Ontology Evolution Analysis with OWL-MeT

Natalya Keberle[1], Yuriy Litvinenko,
Yuriy Gordeyev, Vadim Ermolayev

Dept. of IT, Zaporozhye National University,
Ukraine
nkeberle@gmail.com, jurlit@rambler.ru,
ygchaos@mail.ru, vadim@ermolayev.com

Abstract. Works on ontology versioning pay special attention to the logical analysis of ontology evolution. The paper considers extensible declarative approach to ontology change description. Metric temporal description logic with metric temporal modalities "*future n*" and "*past n*" and hybrid satisfaction operator @ is proposed as the logical basis for declarative ontology evolution analysis. Underlying time structure assumed to be linear and discrete, which is acceptable for modeling of ontology versions sequence. Introduced is OWL-MeT – metric extension of the Web ontology language OWL, which is supported with a reasoning engine under development on the basis of Pellet reasoner.

1 Introduction

Ontology versioning and change detection are one of the Semantic Web research challenges [1]. Many previous efforts at providing ontology versioning have focused on the differentiation of the conceptual and explication changes [2], on the identification of an ontology in the Semantic Web [2], [3], on ontology change operations and their effects at the instances level [3], [4]. Change detection between similar ontologies or between versions of the same ontology when there is no version log is discussed in [5], [6]. Change detection and propagation between versions of the same ontology, when there is a version log, is investigated in [7]. Special attention is paid to the change management organization for distributed and modular ontologies [8], [9], inconsistent ontologies [10].

The approach reported in the paper focuses on the analysis of ontology evolution, namely, compatibility and interchangeability of ontology versions, and proposes a formal declarative basis for such analysis. Indeed, an ontology is a formal theory [11], and for any two versions of the same ontology it is interesting to know whether these versions are compatible from the logical point of view, whether it is possible to use definition of an object in one version to access instances of that object in another version.

[1] The work of the author is partially supported in frame of PSI project. Performance Simulation Initiative (PSI) is the R&D project of Cadence Design Systems, GmbH.

Presented research aims at the development of logical means which will facilitate the various tasks of ontology evolution analysis. It is proposed to use temporal logic, and to combine temporal logic with explicit use of metric properties of time. Propositional metric temporal language (first introduced by A.Prior [12]) and calculus (investigated in [13]) serve as the basis for the presented research. Constructive proofs of soundness, completeness and decidability of propositional metric temporal calculus [13] are based on the tableau technique.

The contributions of the paper are: (i) further development of proof-theoretic approach – introduction of metric temporal description logic $\mathcal{ALCIO}(\mathcal{MT})$ with additional sort of nominals representing ontology versions; (ii) introduction of OWL-MeT – metric extension of OWL with constructs of $\mathcal{ALCIO}(\mathcal{MT})$ in a way suitable for the Semantic Web applications; (iii) introduction of reasoning engine for OWL-MeT, based on Pellet[2] – the open-source OWL-DL reasoner in Java.

The paper is organized as follows: the Section 2 presents the related work and the motivation to enhance analytical means for ontology evolution; metric temporal description logic $\mathcal{ALCIO}(\mathcal{MT})$ with metric temporal modalities and hybrid satisfaction operator is described in Section 3; Section 4 introduces OWL-MeT – the extension of OWL with the constructs of $\mathcal{ALCIO}(\mathcal{MT})$ and describes Pellet-based reasoning engine for OWL-MeT; examples of ontology evolution analysis with OWL-MeT are given in the Section 5, and Section 6 presents some conclusions made in frame of the presented research.

2 Related Work and Motivation

Approaches to ontology changes detection and storage basically work on the syntactic layer. In [2] it was proposed to use rule-based approach to change detection. The idea behind the approach is to precisely formulate what can be considered as change for every syntactical construct of ontology description language. The complete list of possible change operations for OWL-Lite was introduced in [3] in the form of "the ontology of ontology change", which is used to mark-up change logs. The framework for ontology change management, OntoView, was described in [5], [3]. Some disadvantages of the rule-based approach to change detection are investigated in [7]. It was shown that in case of multiple changes occurred in the ontology some changes may interfere, and in the worst case even cancel one another, as far as change detection rules work independently of each other.

