Semantic Web 0 (0) 1
IOS Press

Tools for building an event-based knowledge graph from legal decisions

María Navas-Loro * and Víctor Rodríguez-Doncel
[a] *Ontology Engineering Group, Universidad Politécnica de Madrid, Madrid, Spain*
E-mails: mnavas@fi.upm.es, vrodriguez@fi.upm.es

Abstract. This paper describes a toolset to transform a legal decision in English language into a collection of events represented in RDF supported by an ontology. Two different sources for judgments have been used for demonstration: the European Court of Human Rights (ECHR) and the European Court of Justice (ECJ). Text documents, preferably structured, go through a pipeline where they are analyzed, annotated and finally ingested in a triple store that can be queried through an open SPARQL endpoint. A translation service permits transforming time related information from/to different formats. The related ontology is publicly available online, the source code is accesible in an open modality and a web portal demonstrates the toolset. The adoption of standards and the service-oriented architecture favor the interoperability and extensibility of this framework respectively. A set of predefined queries facilities retrieving information from the knowledge graph.

Keywords: Event, Temporal Expression, Legal Domain, Event-extraction, Event-based knowledge graph, Ontology

1. Introduction

Ludwig Wittgenstein opened his *Tractatus Logico-Philosophicus* observing that the world is the totality of facts, not of things –quite a reasonable observation for a logician who was interested in the truth of propositions at that time. When evaluating the events described in a legal decision, focusing on the events and their logical sequence seems also very reasonable and the storyline is of pivotal importance. This paper assumes that a judgment can be described as a series of time-marked happenings that we call *events* instead of focusing on the other entities (things).

This is not the first event-related knowledge graph. Event-Centric Knowledge Graphs were first formulated in 2016 [1] and have already been implemented in diverse domains, such as article processing [2], news [1] or even tourism [3]. In these cases, an Event-Centric Knowledge Graph (ECKG) is "*a Knowledge Graph in which all information is related to events through which the knowledge in the graph obtains a temporal dimension*" [1]. Differently to regular knowledge graphs, where the information usually gravitates around a number of central entities, ECKGs put the focus on specific events, retrieving information about them from different sources and combining it in order to properly describe them.

Differently to this approach, our aim is to describe legal decisions using the events as the basis, being *blocks* that describe the legal judgment. We consider a case to be a narrative of events in different dimensions, namely *procedural* or *relative to the case under judgment*, and that representing a case as a succession of events can be extremely useful for various applications within the legal domain. Since this concept is slightly different to the previous definition of what an ECKG is, we have decided to name our approach Event-Based Knowledge Graph, although both approaches share several common points, and tools presented could also be used to build an ECKG.

In the legal domain, several proposals have recently delved into building knowledge graphs [4], including initiatives such as the Lynx Project [5], that aims to build a multilingual knowledge graph to support compliance-related services. In spite of these recent efforts, none of them tackle event processing.

*Corresponding author. E-mail: mnavas@fi.upm.es.

Unlike previous related works, such as EventsKG [2], we have no previous structured knowledge bases in the domain in order to help us building our event-based knowledge graph, but only repositories with legal documents without annotations. Therefore, our first step had to be the retrieval and the processing of raw documents in order to extract relevant events from them. Although our approach focuses on events, as an Event-Centric Knowledge Graphs does, we do not understand events in the same way projects like EventKG did. We process and represent the relevant events (actions or happenings) mentioned in legal texts that shape the legal case, not events in the sense of a ceremony, a "named event" (like a specific war or regular sporting events such as the Olympic Games) or a journalistic event, with contributions from different sources. Even though eventually other types of resources could be integrated, such as news related to a case, or appeals to other courts such as nationals, we keep the focus on the events mentioned in a judgment. Our definition of an Event-Based Knowledge Graph would be therefore a Knowledge Graph where information is represented as a series of events, although additional information can be introduced, such as the annotations from which the events were derived.

Before undertaking the event extraction task, an analysis of the previous approach in the legal domain was carried out [6]. One of the suggestions made during the presentation of this work was to take into account the discourse extraction when dealing with events relevance. We have taken this into account, and it is further discussed in Subsection 3.1. Additionally, the differences detected among courts in the previous corpus temporal annotation work [7] led to the choice of some of them for implementation. This choice also invited to a first discourse analysis module dependant to the kind of document, that selects the relevant parts of the texts that the event extractor core will work on. In order to show our event extraction method is easily generalizable, we used two different sources to retrieve legal documents, namely the European Court of Human Rights (ECHR) and the European Court of Justice (ECJ), that allow to reuse their judgments in this context[1].

Once the events from the documents are extracted, they are translated to RDF format, using an ontology and a converter expressly created for this purpose. Finally, the document annotations with the events extracted are sent to the knowledge graph, that can be later queried. Taken into consideration that the legal domain practitioners are not usually familiar to semantic web technologies, we provide a service with a series of predefined queries in order to facilitate consulting the knowledge graph.

The primary contributions of this work are the following:

a) a service able to retrieve a document from European Courts, extract the relevant events in it and build a timeline, allowing humans to easily navigate through the document[2]
b) an ontology supporting the representation of temporal information, which eases the translation between time-related formats[3]
c) a converter that takes temporal annotations in various forms and outputs them as RDF[4]
d) an event-based knowledge graph of legal judgments in English that can be easily queried[5].

Additionally, in order to facilitate testing the interaction of these contributions, we have created a webpage that allows to test the pipeline step by step[6].

The paper is organized as follows. Section 2 explores previous related work in literature. Section 3 introduces our tool to extract events from European Courts and build a timeline with them. Section 4 presents the ontology built to represent events and temporal information, while Section 5 covers the translation tool we created to do the transition among formats. Section 6 presents the event-based knowledge graph generated from the previous tools, as well as possible exploitation options. Finally, Section 7 outlines the main contributions and the future research lines to explore.

[1]Documents provided by the European Court of Human Rights can be reproduced for private use or for the purposes of information and education in connection with the Court's activities when the source is indicated and the reproduction is free of charge (https://echr.coe.int/Pages/home.aspx?p=disclaimer&c=). The same policy applies to documents retrieved from EUR-Lex whose documents are allowed to be reused in conjunction with the Commission Decision of 12 December 2011 on the reuse of Commission Documents (https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32011D0833) for commercial and non-commercial purposes given the source is acknowledged (https://eur-lex.europa.eu/content/legal-notice/legal-notice.html#droits).

[2]https://fromtimetotime.linkeddata.es/whenthefact.html

[3]https://fromtimetotime.linkeddata.es/ontology.html

[4]https://fromtimetotime.linkeddata.es/service.html

[5]https://fromtimetotime.linkeddata.es/sparql.html

[6]https://fromtimetotime.linkeddata.es/pipeline.html

2. Related Work

Since the present work involves several tasks, this section is necessarily interdisciplinar. We will therefore present the revision of related literature in different parts.

2.1. Representation of Temporal Information

Representation of temporal information has been tackled in literature mainly in two ways: ontologies and annotation schemas. Ontologies, on the one hand, cover time-related information from a top approach. This is, they facilitate classes to represent different aspects relevant to temporal information, but do not tend to go deeper on each of their realizations in real world, handling just abstract information about them. Annotation schemas, on the other hand, tend to focus on detect appearances of certain predefined temporal information, such as event taxonomies and their arguments in texts. They therefore specify subtypes and expected arguments for each kind of event, admitting also other information per event instance, such as its probability or factuality. To summarize this idea, ontologies offer a more flexible and abstract representation option, while annotation schemas have a more strict and predefined target, oriented to an NLP task.

2.1.1. Annotation Schemas

Whereas most schemas and standards that have been proposed in literature are generic, describing temporal information without targeting an specific domain, some of them usually try to cover the needs of different tasks, focusing on different aspects and emphasizing the features that are required for an specific use case. Among all of them stands out the TimeML ISO standard [8], the most widespread time-focused mark up language for temporal annotation.

The TimeML standard covers different types of temporal information. Temporal Expressions, on the one hand, are "constructions referring to points or intervals on the timeline" [9]. They can be of type DATE (calendar dates or references), TIME (used for day times that are smaller than a day), DURATION (that denote the lasting of something) or SET (applied to repetitive time expressions), and TimeML marks them up using the TIMEX3 tag. Finally, events in TimeML can be expressed as verbs, nominalizations, adjectives, predicative clauses, or prepositional phrases. Additionally, the TimeML guidelines include relations and SIGNAL and MAKEINSTANCE tags.

Besides TimeML, among annotation standards we find for instance TIDES TIMEX2 [10], in which was partially based TimeML. Although there exist some corpora annotated with TIMEX2 tags, nowadays this format is no longer used. Other general purpose annotation standards can also be used to represent Temporal Expressions, such as the W3C Web Annotations[7], the NLP Interchange Format[8] (NIF) [11] or NLP Annotation Format (NAF)[9]. These formats are not specifically designed for temporal information representation, but they support data from NLP annotations. We can also find in literature extensions of TimeML for specific domain, such as the medical extension done for the THYME project [12].

Regarding events, the main source of related annotation schemas are the different challenges carried out in past years, and several analysis comparing them have been performed in literature[13, 14]. The ACE model [15] has been widely-used in previous literature, and focus on different types of events. On the other hand, ERE covers the annotation of entities, relations, and events, as well as their attributes, according to a taxonomy. There are two versions of ERE, named *Light* ERE and *Rich* ERE. *Light* ERE is basically a lighter version of ACE aimed to make annotation easier and more consistent [16]. These simplification includes for instance tagging just actually happening events or not including subtypes of entities. On the opposite, *Rich* ERE [17] expands *Light* ERE incorporating more types and subtypes of events (and re-classifying part of those in *Light* ERE), annotates also future, hypothetical or conditional events. On another note, the Knowledge Base Population (KBP) edition of 2017 included a Event Sequence task, aimed to retrieve the chronological order of events. KBP introduced the concept of "Event Nuggets", defined as a semantically meaningful unit that expresses the event in a sentence to annotate events in text [18], while in the following year edition this definition was redefined to "the smallest, contiguous extent of text (usually a word or phrase) that most saliently expresses the occurrence of an event" [19]. Besides the already exposed, there are also other proposals in literature, such as Richer Event Description (RED) [20], old challenges like MUC[21], or alternatives used to annotate corpora like OntoNotes[10] and FrameNet [22].

[7]https://www.w3.org/TR/annotation-model/
[8]http://persistence.uni-leipzig.org/nlp2rdf/ontologies/nif-core#
[9]https://github.com/newsreader/NAF
[10]https://catalog.ldc.upenn.edu/LDC2013T19

In addition to the previous efforts, usually focused on news annotation and covering a wide scope of events, we also find efforts targeting specific domains. In this line, the work by Sprugnoli et al. [23] presents annotation guidelines to mark up and classify flexible extents of historical events. There are also initiatives, such as the Open Event Date Alliance[11], that focus on parsing events from news sources and generate replicable data from them, instead of annotating them in text. Inside the wide universe of news-oriented event extraction or annotation, we also find more specific use cases, such as protest-event representation options like the CAMEO ontology [24], that are often based in previous approaches. Usually these efforts require full projects or PhD thesis to be properly analyzed and tackled [25]. Additionally, sometimes several ontologies are used at the same time in order to annotate events. This is the case for instance of the GAF annotation Framework for Events, that relies on the SEM ontology and TAF (TERENCE Annotation Format), the latter being at the same time based in TimeML and adapted to cover children's stories events.

