BO-ECLI Parser Engine: the Extensible European Solution for the Automatic Extraction of Legal Links

Tommaso Agnoloni
Institute of Legal Information Theory and Techniques
ITTIG-CNR
Firenze, Italy
tommaso.agnoloni@ittig.cnr.it

Lorenzo Bacci
Institute of Legal Information Theory and Techniques
ITTIG-CNR
Firenze, Italy
lorenzo.bacci@ittig.cnr.it

Marc van Opijnen
Publications Office of the Netherlands
UBR|KOOP
The Hague, the Netherlands
marc.opijnen@koop.overheid.nl

ABSTRACT

This paper presents the BO-ECLI Parser Engine, an open source Java framework for the automatic extraction of case-law and legislation references from case-law texts issued in the European context.

Differences of languages and jurisdictions are tackled with an extensible design that guides and facilitates the development of pluggable national extensions, resulting in a considerably reduced effort with respect to the development of a full national legal link extractor from scratch.

Thanks to a well-defined pipeline of services that synthesize the whole extraction process and to an internal annotation system that is used to convey the information along the pipeline, the software ensures both overall efficiency and flexibility in absolving language and jurisdiction dependent tasks.

Services can be provided either by the common part of the software or by a national extension. For the implementation of services performing rule-based textual analysis (like entity identification), JFlex is used in the common part and recommended in the national extensions. Finally, through identifier generation services, the BO-ECLI Parser Engine can produce standard identifiers, like ECLI or CELEX, for each recognized legal reference.

Starting from a Template project, two different national extensions have been successfully developed and tested in order to support the extraction of legal links from case-law texts written in the Italian and Spanish languages.

KEYWORDS

Legal citations, Reference parsing, Multilinguality

1 INTRODUCTION

Among the goals of the European Case Law Identifier (ECLI) established in 2010[1] is the publication of national case-law by courts of European member States via the ECLI Search Engine on the European e-Justice Portal. Besides being uniformly identified, decisions should be equipped with a minimal set of structured metadata describing their main features. Among the (optional) metadata prescribed by the ECLI Metadata Scheme, *references* metadata describe relations of the current document with other legal (legislative or judicial) documents, formally expressed using uniform identifiers (the aforementioned ECLI for case-law, ELI for legislation, national identifiers, CELEX identifiers for European legal sources).

These relational metadata are at the same time among the most useful case-law metadata, in that they allow the enhancement of legal information retrieval with relational search, and among the most difficult to have valued, especially for legacy data and for less resourced languages and jurisdictions.

In the legal domain, citations are an integral part of a text and their instrumental use for a variety of purposes (substantive, procedural, argumentative) is a familiar tool for legal professionals performing their daily duties. Search by relationship [1] is therefore popular among users as it conforms with the typical attitude of legal professionals confronted with the reconstruction of the sources relevant for a legal issue at hand. Nonetheless it is poorly supported by typical search engines relying on full text indexing and necessarily requires an explicit reference tagging to be dealt with by machines.

Manual reference tagging is an extremely costly procedure, not viable in the public domain and especially unable to cope with the growing amount of data published in national case law databases. Automatic legal reference extraction, on the other hand, has been successfully applied in several national contexts [2], [3], [4], despite the complexity of coping with a diversity of styles, variants and exceptions to existing drafting rules and citation guidelines.

Due to the national specificities, national citation practices, and language dependency of the task, the problem scales in complexity when approaching it from a multilingual and multi-jurisdictional perspective. Previous efforts in such direction within the EU-funded EUCases project [5] was limited to the extraction of references from national case law to EU legal sources.

Starting from an analysis of approaches and existing solutions to the "Linking data" problem [6] and based on the results of a survey on citation practices within EU and national Member States' courts [7], the BO-ECLI Parser Engine presented in this work and developed within the EU funded project "Building on ECLI"[2], tackles the problem from an EU-wide multilingual and multi-jurisdictional perspective. The aim of the proposed framework is to lower the entry barrier for national data providers willing to develop their own legal reference extraction solution by providing a proven methodology, shared common knowledge and reusable and extendable components.