Proof-theoretic approach to ontology evolution analysis, first introduced in MORE [14], [4] overcomes the drawbacks of other approaches. First of all, it is ontology language-independent: meta-level temporal language LTLm was used for changes analysis. Secondly, reasoning over ontology changes instead of querying changes was proposed, which allowed to rely not only on heuristics/rules of change detection, but to deduce complex changes. Finally, usage of temporal logic is natural as far as changes are characterized with time moments, when they occur.

[2] Pellet homepage is http://pellet.owldl.com

However, analysis of [14] has shown that the usage of temporal logic LTLm has some drawbacks. LTLm is past-oriented, therefore complex statements binding future and past moments can not be described. Further, statements about states of affairs in two ontology versions which are at a given distance from each other, without naming these versions are impossible. Finally, the developers of MORE [4] have outlined for the future work the convenience of explicit usage of version names in a temporal formula, i.e. hybridization of LTLm.

Interesting ideas for the detailed change analysis of both the extensional and the structural levels are proposed in [15] for evolving collections of XML documents. The authors of [15] provide a classification of query types for the analysis of structural changes occurred in the versions of an XML document: structural projection, historical queries, temporal selection, content-based selection. Use of these query types might seriously enhance the ontology versioning frameworks. The steps in this direction are made in the SEKT[3] project [4].

The review of the systems and the approaches presented above shows that detailed analysis of ontology evolution can be facilitated with the two components: (i) – there are either all versions of the ontology or the version log for that ontology, and (ii) – reasoning over the versions is performed with help of temporal logics. The motivation of the approach proposed in the paper is based on the idea of an ontology versioning system facilitating the analysis of evolution. Following [14] and [15] possible analytical queries may include:

- **Reasoning** queries:
  - (non-)derivability of a fact in one version with respect to another version, i.e. whether the fact belongs to the intersection/difference of ontology models
  - (non-)derivability of a fact in a given version
- **Meta-level ontology-specific** queries:
  - compatibility of versions (e.g. "Is the version 1 is subsumed by the version 5?")
  - compatibility of concept definitions (e.g. "Is the concept *Child* defined in the version 1 subsumed by the concept *Child* taken from the version 5?") etc.
  - most common part of concept definitions taken from different version (e.g. "What properties remain unchanged in the concept *Child* from the version 5 to the version 10") etc.
  - detection of conceptual relations [3] between the versions of ontology (e.g. "What version subsumes the version 5?")
- **Retrieval** queries
  - temporal selection (e.g. "Get the version 5")
  - historical queries (e.g. "What was added in the version 5?", "What was added in the version 5 as compare to the version 2?", "What become obsolete in the version 5 as compare to the version 10?")
  - structural projection (e.g. "Get the definition of the concept *Child* in the version 5", "Get the new/retired/unchanged instances of the concept *Child* in the version 5 as compare to the version 2")

[3] MORE is developed in frame of SEKT project, http://www.sekt-project.org

- content-based selection (e.g. "Check all the versions and get all the concept definitions where role *hasChild* is used", "Check the versions from 5 to 10 and get all the concept definitions where the role *hasChild* is used").

Presented research assumes that all ontology versions are available for analysis. Its usage in the case of presence of a version log goes beyond the scope of the paper.

Mentioned queries explore the concept of time moment (expressed in an ontology version number). Moreover, as the time moments / ontology version numbers are used explicitly in queries, there is a need to have correspondent means in underlying logic formalism for referencing time points explicitly.

Hybrid logics [16], [17] which allow naming particular possible world may provide means for referencing time moments explicitly in a formulae. Metric temporal logics [18] allow to reference time moments at a given distance to the future and to the past.

First introduced in [18], metric modalities "*future n*" and "*past n*" were then investigated by Clifford in [19], where the semantics of the metric temporal language was given. Other temporal modalities (such as Scott's unary temporal operator "next instant" [12] or Kamp's binary temporal operators "*until*" and "*since*" [12]) can be defined via metric ones. Later on metric modalities were extensively learnt in the research on real-time logic, and in the recent time, in the research on distance logics [20]. Composition of hybrid and metric temporal logic may provide means necessary for ontology evolution analysis.

