FACULTEIT DER RECHTSGELEERDHEID

ICT en wetgeving

Emile de Maat

Tom van Engers

Radboud Winkels

Datum 22-12-2008

Inhoudsopgave

Samenvatting

Dit rapport beschrijft een onderzoek uitgevoerd in opdracht van het WODC. Het doel van dit onderzoek was om een (deel)antwoord te geven op de vraag *Hoe kan de transparantie en productie van (wijzigings)wetgeving worden verbeterd met behulp van ICT?*

Voor dit onderzoek was de opdracht om een omgeving te ontwikkelen voor digitale ondersteuning bij het maken van wet- en regelgeving. Deze omgeving moest de volgende functionaliteit bieden:

- Een redactieomgeving waarin men doorlopende teksten kan importeren en bewerken.
- Het automatisch genereren van wijzigingswetten op basis van twee doorlopende teksten.
- Doorlopende (semi-)automatische controle op samenloop tussen wetten die in procedure zijn.
- De automatische consolidatie van wijzigingswetten in elke stand van de procedure
- Werkstroombesturing
- Een webviewer waarmee de teksten ook zonder de functionaliteiten om teksten te creëren (ondersteund in de editor) geraadpleegd kunnen worden.

Werkstroombesturing is geen nieuwe toevoeging binnen het wetgevingsproces; bij de verschillende ministeries zijn er al diverse systemen voor werkstroombesturing in gebruik. Derhalve is besloten om voor de werkstroombesturing niet opnieuw software te ontwikkelen, maar bij implementatie in een later stadium aan te sluiten op een dergelijke systeem.

Tijdens het onderzoek is intensief samengewerkt met een werkgroep bestaande uit verschillende wetgevingsjuristen van diverse departementen. Zij hebben feedback geleverd op de opzet van de praktijkproeven en de tussentijdse onderzoeksresultaten. Op deze manier is aansluiting gehouden met wensen en aanbevelingen in de praktijk. Concreet heeft dit geleid tot twee extra gewenste functies waarvoor de redactieomgeving ondersteuning moet bieden:

- Het kunnen weergeven van verwijzingen naar de tekst die bewerkt wordt.
- Het automatisch hernummeren van de onderdelen van een voorsteltekst.

Uiteindelijk is een proefomgeving gerealiseerd met daarin de oorspronkelijke punten (exclusief de werkstroombesturing) en de twee toegevoegde punten. De basis van deze proefomgeving is dat alle documenten worden opgeslagen in een XML formaat (MetaLex/CEN). Hierbij wordt niet alleen de tekst opgeslagen, maar ook extra informatie zoals relevante datums, verwijzingen en structuurinformatie over het document (metadata). Deze extra informatie maakt het mogelijk om de gevraagde functies te realiseren, hetgeen met alleen tekstopslag niet mogelijk zou zijn geweest.

Het XML formaat is alleen bedoeld voor verwerking door de computer. Alle gebruikers krijgen normale tekst te zien, zonder toegevoegde computer codes. Mensen die de teksten willen raadplegen kunnen gebruik maken van de viewer. Deze viewer zet het XML bestand om naar leesbare tekst (een zogenaamde XSLT transformatie).

Wetgevingsjuristen die de teksten schrijven gebruiken de redactieomgeving. Deze omgeving is What-You-See-Is-What-You-Get (WYSIWYG), wat inhoudt dat ook de omgeving de uiteindelijke tekst laat zien, en niet de XML versie. Wel wordt van de wetgevingsjurist verwacht dat hij de structuurinformatie en andere metadata toevoegt aan het document. Ook hiervoor is het niet nodig om over XML kennis te beschikken. De redactieomgeving levert sjablonen aan waarmee de markeringen worden aangebracht. Effectief houdt dit in dat de wetgevingsjurist bij aanvang van elk nieuwe structuurdeel (bijvoorbeeld een hoofdstuk) een knop moet aanklikken om aan te geven dat hij dat nieuwe structuurdeel begint.
Voor dit extra werk krijgt de wetgevingsjurist ook het een en ander terug. Om te beginnen zijn er niet alleen sjablonen voor de structuur, maar er zijn ook sjablonen voor regelmatig terugkerende teksten, zoals degenen die in de *Aanwijzingen voor de regelgeving* worden voorgeschreven. Daarmee zijn dergelijke teksten makkelijker in een nieuw voorstel in te voegen.
Zoals gezegd maakt de structuurinformatie ook de bovengenoemde functionaliteit mogelijk, die ook bedoeld is voor de ondersteuning van de wetgevingsjurist. Automatisch hernummeren vermindert de hoeveelheid werk die in sommige voorstellen gepaard gaat met het hernummeren van artikelen. Weergave van verwijzingen naar het document dat wordt bewerkt maakt het makkelijker om het overzicht te behouden van de mogelijke gevolgen van een aangebrachte wijziging. Ook is er de mogelijkheid om wijzigingswetten automatisch te genereren. Dit houdt in dat de wetgevingsjurist zijn wijzigingen direct kan aanbrengen in de tekst, en deze niet hoeft te formuleren als wijzigingsinstructies. Deze laatste stap wordt door de computer uitgevoerd. Het voordeel hiervan is niet alleen dat het schrijven van de wijzigingen simpeler en overzichtelijker wordt, maar dat de computer ook een overzicht heeft van de aangebrachte wijzigingen. Hierdoor wordt het mogelijk om samenloop op te sporen, door de aangebrachte wijzigingen te vergelijken met andere wijzigingen (mits die ook op deze wijze zijn geregistreerd).

De wetgevingsjurist is niet de enige die mogelijk profijt heeft van dit proces. Doordat de documenten in XML worden opgeslagen, zijn ze makkelijker te doorzoeken en om te zetten naar andere formaten. Latere publicatie, bijvoorbeeld naar *wetten.nl* zal minder arbeidsintensief zijn.
De aanpak voor de wijzigingswetten heeft tot voordeel dat door het apart opslaan van de wijzigingen, in een formaat dat begrijpelijk is voor de computer, het mogelijk is om op elk moment een consolidatie-on-the-fly te maken, dat wil zeggen dat een geconsolideerde tekst gemaakt kan worden die het effect van de wijzigingen weergeeft op ieder gewenst moment. Zo'n consolidatie kan ook betrekkingen hebben op een mogelijk scenario: hoe komt de wet er uit te zien als dit voorstel wordt aangenomen? Zo'n consolidatie-on-the-fly maakt het mogelijk voor bijvoorbeeld kamerleden en belangengroepen om de gevolgen van een wijzigingswet sneller te overzien; zij hoeven dan niet de instructies uit de wijzigingswet toe te passen, maar zien direct het resultaat.

De werkgroep was enthousiast over de redactieomgeving, en zag veel mogelijkheden voor het gebruik van deze omgeving in het wetgevingsproces. Wel was er de kanttekening dat de gebruikersinterface van de proefomgeving, hoewel die in de loop van het onderzoek sterk verbeterde, nog niet voldeed aan de wensen en verwachtingen van toekomstige gebruikers.

Dit onderzoek heeft aangetoond dat de functionaliteit zoals die in de opdracht werd geschetst, haalbaar is. Gecombineerd met onder andere de enthousiaste reacties uit de werkgroep is de aanbeveling dan ook om deze ontwikkeling voor te zetten.

Zoals aangegeven is er binnen dit project een proefomgeving ontwikkeld. Deze omgeving laat zien dat de functionaliteit mogelijk is, maar is niet beoogd om zonder doorontwikkeling als productiesysteem ingezet te worden. Er zijn extra inspanningen nodig om dit in de toekomst mogelijk te maken.

Om te beginnen kan de gebruikersinterface nog verder worden verbeterd zodat deze beter aansluit bij de verwachtingen van de gebruikers. Dit is een belangrijk onderdeel van de omgeving; succes van de omgeving staat of valt bij de kwaliteit van de gebruikersinterface. De huidige interface volstaat voor demo doeleinden, maar de onderzoekers achten deze in de huidige vorm nog onvoldoende gebruiksvriendelijk voor een rimpelloze ingebruikname in de hectische omgeving van de wetgevingsjurist.

De opsporing van samenloop, waarbij twee parallelle wijzigingen met elkaar botsen, is nog in een relatief pril stadium. Samenloop wordt op dit moment gedetecteerd als twee wijzigingen botsen qua structuur. De computer zal voorlopig niet in staat zijn om andere vormen van samenloop met zekerheid te detecteren; daarvoor is begrip van de tekst nodig. Wel kunnen er vaker waarschuwingen omtrent mogelijke samenloop worden gegeven. De huidige omgeving geeft die nog niet; extra onderzoek naar de mogelijkheden op dit gebied is gewenst.
De andere functionaliteiten zijn verder uitgewerkt, maar kunnen nog wat gepolijst worden voordat ze worden uitgezet.

Tot slot is er onderzoek nodig om deze omgeving in te kunnen zetten binnen de huidige processen. Integratie in de huidige processen zal enige inspanning vereisen. De editor is tot nu toe ingezet in stand-alone situaties; voor een systeem met verschillende, samenwerkende gebruikers zal een grotere architectuur moeten worden ingericht. Voor de aansluiting op bestaande systemen (werkstroombesturing en contentmanagement) zal specifieke software moeten worden ontworpen.
Naast de technische aansluiting moet de redactieomgeving en het gebruik daarvan ook worden ingebed in de bestaande gang van zaken. Dit vereist mogelijke aanpassingen in de processen.

De onderzoekers menen dat op de huidige omgeving een goede basis biedt voor doorontwikkeling tot een uiteindelijk in productie te nemen omgeving voor de wetgevingsjurist. Ook heeft de ontwikkeling ervan en de discussies over de functionaliteiten de beoogde gebruikers geholpen zich een reëel beeld te vormen van een dergelijke omgeving en de impact die het werken daarmee op termijn zal hebben bij hun werkzaamheden.

Summary

This document reports on research commissioned by the WODC. Goal of this research was to provide a (partial) answer to the question *In what ways can transparency and production of (change) laws be improved with the aid of ICT?*

Within this project, a software environment would be developed for the drafting of regulations. This environment would have the following functionalities:

- An editor in which complete texts may be imported and edited.
- Automated generation of change law based on two versions of a text.
- Semi-automatic control on clashes between change laws that are being developed.
- Automated consolidation of change laws, at any point of the procedure.
- Workflow management.
- A web viewer that allows for the viewing of texts without the functionalities for editing texts.

Workflow management is not a new addition to the legislative drafting process; the various ministries already use workflow management systems. Therefore, it has been decided not to develop new software for workflow management, but to connect to existing systems in a later phase of the implementation.

During the project, there has been intensive collaboration with a working group consisting of several legal drafters from different departments. They have given feedback on the early prototypes and the initial research results. In this way, the research connects to the wishes and recommendations from actual users. From the feedback two additional desired functionalities for the editor emerged:

- Showing of references to the text being edited.
- Automatic renumbering of parts of a proposal.

A prototype has been developed which includes the original requirements (excluding the workflow management) and the two additional requirements. This prototype is based on storage of all documents in an XML format (METAlex/CEN). In this format, not only the text is stored, but extra information (metadata) as well, such as relevant dates, references and information on the structure of the document. This additional information makes it possible to implement the requested functions, which would not be possible with text-only storage.

The XML format is intended for processing by a computer only. All human users will be shown regular texts, without added computer codes. People wanting to read texts can use the viewer, which transforms the XML file to readable text (using an XSLT transformation). Legal drafters will use the editor. This editor is What-You-See-Is-What-You-Get (WYSIWYG), meaning that it shows the final texts as well, and not the XML version. However, it is expected from the legal drafter that he adds the structure information and other metadata to the document. This requires no knowledge of XML. The editor supplies templates which will apply the correct markings. In effect, this means that the legal drafter has to click a button each time he introduces a new part (such as a chapter).

In exchange for this added work, the legal drafter will receive certain benefits. First of all, there are not only templates for structure, but for frequently used texts as well, such as the text prescribed by the *Guidelines for Legal Drafting* (Aanwijzingen voor de regelgeving). These templates make it easier to add such texts to a new proposal.
As mentioned, the added structure information also enables the addition of the required functionalities, which are meant to support the legal drafter as well. Automated renumbering reduces the amount of work which is sometimes required when renumbering articles. Displaying references to the edited document makes it easier to keep an overview of the consequences of changes that are being made. In addition, there is the possibility of automated generation of change laws. This allows the legal drafter to make his changes directly in the text, instead of describing these changes by means of instructions for change. This last step is performed by the computer. This does not only have the benefit of making the drafting of change laws easier and more transparent, but also results in the computer having an overview of the changes that have been made. Based on this overview, detecting clashes with other change laws (that have been made using the same method) is possible.

The legal drafter is not the only person who benefits from this process. Because the documents are stored as XML, it is easier to search through them and to convert them to other formats. Later publication to i.e. *wetten.nl* will require less work.
The approach for change laws has the advantage that by storing the changes separately, in a file that can be interpreted by the computer, generating consolidations-on-the-fly becomes possible. This means that it is possible to create a consolidated text which shows the (textual) outcome of changes at any desired moment. Such a consolidation can also reflect a fictional scenario: what would the law look like if this proposal is accepted? Such a consolidation-on-the-fly makes it easier for i.e. members of parliament and stakeholders to quickly see the impact of a proposed change law; they are no longer required to apply the changes themselves, but can immediately view the resulting document.

The working group was enthusiastic about the prototype environment, and saw many possibilities for the use of this environment in the legal drafting process. The user interface of the prototype did not yet conform to the wishes and expectations of future users, despite advancements made during the project.

This project has shown that functionality as it was described in the assignment is possible. Combined with the enthusiasm of the test users, it leads to the recommendation to continue this development.

As mentioned, the software developed within this project is a prototype. The current software shows that the requested functionality is possible, but is not intended to for real use. More effort is needed to make this possible.

First of all, the user interface should be further improved to make it conform to the expectations of users. This is an important part of the software; a high quality user interface is an essential requirement for successful deployment of the software. The current user interface suffices for demo purposes, but the researches deem it insufficiently user friendly for use in the turbulent environment of the legal drafters.

The detection of clashes is still in a relatively early stage. Clashes are currently detected if two changes interfere with each other with regard to structure. More complicated clashes

will not be detected by the computer for the time being, as that would require understanding of the text. However, it is possible to give more frequent warnings for potential clashes. Currently, the software does not give such warnings; further research into this topic is desirable.
The other functionalities have been developed into a further stage, but some edges could be removed before the software is deployed.

Finally: research will be needed to deploy the software within the current processes. Integration with the current processes will require some effort. The software has so far been deployed in stand-alone situations. For a situation with several cooperating users, a bigger infrastructure is needed. For the connection to existing systems (workflow management and content management) specific software needs to be developed.
In addition to this technical integration, it will also be necessary to embed the software and the use of the software in the current practices. This may require adaptation of the current processes.

The researchers are of the opinion that the current prototype forms a solid basis for continued development of a production version of software for legal drafting. The development of this prototype and the discussion on its functionalities has helped the users to form a mental image of such software and the possible impact it may have on their work.

1. Inleiding

1.1. De vraagstelling

In onze samenleving is Informatie- en Communicatie Technologie (ICT) niet meer weg te denken. Bedrijven maken gebruik van databases voor de administratie en van webpagina's voor reclame en communicatie. Steeds meer mensen hebben een Internetaansluiting en zien het Internet als een belangrijke, zo niet hun belangrijkste, informatiebron.

Om het contact met de maatschappij te behouden is het belangrijk dat ook de overheid zich op het Internet begeeft. Dit gebeurt dan ook: veel van de communicatie van de overheid vindt nu (ook) plaats via websites. Voor het regelgevingproces zijn belangrijke voorbeelden hiervan sites als overheid.nl en wetten.nl. Hier worden veel documenten aangaande de regelgeving in digitale vorm beschikbaar gesteld. In de gewijzigde Bekendmakingswet is digitale publicatie zelfs het belangrijkste communicatiekanaal en vormt de digitale versie van de regelgeving de authentieke bron. Op deze wijze wordt ICT gebruikt aan het "eind" van een regelgevingtraject.

Maar ook bij het creëren van regelgeving kan ICT nuttig worden ingezet, zowel inhoudelijk, dat wil zeggen bij het tot stand brengen van de inhoud van regelgeving, als bij het efficiënt laten verlopen van het proces van regelgeving en de communicatie tussen de verschillende bij de totstandkoming betrokken actoren.

Dit rapport is de weerslag van een verkenning naar een meer effectieve en efficiënte vorm van ondersteuning van het proces voor het maken van (wijzigings)wetgeving. Het beoogt een antwoord te geven op de volgende onderzoeksvraag:

Hoe kan de transparantie en productie van (wijzigings)wetgeving worden verbeterd met ICT?

