Automatic Translation from Textual Representations of Laws to Formal Models through UML

Pietro Mercatali and Francesco Romano
ITTIG-CNR, Via Panciatichi, 56/16 - 50127 - Florence, Italy
{mercatali, romano}@ittig.cnr.it

Luciano Boschi and Emilio Spinicci
Dipartimento di Sistemi e Informatica – Università degli Studi di Firenze, Italy
{boschi, spinicci}@dsi.unifi.it

Abstract. A combination of UML and text mining, or more in general Information Extraction, can provide a valuable help to people involved in research about the linguistic structure of statutes, and, as a side effect, can be the seed for a new generation of applications in the legal domain. In particular, in this paper we present LexLooter, a prototype for amendments modelling and legislative text co-ordination based on UML and Natural Language Processing.

Keywords. Amendments, Legimatics, LexEdit, LexLooter, NLP, UML

1. Scope

We present in this work tools and techniques for the generation of a formal representation of legislative documents using UML modelling [1,2,3] with the purpose to represent connections between normative dispositions inside different normative acts. Such techniques exploit NLP for information extraction and syntactical and semantic tagging via XML markup of the elements of a normative text. The structural tagging allows to build a UML model representing the external structure of legislative acts (articles, paragraphs, commas, letters, etc.), while the text embodied inside each tag is used to instantiate each modelled element.

In this context, we have applied this modelling technique to Express Text Amendments (ETAs), referring to a given normative text, for the compilation of the coordinated text through the corresponding UML model.

The readability appointed to the *coordinated model* by UML improves the processing of modification applied to the original normative text by its ETAs, thus constituting an added value for the representation and the interpretation of the legislative text.

This paper is structured as follows: legal rules that constitute the basis for the UML representation of normative acts (and in particular of Express Text Amendments) are detailed in sections 2 and 3; section 4 briefly introduces NLP tools for legislative information markup, developed by the authors in previous works [4], [5] on which the prototype presented in this paper is based; section 5 details the UML model instantiated by LexLooter (described in

section 6), the tool implemented for the generation of the coordinated model and text; in the final section, perspectives and potential applications of the methodology will be discussed, to contribute to the drafting, the communication and the interpretation of the legislative acts.

2. Identification and Description of the Models on the Basis of Legal Rules

The legislative instrument has, by definition, a prescriptive function, or in other words, it influences the behaviour and status of the addressee, who cannot escape from this function.

In virtue of this, the request (it is based on the same principle of representative democracy which legitimates and, at the same time, binds those who draft legislative acts, namely, the legislator) is that the legislative discourse relies on a set of rules that dominate and, at the same time, stand beside, integrate, and sometimes modify the rules that make up natural language, in our case Italian.

These rules can be defined as legal rules. The category is broad and varied; it includes within it rules with prescriptive force, that vary remarkably on the basis of the source from which they come, to whom they are addressed and the sanctions they bring with them, etc..

Concerning the enforcing and effectiveness of the models constructed on the basis of these rules, we would like to add several considerations. We are able to list three characteristics on which their efficiency largely depends:

1. Flexibility of the model, which must be adaptable to the many structural, functional and thematic characteristics of legislative instruments, which use that extremely changeable and unpredictable vehicle known as language.
2. Precision in the definition of the model itself, thanks to the presence of the legal rules which are in many cases detailed and precise, and, for the prescriptive nature of the legislative instrument, it goes without saying that reconciling flexibility and precision constitutes a crucial point in the construction of models.
3. Authoritativeness with which the model must be endowed, in order to be shared and held to be valid by all the users of the system. It is clear that this authoritativeness is guaranteed more if the model is derived from the legislative rules, in the strict sense, according to their hierarchy, while it will be less guaranteed, for example, by those rules which only the drafter must follow. It could be more or less guaranteed, to a different extent, by models based on consolidated practices, legislative theories shared by the major part of legal authority, and so on. We can, nevertheless, say that while could greater precision correspond to “more authoritative” models, greater flexibility should correspond to "less authoritative" models.