[1] Council conclusions inviting the introduction of the European Case Law Identifier (ECLI) and a minimum set of uniform metadata for case law (CELEX:52011XG0429(01)).

[2] http://bo-ecli.eu

In: Proceedings of the Second Workshop on Automated Semantic Analysis of Information in Legal Text (ASAIL 2017), June 16, 2017, London, UK.

Published at http://ceur-ws.org

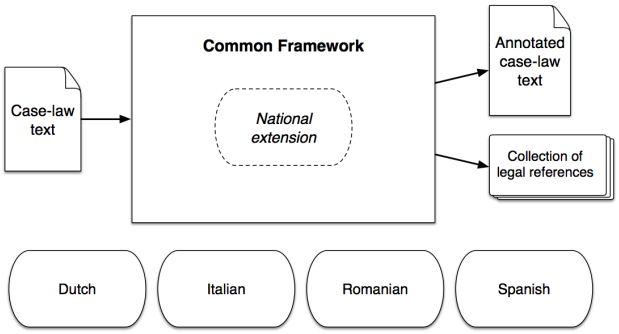

Figure 1: The overall architecture of the BO-ECLI Parser Engine: a common framework with pluggable extensions for supporting the parsing process of a text written in a specific language or issued within a specific jurisdiction in order to get a collection of legal references and the original text with inline annotations.

2 PARSER ENGINE

The BO-ECLI Parser Engine is an extensible framework for the extraction of legal links from case-law texts. It is written in Java and distributed as open source software[3]. It targets citations to both case-law and legislation, expressed as lists of textual features (authority, type of document, document number, date, etc.) or as common names (i.e. aliases). Multiple citations, intended either as citations to more than one partition of a single document or as citations to more than one document issued by a single authority, are also covered and distinct legal references are generated in correspondence to each partition and each document. A distinguishable characteristic of the software consists in the capability to be extended in order to support the extraction process from texts written in different languages or issued within different jurisdictions.
In order to realize such design, two practical steps are required:

- dividing the process of legal link extraction into a generic and customizable sequence of atomic services, following a pipeline pattern;
- defining an annotation system able to convey the work done by each service along the pipeline.

Distributed as Java Libraries with a standard Java API, the software can either be integrated within an existing environment for an automatic batch parsing over a large corpus or be wrapped into an even more interoperable HTTP API and queried by a remote user-interface. Especially for this use case, the overall efficiency of the software guarantees a quick response (in terms of user experience) even with large case-law texts as input.

While the input of the software is as simple as text and additional metadata, the result of the parsing process consists in a collection of legal reference objects, possibly accompanied with legal identifiers, and in the original text with added inline annotations, possibly with hyperlinks, in correspondence with the recognized citations.

[3]http://gitlab.com/BO-ECLI/Engine

3 A PIPELINE OF SERVICES

One way to synthesize a generic process of legal link extraction from texts is, first, to divide it into three consecutive phases:

(1) the entity identification phase, where the fragments of text that can potentially represent a feature of a citation are identified and normalized;
(2) the reference recognition phase, where patterns of identified features are read in order to decide whether they form a legal reference or not;
(3) the identifier generation phase, where the recognized legal references are analyzed so that standard identifiers, and possibly URLs, can be assigned to them.

Secondly, within every single phase, a number of different services can be placed, each specialized in performing one task. For example, within the entity identification phase, there could be a service specialized in the identification of case numbers.

By modelling the process of legal link extraction with a sequence of services that belong to these three distinct phases it is possible to concretely achieve a separation in design between a common part and an extension part. Specifically, the common part (i.e. the framework) defines the classes, the interfaces and the methods that guide the implementation of any specific task and provides the default implementations for services common to every jurisdiction, like the generation of a *CELEX* identifier for legal references to European legislation. On the other hand, the extension part must provide the implementations for services like the identification of national issuing authorities, a strictly language dependent task.

In the Java domain the described separation between the common framework and the national extensions is realized through the Service Provider Interface paradigm, which is part of standard Java and its adoption is straightforward. Following SPI, the integration among the framework and all the different national extensions is as simple as publishing them as standard Java *jar* libraries and making them visible in the *classpath*.

