Legislative document content extraction based on Semantic Web technologies

A use case about processing the History of the Law

Francisco Cifuentes–Silva[1,2] francisco.cifuentes@weso.es and
Jose Emilio Labra Gayo[1] labra@uniovi.es

[1] Department of Computer Science, University of Oviedo, Oviedo, Asturias, Spain
[2] Biblioteca del Congreso Nacional de Chile - BCN, Valparaíso, Chile
https://www.bcn.cl

Abstract. This paper describes the system architecture for generating the History of the Law developed for the Chilean National Library of Congress (BCN). The production system uses Semantic Web technologies, Akoma-Ntoso, and tools that automate the marking of plain text to XML, enriching and linking documents. These documents semantically annotated allow to develop specialized political and legislative services, and to extract knowledge for a Legal Knowledge Base for public use. We show the strategies used for the implementation of the automatic markup tools, as well as describe the knowledge graph generated from semantic documents. Finally, we show the contrast between the time of document processing using semantic technologies versus manual tasks, and the lessons learnt in this process, installing a base for the replication of a technological model that allows the generation of useful services for diverse contexts.

Keywords: Linked Open Data · Legal Information Systems · Legal Domain · Legal Knowledge Base · Automatic Markup · Semantic Web

1 Introduction

The legal and legislative scenario has benefited greatly from the development of technologies that allow automating tedious tasks, such as the allocation of metadata or the marking of documents, which are nevertheless essential tasks for the construction of products that allow consulting, information synthesizing and extracting knowledge that resides in archived repositories. Indeed, the OASIS standard Akoma-Ntoso (AKN) provides electronic representations of parliamentary, normative and judicial documents in XML[3] using semantic markup and annotation of textual documents through Linked Open Data (LOD). The main motivation to employ Semantic Web Technologies in this context is the reuse of data in various products and services, and the implementation of a Legal Knowledge Base (LKB) for public access. In 2011, the BCN started a project

[3] http://docs.oasis-open.org/legaldocml/ns/akn/3.0

to automate the elaboration of the History of the Law, a perfect scenario to use an open interoperability standard, given the public nature of the data. An initial exploration of the use of LOD was described in [8], where we described how it can be used to publish legal norms. In this paper, we present the History of the Law project: the generation process, the key tools and strategies adopted for its elaboration and the results that have been obtained integrating Semantic Web technologies in the production process.

2 History of the Law and Parliamentary Labor Projects

A *History of the Law* (HL) is the collection of all the documents generated during a law's legislative processing; since the initiative that gives life to the bill, until its discussion in the Congress, the reports of the parliamentary committees that studied it and the transcripts of the debates in the sessions rooms, gathering their traceability [9] within the legislative process.

The HL allows someone to collect the so-called *spirit of the Law*, allowing its interpretation in a precise way in relation to the scope and sense that was given to the norm when it was legislated. This legal instrument is particularly useful both for judges when preparing judgements and for lawyers when they use certain rules to support their arguments. Similarly, the *Parliamentary Labor* (PL) is a compilation of all the legislative activity carried out by a parliamentarian during the exercise of his office, such that it has been registered in printed media belonging to the legislative power, such as a parliamentary motion, a session journal or a commission report.

In Chile until 2011, both products were made by legal analysts, only for specific requests, and by processing manually each document related to a Law.

For the electronic and automated elaboration of both documentary collections, it is required to have a granular database, which registers all the documents of the legislative process where any reference to bills or parliamentarians is made, allowing later to extract and recover selectively, what was discussed around a bill that will become law, as well as what a certain legislator has said in any context. For this reason, AKN has been used for the construction of this LKB, since it allows the addition of semantic marks on the text, which in turn allow the precise identification of the location of parliamentary interventions, the presence of debates around a bill, the processing phases, and many other types of metadata.

2.1 Production Workflow

To carry out document processing, business processes have been designed that were implemented in a workflow environment. These processes divide and systematize the necessary tasks of planning, execution and quality assurance (QA), among others. A simplified outline of our process is presented in the Figure 1.