3 Metric Temporal Description Logic

Highly-expressive and decidable Description Logics are of great interest in the Semantic Web applications. Known decidable temporal description logics (see, e.g. [21]) focus on topological properties of time, and provide descriptive means for both interval-based and point-based time structure. Such logics are usually seen as combination of a propositional modal logic (as far as description logics have strict correspondence with propositional multimodal logics) and temporal (therefore, also modal) logic. Expressive power of obtained composed logic depends on the degree of interaction between two modal logics.

Metric description logics are relatively new members of the description logics family. The research on metric description logics was initiated with the development of logic of distances [20]. Since then many classes of such logics are investigated for the purpose of decidability.

Metric temporal description logic $\mathcal{ALCIO}(\mathcal{MT})$ for point-based time structure is the composition of $\mathcal{ALCIO}$ – description logic with role inverses and nominals and propositional metric temporal logic. Particularly, $\mathcal{ALCIO}(\mathcal{MT})$ allows application of temporal and hybrid operators only to concepts (both non-temporal and temporal), and role restrictions are not applied to temporal concepts.

Let A, B denote atomic **non-temporal** concepts, R - atomic role, E, F - complex non-temporal concepts, P - complex role, C, D - complex **temporal** concept, $\{o\}$ - **object** nominal (denoting an individual in some possible world), $\{a\}$ - **temporal** nominal (denoting possible world, e.g. ontology version).

Then the rules presented in the Fig.1 generate complex concepts/roles.

$$
\begin{aligned}
E, F &\rightarrow A \mid top \mid bottom \mid E \sqcap F \mid E \sqcup F \mid \neg E \mid \exists R.E \mid \forall R.E \mid \{o\} \\
P &\rightarrow R \mid P^{-1} \\
C, D &\rightarrow E \mid \{a\} \mid C \text{ intersection } D \mid C \text{ union } D \mid \text{not } C \mid C@\{a\} \mid \text{future n } C \mid \\
&\quad \mid \text{past n } C \mid \text{somefuture } C \mid \text{somepast } C \mid \text{allfuture } C \mid \text{allpast } C
\end{aligned}
$$

Fig. 1. Syntax rules for $\mathcal{ALCIO}(\mathcal{MT})$ concepts/roles construction.

If C and D are temporal concepts, and $\{a\}$ is a temporal nominal, then *C equivalent D*, *C subclassof D*, $C(a)$ are temporal formulae. If φ and ψ are temporal formulae, then φ *union* ψ, φ *intersection* ψ, *not* φ are also temporal formulae.

$\mathcal{ALCIO}(\mathcal{MT})$ is interpreted over Kripke model $\mathrm{M} = < \Delta, dist, \{R_F, R_P\}, I, V >$, where $\Delta = \{\Delta^k, k \in Z\}$ is a set of possible worlds, Δ^k is a set of individuals in k-th possible world, $dist : \Delta \times \Delta \rightarrow N \cup \{0\}$ is a metric on Δ, R_F, R_P are accessibility relations, I is an interpretation function, and V is a hybrid valuation function.

Interpretation I associates with each Δ^k an $\mathcal{ALCIO}$-interpretation $I(k) = < \Delta^k, \cdot^{I(k)} >$.

Temporal nominals are used as identifiers of ontology versions. Satisfaction operator $@\{a\}$, where $\{a\}$ is considered as an identifier of an ontology version, is adopted from hybrid logics [17].

Additionally, let function $den : \{a\} \rightarrow Z$ be an encoding of temporal nominals into integers. Then, given temporal nominal $\{a\}$, *hybrid valuation* V assigns $\{a\}$ a unique world $\Delta^{den(a)}$ - singleton subset of Δ.