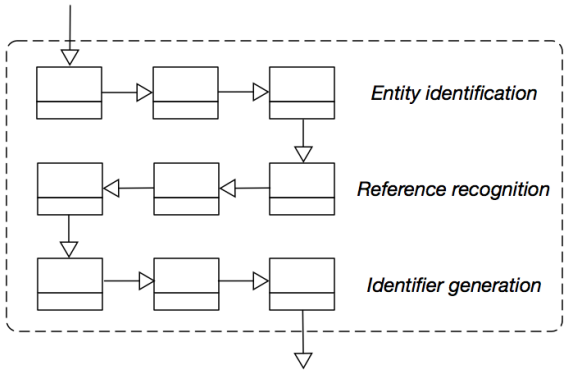

Figure 2: A schematic representation of the execution of a sequence of atomic tasks through a pipeline of service implementations belonging to the three different phases of the parsing process.

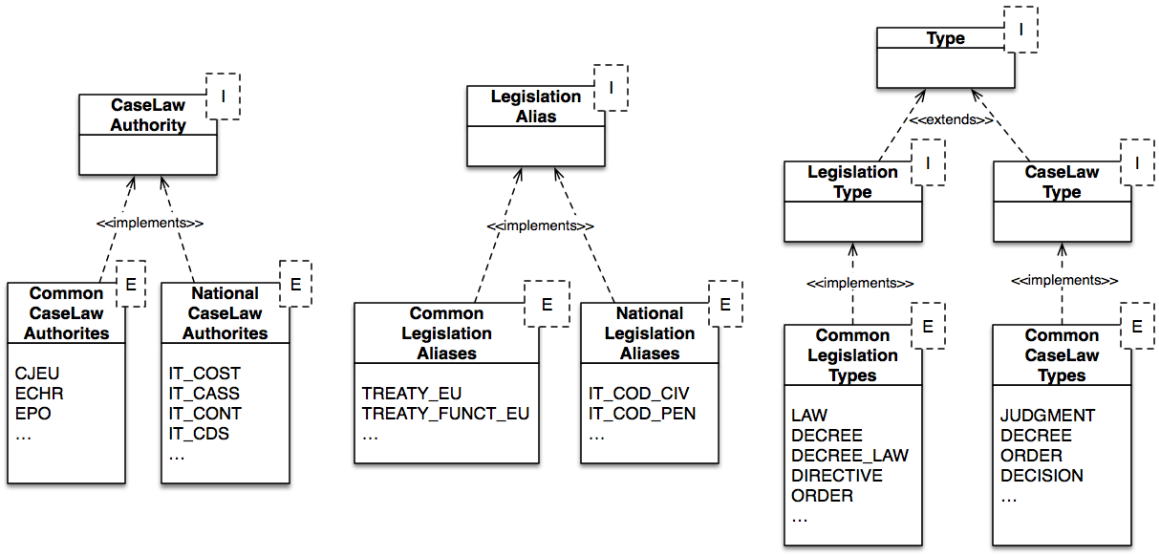

Figure 3: Empty interfaces representing annotation categories like authorities, aliases and types of document are defined in the framework and are implemented by Java Enumerations provided by the national extensions or by the framework itself, thus realizing extensible lists of normalized values for annotations that share a common Java type.

4 ANNOTATION SYSTEM

The BO-ECLI Parser Engine framework defines an internal annotation system to allow every service implementation, especially the ones belonging to entity identification and reference recognition, to save the specific results of their execution directly in the text. Thus, every service can be seen as a module that receives an annotated text as input and produces an annotated text as output, enabling the complete customizability of the pipeline: depending on the language, jurisdiction or other specific metadata of the input text, modules can be replaced, enabled, disabled or arranged differently. Within the BO-ECLI Parser Engine, annotations are used to assign a *category* (hence, a meaning) to a fragment of text, while, through normalization, annotated fragments of text can acquire a language independent *value*. For example, the Italian fragment of text *"sent. della Corte Costituzionale"*, meaning a judgment issued by the Italian Constitutional Court, at a certain point along the pipeline, is annotated as follows:

```
[BOECLI:CASELAW_TYPE:JUDGMENT]sent.[/BOECLI] della
[BOECLI:CASELAW_AUTHORITY:IT_COST]Corte Costituzionale[/BOECLI]
```

The controlled lists for the annotation categories and annotation values can be provided either by the common framework or by the national extensions through Java Enumerations. Thanks to the annotation system, the work of each service is conveyed and shared along the pipeline in a language independent way.

