AN OWL ONTOLOGY SET REPRESENTING JUDICIAL INTERPRETATIONS

Marcello Ceci¹
Aldo Gangemi²

¹CIRSFID, University of Bologna
²ISTC-CNR, Rome, Italy
m.ceci@unibo.it
gangemi@ip.rm.cnr.it

Abstract: The paper shows how to model judgments starting from the decision's text and creating an ontology which represents the interpretations performed by the judge while conducting its discourse towards the adjudication. The modelling is carried out using the OWL standard in order to be compliant with the ontology layer of the Semantic Web Cake. The goal of this approach is to build a complete ontology framework capable of detecting and modelling jurisprudence directly from the text, performing some basic reasoning on the knowledge base and providing semantically rich information for the logic and proof layers.

1. Introduction

Precedent is a main element of legal knowledge worldwide: by settling conflicts and sanctioning illegal behaviours, judicial activity enforces law provisions within the national borders, therefore supporting the validity of laws as well as the sovereignty of the government that issued them. Moreover, precedents (or case-law) are a fundamental source for law interpretation and it paradoxically happens that the exercise of jurisdiction can influence the scope of the same norms it has to apply, both in common law and civil law legal systems – even if to different extents. The AI & LAW research community has presented significant outcomes in this topic since the 1980s, with different approaches: legal case-base reasoning [1], [5], ontologies [12], and more recently also argumentation [8], [9].

The goal of the present research is to define a complete framework for the precedent modelling following the Semantic Web cake, starting from the text and filling the gap through metadata modelling up to the rules [3], [13]. Cornerstone of the framework is the ontology, intended in its computer science meaning: a shared vocabulary and taxonomy which models a domain of knowledge by defining objects, concepts, their properties and their relations. A formalization of the main structure of the case-law, the metadata connected with the judicial legal concepts, and the ontology constitute the basis of a semantic tool which enriches the XML mark-up of precedents and supports legal reasoning. We believe that the new features of OWL 2.0 could unlock potentialities for legal concept modelling, to be combined with that of rules. Our aim is hence to formalize the legal concepts and the argumentation patterns contained in the judgment in order to check, validate and reuse the discourse of the judge and the argumentation he produces, as expressed by the text. To achieve this, four different models are necessary:
a) a *document metadata structure*, capturing the main parts of the judgment to create a bridge between text and semantic annotation of legal concepts;

b) a *legal core ontology*, modelling the abstract legal concepts and the institutions to capture the main parts of the rule of law [16];

c) a *legal domain ontology*, modelling the main legal concepts in the specific domain concerned by the case-law (e.g. contracts, e-commerce, tort law, etc.);

d) an *argumentation system* [6], modelling the structure of argumentation (arguments, counterarguments, premises, conclusions, rebuttal, etc.).

The paper, following this path, describes the structure of the core and domain ontologies - points b) and c) - which have been designed to organize the metadata coming from the decision's text, and to infer relevant knowledge about precedents.

This approach allows, under the technical point of view, to reach results such as:

- **IR and query:** it becomes possible to perform some very complex querying, applying the Semantic Web techniques (SPARQL-DL) on the qualified parts of the judgment text. It is possible, for example, to make the following request: "give me all the judgments in the last year, with a dissenting opinion, in the e-commerce field and where the main argument of the decision is the application of Consumer Law, art. 122";

- **NLP:** the machine can detect relevant parts of the speech using the semantic annotation and the ontology;

- **Rules:** the ontologies provide information for the rule engine to perform the legal case-based reasoning [6].

### 2. Legal Ontology Methodology

This research is based on a middle-out methodology: bottom-up for capturing and modelling the core and legal domain ontology classes and top-down for modelling the argumentation theory components and their relationships [7]. The research relies on a sample of Italian case law constituted by a set of 27 decisions of different grade (tribunal, court of appeal, Cassation Court) concerning the legal field of oppressive clauses in Consumer Contracts. The matter is specifically disciplined in the Italian "Codice del Consumo" (Consumer Code) as well as in most foreign legal systems, which will allow an extension of the research to foreign decisions (and laws).