The model-theoretic semantics of $\mathcal{ALCIO}(\mathcal{MT})$-specific operators is defined as follows:

$$
\begin{aligned}
(\text{future n } C)^{I(k)} &= \{o \in \Delta^k : \exists j = k + n, o \in C^{I(j)}\} \\
(\text{past n } C)^{I(k)} &= \{o \in \Delta^k : \exists j : k = j + n, o \in C^{I(j)}\} \\
(\text{somefuture } C)^{I(k)} &= \{o \in \Delta^k : \exists j \geq k, o \in C^{I(j)}\} \\
(\text{somepast } C)^{I(k)} &= \{o \in \Delta^k : \exists j \leq k, o \in C^{I(j)}\} \\
(\text{allfuture } C)^{I(k)} &= \{o \in \Delta^k : \forall j \geq k, o \in C^{I(j)}\} \\
(\text{allpast } C)^{I(k)} &= \{o \in \Delta^k : \forall j \leq k, o \in C^{I(j)}\} \\
(C@\{a\})^{I(k)} &= \{o \in C^{I(den(a))}\}
\end{aligned}
$$

For the purposes of ontology evolution analysis we restrict the domain Δ to be a finite linear sequence of ontology versions – *time structure*, or in terms of [14] *version space*. *Time structure* is a finite sequence of temporal nominals, each of which identify particular ontology version. *Time structure* is ordered with the precedence relation, set between encodings of temporal nominals: if $den(a_1) < den(a_2)$, then the version identified with a_1 precedes the version identified with a_2.

Satisfiability problem for hybrid multi-modal tense logic with nominals, whose syntactic variant is $\mathcal{ALCIO}(\mathcal{MT})$, is EXPTIME-hard [16].

Releasing the domain Δ from being finite leads to analysis of the decidability issues for nonbranching discrete transitive and reflexive unbounded frame. Tableau rule set for hybrid and metric operators provides a ground for constructive proof of decidability of satisfiability problem for $\mathcal{ALCIO}(\mathcal{MT})$. Termination of the decision procedure is controlled with the application of looptest rules (which are applicable for transitive frames, see, e.g. [22]), and of the formulae marking rules (see, e.g. [23]). However, the satifiability problem is still EXPTIME-hard, as the results in [16] were obtained for arbitrary frame. Results obtained in [24] for NP-completeness of satisfiability problem for linear frame (irreflexive, transitive and trichotomous) seem not to be applicable here, as the behaviour of metric operators "*future n*" / "*past n*" when *n = 0* requires the accessibility relations to be reflexive.

4 OWL-MeT

OWL-MeT (abbreviation for OWL-MetricTime) is built on top of the language OWL; it has been assigned namespace *owlmet*. Main constructs for definition of temporal concepts of OWL-MET are **TClass** and **TRestriction**. **TClass**, representing named temporal concepts, is the direct subclass of *rdfs:Class*, **TRestriction**, representing unnamed temporal restrictions, is the subclass of *owlmet:TClass*. Standard OWL class *owl:Class* is defined as subclass of *owlmet:TClass*, therefore all OWL concepts are also OWL-MeT temporal concepts. Indeed, each non-temporal concept *A* (which is the instance of *owl:Class*) can be considered as equivalent to an instance of unnamed temporal restriction *future 0 A*. In OWL-MeT temporal concepts are allowed to form unions, intersections and complements, to define axioms of equivalence and subsumption between them.

OWL-MeT introduces the special sort of nominals for ontology versions – temporal nominals, or instants. **Instant** is also the subclass of *owlmet:TClass*. **TimeStructure** construct fixes the sequence of instants and it is defined with help of *rdf:sequence* construct.

Metric temporal operators are defined as instances of *rdf:property*, namely "**future**", "**allfuture**", "**past**", "**allpast**", "**somefuture**", "**somepast**"; hybrid satisfaction operator "**at**", and its supplementary operator "**happens**" are also the instances of *rdf:property*.

The abstract syntax of OWL-MeT extends the OWL abstract syntax, OWL-MeT new constructs are presented in the Fig. 2. Complete definition[4] of OWL-MeT includes abstract syntax definition, mapping to RDF graphs and usage examples.

The reasoning engine for OWL-MeT is now under development. It is grounded on the open-source Java-based OWL-DL reasoner Pellet. Jena 2.4[5] is used for the validation of OWL-MeT constructs and for correspondent RDF models building.