It should further be noted that very authoritative models have greater communicative value (they are directed at and basically are accepted by everyone, because they are enforced by strong and precise rules), whilst as authoritativeness gradually decreases, the models assume an always increasingly interpretative value, that is, they are a “subjective” definition of the function of the single textual structures or of the entire legislative instrument.

If we can, therefore, talk about “Normativistic Models” and “Interpretative Models”, it becomes fundamental to evaluate how we want to use them, and consequentially, refer to whether to the former or the latter.

For example, for the description of a legislative instrument, for the purpose of communication *erga omnes*, we cannot fail but to refer to a “normativistic model” while the use of an “interpretative model” could be dangerous and misleading.

It should be also specified that the models supposed to be defined for the corresponding

UML formalization can refer either to the whole legal system or to individual acts or dispositions, as well as elements inside each disposition.

Such models are however constituted by textual elements and do not allow the formalization of extra-textual elements implying the interpretation of the text on the basis of real world situations. So, we can talk about a normativistic approach rather than an interpretative approach, directed therefore to investigate the unity, consistency and completeness of the modelled element (legal system, act, and so on) rather than assumptions and premises or implications and, particularly, consequences of such an element [6].

3. Express Text Amendment or Novella: Definition of the Structure and of the Constituent Elements

We propose the modelling of text amendment dispositions, both because this fits for the particular application of this study, and because the Express Text Amendment or *Novella* is a structure peculiarly bound with linguistic and the legislative technique rules used for its definition. Moreover, the treatment of amendment dispositions is also subject of analyses aimed to the reconstruction of the normative coordinated text [7].

Amendment provisions, according to Sartor, fall within the main types of legislative links "nessi", classified on the basis of their impact on the legal provision involved. Amendments distinguished from the other large branch of referrals or references, are legislative links characterised by the fact that the active provision affects the passive provision, eliminating it, changing the text or changing the legal significance (whilst leaving the text unchanged). This effect is, instead, lacking in the referral, where the active provision avails itself of the passive provision to complete its meaning, without influencing the latter [8].

In relation to the nature of the impact of the amendment on the passive provision, we distinguish between textual amendments, time-based amendments (that influence the period of time of the applicability of the passive provision), material amendments (that amend the legislative content of the passive provision without affecting the text). We shall only look at the first type, the express amendments of the text which, traditionally, lawyers in Italy name the *novelle*.

Indeed, it is perhaps more correct to say that the function of express legislative amendment is expressed through the following three aspects:

- the structure of the *novella*, made up of an introductory part, called *subparagraph*[1] and a part that contains the *express textual amendment;*.
- the characteristics of the amending legislative act and the amended act: indispensable for subsequently being able to reconstruct the amending links between the different legislative sources;
- the citation with which the document to be modified is cited, that expresses the legislative reference (also a textual reference), a fundamental element of the amending provisions.

On the basis of the three aspects mentioned here, we have endeavoured to define and describe the qualifying elements of the amendment provision as detailed in the De Lorenzo's assay [9] and in the authors' previous works [11].

[1] Understood as the 'part of the provision that introduces the amendment': it contains the purview aimed at specifying the relationship (substitution or integration or abrogation) between the provision in force previously and that provided by the textual amendment. The new sub-section generally ends with a colon, followed by the textual amendment placed between inverted commas.

Once the elements making up the amending provision had been identified and described [11], we have been able to propose a classification based on two of the elements we believed to be particularly important: the action of amending and its object.

In particular, on the basis of the action of amending, a distinction can be made among the following: repeal, integration and substitution.

As far as the object is concerned, the amendment, instead, operates on either a part (supra-part, article, paragraph, etc.) or on a part of the legislative discourse. It is obvious that each of the identified actions can operate on both the object *"part"*, and on the object *"part of the discourse"*.

The model currently implemented does not take into account elements (both textual and extra-textual ones) that may influence the amendment action unfolded by the novella upon the modified text. In particular, we have assumed as default the period of being-in-force equals to the one of the modifying act, even if this does not prevent the system to be afterwards able to treat a different being-in-force period of the amendment disposition, with respect to the general one of the modifying act.

Instead, according to the normativistic approach previously introduced in section 2, the implemented system is not able to treat different effectiveness periods, because such elements are strictly dependent upon extra-textual aspects which are not included in the model[2].

