MetaVex: Regulation Drafting Meets the Semantic Web

Saskia van de Ven, Rinke Hoekstra, Radboud Winkels, Emile de Maat, and Ádám Kollár

Leibniz Center for Law, Faculty of Law, University of Amsterdam
PO Box 1030, 1000 BA, Amsterdam
{s.vandeven,hoekstra,winkels,e.demaat,a.i.kollar}@uva.nl

Abstract. Currently almost all legislative bodies throughout Europe use general purpose word-processing software for the drafting of legal documents. These regular word processors do not provide specific support for legislative drafters and parliamentarians to facilitate the legislative process. Furthermore, they do not natively support metadata on regulations. This paper describes how the MetaLex regulation-drafting environment (MetaVex) aims to meet such requirements.

Keywords: XML, RDF(S), OWL, metadata, regulation, drafting, semantic web.

1 Introduction

Legislative drafting and designing amendments to existing or new legislation are important parts of the work done by national parliaments, regional assemblies, city councils and ministries in Europe. Currently almost all of these legislative bodies use general purpose word-processing software to create legal documents.

However, these regular word processors generally do not provide users with targeted support to facilitate the legislative process. They are often badly integrated with legacy systems that support storage, search and publishing facilities, and provide no streamlined environment for drafting and discussing legislation and other kinds of regulations. Such an environment would integrate workflow, search facilities, tools to support group dynamics (including versioning and distribution) and features that facilitate publication. It should provide access to other legal sources through intra- or Internet for direct referencing (see below), but also for background information. A legal drafter changing a particular law might for instance be interested in certain cases or commentaries that point out weaknesses in the current version. These cases and commentaries will be published and maintained by other organisations than the one that employs the legal drafter. In other words, the drafting environment should be able to cope with distributed sources.

Legislative drafting is a complex process that takes place in a political and dynamic environment, which involves many stake-holders. Since a new or adapted regulation is often connected to existing laws, the drafters and other stake-holders should be aware of relationships between the law under construction and those existing legal sources. Legal drafting practice has learned that legal quality can benefit from the use of specific legal drafting patterns.

P. Casanovas et al. (Eds.): Computable Models of the Law, LNAI 4884, pp. 42–55, 2008.

The SEAL project (Smart Environment for Assisting the drafting and debating of Legislation)[1] develops a supportive environment that enables easy construction of legal drafts using drafting patterns and creation of connections from and to existing legal sources. The infrastructure will provide access to a repository with existing laws, draft versions and amendments and will offer easy to use access methods. Collaboration between stake-holders will be supported by groupware facilities such as automated signalling functions and routing of drafts and amendments.

This environment will be developed for three European parliaments. An initial working environment is foreseen in the end of 2007. This will be tested, refined and implemented in co-operation with the parliaments and legislation drafters during the project.

The MetaLex regulation-drafting environment (MetaVex) is developed at the Leibniz Center for Law and is one of the three environments being evaluated in SEAL. The other two environments are the xmLegesEditor: owned/provided and maintained by CNR-ITTIG [1] and The Norma Editor: owned under licence and maintained by CIRSFID University of Bologna [8]. In the following sections we identify the requirements, introduce the XML document standard underlying the system, and describe its current status.

2 Requirements

MetaVex aims to streamline the legislative process by addressing the problems discussed in the introduction. In this section we describe the requirements against which the environment is evaluated. These criteria can be summarized as follows:

Look and Feel. The editing environment should provide a look and feel similar to normal word processors. Document editing should be done in a WYSIWYG[2] interface; this way legal drafters can create document structure and content without knowledge of specific commands or technical notations.

Drafting Patterns. Legislative drafters should be supported by the editor in complying to prescribed legal drafting patterns. Offering users suggestions and predefined phrases in the form of templates improves and speeds up the process of generating document structure and content.

Referencing. The use of references to other legal sources is an important way in which drafters add structure and meaning to a document. The editor should facilitate the frequent use of these references and offer ways to validate the legal sources they cite. References should be *detailed*, i.e. point to the smallest relevant element of a regulation.

Metadata. A way to store extra information about a document e.g. author, version, modification, should be provided. Possibilities to add information concerning document structure as well as content is regarded as an advantage.

Version Management. The environment should offer support to manage document versions, starting from the first draft until and beyond the time at which the document is published. This allows users to always be able to identify the latest version of a document.

[1] SEAL is a project in the e-Participation initiative of the European Commission.
[2] What You See Is What You Get

Groupware. By using groupware facilities, drafters can collaborate on the same project. These facilities will not only consist of sharing comments or amending existing legislation, but will allow for elaborate authorisation and accountability management.

Workflow. Workflow support should be an integral part of the environment to be able to divide tasks into sub-tasks, assigning them to people and keeping track of progress.

