NATIONAL AND KAPODISTRIAN UNIVERSITY OF ATHENS

SCHOOL OF SCIENCE
DEPARTMENT OF INFORMATICS & TELECOMMUNICATIONS

GRADUATE THESIS

Nomothesi@, Greek Legislation API

Ilias I. Chalkidis

Supervisor: Dr. Manolis Koubarakis
Phd Candidate Charalampos Nikolaou

ATHENS
May 5, 2014

ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Νομοθεσί@, Διεπαφή προγραμματισμού εφαρμογών (API) για την Ελληνική νομοθεσία

Ηλίας Ι. Χαλκίδης

Επιβλέποντες: Μανώλης Κουμπαράκης, Καθηγητής
Χαράλαμπος Νικολάου, Διδακτορικός Φοιτητής

ΑΘΗΝΑ
May 5, 2014

GRADUATE THESIS

Nomothesi@, Greek Legislation API

Ilias I. Chalkidis

Supervisors: Dr. Manolis Koubarakis
Phd Candidate Charalampos Nikolaou

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Νομοθεσί@, Διεπαφή προγραμματισμού εφαρμογών (API) για την Ελληνική νομοθεσία

Ηλίας Ι. Χαλκίδης

Επιβλέποντες: Μανώλης Κουμπαράκης, Καθηγητής
Χαράλαμπος Νικολάου, Διδακτορικός Φοιτητής

Abstract

This paper introduces Nomothesi@, an API to provide access to Greek legal documents, by means of a legal XML syntax and Linked Data. Nomothesi@ proposes a XML standard for Greek legal documents and adopts CEN Metalex OWL ontology to describe the legal relationships and events. On these fundamentals, we build a Content Management System (CMS) for searching and browsing legal documents.

SUBJECT AREA: Semantic Web (AI), Software Engineering
KEYWORDS: Legislative Knowledge Representation, Open Data, RDF/OWL Metadata, XML, Content Management System (CMS), Universal Resource Identifiers (URI)

Περίληψη

Η συγκεκριμένη εργασία παρουσιάζει μια Διεπαφή προγραμματισμού εφαρμογών (API), που ονομάζεται Νομοθεσί@ και σκοπό έχει να δώσει πρόβαση στην ελληνική νομοθεσία, με την χρήση ενός νομικού XML προτύπου και τα Διασυνδεδεμένα Δεδομένα (Linked Data). Το Νομοθεσί@ προτείνει ένα νεό XML πρότυπο για την αναπαράσταση τον νομοθετικών εγγράφων και ταυτόχρονα υιοθετεί την OWL οντολογία CEN Metalex με σκοπό να περιγράψει τις νομικές σχέσεις και γεγονότα. Πάνω σε αυτές τις αρχές, χτίζουμε ένα Σύστημα Διαχείρισης Περιεχομένου (ΣΔΠ, Content Management System, CMS) με σκοπό την αναζήτηση και παρουσίαση της Ελληνικής νομοθεσίας.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: Σημασιολογικός Ιστός, Τεχνητή Νουμόσυνη, Τεχνολογία Λογισμικού
ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: Αναπαράσταση Γνώσης νομοθεσίας, Ανοιχτά Δεδομένα, RDF/OWL Μεταδεδομένα, XML, Σύστημα Διαχείρισης Περιεχομένου (ΣΔΠ, Content Management System, CMS), Καθολικά Αναγνωριστικά Πηγής (URIs)

Acknowledgements

First of all I would like to acknowledge my supervisor Dr. Manolis Koubarakis. His courses AI I and II had directly contributed in discovering my personal interest in AI and more specifically assisted me in selecting this topic. I would also like to acknowledge PhD candidate Charalampos Nikolaou, for his valuable assistance and guidance as a second supervisor in the whole process of this thesis.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In the last decade , many countries have directly provided to the public web access to legislation. Various legislative Content Management Systems (CMS) provide online access to legislation via thematic search, verbal or chronological using different types of data models. These systems also aim to provide services and data to third party developers.

Content and information of legal documents are drafted and archived on Information Systems, where these documents are accessible and retrievable. Web standards such as XML and RDF facilitate the separation between content based information and ontological information to help best presentation and utilization of existing information presented and / or drawn from documents therefor. [1]

The usability / quality of these systems is determined on the basis of opportunities related to the search of legal documents and their partitioning (eg articles, paragraphs etc) based on criteria (thematic, verbal, chronological), on the representation of chronological versions (eg enacted, current, 3rd revised), on the publication of metadata and generally capabilities of reprocessing by third party CS engineers. The quality of these capabilities is immediate consequence of the adoption of the new web technologies and more specifically the semantic web ones, along with the revised XML ones. [1]

For the development of such legislative Open Data systems, international practice is concentrated in the following standards:

- Create an XML schema to represent the content of legal documents.
- Create an ontology mainly for the representation of legislative relations(modification, citation etc.) between the legal documents and legislative events (publication, amendment, withdrawal etc.). Crucial entity metadata, also.
- Create a permanent system of Universal Resource Identifiers (persistence URIs).

Creating an XML schema for the representation of legal documents for a country or an international organization (eg EU) requires both subtractive approach, to ensure the usability and scalability of the figure, and also the full cover of specificities on each legislative system (eg Greek Legislation). In this context, international practice has not implemented a widespread standard, despite very promising developing standards like CEN Metalex [2][3], but mainly develops standards "from scratch" according to the specific legislation. In contrast to the last decade, they have been presented several ontologies for legislative representation, in which the proposal of CEN Metalex, seems to be adopted more consequently, mainly regarding the core ontology. The starkest example is the United Kingdom(legislation.gov.uk API) [4]. Finally, the creation of stable Universal Resource Identifiers (URIS) concerns a standard, which is fully adopted by all projects involving legal knowledge internationally, and has generally been researched as well as given as a directive from international programs

dealing with eGovernment as ISA[5][6].

All these fundamental standards are going to be unified in an EU level, with the first step towards this approach being the European Legislative Identifier (ELI) standard.[7]

The objective of this thesis is, based on international practice, to design such a CMS for the Greek legislation and to implement it in demo version, hoping that it would set a foundation for the discussion of the representation of legal knowledge and its integration in the area of Open Data in Greece. The project aims to enhance transparency and the right for information through the idea of Open Data. The need of transparency becomes clearer when we look at who the users of such systems are. People using public legislation web services are not just lawyers, but a much wider group of people who need to be informed, to cite, or use legislation as part of their job e.g. Civil servants, freelancers, citizens defending their rights. They are the people who need to understand the law, and they tradionaly used to search web services like Google. All these people can directly benefit from such systems. Last but not least, our research wants to challenge the whole legislative drafting process, in order to lead in new better services. As has rightly been said: "We shape our tools and then our tools shape us", Martin McLuhan.

Chapter 2

Theoretical Basis

2.1 From Government Gazette to legal documents

According to Article 7 of Law 3861/2010 (Government Gazette 112/13.7.2010, Issue A) for the Diavgei@ program about transparency through mandatory disclosure of government's decisinstitutionions and acts, the Government Gazette (FEK) is now available to citizens free of charge from the website of the National Printing House for reading, storing and printing. The Article 3 of the same Act refers to the publication on web, for all legislation documents(such as laws, decrees) as separate entities from individual institutions, something that currently happens occasionally and in any case not through / via an organized Information System. The result of this inconsistency is that still the issues of Government Gazette are be considered by the public as the key elements of the legislative process instead of the actual legal documents. One of the goals of this thesis is to create a Content Management System (CMS) that will put in general force the Article 3 of Law 3861/2010 on "Publication's Liabilities on the Web".