To summarize, when dealing with event annotation we have two main approaches. The TimeML approach, on the one hand, is very linguistic oriented, and just links events in the text to other temporal information.This allows to cover all event mentions, but constraints the information that can be related to them. On the other hand, challenges such as ACE provide a series of templates for annotation, predefining the information to be found in the text, such as arguments or roles. This allows to store more information, but of course leaves a lot of not considered events aside.

2.1.2. Ontologies

Regarding ontologies mainly focused on time, very complete overviews of time-related ontologies, are provided in literature [26, 27]. In the following we will presented some of them that also cover event representation. There are of course more ontologies dealing with events, such as Model F [28], but due to the extensive literature on the topic we tried to analyze just the most related and well-known proposals.

The Time Ontology Recommendation[12] is the most well-known ontology for representing time, and provides the means for anchoring events in time. It represents dates, durations, intervals and temporal relations. Another well-known ontology is the Simple Event Model[13]. SEM is an ontology created to model events in various subject domains, such as history, cultural heritage, geography or multimedia. The four core classes in this ontology are *Event* (to record what happens), *Actor* (who or what participated in the event), *Place* (where did it happen), *Time* (when did it happen). Latter efforts in event representation have been done over SEM, such as the EventKG schema, used in EventKG, a multilingual event-centric temporal knowledge graph that incorporates over 690 thousand contemporary and historical events [2]. The Time Event Ontology, or TEO[14], is an ontology that allows to represent different temporal information for the purpose of further reasoning. [29]. As developed for the medical domain, event subclasses such as Clinical Intervention or Patient Accident are covered. Nevertheless, the time-related part of the ontology is in terms for temporal expressions. On another page, the Event Ontology and Timeline Ontology of Yves Raimond[15] provides a basic and flexible representation for a general event despite of being conceived in the frame of musical events. Finally, the Event and Implied Situation Ontology (ESO) [30] is a manually constructed resource which formalizes the events and the implied situations before, during and after an event and the roles of the entities affected by some event[16]. It was developed together with the Circumstantial Event Ontology for Calamities (CEO)[17]. Additionally, a very good reference to see the evolving interest in events and how their representation shaped over time is the timeline by Sprugnoli and Tonelli [31][18]

Regarding event representation in the legal domain, one of the most well-known upper ontologies in the legal domain is LKIF[32] (Legal Knowledge Interchange Format), that includes more than 200 classes. In LKIF, events are considered *changes* that "*occur against this canvas of temporal and spatial positions*" [33]. Another well-known representation option is LegalRuleML [19], a format for expressing and inferencing over legal knowledge.It does not model events *per se*, but only temporal dimensions of the norms and other concepts such as participants, time, locations, jurisdictions, artifacts, and compliance. Finally,

[11] https://openeventdata.github.io/
[12] https://www.w3.org/TR/owl-time/
[13] https://semanticweb.cs.vu.nl/2009/11/sem/
[14] https://sbmi.uth.edu/bsdi/TEO_1.0.0.owl
[15] http://motools.sourceforge.net/event/event.html#
[16] https://github.com/RoxaneSegers/ESO-Ontology
[17] https://github.com/RoxaneSegers/CEO-Ontology
[18] Available here: http://dhlab.fbk.eu/Timeline_events/
[19] https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=legalruleml

the Oasis standard Akoma Ntoso [20] has become widely known in the last years. Akoma Ntoso is an XML markup schema for describing legal resources of various types, for example, laws, regulations and court decisions. Events are considered "Actions and occurrences", although they are not specifically targeted and are considered "other concepts"[21].

2.2. *Event Extraction in the legal domain*

Beside generic efforts in event extraction such as the carried out by temporal taggers following TimeML [34–37] or related tasks such as frame-semantic parsing [38–41], semantic role labeling [42, 43] or open information extraction[22], some proposals have been made specifically in the legal domain. These works often involve *ad hoc* definitions of events, ignoring general event annotation schemes.

In the context of legal information retrieval, events can be considered as temporally bounded objects that have entities important as participants that played a significant role in a case. To this aim, Lagos et al. [44] propose an NLP semi automatic approach to enable the use of entity related information corresponding to the relations among the key players of a case, extracted in the form of events. They are interested in the topic, the roles, the location and the time, and consider different types of events. On the other hand, Maxwell et al. [45] reviewed 150 events extracted 18 sentences from the Canadian Supreme court and compared them with automatic extraction using SRL (Semantic Role Labelling) on two cases. Another approach was done for Spanish [46], looking for patterns in documents that help them identify legal events and related information (*who, what, to whom* and *where*), and analyzing the verbs that occur in the texts. In order to improve information retrieval in Brazilian courts, also a similar work was performed for Portuguese [47]. In this work, legal events are understood as the cognitive connections that specialists make when they are reading a legal document, and the authors try to recognize possible legal event structures to be described in legal documents. They use semantic frames with participants and properties. Nevertheless, this work was reported to be just manual for now, and just 10 legal frames have been already identified.

[20] http://www.akomantoso.org
[21] http://www.akomantoso.org/?page_id=47
[22] https://stanfordnlp.github.io/CoreNLP/openie.html

Another possible application of event extraction is the collection of rights or obligations from regulations. This is a different approach because it does not relate to events that actually happen at a precise time but some entities, but to *abstract* events that describe an hypothetical situation that might have some consequences, with some conditions and related constraints. For instance, the work by Kiyavitskaya et al. [48] aims to automatically extract legal requirements from legal text, namely rights and obligations. On the other hand, the Nomos framework [49], extracts legal metadata in an automatic fashion. Although events are not explicitly considered, other core concepts related to events (situations, roles) are tackled. Additionally, events are also targeted in other works in legal literature [50, 51].

In summary, legislation systems consist still of semiautomatic or even manual approaches. It can also be observed that most of the proposals within the legal domain are tend to be supported by patterns, using manually crafted rules or semantic role labeling techniques [44, 45, 48].

3. Event Extraction

Based on a previous works about temporal expressions in the legal domain [7], first step for building a knowledge graph was to decide the source of the documents, since there are important differences among jurisdictions, even when they share the language. Due to the ease of importing and reusing judgments from their respective repositories, as well as the multilingual challenge it offers and the possible associated documents that could eventually be added in a knowledge graph, we decided to work with decisions from European courts, namely the European Court of Human Rights (ECHR) and European Court of Justice (ECJ). Choosing a specific source also allowed us to analyze more effectively the structure of the documents, which will greatly improve the ability to extract relevant events [6]. Regarding the format of the annotations, we will use the one specified in the EventsMatter corpus [52], in which a very preliminary version of the tool presented in this section was briefly introduced.

The remaining of this section is as follows. First we will show how the structure extractor of the judgments works (Section 3.1). Then, the different training strategies used will be presented in Section 3.2. Finally, Section 3.3 will detail the pipeline of the event extraction algorithm, that applies the two previous techniques.

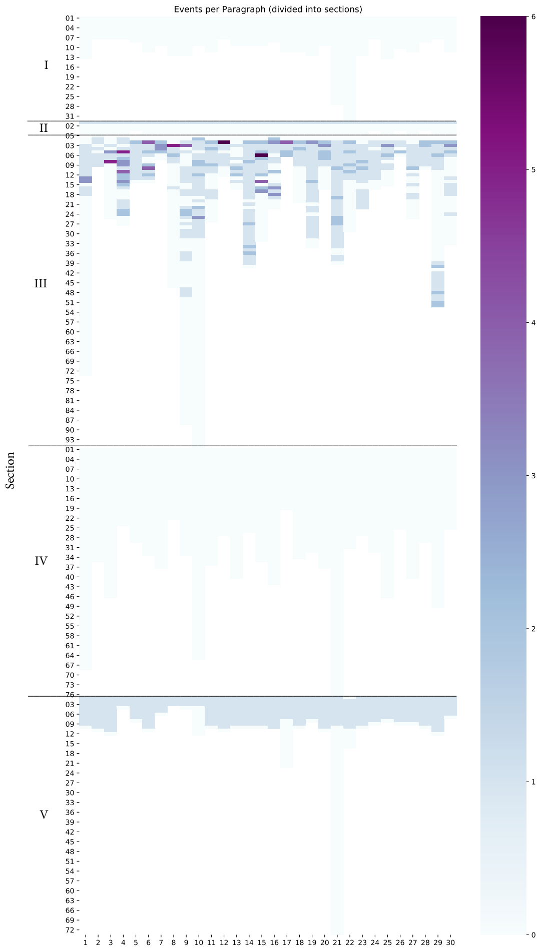

Fig. 1. Events per Paragraph in the documents in the EventsMatter corpus.

3.1. *Structure Extraction*

To illustrate the importance of structure extraction when dealing with relevant events, let us analyze their presence along the different sections of the documents in the EventsMatter corpus [52], the only available corpus of judgments annotated with events. Fig. 1 represent the distribution of events along paragraphs and sections. Fig. 2 displays this presence in the paragraphs of each specific section. Fig. 3 shows the distribution per section, while Fig. 4 does so on average per section.

Fig. 1 depicts the distribution of events along each of the thirty documents in the EventsMatter corpus. Regarding the colors, since not all the judgments have the same amount of paragraphs per section, white means there is not such paragraph in that document. Lightest blue indicates the paragraph exists, but contains no events, while form darker blue to purple colors denote the existence of one or more events (until six), depending on the darkness. This is applicable to the four images in this section, changing just the meaning of the color scale.

The Y axis represents the sections (roman numbers), and the number of paragraph for each of them (arabic numbers). Section I comprises all the content before the judgment itself, including information such as the name of the case or the members of the Chamber. Since it is not titled, we will name it "INTRODUCTION". Section II is the "PROCEDURE", usually short, where we can see there is just one event in the first paragraph, corresponding to the event of "lodge an application" that originated the case under judgment. Section III is "THE FACTS" and, as can be appreciated in the figure, contains most of the events, distributed heterogeneously through the section. Due to this, Fig. 2 reproduces in more detail this section, where we can see that the amount of events and their distribution is not necessarily related to the length of the section; more paragraphs do not imply more events. Section IV, "THE LAW", contains no events, since it refers to the European and national legislation to which the case is related, citing it along with other merits and pertinent considerations. Finally, Section V includes the "FINAL DECISION" by the court, always following the structure:

> FOR THESE REASONS, THE COURT, UNANIMOUSLY,
>
> 1. {Decision I}
> 2. {Decision II}
> 3. {...}
>
> {Information about the date and language of the writing, along with the signatures and any annex attached.}

Fig. 3 represents the amount of events in each of the five sections previously described. The "INTRODUCTION" section, as already pointed out, has always one event, while "THE FACTS" presents a very variable amount of them, reaching in some case forty events. This might be attributed to the different length of the section in each of the judgments, but Fig. 4, showing the average events per paragraph on each section, belies it. Finally, the "FINAL DECISION" section is very uniform, except of some documents that present annexes or have longer sections for other reasons.

