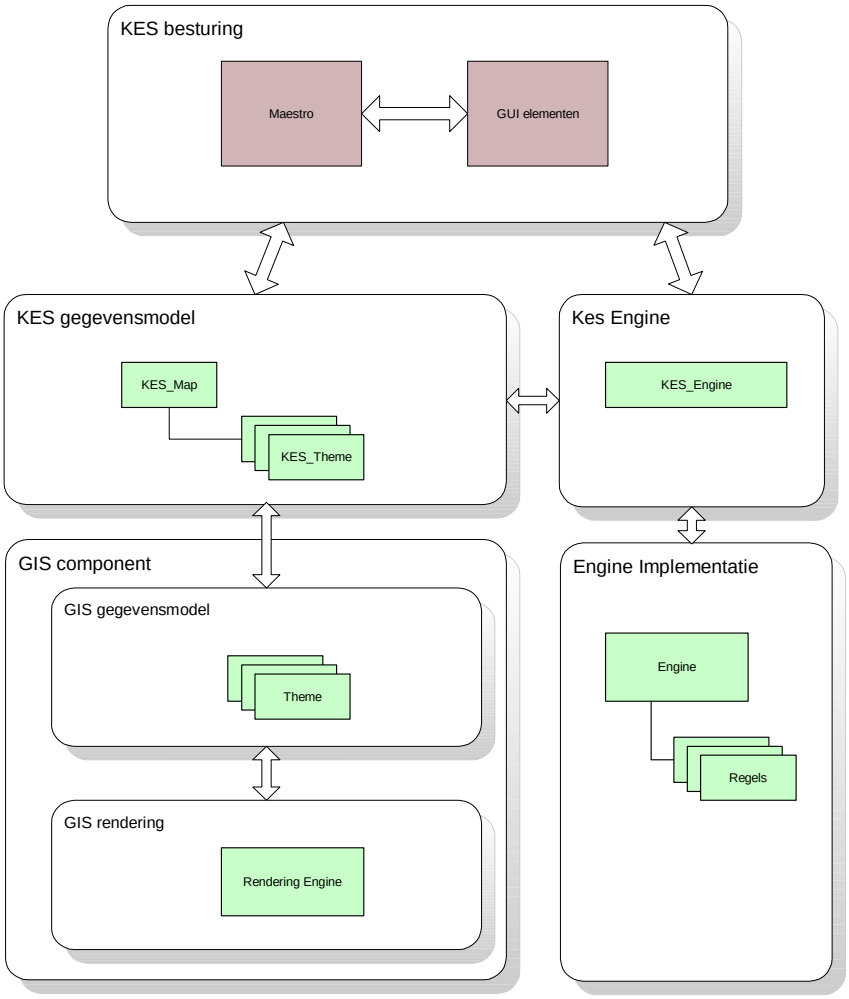
Figuur 4.6 KES draaiend op het client platform zelf

Hoewel een KES dus ingebouwd zou kunnen worden op meerdere plekken in het netwerk tussen client en databronnen, wordt in eerste instantie gekozen voor implementatie aan de clientkant. Hierdoor is ontwikkelen en testen met verschillende datasets eenvoudiger, zonder compromis aan de mogelijkheid om de functionaliteit te beoordelen.

4.4.3 Objectmodel en Framework

Op basis van de requirements en het functioneel ontwerp is het volgende objectmodel opgesteld voor een compleet KES systeem. Bij het ontwerp is rekening gehouden met de beoogde genericiteit van het systeem door definitie van een framework van KES-specifieke objectklassen.

Het framework biedt de mogelijkheid om externe objectklassen, zoals die van een GIS component, in te kapselen. Het kan worden gezien als een verzameling 'templates' of 'interfaces'. Hierdoor kan op een eenvoudige en elegante manier een alternatieve implementatie voor een van de componenten worden gemaakt. Zo kan bijvoorbeeld een GIS tool van een andere fabrikant worden 'aangesloten' op het KES. Dezelfde genericiteit geldt voor de engine module. Omdat er verschillende mogelijkheden voor de implementatie van een KES "motor" zijn, is een KES engine-interface gedefinieerd die gebruikt kan worden voor verschillende implementaties.



Figuur 4.7 Structuur KES Framework ontwerp

Een belangrijk onderdeel in bovenstaand framework is het **KES gegevensmodel**. Dit model is wat een engine en een userinterface te 'zien' krijgen en waarmee ze kunnen manipuleren. Het gegevensmodel biedt een gestandaardiseerde manier om DLM en DKM informatie toegankelijk te maken voor het KES.

In de module **KES besturing** worden objecten ondergebracht die zorgen voor gebruikersinteractie, en voor het op een centrale plek registreren van *events* of gebeurtenissen in het KES. Zo kan een engine module door een *Maestro* object uit de besturing module op de hoogte worden gebracht van het feit dat een gebruiker de kaartlaag volgorde heeft aangepast.

Voor de module **GIS component** wordt een bestaand GIS viewer component gekozen. Het interne objectmodel dat zo'n component hanteert voor het representeren van DLM en DKM wordt door het **KES gegevensmodel** 'vertaald' voor gebruik in het KES.

4.4.4 Software Componenten Selectie

Voor de invulling van het objectmodel bij de implementatie hebben we (hulp)componenten nodig, die zorgen voor bijvoorbeeld het inlezen van het DLM, en het maken van een kaartafbeelding. Deze functies worden onder anderen door GIS desktop pakketten uitgevoerd, maar omdat we werken aan een KES dat inzetbaar is op internet zullen we nagaan of er 'plugin' componenten beschikbaar zijn die voldoen om de functie van GIS viewer in het KES te vervullen. Daarna kijken we of er meer onderdelen uit het KES objectmodel reeds beschikbaar zijn.

4.4.4.1 GIS viewer

Voor de GIS viewer module die voor de weergave moet zorgen zijn de volgende alternatieven overwogen:

- GeoTools, een open-source software pakket geschreven in Java
- Lava, een GIS viewer die gekoppeld kan worden aan Magma, een GIS server.
- MapXtreme van Mapinfo, een GIS viewer component
- MapObjects, een GIS viewer solution van Esri

Feature	GeoTools	Lava	Esri MapObjects	MapInfo MapXtreme
Open source	Ja	Ja[1]	Nee[2]	Nee[1]
Open Standaarden (o.a. GML) invoer	Ja	Nee	Nee[3]	Nee
Geavanceerde grafische variabelen[4]	Nee	Nee	Ja	Ja
Onafhankelijk van serverplatform	Ja	Nee	Nee[5]	Nee
Minimaal vereiste JDK	V 1.1	V1.1	V 1.3	V 1.3
updates/bugfixes	Ja	Nee	Ja	Ja

[1]Source beschikbaar voor onderzoeksprojecten
[2]Niet open source, wel een gedocumenteerde API (Application Programming Interface)
[3]Niet standaard, ondersteuning voor andere formaten is eventueel te ontwikkelen
[4]Bijvoorbeeld de attributen transparantie, animatie, arcering
[5]Werkt met ArcIMS (internet map server) serversoftware

Er is gekozen voor GeoTools. Doorslag bij de keuze voor GeoTools gaven de eigenschap dat de GeoTools componenten in elke Internet Browser kunnen draaien (eventueel met een standaard Java v1.1 VM), dat het pakket opensource is, hetgeen aanpassingen mogelijk maakt, en dat het meerdere standaarden ondersteunt, waaronder OpenGIS standaarden als WMS en GML.

De mogelijkheden van de GeoTools component zijn onder anderen:

- inlezen en weergeven van verschillende populaire GIS bestandsformaten
- een intern gegevensmodel dat voor alle invoerformaten te gebruiken is
- de mogelijkheid om externe componenten te informeren over events in het GIS component.
- basic hulpcomponenten zoals *zoom* en *pan* tools en een legenda control

Zie appendix 9.2 voor een kort overzicht van de structuur en het objectmodel van GeoTools.

4.4.4.2 Engine

De Engine module gaat de regels in het KES bevatten en toepassen. Voor deze module zijn er verschillende implementatie opties:

- Direct gebruik maken van een traditionele programmeertaal (procedureel, of Object geörienteerd). De functionaliteit moet hiervoor strikt geformaliseerd zijn, uitzonderingssituaties moeten van tevoren bekend zijn en ondervangen worden.
- Implementatie van een engine met een expertsysteem. Kartografische kennis wordt in dit geval in meer impliciete vorm opgegeven. De handelingen die het KES moet verrichten vloeien in deze opzet meer voort uit de definitie van een doel, niet zozeer uit de definitie van een algoritme.

De tweede optie leek bij aanvang van het project het best aan te sluiten bij de slecht te formaliseren kartografische kennis die het systeem zal moeten toepassen.

5 Implementaties KES

5.1 Introductie

Bij de implementatie van het KES wordt het gemaakte ontwerp gerealiseerd.

De implementatie fase van het KES is onderverdeeld in 3 projecten:

- het ontwikkelen van het framework voor het KES.
- het ontwikkelen van een KES engine implementatie met een expert systeem shell
- het ontwikkelen van een KES engine implementatie met een procedurele engine.

Om na te gaan of de inzet van een expertsysteem voor het KES werkbaar is, is eerst een "proof of concept" met de expertsystemshell Jess gemaakt, dat in paragraaf 5.3 beschreven wordt. Een volledige implementatie van het KES met Jess wordt beschreven in hoofdstuk 5.4. Als alternatief wordt onderzocht of op basis van het KES framework een procedurele engine geïmplementeerd kan worden in paragraaf 5.5.

Bij de implementatie van het ontwerp moeten keuzes gemaakt worden voor de invulling van verschillende onderdelen uit het ontwerp van het systeem. Voor de engine die de Kartografische kennis gaat herbergen en voor de GIS viewer zijn verschillende opties die we hierna bespreken.

5.2 Framework

Het KES framework is bedoeld als generiek raamwerk waarin software componenten geïntegreerd kunnen worden tot een Kartografisch expert systeem. Het biedt een mate van abstractie van de onderliggende objectstructuur van de gebruikte softwarecomponenten.

De objectklassen voor het framework werden speciaal voor dit project gemaakt. De Maestro module is de spin in het web van het KES. Signalen uit alle onderdelen komen hier op een centrale plek binnen. Een signaal uit de GIS viewer software dat de kaartlaagvolgorde door een gebruiker is aangepast kan zo worden doorgegeven aan een engine component.