2.2 Encoding of legislation

An important part of this work is to understand the encoding of Greek legislation, which means knowing the components of a legal document, the connection between those components, their syntax, their value, any other crucial information. The encoding of Greek legislation follows the rules set out in "Manual for encoding of the legislation" [8], which was adopted from the Central Committee of Encoding standards as required by Law 3133/2003. As we observed, consistent encoding is implemented after 2003 and still the whole persistency of the draft process is been challenged. The appropriate structure throughout the legal text has the following parts:

Articles

The basic divisions in the text of legal document are the articles, which are numbered with Arabic numeral(1, 2, 3 ...) or, in the case of insertion of a new article in an existing legal document, by combining Greek and Arabic numeral chapter letter (A, B, Γ...). Adding titles in the articles is useful because it helps in the systematic classification of the substance of the legal document. The numbering of subdivisions of the content of the article should be uniform in all the articles of the legal document.

Paragraphs

Paragraphs should be numbered with Arabic numerals (1, 2, 3 ...). If an article has a single paragraph, this single paragraph is not numbered, which means that number 1 is not placed.

Cases
The cases should be numbered with small Greek letters (α, β, γ ..), the subindents double small Greek letters ($\alpha\alpha$, $\beta\beta$ ...).

Passages
A passage is defined as the verbal period between two dots. The passages are numbered and set out in writing contiguous, ie without break new line in text.

Books - Chapters - Sections - Parts
The legal document may also be subdivided according to their size at larger units, such as books, chapters or sections, which are numbered with Greek uppercase letter characters. The "books", "sections" and "parts" are numbered verbally and written in capital letters (eg BOOK ONE). The chapters are numbered with capital letters of Greek alphabet (eg A, B, Γ, etc.). In each case precedes the numbered subset of the code and follow the number (eg BOOK ONE , PART ONE , SECTION ONE , CHAPTER A).

From the above it can be easily concluded that the main division of legal documents is the article. Articles comprise of sections. Paragraphs may enumerate cases. Paragraphs consist of passages, namely recommendations. In some cases it is appropriate to construct larger divisions of legal text therefore used subdivisions book, section, chapter, part. This is likely to happen with legislative works such as the "Civil Law", the "Criminal Law" etc.

Important elements of the structure of the legislation is also the following :

- Any legal document is characterized by the following: Type, Year, Number (ID), Title. The first three of them form a unique persistent key to each legal document.
- The above data follows the statement of the ministry, which relates with the legal document (eg Ministry of Finance).
- Below that, there are citations to provisions of prior legal documents, European directives, even thoughts - assessments. This introduction begins with the standard phrase "Having considered the following:" and then it numbers citations with Arabic numerals (1, 2, 3 ...).
- At the end of the legislative document are the signatures of the competent Ministers of the Greek Government and the President of Greek Republic. These are considered as the legislators.

2.3 Legislative Modifications

It is common international practice, the amendment of divisions of legal document by later legal documents. This change is not fundamental, but merely modifying and / or removing divisions (such as article, paragraph, sentence). In Greek legislation, there are often modifications through replacements, additions or extensions (add-on phrases) in passages and paragraphs' cases. Therefore a legislative modification has independent significance but also expressed in a certain way (replacement or extension) on the legislative text.

2.4 The passage as the core of legislation

Studying legal documents that have been published in the Government Gazette, we noticed that most modifications are related with passages or cases (eg "At the end of paragraph X add paragraph as follows", "The X passage of paragraph Y of Article Z is replaced as follows"). This directly affects the design of our data models. Looking the legislative document as a tree structure from top to bottom, we understand that our "children" nodes should be at the level of passages and cases as structural entities in our data models. It is necessary to mention that some times modifications are called in the level of words and phrases. This type of lawmaking is close to "non-legitimate", it creates some issues to the whole mechanical process and it needs to be avoided.

Chapter 3

Design

3.1 XML Schema

The XML schema, which is essentially the structure of a new markup language, as mentioned earlier, must meets two criteria. To be as abstract and as extensible as possible and simultaneously to apply in the conditions (rules) of the Greek legislation encoding. Also we have to foresight the case of amendment and to include the appropriate elements and attributes for this requirement. In our CMS each legal document is represented by a separate XML file and without storing any updates. Updates happens mechanically instead of manually. Based on international practices and the specific criteria, mentioned above, we propose the following schema (standard), on Figure 3.1. The meaning of the XML elements used in this schema, are reported on Table 3.1.

Element	Definition
Legislation	defines a legal document
Metadata	defines the metadata of the legal document *(eg title, signer, creation date etc)*
Introduction	defines the introductory part of the legal document, which includes the citations
Citation	defines a citation
Body	defines the main legal text
Article	defines an article
P	defines a paragraph in an article
P2	defines a passage in a paragraph *(Same for Book, Section, Chapter, Part)*
List	defines a group of cases in a paragraph
Case	defines a case in a group of cases
Modification	defines a modification in a paragraph
Table	defines a table in an article
Title	defines the title of an XML element (eg Article)
Text	defines the text in a passage or a citation
Number	defines the serial number of an XML element in a group of XML elements

