Law Modeling with Ontological Support and BPMN: a Case Study

Aaron Ciaghi, Komminist Weldemariam, Adolfo Villafiorita
Fondazione Bruno Kessler
Via Sommarive 18, Trento 38100, Italy
Email: (ciaghi,sisai,adolfo)@fbk.eu

Abstract—Modeling and analysis of legal documents is becoming more widely used in eGovernment practices. To support these activities, various frameworks, standards, and ICT-based tools have been developed in the recent years. These approaches are mostly oriented towards defining common standards, managing legal documents and check compliance with current regulations. Along this line, we have devised a tool-supported methodology that allows to model and analyze laws and procedures within public administrations. The approach used in this paper is based on BPMN for the visualization and formalization of business processes and on OWL-DL for the specification of a business process ontology. In this paper we demonstrate the applicability of our approach by using the part of the Italian Immigration Law concerning Family Reunification as a case study. From this initial investigation, our approach can be used on actual laws that describe or affect procedures.

Keywords—Laws, Procedures, Modeling, Ontology, Business Processes, Public Administration.

I. INTRODUCTION

Modeling the semantics of laws is gaining attention in the field of legal informatics. Providing a graphical representation of a law can be of great advantage to those who want to understand or analyze it (e.g., citizens or jurists) as well as those who need to implement it. Furthermore, law modeling can play a key role in software engineering (e.g., [1], [2]) for the automation of Public Administration (PA). However, law modeling is typically not qualified, and thus there is no guarantee that the models representing a law are correct and, more importantly, traceable. Therefore, such models still need to be fully linked to the regulations by which they are defined and directed.

Legal documents must be made available and accessible in order to facilitate any type of analysis. In order to address these issues, the governments of several countries have adopted XML-based standards for storing and structuring legal documents (for an overview and a critique of available standards see [3]). The use of XML creates new possibilities of integration of laws with other knowledge management technologies, such as ontology-based reasoning techniques and natural language processing [4], [5]. Maat and Winkels [6] also argue that in order to make law sources available to machines, they need to be translated from natural languages to some form of formal languages.

We have been working on the development of tool-supported methodologies that facilitate the modeling and analysis of laws in PAs. Among these, we mention our VLPM 2.0 tool [7], [8] that aims at helping the modeling of processes defined by laws, by semi-automatically extracting processes from a legal text marked with special XML tags. In the most recent version of our tool design, we introduced an ontology based intermediate representation of the information contained in laws. Our ontology is written in OWL-DL as an extension of the LKIF core ontology [9].

This paper is a continuation of our previous work [8], in which we present the application of the VLPM 2.0 approach to a concrete case study. The case study we consider is the procedure that permits legal immigrants to apply for Family Reunification, as defined by the Italian Immigration Law. Requirements for the family reunification request depend on various conditions as detailed in the law (e.g., the availability of suitable housing and sufficient income). We focus on (legal) documents that define, regulate or in some way affect the family reunification procedures. Notice that such legal documents should ideally be shown to be contradiction-free both internally and with respect to the governing policies that need to comply with certain regulations. Moreover, there must be a mechanism ensuring that the procedure is respected. In other words, procedures should be modeled and made available for further analysis. In this paper we are only interested in the modeling aspect of the family reunification procedures, and are not concerned with their analysis —possibly formally against legal requirements. Our investigation shows that our approach is reasonably enough to model laws that describe or in some sense affect procedures.

The goal of this paper is that of demonstrating our approach on a real case study and to prove its practicality. We also intend to discuss the difficulties of applying the current approach as well as its shortcomings. Thus, in the next section we present the background materials. We present the core concepts of VLPM 2.0 approach and describe its modeling steps in Section III. In Section IV, we apply these steps onto the family reunification case study. Finally, in Section V, a brief analysis of the current limitations and outline possible future work.

II. BACKGROUND

A. Law Modeling

Legal informatics deals with methods in which information is handled, and possible ways in which technology can make a legal system both easier to work with and more efficient.
In fact, the coding of laws and obligations is typically done with the aid of natural language. Hence the visualization of the encoded information is restricted to simple organizational mechanisms such as numbered paragraphs. Dependencies between them are coupled and can be visualized by pointers to other paragraphs.

“Procedures were seen as sequences of acts generating and processing legal information, and its automation was the realm of what was then called management legal informatics” Aguilό-Regla [10].

In order to assist this, a number of tools and techniques have been designed to aid lawmakers and public officers in day-to-day activities vital to a public administration. A variety of new techniques, methods, and technological applications will likely be developed in order to assist the understanding of often overlooked legal documents, as well as new methods of storing and analyzing such documents.