Andere objecten in het framework zijn abstraties van een kaart en van een thema object. Eigenschappen van deze entiteiten die belangrijk zijn voor het KES, zoals kleur en identificatie (naam), zijn in deze objecten opgenomen

Om de comunicatie tussen de objecten in het framework te vergemakkelijken zijn er enkele hulpklassen zoals eventlisteners gemaakt.

5.2.1 GeoTools

Bij het maken van het KES framework is gewerkt met GeoTools, een opensource GIS viewer applicatie in Java geschreven. [URL 12].

De GeoTools module kan geografische gegevens inlezen volgens een aantal standaarden, waaronder GML 1, MapInfo MIF en Esri Shapefiles. Omdat de interne datastructuren van GeoTools praktisch gelijk zijn voor verschillende typen invoerfiles, hebben we ervoor gekozen om met het ruim beschikbare Esri Shapefile formaat te werken, en niet met het open GML formaat. In paragraaf 2.9 werd het belang van open standaards aangegeven, en in een ideale situatie zou GML de voorkeur hebben gehad. De keuze voor ShapeFiles heeft dus slechts een praktische achtergrond, en geen gevolgen voor de genericiteit van het systeem.

De GeoTools GIS viewer software is voor dit project op een aantal manieren uitgebreid. Zo is standaard geen uitgebreide faciliteit in GeoTools aanwezig om de eigenschappen van thema's, zoals kleur en volgorde te veranderen. De ThemePanel objectklasse is uitgebreid met deze functionaliteit. Een themepanel geeft nu de legendakleuren van een thema weer, en door met de rechter muisknop op een kaartlaag in de legenda te klikken kan de kleur worden aangepast. De thema volgorde kan worden aangepast door een thema in de legenda naar een nieuwe plek te slepen met de muis.

GeoTools is gebouwd voor het Java 1.1 platform, waardoor grafische effecten die in Java 2 aanwezig zijn hier niet gebruikt konden worden. Om toch een extra grafisch attribuut te kunnen gebruiken in het KES is de mogelijkheid aan GeoTools toegevoegd om een kaartlaag te arceren.

Kleinere aanpassingen waren nodig om het objectmodel van GeoTools te ontsluiten. In de sourcecode zijn deze wijzigingen met de initialen IK in een commentaarblok gemarkeerd.

5.3 Een proof of concept engine met Jess

Om na te gaan of de expert system shell Jess geschikt zou kunnen zijn om te dienen als redenatieengine voor een KES is eerst, zonder gebruik te maken van het KES framework, een prototype gemaakt.

5.3.1 Introductie tot Expert System Shell Jess

Jess is een expert system shell geschreven in de Java programmeertaal. Het is in staat om regels (rules) toe te passen op feiten (facts). In het jargon van expert systems spreekt men van het vuren (firing) van een regel als de feitenverzameling daar aanleiding toe geeft. Regels hebben een left-handside (LHS) die een conditie aangeeft en een right-hand side (RHS) die een actie representeert. De expertsystemshell is geoptimaliseerd om aanwezige feiten te matchen op de verzameling LHS van de regels. Het vuren van een regel kan leiden tot een actie, varierend van het creëren van nieuwe feiten, het verwijderen van feiten, het aanroepen van programmacode, of het tonen van een mededeling aan de gebruiker.

Om de Jess scripts beter te kunnen lezen, is in appendix 9.1 een (compacte) opsomming van sleutelbegrippen en structuur opgenomen. De termen zijn niet vertaald naar Nederlandse termen om verwarring te voorkomen. Voor uitgebreidere informatie over het gebruik van Jess zie [URL 6], waar een handleiding van Jess is te vinden. Voor een beschrijving van concepten in Jess en het Rete-netwerk algoritme dat gebruikt wordt voor de pattern-matching in Jess zie [Friedman-Hill 1997].

5.3.2 Eerste "proof of concept" met Jess

Een klein kennissysteem werd opgezet om te beproeven of het principe van kleurconflicten tussen lagen opgelost kan worden met behulp van de Jess expertsystem shell.

Uitgangspunten voor deze vereenvoudigde opzet waren:

- kleuren worden geïdentificeerd door een indexnummer (ID), en zijn niet nader gespecificeerd in bijvoorbeeld RGB waarden. Een ID symboliseert een unieke RGB combinatie.
- een kaartlaag heeft een enkele kleur
- een kaartlaag heeft een type (lijn/vlak/bitmap)
- een kaartlaag heeft een aantal reserve voorkeurskleuren, die in geval van kleurconflicten kunnen worden ingezet.
- het expertsysteem is niet gekoppeld aan een GIS viewer, de uitvoer is tekstueel
- doel van dit expertsysteem is het onderscheidbaar maken van de kaartlagen, door het oplossen van het kleurconflict.

De werking van dit prototype is als volgt:

Er worden vier (fictieve) kaartlagen opgevoerd in de vorm van feiten in het expertsysteem. Deze feiten hebben attributen (of 'slots' in Jess jargon) die eigenschappen zoals naam, voorkeurskleuren en kaartlaag type vastleggen.

Een voorbeeld van een Jess statement dat een kaartlaag-feit toevoegt is:

```
(assert (MapLayer (name Rijks-Wegen) (type line) (colors 2 6 8) ))
```

Het laatste attribuut, een zogenaamd multi-slot attribuut, legt vast dat het ID van de eerste voorkeurskleur 2 is, de tweede voorkeurskleur ID 6 heeft, en de derde heeft ID=8.

In onderstaand voorbeeld hebben we expres vier kaartlagen met dezelfde drie voorkeurskleuren opgevoerd om het KES haar werk te kunnen laten doen, in dit geval het oplossen van kleurconflicten. Als de kaartlagen zouden worden afgebeeld zouden de kaartsymbolen immers met dezelfde kleur worden getekend waardoor het onderscheid wegvalt.

In onderstaande uitvoer zijn begin en eindsituatie vetgedrukt. In de uitvoer is te zien dat na afloop alle lagen een unieke (eerste) kleur hebben, zoals de bedoeling is. De laag 'OnverhWegen' heeft een willekeurige kleur toegewezen gekregen omdat alle voorkeurskleuren (2,6 en 8) een conflict zouden geven, en de laag toch onderscheiden moet kunnen worden van de andere lagen.

Als de fictieve kaartlagen die hier worden gemanipuleerd kunnen worden vervangen door kaartlagen uit een GIS, dan zou dit bescheiden script voldoende zijn om een kleurconflict op te lossen.

Aangetoond is dat met een relatief kort script[1] een onderdeel van het KES kan worden geïmplementeerd. De bruikbaarheid van een expertsystem shell voor de ontwikkeling van een KES is hiermee aannemelijk gemaakt.

[1] zie bijlage 9.4.1 voor complete afdruk van het script

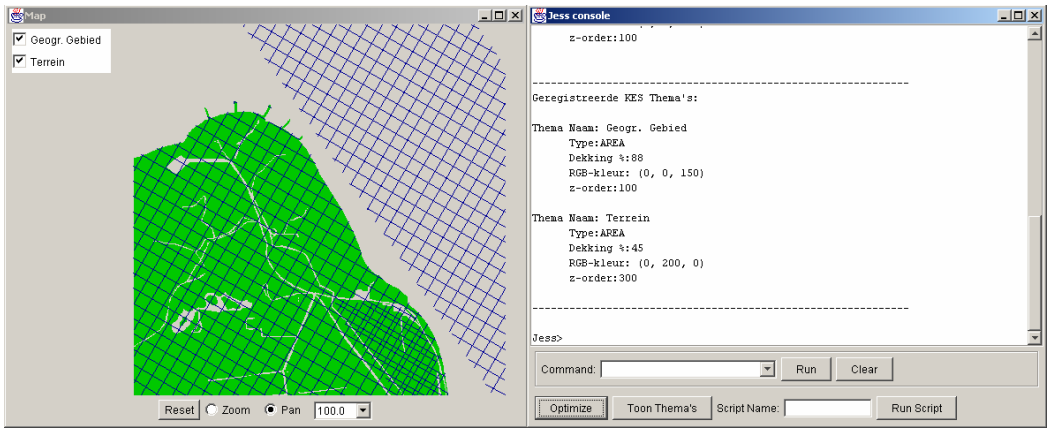
```
layer3.clp uitvoer
layernaam: PerceelGren, type: line, kleuren: (2 6 8)
layernaam: OnverhWegen, type: line, kleuren: (2 6 8)
layernaam: ProvinWegen, type: line, kleuren: (2 6 8)
layernaam: Rijks-Wegen, type: line, kleuren: (2 6 8)
!----  Geen Layers met eerste kleur 2 meer behalve layer: PerceelGren
!----  Dubbele kleuren voor: PerceelGren OnverhWegen
!----  kleuren van layer: OnverhWegen worden aangepast van 2(6 8) naar (6 8)2
layernaam: OnverhWegen, type: line, kleuren: (6 8 9)
!----  Geen Layers met eerste kleur 6 meer behalve layer: OnverhWegen
!----  Dubbele kleuren voor: ProvinWegen PerceelGren
!----  kleuren van layer: PerceelGren worden aangepast van 2(6 8) naar (6 8)2
layernaam: PerceelGren, type: line, kleuren: (6 8 59)
!----  Geen Layers met eerste kleur 6 meer behalve layer: PerceelGren
!----  Dubbele kleuren voor: PerceelGren OnverhWegen
!----  kleuren van layer: OnverhWegen worden aangepast van 6(8 9) naar (8 9)6
layernaam: OnverhWegen, type: line, kleuren: (8 9 31)
!----  Geen Layers met eerste kleur 8 meer behalve layer: OnverhWegen
!----  Dubbele kleuren voor: Rijks-Wegen ProvinWegen
!----  kleuren van layer: ProvinWegen worden aangepast van 2(6 8) naar (6 8)2
layernaam: ProvinWegen, type: line, kleuren: (6 8 0)
!----  Dubbele kleuren voor: ProvinWegen PerceelGren
!----  kleuren van layer: PerceelGren worden aangepast van 6(8 59) naar (8 59)6
layernaam: PerceelGren, type: line, kleuren: (8 59 42)
!----  Geen Layers met eerste kleur 8 meer behalve layer: PerceelGren
!----  Dubbele kleuren voor: PerceelGren OnverhWegen
!----  kleuren van layer: OnverhWegen worden aangepast van 8(9 31) naar (9 31)8
layernaam: OnverhWegen, type: line, kleuren: (9 31 29)
!----  Geen Layers met eerste kleur 9 meer behalve layer: OnverhWegen
f-0    (initial-fact)
f-1    (MapLayer (name Rijks-Wegen) (type line) (colors 2 6 8))
f-8    (MapLayer (name ProvinWegen) (type line) (colors 6 8 0))
f-9    (MapLayer (name PerceelGren) (type line) (colors 8 59 42))
f-10   (MapLayer (name OnverhWegen) (type line) (colors 9 31 29))
For a total of 5 facts.
```

Figuur 5.1 Uitvoer van het Jess script

5.4 KES implementatie met Jess

Voor een geïntegreerde implementatie van het KES ontwerp is de expert system shell Jess in het ontwikkelde framework geplaatst. De opzet hiervan is om middels het redenatiemechanisme van een expertsystemshell de weergave van de landschapsinformatie te sturen.