Table 3.1: XML elements definitions

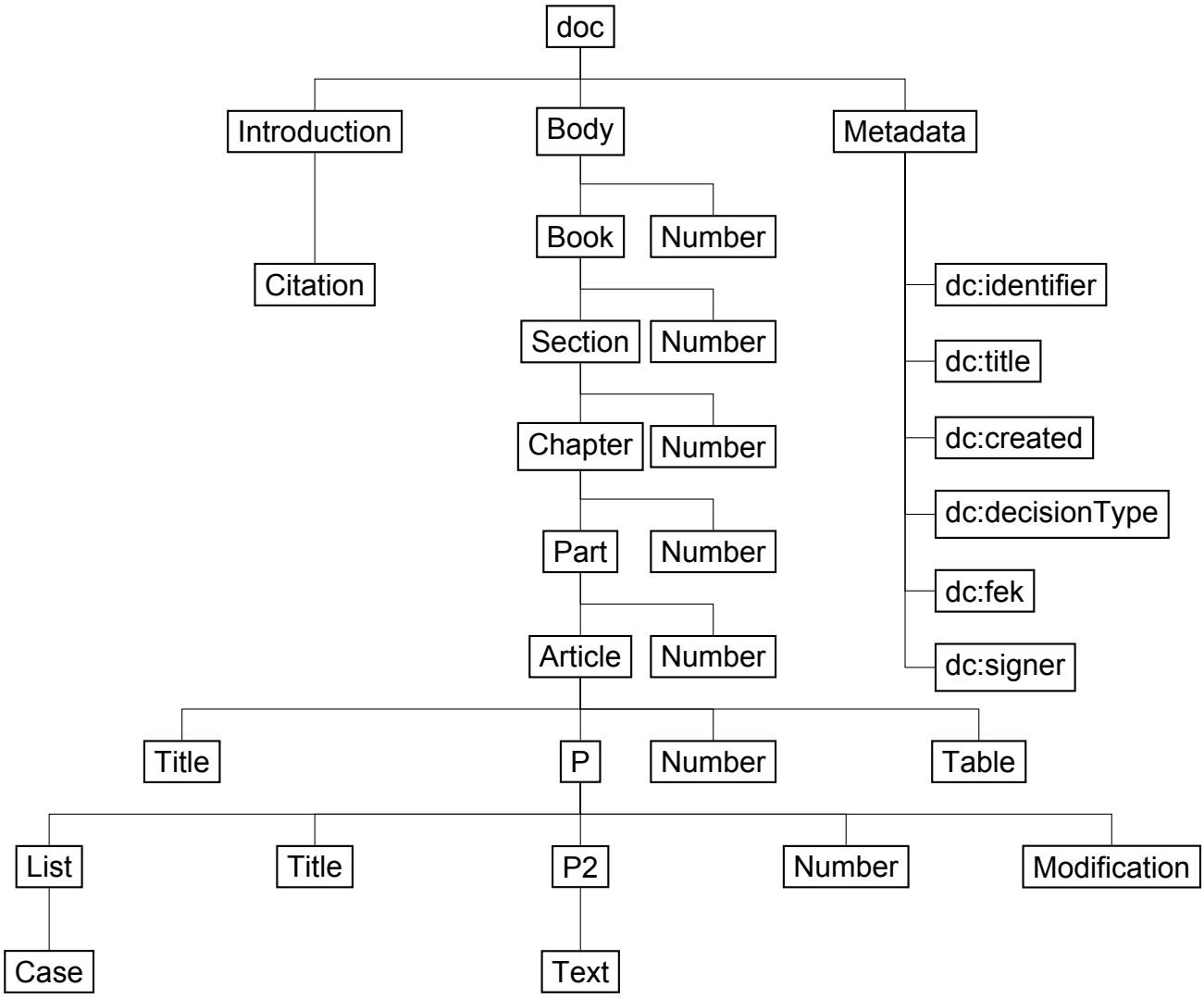

Figure 3.1: Greek Legislation XML schema

3.2 OWL Ontology

3.2.1 Legal Documents - Versions - Files

The use of Semantic Web technologies in the context of this project focuses mainly to representing information on legislative events (publication, modification, repeal), which affect the validity, the content and general the behavior of the legal document (work). For this matter, we adopt the standard CEN Metalex OWL ontology [9] [10], whose core is the most suitable to represent such information. The CEN Metalex OWL ontology uses a model based on legal representation of legislative events involving the legal documents and the features that they entail. The main entities used for describing interactions in this model are:

Bibliographic Work is any legal (bibliographic) document (eg law, presidential decree, etc.), created and published by legislative procedures at a given time.

Bibliographic Expression is every expression (version), which is expressed by modifying the original content of a specific legal (bibliographic) work.

Bibliographic Manifestation is every representation (type) of file (eg PDF, XML, RDF/XML), which represents a different standard to form an legal document.

Legislative Modification defines any modification in content (eg article, passage, case etc.) on a legal document.

Legislative Competence Ground defines legal documents, which contains modifications refering to other legal documents.

In Figure 3.2, we describe the interactions (relationships) between these elements. For each legal document, we have the same cognitive knowledge. Which versions (BibliographicExpression) have occurred, which modifications (LegislativeModification) led on these versions, which legal document (LegislativeCompetenceGround) placed these modifications. This dataset, through queries, will indicate those individuals that are appropriate each time in order to produce suitable results.

We also need to store crucial RDF metadata for each entity. These means chronological data (dates), thematic (title, tags, organizations involved etc) data and many other government information. For this purpose we started to use some DC Terms[1]. In the future we will enrich our metadata using eGorvernment Core Vocabularies[2] (Core Person, Core Business, Core Location), assisted by ISA in the effort to integrate RDF metadata across EU and combat roadblocks to metadata governance and management.

[1] See http://dublincore.org/documents/dcmi-terms/

[2] See https://joinup.ec.europa.eu/community/core_vocabularies/description

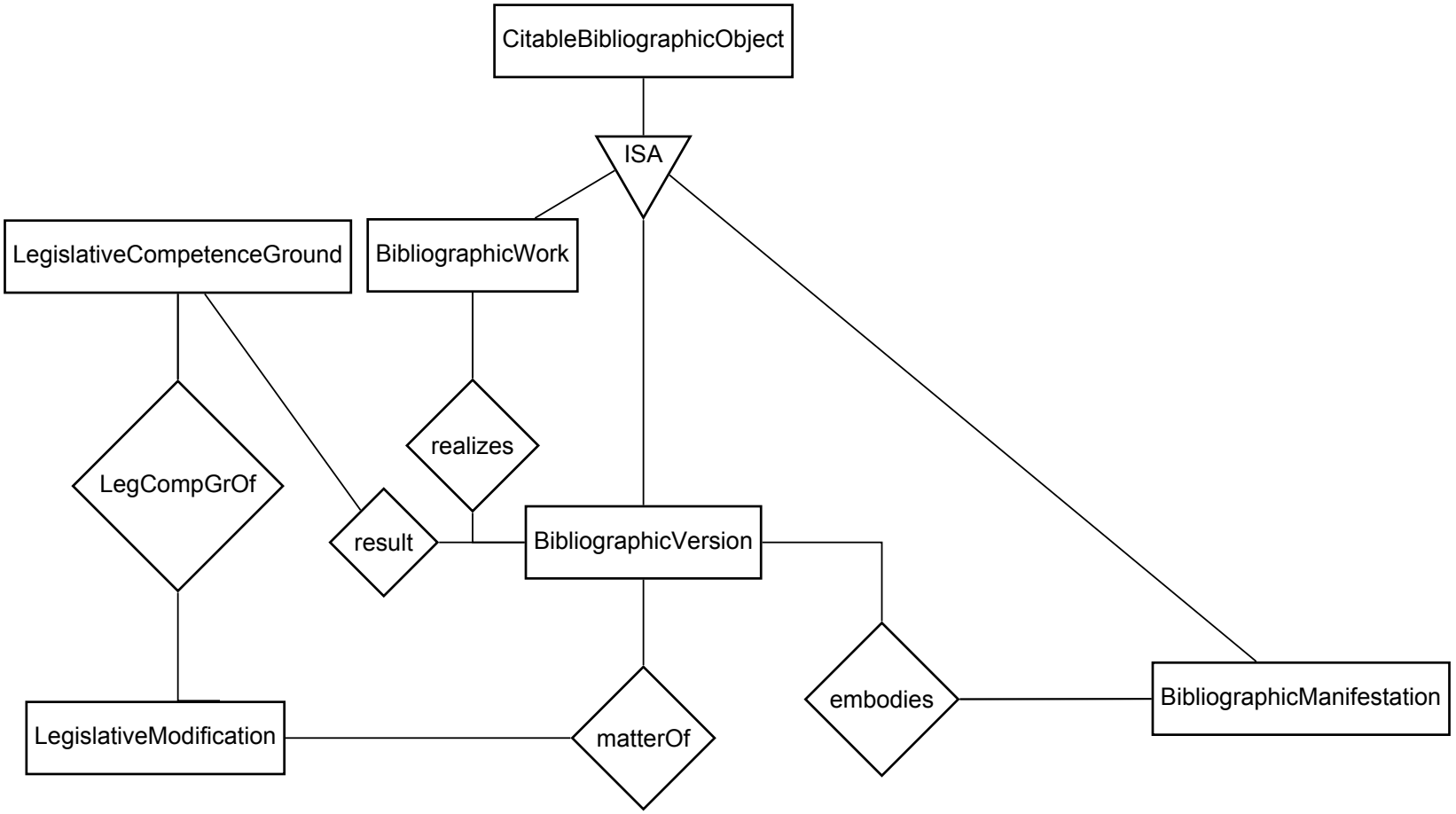

Figure 3.2: CEN Metalex OWL Ontology Core ER[1]

1

[1](Figure 3.1) *LegCompGrOf*: refers to LegislativeCompetenceGroundOf relation.