Storage. Users should be able to store documents in a *local data repository*, providing them with advanced search mechanisms. It should also be possible to connect to a server with various types of clients over the *internet*, e.g. by using a browser.

Publishing. The environment should allow straightforward publishing of texts in legacy formats. This allows publishing of legal drafts in an early stage, which makes it possible to interact with the public (businesses, citizens and interest groups) during the drafting process.

3 Syntax and Semantics: MetaLex

MetaVex is a regulation-drafting environment for MetaLex documents: texts are saved as XML documents that comply with the MetaLex format for legal sources. This standard provides a generic and easily extensible framework for the XML encoding of the structure and content of legal documents. It addresses many of the requirements introduced in the previous section, as is described in e.g. [3]. In this section the advantages of MetaLex will be explained. Section 4 will address how users of MetaVex can benefit from these advantages, unless stated otherwise.

MetaLex is currently undergoing a CEN standardisation process. It is input to the CEN workshop on an Open XML interchange format for legal and legislative resources[3]. The MetaLex/CEN schema is based on best practices from amongst others the previous versions of the MetaLex schema, the Akoma Ntoso schema [11], and the Norme in Rete[4] DTD. A first version of this schema was adopted as part of a CEN workshop agreement on 6 December 2006[5].

The use of a standard interchange format enables *public administrations* to link legal information from various levels of authority and different countries and languages. Moreover, the standard will enable *companies* that are active in the field of legal knowledge systems to connect to and use legal content in their applications, which allows them to support a much larger market. An open interchange format will also protect customers of such companies from vendor lock in. Finally, the standard will help to improve transparency and accessibility of legal content for both citizens and businesses.

MetaLex provides extensive mechanisms to add metadata both to specific parts of a document and to the document as a whole. Every element of a legal text can be uniquely identified through a URI, and annotated with information regarding e.g. its version, publication date, validity interval, efficacy, language, jurisdiction, and authority. Furthermore, the standard introduces the possibility for marking references,

[3] http://www.cenorm.be/cenorm/businessdomains/businessdomains/isss/activity/ws/_metalex.asp
[4] http://www.nir.it
[5] http://www.metalex.eu/wiki/

both to elements of (other) regulations and to individual entities not part of a regulation, such as institutions or concepts defined by the regulation.

As the standard is primarily intended as an interchange format, annotating legal texts with metadata allows a single MetaLex document to contain several versions of a text. MetaLex not only includes an event-based model for managing multiple versions of legal documents *through time* [4], but multiple *language* versions of the same text can be included in just one MetaLex document as well.

All metadata statements in MetaLex conform to the triple model of RDF[6]. This means that any MetaLex metadata can be used to generate an RDF triple: statements about entities are interpreted as subject, predicate, object triples. And conversely, because every MetaLex element has a unique identifier, it is possible to make external statements in RDF referring to any element of a legal text.

MetaLex provides a strong connection to other semantic web standards as well, such as RDF Schema and OWL[7] as both have RDF/XML syntax. A MetaLex XML document can be translated into OWL by means of XSL transformations (XSLT's). An XSLT provides a mapping between an XML source document and a desired destination format. For instance, we can translate any MetaLex XML document into HTML, XSL:FO and RDF/OWL using such stylesheets. Consider the following piece of MetaLex XML which denotes an article:

```
<Article id="a1">
  <IndexDesignation>
       <Category>
          <TextVersion xml:lang="en">Article</TextVersion>
       </Category>
       <Index>
             <TextVersion xml:lang="en">1</TextVersion>
       </Index>
  </IndexDesignation>
</Article>
```

Using the standard metalex2owl.xsl transformation, we can produce the following RDF/XML code:

```
<metalexrdf:Article rdf:about="http://www.metalex.nl/ec/2002/58#a1">
  <metalexrdf:properStructuralSuccessor
  rdf:resource="http://www.metalex.nl/ec/2002/58#a2"/>
       <metalexrdf:properStructuralMember>
              <metalexrdf:IndexDesignation>
                     …
              </metalexrdf:IndexDesignation>
       </metalexrdf:properStructuralMember>
</metalexrdf:Article>
```

As RDF is order-independent this kind of transformation would contain the risk of messing up the original document order. Document order is important when legislative documents are concerned, therefore the XSLT should address the order of all document elements explicitly. To achieve this goal the successor for each element in the original document is identified and passed through to the destination document in RDF. As a result the MetaLex RDF encoding will contain explicit *sequences* to

[6] The Resource Description Framework. See http://www.w3.org/RDF/
[7] The Web Ontology Language. See http://www.w3.org/2004/OWL

represent the sequences of articles, parts, sentences etc., preserving the order of the document elements present in the original XML file. The container membership property "structuralMember" and the sequence property "structuralSuccessor" are used to represent these kind of sequences. More information about transforming MetaLex XML into RDF/OWL and issues regarding this subject can be found on the MetaLex website[8].