B. Technology and Approach

Existing technologies and techniques for law modeling span from making legal information available (e.g., AKOMA NTOSO [11], [12]), annotating laws with context-specific legal ontologies (e.g., Legal Knowledge Interchange Format (LKIF) core ontology [9]) to modeling and formally checking laws against legal requirements [13], [14], [15]. Moreover, works that concentrate on the use of visual modeling languages to represent Public Administration procedures as business processes in order to redesign such procedures have been discussed, e.g., in [16], [17].

The approach used in this paper is based on BPMN for the visualization and formalization of business processes and on OWL-DL for the specification of a business process ontology. The LKIF core ontology is designed as part of a generic architecture for legal knowledge systems. It supports concepts like actions, agents (which correspond to UML actors, with the difference that agents must play a role to perform an action) and organizations (that in UML would be represented as actors or groups of actors). However, the support that it gives to process modeling is at a very high level.

Thus, we developed an OWL-DL ontology (from now also called VLPM 2.0 ontology) in order to add semantic information about processes described in legal texts, by extending the concepts of LKIF core with a business process meta-model that borrows several entities from the BPMN meta-model [18]. We then defined some concepts that can effectively address our needs. We used Pellet Reasoner to identify and consolidate equivalences and other relations with LKIF-core classes. The VLPM 2.0 ontology is not a specification of the BPMN meta-model in OWL. Instead, it abstracts the core entities of a business process from the BPMN meta-model in order to obtain a smaller but more generic ontology. In the sense that a set of instances of the classes in such ontology can easily be transformed to BPMN as well as UML Activity Diagram (AD) [19].

C. Other Approaches

The integration of two complementary methodologies was introduced in [20], as it was inspired by the VLPM [7] and Nόmos [21] methodologies. While the latter is a modeling framework that extends goal-oriented modeling paradigm for arguing about compliance of requirements, the former is a modeling methodology that follows a BPR-based approach with a particular focus on PAs’ processes. The integration of these two approaches suggests a top-down reasoning in which the leaves of a Nómos model are the procedures of a related VLPM model.

The integration of different knowledge management technologies is gaining interest as a tool to aid the introduction of eGovernment solutions. For example, Francesconi et al. [5] introduce an integration of ontology with law modeling and analysis to help in assessing decisions in software design of public administration applications. Agnoloni et al. [22] argue on the growing interest towards linguistic and semantic technologies due to the need to overcome the problems of access and knowledge of the legal information. These instruments developed for different linguistic and legal systems are also a methodological necessity to approach the ever growing problems related to multilingualism in legal text, to the harmonization between EU and National legislation and to the comparative analysis of Law.

Finally, works that describe how to use modeling languages and formal techniques for modeling, specifying, and analyzing business processes and workflows are well described, e.g., in [23], [24]. However, little is usually said on the attempt to model laws and procedures in favor of public administration.

III. FROM LEGAL DOCUMENTS TO MODELS

In this section we describe how our approach can be used to identify information in a set of legal documents that is relevant to our modeling. We divide this into three phases:

1) Markup: in this phase we add semantic information to parts of the text that are relevant to the domain that we are going to model. We do so by marking them with tags defined by the AKOMA NTOSO schema. This identifies parts of the text as elements of our model.
2) Transformation to RDF: the parts of the text marked as elements of our model are replicated as instances of classes of our Business Process Ontology. We obtain a set of RDF statements that represent our model and that are used for traceability.
3) Conversion to BPMN: finally, we convert the RDF statements into a BPMN model of the process by using a set of translation rules.

These phases can hardly be automated. For this reason, user interaction is required in order to produce an accurate model. However, in this section we will not discuss an actual implementation of this approach and we thus omit any reference to user intervention. In what follows, we discuss these three phases.
A. Markup

As noted earlier, several formats for legal documents markup are available. Since we intend to make our approach applicable to any legal system and country, we have chosen the AKOMA NTOSO (AN) framework as our input format. AN presents a clean and reiterated structure as well as ontology support (which is a stronger requirement in our case since we intend to use RDF/OWL as interchange format). Furthermore, AN is supported by tools (developed in the context of the Africa i-Parliaments initiative\(^1\) by UNDESA) for authoring and managing legal documents.