The most relevant business processes implemented in the system are the *Documents' entry* process and the History of the Law generation process. The

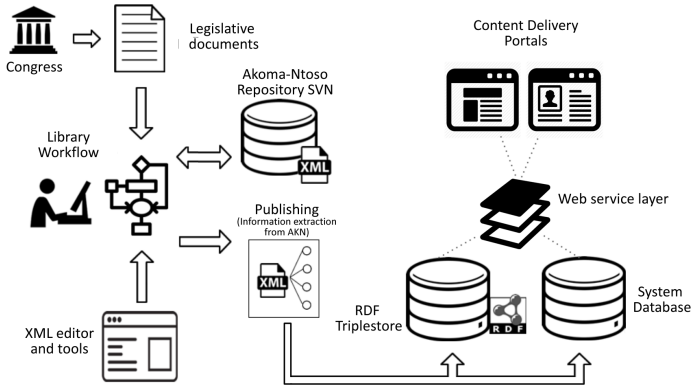

Fig. 1. Processing of legislative documents in BCN

first, allows to enter and mark multi-purpose documents, such as the session log, which may contain references to various laws, as well as the parliamentary interventions in session that will give life to the PL. Among the most important tasks of the process are automatic marking and manual editing of XML, the passage through QA and the document publication. The latter will be reflected in the PL of each assistant to the session processed. The second process, allows to create a chronological record of a bill, which contains all the documents in which the published law will be discussed. Among the most important tasks of the process are the entry of specific documents, marking (automatic, manual and XML editing), publication of marked documents and publication of the HL as a dossier.

3 Software environment

In this section, we present the most relevant elements of the system. An overview of the whole system has also been presented at [7] with a less technical focus.

3.1 Linked Open Data

The architecture of the system is based on the use of Linked Open Data, with the goal to establish a natural interoperability mechanism that enables sharing the public information that is generated. The project adopted Linked Data best practices [6] such as the use of URIs for identifiers within the system, a Linked Data frontend to access RDF resources[4], dereferenceable URIs modeling, reuse of existing vocabularies such as Dublin Core, SKOS, RDFS, FRBR[5], FOAF,

[4] https://code.google.com/archive/p/weso-desh/
[5] `http://purl.org/vocab/frbr/core#`

GeoNames[6] and GEO[7]; the implementation of multiple ontologies in RDF that allowed to shape the published datasets, and the publication of the data through a public SPARQL endpoint[8]. The query in Figure 2 can be executed in the SPARQL endpoint for getting the RDF, AKN and plain text representation of session documents.

The first proof of concept was the publication of an ontology and a dataset extracted from the legal database Leychile[9] as LOD, exporting basic metadata and relationships among norms and allowing the real definition of a graph with approximately 300,000 norms equivalent to 8 million RDF triples[8]. Later, a set of ontologies and datasets were published[10] for representing several domains, among others the HL and PL project, an entity dataset (people, organizations, etc.) used for entity linking, another ontology and dataset of geographical units, and an ontology to describe complex metadata related to the legislative business which is used for RDF publication documents marked in AKN. At this moment, in addition to what was mentioned before, the LOD database stores the RDF triples generated from the approximately 24,000 published AKN documents, as well as other datasets, such as the National Budget data, surpassing 28 million RDF triples in total (mid–November 2018).

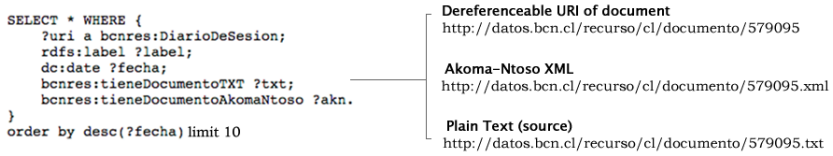

Fig. 2. SPARQL query for getting session documents in RDF, AKN and TXT

3.2 Automatic XML Marker

This tool[11] transforms a plain text document without format into an XML document based on Akoma-Ntoso schema. This problem has already been addressed using several approaches like machine learning[2], making use of visual properties of the text[5], looking for patterns associated with the content[4] or a combination of rules[1]. We defined three ways of approaching the problem, being the third a hybrid approach:

– **Knowledge engineering approach**: in this approach, the solution is based on the manual implementation of rules or ad-hoc algorithms, which are developed by a human specialist, a so-called *knowledge engineer*. The degree

[6] `http://www.geonames.org/ontology#`
[7] `http://www.w3.org/2003/01/geo/wgs84_pos#`
[8] `http://datos.bcn.cl/sparql`
[9] `https://www.leychile.cl`
[10] `https://datos.bcn.cl/es/ontologias`
[11] A test tool of the Automatic XML Marker can be found in `http://bcn.cl/28n7h`

of precision and effectiveness of this type of solutions is limited by the quantity and quality of the implemented rules so improvements in the tool will be directly related to manual adjustments and modifications. In our implementation, the techniques used with this approach, both for the detection of structural elements and entities identification in the text, were the application of regular expressions and exact matching. For this reason, for the implementation of structural marking through this approach, analysis and verification of at least a sample of the documents to be processed is required, as well as that they share some norms in the writing which can be identified and subsequently codified and integrated. In the case of entity recognition with this approach, it will be necessary to obtain a complete list of entity descriptors, names, labels and their variants.