By integrating MetaLex with the semantic web standards RDF, RDFS and OWL, metadata both on the elements of legal texts themselves, as on the *contents* of those texts can be described. OWL and RDFS can be used to describe the contents of legal texts: the *concepts* that occur in them, but also their *normative* content. These formal representations of the semantics of legal texts can be used to perform elaborate legal reasoning, such as consistency checking, legal assessment etc. and for building knowledge-based applications which can be used by citizens to gain advice on complex legal issues. OWL provides additional expressive power, which can be used to describe not only the *content* or *domain* of a regulation, but also the *authority* through which a regulation is enforced, and the history and background of *modifications* of the regulation, as is described in [12]. The MetaLex CEN schema defines a general framework for describing events and actions in OWL. More information about this framework can be found at the MetaLex CEN Wiki[9]. At this moment MetaVex does not support the use of the mentioned semantic web standards yet, as is discussed in section 5.

The MetaLex CEN workshop furthermore adopted the RDFa[10] standard for embedded metadata. RDFa does not have its own namespace: the significance of XML elements and attributes to RDFa processors is determined entirely by names. An RDFa element is defined as any XML element that contains one or more RDFa attributes: about, property, rel, href, instanceof or content. The following example show an article in MetaLex XML:

```
<Article id="a1">
  <IndexDesignation>
       <Category>
              <TextVersion xml:lang="en">Article</TextVersion>
       </Category>
       <Index>
              <TextVersion xml:lang="en">1</TextVersion>
       </Index>
  </IndexDesignation>
</Article>
```

This article could be augmented with RDFa in the following way:

```
<metalex:Article id="a1" about="http://www.metalex.nl/ec/2002/58#a1">
instanceof="metalexrdf:Article"
property="metalexrdf:properStructuralMember"
       <IndexDesignation instanceof="metalexrdf:IndexDesignation"
       property="metalexrdf:properStructuralMember">
              <Category instanceof="metalexrdf:Category"
              property="metalexrdf:textversion">
                     <metalex:TextVersion
                     instanceof="metalexrdf:TextVersion"
                     xml:lang="en">Article
                     </metalex:TextVersion>
```

[8] http://www.metalex.eu/information/guidelines
[9] http://www.metalex.eu/wiki/
[10] http://www.w3.org/2006/07/SWD/RDFa/

```
                </metalex:Category>
                <metalex:Index instanceof="metalexrdf:Index"
                property="metalexrdf:textversion">
                        <metalex:TextVersion
                        instanceof="metalexrdf:TextVersion"
                        xml:lang="en">1
                        </metalex:TextVersion>
                </metalex:Index>
        </metalex:IndexDesignation>
</metalex:Article>
```

An RDFa processor can be used to generate RDF triples from the RDFa elements. The real power of RDFa is that it enables you to add semantic values to XHTML documents. This starts with adding some simple statements, but can develop to using full RDF power inside of an XHTML document. At this moment the use of RDFa is not yet beneficial when used within MetaVex, as is mentioned in section 5.

More importantly, MetaLex allows formal representations of legislation to refer to and be grounded in the documents containing the official texts. An example is LKIF, the Legal Knowledge Interchange Format [2], currently being developed in the ESTRELLA project[11], a vendor neutral representation format for legal knowledge. Existing Semantic Web initiatives are aimed at modelling concepts (OWL "ontology") and rules (RuleML, RIF). The LKIF builds on but goes beyond this generic work to allow further kinds of legal knowledge to be modelled, including: meta-level rules for reasoning about rule priorities and exceptions, legal arguments, cases and case factors, values and principles, and legal procedures. It is based on a layered approach, providing a method of using OWL and RIF and contains two sublanguages, for defeasible rules and for subjunctive betterness. Furthermore, the LKIF is grounded in a core ontology of basic legal concepts: LKIF Core [6]. The ontology covers a base level of components required for explaining epistemological, situational, and mereological patterns as they occur in legal reasoning.[12]

An example of the flexibility of the MetaLex schema is the combination of regulations, maps and spatial planning adopted in the Legal Atlas tool [13][13]. The MetaLex region attribute can be used to refer to the geographical region to which rules in a regulation can be applied: i.e. it specifies geographical jurisdiction. In Legal Atlas, this is used in combination with RDF and GML[14] to show spatial planning regulations both as maps and as texts.

4 MetaVex

MetaVex is a platform independent open source editor, and shares a large part of its codebase with the Visual Editor for XML (Vex)[15]. It is developed within the Java Eclipse[16] development platform, which allows future development of plug-ins and add-on functionality.

