Modelling Temporal Legal Rules

Monica Palmirani
CIRSFID
University of Bologna
monica.palmirani@unibo.it

Guido Governatori
NICTA
Queensland Research Laboratory
Australia
Guido.Governatori@nicta.com.au

Giuseppe Contissa
Department of Law
European University Institute
Florence, Italy
giuseppe.contissa@eui.eu

ABSTRACT
Legal reasoning involves multiple temporal dimensions but the existing state of the art of legal representation languages does not allow us to easily combine expressiveness, performance and legal reasoning requirements. Moreover we also aim at the combination of legal temporal reasoning with the defeasible logic approach, maintaining a computable complexity. The contribution of this work is to extend LKIF-rules[1]with temporal dimensions and defeasible tools, extending our previous work [17].

Keywords
LKIF-Rule, Temporal dimension, Rule modelling.

1. INTRODUCTION
The main goal of the research is to extend LKIF-rules [5] [6]with temporal dimensions and defeasible elements with the aim of fulfilling the requirements to be expressive, concise, not redundant, detecting times at both rule level and sentence level, to have versioning of rules, and finally to satisfy the isomorphism principle. We find several related works in literature concerning this domain [1] [2][3][15][19], nevertheless our research aims go beyond the limitations of the current state of the art. Some of those limitations were presented in [4]:

- often the temporal model, mostly based on the events, is not specific for the legal domain that requires at least tree axis: time of in force, time of efficacy and time of application of the norms. Those dimensions are structured as events and a pair of two events (start and end) defines an interval;
- the temporal model proposed in this work permits to assign the temporal model above mentioned to each part of the rule: antecedent, consequent, result of reasoning. This granularity of assertions opens new frontiers for legal reasoning especially in term of expressiveness, compactness and performance;
- our LKIF-rules extension follows the isomorphism principle stated by Bench-Capon [1] and Karpf [13] for connecting legal resources (text) with formalized rules;
- the model proposed is designed for reducing the redundancy of references to textual sources, definitions of temporal events and intervals, and connections with the ontology concepts (TBox). This approach is clean, compact, and guarantees high maintainability of the standard over time.

The current LKIF-rules syntax permits to define temporal predicates but this approach has different disadvantages:

- temporal arguments are necessary on deontic predicates or behavior operators;
- nesting of predicates within predicates results in a high logical complexity and this affects strongly the performance, and the simplicity of the representation of the knowledge;
- each knowledge engineer could create different temporal predicates and vocabulary for defining the same temporal set and the interoperability between different rule-modelling institutions is neglected;
- temporal predicates impose to proliferate rules and to increase rule-bases collections with problem of maintainability;
- we need to have the time-set on the rule, body, head and not only on the sentences. This is important issue that make more expressive the logic represented. With the predicate approach it is not possible to implement this situation;
- the rule may have a different time respectively in the antecedent and in the consequent (r_{t3}: $A_{t1} \Rightarrow B_{t2}$ with $t3 \leq t2 \leq t1$ – repeal of the norm X in t1 with retroactive effect in t2, but with application due some conditional in t3.).

2. LKIF-Rule EXTENSION

2.1 Temporal Dimension Elements
Considering the list abovementioned we have modified LKIF-rule by adding temporal arguments as presented in a first version of our work [16][17] and now we propose a refinement and a further enlargement. The extension is made adding a new metadata blocks that permits to add new features:

- `identification` block defines the authors of the rule modelling;
- `references` block defines the external resources (URI to the text) as a unique repository of the URI;
- `sources` block connects the textual resources with the rules using the URI described in the reference block;
- `events` block models the temporal events in neutral way;
- `timesInfo` block models the legal temporal intervals and provide interpretation to the events;
- `rulesInfo` blocks classifies the rules, connect them with the author and assign the time parameters.

[1] LKIF is a proposed mark-up language designed for legal documents and legal knowledge in ESTRELLA Project IST-2004-027655.