- **Machine learning approach**: The main idea is that there is a set of documents previously marked or labeled by humans, structurally and/or their entities, which are randomly divided into a training set and one of tests in a proportion generally varying from 60% training - 40% testing to 90% training - 10% testing, depending on the number of elements available for training. The training documents are delivered as input to different classification algorithms based on the application of pattern recognition techniques (some of them being used as Hidden Markov Models - HMM, Naive Bayes - NB, Conditional Random Fields - CRF or Neural Networks) [12]) obtaining as a result a classification model that is used to mark the sources of testing documents. The quality check of the classifier is given by metrics such as Accuracy, Precision, Recall and F-Measure [3] that result from quantitatively and exhaustively comparing the results of manual labeling with those of automatic labeling.
  In practice, the marking components that most use this approach are mainly entity recognizers, and there is currently a wide range of off-the-shelf open source products that provide functionality, among which are the Stanford NER[12], spaCy[13] and OpenNLP[14].
- **Hybrid approach**: this is the one that we use in production, trying to capture the strengths of each approach to support the other's tasks. In this way, the most complex tasks of structural marking are carried out under the focus of knowledge engineering supported by recognition of entities (machine learning approach). In the same way, the recognition of entities is carried out only for certain structural sections, improving the efficiency and precision of the marking, and limiting it to the exclusively necessary.

In HL, given the complexity and detail of the marking of the different types of legal documents in which multiple tasks coexist, such as the detection and disambiguation of named entities, structural recognition of parts of the document and specific formatting, this function is implemented by four components spe-

[12] `https://nlp.stanford.edu/software/CRF-NER.shtml`
[13] `https://spacy.io/usage/linguistic-features#section-named-entities`
[14] `https://opennlp.apache.org`

cific orchestrated in a pipeline through HTTP Web services which are described below.

Named–Entity Recognizer: The task of Named Entities Recognition (NER) is to perform two fundamental functions: firstly identifying mentions of proper names or 'entities' presents in the text, for example dates or figures written in narrative prose, which can be composed of one or more words and secondly to identify the type of entity which the acknowledged name refers to.

We use the Stanford NER software implemented by a CRF classifier which, due to its probabilistic nature, delivers the input text marked with the recognized entity, associated with the possible types of entities (see Table 1 list of types) and its score of confidence in the match, being the type with highest score of the list that will be assigned as the one recognized for the entity. As a training corpus, text of session documents was used, composed of 64,727 words, which depending on the case were associated to their type by means of a label. Regarding the evaluation of the model in production, 10 fold cross-validation (90% training - 10% testing) was used, which reports that the NER manages to detect on average 97% of the entities present in the text and correctly assign the type of entity in 89% of cases. For the integration of the tool in the system, its functionality is provided by an HTTP Web service programmed in Java that receives plain text by POST and delivers the text with entities marked in XML.

Mediator This tool assigns the URI associated with an entity mentioned in

Table 1. Types regognized by NER and number of entities in the Knowledge Base

Named–Entity Type	Example	Total in LKB
Person	Salvador Allende Gossens, Sebastián Piñera Echenique	5.139
Organization	Ministerio de Salud, SERNATUR	2.848
Location	Valparaíso, Santiago de Chile	1.251
Document	Ley 20.000, Diario de Sesión N°12	732.497
Role	Senador, Diputado, Alcalde	428
Eventos	Nacimiento de Eduardo Frei Montalba, Sesión N° 23	14.389
Bill	Boletín 11536–04, Prohibe fumar en espacios cerrados	12.737
Date	27 de febrero de 2010, el próximo año, el mes pasado	20.632

the text performing the task of *Entity Linking* or *Disambiguation*. It is based on an RDF–based LKB which can be accessed through a SPARQL endpoint that stores information about basic descriptions (such as names or descriptors of text, dates or others) and more complex structures such as periods of membership or occurrence of events, all associated with entities of the types described in Table 1. For its operation, the mediator must load in memory pairs of (URI, label) that are queried to the triplestore via SPARQL and indexed internally. Once deployed, the job of linking or disambiguation is performed by text similarity algorithms based on Apache Lucene, comparing the input text with the tags that are associated with a map indexed in memory. Its simplest form of use is to send via REST a label by parameter, with which the mediator will return a list of suggestions in JSON with the structure (uri, label, score) ordered in a decreasing way by score. Some parameters can be added to improve their accuracy, such as the type of entity or a session ID. Additionally, it was implemented with another operating option in which an XML file with recognized entities is sent as input.