---

[11] ESTRELLA: European project for Standardized Transparent Representations in order to Extend Legal Accessibility, IST-2004-027655, http://www.estrellaproject.org

[12] http://www.estrellaproject.org/lkif-core

[13] http://www.leibnizcenter.org/projects/current/legal-atlas

[14] Geography Markup Language.

[15] Vex is currently no longer under active development. See http://vex.sourceforge.net/

[16] http://www.eclipse.org

MetaVex is specifically intended to support the creation of documents complying with the MetaLex standard for legal sources, but flexible enough to be easily adapted to different XML schemas.

Target users of MetaVex will be drafters and members of parliament. These users cannot be expected to be familiar with editing an XML-structure directly. For this reason, the editor offers a WYSIWYG interface, which does not require any knowledge of or experience in creating XML-code. In fact, the editor shows close resemblance to a conventional word processor, and at the same time allows a user to alter or create content while keeping the integrity of the underlying structure intact.

Since XML documents themselves do not carry information about how to display the document MetaVex uses CSS[17] to determine formatting. A user will be able to choose from different types of predefined formatting, but cannot change the formatting in line.

The use of XML as an underlying document structure makes it easy to validate the structure of documents created with MetaVex against the rules defined in the MetaLex schema file. This schema file restricts element and attribute names and allowed combinations. MetaVex uses this schema file to check which elements can be inserted at a certain position in a document, while sustaining a well-formed and valid XML structure.

To enforce this structure during the composing or editing of a document, the editor provides a context sensitive menu of the elements valid at a particular position within the document. Users can only insert elements available in this menu, or elements to which the content model of the current element is agnostic. This procedure ensures schema compliance at every stage of document creation using the editor.

This functionality is one of the major differences between MetaVex and normal word processors. When using a normal word processor, a user can just start typing and does not have to bother about adding specific text elements. The use of templates strongly reduces this difference by offering users a way to add new elements or whole blocks of elements at once: creating e.g. an article is similar to form-filling. MetaVex contains a pane offering the user specific templates to choose from. Furthermore the new document wizard offers a user the possibility to start a new document, based on a predefined template. This way the user does not have to start from scratch.

In the Netherlands, legislative documents are required to be composed according to what is prescribed in the Dutch Guidelines for Legal Drafting [5]. These guidelines do not only apply to technical aspects of writing legislation, but emphasise content too. MetaVex provides a set of templates that are structured according to these guidelines. These templates can provide not only structure, but standard content as well. Currently, the templates included in MetaVex follow Dutch guidelines and cannot be used to support drafting in other countries. However, other templates can be easily imported and used in MetaVex as well. This extensive use of templates not only offers guidance, but can also save users a lot of work.

The MetaVex user interface (see Figure 1) offers an “Insert Templates” panel that shows a list of the mentioned templates. Users can choose and click on one of the templates to insert a prebuilt collection of elements into the document. These elements together form for example a whole chapter or article. Not all templates are

[17] Cascading Style Sheets, see http://www.w3.org/Style/CSS

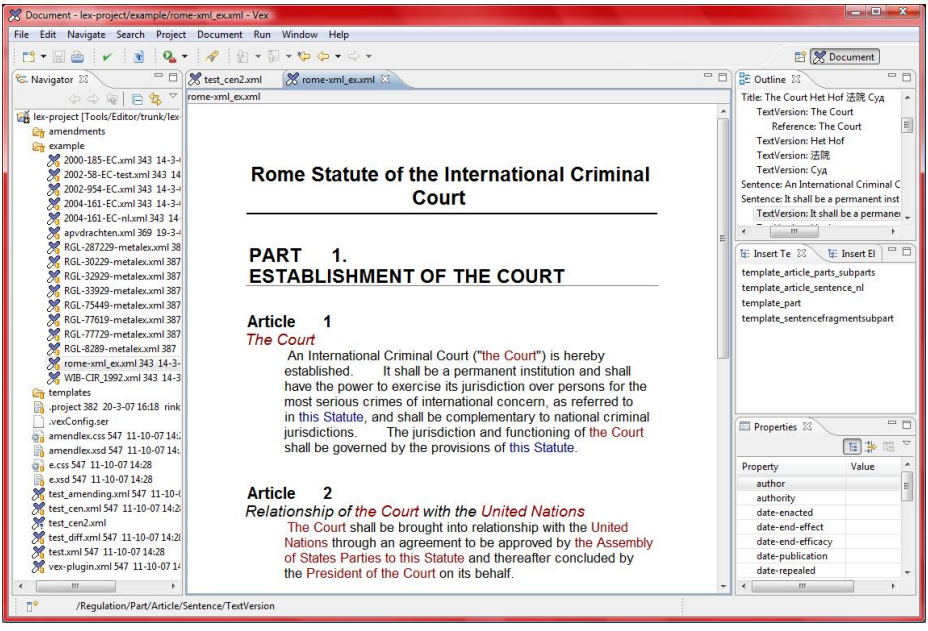

Fig. 1. Editing the statute of Rome in MetaVex

shown in the list, only the ones that can be inserted at the current cursor position, maintaining a valid and well-formed XML-structure underneath.