4. Tools for Legislative Information Markup

For the UML modelling of Express Text Amendments, we need the extraction of semantic information for the recognition and markup of the amendment disposition, as well as its composing elements [11].

To locate the *novella* in the modified normative text, the extraction of structural information, for the markup of the partitions constituting the structure of the normative act (articles, commas, letters, and so on), is also needed, in order to generate some kind of mapping of the text.

For this last function we have used the structural analyzer implemented in the prototype of LexEdit XXI[3], while the extraction of semantic information, needed for the recognition of the amendment disposition, is provided by Sophia 2.1 Linguistic Parser by CELI of Turin, Italy[4]. Both tools will be briefly introduced in the following subsections.

---

[2] The difference between force and effectiveness is often exemplified [10] by the war criminal code: in peace time the code is certainly enforced but not effective. The effectiveness will depend from wartime declaration, but also from real situations not calculable by a representative model of legislative system because extraneous to the system itself. The war criminal code considers, in fact, cases of "automatic" application of war criminal law connected to some events, without the needs of a bill, order or other and, in particular, the war declaration which enforced, for general rule, the application of the war criminal code (art. 3 c.p.m.g.).

[3] One of the first legimatic software programs was LexEdit which, specifically, developed checking functions regarding the correctness of the text.

A new project about evaluating and checking the legislative text, is now underway at ITTIG (Istituto di Teorie e Tecniche dell'Informazione Giuridica) for basing the checking functions, typical of LexEdit, on those of a parser making the recognition of the legislative language much more efficacious. The first release of the project is the prototype LexEdit XXI [5].

[4] See http://www.celi.it/sophia.htm

4.1. Sophia

The parsing system Sophia 2.1 has been identified as a suitable tool for the recognition and tagging of the amendment provisions[5].

Sophia uses the techniques of finite state automata and finite state transducer, which make this software flexible and configurable, thus enabling rules and specific models (already defined or in the course of definition) to be formalised. In particular, we are working with this software on analysing and tagging the first sample of legislative instruments in the following phases:

- normalisation of the entry text, properly tagging all those structures and textual segments that can be recognised on the basis of characters or, in other words, without resort to or consultation of the lexicon-dictionary;
- lexical (syntactical category) and morphological (flexion passages) analysis of the text in input;
- disambiguation of the syntactical category of the words (*Part of Speech Tagging*);
- partial syntactical analysis (called *chunking*), aimed at identifying the minimum syntactical groups present in the text in input and at grouping them in constituents;
- semantic analysis and identification of the relevant conceptual structures in the text in input;
- conversion of the analysed document from the original format (Microsoft® Word, HTML, RTF, txt, etc.) into the XML format, according to the established DTD.

The compilation of the grammar in the syntax of the chosen parser takes place by using the semantic module of the system, through drafting legislative rules, that formalise the defined models of "novella", and permit the automated identification of the described linguistic structures and the information extraction.

4.2. The Zoniser of LexEdit XXI

LexEdit XXI is a tool composed by several modules. Among them, we have used for this analysis the **Zoniser** [5], whose purposes are:

- The identification of the structure of the document in terms of articles, paragraphs, etc.
- The production of a XML document in output according as far as possible to the DTD of NIR[6].
- To point out to the user any anomalies or decisions that cannot be made automatically.

The Zoniser receives as input a document in RTF format, which is validated according to the NIR DTD. After the validation step, the document is saved in a XML file containing the structural tagging; such output will be further exploited to build the UML representation of the processed normative act.

---

[5] Considered the general reliability of the linguistic parser, we have used one of these tools whose syntax we had already implemented with specific rules of the legal domain.

The most recent tools integrating the techniques of symbolic analysis (using dictionaries, grammars, thesaura, etc.) with the statistics analysis algorithms seem to distinguish themselves for the greater speed and efficiency; for the experimental and methodological goals of the proposed research, we have assessed the reliability a decisive characteristic.