[4] OWL-MeT Web site: http://ermolayev.com/owl-met

axiom ::=	temporalAxiom | owlAxiom
temporalAxiom::=	'TClass(' **classID modality {annotation} {description}**')' |
	| 'DisjointClasses(' **description description { description }** ')' |
	| 'EquivalentClasses(' **description { description }** ')' |
	| 'SubClassOf(' **description description** ')' |
	| 'EnumeratedClass(' **classID { annotation } { individualID** }')' |
	| 'TimeStructure(' **instantID {instantID}** ')'
modality::=	'complete' | 'partial'
description::=	**classID** |
	| **trestriction** |
	| **resriction** |
	| 'unionOf(' **{ description }** ')' |
	| 'intersectionOf(' **{ description }** ')' |
	| 'complementOf(' **description** ')' |
	| 'oneOf(' **{ individualID }** ')'
trestriction::=	'TRestriction(' **timeProperty** ')'
timeProperty	'allfuture(' **description** ')' | 'somefuture(' **description** ')' |
	| 'allpast(' **description** ')' | 'somepast(' **description** ')' |
	| 'future(' **non-negative-integer** ')' **modality description** |
	| 'past(' **non-negative-integer** ')' **modality description** |
	| 'at(' **instantID** ')' **modality description**

Fig. 2. Abstract syntax of OWL-MeT (the added part as compare to OWL).

Extension of Pellet to reason over OWL-MeT descriptions requires also reworking of normalization and internalization procedures, and extending of the decision procedure with metric and hybrid tableau rules.

In short, hybrid extension of tableau rules creates for each temporal nominal $\{a\}$ presented in a given OWL-MeT formula a particular tableau, and establishes accessibility relations between these tableaux depending on values of $den(a)$. Metric extension of tableau rules describes as movement across the tableaux sequence using that accessibility relation, as well as creation of a particular tableau. For example, operator "*future 1 C*" checks if the tableau at the distance 1 to the future exists, if not - creates that tableau and establishes accessibility relation R_F between the initial tableau and that new. Operator "*future n C*" is considered as "*future 1 future (n-1) C*".

Operators "*allfuture C*" and "*allpast C*" due to the transitivity of time copy *C* to all tableaux accessible via accessibility relation R_F (R_P) from the given one. Operators "*somefuture C*" and "*somepast C*" create alternating tableau branches (due to the reflexivity of the model M): one branch will include *C*, and the other will include "*future 1 somefuture C*"/ "*past 1 somepast C*" respectively. Tableau termination is controlled with the help of adopted looptest rules, avoiding the introduction of both $\mathcal{ALCIO}$ -node fully contained in a predecessor $\mathcal{ALCIO}$ - node, and the introduction of a temporal nominal node fully contained in a predecessor node.

[5] Jena is the open-source Java framework for building Semantic Web applications. It provides a programmatic environment for RDF(S), OWL, SPARQL and includes a rule-based inference engine. Available at http://jena.sourceforge.net

Consider several examples of temporal concept definitions. Let concept "PerpetuumMobile" be defined as some engine that will work always in the future. Correspondent OWL-MeT definition with the non-metric operator "*allfuture*" may be as shown on the Fig. 3. Statements about a temporal concept, relating both future and past moments use metric operators "*future n*" and "*past n*": for example, the definition of "Student" as "UniversityEntrant" in one moment before is shown on the Fig. 4. Finally, statements with explicit naming of time moments use hybrid satisfaction operator "*at*". Mapping to RDF(S) due to certain limitations of RDF(S) had required the introduction of supplementary operator, which is called "*happens*" and which allows to partition temporal concepts occurring at different points. The definition of "HappyFather" (see Fig.5) is given as the union of "HappyFatherInXXCentury" and "HappyFatherInXXICentury", where the disjuncts are defined at different time moments. Time moments "XXCentury" and "XXICentury" are the individuals of the *owlmet:TInstant*.

```
<owlmet:TClass rdf:ID="PerpetuumMobile">
 <rdfs:subClassOf>
 <owlmet:TRestriction>
 <owlmet:allfuture>
  <owlmet:TClass>
    <owl:intersectionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#Engine"/>
    <owl:Restriction>
    <owl:allValuesFrom> <owl:Class rdf:about="#Thing"/> </owl:allValuesFrom>
    <owl:onProperty> <owl:ObjectProperty rdf:about="#works"/> </owl:onProperty>
    </owl:Restriction>
    </owl:intersectionOf>
 </owlmet:TClass>
 </owlmet:allfuture>
 </owlmet:TRestriction>
 </rdfs:subClassOf>
</owlmet:TClass>
```