AN documents contain a metadata section that contains several subsections to specify identifiers for the documents and information related to the publication of the paper-based version of the document. The `<lifecycle>` and the `<references>` sections are of particular interest to us as the former allows to specify the events in which each document has undergone, while the latter allows to list the entities, individuals, concepts and other documents. All references must thus be explicitly declared in the `<references>` subsection of the `<metadata>` section and are classified by Top Level Classes (TLCs). Each reference has a URI attribute that points to an external resource —in our case, this will be the URI of the instance of a class of our ontology—and an ID attribute that identifies the use reference inside the document in which it appears, as illustrated by the following piece of code:

```xml
<references source="source-link">
  <TLCRole id="name" href="..." showAs="..."/>
  <TLCRole id="author" href="..." showAs="..."/>
  ...
</references>
```

Moreover, AN provides several mechanisms to link snippets of the text to declared references. In general, the optional `refersTo` attribute can be used for any block element. However, since we want to keep traceability as fine grained as possible, we consider the use of inline elements as a better solution. Therefore, a piece of text should be tagged with a `<span>` tag with the `refersTo` attribute set to the ID of a declared reference, such as in the following snippet:

```xml
<span refersTo="applicant">
  [a] legal alien resident who applies for Family Reunification
</span>
```

Moreover, AN provides several mechanisms to link snippets of the text to declared references.

B. Transformation to RDF

Our next step is mapping textual elements to instances of ontology classes in an RDF Store. Figure 1 shows the core classes of our ontology that represent (business) process entities. The ontology is based on LKIF-core in order to be able to connect business process concepts to legal concepts. This allows us to maintain the models and the laws “synchronized” by relying on a triplestore containing instances of the classes of our ontology. A triplestore is a purpose-built database for the storage and retrieval of RDF meta-data, in this case backed by an OWL ontology. We generally call this an “RDF Store”. All the above-mentioned concerns are implementation details and are incorporated in the design of VLPM 2.0. Due to space limitation we are unable to present them; the technical details can be found in \([8, 25]\).

Traceability has to be maintained both from laws to models and from models to laws. AN’s ontology and RDF Store support provide us with all the machinery needed to maintain traceability between laws and models by establishing links.

\(^1\)Africa i-Parliaments: \(\text{http://www.parliaments.info/}\)
between fragments of documents and model elements. We achieve this by declaring references to entities in the RDF store in the `<references>` block of an AN document using TLCs. An inline reference points to a TLC reference using its local ID. As said before, each TLC reference has a URI that points to an entity in the RDF store, thereby allowing an inline reference to be connected to such entity. Backwards traceability is achieved by using an object relationship that connects a model element to an object with the same URI of the inline reference. In this way we keep laws and models synchronized, thereby allowing the evaluation of the impact of changes on both sides. A very similar mechanism is used to achieve traceability between the RDF representation and the visual representation.

C. Translation to BPMN

At this point we have instances of the ontology classes represented in RDF Store. Thus, our next step is mapping them into the process model entities. Our ontology of business process entities is designed using principles of UML-AD [19] and BPMN [18]. This simplifies the generation of a model in one of these two notations from the contents of the RDF store. It should be noted that this requires to understand the meta-model elements of both the source and target. We devised a translation table (not shown in this paper due to space limitations) that maps ontology classes to AN TLCs, BPMN entities and UML entities. For example, the `Actor` class is mapped to Person in AKOMA NTOSO TLC, which is translated to Pool/Swimlane, and Swimlane (AD) and Actor (UCD) in BPMN and UML, respectively. Notice, however, that when mapping to activity diagrams, that automatic extraction of information about the sequentiality of activities is not an easy task. Thus, we must provide a way to personalize the elements in the RDF store by adding relations and properties that are needed to model sequentiality and temporal relationships in general. Finally, we shall also mention the difficulties of linking some business process modeling notations to fragment of texts. For example, one of the core elements of business process modeling notations is the Gateway object; however, there is no way to link a gateway to a fragment of text. We handle such cases by manually adding the required information when performing the mapping.

IV. CASE STUDY: THE ITALIAN FAMILY REUNIFICATION LAW

This section presents the execution of the steps discussed previously on the Italian Family Reunification regulation case study. Without going into specific details, a legal permanent resident alien in Italy who wishes to apply for family reunification must first obtain a set of documents to prove that he or she will be able to sustain his or her family. Among these documents, one of the hardest to obtain is a certification that the applicant’s house can accommodate his or her family. Once all such documents have been obtained, the applicant has to submit the actual application electronically through the website of the Ministry of Internal Affairs. The procedure includes several transactions with different public offices. For this reason, Family Reunification can be an interesting example on which we can test our approach.

We chose the Family Reunification law as case study because it is well supported by a local public organization called CINFORMI whose objective is that of facilitating immigrants’ access to public services. We are interested in providing CINFORMI with an automated system to provide information on such services and to automate, where possible, the interaction with its “clients”. Therefore, we have taken into account different types of documents (i.e., the text of the Immigration Law and the instructions that CINFORMI provide to immigrants) to represent the whole procedure, including the interaction with CINFORMI. In conducting our case study, we followed the ideal workflow of VLPM 2.0 (cf. Section III).

1) Marking-up: We added AN markup to the relevant parts of the Unified Text and the Implementation Regulation that compose the Italian Immigration Law. We started from the text of the law (originally in PDF format) and we replicated the structure of the law using AN tags.