3.2.2 Legislative timestamps

An important piece of legislation knowledge are the timestamps framing legislative events.[11] Thus we have the following timestamps for each legislative procedure:

date-publication is the time the element is officially published or announced, in the sense that the element becomes an entity at that point in time to which references can be made from outside the containing document.

date-enacted is the time the content becomes applicable in decisionmaking and is always later than or the same time as date-publication

date-modified is the time the content was modified by the content of other legal documents, which were published at a later time.

date-repealed is the time the content becomes inapplicable in decisionmaking.

The recording and appropriate representation of timestamps provide us with the ability to observe the life cycle of legal documents and infer crucial information according to the legislative events (publication, enactment,amendment, repeal) that happened in these specific times. A timeline of the life cycle of legal documents can be seen in Figure 3.3. The inferred information helps us to search and present legal documents based on chronological criteria and compute interesting statistics around the whole legislative process.

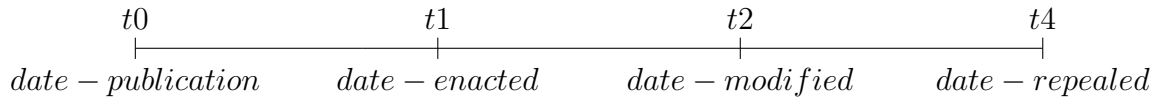

Figure 3.3: Life Cycle of legal documents

3.2.3 Time versions

We have multiple versions (BibliographicExpression) of the same legal documents (BibliographicWork). Any legal document, which contains modifications for a precedent legal document, leads to a new version of this legal document. In technical words, an XML element called Modification encapsulate a newly published regulation refers to another XML element of the same conceptual unit (eg article, paragraph) in a precedent legal document to repeal, enact, or change it. in Three different types of versions could be standardized:

enacted version is the version of the legal document, as it was published in the Government Gazette on date-enacted.

current version is the version of the legal document, as amended and applied, taking into account all the modifications in the legal text, which probably have occurred during the time.

date version is the version of legal document as applied on a specific date after its publication, taking into concideration all the modification, which have probably occurred until the specific date.

3.3 Persistent URIs

The Universal Resource Identifiers (URIs) provide a high level of usability as they recognize the objects in each field, applied to a structural way, and facilitate the stability and recovery of information described.[6] Fixed URIs to divisions of legislation are very important, as they are on legal documents in general. Various initiatives are trying to upgrade reliable classification for the legislation to existing bibliographic scheme. Their aim is to facilitate the process of creating URIs for legal sources, regardless of the availability of a document on the web, location of a document, and the way to access it. Initiatives such as PRESTO in the UK describe a system for legislation and public information in which ``All documents, views and metadata at all significant levels of granularity and composition should be available in the best formats practical from their own permanent hierarchical URIs."[12]. Based on international practice and the particularities of Greek legislation, we proposed a schema of URIs, which is very similar with the UK.

The schema of URIs defines the following pattern:

http://www.nomothesia.gov.gr/legislation/{type of legislation}/{year}/{id}

Correspondingly, we can follow a structural search:

http://www.nomothesia.gov.gr/search?keywords={title of legal document}&type={type of legislation} &year={year}&id={id}&date={YYYY-MM-DD}

As a type of legislation, we mean all different types of Greek legislation, we are using the encoding of Table 3.2

Type of Legislation	Code
Constitution	con
Law	law
Presidential Decree	pd
Cabinet Decision	amc
Legislative Act	la
Ministerial Decision	md
Proclamation	com
Call of Referendum	rd

Table 3.2: Encoding Types of Legislation

We also refer to specific conceptual unit of the legal document, by collocate the branch of the calling division after the ID. So for example if we want to address in the first passage of the third paragraph of the fourth Article in Law 312/2003, the corresponding URI is:

http://www.nomothesia.gov.gr/legislation/law/2003/312/article/4/paragraph/3/passage/1

As we mentioned already, there are three type of versions for legal documents. In order to refer on them, we follow the URI's extensions, in Table 3.3.

Version	URI extention
current)	-
enacted	enacted
date-version	YYYY-MM-DD[1]

Table 3.3: Encoding Versions of Legislation

Eg http://www.nomothesia.gov.gr/id/law/2003/312/2010-12-23

Any legal source has multiple file formats available , both for internal use of the system and also to share open with third party CS engineers. The available file formats are: XML file (.xml), RDF file (.rdf) and PDF file (.pdf). For the disposal of these files we use the following extensions, mentioned on Table 3.4.

File - Manifestation type	URI extension
XML	data.xml
RDF	data.rdf
PDF	data.pdf

Table 3.4: Encoding Types of File

[1] See http://en.wikipedia.org/wiki/ISO_8601

Chapter 4

Implementation

4.1 Apache Cocoon Framework

Searching the best solution to merge our theoretical work (XML schema, OWL ontology, URIs), based on the international experience[13], we conclude we should be build our CMS on Apache Cocoon Framework[1]. *"Apache Cocoon is a Spring-based (since version 2.2 of Cocoon) framework built around the concepts of separation of concerns and component-based development. Cocoon implements these concepts around the notion of component pipelines , each component on the pipeline specializing on a particular operation. This makes it possible to use a Lego(tm)-like approach in building web solutions, hooking together components into pipelines, often without any required programming."* Cocoon relies on the pipeline model: an XML document is pushed through a pipeline, that exists in several transformation steps of your document, as we see in Figure 4.1. Every pipeline begins with a generator, continues with zero or more transformers, and ends with a serializer. The main three components could be described like this:

Generators Generator is the starting point of an xml pipeline. Usually, it generates XML content as SAX[2] events and initializes pipeline processing. XML content can be either retrieved from an XML file or produced from other data models based on request parameters.

Transformers A transformer is the central point in a sitemap pipeline. Within a pipeline match, transformers consume SAX events and emit SAX events. It can be compared to an XSL: it gets an XML document (or SAX events), and generates another XML document (or SAX events).

Serializers A Serializer is responsible for transforming SAX events to a presentation format. We have Serializers for generating HTML, XML, PDF and of course you can create your own.

4.2 XML DataBase Server

In order to organize our XML documents for querying processes, we use a native XML DataBase, BaseX[3]. BaseX is a scalable and high-performance, yet very light-weight XML Database engine and XPath/XQuery Processor, which can run remotely as a Server in a Client/Server architecture, whose client is Cocoon. The XQuery language give as the capabilities to search based on multiple criteria and also our project is supported with full-text search engine of Apache Lucene[4], which is used by BaseX.

[1] See http://cocoon.apache.org

[2] SAX (Simple API for XML) is an event-based sequential access parser API, which provides a mechanism for reading data from an XML document. SAX parsers operate on each piece of the XML document sequentially.