Een Jess script legt de regels vast die worden uitgevoerd. Om Jess te laten werken met geografische informatie is het uitgebreid met een aantal *user functions* die bijvoorbeeld het meten van kleurafstanden mogelijk maken. Deze functies kunnen in de scripts worden gebruikt bij het opstellen van regels.



Hierboven is de vulkleur van een polygoon kaartlaag vervangen door een arceerpatroon, om een onderliggende laag zichtbaar te maken.

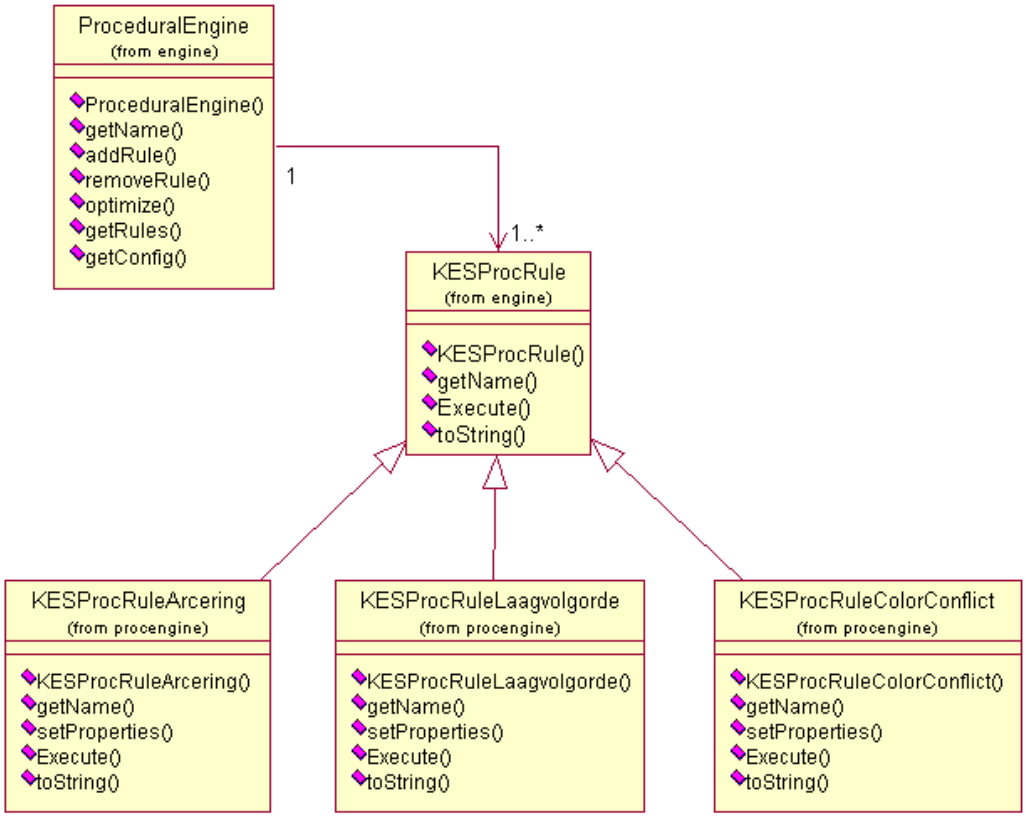
5.4.1 Regels

De kartografische regels zijn in dit systeem geformuleerd in de CLIPS scripting language. Als voorbeeld staat hieronder de kleurconflictregel afgedrukt.

```
(defrule KleurenConflict "Checkt op vergelijkbare kleuren"
    ?fact1 <- (KEStheme (name ?n) (themetype ?t) (color_rgb ?c1) (OBJECT ?o1))
    ?fact2 <- (KEStheme (name ?m&~?n&:(< 0 (str-compare ?n ?m))) (themetype ?t)
(color_rgb ?c2) (OBJECT ?o2))
    (test (< (kes_comparecolors ?c1 ?c2) 0.04 ) )
    =>
    (printout t "-------  Kleurconflict tussen: " ?n " , " ?m crlf)
    (printout t "-------   kleurafstand (CIE): " (kes_comparecolors ?c1 ?c2) crlf)
    (printout t "-------   kleuren van layer: " ?m " worden aangepast van "
(kes_pp_rgbcolor ?c1) " naar nieuwe kleur" crlf)
    (call ?o1 brightenColor )
    (call ?o2 darkenColor )
   )
```

5.5 KES implementatie: procedureel

Hetzelfde framework van GIS viewer en kaartobjectmodel is gebruikt om een procedurele implementatie te maken van het KES. Doel van deze implementatie is om de voor en nadelen te toetsen ten opzichte van een KES gebaseerd op een expert system shell engine .



Figuur 5.2 Objectmodel voor de procedurele engine en bijbehorende regels

Alle procedurele regels erven over van KESProcRule, die het geraamte voor een KES regel bevat, zonder kartografische functionaliteit. Er zijn hier drie regels gespecificeerd, maar dat aantal kan zonder problemen worden uitgebreid.

5.5.1 Regels Procedurele engine

De regels die de KES functionaliteit beschrijven zijn in het geval van de procedurele engine niet vastgelegd in een CLIPS script, maar in Java programmacode. De programmacode is gedeeltelijk vanuit een configuratiebestand te beinvloeden door het opgeven van parameters zoals drempelwaarden, en het aan- of uitschakelen van bepaalde regels.

Hieronder in Figuur 5.3 is een voorbeeld afgedrukt van hoe de implementatie van de *execute()* methode van zo'n regel eruit ziet. Om een dergelijke regel op te stellen, is in elk geval enige Java programmeerkennis vereist.

```
public void Execute(KES_Map km) throws KESexception {
KEStheme2 kt1;
int i=0;
Vector tempThemes;
       super.Execute(km);
       tempThemes = km.getKESthemes();
       Collections.sort(tempThemes,new KEStheme2DekkingComparator());
       for (i=0 ; i< tempThemes.size() ; i++) {
           kt1=(KEStheme2)tempThemes.elementAt(i);
           kt1.setZorder(i);
       }
}
```

Figuur 5.3 Code fragment uit de afdekking regel uit de procedurele engine

Voor het maken van nieuwe regels kan gebruik worden gemaakt van de basisklasse KESProcRule.

Parameters voor deze regels kunnen worden opgegeven door middel van een configuratiebestand. Zo kan bijvoorbeeld het maximaal aantal te arceren kaartlagen worden ingesteld, of de drempelwaarde voor het bepalen van overeenkomst tussen twee kleuren. Ook kunnen complete regels worden geactiveerd of gedeactiveerd.

De regels uit een configuratiebestand dat de kleurconflictregel aanzet en de drempelwaarde voor het bepalen van kleurverschil op 12% zet, ziet er bijvoorbeeld zo uit:

```
EnableKESProcRuleColorConflict = true
Kleurverschildrempelwaarde     = 0.12
```

5.5.2 Kleurconflictregel

De strategie voor het oplossen van kleurconflicten in de procedurele engine is licht gewijzigd ten opzichte van de Jess implementatie. Er wordt nog steeds een kleurafstand gemeten in de CIELAB ruimte. Een eventuele aanpassing van een kleur gebeurt door aanpassen van de 'brightness' of helderheid van een kleur. Een probleem is nu dat het meten van de kleurafstand slechts aangeeft dat kleuren een bepaalde 'zichtbaarheids' afstand hebben, maar ze geeft niet aan welke kleur er 'lichter' is. De afstand kan immers veroorzaakt worden door parameters L, a* en/of b* (zie paragraaf 4.3.2.1).

Om te weten in welke richting de helderheid van conflicterende kleuren moet worden aangepast, wordt naast de kleurafstand ook de helderheid apart gemeten. Op basis hiervan wordt de kleur met de grootste helderheid nog iets helderder, en de andere kleur iets minder helder. Als deze controle niet gedaan zou worden, zou een aanpassing van de helderheid kunnen resulteren in vernauwing van de kleurafstand, terwijl we vergroting willen.

5.5.3 Afdekkingsregel

De daadwerkelijke afdekkingsrelatie (containment) tussen 2 thema's kan met de huidige software niet worden bepaald. De afdekkingsregel gebruikt een heuristiek om afdekking tegen te gaan, namelijk dat kaartlagen worden gesorteerd naar dekkingspercentage met de kleinste dekking bovenop.

5.5.4 Overlapregel

Net als afdekking kan ook overlap niet worden gemeten door beperkingen in de functionaliteit van de gebruikte GIS software. Als vuistregel is hier gebruikt dat er sprake is van overlap als er twee kaartlagen zijn die elk meer dekken dan 50% van het kaartbeeld, dan moeten ze ergens overlappen, en kan aan de hand van de regel één van beiden gearceerd worden.

De implementatie van deze regel in de procedurele engine is als volgt:

- De regel verwacht gesorteerde kaartlagen (door de laagvolgorde regel)
- De onderste (meest dekkende, vanwege de laagvolgorde regel) wordt overgeslagen met arceren. De eerstvolgende laag wordt gearceerd als deze een groter dekkingspercentage heeft dan is ingesteld.

6 Test en Evaluatie

6.1 Test criteria

Na de ontwikkeling van het geintegreerde KES zijn een aantal tests uitgevoerd om de prestaties te toetsen.

Aandachtspunten bij het testen en evalueren:

Algemeen
- voldoet het KES aan de requirements? Kan het de taken vervullen waarvoor het werd ontworpen?

Functionaliteit
- worden alle conflictsituaties naar verwachting opgelost?
- worden de regels goed uitgevoerd?

Performance
- Werkt het KES snel genoeg voor de inzet in praktijksituaties?