[6] See the *Norme In Rete* (Law On The Net) *NIR* project: http://www.normeinrete.it

5. UML Model for Normative Acts

Once defined the scope and the target of this work, we need to formalize the subject of the analysis through the use of UML language; in other words, we need to define a model expressing both the structural aspects of a legal text (articles, commas, etc.) and the semantic ones (that is, the amendment information a novella applies upon an existing legislative text).

Structural aspects have been defined taking into account the NIR DTD in its loose version [12,13] and the information of structural correction provided in LexEdit XML output. The determined elements and their associations are represented in the diagram of Figure 1. According to the adopted DTD, all the elements belong to the NIR package: the main Class is *Legge* (Law) which includes several attributes, recording the identifiers of the normative act, as well as emission, publication, and the enforced date. Such class can be connected to a unique *Articolato* (Articles) class, linked on its own to one or more *AbstractPartition* classes (a generic partition of the Articles). The Articles class has been modelled to conform to the NIR DTD. A partition can be one of the sections included in the Articles, such as *Libro* (Book), *Parte* (Part), and so on. All of them are identified by a *num* attribute (the index of the partition, inherited by *AbstractPartition*) and can be associated to Articles.

Finally, each kind of partition may include other partitions: this is needed to model the nesting of the legislative structure (e.g. a *Titolo* is composed by one or more *Articolo,* which may include several *Commas*, which may include letter elements *EL*).

Moreover, according to the loose version of the NIR DTD, the proposed model allows also for the validation of structures that can be considered as incorrect, referring to the most recent rules of legislative drafting; for example, instances of *Sezione* could be used as subpartitions of an article, or different kinds of partitions could be combined together. This makes the model more flexible and able to treat also normative text preceding the adoption of the NIR DTD.

In the same way, text components have been modelled using a generic *Elemento* class, (which can be, for example a *Rubrica*) containing in a text attribute the body of the element.

The model previously described represents the formal structure shared by the legislative texts. Express Text Amendments extends the structural model with the elements defined in De Lorenzo's classification of ETAs: according to this definition, two other kinds of partition have been added to model *integrations*, *substitutions* and *repeals*: *IntegrazionePartizione* and *IntegrazioneStringa* have been included to model the integration or substitution respectively of an entire partition, and of single words or sentences; similarly, *AbrogazionePartizione* and *AbrogazioneStringa* have been included to model the corresponding repeal actions. Inside these classes, suitable text attributes identify, where needed, the parts of text to be modified and the modifying ones. The *qualifier* attribute is used to locate the position inside the partition where the amendment will be applied to. The *Riferimento* (Reference) class is used to detect the partition which the amendment provision will be applied to. Finally, instances of the *Change* class are added in the model describing the text in force, to take into account all the amendments applied over a period of time. The needed information to instantiate the previous additional elements is extracted from Sophia XML markup; the obtained UML elements are added to the structural model of the ETA, by linking each amendment to the partition including it.