This domain is particularly fit for this research because it includes situations where the strong rules and the strict deductive logic are not sufficient to cope with the legal reasoning of the judge. We need to evoke defeasible logics to represent the legal rules concerning the subject. In fact, many norms concerning contracts are not absolutely mandatory: they can be overlapped by different discipline through specific agreements between the parties. The problem of representing "defeasible" rules, in fact, is a core problem in legal knowledge representation. Exploring how the OWL 2.0 could prepare the background for the application of defeasible logic is therefore a main goal of the present research: in fact, the OWL language (even in its 2.0 version) is not fully fitted for managing defeasibility, being only able to capture the static factual and legal
knowledge to be reused in the rule layer. Nevertheless the gap between ontology and rules is often underestimated, and the benefits coming from the OWL 2.0 computation are neglected. For this reason, well aware of the limitations of the OWL 2.0 in representing defeasible logics, we have the intention to stress the axiom definitions as much as possible to improve performances, computability, and management of the classes over the time, and to foster the Semantic Web tools and applications which are already available in this sector.

Under a different perspective, the law of contracts is an interesting field because the (either automatic or manual) markup of contract parts allows the highlight of single clauses and their comparison to general rules as well as to case law concerning the matter. These possibilities can be used to introduce a semi-automatic check compliance of a contract draft.

The knowledge is modelled through a set of two ontologies [14]:

• a Core Ontology describing the constituents of a precedent in terms of general concepts, through an LKIF-Core [4], [11] extension;
• a Domain Ontology representing the concepts and the rules expressed by (and used in) the Italian ”Codice del Consumo” and in artt. 1241-1242 Civil Code, as well as all relevant knowledge extracted from a set of Italian sentences containing interpretation of private agreements in the light of those laws.

The software used to model the ontology (and from which the images of this paper are taken) is Protégé 4.1.0, supporting many of the features introduced by OWL 2.0.

2.1 Core Ontology

The core ontology introduces the main concepts and interactions in the legal domain, defining the classes which will be later filled with information taken from the judicial decisions. Even though the core ontology should be domain-generic and not modeled upon a specific legal subject, the model presented here was conceived to successfully represent the interaction in the civil law subject, when contracts, laws and judicial decisions come into play. Obviously, it will be necessary to add further classification prior to successfully expand the ontology set to a different domain (es. Public contracts, administrative law, tort law).

The backbone of the Core Ontology is represented by three LKIF-Core classes:

![Core Ontology's specification of LKIF-Core](image)
• **Qualificatory Expression** (subclass of Mental_Entity>Mental_Object>Proposition>Expression>Legal_Expression) represents a legal expression which ascribes a legal status to a person or an object (for example, “x is a citizen”, “x is an intellectual work”, “x is a technical invention”).

• **Qualification** (Mental_Entity>Mental_Object>Proposition) expresses e.g. a judgement: the thing qualified by the qualification is comparable to something else.

• **Qualified** represents anything which is the object of some qualification.

Since the main object to be represented in the present set of ontologies is the normative/judicial qualification brought forward by performative utterances (contractual agreements, legal rules and judicial interpretations), the classes presented above constitute the nucleus of the Core Ontologies. The LKIF-Core Qualification and Qualified classes are linked only by a single property (qualifies/qualified_by), but what we rather want to model is an n-ary relation between (1) a qualifying expression, (2) the kind of qualification and (3) the object being qualified. In order to represent this, the property “qualifies” has been forked into two new properties: “considers” and “applies”. The first one, “considers” (modeled as superclass of the LKIF-Core properties “evaluates”, “allows”, “disallows”) represents the object of the qualification. The second property, “applies”, shows towards which concept the qualification is made. For example, a Contractual_Agreement considers a Material_Circumstance and applies a Legal_Status; a Legal_Rule considers a Legal_Status and applies a Legal_Consequence; a Judicial_Interpretation considers a Material_Circumstance and applies a Legal_Status; an Adjudication considers a Judicial_Claim and applies a Judicial_Outcome.