6.2 Datasets

Er zijn tests uitgevoerd met twee verschillende datasets afkomstig uit de Top10 database van de Topografische Dienst Nederland:

- Texel:
  - Zes kaartlagen van een gebied aan de rand van Texel
    - Geografisch gebied
    - Bebouwing
    - Terrein
    - Waterdeel
    - Wegdeel
    - Administratief gebied

- Tiel:
  - Zeven kaartlagen van een wijk in Tiel:
    - Geografisch gebied
    - Bebouwing
    - Terrein
    - Waterdeel
    - Wegdeel
    - Spoorbaan
    - Administratief gebied

6.3 Scenario's

Met een dataset zijn verschillende scenario's te maken. Een scenario bestaat uit een verzameling kaartlagen met een bepaalde kleurenconfiguratie en laagvolgorde. Door opzettelijk conflictsituaties te specificeren kunnen we nagaan in hoeverre het KES een bruikbaar resultaat opleverd.

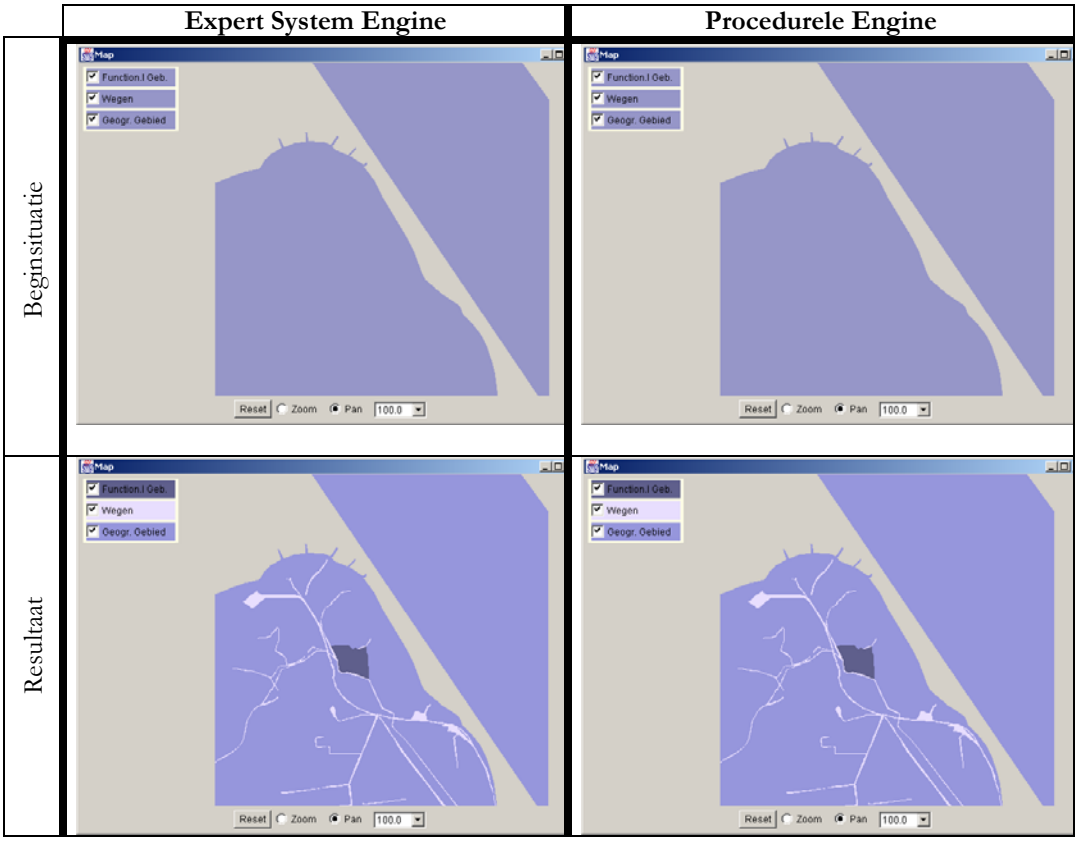
De scenario's worden met beide KES engines getest, de resultaten zijn naast elkaar afgebeeld ter vergelijking.

6.3.1 Scenario 1

In het eerste scenario bekijken we het kleurconflict oplossend vermogen van het KES.

Uitgangssituatie

Dataset:	Texel, drie kaartlagen: Geogr. gebied, Function. Geb., Wegdeel
Voorkeurskleuren:	Voor alle kaartlagen gelijk: lichtblauw
Kaartlaagvolgorde:	Juiste weergave volgorde: (van onder naar boven: Geogr. geb., wegen, function. geb)



Figuur 6.1Uitgangssituatie (boven) en resultaat (onder) na uitvoering van de kleurconflictregel in de twee KES engines

Resultaat
Beide engines lossen het kleurconflict correct op. Dat ze hetzelfde resultaat opleveren is toevallig omdat de volgorde van conflict oplossen bij de Jess engine niet vast ligt, en bij de procedurele engine wel. Het had goed kunnen zijn dat de Jess engine het kleine functioneel gebied juist lichter dan donkerder had gemaakt.

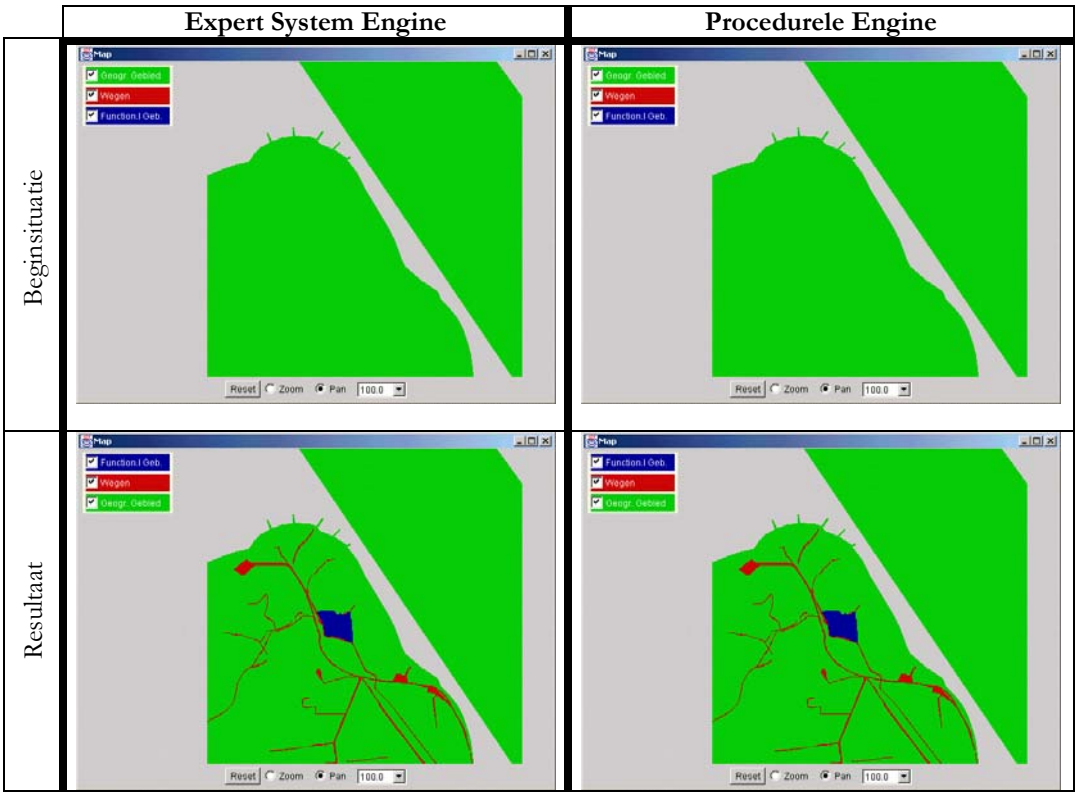
De Jess engine had ongeveer 3 keer zoveel tijd nodig als de procedurele engine om de optimalisatie te voltooien.

6.3.2 Scenario 2

Het tweede scenario is ingericht om de laagvolgorde te optimaliseren.

Uitgangssituatie

Dataset:	Texel, drie kaartlagen: Geogr. gebied, Function. Geb., Wegdeel
Voorkeurskleuren:	Voor alle kaartlagen verschillend: Rood, groen en blauw
Kaartlaagvolgorde:	Niet optimaal: de grootst dekkende laag (Geogr. geb.) ligt bovenop de anderen.



Figuur 6.2Uitgangssituatie (boven) en resultaat (onder) na uitvoering van de laagvolgorde regels van de twee KES engines

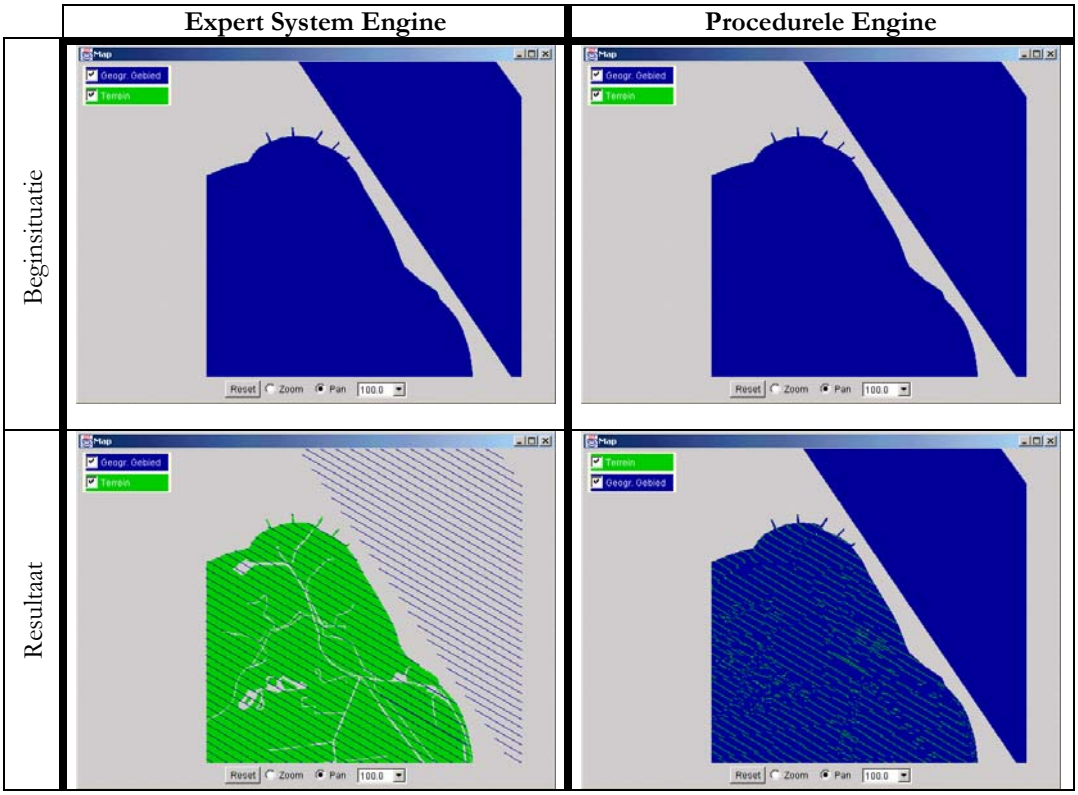
Resultaat
Beide engines sorteren de lagen correct naar dekkingspercentage. De Jess engine had 4 keer zo lang nodig als de procedurele engine om de optimalisatie te voltooien.