Bij de productie van wetgeving zijn veel personen en organisaties betrokken, ieder met specifieke rollen en verantwoordelijkheden. Mensen die direct betrokken zijn bij het schrijven van de wetteksten, zoals wetgevingsjuristen op de betrokken vakdepartementen en de Tweede Kamer, behorende tot de directe doelgroep van dit systeem. Zij zijn de mensen die met de omgeving moeten gaan werken. Daarnaast zijn er nog veel organisaties die baat hebben bij de resultaten van dit systeem: parlementariërs, medewerkers van het parlement, de Raad van State en Binnenlandse Zaken en Koninkrijksrelaties als verantwoordelijke voor de publicatie van de teksten. Het systeem kan ook goed worden ingezet voor andere regelgeving, hetgeen betekent dat bijvoorbeeld ook de Raad voor de Rechtspraak en de gemeentes als potentiële gebruiker kunnen worden aangemerkt. Aanleiding voor het onderzoek waarvan dit rapport de weerslag vormt zijn de internationale en nationale ontwikkelingen op het terrein van de digitalisering van het wetgevingsproces en de toegenomen mogelijkheden voor het ondersteunen van complexe taken met behulp van geautomatiseerde hulpmiddelen.

In eerste instantie betrof dit proces voornamelijk digitalisering van de publicatie van regelgeving, maar in toenemende mate zien we in Europa de tendens ook de digitale totstandkoming van regelgeving te willen ondersteunen. Ten behoeve van die digitale totstandkoming en beschikbaarstelling van regelgeving zijn in Europa, maar ook daarbuiten, inmiddels standaarden en ondersteunende gereedschappen ontwikkeld die het creëren en wijzigen van wetgeving en het kenbaar maken van nieuwe dan wel gewijzigde regelgeving aanzienlijk vereenvoudigen.
Voorbeelden van dergelijke systemen zijn het Australische EnAct, de Italiaans Norma editor in combinatie met de Norme-in-Rete XML standaard en het systeem de MetaVex-editor en de MetaLex/CEN XML standaard[1] die centraal staan in dit onderzoek. [2]

Al deze ontwikkelingen zijn gericht op het verbeteren van het wetgevingsproces, het vergroten van de transparantie en toegankelijkheid van regelgeving en het verbeteren van de uitvoeringpraktijk, met name waar het routinematige toepassing van het recht door uitvoeringsorganisaties betreft.

Het accent in dit onderzoek heeft in het bijzonder gelegen op het in de praktijk verkennen van de mogelijkheden om het opstellen van wetgeving te faciliteren. Daarbij is een systematiek gehanteerd die in het verlengde van het opstellen ook het publiceren en consolideren van wetgeving ondersteunt. Het ondersteunen van het werkproces zelf is slecht gedeeltelijk empirisch onderzocht. Desondanks kunnen op basis van dit onderzoek en op grond van een theoretische verkenning wel uitspraken worden gedaan over de mogelijkheden tot verdergaande vormen van document- en werkstroombeheersing. In het onderzoek zijn de voor de Nederlandse wetgevingspraktijk karakteristieke elementen meegenomen in samenhang met de kenmerken van de Europese wetgevingspraktijk.

Het onderzoek is begeleid door een commissie onder voorzitterschap van prof. mr. W.J.M. Voermans (zie Bijlage A). De onderzoekers hebben tijdens het onderzoek intensief samengewerkt met verschillende wetgevingsjuristen van diverse departementen. De feedback die zij hebben geleverd op de opzet van de praktijkproeven en de tussentijdse onderzoeksresultaten hebben er toe bijgedragen dat dit onderzoek naast bevindingen van algemene en theoretische aard toch vooral ook praktijkgericht is gebleven. Ook is het hierdoor mogelijk geweest een eerste aanzet te geven voor een in de praktijk bruikbare editingomgeving voor het maken van (wijzigings)wetgeving. Daarbij zijn specifieke kenmerken van de Nederlandse wetgevingspraktijk als uitgangspunt genomen, terwijl ook internationale ontwikkelingen, zoals op het gebied van standaardisatie van rechtsbronnen, zijn meegenomen.

---

[1] De MetaLex/CEN standaard is bedoeld als uitwisselingsformaat. Onder deze standaard kunnen meerdere meer gedetailleerde subschema's worden gehanteerd zoals in dit geval MetaLex/LS. Geautomatiseerde omzetting naar andere (XML) formaten zoals het BWB formaat is mogelijk.
[2] Zie bijvoorbeeld:
T. Arnold-Moore, *Information systems for legislation*, PhD Thesis, Melbourne, Australia, 1998.
M. Palmirani, B. Brighi, *An XML Approach for Italian Legislation Acts*, paper, JURIX 2002 Fifteenth Annual International Conference on Legal Knowledge and Information Systems, December 16-17, 2002, London,
A. Boer, R. Hoekstra, R. Winkels, T. van Engers, and F. Willaert, *Proposal for a Dutch Legal XML Standard*, paper, JURIX 2002 Fifteenth Annual International Conference on Legal Knowledge and Information Systems, December 16-17, 2002, London.

1.2. ICT voor de regelgeving

Nederland heeft een traditie hoog te houden als het gaat om het inzetten van ICT voor de regelgeving. In Nederland wordt op ruime schal gebruik gemaakt van mogelijkheden tot digitale publicatie van regelgeving en rechtspraak. Ook is er inmiddels een begin gemaakt met het stroomlijnen van het regelgevingproces. In Nederland wordt door de Tweede Kamer Parlis ontwikkeld, het parlementair informatiesysteem. Dit is een geautomatiseerd systeem voor documentbeheer, werkstromen en vergaderondersteuning. Eerder al heeft Oostenrijk een dergelijk systeem in gebruik genomen, genaamd E-LAW. Ook andere Europese landen kennen vergelijkbare systemen of hebben deze in ontwikkeling.

Op het terrein van de inhoudelijke ondersteuning bij het maken van wet- en regelgeving is inmiddels ook de nodige ervaring opgedaan. Al in het begin van de jaren negentig is geëxperimenteerd met zogenaamde “intelligente” applicaties, waarbij de computer met adviezen komt ten aanzien van een onderhanden stuk regelgeving. Een bekend Nederlands voorbeeld van een dergelijke toepassing is het LEDA systeem, ontworpen om wetgevingsjuristen te helpen bij het voldoen aan de Aanwijzingen voor de Regelgeving. Met LEDA hebben uitgebreide proefnemingen plaatsgevonden.
Ook in andere landen zijn dergelijke initiatieven ontplooid. In Vlaanderen is een uitbreiding voor Word ontwikkeld genaamd Systeem ter Ondersteuning van Legistiek en het Ontwerpen van Normen (SOLON). Dit is een verzameling macro’s voor MS Word die de gebruiker helpt bij het organiseren van de tekst van een nieuw stuk regelgeving. Daarnaast bevat SOLON (ondersteuning voor) een drietal kwalitatieve controles (taalkundig, formeel en materieel inhoudelijk). Een inhoudelijke test was ook aanwezig in de POWER omgeving die ontwikkeld is bij de Nederlandse Belastingdienst. Deze omgeving bevatte geen “schrijfhulp” voor wetgeving; de beoogde gebruiker was niet de wetgevingsjurist, maar een analist. Deze kan met de omgeving een model maken van de wet, waarna deze kan worden getest op consistentie en mogelijke effecten.

Een aantal van dit soort systemen heeft ingang gevonden in de rechtspraktijk, maar wel ging dit proces aanzienlijk langzamer dan de ontwikkelaars indertijd hoopten. De aansluiting met de rechtspraktijk bleek vaak ingewikkelder dan tevoren voorzien en bovendien is de wetgever om begrijpelijke redenen voorzichtig om taken over te laten aan software. Regelgeving is immers een belangrijk product waarvan de kwaliteit van groot belang is voor de uiteindelijke effectiviteit. Die kwaliteit is eveneens van belang voor de vertrouwensrelatie tussen de overheid en haar burgers. Bovendien komt regelgeving vaak onder tijdsdruk tot stand. Er is daarbij weinig tijd voor de betrokkenen bij wetgeving om zich een andere werkwijze eigen te maken.

Toch wordt inmiddels algemeen erkend dat goedwerkende ICT ondersteuning noodzakelijk is om fouten in de regelgeving te voorkomen, en de kwaliteit van de regelgeving omhoog te kunnen halen. Daarnaast kan dergelijke software de hoeveelheid werk die nodig is voor de creatie van regelgeving helpen verminderen. Het onderzoek naar en ontwerpen van systemen die het wetgevingsproces ondersteunen en de effectiviteit en efficiëntie verder kunnen verbeteren gaat dan ook onverminderd door.

Zoals reeds vermeld bestaat er inmiddels een aantal redactieomgevingen die inmiddels zodanig volwassen zijn dat ze het experimentele stadium hebben overstegen. De bekendste voorbeelden zijn de Italiaanse NIR-Editor, de Nederlandse MetaVex editor, de

eveneens Italiaanse Norma plug-in voor Word en het Australische EnAct. Deze omgevingen richten zich in eerste instantie vooral op het bewaken van de structuur van de regelgeving, zodat deze op een voor de computer te herkennen wijze kan worden opgeslagen. Dit maakt het eenvoudiger om deze teksten te publiceren, doordat de tekst aan de hand van deze structuurinformatie snel kan worden opgemaakt (als af te drukken tekst in digitaal leesbare vorm, bijvoorbeeld als hypertext (xHTML) op het Internet).

Met het EnAct systeem is in Tasmanië uitgebreide praktijkervaring opgedaan. Dit systeem slaat net als de eerder genoemde omgevingen niet alleen de tekst van de regelgeving op, maar ook de structuur. Daarnaast bevat het functionaliteit voor het ondersteunen van het schrijven van wijzigingswetten en het creëren van consolidaties. Een wetgevingsjurist kan een wijzigingswet aanmaken door de originele tekst te wijzigen. De wijzigingen worden door de computer opgespoord en opgeslagen. Op basis van de opgeslagen wijzigingen wordt een tekst voor de wijzigingswet gegenereerd. Door de opgeslagen wijzigingen toe te passen op de originele tekst kan een consolidatie worden gegenereerd. Deze functionaliteit treffen we ook bij de MetaVex editor.

Recent zijn al deze ervaringen (en meer) verzameld in een rapport, *ICT voor wetgeving*[3], dat ook de aanleiding vormt voor dit onderzoek. In dit rapport wordt een aantal aanbevelingen gedaan voor het gebruik van ICT in het Nederlandse wetgevingsproces. Er wordt onder meer aanbevolen de volgende functies te ontwikkelen:
– Het gemeenschappelijk maken van wetgeving, met een correcte structuur en nummering en voorzien van de juiste verwijzingen;
– De metadatering van wetgeving t.b.v. beleid, ontsluiting en toepassing;
– De consolidatie van wetgevingsvoortgang;
– Bulklevering van wetgeving en gerelateerde informatie volgens open standaarden.

Binnen dit project is getracht deze functionaliteit te ontwikkelen, om zo inzicht te krijgen in de haalbaarheid van een ICT-voorziening waarmee het wetgevingscreatieproces kan worden verbeterd en vergemakkelijkt. Ook zouden mogelijke knelpunten voor de implementatie van een dergelijke oplossing in dit project moeten worden gedetecteerd.

2. Gevolgde Aanpak

2.1. Opdracht

Het Leibniz Center for Law heeft een offerte uitgebracht om een proefomgeving voor het maken van wet- en regelgeving te ontwikkelen. Voor de proefomgeving waren de volgende componenten ingepland:

- Een wetgevingseditor waarin men doorlopende teksten kan importeren en bewerken.
- Het automatisch genereren van wijzigingswetten op basis van twee doorlopende teksten.
- Doorlopende (semi-)automatische controle op samenloop tussen wetten die in procedure zijn.
- Automatische consolidatie van wijzigingswetten in elke stand van de procedure (dus niet alleen bij de inwerkingtreding, maar op elk moment dat iemand wil

[3] S.W. Mul, *ICT voor wetgeving*, Kenniscentrum Wetgeving, Ministerie van Justitie, 2007.

weten hoe de wet zou luiden als de wijzigingswet bij deze stand van zaken werd aangenomen).
- Werkstroombesturing
- Een webviewer waarmee de teksten ook zonder de editor geraadpleegd kunnen worden.

Met uitzondering van de werkstroombesturing is elk van deze componenten in het prototype opgenomen (zie secties 2.3 en 5.4 voor de aanpak rondom de werkstroombesturing). Het gaat echter om basisfunctionaliteit. Bij het genereren van wijzigingswetten is nog geen ondersteuning voor het wijzigen van wijzingswetten. De controle op samenloop is gericht op situaties waarin de wijzigingen elkaar overschrijven. Er wordt niet gekeken naar de betekenis van de teksten; situaties waarbij er na de wijziging een onlogische tekst staat worden niet gedetecteerd.

Naast deze geplande componenten zijn er tijdens het project nog twee extra functionaliteiten voor de editor, namelijk het automatisch hernummeren (in basis geïmplementeerd) en het weergeven van referenties naar het huidige document (geïmplementeerd).

2.2. Begeleidingscommissie en werkgroep

Het onderzoek is uitgevoerd onder begeleiding van een commissie, waarvan de samenstelling is te vinden in Bijlage A.

Om zo goed mogelijk bij de praktijkwensen- en noodzakelijkheden aan te sluiten, is een werkgroep bestaande uit potentiële gebruikers samengesteld. Deze werkgroep is enkele malen gevraagd om suggesties voor de omgeving aan te leveren, en om de omgeving te beoordelen in een praktijksituatie.

De werkgroep is drie keer samengekomen. Bij de eerste bijeenkomst zijn de plannen voor dit onderzoek gepresenteerd, en zijn de onderzoekers begonnen met de diverse problemen en mogelijkheden in kaart te brengen. Het initiatief lag bij deze sessie bij de onderzoekers, die vragen stelden aan de werkgroep.

De twee latere sessie waren gewijd aan het testen van de proefomgeving. De leden van werkgroep werd gevraagd om met de omgeving aan het werk te gaan, hetzij met eigen meegebracht werk, hetzij met de beschikbare *tutorial*. Alle problemen, suggesties en commentaar die hierbij boven kwamen zijn door de onderzoekers genoteerd en meegenomen bij de doorontwikkeling van de proefomgeving.

2.3. Verloop van het onderzoek

Zoals aangegeven in paragraaf 1.2 bestaat er een groot corpus aan literatuur over verschillende projecten die beogen het leven van de wetgevingsjuristen en de andere bij de totstandkoming van regelgeving betrokkenen te vergemakkelijken. Omdat deze voor een groot deel afspeelden in een andere wetgevingsculturele context is er in dit onderzoek voor gekozen om naast een grondige oriëntatie op de actuele ontwikkelingen op dit gebied, ook een proefomgeving te realiseren. Hiermee kon de ontwikkelde aanpak worden beproefd in de concrete Nederlandse wetgevingspraktijk.

Na een inventarisatie van de in het wetgevingcreatieproces betrokken actoren is de hiervoor vermelde werkgroep samengesteld De leden van deze werkgroep gaven input ten behoeve van de uitgevoerde inventarisatie van eisen en wensen. Daarbij zijn zowel de ervaringen met de huidige ondersteuning voor het proces meegenomen, alsook de wensen die zijn geuit op basis van ervaringen met verschillende versies van de prototypeomgeving.

Om het belang van de verschillende geuite verbetervoorstellen te kunnen wegen is een prototype van een redactieomgeving gerealiseerd. Als basis voor deze omgeving is de MetaVex editor gebruikt, ontwikkeld door het Leibniz Center for Law van de Universiteit van Amsterdam. Deze keuze is ingegeven door een aantal praktische en theoretische overwegingen. Zo kan worden geput uit internatonale ervaringen die samenkomen in de onder het Europese normalisatieinstituut (CEN) vallende werkgroep voor juridische bronnen (CEN/MetaLex) en kan optimaal gebruik worden gemaakt van de aanwezige kennis van de medewerkers van het Leibniz Center for Law. Bovendien is de MetaLex standaard een *open* standaard en de MetaVex editor een *open source* product.
Via een kortcyclisch ontwikkeltraject zijn de wensen en eisen van de werkgroep gerealiseerd in een volgende versie van de prototypeomgeving, waarna een uitgebreide evaluatie van deze omgeving plaatsvond. Daarbij is steeds meegenomen wat de impact van een dergelijke vorm van ondersteuning op het toekomstige proces zou zijn en welke implementatieconsequenties de oplossing met zich mee zou brengen.