![Diagram](image_url)

Fig. 2 – Interactions between qualifications, qualifiers, and qualified things.

**Qualifying Legal Expressions** - To overcome the limited expressivity of the original LKIF-Core classes a new conceptual class called
“Qualifying_Legal_Expression” has been conceived, putting together the characteristics of the Qualificatory_Expression and Qualification classes, enhanced by the fork of the qualifies property. This class represents the formalization of dispositions, such as the three legal expressions involved in contract law-related judicial decisions: Contractual_Agreement, Legal_Rule and Judgement.

As Qualificatory_Expression sub-classes, the Qualifying Legal Expressions contain all information related to their original “speech act”: its semantic bonds with the externalization, the legal power and the agents ensure a complete representation of all aspects that may come into play when facing a legal issue (the legitimacy of the legislative body/court/legal party, the characteristics of the corresponding legal document, the identity/characteristics of people/bodies involved...). Their main properties are “medium” and “attitude” (see below for a specification of the Medium, Attitude and Agent classes).

As Qualification subclasses, the Qualifying Legal Expressions contain all information related to the effects they have in the legal world: the legal categories/obligations/legal effects they create, modify or repeal. A subdivision can be made between one direct subclass (Judgement, which in this perspective is furtherly divided into the Judicial_Interpretation and Adjudication subclasses) and two subclasses of Norm (Legal_Rule and Contractual_Agreement). As explained before, the property “qualifies” - linking the qualifying expression to the Qualified expression - has been forked into two new properties: “considers” and “applies”, representing respectively the direct object and the “destination” of the qualification.
Qualified Expressions - All the ranges of the “considers” and “applies” properties presented above are subclasses of the Qualified class. Its subclasses are Normatively_Qualified, a class already present in LKIF-Core, and Judicially_Qualified, created anew.

Normatively_Qualified expressions include Material_Circumstance, Legal_Status and Legal_Consequence. They represent the expressions that can be directly bound to a Norm: while Material_Circumstance represents any fact or act which is taken into consideration by the Norm, Legal_Status represents an institutional fact (i.e. fulfillment of contract, oppressive clause, contract breach) that is normally considered by a Legal_Rule and applied by a Contractual_Agreement or a Judgement. As we will see, the link between a Contractual_Agreement and the Legal_Status it applies is a “weak” link until a Judicial_ Interpretation has confirmed (or denied) it. Finally, Legal_Consequence represents the sanction provided by the law in the presence of some Legal_Status or Material_Circumstance. It covers all cases when the Legal_Rule considers some Normatively_Qualified expression, but does not simply allows, disallows or evaluates it.
Judicially_Qualified expressions include Judicial_Claim, Judicial_Outcome and all elements taken into consideration during a legal proceeding (i.e. Contractual_Agreement, but also Legal_Rule, especially in Cassation Court and Constitutional Court sentences). Judicial_Claim is the claim of the legal proceeding. It is considered_by an Adjudication, the answer of the judge to the claim (subclass of Qualification>Judgement). The content of the answer (rebuttal/acceptation of the claim or any other possible outcome foreseen by the law) is represented by the Judicial_Outcome class, applied_by the Adjudication. So the representation is the following: a Judicial_Claim is considered_by an Adjudication that applies a Judicial_Outcome.