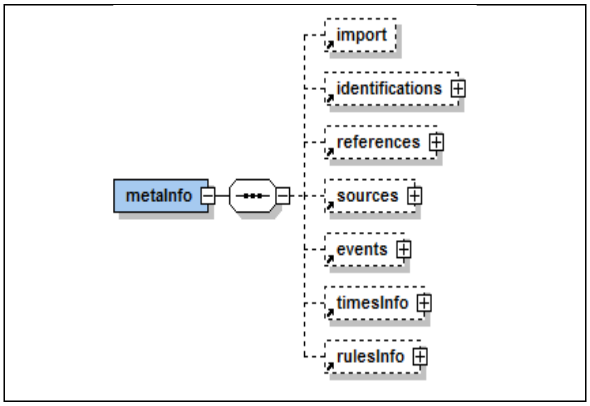


Figure 1. Metadata block.

In this section we focus on four sub-blocks: `references`, `sources`, `events`, and `timesInfo`.

`MetaInfo` block contains the metadata concerning where the rules are expressed into the text. We can notice that the relationship between text and rules has cardinality N:M and for this reason we need to split the representation of this knowledge in two blocks: `references` defines the textual resources involved in the rule modelling using URI and `sources` that connects rules, sentences, header, body with the text.

The block `sources` defines the link between the rules and a textual fragment using `refersTo` attribute (`#order2007`) for connecting the references citied in the theory. In the following example the rule `#rule1` is referred to the fragment in the text `section 25, subsection1, point a)` of the Terrorism Act, 2006 of UK, but also the rule2 is referred to the same fragment. In this way we avoid the duplication of the URI, minimizing the possibility of errors.

```
      <references>
        <reference id="order2007"
uri="http://text1#art1"/>
        <reference id="sec25_2006_1_a"
uri="http://text2#sec25_1_a"/>
      </references>
      <sources>
        <source element="#rule1" refersTo="#
sec25_2006_1_a"/>
        <source element="#rule2" refersTo="#
sec25_2006_1_a"/>
      </sources>
```

Figure 2. Source element: sentence s1 linked to the text.

The block `events` tracks all the temporal events involved in the rules: external and internal times[2]. The following example defines four events used in the `timesBlock` to provide the semantics of each event. This mechanism prevents the explosion of temporal information and minimizes the redundancy in the syntax for the annotation. In meantime `e1` assumes the twofold meaning: time of enter into force and the efficacy date of the Order 2007. So it is interesting to notice that the events are neutral respect the meaning, and only in the `timesInfo` block they assume the connotation, contextually of the rule situation. This permits to avoid redundancy and have a clean semantic annotation.

```
<events>
   <!-- events of the Order 2007 UK-->
   <event id="e1" value="2007-07-25T01:01:00.0Z"/>
   <!-- Terrorism Act of 2006 -->
   <event id="e2" value="2006-03-30T01:01:00.0Z"/>
   <event id="e3" value="2006-07-17T01:01:00.0Z"/>
   <event id="e4" value="2006-07-25T01:01:00.0Z"/>
</events>
```

Figure 3. Events into LKIF-Rule.

The block `timesInfo` is a container of metadata concerning the temporal dimensions and assigns for each sub-block times the semantics of each interval or instant of time.

```
      <timesInfo>
        <!— order 2007 -->
        <times id="t1">
          <time start="#e1" timeType="efficacy"/>
          <time start="#e1" timeType="inforce"/>
        </times>
        <times id="t2">
          <time start="#e1" duration="P01Y"
timeType="application"/>
        </times>
        <!-- Terrorism act 2006 -->
        <times id="t3">
          <time start="#e3" timeType="efficacy"/>
          <time start="#e3" timeType="inforce"/>
        </times>
        <times id="t4">
          <time start="#e4"
timeType="application"/>
        </times>
      </timesInfo>
```

Figure 4. Times block into LKIF-Rule.

It is possible to model well defined intervals using the attributes `start` and `end`, or to define a simple instant setting `start` equal to `end` and finally to use the undefined interval expressed by a duration. In the following example time `t2` is the application date of a rule and the interval is defined with a starting date (event `e1`) and with a duration[3].

```
     <times id="t2">
       <time start="#e1"
duration="P01Y"timeType="application"/>
     </times>
```

Figure 5. Time duration modelling and representation.