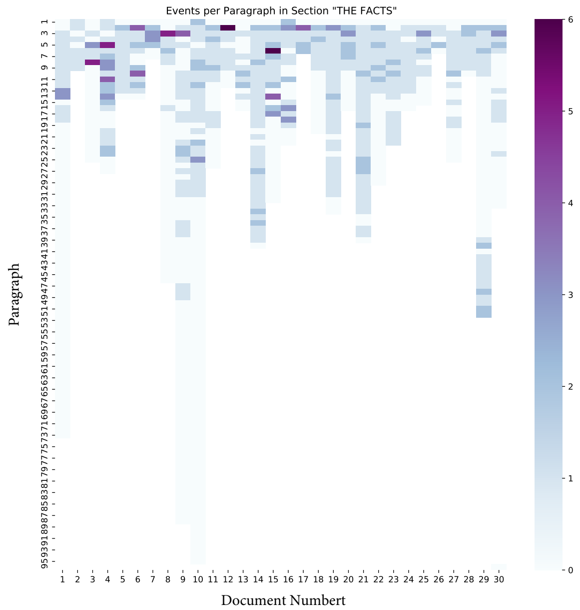

Fig. 2. Amount of Events in the section "THE FACTS" in documents in EventsMatter corpus.

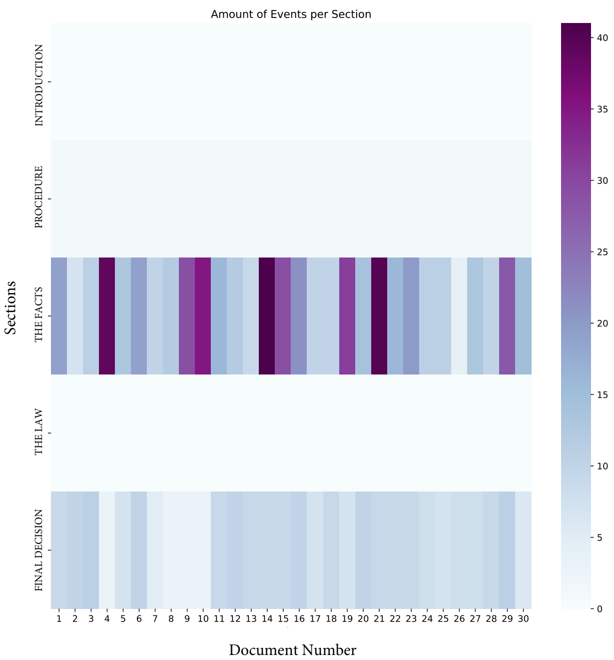

Fig. 3. Amount of events per section in the documents in EventsMatter corpus.

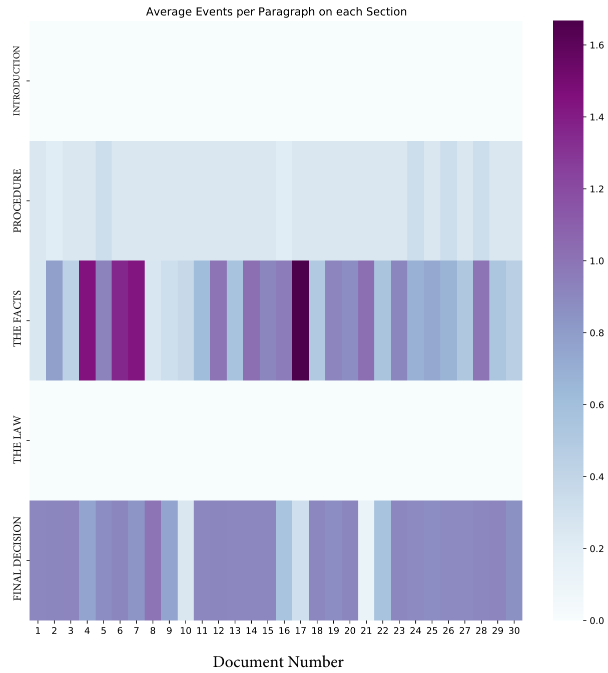

Fig. 4. Average events per paragraph per section in the documents in EventsMatter corpus.

From the analysis performed in the EventsMatter corpus, we can confirm the importance of the sections in identifying which events are relevant and which are not. To this end, we have developed a Structure Extractor that

1. Detects the structure of the document (from a .doc or an .html file) and divides in into parts with a title, a type, a parent and the begin and end offsets.
2. Looks for the most relevant sections in a judgment and send the sentences within to the algorithm that extracts the events, ignoring sections such as references to laws.

This Structure Extractor is currently able to handle the structure of the ECHR and ECJ documents, but in such a way that a new document type can be easily added. Additionally, if for any reason the processed documents did not adhere to the expected structure (for example, with very old cases that followed a different format), it would simply return all sections.

3.2. *Training Strategies*

Regarding the training strategy of the event extraction system, we used both semantic and syntactic considerations. On the one hand, we collected all the events and attached arguments annotated in the training set of the EventsMatter corpus [52]. The EventsMatter corpus is a collection of 30 legal decisions manually annotated with events and their arguments (namely, *who*, *when* and *what*, called *core*). Once collected, we stored both the core of the events and the relations among their different parts. On the other hand, we also used an external semantic resource, FrameNet, to enrich the keywords we use to identify legal events. Subsequent sections provide a detailed description of both approaches.

3.2.1. EventsMatter Training Set

The first step of the training phase was to collect all the event mentions in the corpus training set. We then isolated the parts of the sentences annotated as core and generate a sentence just with it, adding has generic subject "They" in order to make them simple to parse and grammatically correct. Thereupon we iterate over all these *simple* sentences, creating a frame for each of the main verbs of the sentences that stored the information of all the mentions of this verbs along the corpus. This is, that for instance the verb "lodge" (that is to some extent a *light verb*[23] in the legal domain) can appear in several sentences carrying different meaning depending on the object attached. Some examples of its use would be the constructions "lodge a complaint", "lodge a request", "lodge an appeal", "lodge an objection" or "lodge an action". It should be noted that most of these cases could be simplified using a single semantic-carrying verb, such as "to complain" or "to request", but that the legal domain tends to recur to these paraphrasing in texts, since they usually imply not just an action but also a formal procedure (usually administrative).

The verbs found in this phase are outlined in Fig. 5, where their type and frequency are also presented. Each of them constitutes a *frame* that will be used to identify and classify future mentions of each of the verbs in new texts. The structure of the Frame class used to store the information gathered for each of these verbs, along with an example of the mentions and information collected for a specific verb, is depicted in Fig. 6.

[23] *Light verbs* are those verbs that have little semantic meaning, needing therefore more words to constitute a full predicate. This is for instance the case of the verbs "make" or "take" in English. For more information on this linguistic phenomenon, please check the work by [53].

Finally, it must be noted that, as shown in Fig. 6, we make distinction between passive and active voice when searching for the dependency parsing relations among the members of the core of an event. This is a consideration that might not be important in general kind of texts, but the legal domain tends to present a high rate of passive verbs. Among the events in the training set, for instance, we find that the 14% of the mentions were expressed as passive sentences.

Two couples of txt files containing (1) the *simple* version of each sentence with a relevant event mention and (2) the type of events of each of the mention are available within the system – a couple for all the sentences of the corpus (named *all*) and another for just the training part (*train*). The collection of events can be easily extended by adding to the files new sentences and their respective types, and then executing the respective main class in the system that creates a events.ser file. This serialized file contains a HashMap of all the events and their information in the form of Frames.

An example of this Frame structure is detailed in Fig. 6. In the case shown, we found seven different mentions of the verb *bring* in our corpus (top right in the figure), where we marked in bold the *mention* of the verb, underlined its *object* and double underlined the *subject* in the case of passive voice. Finally, the text box in the bottom-right shows how would be the frame extracted from these seven sentences. There we can see the different objects (obj) found (*proceedings, claim, action, counterclaim*), as well as a *P* in the fifth position of the array, meaning that that sentence was passive. In the **passRels** and **actRels** we see the relations that connect the different parts of the core in the dependency parsing of the sentences (**passRels**, passive relations, from the 5th sentence, and **actRels** from the rest of them). Regarding **typeEvent**, it stores the different type of event (*circumstance* or *procedure*) the verb "bring") plays on each of the sentences. Finally, the percentage of these types is stored in the fields **percCirc** and **percProc**, that will help to decide if a mention found in a text is of one type or the other.

3.2.2. FrameNet training

It is straightforward that some events not present in the training set of the EventsMatter corpus should be detected in other documents, and even that events considered not relevant in those documents can be relevant in other cases.

This is why, in addition to the events gathered from the training set explained previously, we decided to

VERB	OCCUR	TYPE	VERB	OCCUR	TYPE	VERB	OCCUR	TYPE	VERB	OCCUR	TYPE
lodge	42	36%	extend	3	100%	note	2	0%	kill	1	100%
uphold	19	37%	provide	3	33%	overturn	2	50%	appoint	1	100%
dismiss	18	39%	indicate	3	33%	admit	2	0%	interview	1	0%
ask	17	35%	place	3	67%	hear	2	0%	open	1	100%
have	15	53%	question	3	100%	bury	2	50%	discuss	1	100%
appeal	13	62%	die	3	100%	summon	1	100%	declare	1	0%
refuse	12	42%	exclude	3	67%	instigate	1	0%	attend	1	0%
find	12	75%	carry	3	0%	commit	1	0%	buy	1	0%
order	11	64%	allow	3	0%	attempt	1	100%	plead	1	100%
issue	11	55%	request	3	67%	put	1	100%	undertake	1	100%
apply	10	40%	publish	3	67%	cover	1	100%	privatise	1	100%
give	9	22%	challenge	3	67%	review	1	0%	fine	1	100%
quash	9	44%	respond	3	33%	deprive	1	0%	leave	1	0%
institute	8	63%	release	2	50%	reduce	1	0%	contact	1	100%
discontinue	8	50%	decline	2	50%	pass	1	100%	claim	1	0%
inform	7	14%	enter	2	0%	remain	1	100%	bear	1	0%
bring	7	57%	sentence	2	100%	agree	1	100%	vacate	1	100%
authorise	7	57%	fail	2	100%	drink	1	0%	consider	1	100%
impose	6	33%	oppose	2	0%	stop	1	0%	amend	1	0%
reject	6	50%	become	2	0%	detain	1	100%	telephone	1	100%
start	6	33%	exercise	2	0%	terminate	1	0%	duplicate	1	0%
marry	5	40%	seek	2	0%	begin	1	100%	complain	1	100%
undergo	5	20%	file	2	50%	object	1	100%	decrease	1	0%
return	5	40%	receive	2	0%	examine	1	0%	keep	1	0%
send	5	60%	learn	2	100%	seize	1	100%	exchange	1	0%
submit	5	40%	stay	2	0%	settle	1	100%	try	1	0%
grant	5	40%	report	2	50%	deliver	1	0%	occupy	1	0%
decide	4	50%	invite	2	50%	dissolve	1	0%	rule	1	100%
conclude	4	50%	arrest	2	50%	speak	1	0%	delete	1	100%
divorce	4	75%	sign	2	0%	convict	1	0%	make	1	0%
register	4	75%	hold	2	50%	acquit	1	0%	restore	1	0%
do	4	25%	initiate	2	50%	charge	1	100%	identify	1	0%
reply	4	25%	suspend	2	100%	set	1	0%	perform	1	100%
move	4	50%	establish	2	50%	forward	1	0%	go	1	0%
state	3	100%	take	2	0%	launch	1	0%	invalidate	1	0%
write	3	33%	transfer	2	0%	draw	1	0%	pronounce	1	0%
accept	3	33%	reopen	2	100%	suspect	1	0%	visit	1	100%

Fig. 5. Events extracted from the EventsMatter training corpus. The second column (*OCCUR*) presents the amount of times that verb was annotated as relevant event. The third column (*TYPE*) shows the percentage of times it was typed as a *procedure* event (being the complementary percentage corresponding to the *circumstance* type).

enrich the system with frames from FrameNet [22]. FrameNet is a database that contains semantic frames together with the words that represent them in text, as well as additional information such as the arguments this frame can present. Since frames represent situations, they can be understood as events to some extent, and incorporating a selection of them to our target events would help to generalize our approach.