There is also an “Insert Elements” panel that shows a list of all single elements that can be validly inserted at the current cursor position. The same list is accessible through a context sensitive right-mouse menu. The left part of the screenshot in Figure 1 shows an “Outline” pane that displays the overall document structure as a hierarchy. This structure can be collapsed or expanded and allows a user to easily navigate through parts of the text. Conversely, the cursor position within the XML structure is reflected both in the selected element in this outline pane and in an XPath expression in the status bar. Furthermore, as mentioned in the previous section, MetaLex supports an extensive set of meta-data attributes which allow users to link many different kind s of extra information to a document. The user interface of MetaVex allows the user to edit the values of these attributes through the “Properties” panel. This panel displays a table of all attributes and their values available on the currently selected XML element. Finally, the “Navigator” panel shows a list of the files available in the current project. Each of these panels can be moved, closed or enlarged to suit a users' preference.

Most prominent in MetaVex is the editing panel. Multiple versions of the same text can be simultaneously edited in a single MetaLex document. The screenshot in Figure 2 illustrates this, showing the Statute of Rome (which introduces the International Criminal Court) in Dutch, English, Chinese and Russian language versions. As this can be confusing to users, MetaVex can hide irrelevant information: users can select a desired time interval or language version and hide other versions available in the

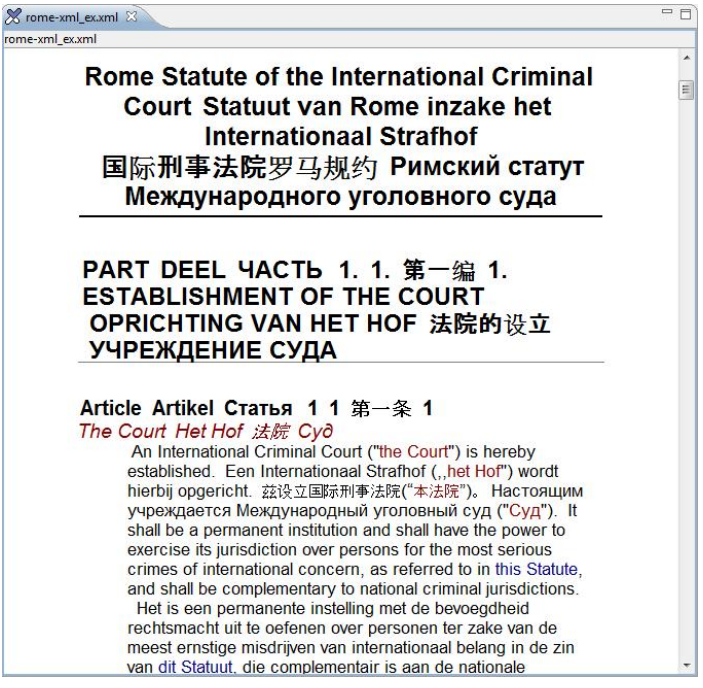

Fig. 2. Editing multiple language versions of the Rome Statute in MetaVex

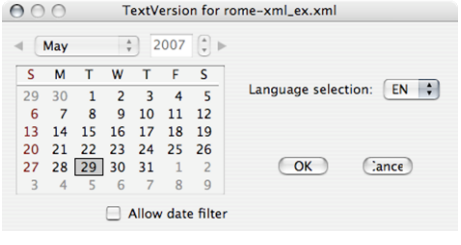

Fig. 3. Selecting a text version

document (see Figure 3). After the selection of for example the English text version, only the English version of the statute of Rome will be shown, as was shown in Figure 1. The selection of text versions is also used by the export functions of MetaVex. Any valid document can be exported to various common formats such as PDF and HTML using export wizards, which apply XSLT[18] transformations to the XML source to produce the desired output format.

Recently functionality is added to MetaVex to support the creation of amendments and the generation of a consolidated version from the original document and the accepted amendment. After a piece of legislation is drafted, there is the possibility to bring in proposals or amendments. Normally such an amendment is created separately

[18] eXtensible Stylesheet Language Transformations, http://www.w3.org/Style/XSL/