In this way we define the events in the metadata and we assign the semantics and the behavior of these events in the `timesInfo` block. With the mechanism of the ID (using URI for permitting an externalization of these metadata in other physical files) we connect the temporal dimension information directly with rule, head and sentence elements.

[2] External time is the time not stated in the norm but that leads the lifecycle of the provision (inforce, validity, efficacy). Internal time is the time specified in the norm. In the example “The additional tax is applicable to the earn of the financial year 2010-2011. This section enters into force at January 1st”, the financial year 2010-2011 is an internal time and the Jan 1st is an external time.

[3] The duration follows the annotation of the `xsd:type` duration in compliance with the W3C standard data type definition. In our case P01Y means one year.

2.2 Temporal Rule Modelling

The aforementioned blocks of information are used inside the rules, statements, head and body for qualifying the temporal dimension of any fragment of the rule. This annotation offers several advantages in term of representation and reasoning:

- a clean separation between events and their interpretations and usage. We can link the interpretation to ontology classes;
- each interpretation could be linked with the name of the actor responsible of the annotation. So it is possible to distinguish "the rules of Governatori" from "the rules of Palmirani" and consequently to provide different reasoning outcome or to deduct "*who said what*";
- a clean separation between rules and text. Nevertheless we implement the *isomorphism* principles connecting the source of the text to the rules, also in which cases where the relationship has cardinality N:M and without syntax redundancy;
- it is possible to use deontic predicates with temporal parameters;
- the head, body, sentences and rules could have different time parameters the head a time of enter into force different from the body. This increase dramatically the potentiality to formal represent odd legal rules, often used in the real legislative norms (both in civil and in common law), and to apply to them the defeasible logic approach and legal reasoning.

In particular regarding the last sentence, we want to enter into the details of retroactive norms[4]: the body's enter into force date is t1, the head's time of efficacy is t2 (e.g. usually in the retroactive norms t2 <t1), the rule is applied in a third time t3 that is the minor or t1 and t2 combined with other eventually conditionals (e.g. suspension, emergency external events, etc.) expressed in sentences of the body. To better understand these aspects we take in consideration an example: the Terrorism Act of UK 2000 and its modifications in 2006 called Terrorism Act 2006.

We take in consideration the case of suspension of Section 25 of the Terrorism Act of UK, 2006, made by the Order 2007. Section 25 modifies the detention period for terrorism actions, defined in the Terrorism Act 2000, from 28 days in 14 days. The suspension made by Order 2007 for a duration of one year creates three scenarios of application of the norm: i) originally the Terrorism Act 2000 defines the detention in 28 days; within 2006-2007 the detention was 14 days due to the Terrorism Act 2006, within 2007-2008 the period of detention was 28 days due to the Order 2007, after 2008 the period of detention was reinstated to 14 days because the suspension of Section 25 terminated. Our aim is to model the rules to permit legal reasoning on this scenario and to detect all these three situations from only one representation, without duplication of rules.

The `timeBlock` attribute embedded into the sentences (<s id="id3b" timeBlock="t1">) defines the "enter into force" and the "efficacy" temporal parameters of either the textual provisions or the conditions. The `timeBlock` of the rule (<rule id="order2007" timeBlock="t2">) indicates when the rule is valid, and finally the `timeBlock` of the head (<head timeBlock="t2">) determines when the consequent is applicable. So in the example we have the following representation: the rule `order2007` models the disapplication of Section 25 of the Terrorism Act 2006 using the predicate `mod:suspension`. This suspension freezes the detection parameter of Section 25.

```
    <rule id="rule1" ruleType="strict"
timesBlock="t2">
    <!-- Disapplication sect. 25 of the Terrorism
act 2006 -->
      <head timesBlock="t2">
        <s pred="mod:suspension" id="id1a">
          <v value="sec25_2006">x</v>is
suspended</s>
      </head>
      <body>
        <s pred="mod:enterInForce" id="id3b"
timesBlock="t1">
          <v id="sec2_2007">x</v> enters into in
force</s>
      </body>
    </rule>
```

Figure 6. Sentence definition in the LKIF-Rule extended.