Since not all the frames in FrameNet are of interest, we manually inspected the database using the FrameGrapher tool[24], that allowed us to navigate through it and find the most relevant frames to our task. After examining the different relations among the frames, we found the most general ones, as well as their children, and imported their information using a Python script and the library `nltk` [54], including `framenet`. These most legally representative parent frames were namely "Committing_crime", "Crime_scenario", "Law", "Obligation_scenario", and "Misdeed". The frames collected from them, together with the lexical units associated to them (that is what we will look for in the text), are detailed in Table 1. The non lexical frames (this is, those that have no lexical units associated), in this case "Crime_scenario"

[24]https://framenet.icsi.berkeley.edu/fndrupal/FrameGrapher

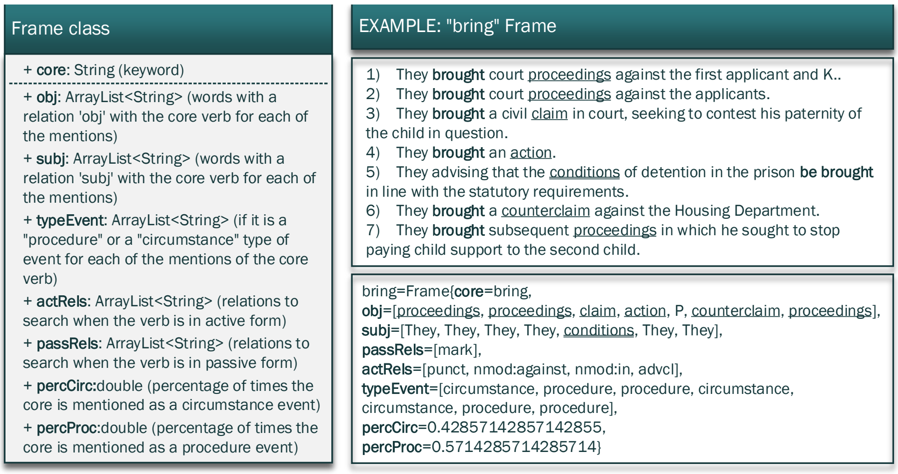

Fig. 6. Frame Class to store the events in WhenTheFact (left side) and example with the verb "bring" (right side).

and "Obligation_scenario", are not shown in the table for space purposes.

A txt file containing all these information is available in the system. In order to add more frames, it is just needed to add them to the file maintaining the same format. The system has a main class named *readFrames.java* that will generate a frames.ser file from it, and is this file that is read by the system in order to facilitate its latter use, storing the information in the form of a HashMap of structures containing the name, the core and the pos.

3.3. Event Extraction

Regarding the event extraction itself, Fig. 7 depicts the pipeline of the tool. We detail the different stages of the processing below.

First step consists of finding the relevant parts of the text to annotate, using for this the Structure Extractor detailed in Section 3.1. If the structure is not recognized, the whole text will be annotated, what obviously impacts in a negative way in the amount and quality of the events. Otherwise, just the relevant parts of the document are processed subsequently.

Next step is to find the sentences involving temporal expressions. To this aim we adapt and integrate the functionality of Añotador [55], a temporal tagger able to recognize temporal expressions. If there is at least one temporal expression in a sentence, we check if it is a special case (namely the application lodgement, that always follows the same syntactic structure). If so, we annotate the arguments and go to the next sentence. If not, we check if the sentence contains any of the events stored in events.ser, that contains the information gathered from the training corpus. If so, we do the dependency parsing (*deppar*) of the sentence (using CoreNLP [56]) and check if it is valid and look for the arguments (see (1) below). If not, we check again for the frames stored in frames.ser (the legal frames specifically selected from FrameNet). If this is the case, we check them similarly that in the events case (see (2)). Once we detected the main event in the sentence, if there was more than one temporal expression in it, we will select the temporal expression that is the closest to the core of the event.

(1) For the events, we check if it is not an auxiliary verb and if it is not in the gerund form. Then we check if it is in passive or active voice. Depending on this, we will look either for the relations stored in events.ser gathered from passive training cases of from active cases.
(2) For the frames, the check function is similar to the events' one, but there are no specific relations stored for each frame, so the argument "who" and the extent of the core are therefore detected using default relations.

Once all the sentences have been explored, we merge all the annotations and produce the output. This output consists of an annotated xml and as a visual HTML that also includes a timeline built from the retrieved events.

Frame	**Lexical Unit (pos)**
Abusing	‘abuse (n)’, ‘abuse (v)’, ‘abusive (a)’, ’batter (v)’, ‘domestic violence (n)’, ‘maltreat (v)’, ‘maltreatment (n)’
Kidnapping	‘kidnap (v)’, ‘abduct (v)’, ‘shanghai (v)’, ‘nab (v)’, ‘snatch (v)’, ’kidnapping (n)’, ‘abduction (n)’, ‘kidnapper (n)’, ‘abductor (n)’, ‘snatcher (n)’, ‘kidnapped (a)’, ‘abducted (a)’
Piracy	‘hijack (v)’, ‘hijacking (n)’, ‘hijacker (n)’, ‘carjacking (n)’, ‘hijacked (a)’, ‘piracy (n)’, ‘pirate (v)’, ‘carjack (v)’
Rape	‘rape (v)’, ‘rape (n)’, ‘rapist (n)’, ‘raped (a)’, ‘sexually assault (v)’
Robbery	‘rob (v)’, ‘robber (n)’, ‘mug (v)’, ‘robbery (n)’, ‘mugger (n)’, ‘mugging (n)’, ‘stick-up (n)’, ‘hold-up (n)’, ‘hold up (v)’, ‘rob blind (v)’, ‘stick up (v)’, ‘ransack (v)’, ‘rifle (v)’
Smuggling	‘smuggle (v)’, ‘smuggling (n)’, ‘smuggler (n)’, ‘contraband (a)’, ’contraband (n)’
Theft	‘steal (v)’, ‘purloin (v)’, ‘filch (v)’, ‘snitch (v)’, ‘pilfer (v)’, ’swipe (v)’, ‘lift (v)’, ‘pinch (v)’, ‘thieve (v)’, ‘thief (n)’, ‘pickpocket (n)’, ‘cutpurse (n)’, ‘pilferer (n)’, ‘snatcher (n)’, ‘theft (n)’, ‘thieving (n)’, ‘pilferage (n)’, ‘light-fingered (a)’, ‘thieving (a)’, ‘snatch (v)’, ’nick (v)’, ‘embezzle (v)’, ‘misappropriate (v)’, ‘shoplift (v)’, ‘stealer (n)’, ‘shoplifter (n)’, ‘shoplifting (n)’, ‘pilfering (n)’, ‘stolen (a)’, ’embezzlement (n)’, ‘embezzler (n)’, ‘peculation (n)’, ‘misappropriation (n)’, ‘larceny (n)’, ‘snatch (n)’, ‘stealing (n)’, ‘pickpocket (v)’, ‘heist (n)’, ‘flog (v)’, ‘abstract (v)’, ‘cop (v)’, ‘rustle (v)’, ‘bag (v)’, ’abstraction (n)’, ‘make off (with) (v)’, ‘abscond (with) (v)’
Committing crime	‘commit (v)’, ‘perpetrate (v)’, ‘crime (n)’, ‘commission (n)’
Offenses	‘assault (n)’, ‘murder (n)’, ‘statutory rape (n)’, ‘sabotage (n)’, ’manslaughter (n)’, ‘hijacking (n)’, ‘theft (n)’, ‘burglary (n)’, ‘robbery (n)’, ‘conspiracy (n)’, ‘larceny (n)’, ‘copyright infringement (n)’, ’negligence (n)’, ‘possession (n)’, ‘felony (n)’, ‘sexual harassment (n)’, ’treason (n)’, ‘battery (n)’, ‘kidnapping (n)’, ‘fraud (n)’, ‘indecent assault (n)’, ‘sexual assault (n)’, ‘child abuse (n)’, ‘homicide (n)’, ‘arson (n)’, ‘rape (n)’
Criminal investigation	‘inquiry (n)’, ‘probe (n)’, ‘investigate (v)’, ‘inquire (v)’, ‘probe (v)’, ’investigation (n)’, ‘lead (n)’, ‘clue (n)’, ‘case (n)’
Arson	‘arson (n)’, ‘arsonist (n)’
Severity of offense	‘actionable (a)’, ‘capital (a)’, ‘indictable (a)’, ‘felonious (a)’
Suspicion	‘suspect (v)’, ‘under suspicion (of) (prep)’, ‘suspect (n)’
Arraignment	‘arraign (v)’, ‘arraignment (n)’
Arrest	‘arrest (v)’, ‘apprehend (v)’, ‘bust (v)’, ‘nab (v)’, ‘collar (v)’, ‘cop (v)’, ‘arrest (n)’, ‘bust (n)’, ‘apprehension (n)’, ‘book (v)’, ‘summons (v)’
Sentencing	‘sentence (v)’, ‘sentence (n)’, ‘order (v)’, ‘send up (v)’, ‘condemn (v)’
Trial	‘trial (n)’, ‘case (n)’
Appeal	‘appeal (n)’, ‘appeal (v)’, ‘appellate (a)’, ‘appellant (n)’
Bail decision	‘set (v)’, ‘fix (v)’, ‘order (v)’, ‘bail (n)’, ‘bond (n)’
Entering of plea	‘plead (v)’, ‘plea (n)’
Notification of charges	‘charge (v)’, ‘charge (n)’, ‘indict (v)’, ‘indictment (n)’, ‘accuse (v)’
Surrendering	‘surrender (v)’, ‘turn in (v)’, ‘give up (v)’, ‘surrender (n)’
Court examination	‘examine (v)’, ‘cross-examine (v)’, ‘cross (n)’, ‘cross-examination (n)’, ’examination (n)’
Jury deliberation	‘deliberation (n)’, ‘deliberate (v)’
Verdict	‘pronounce (v)’, ‘find (v)’, ‘finding (n)’, ‘ruling (n)’, ‘convict (v)’, ’conviction (n)’, ‘acquit (v)’, ‘acquittal (n)’, ‘verdict (n)’, ‘clear (v)’, ’guilty (a)’, ‘not guilty (a)’
Law	‘law (n)’, ‘code (n)’, ‘protocol (n)’, ‘act (n)’, ‘statute (n)’, ’regulation (n)’, ‘regime (n)’, ‘policy (n)’, ‘order (n)’
Legality	‘illegal (a)’, ‘legal (a)’, ‘lawful (a)’, ‘unlawful (a)’, ‘wrongful (a)’, ’illicit (a)’, ‘licit (a)’, ‘permissible (a)’, ‘wrongly (adv)’, ‘wrong (a)’, ’prohibited (a)’, ‘legitimate (a)’, ‘fair (a)’, ‘criminal (a)’
Prohibiting or licensing	‘ban (v)’, ‘forbid (v)’, ‘prohibit (v)’, ‘proscribe (v)’, ‘outlaw (v)’, ’ban (n)’, ‘prohibition (n)’, ‘bar (v)’, ‘allow (v)’, ‘entitle (v)’, ‘permit (v)’, ‘sanction (v)’
Being in effect	‘effective (a)’, ‘effect (n)’, ‘force (n)’, ‘valid (a)’, ‘void (a)’, ‘null (a)’, ‘binding (a)’
Compliance	‘adhere (v)’, ‘comply (v)’, ‘observe (v)’, ‘adherence (n)’, ‘compliance (n)’, ‘follow (v)’, ‘observance (n)’, ‘break (v)’, ‘violate (v)’, ‘contravene (v)’, ‘breach (v)’, ‘violation (n)’, ‘contravention (n)’, ‘breach (n)’, ’flout (v)’, ‘conform (v)’, ‘obey (v)’, ‘compliant (a)’, ‘transgress (v)’, ’transgression (n)’, ‘lawless (a)’, ‘contrary (a)’, ‘conformity (n)’, ‘keep (v)’, ‘honor (v)’, ‘abide (by) (v)’, ‘obedient (a)’, ‘observant (a)’, ‘play by the rules (v)’, ‘circumvent (v)’, ‘noncompliance (n)’, ‘(in/out of) line (n)’, ‘disobey (v)’, ‘in accordance (a)’, ‘by-pass (v)’
Documents	‘visa (n)’, ‘passport (n)’, ‘subpoena (n)’, ‘warrant (n)’, ‘certificate (n)’, ‘papers (n)’, ‘license (n)’, ‘summons (n)’, ‘diploma (n)’, ‘deed (n)’, ’lease (n)’, ‘agreement (n)’, ‘treaty (n)’, ‘charter (n)’, ‘authorization (n)’, ‘deposition (n)’, ‘brief (n)’, ‘writ (n)’, ‘affidavit (n)’, ‘will (n)’, ’testimony (n)’, ‘testament (n)’, ‘ruling (n)’, ‘finding (n)’, ‘opinion (n)’, ’title (n)’, ‘orders (n)’, ‘contract (n)’, ‘permit (n)’, ‘document (n)’, ’contractual (a)’, ‘accord (n)’, ‘confirmation (n)’, ‘identification (n)’, ’business card (n)’
Enforcing	‘enforce (v)’, ‘enforcement (n)’
Strictness	‘authoritarian (a)’, ‘indulgent (a)’, ‘lenient (a)’, ‘liberal (a)’, ’strict (a)’, ‘tolerant (a)’, ‘severe (a)’
Giving in	‘relent (v)’, ‘acquiesce (v)’, ‘yield (v)’, ‘cave in (v)’, ‘give in (v)’, ’give way (v)’, ‘capitulate (v)’, ‘fold (to demands)’ (v)’, ‘cave (v)’, ’submit (v)’
Terms of agreement	‘condition (n)’, ‘stipulation (n)’, ‘provision (n)’, ‘clause (n)’, ‘term (n)’, ‘parameter (n)’
Misdeed	‘misdeed (n)’, ‘sin (v)’, ‘sin (n)’, ‘transgress (v)’, ‘transgression (n)’, ‘peccadillo (n)’
Guilt or innocence	‘guilty (a)’, ‘innocent (a)’, ‘guilt (n)’, ‘innocence (n)’, ‘blood on hands (n)’