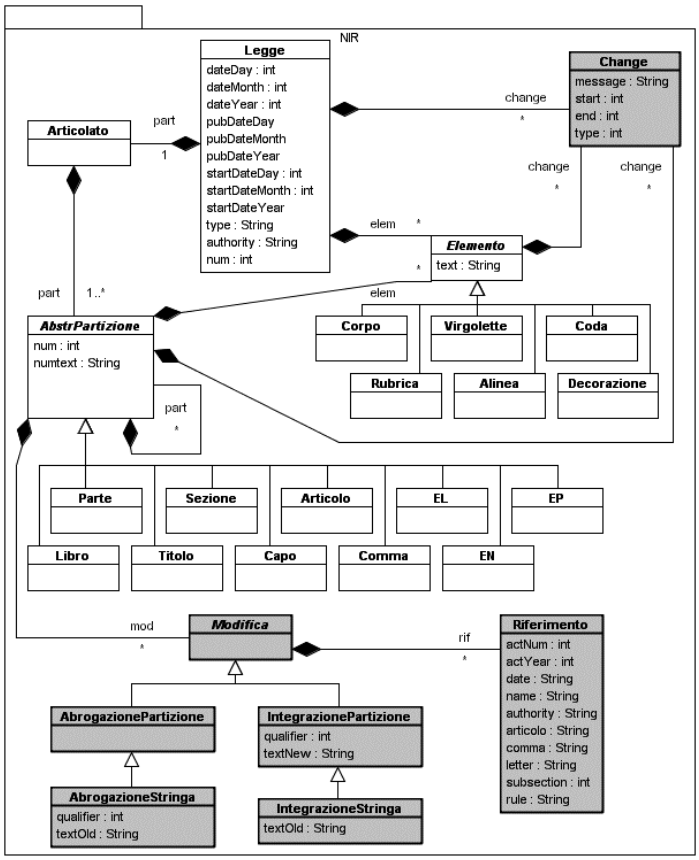

Figure 1. UML model of a legislative act with structural (in white) and amendments elements (in grey).

6. LexLooter Plug-in

LexLooter is a plug-in designed for the Eclipse 3.0 framework which is able to generate a UML representation of a legislative document, both in the original version and after the application of Express Text Amendments (ETAs), using the NIR DTD rules for the extraction of the articles structure from the legal text, and the model of the *novella* detailed in section 3.

Related to a given text of a normative instrument, LexLooter receives two kinds of XML input documents: the original legislative text with the XML markup of articles structure introduced by LexEdit XXI, and the XML files of Express Text Amendments, one including LexEdit articles structure of the novella, and another one with the XML tagging of amendment actions provided by Sophia. To generate the UML representation of a legislative document, LexLooter executes a sequential parsing of the marked up text: for each of the structural tags found, an instance of a suitable UML object is generated in the model (e.g. for each <ARTICOLO> tag, an instance of the UML class "Articolo" is generated); while the text embodied between corresponding tags is used as value for the attribute belonging to the object identified by the tag (e.g. the text corresponding to a <RUBRICA>).

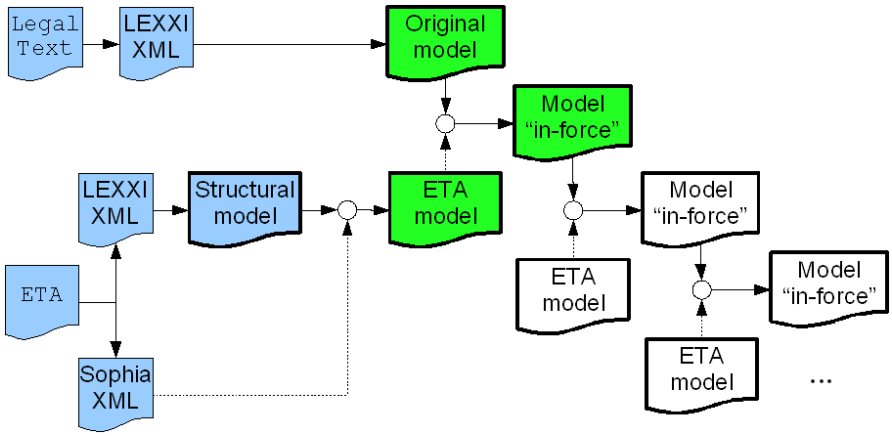

Figure 2. LexLooter processing chain of original law and ETA text.

A UML representation of Express Text Amendments can also be generated using LexLooter; in this case, the model is generated by exploiting also the text of *novella* processed by Sophia, whose markup allows to locate and extract all the amendment actions.

Once obtained the UML model of a *novella*, the model and the corresponding text of the original normative instrument can be updated by applying all the amendment actions previously extracted; the current law-in-force model and text can be finally obtained if applying iteratively each ETA to the original text of the law (Figure 2).