The mediator will return an XML file in which it will add the URI and label attributes in each entity, establishing the URI with the highest score calculated for each case, as long as this value is greater than or equal to a threshold, which can also be configured. Although tools similar to this one such as DBpedia Spotlight[13], AGDISTIS[16] or KORE[10] are becoming more common (and almost all developed after our tool), there are three characteristics that make the mediator fits our use case:

- **Use of context information**: To improve the precision of the tool, it is possible to assign context data that allow to delimit the set of possible alternatives when selecting the URI for each label to be identified. Since this tool is used to disambiguate entities in documents of the National Congress, some of the context data that can be provided and that in turn are useful are: the date of the session, the chamber of the document, the number of the session or the legislature. A simple example in the use of context information is to narrow the list of URIs of possible people to recognize only those who are in the exercise of their position on a specific date and camera. If context information is sent to the mediator, it will generate a session ID that will be stored and associated with the context data and that will have to be used when using that specific data. Additionally, this ID will allow learning services to be used, given that if the tool assigns a wrong ID, it is possible to feed it back with the correct value, and as a result, upon being consulted again, the answer will be given by the feedback value.
- **Filters and association heuristics**: they are programmatic classes that implement logic to narrow the search results. These filters allow to increase the accuracy in URI assignment, allowing to limit the list of candidates according to criteria such as if on some date X is talking about a father or a son (political families are a common phenomenon), if a person is alive on the date that is being processed or if a person X belongs to a specific chamber.
- **Specialized Web Services**: within these are the assignment service to full document, which receives an XML document returning all the entities recognized in it, making it possible to optimize the number of requests; the reindexing options by type of entity, the feedback services, and the services that allow obtaining suggestion lists with or without session ID.

Structural marker: We define as structural detection of text the task of identifying groups of consecutive strings that in the view of a human reader correspond to elements such as: titles, subtitles, paragraphs, sections (groups of paragraphs under the same title or subtitle) and other structures such as enumerations and lists. Additionally, given the context of application in documents containing parliamentary debate, a special type of structural element called Intervention (speech) will be defined, in which what is spoken by a person is described, and can be composed of one or more consecutive paragraphs. Then, we define the structural marker as the tool that performs the task of performing structural detection by adding marks to the text that indicate the beginning and end of each structural element. For our case, the marks added to the output of the text processing are in XML.

The main strategy used for the detection of structural and hierarchical sections in text documents is through the combined use of regular expressions and the application of rules that encapsulate programming logic that is applied when a certain regular expression is detected in the text. This solution is especially practical in the context of the documents generated in the National Congress, since there are usually some drafting rules that are standardized. In this way, the tool allows to identify structural sections of first, second and third level of document, sequences of elements based on numbered and unnumbered lists (even nested), as well as participations of parliamentarians. As the documents to be processed are mainly political debates, the main element to be recognized in the document is a block called *participation*. It is a block composed of one or more speeches in which an actor who moderates the session (usually the President of Chamber), an actor who owns the participation (who fundamentally speaks) and, eventually, some other actor who interrupts. In this composite block, it must be automatically identified which of the speakers is the participation author, wherewith an analysis of the discourses was necessary in order to detect the underlying structures of participation and be able to implement a rule with automaton characteristics. This particular implementation was needed given that in structural terms, participation is specific because it lacks a conventional structure of title and body, and it is included in other recognized structural sections of the debate. At the technical level, the structural marker receives plain text with or without entities as input. With this text, an object of type DocumentPart is generated that contains a property with all the text and an empty list of DocumentPart objects (sub-parts). The main idea is that depending on the type of document and the depth level of the DocumentPart object, an executor runs specific rules. In this way, each rule receives the text of a DocumentPart object and returns all the DocumentPart objects that can be consistently identified by adding them to the sub-parts list, so the last task of the process will be to serialize the object in XML.

XML converter to Akoma–Ntoso: This component performs the task of transforming the XML resulting from the structural marking processes, entity recognition and entity linking, which we will call raw XML, in AKN format. The resulting version of this process will comply with the standard, and consequently is editable by tools such as LIME[15], Legis Pro[16], AT4AM[17], Bungeni[18], xmLegesEditor[19] or LEOS[20].