Table 1

Final selection of legal-related frames from FrameNet used in WhenTheFact.

4. FT3 Ontology

In order to properly represent the temporal information extracted from the documents, we have created an ontology named fromTimeToTime (ft3). The purpose of this ontology is double-folded: on the one hand, we want it to be able to represent information from the annotations related to time and events that the current ontologies do not cover. On the second hand, we want to facilitate the translation between one annotation format or temporal representation format to another.

In this section we will briefly introduce this new ontology, stressing the main design decisions. The later section, that will describe the format converter, will also present some examples of the expected use of the ontology.

4.1. Temporal Expression representation

One of the objectives of this ontology is to be able to represent any time-related annotation format. Due to this, we created some high-level classes, namely

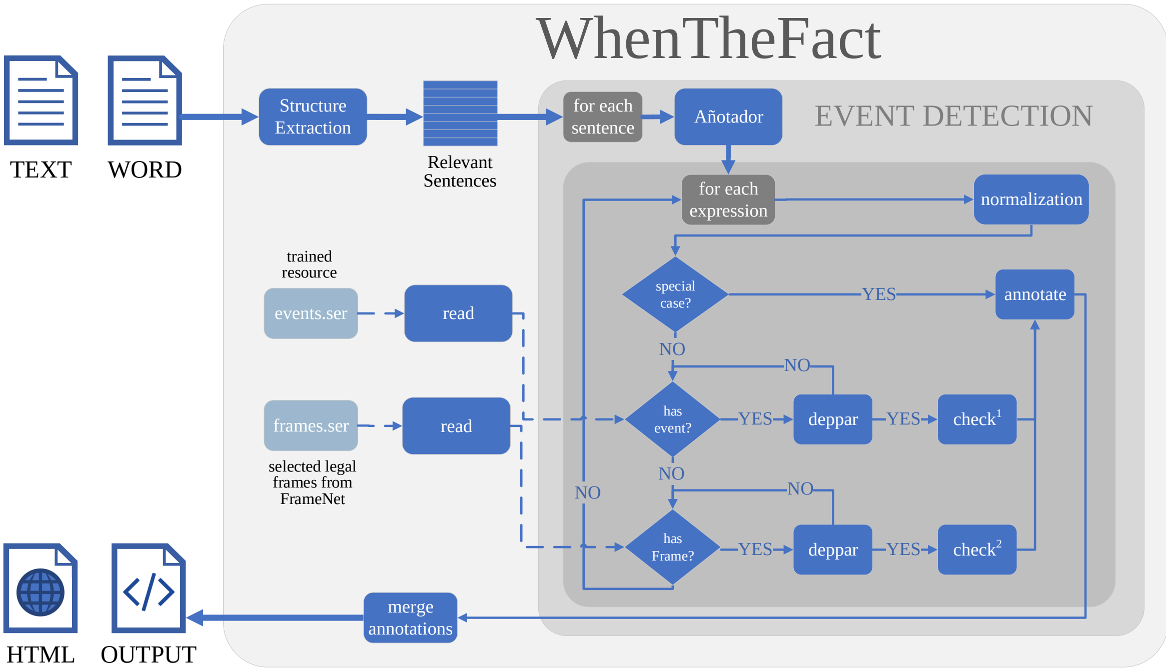

Fig. 7. Pipeline of WhenTheFact.

Guidelines, *Annotation* and *Argument*, that allow to create subclasses and instances for specific implementations. Additionally, we also added some abstract classes that allow us to unify to some extent the different representations, such as the case of the class *temporal expression*.

We implemented, as an exemplary, the different tags and concepts in the TimeML annotation standard, the most well-known annotation format for temporal expressions. Thus, the ontology offers, for example, the different arguments for the concepts considered in the annotation standard (temporal expressions, events, event instances and signals), with instances for the valid values of these arguments, but leaving the option of eventual extensions. These are, at the same time, related to other classes in the ontology, like the case of the class *temporal expression* shown in Fig. 8.

Fig. 8 depicts the relation implemented in the ontology among the class *ft3:Temporal Expression*, the class *ft3:TIMEX3Annotation* and the class *sem:Time* from the Simple Event Model, used as abstract class to represent Time. This class is also linked to classes from the Time ontology, and can as well be associated to any other temporal representation option.

Additionally to the integration of these already existing representations, we decided to add also the class

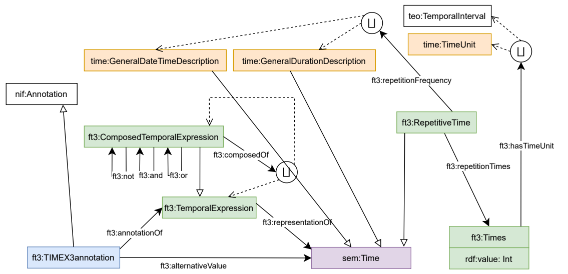

Fig. 8. Excerpt of the ft3 ontology related to temporal expression representation.

ft3:ComposedTemporalExpression in order to be able to represent temporal expressions not currently covered by the existing standards. This class enables to join, intersect or negate a temporal expression, allowing to represent in a simple way complex expressions such as "*All days but Mondays*" or "*On Monday or Tuesday*".

4.2. Event representation

Regarding events, the main consideration we wanted to represent in the ontology is the distinction about the following concepts:

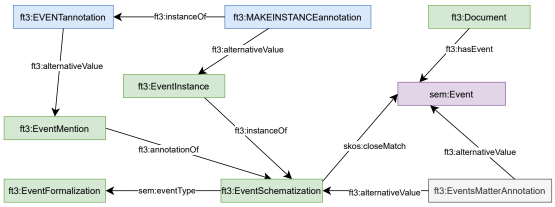

Fig. 9. Excerpt of the ft3 ontology related to event representation.

- Event mention: the textual reference in the text. There can be several references to an event in a text (correference). Also a mention can be related to several events or subevents. This event mention can have attached an annotation.
- Event schematization: the abstract representation of the information we have about an event, such as *who*, *where*, and so on. It is a midpoint between text, reality and abstraction. This representation can be useful when dealing with QA.
- Event instance: actual happening of an event in reality. One mention can imply several instances. Also, in some cases, we cannot derive the amount of instances. This is concept is specially important for timeline building.
- Event formalization: it is an abstract representation of the event, a possible formalization in the form of frame, for instance. We can consider it as a way to classify events by linking them to resources such as WordNet or FrameNet.

Fig. 9 shows how these concepts were formalized in the ontology. Besides the four main concepts, we see how TimeML event-related concepts MAKEINSTANCE and EVENT are associated to *ft3:EventInstance* and *ft3:EventMention*, respectively. Similarly, the event annotations from the EventsMatter corpus are also represented but linked to *ft3:EventSchematization*, since it provides information such as *who* and *when*. Finally, as happened with *sem:Time*, we also relate our event representation to the equivalent for event in the SEM ontology, *sem:Event*.

Furthermore, in order to clarify how these concepts reflect real annotations, Table 2 shows different examples of sentences and how they would comply to this representation. Some of these examples are discussed further below:

a) This example is the simplest. One temporal expression and one mention of an event lead to a single schematization and a single instance.

Table 2

Example of sentences and the correspondent representation attributions. First column shows the letter we assigned to the example, while second column presents the event we are focusing on and third one the example sentence itself. Last four columns show the amount of Temporal Expressions (TEx, underlined in the sentence), Event Mentions (Men, in bold in the sentence), Event Schematizations (Sch) and Instances (Ins) the sentence would produce. (*) The stroll has been considered a meronymic correference of the event "go", but could also be considered a subevent.