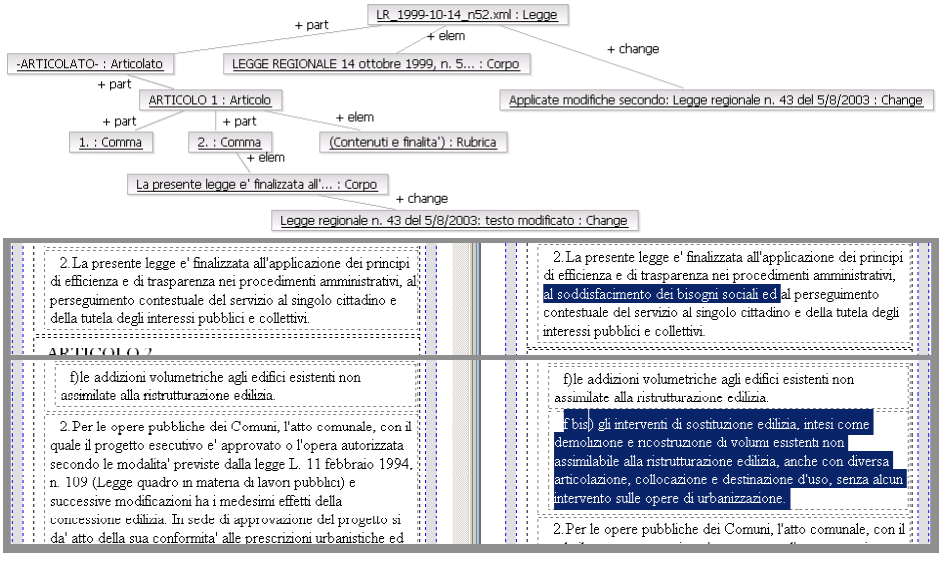

Figure 3. Coordinated UML diagram and text produced by LexLooter.

Figure 3 shows a partial UML diagram of the coordinated model and text produced by LexLooter: the UML object diagram represented in the upper part of the figure has been generated by instantiating the elements composing the structure of the original text; amended elements are associated to one or more c*hange* elements which point out the kind of applied modification.

More in detail, the lower part highlights the coordinated text after the application of an *insertion* amendment action (right) to Comma 2, shown in the diagram, compared with the original text (left). The availability of a UML object diagram of the amended text, combined with the generation of the coordinated text, allows for a very easy text consultation and a valid tool to verify or simulate the effects of the amendments.

LexLooter allows also for the generation of a Gantt diagram to trace the periods of enforcing of the normative instrument after the application of each ETA.

Table 1. Results of ETAs modelled by LexLooter applied to L.R. 52/1999.

L.R. 52/1999 (2 Novellas applied)	*Express Textual Amendments*					
	Marked by Sophia			*Applied by LexLooter*		
	Str. Repeal Part. Repeal	*Str. Subst. Part. Subst.*	*Str. Ins. Part. Ins.*	*Str. Repeal Part. Repeal*	*Str. Subst. Part. Subst.*	*Str. Ins. Part. Ins.*
Number of modelled ETAs	***42***			***39***		
	0	9	4	0	9	2
	9	15	5	9	15	4

Table 1 shows some figures related to a significant example of LexLooter application to a normative text (namely, Regional Tuscany Statute 52/1999), and to its 2 corresponding ETAs: the figures point out that the number of ETAs modelled in the overall by the tool (39) correspond to the 92% of the ETAs that has been marked by the parser Sophia (42).

Two cases have been currently experienced for missing modelling:

- Typing errors in the text of the *Novella*, which result in incorrect extractions of the textual amendments. Current implementation normalizes to ASCII characters the text to be extracted, in order to optimize the matching with the set of adopted regular expressions; these will also be further refined in next releases to improve amendments extractions.
- Missing LexEdit XXI structural markup of text inserted by an amendment, which appears commonly quoted in an ETA; this prevents LexEdit to recognize possible added partitions (i.e. one comma substituted by more commas); the missing structural markup of the modifying text results in the generation of an incomplete coordinated model. Next releases will include a better integration with the structural marker, to generate a full model of a coordinated text.

Moreover, we have also experienced a number of amendments that have resulted not marked by Sophia: though able to estimate such amendments (via suitable warning messages indicating the number of occurrences in the text of modification actions outside Sophia markups), LexLooter cannot model them. This is due to the fact that Sophia can mark only *well-formed* amendments, accordingly to the modern drafting rules that have been translated in the extraction model of the parser; by loosing such rules, Sophia will be able to mark also *bad-formed* amendments, thus allowing LexLooter to include them in a coordinated model. We also provide for the implementation in LexLooter of suitable verification algorithms in the coordinated model for the compliance of the amendments to the legislative drafting, either by the validation of the structure, if consistent with structural constraints (expressed with OCL) or with the insertion in the Change class of correctness attributes highlighting possible *bad-formed* amendments.