For the conversion process from raw XML to AKN, we initially tried XSLT, however, the XSLT style sheets to perform some operations such as identifying references to entities and grouping them in the header, a fundamental practice in AKN, were highly complex. Likewise, given the high variability of documents

[15] `http://lime.cirsfid.unibo.it`
[16] `https://xcential.com/legispro-xml-tech/`
[17] `https://at4am.eu`
[18] `https://github.com/bungeni-org`
[19] `http://www.ittig.cnr.it/lab/xmlegeseditor`
[20] `https://ec.europa.eu/isa2/solutions/leos`

structure, we finally opted for a programmatic approach which takes the input XML file, generates a representation similar to a DOM, and traverses the XML node tree by converting each raw node to text in AKN XML. For this, each type of raw node can be implemented as a class that encapsulates the logic associated with the different types of AKN node, allowing to perform operations such as searching for specific expressions in internal nodes or their attributes.

3.3 Publishing

Each law has a life cycle that, in general, originates with a bill, then it is discussed, processed, published and with the passage of time, changing the cultural and political context, is explicitly or tacitly derogated from total or partial, ending its cycle. It is just before this last state, when the law is still in force and has maximum relevance. During the period of validity, the Law remains static, so that once having been published it does not change. In this way, both the Law and its HL can be filed, published and made available to the public. For this reason, at the end of the background collection and processing of documents that are part of HL, a publication process is run, which generates a physical file that is stored in a repository and archived by library specialists who index its content as part of the legal database. For this, every document associated with the HL must pass two final stages: quality control and document publication.

Quality Assurance : To ensure the consistency and correctness of HL products, QA processes have been implemented before a document is finally published. A QA analyst will exhaustively verify the work done in the previous phase and decide if it can be published or if, in case of some error having been found, it must be processed again. The QA process for the documents (such as session logs) is translated into a change of state in the workflow where a visual inspection is made using the XML editor of the document, checking things like the correct assignment of entity URIs, the correct metadata assignment and the verification of a well-formed document structure, for this reason it does not require an implementation other than the visualization of the document being edited. Regarding the QA process for HL publication, it is implemented using a combination of SPARQL queries to the RDF triplestore and relational queries to the system database in specialized user interfaces. In the HL case, the triplestore is queried in all bill processing and a preliminary view of the HL is generated based on the documents described therein.

AKN2RDF : Once the AKN files have been annotated both automatically and manually and have successfully passed the QA phase, the publication process is performed which extracts the knowledge expressed in the document in the form of RDF triples and tuples for a base of query data. This process, packaged as a web service, implements a parser that traverses the XML tree of the AKN document, looking for the predefined structures within the different document sections. At the technical level, each type of document implements a collection

of small data extractors per each document section, which are encapsulated in a specific class. In this way, classes that have the same behavior in different documents can be reused, and otherwise specific implementations can be made to require it.

3.4 Content Delivery

The BCN has made available to the public the portals of History of the Law[21] and Parliamentary Labor[22]. Both Web portals are built on a LKB supported by the extraction of information available in the documents generated in the National Congress. From a technological point of view, the portals have been developed using open source technologies (such as TYPO3 CMS, Python, Varnish, Java, Apache Lucene) and are mounted on a layer of REST web services that connect to both the RDF triplestore and a relational database depending on the case, offering in both cases search functions and data export to formats such as PDF, DOC and XML. A practice adopted is that the URIs of the parliamentary profiles are based on the same identifiers that form the URIs of people in the RDF triplestore.

4 Results

The growth of LKB BCN is a direct result of the publication and transformation of AKN XML documents to RDF, which has been carried out in four phases or data entry projects, where each one has been mainly oriented to the processing of session's documents associated with specific years intervals since 1965 until today. Table 2 describes the RDF triples that are obtained by type of document, evidencing that the largest contribution of triples is given by documents of the Chamber of Deputies. In terms of size, related to RDF triples obtained from session's documents we have around 14.250.000 RDF triples counted in November 4th of 2018 in our LKB.

Table 2. Description of RDF triples generated by document type

Dataset		**RDF triples by document**						
Document type	*Total docs.*	*Min.*	*1st Qu.*	*Median*	*Mean*	*3rd Qu.*	*Max.*	*Stdev.*
Debate Senate	3.614	35	665	1.001	1.097	1.353	6.883	668,67
Debate Chamber of Deputies	4.298	35	1.166	1.640	1.788	2.199	11.771	961,42
Bills	2.514	37	109	308	689	864	15.934	1055,93
Others types	13.942	37	40	40	49,78	40	15.720	245,68
Total	24.368							

Another perspective of the data generated is the number of triples by type of document published in the LKB per year (of document), which is shown in Figure 3. Except for the period from September of 1973 to March of 1990, when

[21] `https://www.bcn.cl/historiadelaley`
[22] `https://www.bcn.cl/laborparlamentaria`

in Chile the dissolution of the National Congress was carried out due to a period of 'non-democracy', this chart allows us to visualize a long-term upward trend, it may seem surprising that the growth of triple RDFs is not always greater than in previous periods.