Uit de literatuur was al bekend dat het overgrote deel van werk van wetgevingsjuristen bestaat uit het maken van wijzigingswetgeving. Dit werd in gesprekken met de werkgroepleden bevestigd. In dit project is daarom veel aandacht besteed aan het maken van wijzigingswetgeving en in het bijzonder aan het automatisch genereren van wijzigingswetgeving.
Bij het wijzigen van een wet is het van groot belang dat de door de wetgevingsjurist gemaakte wijzigingen nauwkeurig kunnen worden opgespoord. Voor het automatisch genereren van wijzigingswetten zijn twee mogelijke aanpakken overwogen. De eerste aanpak houdt in dat alle bewerking die door de wetgevingsjurist worden uitgevoerd worden geregistreerd door de toepassing. De tweede aanpak is het achteraf vergelijken van het gewijzigde document met het originele document.
Het voordeel van de eerste aanpak is dat uit de handelingen van de gebruiker goed te achterhalen valt wat er precies veranderd is, en hoe. Een nadeel is dat deze methode het beste werkt als de gebruiker direct, in een keer, zijn wijzigingen aanbrengt, zonder op zijn schreden terug te keren. Als hij meer normaal gebruik maakt van de redactieomgeving als tekstverwerker, dan moeten de handelingen veel meer gefilterd worden om op de juiste uitkomst te komen. Bovendien werkt de methode niet zondermeer als er door meerdere mensen parallel aan dezelfde tekst gewerkt wordt. Bij de tweede methode is deze afhankelijkheid er niet. Voor dit project is daarom gekozen voor de tweede methode.
De uiteindelijke uitwerking van deze aanpak is beschreven in bijlage C.

Naast het genereren van wijzigingswetten is ook aandacht besteed aan het consolideren van wijzigingen. In het bijzonder ging het daarbij om automatische consolidatie. Het automatisch genereren van geconsolideerde teksten uit een oorspronkelijke tekst plus wijzigingen vraagt om speciale voorzieningen. In de praktijk treffen we niet altijd correcte consolidaties aan; het proces is tamelijk arbeidsintensief en foutgevoelig. Automatische consolidatie vereist dat de omgeving op een of andere manier ‘weet’ welke wijzigingen moeten worden toegepast. Bij het automatisch genereren van

wijzigingsteksten wordt automatisch een dergelijke lijst van wijzigingen gemaakt. Binnen het prototype is er voor gekozen om deze bestanden te hergebruiken (beschreven in bijlage D). Dit heeft als voordeel dat optimaal gebruik kon worden gemaakt van in de MetaVex editor reeds aanwezige functionaliteit en er geen nieuwe (deel-) functionaliteit hoefde te worden ontwikkeld. Een beperking is dat het alleen geschikt is voor consolidaties gebaseerd op wijzigingswetten die ook met die omgeving zijn geschreven; het kan niet worden ingezet voor het werken met oudere teksten of teksten die met een ander programma zijn gecreëerd, althans niet zonder additionele bewerkingen (die buiten het kader van dit onderzoek vielen). Daarnaast vereist deze methode dat er bij elke wijziging van de wijzigingstekst ook de opgeslagen betekenis van de wijzigingswet wordt bijgewerkt.

In het eerste overleg met de werkgroep (d.d. 8 april 2008) is door de werkgroep de interesse in bovenstaande functies uitgesproken. Verder is nog een aantal wensen voor functionaliteiten naar voren gekomen die niet volledig konden worden gerealiseerd binnen de scope van het huidige project. Wel is getracht een eerste voorlopige versie daarvan in de prototypeomgeving op te nemen. Het gaat hierbij om:
– Het gebruik van sjablonen (al opgenomen in de basis van het prototype);
– Automatische hernummering (beschreven in bijlage G);
– Het weergeven van inkomende referenties (beschreven in bijlage H).

Aanvankelijk was het de bedoeling in dit project ook de werkstroombesturingselementen van het wetgevingsproces mee te nemen. In overleg met de werkgroep en de stuurgroep is besloten om de nadruk te leggen op ondersteuning van de wetgevingsproductie. Voor de werkstroombesturing (het logistieke proces) bestaan al redelijk functionerende systemen. Bovendien zijn deze niet specifiek voor het wetgevend proces, al moeten ze wel worden geparametriseerd voor de specifieke gebruikscontext. De nu gerealiseerde editingomgeving is voorzien van een rudimentaire werkstroombesturingscomponent en deze is bovendien zo opgezet dat inbedding in een ander, meer geavanceerd *workflow management* system, gemakkelijk kan worden gerealiseerd.

3. Redactieomgeving voor het wetgevingsproces

De centrale vraag binnen dit project was hoe de transparantie en productie van (wijzigings)wetgeving kan worden verbeterd met behulp van ICT-ondersteuning voor de bij het wetgevingsproces betrokken actoren. Op basis van de reeds bestaande MetaVex editor is een prototype versie van een redactieomgeving gemaakt die geschikt is voor het maken van nieuwe en wijzigingswetgeving en waarmee tevens het genereren van wijzigingsteksten, het opslaan van wijzigingsteksten en leveren van de bijbehorende consolidaties kan worden ondersteund.

In deze sectie wordt de werking van die redactieomgeving beschreven. Ten behoeve van de leesbaarheid is getracht het gebruik van technische termen zoveel mogelijk te voorkomen.

3.1. Werking

Een belangrijke component in een redactieomgeving is de tekstverwerker (editor). Deze moet de gebruikelijke functies voor het bewerken van tekst bevatten. Om hulp te kunnen

bieden bij het schrijven van nieuwe of wijzigen van bestaande regelgeving moet de (toekomstige) editor daarnaast een aantal speciale functies bevatten.

Omdat bij het maken c.q. wijzigen van regelgeving vaak volgens een vast stramien gewerkt wordt, kunnen sjablonen voorgestructureerde stukken tekst die in wetgeving veelvuldig voorkomen, worden gebruikt. Deze sjablonen kunnen eenvoudig als geheel worden ingevoegd, zodat ze niet letter voor letter hoeven worden ingetypt. Dit type functionaliteit is niet specifiek voor een wetgevingseditor en ook Microsoft Word bijvoorbeeld kent dergelijke functionaliteit om sjablonen te maken in de vorm van zogenaamde macro's. Deze zijn in dat geval niet wetgevingspecifiek, terwijl voor wetgeving specifieke voorgestructureerde tekstblokken nodig zijn. Daarnaast wordt er ook regelmatig uit bestaande regelgeving tekst gekopieerd, hetgeen eigenlijk een primitievere versie is van deze sjablonen.

Bij het schrijven van een wetsvoorstel kan veel tijd verloren gaan aan het bijwerken van de nummering van een document, en het vervolgens bijwerken van de referenties binnen het document. In de voorgestelde omgeving is het mogelijk om dit hernummeren door de applicatie te laten uitvoeren. Hierbij worden dan ook de verwijzingen in de regeling zelf (en aanverwante documenten, zoals de toelichting) bijgewerkt.

Bijzondere aandacht is er voor het wijzigingen van een regeling. Een wijzigingswet bestaat doorgaans uit een lijst instructies die aangeven hoe een bestaande wet gewijzigd moet worden. De wetgevingsjurist moet daarom bedenken op welke tekst hij uit wil komen en vervolgens de instructies bedenken die uitleggen hoe van de originele tekst naar de bedoelde wijziging te komen. Dit is een zeer complexe taak die aanzienlijk vereenvoudigd kan worden in een moderne redactieomgeving. In een dergelijke omgeving brengt de jurist de gewenste wijzigingen direct aan in de brontekst, waarna de wijzigingsinstructies automatisch worden gegenereerd op basis van deze wijzigingen.

De redactieomgeving toont de wetgevingsjurist bij elk onderdeel van de te wijzigingen regeling welke andere regelingen er naar verwijzen (en dus potentieel beïnvloed worden door een wijziging). Deze informatie helpt de wetgevingsjurist om de impact van elke wijziging te overzien.

Een belangrijk probleem bij de wijzigingen ontstaat door de mogelijke samenloop van wijzigingen. De redactieomgeving zal ook hierbij assistentie verlenen door de uitkomst van verschillende voorgestelde wijzigingen te vergelijken en aan te geven waar zich mogelijk samenloop zal voordoen. De wetgevingsjurist zal dan eenvoudiger kunnen bepalen of er inderdaad samenloop bestaat en zo ja, een samenloopbepaling kunnen opnemen ter oplossing van het probleem.

De redactieomgeving is in staat om consolidaties te genereren op basis één of meer wijzigingen. Dit maakt het eenvoudiger om een consolidatie te publiceren, ook tussentijds. Dit is vooral interessant voor anderen dan de wetgevingsjuristen die de regelingen tussentijds willen raadplegen. Het geeft geïnteresseerden een beeld van hoe een regeling er uit komt te zien indien bepaalde wijzigingen worden aangenomen (te denken valt aan wetgevingstoetsers, Raad van State, kamerleden etc.).

3.2. Inzetbaarheid

Naast de vraag naar de impact van een redactieomgeving op het werk van wetgevingsjuristen is ook breder gekeken naar de inzetbaarheid van een dergelijke ICT ondersteuning voor regelgevers. Hoewel we in dit rapport vooral het gebruik van de redactieomgeving voor wetgeving belichten, is de editor inzetbaar voor alle niveaus van regelgeving: niet alleen wetten, maar ook bijvoorbeeld algemene maatregelen van bestuur of gemeentelijke regelingen. Hiervoor is wel vereist dat voor elk niveau waarop de editor gebruikt wordt specifieke sjablonen en een eigen schema wordt gecreëerd en toegevoegd aan de editor.
De editor biedt vooral voordelen voor documenten die een duidelijke structuur hebben, regelmatig standaardteksten bevatten en veel formele verwijzingen gebruiken. Voor documenten die niet deze eigenschappen hebben, zal het gebruik van de editor niet veel toevoegen. Echter, het grootste deel van de regelgeving heeft wel deze eigenschappen. Dat geldt overigens ook voor rechtspraak. Ook deze zou met de ontwikkelde editor kunnen worden gemaakt, waarbij de voordelen van XML en metadata kunnen worden benut om samenhang tussen deze uitspraken en regelgevende bronnen zoals regelgeving en andere uitspraken te verbeteren. Uiteraard zouden dan daartoe geëigende *templates* moeten worden gebruikt.

Om optimaal gebruik te maken van de mogelijkheden van de redactieomgeving is het nodig dat alle partijen die wetteksten bewerken gebruik maken van de redactieomgeving. Desalniettemin is het ook mogelijk om de redactieomgeving bij slechts een deel van deze partijen in te voeren. De wetgevingsjuristen die de editor gebruiken, zullen profijt kunnen hebben van de tekstverwerker: de sjablonen, het automatisch genereren van wijzigingsteksten en het genereren van consolidaties werken allemaal in dit scenario. De controle op samenloop wordt wel beperkter: de controle kan alleen plaatsvinden ten aanzien van wetsvoorstellen die van die ministeries komen die gebruik maken van de omgeving.
Communicatie van en naar instanties die geen gebruik maken van de omgeving is mogelijk door middel van import- en exportfuncties. Ervan uitgaande dat er in zulke gevallen wordt geëxporteerd naar MS Word heeft het geen zin om bij veelvuldig converteren de metadata in te voeren, aangezien deze bij de conversies verloren gaat. Dit heeft wel als gevolg dat de zoekmogelijkheden door gebruik te maken van XML met metadata niet kunnen worden benut hetgeen tot aanzienlijk slechtere zoekresultaten leidt.

Zoals hierboven vermeld, hangen de genoemde effecten af van het gebruik van de redactieomgeving door de partijen die wetteksten bewerken. De partijen die de teksten alleen raadplegen zullen gebruik kunnen maken van de viewer. Dit biedt voor hen het voordeel dat zij de extra informatie die aanwezig is in de XML bestanden kunnen raadplegen en gebruiken bij het zoeken, het markeren van gemaakte wijzigingen, etc. Indien een van deze partijen niet overgaat naar de nieuwe omgeving, dan heeft dat alleen gevolgen voor die partij, die dan geen gebruik kan maken van de toegevoegde informatie. Zij kunnen de teksten dan verkrijgen in bijvoorbeeld MS Word of PDF formaat, en verder blijft voor hen alles hetzelfde.

4. Gevolgen voor de praktijk

Uit de verschillende evaluaties en proefnemingen die onderdeel uitmaakten van dit project is een aantal lessen te trekken en voorspellingen te doen over de gevolgen van de implementatie van een specifieke redactieomgeving voor het wetgevend proces.

4.1. Wetgevingsjurist

De wetgevingsjurist krijgt een extra verantwoordelijkheid: hij moet niet alleen de tekst produceren, maar er ook voor zorgen dat de structuur op de juiste wijze is gemarkeerd. Dit beperkt zijn vrijheid van handelen enigszins. Bij de huidige omgeving (voornamelijk gebaseerd op MS Word) kan hij naar believen tekst invoeren, en eventueel (achteraf) enige opmaak aanbrengen. Bij de wetgevingseditor kan er niet zonder meer door worden getypt; bij elk nieuw structuur onderdeel moet eerst worden aangegeven dat er met een nieuw onderdeel wordt begonnen. Met een intelligente user-interface kunnen deze handelingen worden versimpeld, maar ze kunnen nooit helemaal worden geëlimineerd. In dit opzicht zal de nieuwe redactieomgeving restrictiever zijn dan de bestaande omgevingen.

Naast het markeren van de structuur zullen er nog een aantal kleinere wijzigingen zijn voor de wetgevingsjurist. De nieuwe omgeving heeft bepaalde functies die de bestaande omgeving niet heeft, en vice versa. Verder verschilt de user interface op bepaalde punten; hoewel er geprobeerd is zoveel mogelijk zaken zo te houden als men gewend is, was het op een aantal punten nodig om een andere user interface te kiezen. Deze veranderingen in de user interface zullen geen grote impact hebben op het werk van de wetgevingsjurist, maar het spreekt vanzelf dat er een inwerkperiode vereist gaat zijn.

Het schrijven van een nieuwe tekst zal in de nieuwe omgeving minder bestaan uit het intypen van tekst. Meer dan bij bijvoorbeeld MS-Word zal de wetgevingsjurist sjablonen invoeren bestaande uit alleen structuur of structuur en tekst. Deze sjablonen zijn bedoeld als hulp voor de wetgevingsjurist: ze vergemakkelijken het invoeren van structuurelementen en standaardteksten. Als voorbeeld een scenario waarbij een wetgevingsjurist een artikel met daarin een evaluatiebepaling toevoegt aan een wetsvoorstel.

In MS Word gaat dit als volgt in zijn werk:
1. De gebruiker plaatst de cursor op de plek waar hij het nieuwe artikel wil invoeren.
2. Hij typt “Artikel “, gevolgd door het nummer, gevolgd door <enter>.
3. Hij typt de tekst van het artikel, of kopieert deze uit een voorbeeldtekst, en past deze aan.
4. Hij selecteert de kop van het artikel, en past het opmaakprofiel aan, zodat de lay-out klopt.

Met de wetgevingseditor verandert dit in:
1. De gebruiker plaatst de cursor op de plek waar hij het nieuwe artikel wil invoeren.
2. Hij selecteert uit een menu het sjabloon voor een nieuw artikel zonder onderliggende leden, en voert deze in. Op het scherm verschijnt de volgende tekst:

   Artikel <nummer>
   <tekst>

De teksten tussen vishaken zijn invoervelden; de tekst verdwijnt zodra de echte tekst is ingevoerd.

3. Hij verplaatst de cursor naar het invoerveld <nummer> en voert het nummer van het artikel in.
4. Hij verplaatst de cursor naar het invoerveld <tekst> en selecteert uit een menu het sjabloon voor een evaluatiebepaling, en voert deze in. Op de plaats van <tekst> verschijnt een nieuwe tekst:

   Onze Minister van <ministerie> *zendt in overeenstemming met Onze Minister van* <ministerie> *binnen* <aantal> *jaar na de inwerkingtreding van deze wet (, en vervolgens telkens na* <aantal> *jaar,) aan de Staten-Generaal een verslag over de doeltreffendheid en de effecten van deze wet in de praktijk.*

5. Hij vult de tekstvelden in door de cursor er naar toe te verplaatsen en de tekst in te voeren. Eventueel past hij de andere tekst uit het sjabloon aan.

Het is natuurlijk mogelijk om de tekst in te voeren zonder gebruik te maken van een tekstsjabloon (zoals de evaluatiebepaling uit het voorbeeld). In veel gevallen zal dit zelfs moeten; alleen voor teksten die relatief standaard zijn zullen sjablonen beschikbaar zijn.

Naast het markeren van de structuur wordt van de wetgevingsjurist ook verwacht dat zij de (nieuwe) verwijzingen markeren. Hierom vereist het invoeren van verwijzingen een speciale handeling, waarmee aan de omgeving duidelijk wordt gemaakt waarnaar verwezen wordt. Dit is een extra handeling voor de wetgevingsjurist, maar hierdoor kan de impact van wijzigingen van regelingen op elkaar in kaart worden gebracht en aan inzichtelijk worden gemaakt. Het voorkomt derhalve onbedoelde neveneffecten en bevordert de coherentie bij samenhang tussen regelingen. Uiteraard heeft de omgeving een brede en uit te breiden bibliotheek van bruikbare sjablonen, met uitleg en/of begeleiding om de juiste sjablonen te selecteren. De wetgevingsjurist zal zodoende voor standaardteksten minder vaak externe bronnen hoeven raadplegen, en niet hoeven te kopiëren uit andere teksten. Ook zal de wetgevingsjurist zijn eigen sjablonen kunnen maken.