2) Tagging: We tagged parts of the text that identified entities of the Family Reunification business process. For the sake of example, we translated part of the unified text on Family Reunification3 as shown in the listing below.

<references>
<TLCProcess name="Provide housing suitability"
href="/ontology/person/actor/applicant"/>
<TLCObject name="Housing Suitability Certification" id="suitable.housing"
href="/ontology/person/actor/applicant"/>
<TLCProcess name="Verify Housing Suitability"
href="/ontology/person/actor/applicant"/>
<TLCOrganisation name="Municipal Office"
href="/ontology/person/actor/applicant"/>
</references>

...
The example specifically shows how actors, tasks and artifacts are tagged. In the example, the statement, despite being relatively vague, gives us information on one of the requirements to obtain clearance for Family Reunification, i.e. obtaining a housing suitability certification. As often happens in laws, information is spread across several documents and still its implementation is mostly left to interpretation. For this reason, in order to build a more accurate model, we had to integrate the contents of our legal sources with non-legal documents such as the instructions published by CINFORMI on their website. We found these kinds of instructions very useful not only because they help for the interpretation of the law also for inferring the order of procedure can be extracted, if they are carefully written. Namely, they help to determine the sequentiality of the activities for the activity diagram.

3) Referencing: The references in tagged documents to instances of classes of our ontology allowed us to establish traceability between text and an intermediate RDF representation of the model. This has proven to be extremely time consuming and error prone without software support. However, currently we are investigating the extension of the Bungeni Editor [26] in order to back these issues faced at the moment.

4) Integrating: Finally, by integrating all the information found in the unified text and in the implementation regulation, as well as “unofficial” sources such as CINFORMI’s instructions, we have been able to build a BPMN model of the process, focusing on the point of view of the applicant. Figure 2 shows the part of the model related to obtaining the housing suitability certification mentioned above.

The model in Figure 2 is just a small part of the complete business process. The complete model includes all the other actors such as the Ministry of Internal Affairs and the procedures to obtain all the other documents required for the application, containing 36 tasks and events.

V. DISCUSSION AND FUTURE WORK

In this paper we have shown how ontology and business process meta-model techniques and tools can be applied to model legal documents. In particular, we have used an extension of LKIF core and BPMN to model the procedure related the Family Reunification in Italy. First, we enrich the parts of the law text with semantic information following the AKOMA NTOSO schema. Second, we apply transformation rules to produce a representation in RDF of the model using the classes defined in our VLPM 2.0 ontology. This increases our confidence in the traceability between the model and the text. Finally, we apply translation rules to convert the RDF statements to a BPMN model by ensuring certain desirable properties.

Related Work

Several strategies have been proposed in the literature to model laws [6], [13], [16], [17]. The main difference with our approach is that in [13] there is no ontology for supporting the model exchange and traceability, instead a graphical editor for the User Requirements Notation (URN) —called jUCMNav— is used to evaluate the compliance of processes to legal requirements and also by establishing a traceability links between elements of the goal model and the procedure. Similarly, in [17] event-driven process chains are used to translate law paragraphs into process models with the support of semantic process language. The main goal of the authors is that of visualizing and formally model a legally regulated process. The work described in [27] is closer to ours. In a sense that the authors first transform unstructured legal text into the MetaLex [28] XML interchange format. Secondly, using MetaLex they are able to find and resolve all references in the text and tag these explicitly. This allows them to easily recognize and classify norms in the legal sources [6].
Future Work

Public Administrations keep facing issues due to the complexity of legislations and the continuous evolution of the body of laws of their country. Laws are continuously added, amended, and repealed, often causing inconsistencies that can go unnoticed even for several decades. With the rise of transnational institutions such as the European Union, this is further complicated.

Despite the fact that we analyze laws from a technological point of view, we should not underestimate the fact that laws are mainly a product of political representatives, who might have an agenda that does not include facilitating understandability. This represents the main obstacle to the introduction of law modeling as a tool to formalize law design. Notice also that the interpretation of a law for a non-jurist remains a bottleneck. This is a time and effort consuming task, usually performed by knowledge engineers with the aid of legal experts. However, some promising approaches (e.g., [6]) can be adopted in our future work, for creating an intermediate model that has an (isomorphic) representation of the structure of the original text before starting the modeling task.

Additionally, at the moment, our approach lacks a software tool to tag the text and to generate the model. Tools to perform the two activities exist, but no integration is currently available. Namely, we do not have a machinery that allows us to automatically reference instances of classes of the ontology in order to establish traceability between text and an intermediate RDF representation of the model. However, the extension to Bungeni Editor [26] that we envisaged in [8] should be able to significantly mitigate these issues.

REFERENCES