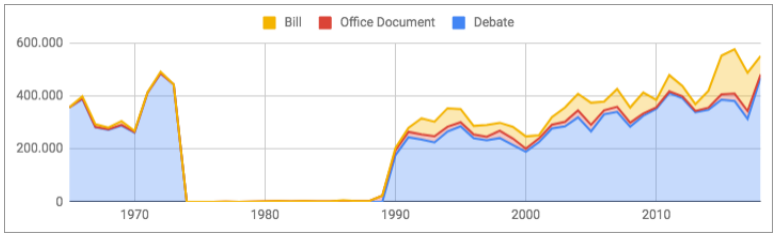

Fig. 3. RDF triples generated yearly by document type

Looking for an explanation, the Figure 4 shows the attendance to the sessions of the parliament and the total of sessions per year adding the data of both chambers. Although both sets of data are part of the LKB, the sessions are created in RDF from a Web service provided by the opendata.congreso.cl portal, and the assistance is extracted from the documents published in AKN. In this case, a direct relationship between the number of sessions and attendance is shown, which could partially explain the RDF triples generation losses of the Figure 3. An interesting thing about these data is that there seems to be some relationship between the number of sessions and the existence of elections (parliamentary or presidential), which may explain the differences between years regarding the number of triples generated by type of document. With regards

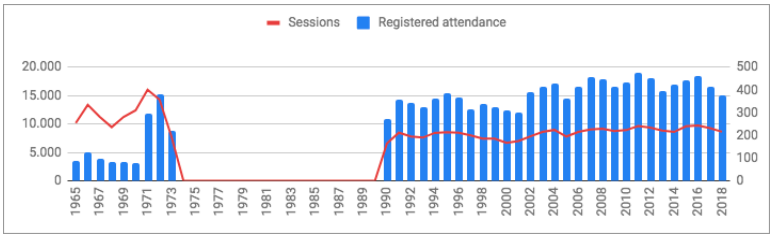

Fig. 4. Session attendance registered in Akoma-ntoso documents by year

to user queries registered by Google Analytics for the Content Delivery portal of History of the Law, between November 2016 and November 2017, 331,481 visits were received with an average browsing time of 2 minutes and 11 seconds. One year later between November 2017 and November 2018, the number of visits is approximately 476,241 and the average browsing time is 2 minutes 26 seconds, which represents an increase in visits of 144,760 equivalent to 43.6%, and an increase in the average time on the page of 15 seconds equivalent to 11.4% of additional time.

A final aspect to be analyzed is the time necessary to carry out the main marking operations associated with legislative documents. For this, the values provided by the literature referring to manual and semi-automatic marking of legal documents were taken as a baseline, which were compared with the usage statistics associated with the journal entry process, generated from the History of the Law system. Specifically for the comparison, data of document marking, transitions and status changes of the work orders (WO) registered in the production workflow were used, taking into account only those WO executed correctly (without errors during the process, whether or not they were resolved), and that they were not suspended explicitly or implicitly. Under these conditions, 2,625 WOs of entry were obtained, for which the information is summarized in Table 3. For purposes of comparison with the results described in [14], an estimate was made based on random sampling of session's documents, which considered that a document has on average 60 pages of content, which was calculated by the number of average document characters (180,000) divided by the average number of characters that has approximately one page (3,000). With this data, and considering that the extension of the document is linearly increased with the number of marks and ultimately, the human analyst's work, the times obtained in Table 3 will be divided by 3, given that the values described as baseline, are based on a 20 page document. Under these considerations, it is possible to construct the Table 4, which shows differences in the estimation of marking times between what is presented in the literature and what is obtained by analyzing process data.

5 Discussion

The comparison in Table 4 shows that our automatic marker generates more marks, which results in marginally less manual processing time and more references per document. Additionally, it is shown that the average time used for processing obtained by usage statistics is less than that described in the baseline, which may be due to the analyst's greater expertise (much more practice in marking), as well as to a more advanced environment of tools to support the marking.