De specifieke functies die aan de omgeving zijn toegevoegd zijn gericht op tijdsbesparing en kwaliteitsverbetering. Hierbij gaat het om het automatisch genereren van wijzigingen, samenloop opsporing, automatisch hernummeren, en het overzicht van inkomende referenties. Ook wordt tijdbesparing gerealiseerd doordat er minder tijd besteed hoeft te worden aan het opsporen van problemen.

4.2. Organisatie

Voor de organisatie is het belangrijkste voordeel dat met inzet van een redactieomgeving voor regelgeving te behalen is, dat de regelgeving onmiddellijk voorzien is van structuurinformatie. Dit betekent dat er makkelijk snel tussentijds kan worden gepubliceerd (d.w.z. voorafgaand aan de officiële publicatie). Het nut van dergelijke snelle publicaties ligt vooral in de communicatie met belangengroepen die geraadpleegd worden bij de totstandkoming van een wet. Door middel van de structuurinformatie is het toepassen van de juiste opmaak eenvoudig. Hetzelfde geldt voor het genereren van *webcontent*.

Een ander belangrijk voordeel ligt in de kwaliteit van de regelgeving. De omgeving dwingt niet af dat de wetgevingsjurist de organisatiestandaarden (zoals de Aanwijzingen voor de regelgeving) toepast, maar maakt het toepassen ervan wel makkelijker. De verwachting is dat dit conformiteit met de standaarden zal verhogen. Daarnaast zijn er een aantal functies die fouten in de tekst voorkomen; hierbij valt vooral te denken aan de automatische hernummering, het aanbrengen van correcte verwijzingen en de detectie van samenloop. De omgeving draagt zo in belangrijke mate bij aan het verbeteren van de kwaliteit van de regelgeving.

4.3. Samenleving

Wetteksten hebben diverse eindgebruikers: juristen, kamerleden, ondernemers en burgers. Voor veel van deze groepen zijn wetteksten slecht leesbaar. Wijzigingswetten zijn nog onhandiger in het gebruik: ze vereisen dat de lezer de wijzigingen in gedachten invoert in de te wijzigen tekst om te begrijpen wat het effect is. Een van de meest zichtbare gevolgen van het gebruik van de redactieomgeving is de mogelijkheid om veel consolidaties-on-the-fly te genereren. In het huidige proces wordt pas laat in het proces een geconsolideerde tekst aangeboden. Met de redactieomgeving is het mogelijk om veel vaker een consolidatie beschikbaar te stellen, zodat de eindgebruiker eenvoudiger de mogelijke effecten van (toekomstige) wijzigingen kan zien.

Verder verwachte voordelen voor de eindgebruikers zijn een verdere verbetering van de zoekmogelijkheden voor digitale wetteksten, en mogelijk een verbetering van de kwaliteit van wetteksten.

Ten slotte zou de nu ontwikkelde omgeving kunnen worden uitgebreid met meer geavanceerde componenten, zoals een component waarmee met behulp van natuurlijke taal technologie automatisch concepten kunnen worden gehaald uit de regelgeving. Vervolgens kan de wetgevingsjurist worden gewezen op eerder gebruik van die concepten zodat voorkomen wordt dat onbedoeld nieuwe begrippen worden geïntroduceerd. Bij een bewuste verruiming c.q. beperking van die begrippen kan de relatie met die begrippen worden geduid. Op deze wijze kan worden bereikt dat de coherentie tussen de regelgeving wordt versterkt. Ook kan op deze wijze aangesloten worden op modellen waarmee de effecten van regelgeving kunnen worden doorgerekend. In het recente Legis project zullen dit soort toepassingen nader worden onderzocht.

5. Conclusies en aanbevelingen

5.1. Gewenste functionaliteit is haalbaar

Uit de bevindingen van dit project blijkt dat de gevraagde functionaliteit technisch haalbaar is. De gevraagde functionaliteit is geïmplementeerd in het prototype, met uitzondering van wijziging van wijzigingen, dat alleen in theorie is uitgewerkt.
Voor daadwerkelijke in gebruikneming van software met deze functionaliteit zal nog wel aan diverse randvoorwaarden moeten worden voldaan. Zo moet er draagvlak onder de wetgevingsjuristen worden gecreëerd, is opleiding van de wetgevingsjuristen nodig en moet het gebruik van de software in de werkprocessen worden ingepast.

Uit verschillende proefnemingen met het prototype voor een redactieomgeving komt naar voren dat met deze omgeving een efficiëntieverbetering te realiseren is. Er kan sneller worden gewerkt, met minder kans op fouten. Ook het uitwisselen, consolideren, detecteren en voorkomen van samenloop, hernummeren en correct verwijzen is met de redactieomgeving aanzienlijk eenvoudiger geworden.

Een cruciaal deel is de gebruikersinterface voor de verschillende functies. De gebruikersinterface van het huidige prototype is adequaat, maar nog zeker niet intuïtief, en wordt nog niet als ideaal ervaren. Voor de meer geavanceerde functionaliteit zal ook een zeer doordachte user interface nodig zijn. Een wijzigingswet op een wijzigingswet vereist bijvoorbeeld dat de gebruiker werkt met meerdere, samenhangende documenten, waarbij hij zijn wijzigingen op verschillende plekken moet invoeren om tot een kloppend eindresultaat te komen. Dit vereist een interface die de verschillende documenten goed kan tonen, maar ook hun samenhang en de toegestane operaties. Ook moet de interface duidelijk aan de gebruiker overbrengen welke operaties hij nodig heeft om bij zijn einddoel te komen.
Meer nog dan het nut van de functies zal de gebruikersinterface het succes van deze omgeving bij de wetgevingsjuristen bepalen. Ondanks de huidige beperkingen van de redactieomgeving, het gaat immers nog om een prototype, zijn de gebruikers die deel uitmaakten van de werkgroep enthousiast. We realiseren ons dat het hier gaat om de zogenaamde *local champions*, hetgeen betekent dat ze positiever staan tegenover dit project dan de gemiddelde gebruiker. Bij een verdere uitrol van de redactieomgeving verdient het de aanbeveling een inwerktraject te plannen zodat de omschakeling voor toekomstige gebruikers zonder al te veel hindernissen kan geschieden.

De ontwikkelde proefomgeving bevat op dit moment vooral basisfunctionaliteit; voortzetting van de ontwikkeling is gewenst om tot een completer resultaat te komen. Hierbij zijn er een aantal aandachtsgebieden te noemen.

Een eerste gebied dat extra aandacht behoeft is de user interface van de editor. Deze schiet nog tekort; hoewel hij voldoende bruikbaar is om de functionaliteit te demonstreren, is hij nog niet goed genoeg voor dagelijks gebruik. Bij vervolgonderzoek is het belangrijk om hier expliciet de aandacht op te richten.

Het genereren van wijzigingsteksten (wijzigingswetten, nota's van wijziging, etc.) is tot dusver alleen geïmplementeerd voor het wijzigen van een originele wet (of andere regeling). Voor het wijzigen van een wijzigingstekst is een uitgebreidere procedure nodig. Deze procedure is al wel uitgewerkt in bijlage C van dit rapport, maar is nog niet geïmplementeerd. Verder onderzoek naar deze functie, en implementatie ervan, is gewenst. Hierbij moet vooral ook worden gelet op de user interface, die voor deze functie erg complex kan worden, omdat de gebruiker in meerdere documenten tegelijk moet werken.

Tot slot behoeft ook de samenloopdetectie extra aandacht; zie daarvoor sectie 5.5.

5.2. Noodzaak standaard gestructureerde opslag

In de voorgaande hoofdstukken is naar voren gekomen dat het voor de gewenste functionaliteit noodzakelijk is om regelgeving op te slaan op een wijze waarin de structuur duidelijk gemarkeerd is. Automatisch genereren van wijzigingen is niet

mogelijk zonder deze informatie, omdat er zonder deze informatie geen houvast is voor het beschrijven van een locatie binnen de regelgeving. Ook andere functies, zoals hernummering en het markeren van inkomende verwijzingen, zijn alleen goed mogelijk als bepaalde informatie in de tekst is gemarkeerd. Aangezien binnen wetten tot op zinsniveau wordt verwezen, is het belangrijk om de elementen tot op zinsniveau te markeren, en te voorzien van identificatie.

Deze informatie wordt bij voorkeur gemarkeerd binnen de bestanden zelf, om te voorkomen dat de documenten en de metadata (structuurinformatie, relevante data, auteur, etc.) continu op inefficiënte wijze moeten worden gekoppeld[4]. Voor het markeren van de documenten bestaat een internationale standaard *eXtenisble Mark-up Language* (XML), die ook voor deze taak zeer geschikt is.[5]

XML geeft een generieke structuur, waarbinnen vervolgens voor specifieke soorten documenten een apart structuurbeschrijving kan worden gemaakt. Voor regelgeving bestaat hiervoor een CEN uitwisselingsstandaard, genaamd METAlex/CEN. Dit is een open standaard, die gebaseerd is op jarenlange ervaring met applicaties voor regelgeving. Gebruik van deze standaard maakt het mogelijk om aan te sluiten op andere applicaties, en voorkomt afhankelijkheid van een specifieke leverancier. Er zijn al automatische vertalers van diverse nationale en andere XML formaten naar METAlex/CEN beschikbaar en deze zijn verder relatief eenvoudig te ontwikkelen en aan te passen. Voordeel is dat dezelfde software gebruikt kan worden voor al deze verschillende formaten nadat ze naar METAlex/CEN zijn omgezet.
Overigens wordt voor de beschikbaarstelling via wetten.nl nu gebruik gemaakt van het XML formaat van SDU (SDU BWB). Het is de verwachting dat de METAlex/CEN standaard geautomatiseerd kan worden omgezet naar dit formaat.[6]

Naast het markeren van de structuur binnen een document zal ook de structuur tussen documenten moeten worden vastgelegd. Regelingen zijn versies van hetzelfde werk; wijzigingen vormen een brug tussen regelingen. Deze informatie is van belang bij het doorzoeken van het corpus, en bij het *on-the-fly* genereren van consolidaties.

Voor de opslag van deze informatie zijn diverse formaten beschikbaar. Het *Resource Description Framework* (RDF) is een interessante kandidaat van het W3C omdat het goed aansluit bij XML.[7]

5.3. Mogelijkheid standaardisering

Op dit moment gebruiken de verschillende actoren in de regelgevingsketen verschillende ICT toepassingen. Toch werken ze met (delen van) dezelfde documenten. Dit betekent dat op diverse momenten tijdens het proces de documenten van het ene formaat in het andere formaat moet worden omgezet.

---

[4] Merk op dat dit alleen geldt voor die metadata die gekoppeld is aan het document. Metadata die bij een groep van documenten hoort, kan beter wel apart worden opgeslagen. Dit voorkomt dubbel opgeslagen informatie en de daarbij behorende onderhoudsproblemen. Mocht het voor presentatie aan de gebruiker nuttig zijn de informatie wel bij de hand te hebben, dan kunnen daarvoor eventueel aparte documenten worden afgeleidt van de originelen.
[5] http://www.w3c.org/XML/
[6] Zie sectie 5.6.
[7] http://www.w3c.org/RDF/

In sommige situaties betreft deze omzetting vooral de lay-out: een wetsvoorstel heeft een verschillende vormgeving bij het ontwerpende ministerie, de 2e kamer, de 1e kamer en de uiteindelijke publicatie in het Staatsblad. Door de inhoud te scheiden van de opmaak, en duidelijk te structureren, kan de tekst van zo'n voorstel eenvoudig in de gewenste opmaak worden getoond (en opgeslagen en gepubliceerd). Een dergelijk systeem is te realiseren met op elkaar afgestemde opmaaksjablonen in Word of OpenOffice. Echter, de meer geëigende standaarden voor deze procedure zijn XML voor de inhoud gecombineerd met XSLT of CSS voor de opmaak. Dit sluit ook beter aan bij de andere aanbevelingen in dit rapport, omdat opmaaksjablonen alleen informatie op kunnen slaan die relevant zijn voor de opmaak. XML is veel breder in dit opzicht, en kan worden gebruikt om de hele structuur te markeren en metadata toe te voegen.

De eerder genoemde omzettingen zijn werkintensief omdat tijdens het proces de inhoud slechts deels of niet wordt gemarkeerd. Bij elke omzetting is het daarom nodig dat de tekst wordt geherinterpreteerd. Dit betekent in feite dat kennis die in een eerder stadium al bekend was, wordt weggegooid en weer opnieuw wordt achterhaald. Door de documenten al tijdens het proces gestructureerd op te slaan, wordt dit voorkomen.

Invoering van de omgeving in de keten zal deze gewenste standaardisatie automatisch realiseren, omdat daarmee de documenten in één formaat, i.c. METAlex/CEN, gestructureerd worden opgeslagen.
Op dit moment zijn echter niet alle documentsoorten die in het wetgevend proces een rol spelen meegenomen. Van deze documenten zullen de structuureigenschappen eveneens in METAlex/CEN moeten worden opgenomen.

Indien het niet mogelijk blijkt om de editor bij alle ministeries en de Tweede Kamer tegelijk in te voeren, dan is het wenselijk dat in elk geval de wetgevingsjuristen bij de Tweede Kamer de editor in gebruik nemen. Zij bevinden zich aan het einde van de keten[8]. Dit betekent dat als zij geen deel uitmaken van de standaardisatie, het effect teniet wordt gedaan: de uitkomst van het proces is dan weer een niet-gestandariseerd document, dat voor publicatie moet worden omgezet. Dit bezwaar geldt niet voor andere documenten waar de Tweede Kamer in de keten ontbreekt zoals Ministeriële Regelingen. Een mogelijkheid zou zijn de hier geschetste technieken eerst in te voeren op ministerieel niveau om de complexiteit in de keten te beperken.

Mocht de editor worden ingevoerd bij de Tweede Kamer en slechts een deel van of geen van de ministeries, dan is het gewenst om ondersteunende functionaliteit te ontwikkelen voor het omzetten van een niet- of semi-gestructureerd document naar een gestructureerd document. Deze omzetting is zwaarder dan de omzettingen die op dit moment plaatsvinden, en zonder dergelijke ondersteuning zou de werkdruk bij de Tweede Kamer juist toenemen in plaats van afnemen als gevolg van de (deels uitgevoerde) standaardisatie.

5.4. Werkstroombesturing

Hoewel het aanvankelijk de bedoeling was meer aandacht te besteden aan de werkstroombesturing van het wetgevingscreatieproces, is dit op instigatie van de werkgroep en stuurgroep grotendeels buiten beschouwing gebleven. De processtappen waarlangs wetgeving tot stand komt verschilt op dit moment van ministerie tot

[8] Voor zover het inhoudelijke wijzigingen betreft.

ministerie. Stroomlijning daarvan maakt uitwisseling tussen deze ministeries eenvoudiger en ook het bewaken van de proceskwaliteit kan dan worden vereenvoudigd. Een toekomstige redactieomgeving kan overigens gemakkelijk in een procesbesturingsomgeving worden ingebouwd. Op dit moment bestaan er echter tamelijk wat van dergelijke omgevingen. Ieder ministerie hanteert zijn eigen werkstroombesturingssysteem. Het verdient aanbeveling een overkoepelend systeem te ontwikkelen dat gebruik maakt van de informatie uit die afzonderlijke systemen, en waarmee de integrale proceskwaliteit kan worden bewaakt.

De redactieomgeving kan los naast de bestaande systemen worden gebruikt, op dezelfde manier waarop bijvoorbeeld nu MS Word zal worden ingezet. Afhankelijk van het contentmanagement systeem dat wordt gebruikt, is het ook mogelijk om de omgeving direct the integreren met het bestaande systeem, door middel van een plugin. Een plugin is een stuk deelsoftware dat naar believen kan worden toegevoegd of weggelaten uit de omgeving. Door functionaliteit als deze onder te brengen in een plugin, kan er voor worden gezorgd dat bijvoorbeeld verschillende ministeries een eigen uitbreiding krijgen om aan te sluiten op hun eigen systeem, zonder dat de functionaliteit voor andere systemen hen in de weg zit. Op dit moment bestaat zo'n plug-in voor het open source content management systeem Subversion[9]. Middels deze plug-in kan de omgeving bestanden direct ophalen uit en wegschrijven in het content management systeem, en is het dus niet nodig om lokale kopieën aan te houden.