6.3.3 Scenario 3

Het derde scenario is ingericht om arcering toe te passen als er overlap is.

Uitgangssituatie

Dataset:	Texel, twee kaartlagen: Geogr. gebied, terrein
Voorkeurskleuren:	Voor de twee kaartlagen verschillend: blauw en groen
Kaartlaagvolgorde:	Onjuiste weergave volgorde.



Figuur 6.3Uitgangssituatie (boven) en resultaat (onder) na uitvoering van de arceringstregel in de twee KES engines

Resultaat

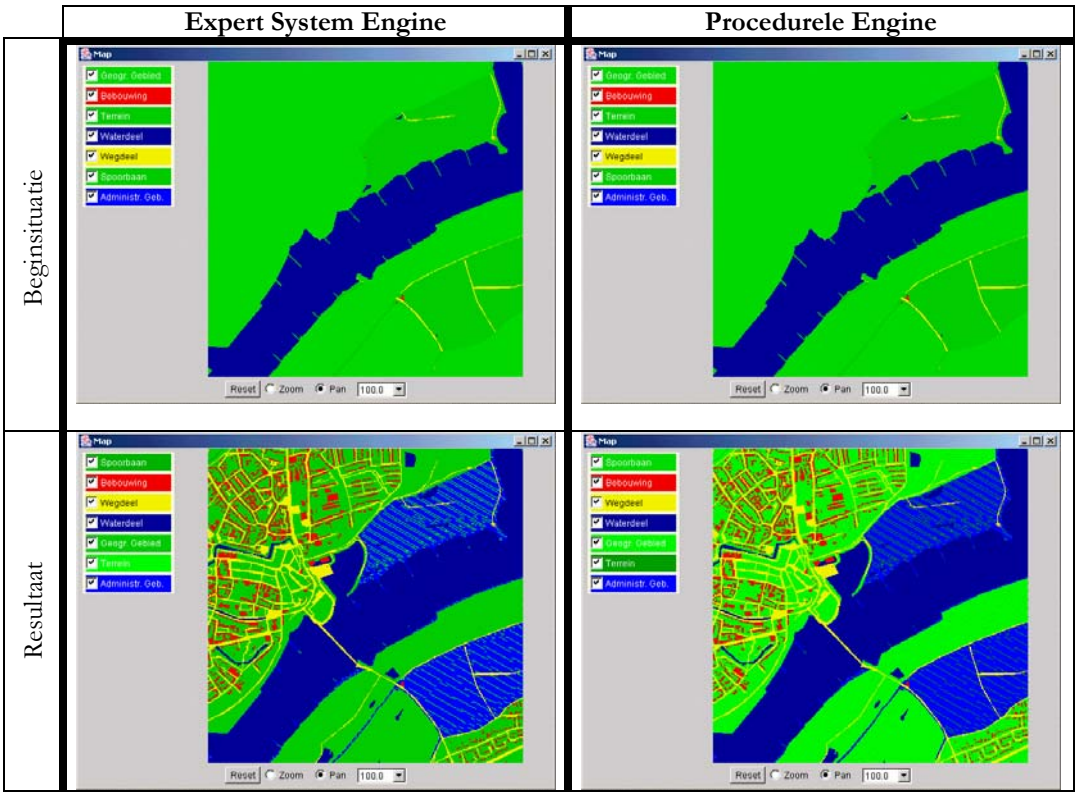
Beide engines arceren één van beide kaartlagen, maar kiezen beiden een andere laag om te arceren. De oplossing van de procedurele engine is niet correct, omdat de groene laag onder de blauwe ligt (de volgorde is voor het screenshot boven handmatig aangepast om de arcering te laten zien). In combinatie met de laagvolgorde regel levert echter ook de procedurele engine een correcte arcering.

6.3.4 Scenario 4

Het vierde scenario is een totaalscenario. Hier worden alle regels tegelijk uitgevoerd.

Uitgangssituatie

Dataset:	Tiel, alle zeven kaartlagen
Voorkeurskleuren:	Diverse kleuren, waarvan een aantal kleurconflicten geven
Kaartlaagvolgorde:	Willekeurig, niet optimaal



Figuur 6.4Uitgangssituatie (boven) en resultaat (onder) na uitvoering van alle drie de regels in de twee KES engines

Resultaat
De twee resultaat kaarten lijken op elkaar, op de keuze van de alternatieve kleuren na. Het kaartbeeld is in elk geval beduidend beter dan bij de beginsituatie. Alle lagen hebben een onderscheidbare kleur, er is arcering toegepast om het effect van overlap te ondervangen, en de laagvolgorde is optimaal.

6.4 Gecombineerde Resultaten

De tests in scenario 1,2 en 4 laten het verwachte resultaat zien, scenario 3 met de procedurele engine levert geen juiste oplossing, echter in combinatie met de laagvolgorde regel wordt dit verholpen.

Tijdens het testen bleek dat de Jess engine beduidend langzamer, een factor 2-10 trager, functioneert. Dit is niet verwonderlijk gezien de complexiteit van het opbouwen van een speciaal Rete-netwerk om het matchen van feiten tegen regels mogelijk te maken. (zie [Friedman-Hill 1997])

6.5 Evaluatie

Een implementatie van een KES met een expert system levert niet de verwachte voordelen op. Met het expert systeem is niet genoeg grip op het kaartmodel te krijgen. Zo is het zonder externe hulpfuncties zeer lastig om de kaartlaag met het kleinste dekkingspercentage op te zoeken. De

'blik' van het KES blijft beperkt tot 2 kaartlagen tegelijk. Hiermee kan bijvoorbeeld wel de laagvolgorde geoptimaliseerd worden, maar dat gaat dan volgens een bubble-sort patroon, waarbij de lagen langzaam in de juiste volgorde 'borrelen'. Het uitvoeren van een kleurconflict oplossing komt met deze opzet zelfs niet gegarandeerd tot een oplossing.

De snelheid van een systeem met een expert system shell laat te wensen over. Het expert system is continu bezig om feiten te matchen met de regels, waarbij ook nog eens speciaal voor het KES gedefinieerde Jess functies worden uitgevoerd om bijvoorbeeld kleurafstand te meten.

Een procedurele engine heeft de volgende voordelen boven het expertsysteem:

- betere performance
- meer controle over analyse en manipulatie van de kaartlagen

Als nadelen kunnen worden aangemerkt:

- Programmeerkennis nodig om functionaliteit uit te breiden. Ten opzichte van het definieren van regels voor een expert system shell is de benodigde kennis echter niet diepgaander.

in een overzicht:

eigenschap	Jess	Procedureel	Opmerking
snelheid	+/-	+	beiden functioneren redelijk snel, de procedurele versie is sneller, en kan beter worden geoptimaliseerd.
uitbreidbaarheid	+/-	++	De procedurele engine biedt veel mogelijkheden tot uitbreiding, de Jess engine minder. De vrijheid van het specificeren van een oplossings algortime is minder groot bij de Jess implementatie.
interactiviteit	+	+	De Jess engine kan blijven 'kijken' tot veranderingen in de kaart optreden. Voor de procedurele engine is het echter ook mogelijk veranderingen te signaleren
schaalbaarheid	+	+	Met schaalbaarheid wordt hier bedoeld een toename van de hoeveelheid data in het systeem.
Geschikt voor uitbreiding zonder technische kennis	-	-	Voor Jess is kennis van de CLIPS scripttaal nodig, voor de procedurele is Java kennis nodig.

7 Conclusies en aanbevelingen

7.1 Conclusies

Na het testen en de evaluatie zijn de volgende conclusies getrokken:

- Het bleek niet makkelijk om kartografische kennis te formaliseren, en om metrieken te vinden om probleemsituaties te detecteren.
- Bij het implementeren van de KES functionaliteit tijdens dit onderzoek, is een implementatie met behulp van een expert systeem niet de beste oplossing gebleken. Direct programmeren van de algoritmes levert een flinke snelheidswinst, gereduceerd geheugengebruik en beter te controleren uitvoering op. De geanticipeerde toegevoegde waarde van het redenatiemechanisme van het expert systeem kwam niet tot zijn recht. Het onafhankelijk van het programma aanpassen van de KES regels is ook op andere manieren dan gebruik makend van een expert systeem te realiseren.

7.2 Aanbevelingen

Bij het detecteren van probleemsituaties is soms gebruik gemaakt van heuristieken, bijvoorbeeld om overlap te 'voorspellen' door slechts te kijken naar dekkingspercentages van de lagen. In vervolgonderzoek kan worden gekeken of de toegepaste heuristieken kunnen worden verfijnd. De beschikbaarheid van topologische operatoren zou een winst zijn voor de detectie van problemen.

Er kan worden gekeken of de KES functionaliteit ingebouwd kan worden in een commercieel GIS pakket. Door de opzet van het KES framework moet het mogelijk zijn om bijvoorbeeld MapXtreme van MapInfo aan het KES te koppelen in plaats van GeoTools.

Het KES zou rekening kunnen houden met beperkingen in afbeeldingsplatform. Nu wordt er van uitgegaan dat het KES op een PC draait, met veel kleuren en voldoende grote resolutie. In principe is het KES zoals het voor dit project is geïmplementeerd geschikt om te draaien op een willekeurig platform dat Java ondersteuning biedt. Dit betekent dat behalve desktop PC's ook haldhelds en zelfs mobiele telefoons in aanmerking komen. Het KES zou de specifieke beperkingen, zoals een gebrek aan kleuren, in overweging kunnen nemen bij het optimaliseren van het kaartbeeld.