#	Event	Sentence	TEx	Men	Sch	Ins
a	go	Yesterday I **went** to the park.	1	1	1	1
b	go	I **went** to the park on the 5th and the 6th.	2	1	1	2
c	go	I **went** to the park.	0	1	1	1
d	go	I **go** to the park every Tuesday	1	1	1	X
e	go	I **went** to the park. During the **stroll**, it started raining.	1	2*	1	1
f	meet	They **met** several times.	1	1	1	X
g	concert	The **concert** was cancelled.	0	1	1	0
h	cancel	The concert was **cancelled**.	0	1	1	1
i	attend	The applicant **did not attend**.	0	1	1	0
j	skip	He **skipped** the sessions.	0	1	1	X
k	attend	He skipped the sessions.	0	1	1	0
l	sessions	He skipped the **sessions**.	0	1	1	X
m	admit	The appeal **was not admitted**.	0	1	1	1
n	refuse	The appeal **refused**.	0	1	1	1

b) In this case, there is still one mention of event and one schematization, but two temporal expressions associated; the action happens twice (one each day) and therefore there are two event instances.

c) In this example there is no temporal expression, but it can be assumed that the event happens once, so just one instance would be derived.

d) In this sentence, periodic temporal come into play. The only expression suggests that the event happens several times, but we have no clue about how many.

e) If we consider that "the stroll" is a mention of the event of going to the park, we have a correference, and therefore just one mention.

f) This case is similar to case d), except for the fact that the temporal expression is not periodic, but simply implies more than one happening.

g) and h) Both examples share the sentence, but depend on which event we focus for its formalization. If we focus on the concert (example g)), it did not happen, so has no instance. On the contrary, in example h), the cancellation is an actual event, so there is one instance. How we decide to interpret this situation will usually depend on the specific use we are dealing with.

i) This is a very interesting example from the legal point of view. The fact that someone did not attend to a view or a trial is commonly reflected in judgments. Although the event of attending did not happen, so there is no instance of it, in the following example we will see similar cases expressed differently.

j) k) and l) Another way to express that someone did not attend a procedural event is to say they "skipped" it. Therefore, being the same case as i), the fact of no acting becomes an act itself, and can have consequences.

m) and n) Here we find again the case of an event that can be both equally described with a verb or its negated opposite. Differently to the case of the concert, the fact of refusing or not admitting an appeal does not mean it does not happen: the appeal actually happened, and this is just the result of the deliberation on it. Therefore, here the negation is clearly still an event, because the fact of not admitting an appeal is an action itself, just expressed as the negation of one of the two possible results.

After these examples, the problematic existing between the different ways of understanding the same event depending on how it interacts with the temporal expressions or on the characteristics of the event itself become evident. There is not a correct way of understanding or representing events, and the meaning extremely depends on the situation and its particularities, the context of the case and the requirements of the use case for which the representation is needed.

Finally, in order to guarantee and facilitate the use of the ontology, it has been documented using the the OOPS! Ontology Pitfall Scanner [57] and the WIDOCO wizard for ontology documentation [58], respectively. The documentation (including evaluationhttps://fromtimetotime.linkeddata.es/ontodoc/OOPSEvaluation/OOPSeval.html, mainly consisting about minor comments and with no critical pitfalls) can be checked in the ontology webpage, where it is published together with the ontology itself. Both are additionally available in Zenodo[25].

[25]https://zenodo.org/record/5034640

5. FromTimetoTime Converter

One of the main lacks we have identified dealing with time-related information is the gap existing between the task of finding temporal information in texts and its latter usage for further tasks. Besides the existence of many time-related ontologies and options, such as Temporal Description Logics in order to reason over them, there is no bridge between them and the pure NLP task.

In order to tackle this lack, we have created a converter able to read different temporal annotation formats and output them in a different formats, including the ontology previously mentioned.

This service is currently able to read TimeML and EventsMatter documents, as well as ft3 ontology documents, and transform them in the following formats:

- EventsMatter: TimeML documents or from our ontology can be translated to the EventsMatter format. Fig. 10 shows an example of this format.
- TimeML: documents from the ontology or in the EventMatters format can be translated to the TimeML standard. In Fig. 11 we can see the TimeML output of the converter for the previously mentioned example.
- ft3: the annotations of both annotation formats will be expressed in the form of the ft3 ontology. Fig. 12 presents the example introduced in Fig. 10 as ft3 RDF.
- ft3+time: additionally to the RDF representation of the annotations, the temporal expressions annotated will be transformed to time-related ontology data, mainly to the Time Ontology, but also to complementary ones from other ontologies.
- ft3+events: in addition to the RDF representation of the annotations, the events detected in the text are also represented as sem:Event classes. They contain the information of the arguments that might be annotated in the original text, such as *sem:hasActor* or *sem:hasTime*.

Beside these formats, it is also possible to extend the converter to include more options. In order to do so, we implemented a pivot class named MAP that can be considered an "interlingua". This class is a map of Strings where the key is the identifier of the argument. In order to know how each type of annotation must be interpreted, when each annotation is read a *metatype* is assigned to it. For instance, both the *Event_when* tag (from EventsMatter format) and the *TIMEX3* one (from the TimeML standard)

Table 3

Correspondence among different annotations and MAP. Each of the values of the column map has a correspondent object property in the ft3 ontology (e.g., TYPE has ft3:hasType).

metatype	TEMPORAL	EVENT		WHO
MAP	TIMEX Event_when	EVENT	Event _what	Event _who
TYPE	type	class	type	
ID	tid	eid	tid	tid
VALUE	value			
SENTID	sentid			
FUNCTIONIN DOCUMENT	functionIn Document			
TEMPORAL FUNCTION	temporal Function			
VALUEFROM FUNCTION	valueFrom Function			
MOD	mod			
ANCHOR TIMEID	anchor TimeID			
BEGIN POINT	begin Point			
END POINT	end Point			
QUANT	quant			
FREQ	freq			
LEMMA			lemma	
STEM		stem		
PROV			prov	

have the metatype *TEMPORAL ANNOTATION*, while *Event_what* and *EVENT* have *EVENT ANNOTATION*. Table 3 shows the correspondence of some of these metatypes, as well as the mapping among the arguments.

```
On <Event_when tid="t4" type="DATE" value="1990-10-
06">6 October 1990</Event_when> <Event_who argument
="who" tid="t4">he</Event_who> <Event_what argument
="what" tid="t4" type="circumstance" prov="eventsma
ttertrain" lemma="marry">married</Event_what> Ms N.
R.
```

Fig. 10. Example of text annotated in the EventsMatter format.

This MAP facilitates the task of translating among all the different formats. Consequently, to add a new format it will be necessary to simply perform the following steps:

– Create a new class that implements the "AbstractAnnotation" class for each new annotation and whose constructors receive MAP as an argument.

```
<?xml version="1.0" ?>
<TimeML xmlns:xsi="http://www.w3.org/2001/XMLSchema
-instance" xsi:noNamespaceSchemaLocation="http://ti
meml.org/timeMLdocs/TimeML_1.2.1.xsd">
On <TIMEX3 tid="t4" type="DATE" value="1990-10-06">
6 October 1990</TIMEX3> he <EVENT eid="t4" class="c
ircumstance">married</EVENT> Ms N.R.
</TimeML>
```

Fig. 11. Output of the converter as TimeML.

```
<https://fromtimetotime.linkeddata.es/doc/samples/
doc002>
a nif:Context , ft3:Document ;
nif:beginIndex "0"^^xsd:nonNegativeInteger ;
nif:endIndex "36"^^xsd:nonNegativeInteger ;
nif:title    "X"^^xsd:String ;
nif:isString """On 6 October 1990 he married
Ms N.R.""" ;
nif:AnnotationUnit [
    <https://fromtimetotime.linkeddata.es/doc/
    samples/doc002/EventsMatter/Event_when
    annotation_t4_5> [
    a ft3:EventsMatterEvent_when ;
    nif:beginIndex "3"^^xsd:nonNegativeInteger ;
    nif:endIndex "17"^^xsd:nonNegativeInteger ;
    ft3:hasID "t4"^^xsd:String ;
    nif:isString """6 October 1990""" ;
    ft3:hasTid "t4"^^xsd:String;
    ft3:hasValue "1990-10-06"^^xsd:String;
    ft3:hasType ft3:DATE ;
   ];
    <https://fromtimetotime.linkeddata.es/doc/
    samples/doc002/EventsMatter/Event_what
    annotation_t4_6> [
    a ft3:EventsMatterEvent_what ;
    nif:beginIndex "21"^^xsd:nonNegativeInteger ;
    nif:endIndex "28"^^xsd:nonNegativeInteger ;
    ft3:hasID "t4"^^xsd:String ;
    nif:isString """married""" ;
    ft3:hasType ft3:circumstance ;
    ft3:hasProv "eventsmattertrain"^^xsd:String;
    ft3:hasLemma "marry"^^xsd:String;
   ];
    <https://fromtimetotime.linkeddata.es/doc/
    samples/doc002/EventsMatter/Event_who
    annotation_t4_7> [
    a ft3:EventsMatterEvent_who ;
    nif:beginIndex "18"^^xsd:nonNegativeInteger ;
    nif:endIndex "20"^^xsd:nonNegativeInteger ;
    ft3:hasID "t4"^^xsd:String ;
    nif:isString """he""" ;
   ];
] .
```

Fig. 12. Output of the converter with the output format *ft3*. Prefixes are not included in order to avoid verbosity.

– Add a constructor to MAP that receives the new class.
– Create a reader of that format that stores the annotations in Document format.
– Add to Document an option to be translated to the new format.

Similarly, for handling the conversion of TimeML values to the ontology format (or to any other temporal

```
    ft3:alternativeValue [
        <https://fromtimetotime.linkeddata.es/doc/
        samples/doc002/Time_t4> [
        a sem:Time,
        time:GeneralDateTimeDescription ;
        time:year "1990"^^xsd:gYear ;
        time:monthOfYear greg:October ;
        time:month "--10"^^xsd:gMonth ;
        time:day "---06"^^xsd:gDay ;];
        ];
    ];
```

Fig. 13. Additional output of the converter with the output format *ft3+time*.

```
ft3:hasEvent [
<https://fromtimetotime.linkeddata.es/doc/
samples/doc002/EVENT_t4> [
a sem:Event ;
sem:EventType "marry" ;
ft3:hasType ft3:circumstance ;
ft3:hasID """t4""" ;
sem:hasTime [
<https://fromtimetotime.linkeddata.es/doc/samples/
doc002/Time_t4> [
    a sem:Time, time:GeneralDateTimeDescription ;
    time:year "1990"^^xsd:gYear ;
    time:monthOfYear greg:October ;
    time:month "--10"^^xsd:gMonth ;
    time:day "---06"^^xsd:gDay ; ];
] ;
sem:hasActor """he"""^^xsd:String ; ]
].
```

Fig. 14. Additional output of the converter with the output format *ft3+events*.

format) we use another pivot map, named TIMEMAP, detailed in Table 4.