This means that the days of detention pass from 14 to 28, as originally stated, and this new rules is applicable in an interval of time starting at `e1` (`2007-07-25T01`) and ending one year later (`P01Y`). With this annotation a reasoning engine can provide two different answers to the query "What is it the applicable term for the detention?": "14 days after July 2008", "28 days during the application of the suspension".

```
   <rule id="rule3" ruleType="strict"
timesBlock="t3">
    <!--Sect.25 Terr. Act/2006 modifies Sch.8
Terr. Act/2000 -->
    <head timesBlock="t3">
      <s pred="mod:substituted" id="id4a">
        <v value="sche8_2000">x</v> is modified as
if for "28 days" there were substituted "14
days"</s>
    </head>
    <body>
      <s pred="mod:intoOperation" id="id4b"
timesBlock="t3">
        <v value="rule4">y</v> into operation.</s>
    </body>
    </rule>
```

Figure 7. Rule modelling.

3. LKIF-Rule EXTENSION TO THE DEFEASIBLE LOGIC

To foster the temporal model combined with the main pillars of the defeasible logic (type and hierarchy of the rules) [12][7] we have further extended LKIF-Rule introducing several new elements.

3.1 Classification of the Rules

The different types of rules are modelled with an attribute `ruleType` that is possible to define in the `rule` element. It could take four possible values: `strict`, `defeasible`, `defeater`, `metarule`. `Strict` means a rule that produces indisputable conclusions, `defeasible` means a rule that can be

[4] Nebraska Code, Chapter 29, Criminal Procedure, Section 29-2264, clause 6: "(6) Except as otherwise provided for the notice in subsection 2 (1) of this section, changes made to this section by Laws 2005, LB 713, shall be retroactive in application and shall apply to all persons, otherwise eligible in accordance with the provisions of this section, whether convicted prior to, on, or subsequent to September 4, 2005."

defeated by contrary evidence and `defeater` means a rule that cannot be used to draw any conclusions. Its only use is to prevent some conclusions. Finally the `metaRule` represents any rule on the rule like the modification of norms. This classification could be annotated follow:

```
<rule id="r1" ruleType="strict" timesBlock="t1">
```

Figure 8. Rule Type modelling.

This representation has a relevant disadvantage: it forces to represent a rule twice in case it assumes different typology over time (e.g. r1 is strict at t1 and it changes the type into defeasible at time t2). For this reason we have enriched the annotation used in our previos work to account for this feature. We are using the block `rulesInfo` for obtaining a twofold benefits:

- this makes possible a multiple interpretation on the same rule using the `identification` id, without duplication of the rule;
- it is possible to capture the dynamicity over the time of the classification of the rule in a non-monotonic logic approach..

In the `identification` block we have the possibility to define the different actors involved in the rule modelling (e.g. author, editor1, editor2, etc.) and it is also possible to connect them to the ontology class coupled of their role.

```
<identifications>
   <identification id="aut1"
uri="http://www.cirsfid.unibo.it/monica.palmirani.
owl" as="author"/>
   <identification id="aut2"
uri="http://www.nicta.com.au/guido.governatori.owl
" as="editor"/>
</identifications>
```

Figure 9. Identification of the actors.

In the `rulesInfo` block we qualify the `rule1` with the attribute `ruleType`, we fix this qualification at the time `t1a` and we assign the paternity of this statement to the actor `aut1`.

```
<rulesInfo>
   <ruleInfo source="#rule1" ruleType="strict"
timesBlock="#t1a" refersTo="#aut1"/>
   <ruleInfo source="#rule1"
ruleType="defeasible" timesBlock="#t8a"
refersTo="#aut2"/>
</rulesInfo>
```

Figure 10. Rule 1 classification over the time.