De regels in de expert system shell engine zijn geïmplementeerd als *forward chaining* regels. De regels constateren een conflict en zorgen dat dat ene conflict wordt opgelost. Een interessant alternatief zou zijn om gebruik te maken van *backward chaining*, zodat het KES de optimalisatie begint door te kijken wat er nodig is om een conflictvrije afbeelding te maken. Deze strategie zou kunnen helpen om gecorreleerde conflicten goed op te lossen. Gecorreleerd wil hier zeggen dat het oplossen van conflict één conflict een ander conflict beïnvloedt (introduceert). Backward chaining zou ervoor kunnen zorgen dat slechts die oplossing wordt gekozen die geen ander conflict introduceert.

8 Literatuurlijst

Alkemade, I., **beeldschermkartografie ten behoeve van multibron internet GIS**, TU Delft, 2000

Andrienko, G., N. Andrienko, H. Voss, J. Carter, **Internet mapping for dissemination of statistical information**. Computers, Environment and Urban Systems, no. 23 (1997), pp. 425-441.

Andrienko, G., N. Andrienko, **Interactive maps for visual data exploration.** International Journal of Geographical Information Science, vol. 13, no. 4. (1999), pp. 355-374.

Andrienko, G. en N. Andrienko, **Making a GIS intelligent: CommonGIS Project view.** AGILE conference, Rome 1999.

Buttenfield, Barbara P., David M.. Mark, **Expert systems in cartographic design.** In: D.R. Fraser (editor), Geographic Information Systems: the microcomputer and modern cartography. Oxford: Pergamon Press, 1991, pp. 129-150.

Cecconi, A., C. Shenton, R. Weibel, **Tools for cartographic visualisation of statistical data on the internet.** 19th International Cartographic Conference with ICA, Ottawa, August 14-21 (1999), session 07-C.

Clarke, Keith C., **Analytical and computer cartography (2nd edition).** Englewood Cliffs : Prentice-Hall, 1994.

Cox, S., Cuthbert, A. et al. (editors), **Geography Markup Language, version 2.0 reccomendation**, 2001, http://www.opengis.org/techno/specs

Doyle, A., **OpenGIS Web Map server interface implementation specification (revision 1.0.0).** OpenGIS project document 00-028, 1999.
http://www.opengis.org/techno/specs/00-028.pdf

Friedman-Hill, E. J., **JESS, The Java Expert System Shell**, SAND--98-8206, November 1997

Shea, K.S., McMaster, R.B., **Elements of cartography**, NewYork, John Wiley.

Kotte, I.F.A., **Een kartografisch expert systeem ten behoeve van presentatie van gedistribueerde geografische informatie**, Delft 2001

Kraak, M., Brown, A., **Web Cartography, developments and prospects**, Londen, Taylor and Francis, 2001

Kraak, M., Ormeling, F.J., **Cartography: Visualisation of spatial data**, Essex England, 1996

Lake, Ron, Adrian Cuthbert (editors), **Geography Markup Language.** version 1.0, 2000.
http://www.opengis.org/techno/specs/00-029/GML.html

MacEachren, Alan M., **How maps work**, New York: The Guilford Press (1995)

Müller, J.C. (editor), Lagrange, J.P. (editor), **GIS and generalization**, Taylor & Francis, Londen 1995

Nebert, D.D. (editor) , **The SDI cookbook draft 1.0**, www.gsdi.org, 2000

Poynton, C.,**Color FAQ**, http://www.inforamp.net/~poynton/, 1999

Rengelink, M., **Ontwikkeling van een Kartografisch Expert Systeem voor internet GIS.** Casestudy, TUDelft, Faculteit Civiele Techniek en Geowetenschappen, afdeling Geodesie, 1999.

Russel, S., Norvig, P., **Artificial Intelligence, A modern approach**, Prentice Hall, 1999

Schans, R. van der, **GIS-kartografie.** Delft: Technische Universiteit Delft, Faculteit Civiele Techniek en Geowetenschappen, Afdeling Geodesie, 1999.

Tuinman, Frank, Peter van Oosterom, **Geo-Shop, een multiserver internet-GIS**. Geodesia, no. 4 (1997), pp. 165-174..

Zhan, F.B., Barbara P. Buttenfield, **Object-oriented knowledge bases symbol selection for visualising statistical information.** International Journal of Geographical Information Systems, vol. 9, no. 3. (1995), pp. 293-315.

8.1 URL's

De volgende internet adressen zijn in het kader van dit verslag bezocht.

#	URL	beschrijving
1	www.graticule.com	GIS op handheld devices
2	www.opengis.org	Opengis consortium
3	www.kartoweb.itc.nl/webkartography/webbook	Site horend bij [kraak 2001]
4	allanon.gmd.de/and/and.html	Descartes GIS browser
5	www.ghg.net/clips/CLIPS.html	CLIPS expert system shell site
6	herzberg.ca.sandia.gov/jess/	JESS Java expert system shell
7	www.statkart.no/isotc211/welcome.html	TC211 site
8	www.ionicsoft.com	OpenGIS compliant software, o.a. servers
9	www.w3.org	world wide web consortium, o.a. SVG en XML standaard.
10	www.gsdi.org	Global Spatial Data Infrastructure organisatie die op globaal nivo ruimtelijke gegevens infrastructure probeert te faciliteren
11	www.ncgi.nl	Nationaal Clearinghouse Geo-Informatie, zoekmachine voor Geo-datasets.
12	GeoTools.sourceforge.net	GeoTools website

9 Bijlagen

9.1 Kernbegrippen Jess

Hieronder een beknopt overzicht van belangrijke begrippen zoals gebruikt in de scripttaal van Jess en CLIPS. Het overzicht is opgenomen om de lezer die onbekend is met deze taal te helpen de scripts in dit verslag beter te kunnen lezen.

Atoms zijn is Jess vergelijkbaar met identifiers in andere programmeertalen. Ze kunnen bestaan uit letters, cijfers en een aantal speciale karakters zoals #,_,-.
Voorbeelden zijn:
`teller, start_maand, deelnemer_12`.

Lists bestaan uit een tweetal haken ( ) met daartussen nul of meer elementen. Elementen kunnen atoms, nummers of andere lists zijn.
Voorbeelden zijn: `(a b c), ( ), (+ 1 2)`.

Variables zijn atoms die voorafgegaan worden door een vraagteken, en waaraan een waarde kan worden toegekend. Toekennen van een waarde kan met behulp van de `bind` funktie. Een voorbeeld van een variabele: `?x`
Voorbeeld van toewijzing van een waarde aan `?x`:
```
(bind ?x 100)
```

Functions zijn een bijzondere toepassing van lists. Uitvoerbare code in Jess wordt met behulp van functieaanroepen in de vorm van lists genoteerd. Hierbij wordt de prefix notatie aangehouden, het eerste element (de *head*) van de list is de functie, gevolgd door parameters.
Voorbeelden:
```
(+ 1 3)
(verplaats_item, 10, (lengte element2)).
```

Facts zijn in een kennissysteem de *feiten* waarop regels worden toegepast (gematched). Facts kunnen, net als in objectgeorienteerde programmeertalen, attributen (ook properties of fields genoemd) hebben. Binnen Jess kunnen deze attributen op twee manieren worden opgenomen in een feit. Er zijn *ordered facts* en *unordered facts* . Bij unordered facts zijn de attributen van een naam voorzien, bij ordered facts hangt de waarde van een attribuut af van de positie in de list van het feit. Attibuten van een unordered fact worden *slots* genoemd.
Een voorbeeld van een ordered fact: `(car blue 5)`
Een voorbeeld van een unordered fact: `(car (color blue) (doors 5))`

Rules oftewel regels bepalen het gedrag van het kennissysteem. Ze zijn enigszins vergelijkbaar met `IF .. THEN` statements. Op basis van feiten die matchen op de Left Hand Side (LHS) oftewel het `IF` gedeelte van de rule wordt aktie ondernomen.

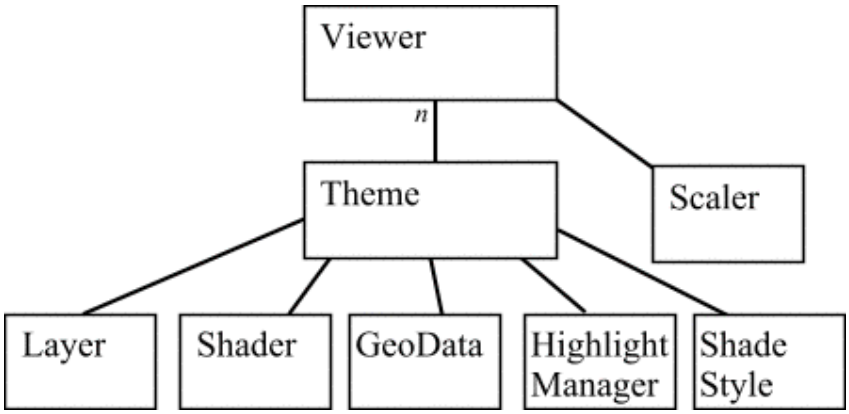
Voorbeeld van de structuur van een rule:
```
(defrule KleurConflict
    (Kleur-Conflict-Is-Aanwezig)
    =>
    (Los-Kleur-Conflict-Op))
```

Voor uitgebreidere informatie zie de website van Jess waar onder andere handleidingen en een FAQ te vinden zijn over Jess. Zie [URL 6]. In dit KES project is gewerkt met Jess versie 5.1.

9.2 Objectmodel GeoTools

GeoTools is een open-source GIS viewer. De viewer is gebruikt als visiualisatie-engine van het Kartografisch expertsysteem.

De viewer is opgebouwd uit de volgende hoofdcomponenten:



De compontenten uit dit model die relevant zijn voor het KES project worden hieronder genoemd:

Viewer
De viewer is het centrale component binnen GeoTools. Het dient als manager voor gebruikersinteractie en als manager van de kaartthema's.

Theme
Een kaartthema in GeoTools bestaat uit een kaartlaag en een aantal modules die de laag afbeelden en interactief maken middels shaders en highlightmanagers.

Layer
Een layer bevat geografische features. Binnen GeoTools kunnen layers uit verschillende bronnen worden gelezen, bijvoorbeeld uit GML data of uit Esri shapefiles.

9.3 Verklarende Woordenlijst

Hieronder worden een aantal begrippen en afkortingen uit de vakgebieden van kartografie, expertsystemen, internet en GIS toegelicht.

GIS systeem	Geografisch Informatie Systeem – KW 10.1.19: "Systeem voor het opslaan, controleren, integreren, manipuleren, analyseren en weergeven van geografische informatie, bestaande uit een bestand met aanduidingen van de posities en eigenschappen van de gegevens en de benodigde op de analyse gerichte programma's."