Aansluiting op de complete werkstroombesturing kan op een soortgelijke wijze plaatsvinden. Indien de werkstroombesturingscomponent reageert op veranderingen in het content management systeem, dan is aansluiting van de omgeving op dat systeem voldoende om de integratie te voltooien. Zo niet, dan is het mogelijk om ook een plugin te ontwikkelen waarmee de omgeving als cliënt van de werkbesturing kan functioneren (wat wil zeggen dat de omgeving met de werkbesturing kan communiceren). Dit vereist wel dat de werkstroombesturingscomponent dit toestaat; zo niet, dan is integratie op dit niveau alleen mogelijk indien die component wordt aangepast.

Functioneel houdt de integratie in dat het relevante deel van de user interface voor het content management systeem en de werkstroombesturing wordt gekopieerd naar de redactieomgeving, zodat gebruikers niet met meerdere systemen tegelijk hoeven werken. Hoeveel en wat er precies zal worden overgenomen hangt af van het bestaande systeem en de werking daarvan.

5.5. Voorkomen samenloop

ICT kan samenloop niet detecteren, laat staan voorkomen. Wat met ICT wel kan, is *mogelijke* samenloop detecteren, en de wetgevingsjurist op deze mogelijkheid wijzen. Vereist hiervoor is dat de wijzigingen niet alleen in tekst beschreven zijn, maar ook in een voor een computerapplicatie leesbaar formaat. Bij automatische generatie van wijzigingswetten wordt een dergelijke beschrijving van de wijzigingen ook automatisch gegenereerd. In andere gevallen moet een dergelijke beschrijving apart worden aangemaakt. Met andere woorden: deze functionaliteit kan waarschijnlijk alleen worden aangeboden indien ook de redactie omgeving in gebruik wordt genomen.

[9] http://subversion.tigris.org/

Het voorzien van de wetgevingsjuristen van een redactieomgeving met samenloop detectie functies is echter niet voldoende om ook daadwerkelijk de samenloop op tijd op te sporen en te verhelpen. Een belangrijke reden dat er problemen in de samenloop optreden is dat de wetgevingsjuristen zich niet bewust zijn van het bestaan van mogelijk conflicterende voorstellen, omdat deze vaak lang buiten de openbaarheid worden gehouden. Succesvol tegengaan van samenloop vereist dus ook dat er in de wetgevingscyclus een tijdstip wordt aangegeven waarop een wijzigingsvoorstel bekend wordt gemaakt en wordt getest op samenloop. Vanaf dat moment zouden verdere aanpassingen van het voorstel direct moeten worden getest op samenloop, zodat de groep "bekendgemaakte voorstellen" altijd samenloop vrij is.

Indien samenloop optreedt, dan zijn er een aantal mogelijkheden om een wijzigingswet te herschrijven zodat de samenloop niet optreedt. De eenvoudigste situatie doet zich voor als men er zeker van is dat één van de twee conflicterende wetten eerder in werking treedt. In dat geval kan de tweede wet worden geschreven op basis van het resultaat van de eerste wet.

Is dit niet het geval, maar kunnen beide wetten nog gewijzigd worden, dan kunnen de conflicterende delen mogelijk worden samengevoegd. Is ook dit niet mogelijk of wenselijk, dan moeten er (ingewikkelde) samenloopbepalingen worden opgenomen. De laatste oplossing is het meest ingewikkeld en daarom minder gewenst. Door beperkingen op te leggen aan de inwerkingstredingsdata is het wellicht mogelijk om de eerste situatie vaker voor te laten komen. Door het controlemoment te vervroegen, zal het vaker mogelijk zijn om conflicterende delen samen te voegen.

Al met al is het belangrijk om de ondersteuning van het wetgevingsproces zodanig in te richten dat samenloop makkelijker op te sporen of te voorkomen is. Een minimum aanpassing is het invoegen van een controlemoment zoals boven genoemd.

De omgeving biedt op dit moment een minimale functionaliteit om samenloop op te sporen, en meldt op dit moment alleen problemen waarbij de wijzigingen botsen in de structuur van de tekst. Verdere waarschuwingen zijn ook mogelijk, maar nog niet geïmplementeerd. In een vervolgonderzoek zou dit verder moeten worden uitgewerkt, waarbij de volgende punten worden verwerkt:

- Loskoppeling van de samenloopdetectie van de consolidatiefunctie. Dit houdt in dat de gebruiker niet meer expliciet aangeeft welke teksten hij wil testen, maar dat de omgeving automatisch de relevante teksten selecteert en test op samenloop.
- Waarschuwingen voor mogelijke samenloop.

Indien een meer gedetailleerd niveau van samenloop gewenst is, kan het onderzoek ook aandacht hebben voor de extractie van semantische informatie door de omgeving, en het inzetten van deze informatie bij het opsporen van samenloop. Hiermee kan een beter resultaat worden bereikt dan met alleen structuur informatie.

5.6. Aansluiting bestaande systemen

Op dit moment wordt van alle wetten de geconsolideerde vorm opgeslagen in het Basis Wettenbestand. Dit bestand is in te zien via wetten.nl. Het wettenbestand wordt opgeslagen in een speciaal ontworpen XML formaat, de BWB XML DTD. Het prototype dat is ontwikkeld in het kader van dit project is gebaseerd op regelingen die zijn

opgeslagen in METAlex/CEN XML, de LS variant. Deze systemen sluiten niet zonder meer op elkaar aan.

Een oplossing voor aansluiting is om beide formaten te handhaven, en een conversie tussen de twee uit te voeren. Dit zou inhouden dat zodra een gebruiker een wijzigingswet wil maken, het originele document uit het basis wettenbestand wordt gehaald en wordt geconverteerd naar METAlex/LS[10]. Tijdens het ontwerpproces blijft het document in METAlex/LS. Indien er mensen geraadpleegd worden die geen gebruik maken van de editor, dan wordt er een Word, PDF of andere versie gegenereerd. Zodra de regeling voltooid is en wordt toegevoegd aan het wettenbestand, wordt het van METAlex naar BWB DTD geconverteerd.

Nadeel hiervan is dat de conversie tussen METAlex/LS en BWB DTD niet triviaal is. METAlex/LS gebruikt meer generieke elementen dan BWB DTD; de informatie die in BWB DTD in de specifieke elementen zit wordt in METAlex opgeslagen in metadata of andere structuren. Verder wordt in de BWB DTD een uitgebreide geschiedenis opgeslagen, die volgens de METAlex filosofie buiten het document wordt gehouden. Dit betekent dat de conversie niet simpelweg het vervangen van het ene element door het andere is.

Conversie kan worden vermeden door er voor te zorgen dat beide systemen gebruik maken van hetzelfde XML formaat. Dit houdt dus in dat of het wettenbestand wordt omgezet naar METAlex, of dat de editor wordt aangepast om BWB DTD te gebruiken.

Omzetting van het wettenbestand naar METAlex impliceert ook dat alle toepassingen die op dit moment gebruik maken van het wettenbestand (zoals de redactieomgeving en wetten.nl) moeten worden aangepast.

Mocht de BWB DTD in de toekomst als basis dienen voor de editor, dan is aanpassing van deze DTD gewenst. Zonder aanpassing kan niet de volledige functionaliteit zoals gerealiseerd in een op METAlex gebaseerde oplossing worden geboden. De documenten binnen het basis-wettenbestand (in BWB DTD) zijn weliswaar gestructureerd opgeslagen, zoals noodzakelijk is voor de nieuwe functionaliteit, maar deze structurering gaat niet dieper dan lidniveau. Zinnen zijn niet gemarkeerd, terwijl wijzigingen idealiter wel tot op zinsniveau worden gelokaliseerd. Idealiter wordt markering van zinnen dus aan de BWB DTD toegevoegd[11]. Bestaande applicaties zouden deze markering moet negeren, wat ze mogelijk al doen of anders slechts een kleine aanpassing vereist. Het bestaande wettenbestand hoeft ook niet volledig geconverteerd te worden; slechts die wetten die nog gewijzigd gaan worden (de huidige versies dus) moeten op dusdanige wijze gemarkeerd worden.

[10] Of een van de nog te ontwikkelen subschema's; het is de verwachting dat de geplande subschema's van MetaLex/LS dichterbij BWB DTD liggen dan MetaLex/LS zelf. Het al dan niet gebruiken van de subschema's heeft geen effect op de nauwkeurigheid van de editor. Wel is het een mogelijkheid om strakkere eisen aan de gebruikers op te leggen. Verder biedt het vooral meer mogelijkheden bij sommige conversies (zoals BWB DTD) en andere programma's die gebruik maken van het schema.

[11] Een andere mogelijkheid is om de applicatie de tekst te laten splitsen in zinnen. Voor wetteksten kan dit redelijk betrouwbaar, maar de kans op fouten is altijd groter dan bij expliciete markering in de XML.

Bijlage A – Begeleidingscommissie

Voorzitter:
prof.dr. W.J.M. Voermans
Universiteit Leiden, Faculteit der Rechtsgeleerdheid

Leden:
mr. T.C. Borman
Ministerie van Justitie, Directie Wetgeving

dr.ir. R. Choenni
Ministerie van Justitie, Wetenschappelijk Onderzoek- en Documentatiecentrum

mr.drs. J.W. Flier
Ministerie van Binnenlandse Zaken, Directie Innovatie en Informatiebeleid Openbare Sector

mr. W.M. de Jongste
Ministerie van Justitie, Wetenschappelijk Onderzoek- en Documentatiecentrum

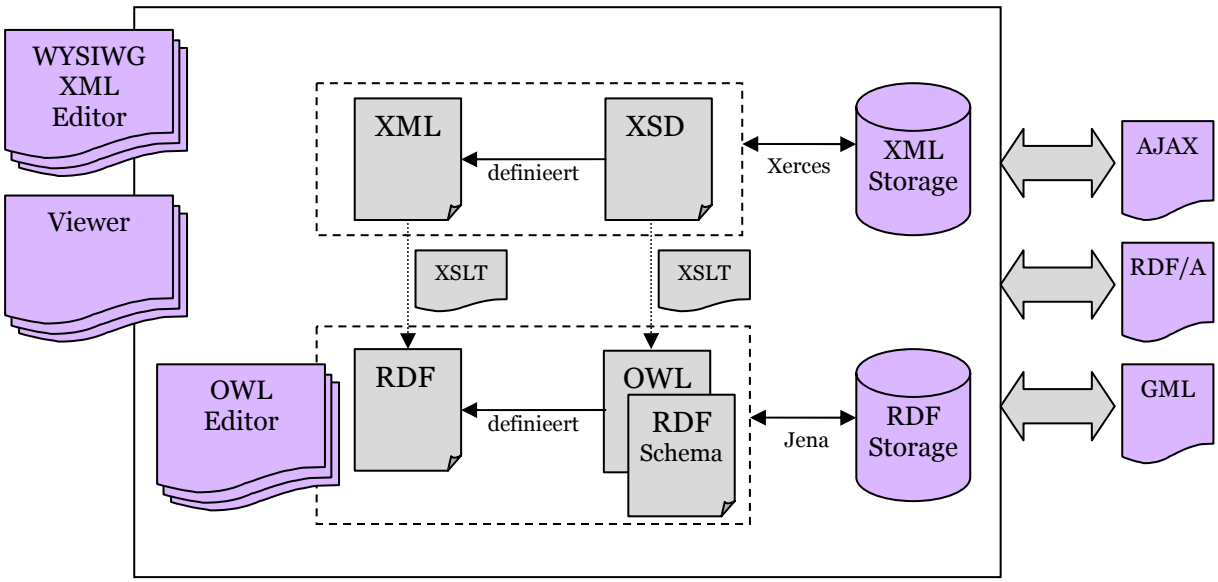
mr. C.J.G. Laan
Ministerie van Justitie, Directie Informatisering

dr. L. Mommers
Universiteit Leiden, Faculteit der Rechtsgeleerdheid

mr. S.W. Mul
Ministerie van Justitie, Kenniscentrum Wetgeving

Bijlage B – Architectuur

De technische aspecten van dit project richten zich voornamelijk op de opslag van regelingen en informatie over regelingen, en de software die nodig is om deze regelingen en informatie te bewerken en te raadplegen. Figuur 1 toont de architectuur voor de opslag en software. De architectuur is gebaseerd op diverse (open) standaarden van het World Wide Web consortium. Dit maakt dat deze architectuur goed aansluit op de ideeën voor het semantisch web en het internet zoals die op dit moment geïmplementeerd worden.



Figuur 1: Architectuur

Voor het goed doorzoeken van regelingen, en voor de functionaliteit die binnen dit project wordt vereist, volstaat het niet om de regelingen in een normaal tekst formaat, zoals Word, RTF of PDF op te slaan. Een meer gestructureerde vorm van opslag is vereist. In deze architectuur worden de regelingen daarom opgeslagen in gemarkeerd met de *eXtensible Markup Language* (XML). XML is een generieke standaard, die de methode van markeren definieert. De markeringen die daadwerkelijk worden gebruikt, moeten worden vastgelegd in een schema (*XML Schema Definition*, XSD). Voor het prototype is gebruik gemaakt van het METAlex/LS schema, onderdeel van de METAlex/CEN standaard.

Naast de regelingen en de structuur van de regelingen is er nog verdere informatie over de regelingen die opgeslagen moet worden opgeslagen. Dit is informatie over de status van documenten, en de relatie tussen de documenten. In tegenstelling tot de structuur kan deze informatie niet worden opgeslagen door stukken van de tekst te markeren. Deze informatie wordt daarom opgeslagen middels het *Resource Description Format* (RDF), een manier om bronnen te beschrijven (in plaats van te markeren). Net zoals XML is RDF een generieke methode, en moet specifiek schema worden gebruikt voor specifieke toepassingen. Dit gebeurt door middel van RDF Schema (RDFS). Alternatief kunnen de relaties worden beschreven in de *Web Ontology Language* (OWL), waarna er

met RDF naar de OWL relaties wordt verwezen. De structuur die opgeslagen is in XML kan middels de XSLT taal worden omgezet naar RDF, zodat alle toegevoegde informatie in hetzelfde formaat aanwezig is.

De XML bestanden worden opgeslagen in een XML opslag, die kan worden aangesproken via Xerces[12]. De RDF informatie is opgeslagen in een RDF opslag die kan worden aangesproken via Jena[13]. Jena en Xerces zijn geen standaarden, maar zijn wel open software, hetgeen ze een goede keuze maakt vergeleken met de vele gesloten alternatieven.

De opslag kan op drie verschillende wijzen worden aangesproken. Een OWL editor, zoals Protegé of Topbraid, kan de OWL bestanden raadplegen en wijzigen. Voor het wijzigen van de eigenlijke regelingen wordt gebruikt gemaakt van een *What You See Is What You Get* (WYSIWYG) editor. In een dergelijke editor krijgt de gebruiker de tekst in het uiteindelijke uitvoer formaat te zien, en niet in het (gemarkeerde) opslagformaat. Voor het prototype gebruiken we de MetaVex editor, een editor voor METAlex documenten, maar de open architectuur maakt het mogelijk dat ook andere editors met de bestanden gaan werken. Naast de editors kan de opslag ook worden geraadpleegd door viewers, programma's die alleen bedoeld zijn om de informatie te bekijken (maar niet te wijzigen).

Door de keuze voor open, veel gebruikte standaarden, kan het systeem goed aansluiten op andere standaarden zoals geodata opgeslagen in GML (de *Geographic Markup Language*)[14] en de AJAX voor interactieve webpagina's[15].

[12] http://xerces.apache.org/
[13] http://jena.sourceforge.net/
[14] http://portal.opengeospatial.org/
[15] AJAX staat voor "asynchronous JavaScript and XML" en bestaat uit een diverse technieken voor interactieve webapplicaties. Voorbeelden zijn te vinden op http://www.dmoz.org/Computers/Programming/Languages/JavaScript/AJAX/

Bijlage C – Genereren van wijzigingsteksten

Een wijzing van een regeling (een nota van wijziging, wijzigingswet, novelle of amendement) bestaat altijd uit een set instructies die aangeeft wat er gewijzigd moet worden. Echter, de auteur van zo'n wijziging zal doorgaans denken in termen van de nieuwe tekst, en niet in termen van de wijziging.

Binnen dit project is daarom een methode ontwikkeld om de "wijzigingsinstructies" te genereren uit een gewijzigde tekst. Dit betekent dat de auteur gewoon de bestaande tekst kan wijzigen, en dit dus niet hoeft uit te drukken in wijzigingsinstructies. Een tweede voordeel is dat de nieuwe tekst direct beschikbaar is, zodat de impact van de wijzigingen ook beter te beoordelen is voor de betrokkenen.