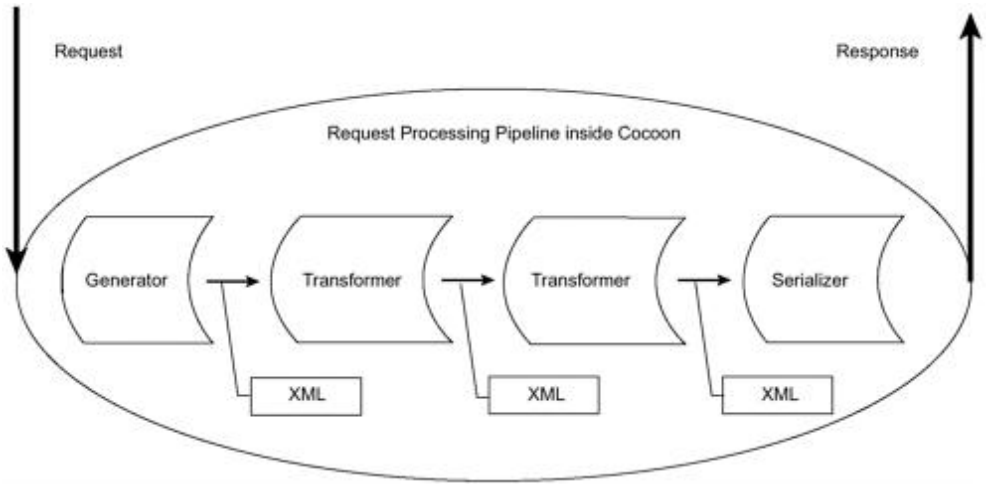

Figure 4.1: Apache Cocoon Pipeline

4.3 Triplestore Server

Our project needs a triplestore to organize RDF/OWL triples. OpenRDF Sesame[5] is a de-facto standard framework for processing RDF data. This includes parsers, storage solutions (RDF databases a.ka. triplestores), reasoning and querying, using the SPARQL query language. It offers a flexible and easy to use Java API that can be connected to all leading RDF storage solutions. So like BaseX, using Sesame remotely as a Server in a Client/Server architecture, whose client through Java API is Cocoon, we can handle CEN Metalex OWL ontology data.

4.4 Architecture of Nomothesi@ API

To recap, we have Apache Cocoon as an web application Server, BaseX as an native XML DB Server and Sesame as an RDFStore Server. So every HTTP request pass from Cocoon, then if needed we settle connection with BaseX and/or Sesame to retrieve appropriate data and turn back to Cocoon to form HTTP response for the user. The abstract architecture is depicted in Figure 4.2.

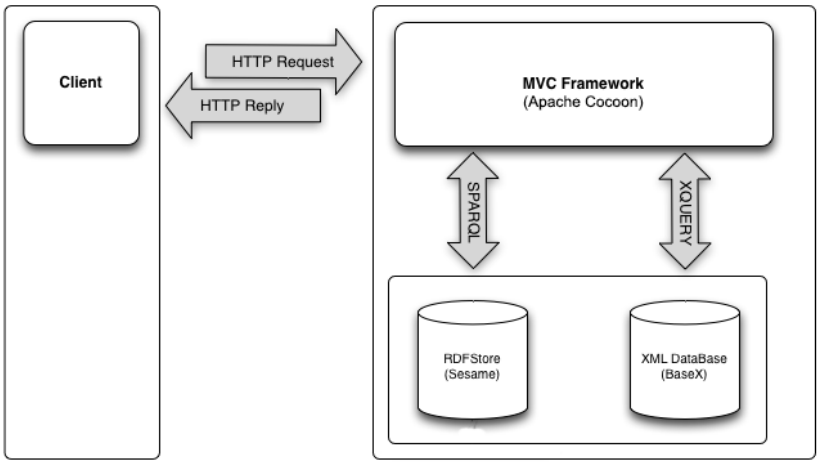

Figure 4.2: Architecture of Nomothesi@ API

4.5 Use Cases

4.5.1 Use Case 1 - Present updated legislative version

Let's say we take the following request: http://www.nomothesia.gr/pd/2011/54, which means to return the last updated version of Presidential Decree (PD) 2011/54. We have the following steps(Figure 4.3):

1. Cocoon match request URI with the appropriate pipeline for PDs.
2. Pipeline calls Version Generator.
3. Version Generator connects with Sesame Server and apply SPARQL query to find any modifications, which have occurred on PD 2011/54. It accepts a list of modifications with 2 main elements: modification's URI and modification's Source File.
4. Version Generator connects with BaseX Server. For each modification in the list, query the modification's Source File and get as a result modification's content (XML branch). Form an XML block of all these XML parts.
5. Pipeline calls XMLGenerator to take the content of XML file, called PD201154.xml .
6. Pipeline calls XSLT Transformer.
7. XSLT Transformer use an XSLT file to apply modifications on XML file's divisions, which have to be modified (replaced, enriched, deleted, added).
8. Pipeline calls XSLT Transformer.
9. XSLT Transformer use another XSLT file to produce HTML page from XML output of the previous XSLT file.
10. Pipeline calls XHTML Serializer to give HTML page as HTTP response.

[3]See http://basex.org
[4]See http://lucene.apache.org
[5]See http://www.openrdf.org

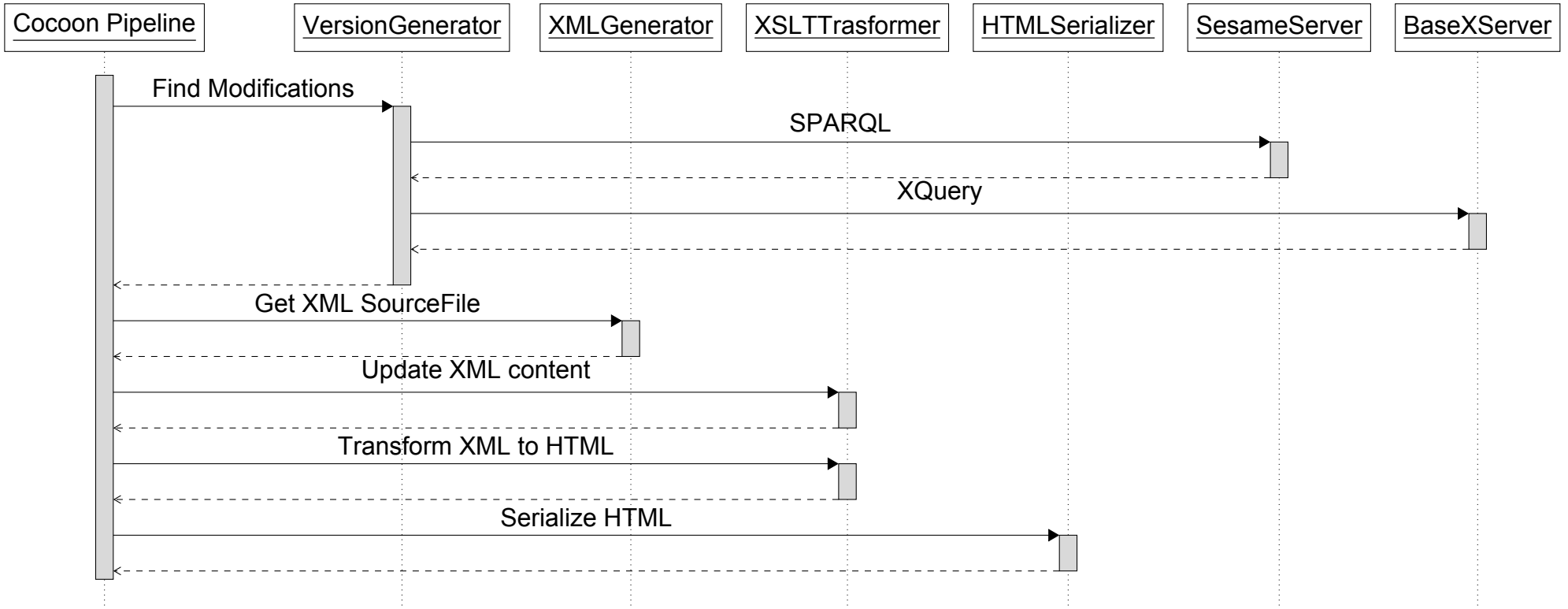

Figure 4.3: UML sequence diagram Use Case 1

4.5.2 Use Case 2 - Search legal documents, based on criteria

We fill search form to find all PDs of 2012. The request URI for this functionality is http://www.nomothesia.gr/search?Type=pd&Year=2013. We have the following steps:

1. Cocoon match request URI with the appropriate pipeline for search function.
2. Pipeline calls Search Generator.
3. Search Generator connects with BaseX to query XML DB with type and year criteria. We find the suitable XML documents and set for each one a record with title, date, typ and URI characteristics. Then we form an XML block with all these records.
4. Pipeline calls XSLT Transformer.
5. XSLT Transformer produces HTML page with a table for all these records.
6. Pipeline calls XHTML serializer to give HTML page as HTTP response.

```
<map:pipeline id="search">
  <map:match pattern="search">
        <map:generate type="searchgen"/>
        <map:transform src="search.xslt"/>
        <map:serialize type="xhtml"/>
  </map:match>
</map:pipeline>
```

$\Downarrow CallSearchGenerator \Downarrow$

```
public class SearchGenerator extends AbstractGenerator{

    static Context context = new Context();
    Request request;

   @Override
   public void setup(SourceResolver resolver, Map objectModel, String src, Parameters params)
   throws ProcessingException, SAXException, IOException {...6 lines }

   public void generate() throws IOException, SAXException, ProcessingException {...191 lines }

   static void xquery(final String query) throws BaseXException, IOException {...3 lines }
```

$\Downarrow Callsearch.xslt \Downarrow$

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="/">
    <html>
    <head></head>
    <body>
        <div id="wrapper">
            .....................................
                <table id="results" >
                    <thead><tr><th>Τίτλος</th><th>Κωδικός</th><th>Ημερομήνια</th></tr></thead>
                    <tbody>
                    <xsl:for-each select="results/result">
                        <tr>
                            <td>
                                <a><xsl:attribute name="href">
                                    <xsl:value-of select="id"/>
                                </xsl:attribute>
                                <xsl:value-of select="title"/>
                                </a>
                            </td>
                            <td>
                                <xsl:value-of select="type"/> 
                                <xsl:value-of select="year"/>/<xsl:value-of select="number"/>
                            </td>
                            <td>
                                <xsl:value-of select="date"/>
                            </td>
                        </tr>
                    </xsl:for-each>
                    </tbody>
                    </table>
        </div>
    </body>
    </html>
  </xsl:template>
```

Figure 4.4: Search Pipeline Transactions

Chapter 5

Presentation (Nomothesi@ Demo)

First we will see Home Page of Nomothesi@ (See Figure 5.1). Home Page provides a search form for legal documents, two tables, one for most requested acts and one for new legislation and an introduction text about our web application.

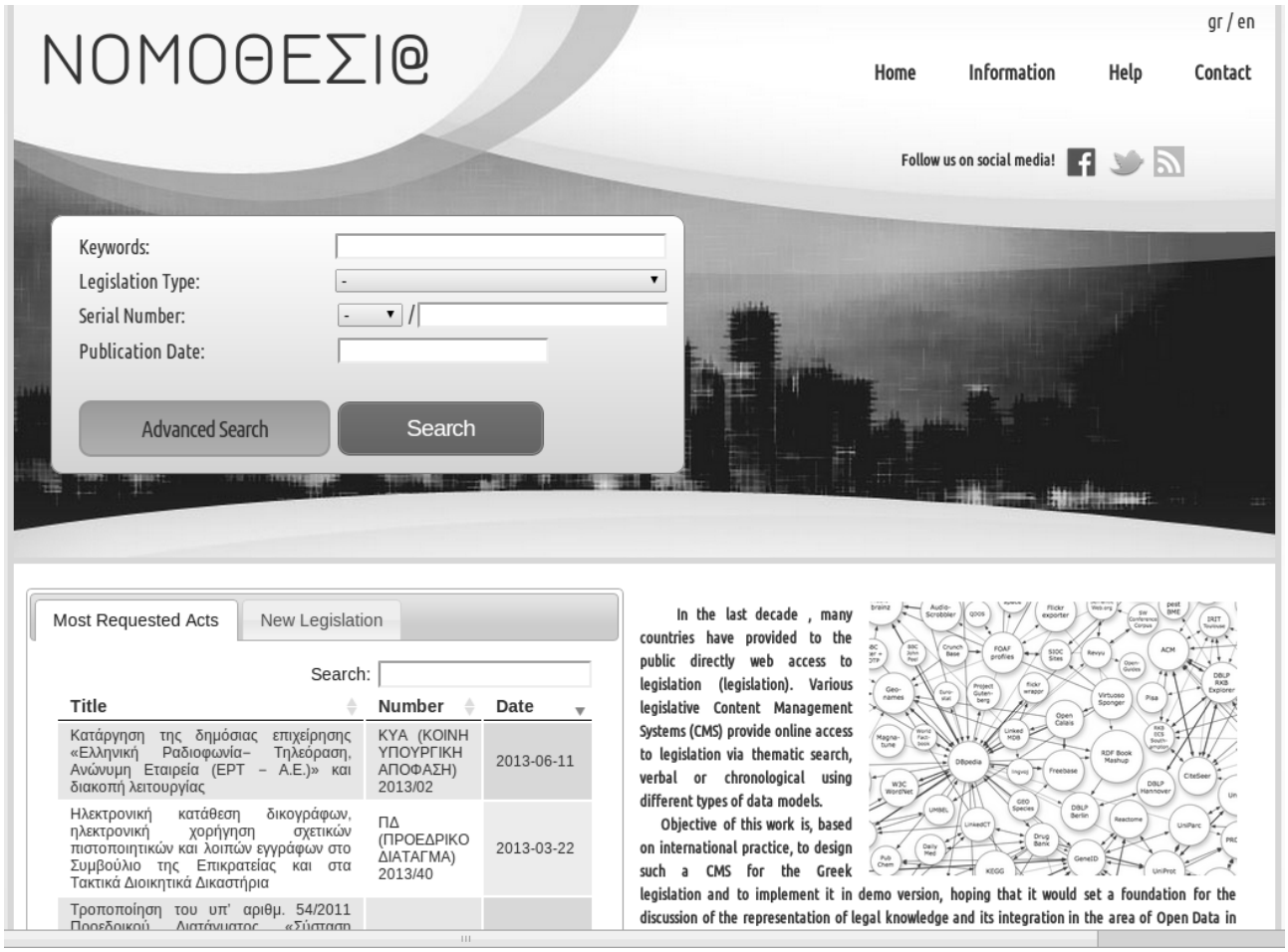

Figure 5.1: Home Page of Nomothesi@