4.1 Auxiliary methods

One of the peculiarities of the framework is to provide the implementor with a number of *auxiliary methods* that are used to produce all the annotations in a transparent way.

For each annotation category during the entity identification phase, by passing a fragment of the input text and optionally a normalized value as argument, an auxiliary method appends to the output the syntactically correct annotation for such fragment of text. By exploiting the auxiliary methods, the implementor is completely guided in producing a syntactically correct annotation with the allowed values for each annotation category.

Reference recognition service implementations can also benefit from specific auxiliary methods that are responsible for converting the annotations of several entities forming a valid pattern into a unique legal reference annotation.

5 SERVICE IMPLEMENTATION

The implementation of an annotation service belonging to either the entity identification or the reference recognition phase simply consists in a piece of software that analyzes an input text, possibly already enriched with annotations, and produces an equivalent output text, possibly with altered annotations. Since the framework provides the methods for accessing the input text and for producing the output text, it is up to the national implementor to decide, for each service implementation, the textual tools to be used in order to perform the matching. For example, those operations could be realized with the methods of the Java String class, a set of regular expressions and the Java regex package, proper lexical scanners, i.e. automata.

The last approach is not only the most powerful and efficient, but also the most fitting for the implementation of an annotation service. The default implementations of the annotation services provided by the framework make use of JFlex[4], a well-known lexical scanner generator for Java. Since the JFlex syntax allows for the insertion of Java code, a *jflex* file can directly make use of the auxiliary methods and Java Enumerations supplied by the framework and by the national libraries.

[4]http://jflex.de

5.1 Default service implementations

A number of implementations for services that belong to each phase of the legal link extraction process are provided by the framework by default. Typically, a default implementation is supplied when the task that the service is in charge of can be considered language independent, pertains to the European jurisdiction or is common in the European context.

5.1.1 ECLI identification. An ECLI code can be used within a case-law text as a feature of a more complete citation or as a citation by itself. Since the ECLI code has a standardized syntax that doesn't depend on the language used in the rest of the case-law text, a default service for ECLI identification is implemented and supplied by the framework.

5.1.2 Partitions identification. A partition is a hierarchical branch of partition elements like articles, paragraphs, letters, etc. While the identification of each partition element is a language dependent task, the framework provides a service that converts sequences of partition element annotations into unique partition annotations, correctly composing the branches, especially in case of multiple citations.

5.1.3 Parties identification. The identification of the names of the parties in a citation should be generally considered as a language dependent task. Nonetheless, the framework provides a default service implementation for the identification of applicants and defendants relying on heuristics based on positioning, upper and lower casing, the *versus* entity and the geographic identification of a country member of the Council of Europe (as a defendant in European Court for Human Rights citations).

5.1.4 Reference recognition. After the entity identification phase, the textual features that can potentially be part of a legal reference are annotated and normalized, hence they can be treated as language independent entities. Although citation practices change from one jurisdiction to another, the framework provides a number of default service implementations for reference recognition that are able to cover the most typical citation patterns and, also, to support multiple citations.

5.1.5 ECLI generation for European Courts. In those cases where a standard identifier can be simply generated as a composition of the features extracted from the textual citation, the framework provides a default service implementation to automatically assign an identifier to a legal reference. This is the case for the generation of ECLI for legal references that have the European Court of Human Rights as the issuing authority, when the type of document, the case number and the date are known.