Een eerste overweging om de wijzigingen door de gebruiker bij te houden was om alle acties van de gebruiker vast te leggen. Het is dan gelijk bekend welke tekst is verwijderd, en wat er is ingevoegd. Dit zou erg efficiënt kunnen zijn, omdat we alleen de tekst hoeven te bekijken die door de gebruiker is bewerkt.
Een nadeel van deze methode is dat hij alleen goed werkt als de gebruiker in een keer alle juiste wijzigingen aanbrengt. In de praktijk zal dit niet het geval zijn: woorden worden ingevoegd, weer weggehaald, toch weer ingevoegd. Een lijst met alle handelingen van de gebruiker zou moeten worden schoongemaakt, wat neer zou komen op het vergelijken van de uiteindelijke tekst met de originele tekst. Er is daarom besloten om het opslaan van de gebruikersacties over te slaan[16], en gewoon de teksten te vergelijken

Dit betekent dat voor het automatisch generen van tekst uit de gewijzigde tekst twee acties moeten plaatsvinden:

1. De gewijzigde tekst moet worden vergeleken met het origineel, om de wijzigingen op te sporen;
2. De wijzigingen moeten in natuurlijke taal onder woorden worden gebracht.

In dit deel van de tekst hebben we het steeds over een wijzigingswet, maar dezelfde aanpak kan ook worden gehanteerd voor een nota van wijziging, amendement of novelle.

C.1. Opsporen van wijzigingen

Voor het vergelijken van twee bestanden bestaat al lange tijd een functie, genaamd *diff* (voor *difference*). Deze functie is ooit (in 1970) ontwikkeld als onderdeel van het besturingssysteem Unix, en waarvan inmiddels ook Java implementaties bestaan[17].

De werking van diff

Diff werkt in twee stappen. De eerste stap is het bepalen van de overlap tussen de twee bestanden. De overlap is uit beide bestanden te verkrijgen door een deel ervan weg te gooien. Als het eerste bestand "Onze Minister van Binnenlandse Zaken" bevat, en het

[16] Er wordt nog wel bijgehouden welke delen van het document zijn bewerkt, zodat bekend is welke delen van het document ongewijzigd zijn (en dus niet hoeven te worden gecontroleerd op wijzigingen).
[17] In het prototype wordt op dit moment gebruik gemaakt van de versie van incava.org.

tweede “Onze Minister van Defensie”, dan is de overlap “Onze Minister van”. Uit het eerste bestand is deze te bereiken door “Binnenlandse Zaken” weg te laten, en uit de tweede door “Defensie” weg te laten.

In veel gevallen zijn er meerdere overlappingen mogelijk, vooral bij lange teksten. Stel dat de inhoud van het eerste bestand “123146” is, en van het tweede bestand “1431967” (om het voorbeeld qua lengte beperkt te houden gebruiken we hier cijfers in plaats van tekst), dan zijn mogelijke overlappingen ondermeer:

- “1”, te verkrijgen door “23146” weg te laten uit het eerste bestand, en “143” en ”967” uit het tweede bestand.
- “1316”, te verkrijgen door “2” en “4” weg te laten uit het eerste bestand, en “4”, “9” en “7” uit het tweede bestand.
- “146”, te verkrijgen door “231” weg te laten uit het eerste bestand, en “319” en “7” uit het tweede bestand.

In dit soort gevallen gaat diff uit van de grootst mogelijk overlap (in dit geval “1316”), en dus dat er zo min mogelijk karakters veranderd zijn.
Voor het vinden van deze overlap bestaan verschillende algoritmes[18].

Nadat de overlap is bepaald, worden de handelingen beschreven die nodig zijn om van het eerste bestand naar het tweede bestand te komen. Voor elke handeling wordt beschreven:

1. het soort handeling: verwijdering, invoeging of verandering;
2. de locatie van de handeling: de positie[19] in het oude bestand (en vaak ook in het nieuwe bestand).

In geval van het eerder gegeven voorbeeld (vergelijk “123146” met “1431967”) zijn de benodigde handelingen dus:

1. vervang het tweede karakter (“2”) door “4”;
2. vervang het vijfde karakter (“4”) door “9”;
3. voeg “7” toe na het zesde karakter.

De abstracte inhoud even daargelaten lijkt deze beschrijving al erg op de inhoud van veel wijzigingswetten.

Diff combineren met METAlex

De uitkomst van de diff functie is niet direct geschikt voor het genereren van een wijzigingswet. Diff geeft de locatie van wijzigingen weer in “aantal karakters vanaf het begin”, terwijl een wijzigingswet de positie aangeeft door te relateren aan structuurelementen van het document (zoals “in artikel 12” en “na het tweede lid”). De computer beschikt over de benodigde informatie om dergelijke positiebepalingen te geven doordat we de structuur van de regelingen markeren door middel van METAlex.

Een mogelijkheid die we nu hebben is om de uitkomst van de diff functie terug te rekenen naar de structuur in het METAlex document. Als het 312e karakter is gewijzigd, en dit karakter bevindt zich binnen artikel 5 van de wet, dan weten we dus dat artikel 5 is gewijzigd. Dit is nogal omslachtig, zeker omdat we een betere aanpak tot onze

[18] Binnen de informatica is de (Engelse) term voor deze overlap “Longest Common Subsequence”; dit is de term die gebruikt wordt in de meeste literatuur.
[19] De positie wordt weergegeven door het aantal karakters te tellen. Een verwijdering heeft bijvoorbeeld betrekking op karakters 210 tot en met 216, tellend vanaf het begin van het bestand.

beschikking hebben. In plaats van de diff te gebruiken op de gehele tekst, kunnen we hem toepassen op de individuele structuurelementen. Dit betekent dat de locatie van een wijziging (in termen van structuurelementen) automatisch bekend is.

Bij het vergelijken van de documenten wordt eerst de documentstructuur (in METAlex) vergeleken. Elk structuurelement heeft in METAlex een unieke aanduiding[20]. Op grond van deze aanduiding kunnen we de volgende conclusies trekken over de wijzigingen binnen het document:

1. Indien een element met aanduiding *x* wel voorkomt in het originele document, maar niet in het gewijzigde document, dan is dit (gehele) element verwijderd. Het is verder niet nodig diff toe te passen op de inhoud.
2. Indien een element met aanduiding *x* wel voorkomt in het gewijzigde document, maar niet in het originele document, dan is dit (gehele) element ingevoegd. Het is verder niet nodig diff toe te passen op de inhoud.

Alleen als het structuur element in beide documenten voorkomt hoeven we diff toe te passen om eventuele wijzigingen op te sporen. Dit betekent dat we de diff functie toepassen op zinnen en zinfragmenten, en niet op grotere stukken tekst.

METAlex geeft ook houvast voor twee specifiekere wijzigingen in een regeling:

1. Verplaatsing van een element. Diff kan dit herkennen, omdat uit wordt gegaan van de langs mogelijke overlap, terwijl hier juist een kleiner stuk overlap moet worden herkend. Door de unieke identificatie binnen METAlex wordt een verplaatsing wel direct herkend. (Het element komt in beide documenten voor, maar niet op dezelfde plaats.)
2. Hernummering is in feite gewoon een wijziging van een aanduiding, en wordt door diff ook als zodanig gerapporteerd. Door koppeling met de METAlex structuur wordt duidelijk dat er sprake is van hernummering.

Het gebruik van METAlex heeft een tweede voordeel (naast de eenvoudigere locatiebepalingen). Het algoritme voor het vinden van de overlap (dat door diff wordt gebruikt) is kwadratisch van aard. Dit betekent dat het minder tijd kost diff tien keer toe te passen op een zin van 100 karakters, dan op een blok tekst van 1000 karakters. Door diff dus toe te passen op de individuele elementen in plaats van de gehele tekst, en daarbij verwijderde en nieuwe elementen buiten beschouwing gelaten, wordt de hele procedure versneld.

Editor

Het gebruik van de METAlex identificatie stelt bepaalde eisen aan de editor en het gebruik van de editor. Om te beginnen mag de editor de identificatie van een verwijderd element niet gelijk hergebruiken bij een nieuw ingevoegd document. Zou dit wel gebeuren, dan zou het algoritme voor het opsporen van de verschillen tot de conclusie komen dat het om hetzelfde artikel gaat. Dit is echter een triviaal probleem, dat makkelijk te voorkomen is in het programma.

Moeilijker is het om onverwachte acties van de gebruiker tegen te gaan. De gebruiker kan het algoritme op twee manieren van slag brengen:

[20] Op dit moment voegt de editor een unieke “schaduw” aanduiding toe om een unieke aanduiding te kunnen garanderen, ook als het bestand zich niet aan de MetaLex eis van een unieke aanduiding houdt.

1. Door een compleet structuur element te verwijderen, en vervolgens dit ongedaan te maken door een nieuw structuur element met dezelfde inhoud in te voegen (in plaats van de "ongedaan maken" functie te gebruiken.
   Het nieuwe element heeft een andere identificatie, en het algoritme concludeert dus dat er een element is verwijderd en een is ingevoegd, terwijl in werkelijkheid de tekst gelijk is gebleven.
2. De gebruiker verwijdert de tekst binnen een structuurelement, en voert daarna nieuwe tekst in. Het element blijft dus behouden, en het algoritme concludeert dat het element simpelweg gewijzigd is, terwijl hij eigenlijk vervangen is.

Voor het eerste probleem is op dit moment geen oplossing geïmplementeerd; de hoop is dat dit probleem zich in de praktijk niet vaak voordoet, en dat gebruikers de "ongedaan maken" functie zullen gebruiken. Mocht dit niet zo zijn, dan is het mogelijk om bij invoegingen op de positie van eerdere verwijderingen de teksten te vergelijken om te bepalen of het niet om dezelfde tekst gaat.

Voor het tweede probleem is wel een controle ingebouwd, omdat dit gerelateerd is aan een andere vraag, namelijk: Hoeveel mag een zin gewijzigd zijn voordat het als een nieuwe zin wordt beschouwd? Deze vraag is relevant voor het genereren van de tekst van een wijzing (zie Sectie B.2).
De controle die wordt uitgevoerd is door het aantal gewijzigde woorden (verwijderd, ingevoegd en veranderd) te vergelijken met het totale aantal woorden (de woorden uit de originele zin plus eventuele toevoegingen). Indien deze waarde boven een bepaalde drempel uitkomt[21], wordt de zin beschouwd als een nieuwe zin die de oude vervangt; onder die drempel wordt nog uitgegaan van een vervanging.

De editor levert nog een bijdrage aan de efficiency van het algoritme. Bij het aanmaken van een gewijzigde tekst markeert de editor die stukken tekst die door de gebruiker zijn bewerkt. Bij het bepalen van de wijzigingen kunnen de niet bewerkte delen worden overgeslagen.

C.2. Genereren van de tekst

Nadat met de hierboven beschreven methode de wijzigingen zijn bepaald, moeten deze nog in natuurlijke tekst worden beschreven. Hiervoor wordt gebruik gemaakt van een aantal standaardformuleringen. Voor de meest voorkomende gevallen zijn dit:

- Verwijdering: *Artikel 12 komt te vervallen.*
- Invoeging: *Na artikel 12 wordt een nieuw artikel toegevoegd, luidende: …*
- Wijziging: *Artikel 12 wordt als volgt gewijzigd: … wordt vervangen door ….*

Een vastgestelde wijziging kan dus eenvoudig worden uitgedrukt in natuurlijke taal door de bij de wijziging behorende standaard zin te nemen en deze aan te passen. De locatie kan worden uitgelezen door binnen de METAlex structuur naar de juiste locatie te volgen, en daarbij de aanduidingen van de elementen aan elkaar te rijgen tot een locatie beschrijving. Bijvoorbeeld: een wijziging vindt plaats in een lijstelement met aanduiding " a.", binnen een lid met aanduiding "1", binnen een artikel met aanduiding "artikel 21". Deze aanduidingen moeten aaneengeregen worden tot "Artikel 21, eerste lid, onderdeel

[21] De waarde van deze drempel moet proefondervindelijk worden vastgesteld; het gaat er om een punt te vinden waar het nog "kloppend voelt" voor de gebruiker. Op dit moment wordt een drempel gehanteerd van 0.8.

a". (Dit betekent naast het aaneenrijgen in dit geval ook de toevoeging van een aantal labels, en de conversie van cijfer naar rangtelwoord.)
De betrokken teksten kunnen worden overgenomen uit de geregistreerde wijzigen.

Op deze manier wordt dus op eenvoudige wijze een lijst met wijzigingen omgezet naar een beschrijving van die wijzigingen in natuurlijke taal. Echter, deze beschrijving komt op bepaalde punten nog niet overeen met de beschrijving die een menselijke auteur zou geven. Op bepaalde punten is dit ook slecht voor de leesbaarheid. Een andere beperking is dat met *diff* precies de tekst is gevonden die is gewijzigd, en het is dus ook deze tekst die terugkomt in de beschrijving. De Aanwijzingen voor de Regelgeving (aanwijzing 231) schrijven echter voor dat bij wijzigingen een gehele begripseenheid wordt vervangen. Dit is mogelijk een groter deel van de tekst dan dat door *diff* gemarkeerd is. Mogelijkheden om dit te verbeteren zijn:

- Altijd een complete (deel)zin vervangen. Dit betekent dat er vaak teveel wordt teruggegeven, hetgeen ook niet perfect is, maar waarschijnlijk beter dan te weinig.
- Gebruik natuurlijke taalverwerking om de zin te ontleden, zodat de juiste eenheid kan worden teruggegeven. Deze oplossing is complexer, maar leidt wel tot een beter resultaat.

Als alternatief kan men stellen dat met de komst van de redactieomgeving eindgebruikers niet langer de wijzigingswet hoeven te raadplegen om zich op de hoogte te stellen van de wijzigingen. In plaats daarvan kan er eenvoudig aan de hand van de verschillende consolidaties worden geïllustreerd wat er veranderd. In dit opzicht is het niet langer nodig (maar natuurlijk nog steeds wel wenselijk) om de wijzigingswet optimaal leesbaar te maken, en is de aanwijzing mogelijk verouderd.

Doorgenummerde artikelen

Het bepalen van de positie gebeurd door het aaneenrijgen van de aanduidingen van alle structuurelementen waarbinnen de wijziging zich bevindt. Als de regeling veel lagen bevat, kan dit leiden tot lange beschrijvingen, zoals: Hoofdstuk 3, Paragraaf 2, Artikel 23, eerste lid, onderdeel j.

In veel gevallen is dit nog niet de correcte beschrijving, omdat de artikelen vaak zijn doorgenummerd. In dit geval kunnen bovenliggende lagen worden overgeslagen. Een volledige beschrijving inclusief hoofdstuk, paragraaf en/of andere lagen is in dit geval niet foutief, maar wel ongebruikelijk, en slecht voor de leesbaarheid.

Om dit te corrigeren moet eerst worden bepaald binnen welke context de artikelen genummerd zijn. Zijn ze door de hele regeling heen doorgenummerd, of slechts per hoofdstuk, paragraaf of een ander deel? Dit kan worden bepaald door eerst alle artikelaanduidingen te bekijken. Zijn ze stuk voor stuk uniek, dan zijn ze doorgenummerd door de regeling. Zo niet, dan wordt gekeken naar de volgende laag (bijvoorbeeld hoofdstukken) en wordt er gekeken of ze binnen elk van die delen van die laag (elk individueel hoofdstuk) uniek zijn, totdat de juiste laag gevonden is.

De beschrijving bestaat dat uit aansluitend:

- alle aanduidingen tot en met de structuurlaag waarin de artikelen uniek zijn; dit is meestal de hele regeling of de hoofdstukken (bij een hele regeling wordt er dus niets aan de beschrijving toegevoegd);
- de aanduiding van het artikel;
- alle aanduidingen van de structuurlagen onder het artikel.

Groepering

Het simpelweg vertalen van elke wijziging naar een zin in natuurlijke taal kan leiden tot opsomming als:
In artikel 22 wordt u *vervangen door* v.
In artikel 22 wordt w *vervangen door* x.
In artikel 22 wordt y *vervangen door* z.

Hoewel dit correct is, is het geen mooie tekst, en zeker niet de tekst zoals die wordt gehanteerd door menselijke auteurs. De leesbaarheid is matig. Doorgaans worden dergelijke beschrijvingen gegroepeerd, hetzij in een zin, hetzij in een lijst.

Bij het generen van de tekst houdt dit dus in dat eerst alle wijzigingen op de juiste wijze moeten worden gegroepeerd, en dat daarna de juiste tekst wordt aangemaakt.

C.3. Samenloopbepalingen

Een manier om samenloop van wijzigingen te voorkomen is door sommige wijzigingen onder voorwaarde op te nemen. Op deze manier kan kunnen de wijzigingen die worden aangebracht door een wijzigingswet veranderen al na gelang de wet wordt ingevoerd voor of na de invoering van een andere wet.

Op dit moment voorziet de editor niet in de functionaliteit die nodig is om dit soort conditionele wijzigingen te maken en toe te passen. Dergelijke functionaliteit vereist een kleine aanpassing aan het opslagformaat. Elke wijziging moet dan optioneel voorzien kunnen worden van een voorwaarde waaronder die wijziging doorgang vindt.