At the beginning of the HL project, a pilot phase of document entry was developed, where differences in the documents marking implementation were experienced with respect to the current one. This could explain what is shown in Figure 4 where, during the 1965-1973 period, it is observed that assistance is much less recorded between the years 1965 to 1969, very different from what happens since 1971 to 1973. From a technical point of view, although the initial idea of the project was to build the whole system natively in RDF using LOD (for both frontend and backend), during the development of the first prototypes associated with the Content Delivery portals, different approaches were tested for consulting and obtaining structure and texts associated with both HL and PL:

The first approach considered the implementation of SPARQL queries on a dataset modeled by an ontology with several classes and properties that extended

Table 3. Statistics for markup related to entry document process, all time in minutes

	Markup from plain text						**Markup from existent XML**					
	Min	*1st Q.*	*Med.*	*Mean*	*3rd Q.*	*Max.*	*Min*	*1st Q.*	*Med.*	*Mean*	*3rd Q.*	*Max.*
total structural marks	0	10	23	503,6	665	2.468	0	6	10	11,14	15	40
total reference marks	0	14	30	33,6	46	525	0	20	32	33,49	44	173
WO Creation time	0,0	0,91	1,13	2,22	1,75	125,16	0	0,80	1,18	3,17	2,28	89,03
automatic markup time	0.0	0.31	0.51	0.67	0.76	16.86	0.0	1.26	1.73	2.15	2.41	31.51
QA automatic markup time	0.0	0.58	1.08	3.74	1.88	296.7	0.0	0.97	1.35	4.26	2.31	303.35
manual markup time	0.0	29.8	71.3	98.1	135.1	581.4	0.0	42.06	80.17	109.01	144.21	593.62
QA manual markup time	0.08	6.22	15.35	19.93	30.91	59.85	0.10	8.70	20.07	21.59	31.76	59.97
publishing process time	0.08	0.40	0.86	1.85	2.03	29.68	0.06	0.73	1.18	1.76	2.00	29.56

Table 4. Statistics for markup related to entry document process, all time in minutes

		Baseline		**Usage statistics**	
		Manual Markup	*With automatic tools*	*From plain text*	*Process from XML*
20 pages document	Average minutes	120	75	42.16	47.34
	Average references	10	10	11.2	11.16
60 pages document	Average minutes	360	225	126.49	142.03
	Average references	30	30	33.6	33.49

from others (rdfs:subPropertyOf, rdfs:subClassOf), defining in turn properties' domains and ranges, which in practice meant that the dataset had fewer specific RDF triples while many others had to be inferred. The reasoner used was the same provided by the Virtuoso database. When testing this approach, it was discarded because it required high running times for simple and complex queries, with some cases where the process didn't finish.

A second approach tested was to generate and add *a priori* to the dataset all those RDF triples that were inferred in the previous approach, which eliminated the query time devoted to inference. In this case, it was actually possible to reduce the queries execution time by getting them to finish, however, the response times were still too high for the implementation of a production system, even more so considering the HL and PL which are composed of a large number of documents. For example, when generating a dossier of PL in Word or PDF, it can have over 10,000 pages of text. The problem became even greater when it was necessary to add a search filter to the SPARQL query to perform searches within the text of *participations*, which is a common use case, and for which the database was not prepared to be specialized in RDF and not in text indexing.

The third approach which was finally adopted was the use of RDF and SPARQL only to give access to specific parts of the database, such as the procedural structure of a bill, to establish a basis for universal identifier data (URI) common to all BCN systems, as well as to provide access to the community to public information generated in the National Congress, but not to access the text

of interventions or metadata associated with them. For these purposes, both a relational database and a text index were implemented using Apache Lucene.

While the first two approaches mentioned above are empirical evidence of the computational complexity associated with complex SPARQL queries [15], it is possible that, having used other approaches such as Linked Data Fragments [17] or even a better query decomposition [15] and/or a scheme of greater redundancy of the service, could have allowed a system based on RDF. Indeed, newer experiences like Wikidata [11] suggest that it is possible to solve highly complex problems. Nevertheless, for the text search task, a tool is required that allows searching and efficiently retrieving over the entire dataset.

Another practical learning from this experience is that there are performance problems in users' browsers when they must edit large documents with many metadata marked on text. We mention this consideration because in our case it was difficult and even impossible to edit documents with more than 100 pages that were marked with a high level of detail (marking of all entity types for all structural sections). In order to correct such inconvenience, the level of detail of marking was reduced only to those structural sections that would later be extracted, and in the same way, only to those entity types of interest for the generation of products.

6 Conclusions

This work allows to validate the use of Semantic Web technologies for implementing systems oriented to the development of products based on semantic marking of texts, in the legislative context, and at the same time, to make available the information extracted as LOD. In addition, when comparing statistics of use about human work in tasks of document markup realized on the implemented architecture, versus the data established in the literature, it is possible to establish that, based on the statistics of our system, the time of human work necessary to carry out the editing and marking of documents are smaller than defined as baseline.