The judged_as Property Chain - The miscellaneous elements that can be taken into consideration during a legal proceeding are included in the Judicially_Qualified class as long as they are actually considered_by some Judicial_Interpretation. So, for example, a Contractual_Agreement can be considered_by some Judicial_Interpretation who applies some Legal_Status to it (i.e. the agreement is oppressive, is inefficacious, represents an arbitration clause, is specifically signed by both parties). In these cases, a OWL 2.0 property chain directly links the Contractual_Agreement to the Legal_Status judicially applied to it. This “strong” link, represented by the property "judged_as", is the the fundamental information that we want to represent – and manage – through this set of ontologies.

Mediums, Propositional Attitudes and Agents - these LKIF-Core classes describe the background of an Expression. The Medium class identifies the support through which the proposition is expressed. It does not represent the material support of the Expression instance but rather its genus (Contract, Precedent, Code). The Propositional_Attitude class was specified with the Jurisdiction, Law_Declaration and Agreement subclasses, representing the enabling powers that stand behind a Judgement, a Legal_Rule and a Contractual_Agreement, respectively. On the contrary, to represent the authors of a Qualifying Legal Expression there was no need to specify the subclasses of Agent already present in LKIF-Core (Person and Organization). This knowledge about agents and attitudes can be important in some judicial cases: i.e. if a claim is based on the lack of contractual power by one of the parties, or on the identity/characteristics of a part, or on the lack of force by some law or other regulation (which can in turn depend by the lack of legitimacy of one of its authors). Also the modeling of roles (already present in LKIF) can be very useful in representing critical factors of particular precedents.
Modularity of the Core Ontology - The expansion brought by the Core Ontology to the LKIF-Core concepts is currently oriented to the representation of the elements involved in civil-law cases regarding contract law. Nevertheless, the Core Ontology provides general – and relatively open - categories for this kind of judicial activity to be represented, and can therefore be considered as a core to be “expanded” with categorization from other branches of law, but not to be “substituted”, since the basic concepts introduced here may come into play also in judgements concerning different subjects.

Fig. 7 – The Core Ontology graph.

Fig. 8 – semantic relations between represented knowledge
2.2 Domain Ontology

Following the structure outlined in the Core Ontology, the knowledge taken from judicial decisions is represented in the Domain Ontology under the perspective of the contents of the documents involved. The modeling should be carried out manually by experts in the legal subject, as automatic information retrieval and machine learning techniques do not yet ensure a sufficient level of accuracy: the activity of building a domain ontology is very similar to that of writing a piece of legal doctrine, thus it should be manually achieved in such a way as to maintain a reference from the model to the author, while at the same time keeping an open approach (i.e. allowing different modeling of the same concept by different authors).

**Modeling of the law** - the laws involved in the domain are represented into the ontology in a quite complex fashion, in order to allow full expressivity of their deontic powers. First of all, they are represented as instances of the **Legal Rule** class, whose only stated property is to apply the **Legal Consequence** indicated in the head of the legal rule (fig. 9). The reasoner will infer knowledge about the rule, linking it (through the **considers** property) to the contractual agreements which fall under the scope of that norm.

![Fig. 9 - Stated property assertion of a Legal Rule instance](image)

![Fig. 10 – Visualization of the expression class, highlighting the subclasses of Contractual_Agreement introduced by the legal rules.](image)
Legal rules are also represented through anonymous subclasses of the Normatively_Qualified class, called Relevant_Ex<rulename> (ex is the latin proposition for indicating a source). An axiom stating the requirements for an instance to be relevant under the legal rule is included in the description of the class, as well as an equivalence linking each of its instances to the legal rule, through the property “considered_by” (fig. 11). Please notice that, in the graph visualizer (fig. 10), these anonymous classes are classified under the Contractual_Agreement class: that is, because the effect of the legal rule in this context is to enrich the definition of Contractual_Agreement, adding subdivisions which depend on the legal framework created by the legal rules of the domain.

**Modeling of the contract** - A contract is a composition of one or more Contractual_Agreements (a Contract for the whole, multiple Contract_Clauses for its parts), each of which represents an obligation arising from the contract. All components of the contract share the same Attitude (the “meeting of minds” between the Agents) and Medium (the kind of support in which the expression is contained). A Contractual_Agreement normally considers some Material_Circumstance and applies some Legal_Status to it.