Let's see a page of enacted version about Presidential Decree (PD) 2011/54 (See Figure 5.1). We notice two abstract divisions. The main division provides the legal text, a table of basic information and a toolbar for table of contents, timeline, citations and full-screen mode. The other division - left - is a menu, where we provide last and enacted version of this legal document and three different formats (XML,RDF,PDF).

Figure 5.2: HTML Page of Enacted version of PD 2011/54

Have in mind enacted version of PD 2011/54, we move in latest available version the same legal document (See Figure 5.3). The GUI environment is exactly the same, but we can notice that divisions of the legal document have been modified through legal modifications.

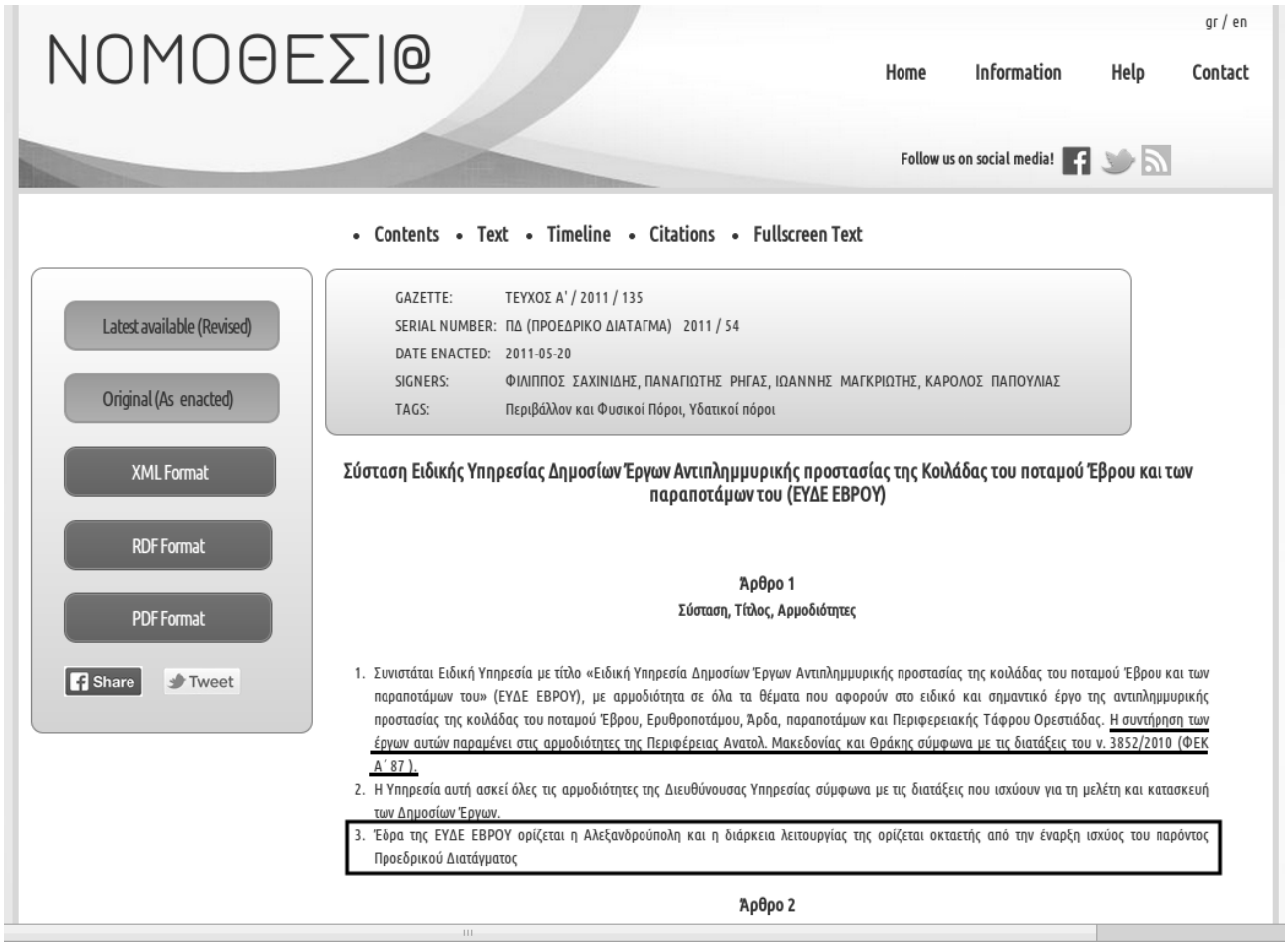

Figure 5.3: HTML Page of Latest version of PD 2011/54

As we mentioned home page includes a search form. We search for any legislation published in 2013. The search results are printed in a table with basic information, from which we can linked to resulted legal documents (See Figure 5.4).

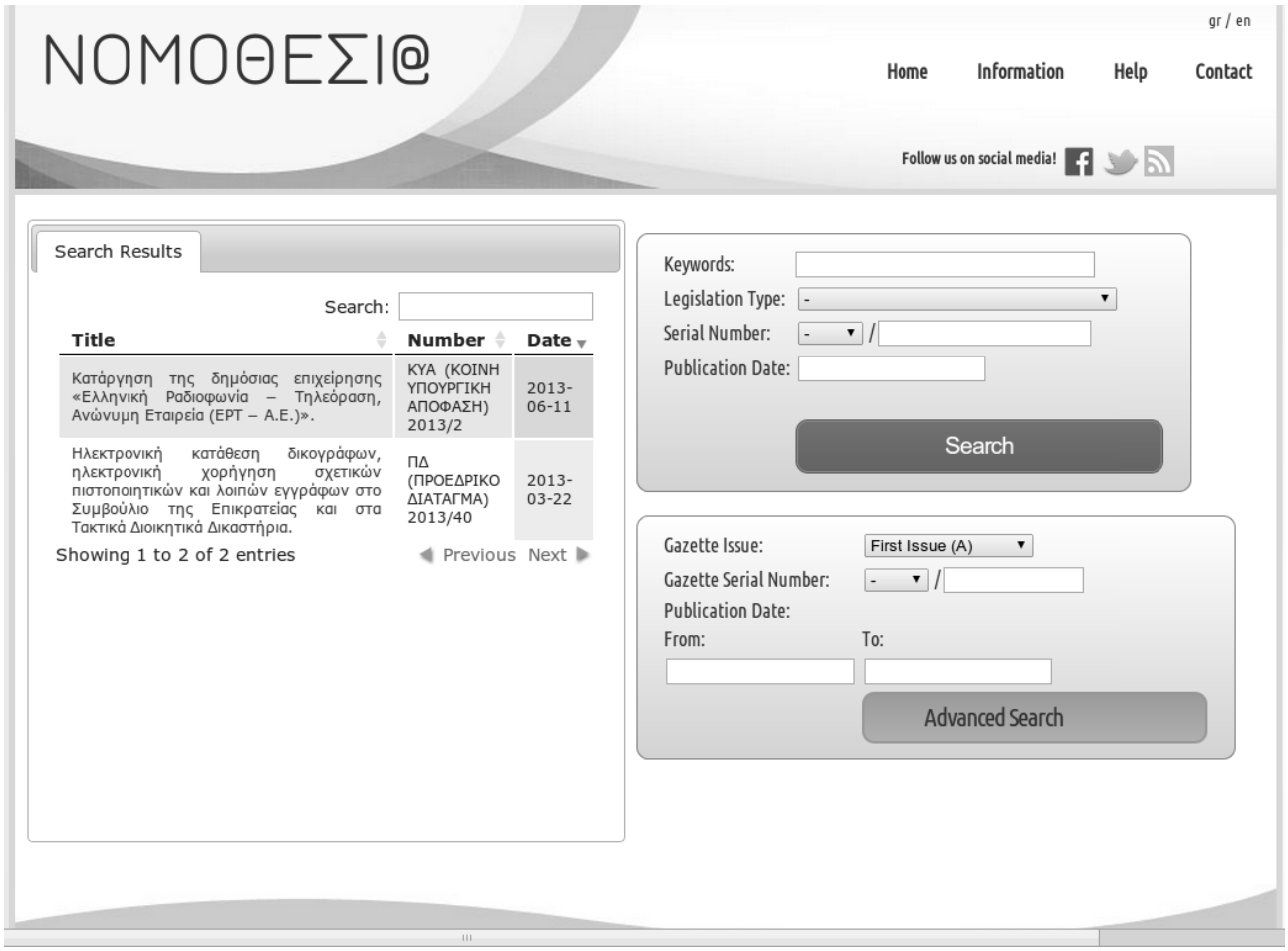

Figure 5.4: Page of Search Results

Chapter 6

Future Work

Although the results presented here have demonstrated the effectiveness of such a model for Greek Legislation, it could be further developed in a number of ways:

Legal Documents Converter

As we mentioned, Greek legal documents are stored as text files (.doc) and published as PDF files (.pdf). In order to apply our API in the real world, we need to enrich our data with as many legal documents as possible. Of course, it cannot happen manually, so we need a tool (converter) to accomplish the hard and lenghtly job. n order to transform legacy XML to Metalex and RDF, Metalex research group implemented the MetaLex converter, an open source Python script. [13] Having in mind that government text documents are not available in a consstent matter (eg pdf, word), we may think of using other third party data sources, that probably are used in private sector.

SPARQL Endpoint

It is in our interest to make our API as useful as possible for third party CS engineers. In this approach the implementation of a SPARQL endpoint is the next big thing. This service will give the opportunity for everyone to query our data sets to draw legislative knowledge in his own special interests. The whole architecture of our system is redefined as shown in Figure 6.1.

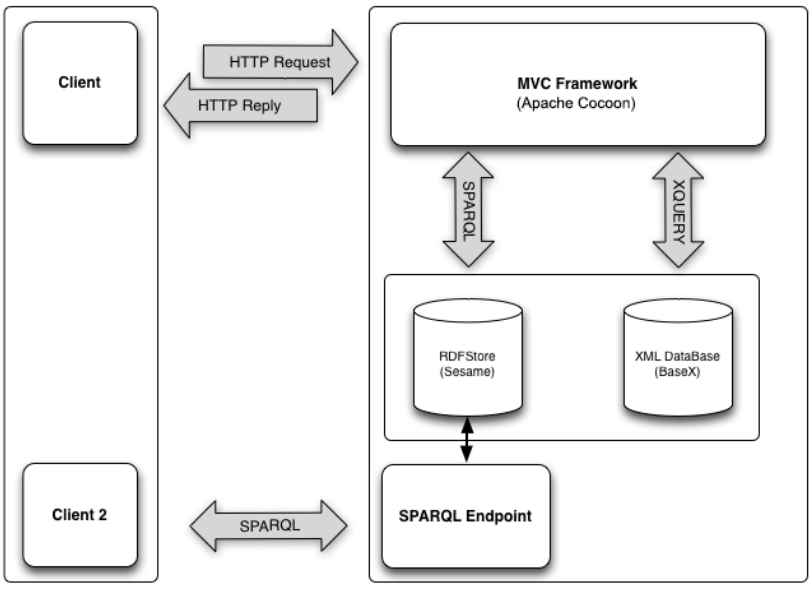

Figure 6.1: Architecture of Nomothesi@ API

Core Vocabularies

The last three years, the Core Vocabularies Working Group of the Interoperability Solutions for European Public Administrations (ISA) Programme has the objective to promote the specification of the Core Person and develop the specification of the Core Business and Core Location vocabularies. These vocabularies will promote interoperability through different data sets, which are published by many organizations across EU. With cautious circumspection in the great issue of interoperability, we will use these vocabularies to redefine and enrich our data sets.

Local Government Decisions