We consider that the experience gained in the development of the HL and PL project within BCN has been overall positive. The use of LOD (data and shared models using RDF and HTTP) and particularly dereferenceable URIs, has allowed to interoperate between internal websites, and in turn, to replicate the interoperability standard in new developments such as the portal of the National Budget[23], which also publishes a knowledge graph over HTTP.

7 Acknowledgements

We wish to thank David Vilches, Eridan Otto, and Christian Sifaqui by their contribution to the development of the HL project, that was funded by the Library of Congress of Chile. The described research activities were partially funded by the Spanish Ministry of Economy and Competitiveness (Society challenges: TIN2017-88877-R)

[23] `https://www.bcn.cl/presupuesto`

References

1. Abolhassani, M., Fuhr, N., Gövert, N.: Information extraction and automatic markup for XML documents. In: Intelligent Search on XML Data. Springer (2003)
2. Akhtar, S., Reilly, R.G., Dunnion, J.: Automating XML markup using machine learning techniques. Journal of Systemics, Cybernetics and Informatics, Vol 2, Number 5 pp. 12–16 (2004)
3. Baeza-Yates, R., Ribeiro-Neto, B.: Modern Information Retrieval: The Concepts and Technology behind Search, vol. 82. Pearson Education Ltd., Harlow, England (2011)
4. Bolioli, A., Dini, L., Mercatali, P., Romano, F.: For the automated mark-up of italian legislative texts in xml. In: Legal Knowledge and Information Systems (Jurix 2002. pp. 21–30. IOS Press (2002)
5. Burget, R.: Automatic document structure detection for data integration. In: Abramowicz, W. (ed.) Business Information Systems. pp. 391–397. Springer Berlin Heidelberg, Berlin, Heidelberg (2007)
6. Chris Bizer, Olaf Hartig: How to Publish Linked Data on the Web - Half-day Tutorial at the 7th International Semantic Web Conference (2008)
7. Cifuentes-Silva, F.: Service-Oriented Architecture for automatic markup of documents. An use case for legal documents. IFLA 2014 LYON p. 10 (2014)
8. Cifuentes-Silva, F., Sifaqui, C., Labra-Gayo, J.E.: Towards an architecture and adoption process for linked data technologies in open government contexts. In: Proceedings of the 7th International Conference on Semantic Systems - I-Semantics '11. pp. 79–86 (2011)
9. Gacitua B, R., Aravena-Diaz, V., Cares, C., Cifuentes-Silva, F.: Conceptual distinctions for traceability of history of law. In: Rocha, A. (ed.) 11th Iberian Conference on Information Systems and Technologies (CISTI). IEEE (2016)
10. Hoffart, J., Seufert, S., Nguyen, D.B., Theobald, M., Weikum, G.: Kore: Keyphrase overlap relatedness for entity disambiguation. In: Proceedings of the 21st ACM International Conference on Information and Knowledge Management (2012)
11. Malyshev, S., Krötzsch, M., González, L., Gonsior, J., Bielefeldt, A.: Getting the most out of Wikidata: Semantic Technology Usage in Wikipedia's Knowledge Graph. In: International Semantic Web Conference (2018)
12. Martinez-Rodriguez, J.L., Hogan, A., Lopez-Arevalo, I.: Information Extraction meets the Semantic Web: A Survey. Semantic Web journal at IOS Press (2018)
13. Mendes, P.N., Jakob, M., García-Silva, A., Bizer, C.: Dbpedia spotlight: Shedding light on the web of documents. In: Proceedings of the 7th International Conference on Semantic Systems. pp. 1–8. I-Semantics '11, ACM, New York, NY, USA (2011)
14. Palmirani, M., Vitali, F.: Legislative xml: principles and technical tools. Tech. rep., Inter-American Development Bank (2012)
15. Pérez, J., Arenas, M., Gutierrez, C.: Semantics and complexity of sparql. ACM Trans. Database Syst. **34**(3), 16:1–16:45 (sep 2009)
16. Usbeck, R., Ngomo, A.C.N., Röder, M., Gerber, D., Coelho, S.A., Auer, S., Both, A.: Agdistis-graph-based disambiguation of named entities using linked data. In: International Semantic Web Conference. pp. 457–471. Springer (2014)
17. Verborgh, R., Vander Sande, M., Colpaert, P., Coppens, S., Mannens, E., Van de Walle, R.: Web-scale querying through Linked Data Fragments. In: Proceedings of the 7th Workshop on Linked Data on the Web. CEUR Workshop Proceedings, vol. 1184 (2014)