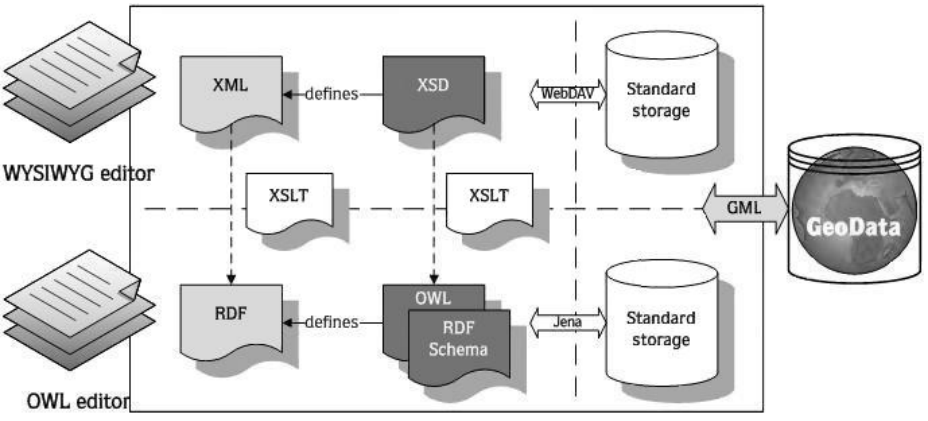

Fig. 4. MetaVex Architecture

from the original document that is being amended. The actual amendment consists of a list of proposed changes that will be applied to the draft bill once the amendment is accepted. Using the editor, amendments can be created just by adjusting the original text and the amendment will be generated automatically. At this moment this amendment support is not fully operational yet, but will be in the future. The next step will be to provide a function to automatically generate consolidated versions of legislation including the modifications resulting from the amending provisions.

Although MetaVex is an editor to alter or produce legislative documents, this does not mean that MetaVex is limited to cope with documents being addressed as "legal" only. In general, for all documents containing rules for a certain domain, a structured way of creating and filing those documents can certainly be useful. MetaVex can be used to suit the needs of many kinds of domains by providing structuring or formatting. For example, companies can develop their own general representation of data in the form of an XML schema file, according to which the intended documents will be composed and validated.

Figure 4 shows the MetaVex architecture and the way in which the various components will be integrated. In short, an XML document is constructed against an XML schema, and can be translated to RDF/OWL using XSLT transformations. MetaLex XML documents are stored in an XML storage facility (to be developed by one of the partners in the SEAL consortium) through a WebDAV interface. The RDF/OWL representation of the MetaLex document uses the vocabulary specified in the MetaLex OWL schema. It is stored using a Jena database backend. Currently, not all of the components shown in the picture have been developed. Future development issues are described in the discussion section, but first this section will end with providing some examples of the useful extras RDF offers in the context of MetaVex.

As mentioned before, links can be made from every element in a MetaLex document to a concept or identity in RDF. This is especially useful because of the frequent use of two kinds of references in legislative documents. First there are citations pointing to other structural elements of a document or another document as a whole, e.g.

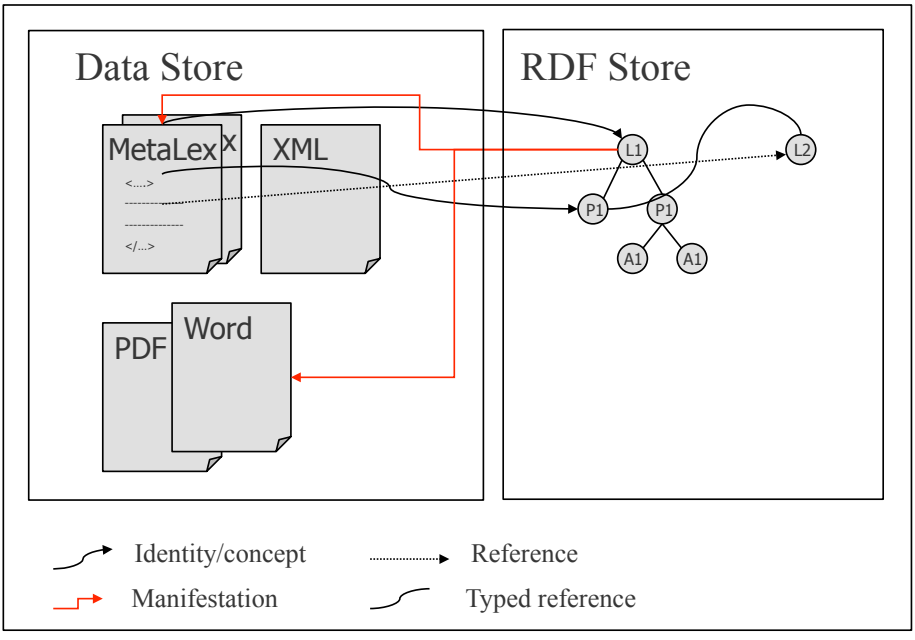

Fig. 5. From data to models

"Chapter 1, Article 2, second sentence" or "The Dutch traffic law". Second there are references to a concept, like 'civil servant' or 'boat'. When trying to validate the latter kind of reference an RDF model proves to be very useful. RDF can for example provide the intermediary concept 'boat' to temporarily resolve such a reference.