In the actual model, the material circumstances considered by the contractual agreement were not included: that is, because this has no relevance when capturing the sheer interpretation instances these agreement undergo: it would rather become useful when delving deeper into the single interpretation, capturing the smaller factors which led to that specific interpretation.

**Modeling of the decision** - The Judgement class includes an instance identifying the case as a whole (the precedent) and several ones identifying its parts: at least an Adjudication and zero or more Judicial_Interpretations. They share a common attitude (a Jurisdiction power) a Precedent medium and some agents (claimant, defendant, and court). The Adjudication contains the Judicial_Outcome of the Judicial_Claim (it considers the claim and applies the outcome), while the Judicial_Interpretation considers a
Material_Circumstance and applies one or more Legal_Status (and zero or more Precedents) to it.
The precedents cited by the judge in the decision are added directly to the Interpretation instance: the reasoner is then capable of distinguishing between legal statuses and precedents, the latter being searchable in queries and other information retrieval applications. Rules expressed by precedents (i.e., if a clause is signed through a recall at the end of the document, it is specifically signed) can be modeled in the same way as legal rules are.

**Reasoning on the knowledge base** - To check the consistency of this knowledge we will use Hermit 1.3.6 queries. This tool was built to extract data from the OWL ontology, but could also be used to check if the ontology gives a unique and correct answer to some formalized question (i.e. asking about the validity of some proof, or about the qualification of factual events under legal principles). When a Contractual_Agreement (the expression brought by a Contract_Clause) is considered by some Judicial_Interpretation, the ontology gathers all relevant information on the documents involved: contract parties, judicial actors, legal status applied to the agreement (eventually in comparison to the one suggested by the contract/judicial parties), the law rules which are relevant to the legal status, the final adjudication of the claim, the part played in it by the interpreted agreement, and so on. In this perspective, the citations to case law constitute the first element to be represented using Akoma Ntoso metadata (Shepardizing) to classify interpretations and argumentations [2].

The first objective for gathering all this semantically-rich information is advanced querying on precedents, but more can be achieved by combining different Judicial_Interpretations with knowledge coming from the contract and the applicable law: the ontology reasoner is in fact capable of predicting – to some extents – the outcome of the judge (i.e. predicting that a clause will be judged as valid/invalid) and to run inferences about the agreement (i.e. as interpreted, the clause is irrelevant for the whole Italian Consumer Law/for the legal rule contained in article 1342 comma 2 of Italian Civil Code).

1 http://hermit-reasoner.com/
This inferred knowledge is important for two reasons: a. by “predicting” the judge’s final statement on the clause (even if not that on the claim), this knowledge represents a logic and deontic check on the legal consequences the judge takes from its interpretation; b. it gives a fundamental element for the argumentation system to support the explanation of the adjudication of the claim. The argumentation system, in fact, will be able to use the (stated and inferred) elements of the decision’s groundings to support and explain the Adjudication contained in the last part of the judgment.

3. OWL 2.0

OWL 2 introduces several features to the original Web Ontology Language, some of which allow a richer representation of knowledge, mostly when dealing with properties and datatypes. Some of these would be useful, but also lead to a great increase of complexity in the models: for example, disjointness between properties has been introduced, but in order to exploit this feature it would be necessary to create as many properties as possible statuses, which in turn would greatly affect computability.

Nevertheless, at least two new features concerning properties deserve attention because they could enhance expressivity without affecting (or even reducing) the complexity of the model built so far:

**Keys:** An HasKey axiom states that each named instance of a class is uniquely identified by a (data or object) property or a set of properties - that is, if two named instances of the class coincide on values for each of key properties, then these two individuals are the same. This feature can be useful for identifying the unique “actors” of the judicial claim, such as the parties, the contract, the norm, and the decision itself.