In our project, we worked with the core of Greek Legislation (Laws, Presidential Decrees and Ministerial Decrees). Other secondary legal documents have to be handled in the same way. Something really innovative is to include Local Government Decisions. Greek Local Government consists of 7 decentralized administrations, 13 regions and 325 municipalities. Each of them publish Local decisions in Government Gazette and there is a great interest in all these paralegal documents. Research groups in our department have published an ontology of Kallikratis plan, the administrative system of Greece.[14] We can enrich our metadata with other from Greek GeoSpacial Administrative ontology in order to infer knowledge about Local Government Decisions.

Case Law

Case Law plays an important role in judicial activities. The term "res judicata" describes the application of legal documents in a specific trial, where similar cases in the future should be handled using the same legal documents in the same manner. It would be a great improvement for our project to cite case laws, which are linked on specific legal documents.

Bibliography

[1] Maria Hallo Carrasco, M. Mercedes Martinez-Gonzalez and Pablo de la Fuente Redondo, Data models for version management of legislative documents, Vol.39 Issue 4 Journal of Information Science W.M. Newman and R.F. Sproull, Principles of Interactive Computer Graphics, McGraw-Hill, 1979, p. 402.

[2] A. Boer, R. Hoekstra, R. Winkels, T. van Engers and F. Willaert, METALex: Legislation in XML, in Legal Knowledge and Information Systems (Jurix 2002), 1– 10, (2002), IOS Press.

[3] R. Hoekstra, The MetaLex document server – legal documents as versioned linked data. In: Aroyo L,Welty C, Alani H, Taylor J and Bernstein A (eds) Proceedings of the 10th international conference on the Semantic Web (ISWC'11), Vol. II. Berlin: Springer, pp. 128–143.

[4] John Sheridan, LEGISLATION.GOV.UK, http://blog.law.cornell.edu/voxpop/2010/08/15/legislationgovuk/

[5] How Linked Data is transforming eGoverment, ISA SEMIC, http://ec.europa.eu/isa/actions/documents/isa_1.1-how-linked-data-is-transforming-egovernement_en.pdf

[6] Phil Archer, Stijn Goedertier and Nikolaos Loutas, Study on persistent URIs, with identification of best practices and recommendations on the topic for the MSs and the EC, PwC EU Services, ISA

[7] John Dann, European Legislation Identifier "ELI", SEMIC 2014

[8] Central Codification Committee of Goverment's SG, Manual directives for encoding of legislation

[9] A. Boer, F. Vitali, and E. de Maat. CEN Workshop Agreement on MetaLex XML, an open XML Interchange Format for Legal and Legislative Resources (CWA 15710). Technical report, European Committee for Standardization (CEN), 2006

[10] A.Boer, Legal Theory, Sources of Law and the Semantic Web, 2009. IOS Press.

[11] A. Boer, R. Winkels, T. van Engers, and E. de Maat. Time and versions in METAlex XML. In Proceeding of the Workshop on Legislative XML, Kobaek Strand, 2004.

[12] Rich Jelliffe, PRESTO - A WWW information architecture for legislation and public information systems

[13] A. Boer, R. Winkels, T. van Engers, and E. de Maat. A content management system based on an event- based model of version management information in legislation. In T. Gordon, editor, Legal Knowledge and Information Systems. Jurix 2004: The Seventeenth Annual Conference., Frontiers in Artificial Intelligence and Applications, pages 19–28, Amsterdam, 2004. IOS Press.

[14] Greek Linked Open Data, Greek Administrative Geography, http://www.linkedopendata.gr/dataset/greek-administrative-geography