All documents containing information about the concept 'boat', can be linked to the concept 'boat' as well. Although resolving references might seem the most difficult, citations can be difficult to resolve as well. When a document contains a citation to a specific law and a new version of this law becomes enacted, the citation might need to be adjusted as well. Furthermore if a cited piece of legislation is not enacted yet, there is no actual document to link to. The RDF model can in this case contain a general concept of the cited law, containing all the versions that exist. These versions can be represented as subconcepts of the general one. This way it is possible to cite a document and all existing or even future versions. Not only different versions can be linked, different manifestations can be added as well, as is shown by the arrows in Figure 5 pointing from the RDF store to the Data store. Because RDF allows for typed references between concepts, relations between the different concepts can be expressed as well.

The advantage of storing such links is obvious: it makes it easier to find the correct version of a document. This is an important advantage for organisations that often have to deal with multiple versions of legislation, such as the Dutch Tax and Customs Administration, as is described in [12]. The new MetaLex/Cen standard will use a same sort of mechanism for storing various versions of a single document.

The use of RDF can also be advantageous in the context of search mechanisms and cataloguing. Ordinary search engines and catalogue systems make use of textual

occurrences of a keyword. But when RDF comes in place, it is possible to search for all relevant pieces of legislation that are applicable on a certain subject, just following all of the provided links from the RDF concept to the XML sources and potential other sources. Linking to current handbooks, guides and regulations on legislative drafting can be considered as well.

5 Discussion and Future Work

As mentioned in the introduction, MetaVex is one of three environments being evaluated in SEAL. The other two environments are the xmLegesEditor and The Norma editor. The Norma editor and the xmLegesEditor are developed in Italy, in the context of the Norme-in-rete project. The Norma editor is designed as an add-on to Microsoft Word. This gives it the advantage that users can create and edit documents in a familiar environment. A disadvantage, however, is that it means that the Norma editor is not yet fully open source. This will change in the future, as the makers of the Norma editor aim to switch from MS Word to Open Office. Another disadvantage is that being dependable on another product in general has some risks. Whenever a new version of the other product is released, the editor has to adapt to this new version, which can be a time consuming process. The Norma editor already offers advanced support to automatic consolidation of documents and to the creation and validation of references. Users do not edit the XML structure directly, but the conversion to XML takes place after the editing has been done. A validation tool is used to parse the document structure and generate a valid and well-formed XML document. The user is notified when the validation tool detects inconsistencies.

In contrast, the xmLegesEditor is a native XML editor. It is rule-driven: it only allows operations that are valid with respect to the underlying document structure. The xmLegesEditor already contains functionality to extract and validate normative references and a parser to read and convert document into XML. It directly produces XML documents, compliant to the NIR DTD, but in the future it will also support XML schema.

Similar to the xmLegesEditor, the MetaVex editor is open source, rule-driven, and a native XML editor. Besides the functionality it already offers, there are quite some items on the requirement list that are currently not supported and will be discussed in the following part of this section.

MetaVex should enable intuitive construction and maintenance of references in legal documents, by functions for adding, editing and validating. [7] and [9] describe ways to automatically detect references in legislation with high accuracy. Embedding such functionality into MetaVex certainly would be a useful extension.

The use of MetaLex means a strong focus on semantic web technologies such as RDF(S) and OWL. For now it suffices to support these formats and edit OWL documents in a separate, already existing OWL-editor, such as Protégé[19] or TopBraid[20]. In the future, MetaVex will offer means to view and edit OWL-documents using an OWL-editor embedded inside MetaVex. The idea is to develop a separate view showing the RDF triples corresponding to a selected MetaLex element. The view should

[19] http://protege.stanford.edu
[20] http://www.topbraidcomposer.com

also enable users to edit the RDF triples directly. Admittedly, concurrent editing of the RDF and XML version requires relatively complex synchronisation.

As mentioned in Section 3, the MetaLex CEN workshop has adopted the RDFa standard for embedding metadata in MetaLex XML. Once MetaLex CEN is finalised, the MetaVex editor should offer functionality that supports the annotation of XML documents with the RDFa attribute set. Of course, this is closely related to a possibility for browsing an RDF graph. A recent development is GRDDL[21], a W3C recommendation, which specifies an even more flexible method for embedding RDF in arbitrary XML. GRDDL can be used to specify 1) per document, an XSL stylesheet for automatic extraction of RDF triples, or 2) a namespace-related stylesheet at XML Schema level. The advantage of GRDDL is that is more general, i.e. it is suited for any XML, and not just XHTML, and allows more flexible embedding of metadata, not restricted to the RDFa attribute set.