De vraag is vervolgens welke vorm deze voorwaarde aan moet nemen:

1. Gewone tekst: dit houdt in dat de applicatie de voorwaarde niet begrijpt en dus ook niet kan toepassen. Bij consolidatie zal de gebruiker moeten aangeven wat de status van elke voorwaarde is.
   (Op een gegeven moment is voor een ingevoerde wijzigingswet bekend wat de waarde van elke voorwaarde is, en kan dit worden opgeslagen, hetgeen betekent dat vanaf dat moment de gebruiker niet meer gevraagd wordt.)
2. Gemarkeerde tekst: dit houdt in dat de voorwaarde op een of andere wijze gemarkeerd is, zodat de applicatie de betekenis van de voorwaarde kent, en hem dus zelf kan toepassen. Dit kan alleen worden gedaan voor die voorwaarden die door bij de applicatie bekend zijn, de rest moet worden afgehandeld als onder punt 1.

Doorgaans zal de tweede mogelijkheid de voorkeur verdienen, maar dit vereist een studie van de mogelijke voorwaarden. Vermoedelijk zijn de meeste voorwaarden gerelateerd aan de data[22] gerelateerd aan de wet zelf, en aan andere regelingen. Dergelijke voorwaarden zijn goed uit te drukken in de vorm van METAlex metadata, en daarmee goed in de architectuur te verwerken. Zijn in de praktijk ook “wildere” vormen van voorwaarden mogelijk, dan zal wellicht voor optie 1 uit de lijst hierboven moeten worden gekozen.

De vorm van de voorwaarden is niet de grootste barrière die genomen moet worden voordat de applicatie conditionele wijzigingen kan ondersteunen. De huidige interface

[22] Zoals de publicatie datum, datum van inwerkingtreding en datum van uitwerkingtreding.

gaat ervan uit dat de gebruiker een originele tekst bewerkt tot het gewenste eindresultaat. Als er conditionele wijzigingen moeten worden gebruikt, is er echter geen sprake van één enkel eindresultaat; er zullen er meerdere zijn.

De interface van de editor zal dus zodanig moeten worden aangepast dat de gebruiker bepaalde wijzigingen conditioneel kan maken, en mogelijk zelfs diverse (conflicterende) wijzigingen moet kunnen creëren die van elkaar gescheiden worden middels voorwaarden. Tevens moet de gebruiker die voorwaarden kunnen koppelen aan de wijzigingen. Als de applicatie de betekenis moet kennen van de verschillende wijzingen, zal er ook een uitgebreidere interface moeten zijn voor de creatie van voorwaarden op een voor de applicatie begrijpelijke manier.

Al met al vereist de ondersteuning van conditionele wijzigingen een uitgebreide ingreep, en dan vooral in de user interface. Hoewel conditionele wijzigingen geen uiterst zeldzaam verschijnsel zijn, komen ze ook niet bij elke wijziging voor. Aan te raden valt dan ook om deze functionaliteit niet de hoogste prioriteit te geven, en eerst zaken uit te voeren waar meer directe winst te behalen is. Wel is het verstandig rekening te houden met een eventuele latere uitbreiding in deze richting.

C.4. Wijzigingen van wijzigingen

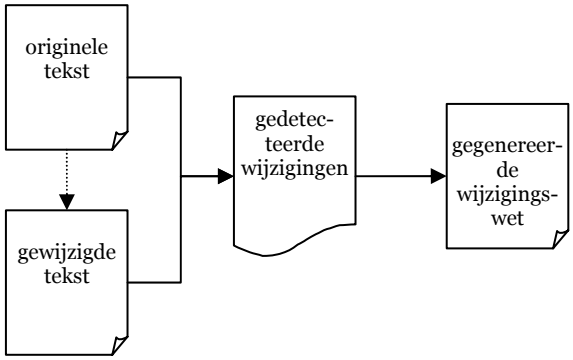
Het beschreven algoritme werkt voor wijzigingen in een originele tekst. Voor het genereren van wijzigingen in een wijzigingswet moeten er wat stappen worden toegevoegd.

In principe kan hetzelfde algoritme worden gebruikt; de wijzigingswet wordt dan voor deze nieuwe operatie gezien als de originele tekst, waarvoor een nieuwe wijziging wordt gegenereerd. Dit heeft echter tot nadeel dat de “echte” originele tekst (diegene waar de wijzigingswet betrekking op heeft) niet in het proces betrokken is. Dit betekent:

1. dat de wetgevingsjurist zijn wijzigingen niet in die “echte” originele tekst kan aanbrengen, maar toch weer zijn wijzigingen moet beschrijven;
2. dat er van de originele wet geen directe consolidatie meer aanwezig is;
3. dat de aan te brengen wijzigingen niet meer door de applicatie herkend kunnen worden, en dat die dus niet meer voor andere toepassingen kunnen worden gebruikt (zoals de generatie van consolidaties, zie Bijlage D).

In feite gaan dus een groot deel van de beoogde voordelen verloren.

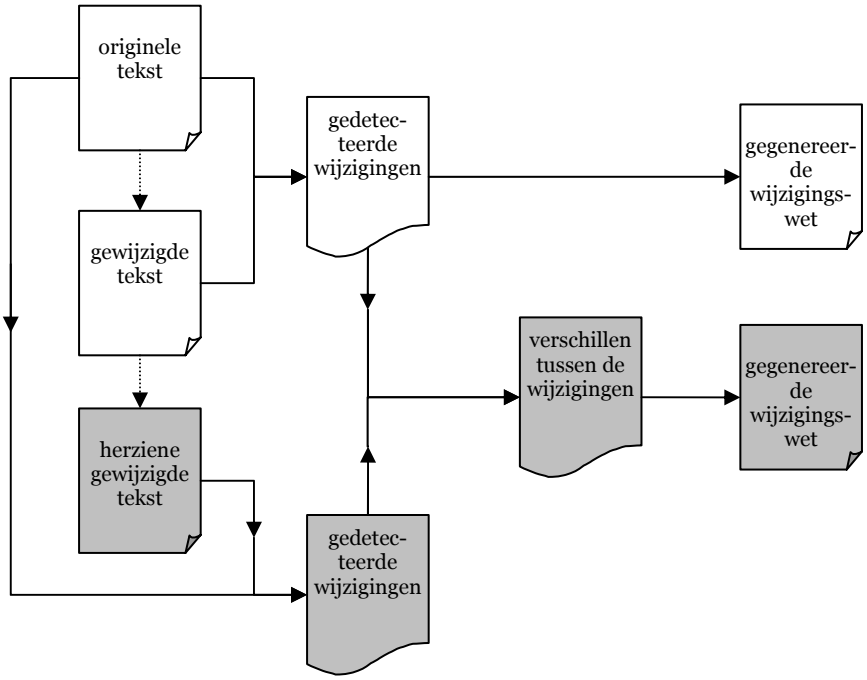
Indien een wijzigingswet is gemaakt met de redactieomgeving, gebaseerd op het beschreven algoritme, dan zijn er een aantal documenten beschikbaar (zie Figuur 2).



Figuur 2: Documenten betrokken bij een "enkelvoudige" wijziging

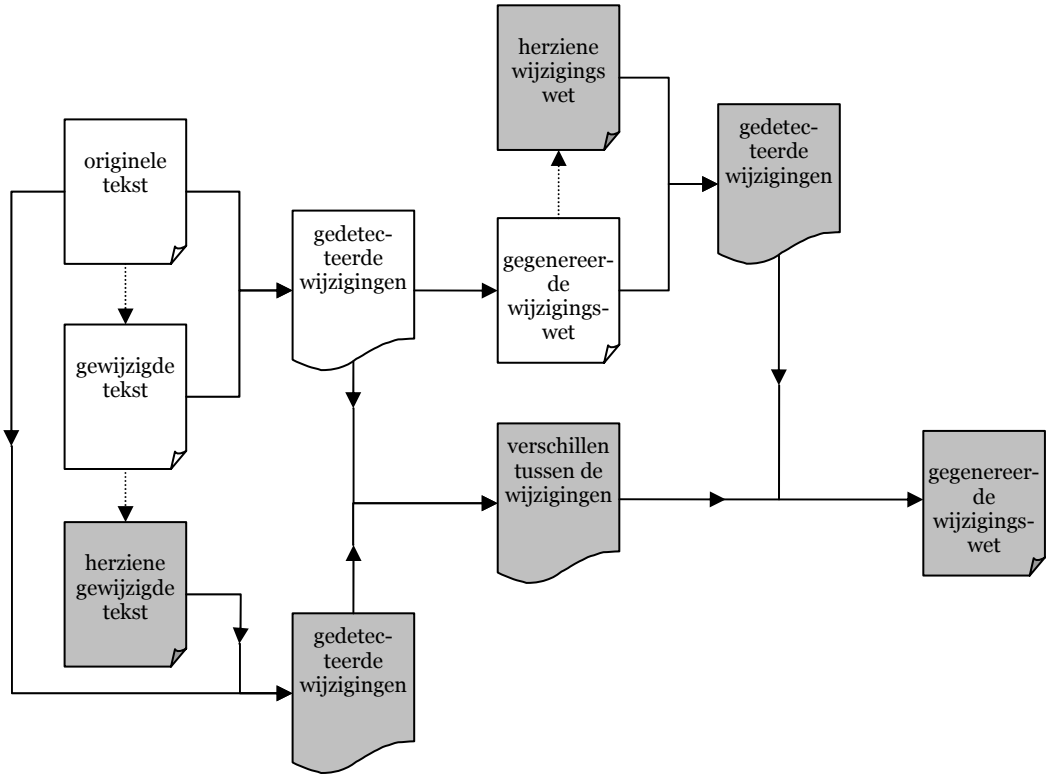
De wetgevingsjurist creëert een gewijzigde tekst uit de originele tekst (gestippelde pijl). De omgeving vergelijkt deze twee documenten en maakt een lijst met gedetecteerde wijzigingen. Uit deze lijst wordt uiteindelijk een tekst voor de wijzigingswet gegenereerd (gewone pijlen).

Om later de wijzigingen aan te passen op een manier die de redactieomgeving herkent, moet de wetgevingsjurist eigenlijk de gewijzigde tekst aanpassen, zodat de redactieomgeving de nieuwe wijzigingen kan opsporen op dezelfde wijze waarop de eerste wijzigingen waren opgespoord.
Vervolgens moeten de wijzigingen met elkaar vergeleken worden, om de verschillen op te sporen. Op basis van die verschillen kan uiteindelijke de tekst worden gegenereerd voor de tweede wijzigingswet (die de eerste wijzigingswet wijzigt). Figuur 3 toont dit proces. De witte vakken tonen het proces om een wijzigingswet te maken, overeenkomstig Figuur 2). De grijze vakken tonen de documenten die betrokken zijn bij het genereren van een wijziging op de wijzigingswet.
Hoe kan de transparantie en productie van (wijzigings)wetgeving worden verbeterd met ICT?



Figuur 3: Genereren van een wijziging op een wijziging

De hele procedure wordt nog complexer als we er rekening mee houden dat de wijzigingswet niet alleen wijzigingen op de originele wet bevat, maar mogelijk ook nog overgangs- en slotbepalingen. Deze bepalingen kunnen ook worden herzien. Dit is echter een wijziging in de tekst van de wijzigingswet, en niet in de originele tekst. Dit betekent dat de procedure "dubbel" moet worden uitgevoerd: de wetgevingsjurist bewerkt de originele wettekst om de wijzigingen in die tekst weer te geven, en hij bewerkt de wijzigingswet om de wijzigingen in de overgangs- en slotbepalingen van die wet weer te geven[23]. In dit geval wordt de volledige procedure dus:



Figuur 4: Genereren van een wijziging op een wijziging, inclusief wijzingen op overgangs- en slotbepalingen

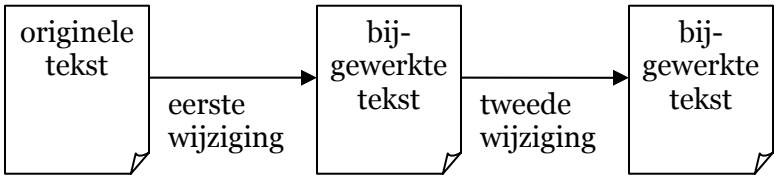
Het algoritme dat gebruikt wordt voor het opsporen van deze wijzigingen is niet anders dan het algoritme dat in eerste instantie gebruikt wordt voor de wijzigingen in de originele tekst. Voor deze functionaliteit ligt de uitdaging vooral in de user interface; het hele complex van wijzigingen moet op een overzichtelijke manier worden getoond aan de gebruiker. Verder moet worden voorkomen dat hij wijzigingsinstructies met de hand wijzigt, aangezien dat maakt dat de door de applicatie geregistreerde wijzigingen niet meer kloppen met wat er op papier staat, en deze gegevens dus niet verder gebruikt kunnen worden.

[23] Dit verschilt niet erg van de situatie waarin een wijzigingswet wijzigingen op twee of meer regelingen bevat.

Bijlage D – Automatisch genereren van consolidaties

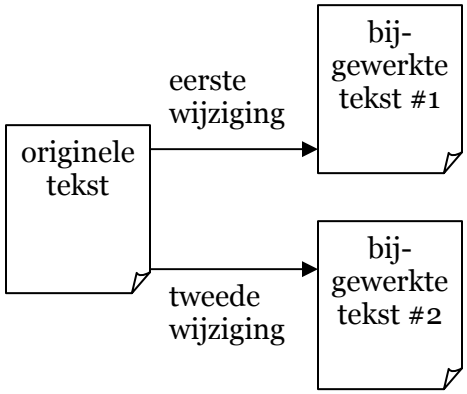
Naast het automatisch genereren van wijzigingen moet ook de tegenovergestelde route beschikbaar zijn: het automatisch genereren van consolidaties. Immers, voor het toepassen van bijgewerkte wetgeving zal het handig zijn om de bijgewerkte tekst te kunnen gebruiken in plaats van in het hoofd de wijzigingen uit wijzigingswetten toe te passen op de originele tekst.

De in Bijlage C voorgestelde methode voor het schrijven van wijzigingswetten zorgt ervoor dat er altijd een geconsolideerde versie beschikbaar is. Dit suggereert dat er geen noodzaak is voor het automatisch consolideren; de consolidatie is al beschikbaar als product van het wijzigingsproces. Dit is helaas niet het geval, het werkt zolang wijziging na elkaar worden geschreven en toegepast, zoals afgebeeld in Figuur 5.



Figuur 5: Wijzigingen in volgorde toegepast

In de praktijk worden wijzigingen niet eenvoudigweg na elkaar gemaakt. Het is goed mogelijk dat meerdere wijzigingen parallel worden voorbereid. In dit geval is het niet mogelijk om op de bijgewerkte tekst van de andere wijziging verder te werken. In dit geval is er nog geen consolidatie beschikbaar om mee verder te werken. Daarnaast komt het voor dat niet bekend is dat er een andere wijziging voorbereid wordt.



Figuur 6: Parallel voorbereide wijzigingen

Merk op dat “voorbereiding” in deze context breed moet worden gezien. Totdat een wijziging is aangenomen en is ingevoerd, is het niet zeker in welke volgorde de wijzigingen moeten worden toegepast.

Doordat wijzigingen parallel worden voorbereid, zijn dus niet alle consolidaties beschikbaar als bijproduct van het wijzigingsproces met de editor. Als in de situatie geschetst in Figuur 6 wijziging #2 als eerste wordt ingevoerd, en daarna wijziging #1, dan is bijgewerkte tekst #2 wel een geldige consolidatie (geldig in de periode tussen de invoering van wijziging #2 en wijziging #1). Bijgewerkte tekst #1 is echter een tekst van een versie die nooit bestaan heeft. De uiteindelijke consolidatie na toepassing van #2 en #1 is nog niet gemaakt. Hiervoor is het dus wenselijk dat de ontbrekende consolidaties automatisch gecreëerd kunnen worden.

Aanpak

Zoals is uitgelegd in Bijlage C worden bij het genereren van een wijziging uit een gewijzigde tekst eerst de wijzigingen opgespoord, die dan later in natuurlijke taal worden beschreven. Voordat de wijzigingen in natuurlijke taal worden omgezet, zijn ze dus al in een wiskundige notatie aanwezig. Deze notatie omvat het soort wijziging (verwijdering, invoeging of wijziging) en de positie in de tekst. Door deze informatie apart op te slaan, kan hij later worden gebruikt voor een nieuwe consolidatie.

De informatie die beschikbaar is, vormt een eenduidige instructie aangaande de aanpassingen die moeten worden gemaakt in de originele tekst om tot de consolidatie te komen. In die zin vergt de generatie van consolidaties dus geen uitbreiding van de architectuur, maar slechts de toevoeging van nieuwe functionaliteit.