**Property Chains:** The OWL 2 construct ObjectPropertyChain in a SubObjectPropertyOf axiom allows a property to be defined as the composition of several properties. Such axioms are known as complex role inclusions in SROIQ. In the present ontology set, the property chain “judged_as = considered_by o applies” is used in two different ways (in interpretations and rule applications) to create a strong interpretational link between a material circumstance and its status. When a Judicial_Interpretation considers a Material_Circumstance and applies a Legal_Status, the judged_as property chain comes into play and creates a direct link between the circumstance and its status, that link being distinguished from the “weak” one introduced directly by the contract (represented by the property applies). Reasoners can therefore treat these two links accordingly. Secondarily, as already said, the legal rule axiom work through an “anonymous qualified class” which links all relevant expression to the legal rule instance through the considered_by property, and the legal rule applies a legal consequence. The judged_as property chain unifies the two properties (from the qualified expression to the law, and from the law to the legal consequence) and brings their semantics to the surface by creating a direct property linking the contract clause to its status (judged_as Inefficacy). A better exploitation of the OWL 2.0 property chains could lead to an ever more direct and complete solution, mainly by removing the need
for the anonymous subclass in order to identify the clause instances considered by the relevant law.

4. An Example of Precedent Modeling

The modeling of the ontology is explained here through a simple example of data insertion and knowledge management by the Domain Ontology:

In the decision given by the 1st section of the Court of Piacenza on July 9th, 2009, concerning contractual obligations between two small enterprises (New Edge sas and Fotovillage srl, from now on α and β), the judge had to decide whether clause 12 of α/β contract, concerning the competent judge (Milan instead of Piacenza) could be applied. The judge cites art. 1341 comma 2 of Italian Civil Code who says “a general and unilateral clause concerning a competence derogation is invalid unless specifically signed”. In the contract signed by the parties there is a distinct box for a “specific signing” where all the clauses of the contract are recalled (by their number). The judge, with the support of precedents (he cites 9 Cassation Court sentences) interprets the “specific signing” as not being fulfilled through a generic recall of all the clauses, and therefore declares clause 12 of α/β contract invalid and inefficacious. The claim of inefficacy of clause 12, brought forward by α, is thus accepted, undercutting the claim of a lack of competence by the judge of Piacenza, brought forward by β, which is rejected.

Fig. 15 - The example graph

---

In order to represent the knowledge contained in that sentence, we need to model three documents: Art. 1341 comma 2 of Italian Civil Code, the contract between the two enterprises α and β, and the decision by the Court of Piacenza.

**Modeling of the law** – following is the law disposition involved in the judicial decision:

Article 1341 comma 2 of Italian Civil Code -- Clauses concerning arbitration, competence derogation, unilateral contract withdrawal, and limitations to: exceptions, liability, responsibility, and towards third parties, are inefficacious unless they are specifically signed by writing.

The disposition is represented as a Qualifying Legal Expression (Legal_Rule) called “art1341Co2” (with a Code medium, a Law_Declaration attitude and a Parliament as agent) and the qualified class Relevant_ExArt1341Co2. Any individual which has the characteristics required by the law is considered by the Legal_Rule, which in turn allows/disallows/evaluates or applies some Legal_Consequence to it. In the example, each Contractual_Agreement which applies “General”, “Unilateral”, “NotSpecificallySigned” and an Oppressive_Status will be considered by “art1341Co2”, which in turn applies the Legal_Consequence of “invalidityExArt1341Co2”. The individuals “competentJudge” and “notSpecificallySigned” are thus created as Legal_Statuses that can be considered_by a Legal_Rule and applied_by a Contractual_Agreement, and the individual “invalidityExArt1341Co2” is created as a Legal_Consequence applied_by the Legal_Rule “art1341Co2”.