7. Work Perspectives

Though still at a prototypal stage, LexLooter is a new step based on previous authors' work about formalizing normative text, because it has shown the capabilities of the UML modelling to represent in an immediate way the relations between the elements inside a normative act. Provided the descriptive capabilities of the model are suitably extended, the tool can be employed for the reconstruction of the normative coordinated text by exploiting the particularly comprehensible and flexible UML graphical formalism.

Current work is in the direction of applying the methodology implemented in LexLooter to a larger corpus of normative acts, in order to verify with an extensive experimentation the efficiency of the system and the use cases.

This methodology for the representation of the normative text also opens up other interesting applications, in particular for the verification of the correctness, consistency and accuracy of the normative speech and of its composing elements. Even if we have referred to a model that can be defined as normativistic, we can however suppose a possible use of the system to support the interpretation, in order to highlight what can be defined as "critical points" of the text (ambiguity, contradictions, etc.), where the interpreter has to make the most difficult choices. The UML notation also fits for the formalization of those extra-textual elements, which, in combination with the textual ones, provide a substantially complete set of information which the interpretation of the legal rule is based on; therefore, further research perspectives arise, relating to the representation of the real world into formalizable models; though fascinating, they currently take no part into this study.

Therefore, such methodology can be adopted for the implementation of suitable tools for supporting the generation, evaluation and consultation of normative texts.

Further extensions of this work could be in the direction of the application of the methodology to administrative documents, as well as any other acts where a formal model of the structure can be defined.

References

[1] G. Booch, J. Rumbaugh, I. Jacobson, *The Unified Modeling Language User Guide*. Addison Wesley, Sept. 1998.

[2] J. Rumbaugh, I. Jacobson and G. Booch, *The Unified Modeling Language Reference Manual, 2nd Ed*. Addison Wesley, Jul. 2004.

[3] T. M. van Engers and E. Glassèe, *Facilitating the legislation process using a shared conceptual model*. IEEE Intelligent Systems, vol. 16(1), Jan. 2001, pp. 50-58.

[4] A. Bolioli, L. Dini, P. Mercatali and F. Romano, *For the Automated Mark-up of Italian Legislative Texts in XML*. In "Legal Knowledge and Information Systems. JURIX 2002: The 15th Annual Conference". IOS Press, 2002.

[5] P. Mercatali, F. Romano, *Automatic legislation evaluation*. In Proc. of ICAIL'05, Bologna, Italy, Jun. 6, 2005.

[6] G. Carcaterra, *Metodologia giuridica*. In *Corso di studi superiori legislativi*, Cedam, Padova, Italy, 1990.

[7] M. Palmirani, R. Brighi, *Norma-System: A Legal Document System for Managing Consolidated Acts Database and Expert Systems Applications*. In Proc. of DEXA 2002, Aix-en-Provence, France, Sept. 2-6, 2002.

[8] G. Sartor, *Riferimenti normativi e dinamica dei nessi normativi*. In *Il procedimento normativo regionale*, Cedam, Padova, Italy, 1996.

[9] M.C. De Lorenzo, *Modelli di novelle*. In *Informatica e diritto*, vol.1, 2002.

[10] G.U. Rescigno, *L'atto normativo*. Giappichelli, Torino, Italy,1998.

[11] A. Bolioli, L. Dini, P. Mercatali and F. Romano, *Models of "Novelle" and Normative Grammars*. In "Legal Knowledge and Information Systems. JURIX 2004: The 17th Annual Conference", IOS Press, Amsterdam, 2004.

[12] C. Biagioli, E. Francesconi, P.L. Spinosa and M. Taddei, *The NIR Project. Standards and tools for legislative drafting and legal document Web publication*. In Proc. of the Int. Conf. of Artificial Intelligence and Law, Edinburgh, Jun. 24, 2003.

[13] A. Marchetti, F. Megale, E. Seta and F. Vitali, *Marcatura XML degli atti normativi italiani. I DTD di Norme in Rete*. In *Informatica e diritto*, vol.1, 2001, pp. 123-148.