KES	Kartografisch Expert Systeem.
CIE	Commission Internationale de l'Eclairage, commissie die standaarden maakt voor colorimetrie en fotometrie.
DKM	Digitaal kartografisch model De presentatie van een DLM middels grafische primitieven zoals vlakken, lijnen en punten.
DLM	Digitaal landschapsmodel. De beschrijving in digitale vorm van geometrie, topologie en attribuutwaarden van geografische objecten.
ESRI	fabrikant van GIS software
GDMC	Geo-Database Management Centre, onderzoeksinstituut van de TU Delft.
GML	Geography Markup language, een XML grammatica om features in vast te leggen. Huidige officiële versie is 1.0 [URL 9], versie 2.0 heeft status recommendation.
OGC	Open GIS consortium
SVG	Scalable Vector Graphics, een XML-gebaseerde standaard om vectorgraphics, animatie en interactie in te specificeren. De status van de standaard is "v1.0 candidate recommendation" [URL 10]
XML	eXtensible Markup Language, de basis voor andere Markup standaards zoals XHTML, GML en SVG
XSL	eXtensible Stylesheet Language, een taal om stylesheets in te specificeren.
XSLT	XSL Transformations (XSLT), onderdeel van de XSL standaard, bedoeld om een XML document in een ander XML document te transformeren.

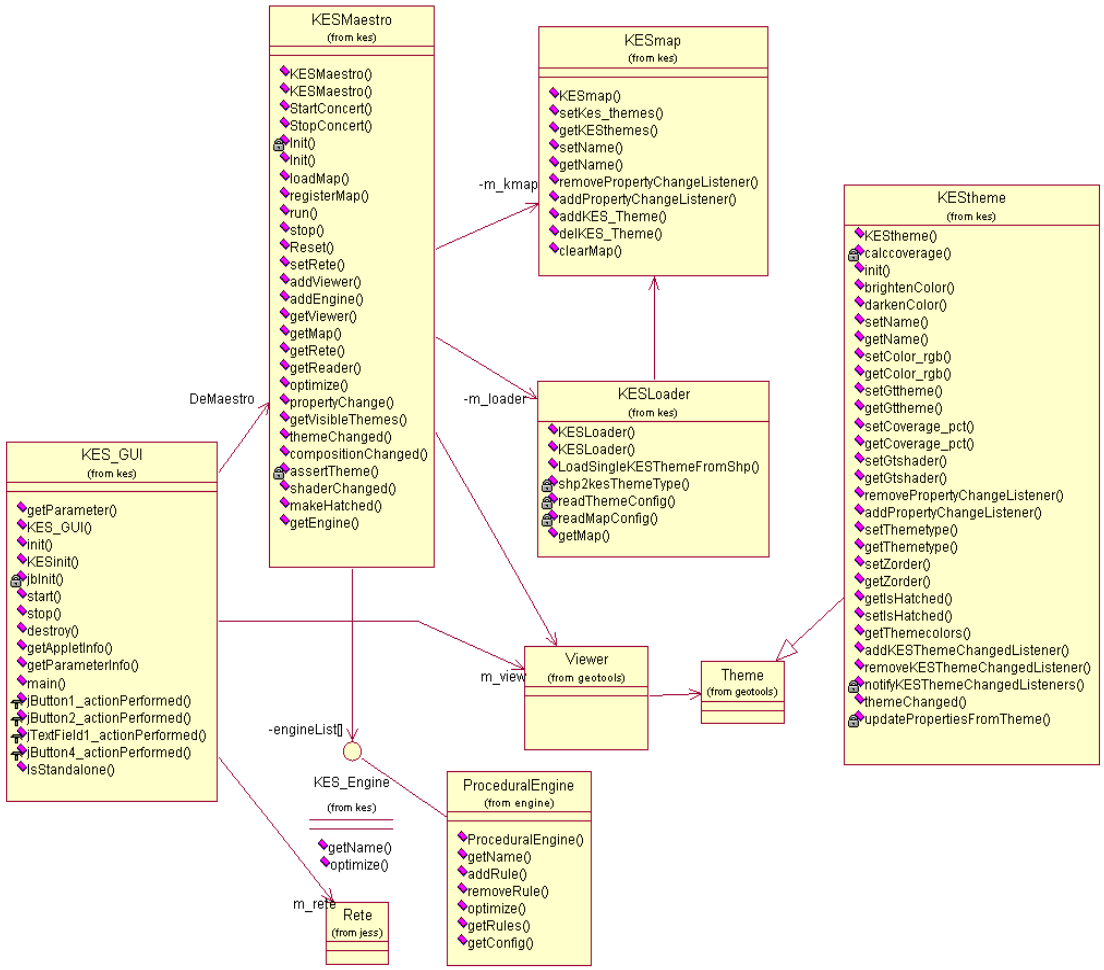
9.4 Sourcecode

9.4.1 "Proof of Concept" met Jess

layer3.clp source

```
;;;======================================================
;;; Voorbeeld kleurconflict oplossen:
;;;
;;; Vier kaartlagen met elk drie voorkeurskleuren worden
;;; als facts in het kennissyteem gebracht.
;;; De kleurenconflict rule probeert elke laag
;;; van een unieke kleur te voorzien door bij een conflict
;;; de volgende voorkeurskleur te kiezen.
;;; Als alle voorkeurskleuren van een laag conflicten
;;; blijken te geven, kiest dit script een random kleur
;;; voor deze laag.
;;; Het script eindigt als de lagen unieke kleuren hebben.
;;;======================================================

(reset)

(deftemplate MapLayer
 (slot name)
 (slot type)
 (multislot colors))

(defrule ppLayer "drukt layers af"
(declare (salience 100))
    (MapLayer (name ?n) (type ?t) (colors $?c))
    =>
    (printout t "layernaam: " ?n ", type: " ?t ", kleuren: " $?c crlf))

(defrule KleurenConflict "Checkt op gelijke kleuren"
    ?fact1 <- (MapLayer (name ?n) (type ?t) (colors ?a1 $?c1))
    ?fact2 <- (MapLayer (name ?m&~?n&:(< 0 (str-compare ?n ?m))) (type ?t) (colors
?a1 $?c2))
    =>
    (printout t "!----  Dubbele kleuren voor: " ?n " " ?m crlf)
    (printout t "!----  kleuren van layer: " ?m " worden aangepast van " ?a1 $?c2 "
naar " $?c2 ?a1 crlf)
    (retract ?fact2)
    (assert (MapLayer (name ?m) (type ?t) (colors $?c2 (long(/ (random) 1000)) )))
)

(defrule KleurenUniek "vuurt als alle eerste kleuren uniek zijn"
    (MapLayer (name ?n) (type ?t) (colors ?a1 $?c1))
    (not (exists (MapLayer (name ?m&~?n&:(< 0 (str-compare ?n ?m))) (type ?t)
(colors ?a1 $?c2))))
     =>
    (printout t "!----  Geen Layers met eerste kleur " ?a1 " meer behalve layer: "
?n  crlf) )


;De volgende 4 lagen hebben dezelfde 3 voorkeurskleuren.
;een zal dus een random kleurwaarde moeten krijgen om kleurconflict te vermijden
(assert (MapLayer (name Rijks-Wegen)       (type line)   (colors 2 6 8) ))
(assert (MapLayer (name ProvinWegen)       (type line)   (colors 2 6 8) ))
(assert (MapLayer (name OnverhWegen)       (type line)   (colors 2 6 8) ))
(assert (MapLayer (name PerceelGren)       (type line)   (colors 2 6 8) ))

(run 200)
(facts)
```

9.4.2 KES Java objectmodel



9.4.3 KES sourcecode

De Java sourcecode van het complete KES is te groot om hier te worden opgenomen. De sourcecode is in het bezit van het GDMC.

9.5 KES Handleiding

KES manual

versie: 1.1
auteur:Ilmar Kotte
datum: juni 2002

Inhoud:

Introductie

Het KES (Kartografisch Expert Systeem) werd ontwikkeld om een optimaal kaartbeeld te maken van een aantal kaartlagen die van verschillende bronnen afkomstig kunnen zijn.

De applicatie bestaat uit een GIS viewer component (GeoTools), een expert system shell (Jess), een procedurele KES engine en Java code om beiden met elkaar te laten communiceren. De applicatie is interactief, de gebruiker kan manipuleren met kaartthema's en scripts om effecten van wijzigingen en de reactie van het KES erop te onderzoeken.

Opstarten

Voor installatie instructies zie het `README.TXT` bestand in het applicatie zipbestand. In elk geval nodig voor uitvoering van de applicatie is de installatie van een recente Java Runtime Environment (JRE), aanbevolen is versie 1.3.1_02.

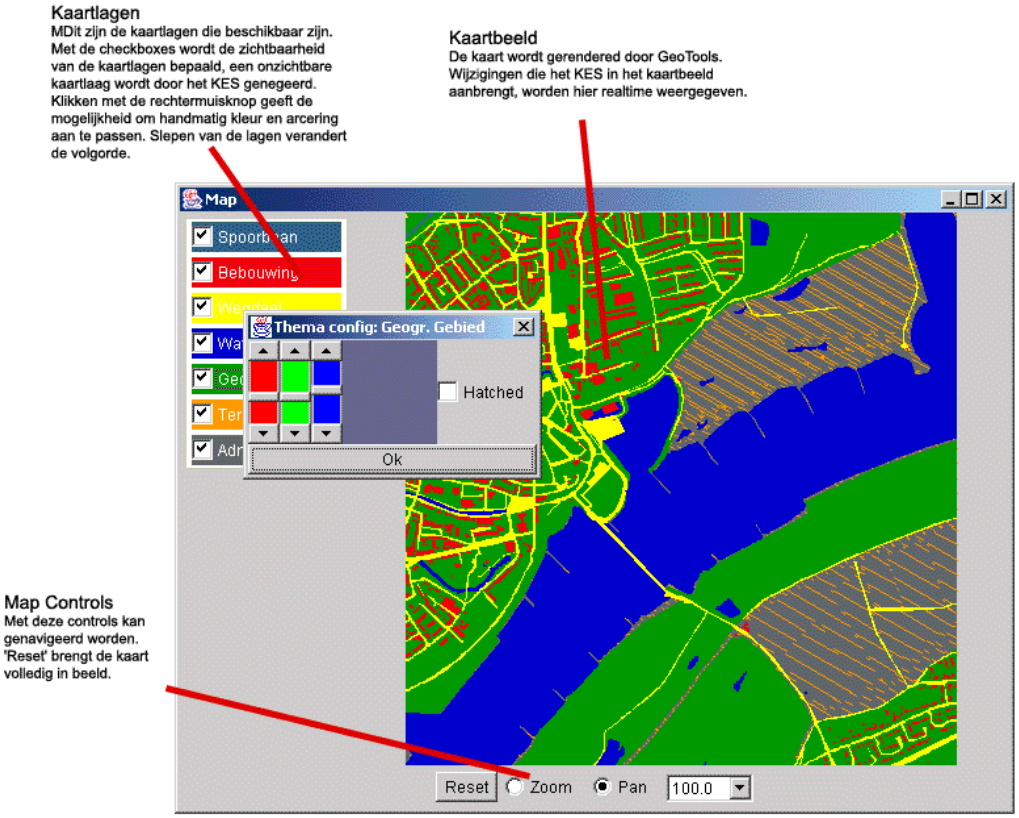
Na het uitpakken en installeren van de applicatie bevindt zich in de applicatiedirectory een bestand StartKES.bat. In dit bestand worden middels commandline parameters het opstartscenario en het optimalisatiescript vastgelegd. Door deze batchfile te editen kunnen deze worden gewijzigd, zodat bijvoorbeeld een andere themaset kan worden geladen. Voor details van deze bestanden, zie kopje bestanden.