5.1.6 CELEX generation for European legislation. Another service implementation supplied by the framework for the automatic composition of a standard identifier is used for legislation references to European directives and regulations. For these types of document, when the referred document number and year are known, a CELEX identifier as well as its ELI identifier can be assigned to the legal reference.

5.1.7 CELEX generation for European aliases. The framework provides a controlled list of values of the main aliases pertaining to the European primary legislation, like the *Treaty on European Union* or the *Treaty on the Functioning of the European Union*, so that a national implementor, by reusing such normalized values, is guided in the implementation of an identification service that covers those references expressed in his specific language. Moreover, within the framework, a default service implementation for identifier generation automatically assigns, for each registered alias value pertaining to the European legislation, the correct CELEX identifier.

5.1.8 HTML rendering. At the end of the pipeline of services, all temporary annotations are discarded and the original input text is only annotated with the "legal reference" annotation category in correspondence of the recognized references. A national implementor can develop a rendering service in order to convert the internal annotation of legal references to a specific format of his choice. By default, the framework provides an HTML style rendering service that transforms the "legal reference" annotation category into the *<a>* tag and uses the optional URL assigned during the identifier generation phase as the value of the *href* attribute.

6 NATIONAL EXTENSIONS

National extensions are used by the framework to allow the extraction process from texts written in specific languages or issued within specific jurisdictions. In order to develop a new extension, the implementor has to:

- extend the controlled lists of normalized values for the annotations with the values pertaining to the new jurisdiction;
- provide a certain number of service implementations for entity identification, reference recognition and identifier generation;
- export the project as a Java *jar* library for compatibility with the Service Provider Interface.

6.1 Template

Along with the framework project, a Template project has been developed in order to facilitate and encourage the adoption of the software for the extraction of legal links in new languages and jurisdictions. The Template project provides a national implementor with:

- a complete Java project with organized packaging;
- a configuration file for setting general parameters like the author, language and jurisdiction of the extension;
- plain files of reusable *macros* of regular expressions to facilitate the parsing of the annotations and to set up common language dependent expressions;
- several Java Enumerations extending the controlled lists of normalized values for annotations with customizable constants;
- a full pipeline of services pertaining to the different phases of the legal links extraction process with dummy implementations in *jflex*.

6.2 The Italian and Spanish extensions

So far, following the Template project, two national extensions have been successfully developed for allowing the extraction of case-law and legislation references from case-law texts in the Italian and Spanish languages. This section contains some brief considerations concerning the development of such extensions.

6.2.1 Incremental annotations. The design of the BO-ECLI Parser Engine, composed by a sequence of entity identification services in charge of atomic tasks and accompanied with an incremental annotation system, makes the identification or disambiguation of complex entities possible, while keeping the code separated, readable and upgradable. For example, it has been possible to correctly annotate the Italian Administrative Regional Tribunals in all their textual variants as well as the Spanish Provincial Court of Appeals, through the execution of distinct services responsible for the identification of geographic entities, followed by the identification of sections and then the identification of local courts:

```
1) T.A.R. Sezione distaccata di Latina
2) T.A.R. Sezione distaccata di [BOECLI:GEO:IT_LT]
   Latina[/BOECLI]
3) T.A.R. [BOECLI:SECTION:IT_LT]Sezione distaccata
   di Latina[/BOECLI]
4) [BOECLI:CASELAW_AUTHORITY:IT_TARLT]T.A.R.
   Sezione distaccata di Latina[/BOECLI]
```

The decoupling between annotation tasks hugely reduces the effort needed for covering all the possible linguistic variants of such complex entities since regular expressions are based on previous annotation *values* that come from controlled lists, rather than on their textual variants.

In the example above, the fragment of Italian text is correctly annotated as a "case-law authority" with the value *IT_TARLT*.

6.2.2 Custom identifier generation. For both extensions it has been possible to develop custom identifier generation services based on a composition of the features of the legal references.

Within the Italian extension, a service implementation is in charge of the automatic generation of the ECLI for case-law references of high courts (Constitutional Court, Supreme Court, Council of State and Court of Auditors) and another one is responsible for composing the national URN-NIR identifier for references to legislation, while in the Spanish extension the ECLI can be automatically produced when the ROJ (a Spanish identifier) is present in the citation and hence among the features of the case-law reference.