3.2 Superiority of the Rules

Another pillar of Defeasible Logic is the superiority relation, a binary relation defined over the set of rules. The superiority relation determines the relative strength of two (conflicting) rules. Inside of the LKIF-Rule we have defined an element `hierarchy` able to model this relationship in the axioms block and to track the dynamicity over the time (`timesBlock`).

```
<axioms id="axm1">
   <hierarchy>
      <range id="rng1" function="superior"
from="rule2" to="rule3" timesBlock="#t1"/>
   </hierarchy>
</axioms>
```

Figure 11. Superiority relationship modelling.

4. LKIF-Rule EXTENSION TO THE DEONTIC LOGIC

For completing the extension of LKIF-rule we have added some elements for implementing in a directly way the deontic and behavior operators and some operators useful for minimizing the redundancy of the syntax [11]. This approach permits us to avoid the usage of predicates [5] (e.g. family:obligatedToSupport).

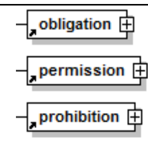
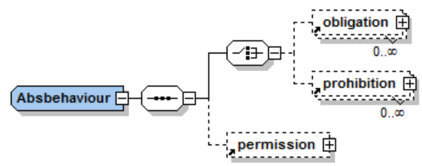
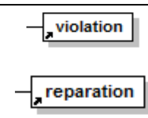
	Deontic operators: obligation, permission, prohibition
	Behavior operators: as sequence of obligations or prohibition and ending with eventually a permission. Penalty is a particular behavior
	Connection Operation: violation, reparation

Figure 12. Deontic and Behavior operators.

The obligation is enriched by `deonticAttributes` that specify the `subject`, the `beneficiary` and the `timesBlock` when the obligation occurs. The penalty is modelled as a particular behavior in the axioms.

```
<axioms id="axm1">
   <penalty id="pnl1">
         <obligation id="obl1" subject="student"
beneficiary="all" timesBlock="#t2">
         <s id="s1" pred="pay1000fine">
            <v>x</v> Pay 1000$</s>
         </obligation>
   </penalty>
</axioms>
```

Figure 13. Penalty modelling.

The `violation` is a unary relationship that refers to the obligation/prohibition subject of the violation. The `reparation` as well is a unary relationship giving a link to the relevant penalty.

```
<rules>
   <rule id="rule1">
      <head>
         <reparation id="rpr1" penalty="#pnl1"/>
      </head>
      <body>
         <violation source="#rule2"/>
      </body>
   </rule>
   <rule id="rule2">
      <head>
         <prohibition id="prh1">
            <s id="s2" pred="smoke">
```

```
                <v>x</v> smokes</s>
            </prohibition>
        </head>
        <body>
            <s id="s3" pred="isStudent">
                <v>x</v> is a student</s>
        </body>
    </rule>
```

Figure 14. Obligation operator combined with the violation.

5. CONCLUSIONS

The LKIF-Rule experience underlines several considerations concerning the gap between the five levels[5] [16] of information that we need to model for describing a legal document especially in the management of the temporal argument. Starting from previous work we have enlarged our scope including the defeasible logic and the deontic operators, jointly with the temporal parameters. We have resolved some previous limitations:

- the relationship between rules and text exhibits an N:M cardinality, so the LKIF-rules syntax was improved in the `references` and `sources` metadata blocks;
- the interpretation of the rules are multiple, using the `identifications` block;
- events are independent by their semantic, using the `events` and the `timesInfo` blocks;
- rule, head, body, sentences are labelled with separate temporal parameters usign `timesBlock`;
- defeasible rule are sensitive to the dynamicity and they could be manifold annotated in the `rulesInfo`;
- deontic and behavior operators could be directly expressed and enriched as well with the temporal parameters.

Our future work will go in the direction to build a reasoner [14] on the base of other works [8] [9] [10] able to demonstrate with a pilot case benefit to this theoretical model.

6. ACKNOWLEDGMENTS

The authors would like to thank the ICT4Law project (http://www.ict4law.org/), coordinated by Ajani G., financed by Piedmont Region, for supporting this research.

NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.