Bij het toepassen van meerdere wijzigingen achter elkaar kan het gebeuren dat bepaalde wijzigingen niet meer mogelijk zijn. In dit geval is er sprake van samenloop (zie Bijlage F). De aanpak die in dit geval moet worden gevolgd is dat de wijziging die niet meer mogelijk is, simpelweg niet wordt uitgevoerd[24].

[24] Uitkomst van de werkgroep bijeenkomst voor dit project, d.d. 8 april 2008

Bijlage E – Opslag van wijzigingen

Voor het complete systeem dat binnen dit project wordt voorgesteld, moeten een aantal soorten informatie worden opgeslagen voor elke regeling:

1. De tekst van de regelingen.
   De tekst is nodig voor mensen die de regelingen willen raadplegen; zij moeten de gewone tekst voor ogen krijgen. Verder vormt de tekst de basis voor wijzigings- en consolidatie-operaties.
2. De structuur van de regelingen.
   Voor het maken van wijzigingen en consolidaties geeft de structuur belangrijke houvast, zoals is beschreven in Bijlagen C en D. Verder geeft de structuur mensen die de regelingen raadplegen extra functionaliteit: gestructureerde tekst is beter te doorzoeken en aan elkaar te linken.
3. De betekenis van wijzigingen die beschreven staan in de regelingen.
   Deze informatie is nodig voor het automatisch genereren van consolidaties.

Door de regelgeving op te slaan in METAlex formaat is zowel de tekst als de structuur beschikbaar. Aangezien METAlex eenvoudig om te zetten is naar andere formaten (zoals PDF) is deze manier ook niet erg beperkend.

De betekenis van wijzigingen is echter geen structuur, en is niet opgeslagen in een METAlex document. In feite gaat het hier om informatie over de tekst: metadata. METAlex beschikt over een mechanisme om metadata op te slaan, en daarmee wordt informatie als de publicatiedatum en de inwerkingtredingdatum bewaard.

De betekenis van wijzigingen kan op dezelfde manier worden opgeslagen. Echter, omdat het om relatief veel informatie gaat, die bij relatief weinig acties wordt geraadpleegd, is het aan te raden om deze informatie niet in hetzelfde bestand op te slaan.

Bijlage F – Samenloop van wijzigingswetten

Samenloop is een bekend juridisch begrip, en gaat over de situatie waarin meerdere regels tegelijk op een situatie van toepassing zijn, en deze regels elkaar mogelijk versterken of tegenspreken.

Een exactere beschrijving komt van Janssen (2002)[25], die stelt dat samenloop zich voordoet (of kan voordoen) als voor hetzelfde rechtsfeit(encomplex) tussen dezelfde rechtssubjecten en in dezelfde rechtsbetrekking verschillende rechtsregels tegelijk van toepassing zijn.

Voor wijzigingswetten is deze definitie wat erg breed, maar de essentie is hetzelfde: twee regels (in dit geval: wijzigingen) zijn tegelijk van toepassing, en botsen met elkaar. Dit probleem ontstaat doordat er parallel wordt gewerkt aan wijzigingen. De impact op wijzigingen op bestaande regelgeving wordt bekeken, maar de impact op andere wijzigingen niet, omdat men simpelweg niet op de hoogte is van deze andere wijzigingen die worden voorbereid. Aangezien het politiek niet altijd wenselijk is om bekend te maken waaraan wordt gewerkt, wordt deze informatie niet gedeeld.

De samenloop kan zich voordoen in de structuur van het document, of in de inhoud (betekenis). Dit is een belangrijk verschil voor dit project; beide vormen van samenloop worden hieronder verder beschreven.

F.1. Samenloop in structuur

Samenloop in structuur houdt in dat twee wijzigingen allebei effect hebben op hetzelfde structuur element. Voor de meest voorkomende veranderingen (invoeging, wijziging en verwijdering), leidt dit tot de volgende situaties, afhankelijk van de volgorde waarin ze uiteindelijk worden uitgevoerd.

Wijziging na wijziging

Indien twee wijzigingen na elkaar worden uitgevoerd, dan kan het zijn dat een van de twee wijzigingen verloren gaat.

Bijvoorbeeld:
Wijziging #1: *In artikel 12 wordt* x *vervangen door* y *en* u *door* v.
Wijziging #2: *In artikel 12 wordt* x *vervangen door* z *en alle voorkomens van* v *door* w.

Als wijziging #1 eerst wordt uitgevoerd, dan wordt de vervanging van *x* door *z* onmogelijk gemaakt, terwijl *u* mogelijk onterecht wordt vervangen door *w*.

Wijziging na verwijdering

Bij een wijziging naar verwijdering kan de wijziging niet worden doorgevoerd.

[25] Janssen, J.F.M. Wanneer is sprake van samenloop? In: Houben, L.S.J. e.a. *Samenloop*. Deventer: Kluwer (2007).

Verwijdering: *Artikel 12 komt te vervallen.*
Wijziging: *In artikel 12 wordt* x *vervangen door* y *en* u *door* v.

Invoeging na verwijdering

Bij een invoeging na een verwijdering is de plaatsbepaling voor de nieuw in te voegen tekst onduidelijk geworden.

Verwijdering: *Artikel 12 komt te vervallen.*
Invoeging: *Na artikel 12 wordt een nieuw artikel ingevoegd, luidende: Artikel 12a xxx*

In de praktijk zal dit geen serieus probleem worden, aangezien doorgaans nog goed kan worden bepaald waar het nieuwe artikel moet komen.

Invoeging na invoeging

Als hetzelfde structuurelement twee keer wordt ingevoegd dan is het resultaat dat er geen unieke aanduiding meer is. Na bijvoorbeeld de volgende twee samenlopende invoegingen is er twee keer een artikel 12a.

Na artikel 12 wordt een nieuw artikel ingevoegd, luidende: Artikel 12a xxx
Na artikel 12 wordt een nieuw artikel ingevoegd, luidende: Artikel 12a yyy

Het gevolg is dat de verwijzingen naar artikel 12a ambigu worden.

Verwijdering na verwijdering

Bij verwijdering na verwijdering is het onmogelijk de tweede instructie uit te voeren, indien het te verwijderen element absoluut genoemd staat, zoals bij *Artikel 12 komt te vervallen.*
Samenloop van twee verwijderinginstructies is in dit geval niet erg, omdat het beoogd effect van beide wijzigingen bereikt wordt.

Indien er geen sprake is van een absolute verwijzing, maar van een relatieve, zoals bij *Het laatste lid wordt verwijderd,* dan ontstaan er wel problemen, omdat samenloop dan leidt tot het verwijderen van de twee laatste leden in plaats van alleen het laatste lid.

Verwijdering na wijziging

Als een verwijdering een wijziging opvolgt, dan betekent dit dat ook de wijziging ongedaan wordt gemaakt.

Totaalbeeld

Er zijn in totaal dus zes mogelijkheden voor structurele samenloop:
1. Wijziging na wijziging
2. Wijziging na verwijdering
3. Invoeging na verwijdering
4. Invoeging na invoeging
5. Verwijdering na verwijdering
6. Verwijdering na wijziging

Verwijdering na invoeging zal doorgaans geen probleem moeten opleveren, aangezien de verwijdering geen invloed heeft op de ingevoegde tekst. Wijziging naar invoeging en invoeging naar wijziging kan ook geen probleem leveren.

Van de situaties waarbij wel een probleem kan ontstaan is er alleen bij een dubbele invoeging altijd sprake van een foutieve situatie; in de andere gevallen kan het eindresultaat nog steeds correct zijn. Zeker bij een verwijdering die volgt op een wijziging is het resultaat doorgaans zoals bedoeld.

Een operatie is in de bovenstaande lijst nog niet behandeld, namelijk de hernummering. Deze komt niet vaak alleen voor, maar hangt doorgaans samen met een invoeging of verwijdering. Indien hernummering wordt uitgevoerd van een tekstelement dat vervolgens wordt gewijzigd, dan lijdt dit vrijwel zeker tot een foutieve situatie. Hernummering is waarschijnlijk de meest voorkomende veroorzaker van samenloop problemen.

F.2. Samenloop in inhoud

Er is sprake van samenloop van de inhoud als beide wijzingen zonder problemen (qua structuur) kunnen worden uitgevoerd, maar leiden tot een tegenspraak in de tekst. Dit is een vrij zeldzaam verschijnsel, maar het kan voorkomen.

F.3. Detecteren van samenloop

In een aantal gevallen leidt de samenloop er toe dat een wijziging niet toepasbaar is. Deze vormen van samenloop zijn eenvoudig te detecteren door een consolidatie proces te doorlopen; onmogelijke acties leiden dan tot foutmeldingen van de applicatie. Hiermee zijn deze vormen van samenloop op te sporen.

Andere vormen van samenloop in structuur zijn niet zo eenvoudig op te sporen, omdat de resulterende tekst mogelijk gewoon correct is. Dit is met een computer, die geen kennis heeft van de betekenis van de tekst, niet op te sporen. Hetzelfde geldt voor samenloop in de inhoud.

Voor deze wijzigingen kan alleen potentiële samenloop worden opgespoord. Dit kan door de verschillende wijzigingsvoorstellen naast elkaar te houden, en te bepalen of er overlap is tussen de gewijzigde elementen. Wijzigen beide voorstellen hetzelfde tekstelement, dan is er mogelijk sprake van samenloop, en kan de gebruiker daarvan op de hoogte worden gebracht.

Bijlage G – Automatisch hernummeren van voorstellen

Uit de werkgroep is naar voren gekomen, dat veel tijd verloren kan gaan bij het hernummeren van een (in voorbereiding zijnd) voorstel en de daarbij behorende documenten. Een interessante uitbreiding voor de redactieomgeving is dan ook een hernummer functie, waarbij ook de verwijzingen worden bijgewerkt.

G.1. Hernummeren

De redactie omgeving ondersteunt op dit moment het hernummeren van een document in zijn geheel. Dit houdt in dat alle onderdelen opnieuw worden genummerd[26]. Dit is een redelijke eenvoudige taak, aangezien met MetaLex is vastgelegd wat de structuur van het document is, en op welke locaties de nummers worden ingevoegd.

Voor het nummeren kan de gebruiker een nummering schema opgeven, waarbij hij voor elke laag in het document aangeeft hoe het genummerd moet zijn. Hierbij kunnen een aantal verschillende aspecten worden behandeld. Om te beginnen kan de gebruiker het soort nummering vaststellen:

- Arabische cijfers
- Romeinse cijfers (hoofdletters of kleine letters)
- Letters (hoofdletters of kleine letters)

Aan het "nummer" kunnen nog andere (altijd aanwezige) tekens worden toegevoegd, zoals een punt of een graadteken (°).

De omgeving gaat er in basis vanuit dat nummering herstart wordt binnen elke nieuwe laag, dus de eerste paragraaf van elk hoofdstuk is altijd 1 (er vanuit gaande dat dat de gekozen nummering is). Eventueel is het mogelijk om de nummering van hoger gelegen lagen op te nemen binnen een nummer. Hiermee kan er bijvoorbeeld voor worden gezorgd dat de eerste paragraaf van hoofdstuk 1 als nummer 1.1 krijgt, terwijl de eerste paragraaf van hoofdstuk 2 het nummer 2.1.
Hierbij mogen lagen worden overgeslagen; het is mogelijk om in de artikelnummering wel het nummer van het hoofdstuk op te nemen, terwijl het nummer van de paragraaf wordt overgeslagen.

Verder is het mogelijk om in plaats van bij elke nieuwe laag te hernummeren een bepaald structuurelement door te nummeren, zoals dat bij artikelen vaak het geval is.

G.2. Bijwerken van de verwijzingen

In een METAlex document is elke verwijzing voorzien van een markering, en van metadata die aangeeft waar de betreffende verwijzing naar verwijst. Op het eerste gezicht lijkt het bijwerken van de verwijzingen dus geen ingewikkelde taak. Voor elke verwijzing kan worden gekeken of datgene waarnaar verwezen wordt nog hetzelfde nummer heeft; zo niet, dan kan de verwijzing worden vervangen.

[26] Een onderdeel dat al juist genummerd is, behoudt natuurlijk zijn nummering.

Dit werkt goed voor enkelvoudige verwijzingen. Een enkelvoudige verwijzing is een stuk tekst dat slechts naar één doel verwijst, zoals de tekst "*artikel 2.13, eerste lid, onder i*". In METAlex is deze tekst als geheel gemarkeerd:

```
<cite>artikel 2.13, eerste lid, onder i</cite>
```

Stel nu dat het eerste lid het tweede lid is geworden, dan kan simpelweg de hele tekst worden vervangen door een nieuwe tekst. Omdat de individuele nummers niet zijn gemarkeerd, is het moeilijk om alleen het nummer te vervangen. Een getal kan tenslotte meerdere keren voorkomen, dus het is niet vanzelfsprekend welk getal vervangen moet worden, zoals in:

```
<cite>artikel 1, lid 1 onder b</cite>
```

Een nadeel hiervan is dat indien de gebruiker een tekst had gebruikt die niet dezelfde is als degene die door de applicatie wordt toegepast, de originele tekst wordt overschreven. De gebruiker zou bijvoorbeeld "sub" kunnen gebruiken om onderdelen aan te duiden; als de applicatie vervolgens "onderdeel" gebruikt, dan is na hernummering niet alleen het nummer binnen de verwijzing aangepast, maar ook de naamgeving.
(Dit probleem is op zich te vermijden door de gebruiker toe te staan de door de applicatie gebruikte termen te wijzigen).

Een probleem ontstaat bij meervoudige verwijzingen. Dit is een stuk tekst waarin meerdere verwijzingen staan, die niet volledig van elkaar te scheiden zijn. Een voorbeeld hiervan is "*artikel 15, eerste lid, onderdelen a en b*". In MetaLex wordt deze zin gemarkeerd als

```
<citegroup>
   artikel 15, eerste lid, onderdelen <cite>a</cite> en <cite>b</cite>
</citegroup>
```

Hieraan wordt als metadata toegevoegd dat de eerste cite verwijst naar artikel 15, eerste lid, onderdeel a, terwijl de tweede verwijst naar artikel 15, tweede lid, onderdeel b. Indien onderdeel b is hernummerd naar c, is het niet mogelijk simpelweg de tekst van het laatste deel te vervangen, aangezien een deel van de tekst zich buiten het element bevindt. Ook is het niet mogelijk aan te nemen dat het laatste stuk alleen het nummer van het onderdeel bevat; vergelijk bijvoorbeeld

```
<citegroup>
   artikel 15, <cite>eerste lid, onderdeel a</cite> en
   <cite>derde lid, onderdeel c</cite>
</citegroup>
```

Het is altijd het grootste deel dat geen overlap meer bevat dat in een apart element wordt geplaatst; dit kan meer omvatten dan alleen het nummer.

Een oplossing voor dit probleem zou er uit kunnen bestaan door extra markeringen toe te voegen voor de diverse tekstuele delen van de verwijzende tekst. Dit is echter niet gewenst, omdat deze markering alleen voor deze applicatie (en zelfs alleen voor deze toepassing) betekenis hebben.

Een andere oplossing is om de het hele citegroup element als geheel te vervangen. Dit vereist een complexere methode voor het genereren van verwijzende teksten. Alle

verwijzingen die hun oorsprong hebben in de tekst moeten worden samen genomen (ook als ze niet gewijzigd zijn) en er moet een tekst voor het geheel worden gegenereerd.

Deze functionaliteit is niet het prototype opgenomen

Bijlage H – Inkomende verwijzingen

Om de effecten van een wijziging op andere regelingen dan de te wijzigen regeling te beoordelen, is het nodig om te weten welke andere regelingen afhangen van de te wijzigen regeling. Dit zijn alle regelingen die verwijzen naar de te wijzigen regeling. Voor de gebruiker is het daarom handig om te weten welke regelingen verwijzen naar de te wijzigen regeling, zodat er een duidelijk overzicht is van welke andere regelingen bekeken moeten worden op eventuele ongewenste bijeffecten.

Als een regeling in METAlex is opgeslagen, dan zijn alle verwijzingen gemarkeerd. Het document bevat dus geen overzicht van de inkomende verwijzingen. Echter, het corpus als geheel bevat die informatie wel[27]. Immers, alle verwijzingen zijn gemarkeerd in de verwijzende documenten. Een lijst met inkomende verwijzingen is dus eenvoudig af te leiden door alle documenten af te zoeken naar verwijzingen naar het te verwijzen document. Het toevoegen van deze informatie aan de editor vereist dus alleen deze zoekfunctionaliteit en een methode van weergave.

Maar: het telkens doorzoeken van het gehele corpus op inkomende verwijzingen is erg inefficiënt. Het is daarom wenselijker te informatie over verwijzingen op te slaan in een aparte, goed geïndexeerde bron. Deze dient dat bijgewerkt te worden bij elke wijziging van een bestand in het corpus.

[27] Althans, voor die documenten die ook daadwerkelijk in het corpus zijn opgenomen.