In the case of DATEs or TIMEs, we represent the information as part of a *time:GeneralDateTimeDescription*. The correspondence of each value of the TIMEMAP to the Time ontology is therefore to properties such as *time:day*. Additionally, for temporal expressions not covered by the Time ontology, as mentioned before, we used the TEO[14] and the INTERVALS[26] ontologies. This is the case of the key PARTDAY, that represents parts of the day such as *morning* or *noon*, where we used *teo:TEO_0000190* (labeled *Instant of the day*) to describe that property and its object (*teo:TEO_0000194* and *teo:TEO_0000195*, respectively). In the case the object was not available, we created one individual in out ontology (e.g. *ft3:NIGHT*). In other occasions, time had the property but not the right object, as in the case of *quarters, trimesters* or *semesters*, where we used INTERVALS. Finally, in some cases, such as references to the past, present or future (represented in TimeML as

[26]http://reference.data.gov.uk/def/intervals/

Table 4

Correspondence between TIMEMAP keys and the information contained different types of temporal expressions in the TimeML standard. The SET type has been divided since its value can be in the form of a DATE or a DURATION. (*) DUR stands for DURATION.

TIMEMAP key	Types of temporal expresions in TimeML				
TYPE	DATE	DUR*	TIME	SET-DATE	SET-DUR*
TIMEUNIT				X	X
TIMEAMOUNT				X	X
REF	X				
YEAR	X	X	X	X	X
SEASON	X	X	X	X	X
WEEK	X	X	X	X	X
WEEKDAY	X	X	X	X	X
HALFYEAR	X	X		X	X
TRIMESTER	X	X		X	X
QUARTER	X	X		X	X
ERA	X				
DAY	X	X	X	X	X
MONTH	X	X	X	X	X
DECADE		X			X
CENTURY		X			X
MILLENIUM		X			X
SECOND		X	X		X
MINUTE		X	X		X
HOUR		X	X		X
PARTDAY			X		

DATEs with values *PAST_REF*, *PRESENT_REF* and *FUTURE_REF*, respectively), we also had to add the property (*ft3:hasTimeRef*).

On the other hand, in the case of DURATIONs, we represent the information as part of a *time:GeneralDurationDescription*. We again priorizited the Time ontology properties and objects, using for instance *time:days* or *time:years* to represent the amount of days and years in the DURATION.

Finally, SETs are described using a class with two different properties, namely *ft3:RepetitiveTime* and the properties *ft3:repetitionFrequency* and *ft3:repetitionTimes*. The first property would represent the frequency of a periodic event, while the second corresponds to the amount and granularity of the repetition. Fig. 15 and Fig. 16 represent the temporal information of the expression "*Twice a week*" and "*Three days every two months*", respectively.

```
ft3:alternativeValue [
 <https://fromtimetotime.linkeddata.es/
 doc/samples/doc002/Time_t1> [
 a sem:Time, ft3:RepetitiveTime ;
ft3:repetitionFrequency [
     time:weeks         "1"^^xsd:decimal ;
    ];
ft3:repetitionTimes [
    ft3:hasTimeUnit    ft3:TIMES ;
    rdf:value         2^^xsd:nonNegativeInteger ;
    ];
];];
```

Fig. 15. Alternative value of the temporal expression "*Twice a week*".

```
ft3:alternativeValue [
 <https://fromtimetotime.linkeddata.es/
 doc/samples/doc002/Time_t1> [
 a sem:Time, ft3:RepetitiveTime ;
ft3:repetitionFrequency [
     time:months         "2"^^xsd:decimal ;
    ];
ft3:repetitionTimes [
    ft3:hasTimeUnit    time:DAY ;
    rdf:value         3^^xsd:nonNegativeInteger ;
    ];
];];
```

Fig. 16. Alternative value of the temporal expression "*Three days every two months*".

The code of the converter is available online[27] and can be freely adapted. The converter can also be tested in the fromTimeToTime webpage.

6. Legal Knowledge Graph

The junction of the different resources and tools detailed in previous sections allow to create a legal event-based knowledge graph. Fig. 17 shows how the different contributions interact in order to populate and query the knowledge graph.

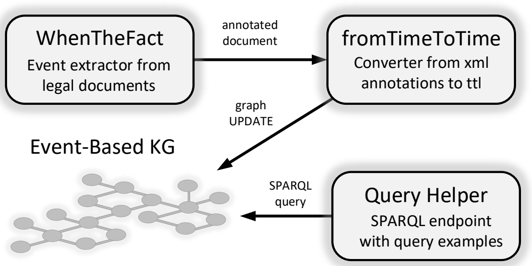

Fig. 17. Pipeline of population and query of the legal event-based knowledge graph

First, the event extractor WhenTheFact process and annotates legal documents from two different European sources. Then, the annotated version of the document (in the EventsMatter format) is sent to the fromTimeToTime converter in order to be outputted as RDF, using the fromTimeToTime ontology. Afterwards, this result is updated to the knowledge graph, that is therefore populated with documents in the format ft3+events (an example was shown in Fig. 14). Finally, the graph can be queried from the SPARQL endpoint enabled for this purpose. In this endpoint, some basic predefined queries help to explore the knowledge graph (such as "return events form a specific year, document or type"), but also free queries can be sent to it.

Currently, the only way to add documents to the knowledge graph is via the WhenTheFact event extractor due to security reasons. Nevertheless, all the code and resources needed to replicate and handle the legal event-based knowledge graph are provided. It is also possible to choose the way to store the triples; for our tests, both Virtuoso[28] and BlazeGraph[29] have been used, and just the parameters of the request (such as the url and the authentication, if needed) need to be adapted.

One of the main applications to exploit the lnowledge graph is the timeline generation, a task that has already been tackled for Event-Centric Knowledge Graphs in EventKG+TL [59]. Being able to build the timeline of the different actors involved in a case would also help to find inconsistencies in the alabi provided by them and other evidences. Additionally, the performance of general tasks such as Question Answering, already targeted in traditional Knowledge Graphs such as the one by the Lynx project [5], could be improved for the time-related questions, that could be much more precise and complex. Summarization tasks can also benefit from an event-based representation, since event-based summarization techniques have already been explored in literature [60, 61]. Moreover, reasoning systems and search engines can make use of event arguments in order to improve their results, being possible to refine event-based searches such as "Give me cases about car accidents where the driver was a man" or "Cases where the accident happened after a criminal action".

[27] https://github.com/mnavasloro/FromTimeToTime

[28] https://github.com/openlink/virtuoso-opensource

[29] https://github.com/blazegraph/database/releases/tag/BLAZEGRAPH_2_1_6_RC

Finally, one of the most interesting application for law firms would be pattern recognition. The possibility of looking for previous judgments with similar narratives in terms of events and temporal spans would an extremely valuable tool for legal practitioners, since it would really enhance the search of jurisprudence and would help to plan possible timelapses in the resolution of the legal procedure.

7. Conclusions

In this paper we have presented a series of tools that allow to create a Legal Event-Based Knowledge Graph. Our approach is based on the assumption that the relevant events extracted from a legal judgment describe it in a way powerful to be exploited.

First contribution of the present work is the WhenTheFact event extractor, able to annotate relevant legal events taking into account the structure of a legal judgment. Once the annotation is done, it is sent to the fromTimeToTime converter, a tool able to output a document in different annotation formats and as RDF. The tool converts the xml annotated document into a turtle file that includes both information about the document and its annotations and a special representation of all the events detected in the document, based on a ontology created for this purpose. Finally, the output is used to populate an Event-Based Knowledge Graph, that can be later queried from a SPARQL endpoint with some predefined queries to facilitate the task to people foreign to the Semantic Web. All the resources are freely available and can be combined with other tools in order to replicate or improve the functionality.

Next steps include enriching the knowledge graph with metadata not related to the temporal information, such as the actors involved in the cases. This for instance would help to solve correference, since currently we just get the textual mention, that can consist of pronouns. Once this is achieved, queries will be able to retrieve for instance the timeline of one actor's involvement in a case.

Also multilinguality is currently being explored. One of the document sources, the European Court of Justice, allows to download most documents in all the languages of the European Union. A very interesting application of our annotations, currently covering just the English language, would be to find the equivalent to the event annotated in English. Although several approaches have been tested already, none of them has been good enough to guarantee acceptable results for all the languages.

Finally, since one of the target users of our contributions are legal practitioners, usually foreign to SPARQL, one of the planned improvements is to adapt Natural Language queries to SPARQL translators to the legal domain terminology. This would help boosting the use of our technologies, as well as to bring the Semantic Web technologies and the legal domain closer together.

References

[1] M. Rospocher, M. van Erp, P. Vossen, A. Fokkens, I. Aldabe, G. Rigau, A. Soroa, T. Ploeger and T. Bogaard, Building event-centric knowledge graphs from news, *Journal of Web Semantics* **37-38** (2016), 132–151. doi:https://doi.org/10.1016/j.websem.2015.12.004. https://www.sciencedirect.com/science/article/pii/S1570826815001456.

[2] S. Gottschalk and E. Demidova, Eventkg: A multilingual event-centric temporal knowledge graph, in: *European Semantic Web Conference*, Springer, 2018, pp. 272–287.

[3] J. Wu, X. Zhu, C. Zhang and Z. Hu, Event-centric Tourism Knowledge Graph—A Case Study of Hainan, in: *Knowledge Science, Engineering and Management*, G. Li, H.T. Shen, Y. Yuan, X. Wang, H. Liu and X. Zhao, eds, Springer International Publishing, Cham, 2020, pp. 3–15. ISBN 978-3-030-55130-8.

[4] E. Filtz, Building and processing a knowledge-graph for legal data, in: *European Semantic Web Conference*, Springer, 2017, pp. 184–194.

[5] V.R. Doncel and E.M. Ponsoda, LYNX: Towards a Legal Knowledge Graph for Multilingual Europe, *Law in Context. A Socio-legal Journal* **37**(1) (2020), 1–4.

[6] M. Navas-Loro and C. Santos, Events in the legal domain: first impressions, in: *Proceedings of the 2nd Workshop on Technologies for Regulatory Compliance co-located with the 31st International Conference on Legal Knowledge and Information Systems (JURIX 2018), Groningen, The Netherlands, December 12, 2018.*, 2018, pp. 45–57. http://ceur-ws.org/Vol-2309/05.pdf.

[7] M. Navas-Loro, E. Filtz, V. Rodríguez-Doncel, A. Polleres and S. Kirrane, TempCourt: evaluation of temporal taggers on a new corpus of court decisions, *The Knowledge Engineering Review* **34** (2019), e24. doi:10.1017/S0269888919000195.

[8] J. Pustejovsky, K. Lee, H. Bunt and L. Romary, ISO-TimeML: An International Standard for Semantic Annotation., in: *LREC*, Vol. 10, 2010, pp. 394–397.

[9] R. Sauri et al., Annotating Time Expressions in Spanish TimeML Annotation Guidelines (2010).

[10] L. Ferro et al., TIDES temporal annotation guidelines-version 1.0.2, Technical Report, MITRE Corporation, 2001.

[11] S. Hellmann et al., NIF: An ontology-based and linked-data-aware NLP Interchange Format (2012).

[12] W. Styler IV et al., Temporal Annotation in the Clinical Domain, *Transactions of ACL* **2** (2014), 143–154.

[13] D. Ahn, The stages of event extraction, in: *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, Association for Computational Linguistics, 2006, pp. 1–8.