7. REFERENCES

[1] Bench-Capon T. J. M., Gordon T. F.: Isomorphism and argumentation. ICAIL 2009: pp. 11-20, 2009.

[2] Boer A., Radboud W., Vitali F.: MetaLex XML and the Legal Knowledge Interchange Format, in Computable Models of the Law, Springer, 2008.

[3] Breuker J., Boer A., Hoekstra R., Van Den Berg C.: Developing Content for LKIF: Ontologies and Framework for Legal Reasoning, in Legal Knowledge and Information Systems, JURIX 2006, pp. 41-50, ISO Press, Amsterdam, 2006.

[4] Gordon T. F., Governatori G., Rotolo A.: Rules and Norms: Requirements for Rule Interchange Languages in the Legal Domain. RuleML 2009: pp. 282-296, 2009.

[5] Gordon Thomas F.: An Overview of the Legal Knowledge Interchange Format. BIS (Workshops) 2010: pp. 240-242, Springer, 2010.

[6] Gordon Thomas F.: Constructing Legal Arguments with Rules in the Legal Knowledge Interchange Format (LKIF). Computable Models of the Law, Languages, Dialogues, Games, Ontologies 2008, pp. 162-184, 2008.

[7] Governatori G., Maher M. J., Grigoris A. and Billington D.: Argumentation semantics for defeasible logics. Journal of Logic and Computation, 14, 5: pp. 675-702, 2004.

[8] Governatori G., Rotolo A., Rivert R., Palmirani M. and Sartor G.: Variants of Temporal Defeasible Logic for Modelling Norm Modifications. In Radboud Winkels, editor, Proceedings of 11th International Conference on Artificial Intelligence and Law, pp. 155-159. ACM Press, New York, 2007.

[9] Governatori G., Rotolo A.: Changing legal systems: legal abrogations and annulments in Defeasible Logic. Logic Journal of the IGPL 18(1): pp. 157-194, 2010.

[10] Governatori G.: On the Relationship between Carneades and Defeasible Logic., ICAIL 2011.

[11] Governatori G.: Representing business contracts in RuleML. International Journal of Cooperative Information Systems, 14 2-3: 181-216, 2005.

[12] Grigoris A., Billington D., Governatori G.and Maher M. J.: Representation results for defeasible logic. ACM Transactions on Computational Logic, 2: pp. 255-287, 2001.

[13] Karpf J.: Quality assurance of Legal Expert Systems, Jurimatics No 8, Copenhagen Business School, 1989.

[14] Lam H. and Governatori G.: The Making of SPINdle. In Guido Governatori, John Hall, and Adrian Paschke, editors, Rule Representation, Interchange and Reasoning on the Web, RuleML-2009. LNCS 5858: pp. 315-322, 2009.

[15] Marín H. R. and Sartor G.: Time and Norms: A Formalisation in the Event-calculus. In: Proceedings of the Seventh International Conference on Artificial Intelligence and Law (ICAIL). New York, N. Y., ACM: pp. 90–100, 1999.

[16] Palmirani M., Contissa G., Rubino R.: Fill the Gap in the Legal Knowledge Modelling. In Guido Governatori, John Hall, and Adrian Paschke, editors, Rule Representation, Interchange and Reasoning on the Web, RuleML-2009. LNCS 5858: pp. 305-314, 2009.

[17] Palmirani M., Governatori G., and Contissa G.: Temporal dimensions in rules modelling. In Radboud Winkels, editor, Legal Knowledge and Information Systems JURIX 2010: The Twenty-Third Annual Conference, Frontiers in Artificial Intelligence and Applications, Amsterdam, 2010. IOS Press

[18] Proceeding of the 12th International Conference on Artificial Intelligence and Law, Barcelona 8-12 June, 2009, ACM, NY, 243.

[19] Sartor G.: Legal Reasoning: A Cognitive Approach to the Law. Vol. 5. Treatise on Legal Philosophy and General Jurisprudence. Berlin: Springer, 2005.

[5] Following the semantic web cake: text, structure, metadata, ontology, rules.