6.2.3 Vocabulary extension. For both the extensions, the lists of controlled values have been effectively extended by customizing the Java Enumerations provided by the Template project. National case-law authorities values are expressed following the ECLI convention for Court codes and include high courts as well as regional and local courts, while aliases values for national legislation include, for example, civil and penal codes.

6.2.4 Pipeline. The pipeline included in the Template project, that provides a number of suggested service implementations as well as a default order of execution, has been used in both the extensions and it has proven effective with only minor adjustments.

7 CONCLUSIONS

We presented the BO-ECLI Parser Engine, an open source framework for the automatic extraction of case-law and legislation references from case-law texts issued in the European context.

Through a detailed description of its architecture and design, the paper showed how national extensions can be developed and plugged within the framework in order to add support for the extraction process to different languages and jurisdictions. Specifically, thanks to a decomposition of the whole process into atomic tasks and to an internal annotation system, the framework is able to provide a number of common services and resources that can be reused and extended by the national implementors.

By defining and providing a complete stack for legal links extraction, the implementation of a national extension is guided and straightforward, and the effort needed for the development of a fully functional national extractor is considerably reduced.

Along with the common framework, a Template project was also created in order to encourage and facilitate the development of new national extensions. Moreover, the paper presents two concrete national extensions that have already been developed by different teams, proving both the feasibility and the straightforwardness of the whole approach.

The BOECLI Parser Engine, as well as the Template and the Italian and Spanish extensions, are open source projects. Their code and documentation are currently hosted on the GitLab software development platform[5].

ACKNOWLEDGEMENT

This publication has been produced with the financial support of the Justice Programme of the European Union. The contents of this publication are the sole responsibility of the authors and can in no way be taken to reflect the views of the European Commission.

REFERENCES

[1] Marc van Opijnen and Cristiana Santos. On the concept of relevance in legal information retrieval. *Artificial Intelligence and Law*, 25(1):65–87, 2017.

[2] Lorenzo Bacci, Enrico Francesconi, and MariaTeresa Sagri. A proposal for introducing the ecli standard in the italian judicial documentary system. In *Proceedings of the 2013 Conference on Legal Knowledge and Information Systems: JURIX 2013: The Twenty-sixth Annual Conference*, pages 49–58. IOS Press, Amsterdam (NL), 2013.

[3] Marc van Opijnen, Nico Verwer, and Jan Meijer. Beyond the experiment: The extendable legal link extractor. In *Workshop on Automated Detection, Extraction and Analysis of Semantic Information in Legal Texts*, June 8-12 2015. held in conjunction with the 2015 International Conference on Artificial Intelligence and Law (ICAIL) San Diego, CA, USA. Available at SSRN: https://ssrn.com/abstract=2626521.

[4] A. Mowbray, P. Chung, and G. Greenleaf. A free access, automated law citator with international scope: the lawcite project. *European Journal of Law and Technology*, 7(3), 2016. Available at: http://ejlt.org/article/view/496/691.

[5] Pavel Popov, Alexander Konstantinov, Hristo Konstantinov, and Livio Robaldo. *EUCases project Deliverable D3.6, Report on Linking Tools'*. 2014. Available at: http://eucases.eu/fileadmin/eucases/documents/eucases_d3.6_linkingtools_report_revised.pdf.

[6] Tommaso Agnoloni and Lorenzo Bacci. *BO-ECLI project deliverable D2.1 Linking Data - analysis and existing solutions.* 2016. Available at: http://bo-ecli.eu/uploads/deliverables/DeliverableWS2-D1.pdf.

[7] Marc van Opijnen, Ginevra Peruginelli, Eleni Kefali, and Monica Palmirani. *Online Publication of Court Decisions in the EU - Report of the Policy Group of the Project, 'Building on the European Case Law Identifier'*. 2017. Available at: http://bo-ecli.eu/uploads/deliverables/Deliverable%20WS0-D1.pdf.

[5] https://gitlab.com/BO-ECLI