The MetaVex architecture (Figure 4) shows a connection to geodata. At this moment, MetaVex does not provide this connection yet, but it will provide one in the future. The connection will be similar to the approach of Legal Atlas.

So far nothing has been mentioned yet about MetaVex' storage mechanism and how certain features like versioning, security, groupware facilities etc. will be integrated. Although an implicit way of maintaining version information using MetaLex' attributes already exists, the ability to cope with several versions of a document in an explicit matter should also be taken into account. A content management system satisfying the requirements of section 2, will be developed within SEAL. However, MetaVex will commit to standards-based interfaces to open source RDF repositories such as Sesame[22] and Jena/Joseki[23].

MetaVex is still under construction and there is a lot of work that needs to be done. Nevertheless a solid, easy extendable and highly adaptable solution for editing XML-structured documents in a user-friendly environment already exists. As soon as all proposed features are fully present, MetaVex will lift the editing of legal documents to a whole new level by its unique combination of syntax and semantics.

References

[1] Agnoloni, T., Francesconi, E., Spinosa, P.: Xmlegeseditor, an OpenSource visual XML editor for supporting Legal National Standards. In: Biagioli, C., Francesconi, E., Sartor, G. (eds.) Proc. of V Legislative XML Workshop, pp. 239–252. European Press Academic Publishing (2007)

[2] Boer, A., Gordon, T., van den Berg, K., Di Bello, M., Förhécz, A., Vas, R.: Specification of the Legal Knowledge Interchange Format (LKIF). Deliverable 1.1, Estrella (2007)

[3] Boer, A., Hoekstra, R., Winkels, R., van Engers, T., Willaert, F.: METALex: Legislation in XML. In: Bench-Capon, T., Daskalopulu, A., Winkels, R. (eds.) Legal Knowledge and Information Systems, Jurix 2002, pp. 1–10. IOS Press, Amsterdam (2002)

[21] Gleaning Resource Descriptions from Dialects of Languages, see http://www.w3.org/TR/grddl/

[22] http://www.openrdf.org

[23] http://jena.sourceforge.net

[4] Boer, A., Winkels, R., van Engers, T., de Maat, E.: A Content Management System based on an Event-based Model of Version Management Information in Legislation. In: Gordon, T. (ed.) Legal Knowledge and Information Systems, Jurix 2004, pp. 19–28. IOS Press, Amsterdam (2004)

[5] Aanwijzingen voor de regelgeving (Directives for regulations), regulations for legislative drafting issued by the Prime-Minister, November 26, Stcrt (1992)

[6] Breuker, J., Hoekstra, R., Boer, A., van den Berg, K., Rubino, R., Sartor, G., Palmirani, M., Wyner, A., Bench-Capon, T.: OWL ontology of basic legal concepts (LKIF-Core). Deliverable 1.4, Estrella (2007)

[7] de Maat, E., Winkels, R., van Engers, T.: Automated detection of reference structures in law. In: van Engers, T. (ed.) Legal Knowledge and Information Systems. Jurix 2006: The Nineteenth Annual Conference. Frontiers in Artificial Intelligence and Applications, vol. 152, pp. 41–50. IOS Press, Amsterdam (2006)

[8] Palmirani, M., Brighi, R.: An XML Editor for Legal Information Management. In: Traunmüller, R. (ed.) EGOV 2003. LNCS, vol. 2739, pp. 421–429. Springer, Heidelberg (2003)

[9] Palmirani, M., Brighi, R., Massini, M.: Automated Extraction of Normative References in Legal Texts. In: Sartor, G. (ed.) Proceedings of the 9th International Conference on Artificial Intelligence and Law (ICAIL 2003), pp. 105–106. ACM Press, New York (2003)

[10] Shadbolt, N., Berners-Lee, T., Hall, W.: The Semantic Web Revisited. IEEE Intelligent Systems 21(3), 96–101 (2006)

[11] Vitali, F., Zeni, F.: Towards a country-independent data format: the Akoma Ntoso experience. In: Biagioli, C., Francesconi, E., Sartor, G. (eds.) Proc. of V Legislative XML Workshop, pp. 239–252. European Press Academic Publishing (2007)

[12] Winkels, R., Boer, A., de Maat, E., van Engers, T., Breebaart, M., Melger, H.: Constructing a semantic network for legal content. In: Gardner, A. (ed.) Proceedings of the Tenth International Conference on Artificial Intelligence and Law (ICAIL), June 2005, pp. 125–140. IAAIL, ACM Press, New York (2005)

[13] Winkels, R., Boer, A., Hupkes, E.: Legal Atlas: Access to Legal Sources through Maps. In: Winkels, R. (ed.) Proceedings of the 11th International Conference on Artificial Intelligence and Law (ICAIL 2007), pp. 27–36. ACM Press, New York (2007)