**Modeling of the contract clause**– The Contract_Clause “α/βClause12” is created and linked to a Contractual_Agreement which applies the Legal_Statuses of “General”, “Unilateral” and “CompetenceDerogation”. This is done because there is no argue between the parties about whether clause 12 concerns a competence derogation. However, as explained before, this kind of link is a “weak” one, considering that the contractual parties have no power to force a legal status into a contract, and that reconducting a
contractual agreement to the legal figure it evokes is the main activity brought forward by judicial interpretation in the contracts field. For this reason, the property “applies” related to a Legal_Status is weak when its domain is a Contractual_Agreement, and prone to be overridden by a contrasting application performed by a Judicial_Interpretation.

**Modeling of the decision** - The Judgment instance is created, as well as its components (single interpretation instances, adjudication...). Among them, the “tribPiacenza_Int1” Judicial_Interpretation is created: it considers the Contractual_Agreement contained in “α/βClause12” and applies the “notSpecificallySigned” Legal_Status. The instance contains also a reference to the precedent (Cass.1317/1998), which represent a semantically-searchable information on the interpretation instance.

**Reasoning on the knowledge base** - In the example, when all the relevant knowledge is represented into the ontology, the reasoner is capable of inferring that “The agreement contained in clause 12 of the α/β contract is invalid ex article 1341 comma 2”. As already explained, this result is reached through a subclass of the Contractual_Agreement and Qualified classes, defined by an axiom representing the rule of law. Clauses that fulfill the axiom are

---

**Fig. 19** - Stated property assertions of the sample judicial interpretation

**Fig. 20** - Inferred Description and property assertions of the contract clause's content.

**Fig. 21** - Explanation for the sample agreement being inefficacious.
automatically classified in that class, and thus considered by the proper law. At this point, a simple property chain gives the clause its final (efficacy/inefficacy) status under that law.

7. Representing exceptions

A critical issue in representing the decision’s content is represented by exceptions to legal rules. How can we model a situation when a material circumstance applies all the legal statuses required by the legal rule, but nevertheless does not fall under that legal rule’s legal consequence because it follows some additional rule, which defeats the first one? As it should be clear, that issue has no straight solution inside DL, such as OWL-DL logics: if we introduced some negative condition for the rule to apply (if (not (exception))), the open-world assumption OWL relies on would require us to explicitly state for each case that no exception applies. This would annihilate the reasoning capabilities of the ontology set we explained so far.

A solution to this problem could rely on the modeling of the exceptional case as a subclass of the normal case: this means that only the material circumstances which are relevant under the “regular” law can be classified as “exceptional”.

Fig. 22 - Explanation of a sample contract clause being not inefficacious because of an exception.
This solution has the advantage of allowing reasoning on exceptions without the need to rely on rules. The backside is that the classification of the circumstance as "exceptional" is added to the classification of inefficacy, not substituted to it. Again, this issue takes origin from the open world assumption, and cannot be easily avoided while remaining inside OWL-DL: whenever we prevent the reasoner to "judge" a circumstance with a legal consequence, asking him to check that no exception exists, the reasoner will be incapable of inferring anything unless all information concerning the exceptions is explicitly stated in the ontology.

This issue represents the main reason why a complete syntactic modelling of legal rules is not reachable inside the ontology set, requiring instead a rule system (such as LKIF-Rules [10], Clojure, or LegalRuleML [15]) to be fully implemented. Nevertheless, the so-built ontology set represents the ideal background for such a rule system to work.

7. Conclusions

This paper presents an innovative approach to case-law management, filling the gap between text, metadata, ontology representation and rules modeling, with the goal of detecting all the information available in the text to be enhanced in the legal reasoning through an argumentation theory. This approach allows to directly annotate the text with peculiar metadata representing the hook for the core, domain and argument ontologies. OWL 2.0 is used to get as close as possible to the rules, in order to exploit the computational characteristic of description logics. On the other hand, the ontology framework has a strong weak point in the management of exceptions.
References