In het opstartvenster (de operating system shell, het 'DOS' venster) worden meldingen van het KES gelogd. Hier is bijvoorbeeld de communicatie tussen Geotools en Jess te volgen. De applicatie zelf verschijnt in twee aparte vensters, een met GIS viewer en een met Jess interface (zie screenshot met uitleg). De applicatie is af te sluiten door het venster met de engine interfaces (zie beneden) af te sluiten op de gebruikelijke Windows manier.

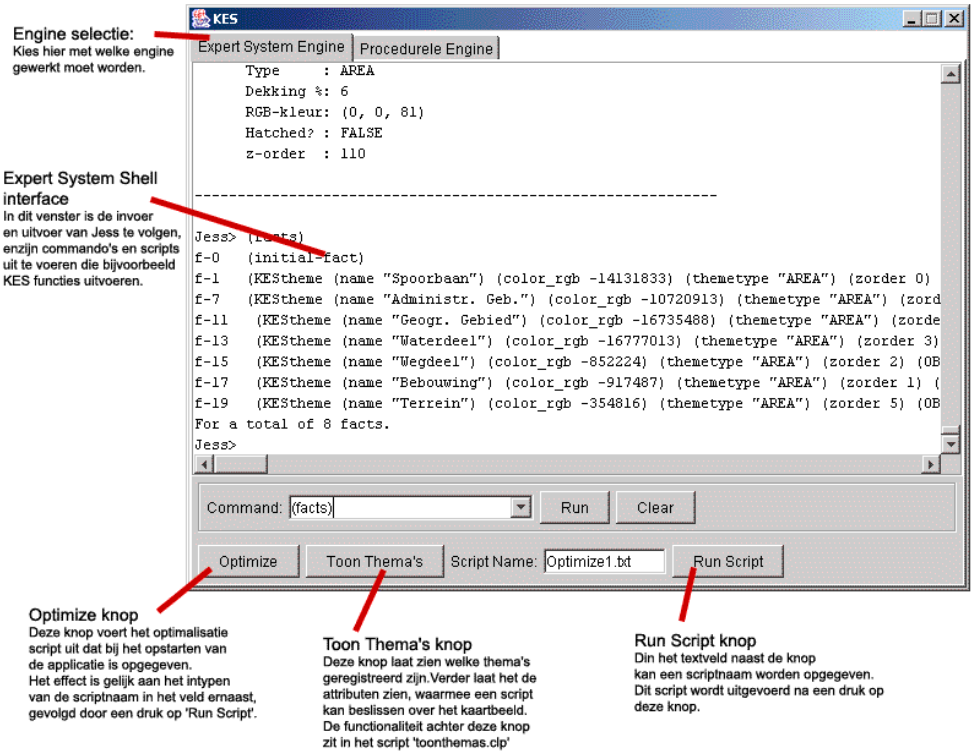
Interface

Hieronder worden de elementen uit de interface van het KES toegelicht.

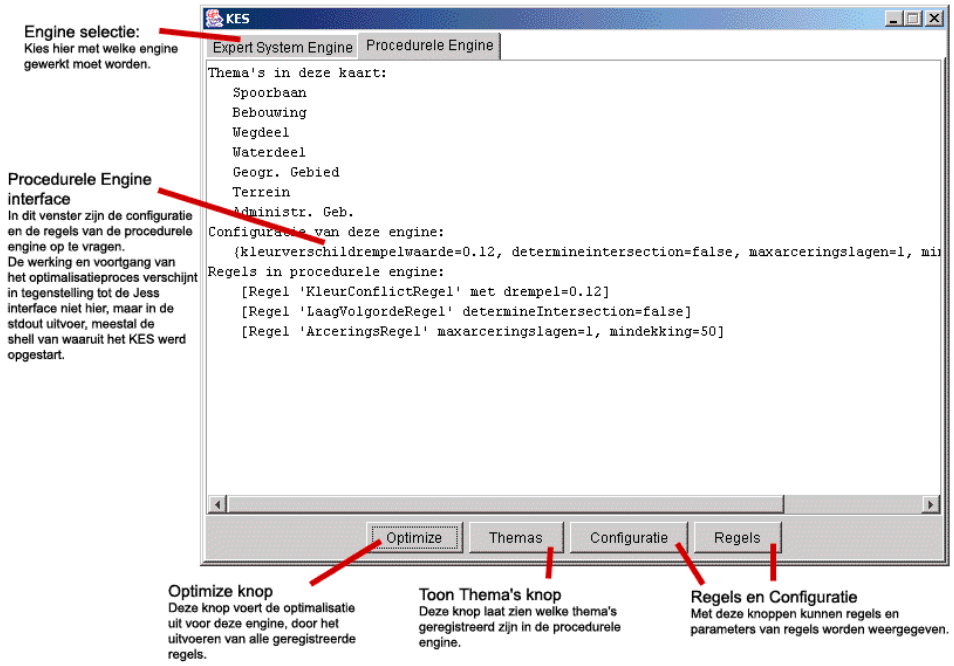
GIS viewer scherm



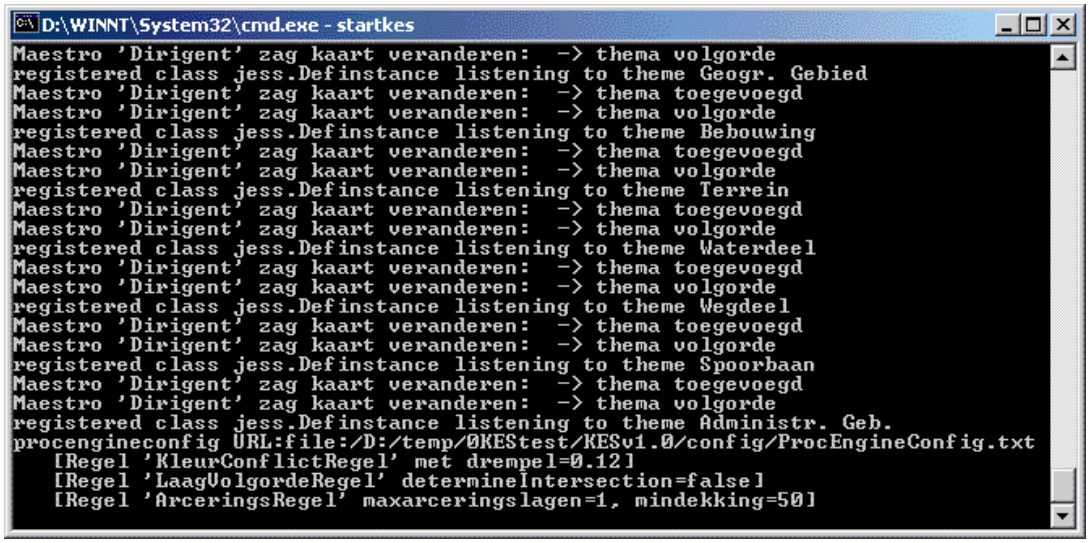
Jess KES engine scherm



Procedurele KES engine scherm



Command line log



Bestanden

Het KES gebruikt vier typen bestanden als invoer:

- een Scenario file
- een optimalisatiescript
- een configuratiebestand voor de procedurele engine
- een of meer Esri Shapefiles met themainformatie.

De scenariofile is een XML file met een eigen XML schema definitie. Het scenariobestand is met een teksteditor te editen, maar beter is om een tool zoals XMLspy te gebruiken, die het bestand kan valideren aan de hand van het schema. Het schema dat wordt gebruikt heet `KES02.xsd` in de subfolder `/config`. Als via de *-scenario* commandline parameter in de `StartKES.bat` batchfile geen scenariobestand wordt opgegeven, dan wordt standaard naar de file `config/default_scenario.xml` gezocht.

Het optimalisatiescript is een tekstbestand. Het bevat expertsystem script code, in een formaat compatible met CLIPS. Voor uitgebreide toelichting zie de site van Jess, waar een manual beschikbaar is. Als via de *-optimizescriptname* commandline parameter in de `StartKES.bat` batchfile geen optimalisatiescript wordt ogegeven, dan wordt standaard naar de file `jess/default_optimize.txt` gezocht.

Het configuratiebestand voor de procedurele engine is ook een tekstbestand. Het bevat parameters voor configuratie van de procedurele engine, in properties formaat. (variabele=waarde). Als via de *-procengineconfig* commandline parameter in de `StartKES.bat` batchfile geen configuratiebestand wordt ogegeven, dan wordt standaard naar de file `config/ProcEngineConfig.txt` gezocht.

De shapefiles moeten staan op de locatie zoals ze in de scenariofile staan. De scenariofile geeft een relatieve locatie ten opzicht van de applicatiedirectory en bevat niet de bestandsextensie. Voorbeeld: de shapefile `awscntry.shp` in de applicatiesubdirectory `ArcIndia` zou als volgt in de scenariofile moeten staan: `ArcIndia/awscntry`.

De scenariofile, de scriptfile en de configuratiefile van de procedurele engine kunnen dus veranderd worden. Wijziging in een optimalisatie scriptfile kunnen worden doorgevoerd terwijl de applicatie loopt (met een teksteditor buiten de applicatie om). Een scenario kan alleen opnieuw worden geladen door opnieuw opstarten van de applicatie.