[14] J. Aguilar, C. Beller, P. McNamee, B. Van Durme, S. Strassel, Z. Song and J. Ellis, A comparison of the events and relations across ace, ere, tac-kbp, and framenet annotation standards, in: *Proceedings of the Second Workshop on EVENTS: Definition, Detection, Coreference, and Representation*, 2014, pp. 45–53.

[15] ACE (Automatic Content Extraction) English Annotation Guidelines for Events.

[16] A. Bies, Z. Song, J. Getman, J. Ellis, J. Mott, S. Strassel, M. Palmer, T. Mitamura, M. Freedman, H. Ji et al., A comparison of event representations in deft, in: *Proceedings of the Fourth Workshop on Events*, 2016, pp. 27–36.

[17] Z. Song, A. Bies, S. Strassel, T. Riese, J. Mott, J. Ellis, J. Wright, S. Kulick, N. Ryant and X. Ma, From light to rich ere: annotation of entities, relations, and events, in: *Proceedings of the the 3rd Workshop on EVENTS: Definition, Detection, Coreference, and Representation*, 2015, pp. 89–98.

[18] T. Mitamura, Y. Yamakawa, S. Holm, Z. Song, A. Bies, S. Kulick and S. Strassel, Event nugget annotation: Processes and issues, in: *Proceedings of the The 3rd Workshop on EVENTS: Definition, Detection, Coreference, and Representation*, 2015, pp. 66–76.

[19] Z. Song, A. Bies, S. Strassel, J. Ellis, T. Mitamura, H.T. Dang, Y. Yamakawa and S. Holm, Event nugget and event coreference annotation, in: *Proceedings of the Fourth Workshop on Events*, 2016, pp. 37–45.

[20] T. O'Gorman, K. Wright-Bettner and M. Palmer, Richer Event Description: Integrating event coreference with temporal, causal and bridging annotation, in: *Proceedings of the 2nd Workshop on Computing News Storylines (CNS 2016)*, 2016, pp. 47–56.

[21] E. Marsh and D. Perzanowski, MUC-7 evaluation of IE technology: Overview of results, in: *Seventh Message Understanding Conference (MUC-7): Proceedings of a Conference Held in Fairfax, Virginia, April 29-May 1, 1998*, 1998.

[22] C.F. Baker, C.J. Fillmore and J.B. Lowe, The Berkeley FrameNet Project, in: *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 1*, ACL '98/COLING '98, Association for Computational Linguistics, USA, 1998, pp. 86–90–. doi:10.3115/980845.980860.

[23] R. Sprugnoli and S. Tonelli, Novel Event Detection and Classification for Historical Texts, *Computational Linguistics* **45**(2) (2019), 229–265.

[24] P.A. Schrodt, Cameo: Conflict and mediation event observations event and actor codebook, *Pennsylvania State University* (2012).

[25] V. Danilova, *Linguistic support for protest event data collection*, Universitat Autònoma de Barcelona,, 2015.

[26] V. Ermolayev, S. Batsakis, N. Keberle, O. Tatarintseva and G. Antoniou, Ontologies of Time: Review and Trends., *International Journal of Computer Science & Applications* **11**(3) (2014).

[27] M. Fernández-López and A. Gómez-Pérez, Searching for a Time Ontology for Semantic Web Applications, in: *Proceedings of the Formal Ontology in Information Systems. Third Interntacional Conference (FOIS-2004)*, IOS Press, 2004, Ontology Engineering Group ? OEG. http://oa.upm.es/5493/.

[28] A. Scherp, T. Franz, C. Saathoff and S. Staab, A core ontology on events for representing occurrences in the real world, *Multimedia Tools and Applications* **58**(2) (2012), 293–331.

[29] F. Li, J. Du, Y. He, H.-Y. Song, M. Madkour, G. Rao, Y. Xiang, Y. Luo, H.W. Chen, S. Liu et al., Time event ontology (TEO): to support semantic representation and reasoning of complex temporal relations of clinical events, *Journal of the American Medical Informatics Association* **27**(7) (2020), 1046–1056.

[30] R. Segers, E. Laparra, M. Rospocher, P. Vossen, G. Rigau and F. Ilievski, The predicate matrix and the event and implied situation ontology: Making more of events, *Proceedings of GWC2016* (2016).

[31] R. Sprugnoli and S. Tonelli, One, no one and one hundred thousand events: Defining and processing events in an inter-disciplinary perspective, *Natural Language Engineering* **23**(4) (2017), 485–506–. doi:10.1017/S1351324916000292.

[32] T.F. Gordon, The legal knowledge interchange format (LKIF), *Estrella deliverable d4* **1** (2008).

[33] R. Hoekstra, J. Breuker, M. Di Bello and A. Boer, LKIF Core: Principled Ontology Development for the Legal Domain., 2009, pp. 21–52. doi:10.3233/978-1-58603-942-4-21.

[34] M. Verhagen et al., Automating Temporal Annotation with TARSQI, in: *Proceedings of the ACL 2005 on Interactive Poster and Demonstration Sessions*, ACL, 2005, pp. 81–84.

[35] H. Llorens et al., Tipsem (English and Spanish): Evaluating CRFs and Semantic Roles in TempEval-2, in: *Proceedings of the Workshop SemEval*, ACL, 2010, pp. 284–291.

[36] N. Chambers et al., Dense event ordering with a multi-pass architecture, *Transactions of ACL* **2** (2014), 273–284.

[37] S. Bethard, ClearTK-TimeML: A minimalist approach to TempEval 2013, in: *Proceedings of the Workshop SemEval 2013*, ACL, 2013, pp. 10–14.

[38] D. Das, D. Chen, A.F. Martins, N. Schneider and N.A. Smith, Frame-semantic parsing, *Computational linguistics* **40**(1) (2014), 9–56.

[39] S. Swayamdipta, S. Thomson, C. Dyer and N.A. Smith, Frame-Semantic Parsing with Softmax-Margin Segmental RNNs and a Syntactic Scaffold, *arXiv preprint arXiv:1706.09528* (2017).

[40] M. Roth and M. Lapata, Context-aware frame-semantic role labeling, *Transactions of the Association for Computational Linguistics* **3** (2015), 449–460.

[41] M. Alam, A. Gangemi, V. Presutti and D.R. Recupero, Semantic role labeling for knowledge graph extraction from text, *Progress in Artificial Intelligence* (2021), 1–12.

[42] M. Gardner, J. Grus, M. Neumann, O. Tafjord, P. Dasigi, N.F. Liu, M. Peters, M. Schmitz and L.S. Zettlemoyer, AllenNLP: A Deep Semantic Natural Language Processing Platform, 2017.

[43] R. Agerri et al., IXA pipeline: Efficient and Ready to Use Multilingual NLP tools., in: *LREC*, Vol. 2014, 2014, pp. 3823–3828.

[44] N. Lagos, F. Segond, S. Castellani and J. O'Neill, Event extraction for legal case building and reasoning, in: *International Conference on Intelligent Information Processing*, Springer, 2010, pp. 92–101.

[45] K.T. Maxwell, J. Oberlander and V. Lavrenko, Evaluation of semantic events for legal case retrieval, in: *Proceedings of the WSDM'09 Workshop on Exploiting Semantic Annotations in Information Retrieval*, ACM, 2009, pp. 39–41.

[46] G. Sierra, G. Bel-Enguix, G. López-Velarde, R. Saucedo and L. Rivera, Event Extraction from Legal Documents in Spanish, in: *Workshop on Language Resources and Technologies for the Legal Knowledge Graph, LREC 2018. Miyazaki, Japan*, 2018.

[47] A. Bertoldi, R. Chishman, S.J. Rigo and T.D. Minghelli, Cognitive Linguistic Representation of Legal Events. Towards a semantic-based legal information retrieval, in: *COGNITIVE 2014 : The Sixth International Conference on Advanced Cognitive Technologies and Applications*, 2014.

[48] N. Kiyavitskaya, N. Zeni, T.D. Breaux, A.I. Antón, J.R. Cordy, L. Mich and J. Mylopoulos, Automating the extraction of rights and obligations for regulatory compliance, in: *International Conference on Conceptual Modeling*, Springer, 2008, pp. 154–168.

[49] S. Ingolfo, I. Jureta, A. Siena, A. Perini and A. Susi, Nòmos 3: Legal Compliance of Roles and Requirements, in: *Conceptual Modeling*, E. Yu, G. Dobbie, M. Jarke and S. Purao, eds, Springer International Publishing, Cham, 2014, pp. 275–288. ISBN 978-3-319-12206-9.

[50] V. Naik et al., Reasoning in Legal Text Documents with Extracted Event Information, *International Journal of Computer Applications* **28** (2011), 8–-13.

[51] K. Ramakrishna et al., A Novel Model for Timed Event Extraction and Temporal Reasoning In Legal Text Documents, *International Journal of Computer Science and Engineering Survey* **2** (2011), 39–48.

[52] E. Filtz, M. Navas-Loro, C. Santos, A. Polleres and S. Kirrane, Events Matter: Extraction of Events from Court Decisions, in: *Proceedings of the 33rd International Conference on Legal Knowledge and Information Systems (JURIX 2020).*, 2020, pp. 33–42. doi:10.3233/FAIA200847.

[53] M. Butt, *The light verb jungle: still hacking away*, in: *Complex Predicates: Cross-linguistic Perspectives on Event Structure*, M. Amberber, B. Baker and M. Harvey, eds, Cambridge University Press, 2010, pp. 48–78–. doi:10.1017/CBO9780511712234.004.

[54] E. Loper and S. Bird, NLTK: The Natural Language Toolkit, in: *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics - Volume 1*, ETMTNLP '02, Association for Computational Linguistics, USA, 2002, pp. 63–70–. doi:10.3115/1118108.1118117.

[55] M. Navas-Loro and V. Rodríguez-Doncel, Annotador: a temporal tagger for Spanish, *Journal of Intelligent & Fuzzy Systems* **39** (2020), 1979–1991, 2. doi:10.3233/JIFS-179865.

[56] C.D. Manning et al., The Stanford CoreNLP Natural Language Processing Toolkit, in: *Proceedings of the 52nd Annual Meeting of the ACL, System Demonstrations*, 2014, pp. 55–60.

[57] M. Poveda-Villalón, A. Gómez-Pérez and M.C. Suárez-Figueroa, OOPS! (OntOlogy Pitfall Scanner!): An On-line Tool for Ontology Evaluation, *International Journal on Semantic Web and Information Systems (IJSWIS)* **10**(2) (2014), 7–34.

[58] D. Garijo, WIDOCO: a wizard for documenting ontologies, in: *International Semantic Web Conference*, Springer, Cham, 2017, pp. 94–102.

[59] S. Gottschalk and E. Demidova, EventKG+TL: creating cross-lingual timelines from an event-centric knowledge graph, in: *European Semantic Web Conference*, Springer, 2018, pp. 164–169.

[60] E. Filatova and V. Hatzivassiloglou, Event-based extractive summarization, in: *Event-basedextractive summarization. In-Proceedings of ACL Work-shop on Summarization*, Vol. 111, 2004.

[61] L. Marujo, R. Ribeiro, A. Gershman, D.M. de Matos, J.P. Neto and J. Carbonell, Event-based summarization using a centrality-as-relevance model, *Knowledge and Information Systems* **50**(3) (2017), 945–968.