Fig. 3. Definition of “PerpetuumMobile”.

```
<owlmet:TClass rdf:ID="Student">
 <rdfs:subClassOf>
      <owlmet:TRestriction>
      <owlmet:past rdf:datatype="&xsd;#NonNegativeInteger"> 1</owlmet:past>
      <owlmet:equivalentClass> <owlmet:TClass rdf:about="#UniversityEntrant"/>
      </owlmet:equivalentClass>
      </owlmet:TRestriction>
      </rdfs:subClassOf>
</owlmet:TClass>
```

Fig. 4. Definition of “Student”.

```
<owlmet:TClass rdf:ID="HappyFather">
<owlmet:equivalentClass>
<owlmet:TClass> <owlmet:unionOf rdf:parseType="Collection">
<owlmet:TRestriction> <owlmet:at rdf:resource="#XXthCentury"/> <owlmet:happens>
<owlmet:TClass rdf:about="#HappyFatherInXXthCentury"/>
</owlmet:happens>
</owlmet:TRestriction>
<owlmet:TRestriction> <owlmet:at rdf:resource="#XXIthCentury"/>
<owlmet:happens><owlmet:TClass rdf:about="#HappyFatherInXXIthCentury "/>
</owlmet:happens>
</owlmet:TRestriction>
</owlmet:unionOf></owlmet:TClass>
</owlmet:equivalentClass></owlmet:TClass>
```

Fig. 5. Definition of “HappyFather”.

5 Ontology Evolution Analysis with OWL-MeT

Let us show how OWL-MeT may serve for the task of ontology evolution analysis. Recall that it was proposed in the Section 2 to differentiate three types of analytical queries: retrieval queries (or, queries on the structure of vocabulary), meta-level ontology queries and reasoning queries.

Reasoning queries answer the questions about derivability of a certain fact across versions. For example, to answer the question “Are individuals of concept *C* in a version *v5* derivable both in versions *v2* and *v10*?” one may check satisfiability of a temporal formula (*C* @{*v5*}) *subclassof* (*C* @{*v2*} *intersection C* @{*v10*}).

The question “What are new individuals of concept *C* in a version *v5*, which were not present two versions before” may be answered by checking satisfiability of a temporal concept (*C intersection* ((*past 2*) *not C*)) @{*v5*}.

The question “What are individuals of concept *C* in a version *v5*, which are not derivable at a distance of two versions in the future?” may be answered by checking satisfiability of correspondent temporal concept (*C intersection* ((*future 2*) *not C*)) @{*v5*}. These examples show the benefits of descriptive means of $\mathcal{ALCIO}(\mathcal{MT})$, as compare to known formalisms, e.g. [4].

To discuss the application of OWL-MeT to meta-level ontology specific queries recall that an ontology *O* is a set of terminological and assertional axioms [25], and satisfiability checking of a knowledge base is equivalent to satisfiability checking of a concept *G*, equivalent to the conjunction of definitions of all concepts and individuals taken from *O*[6]. From the other side, meta-level queries have obvious relationship with the theory of modular ontologies [9]. Indeed, ontology *versions* in their usual form are ontology *modules* without concepts, defined externally.

[6] Internalization procedure is applicable to logics allowing the definition of a universal role [25]; OWL-Lite and OWL-DL as the languages of ontology version definition possess role-forming operators – union and reflexive transitive closure – necessary to construct a universal role.

Let O_i be an ontology, actual at the version i, and let G_i be a concept, equivalent to the conjunction of definitions of all concepts and individuals in O_i. Such G_i may be considered as the concept, defined *internally* [9] in an ontology version (or module) i. Then simple checking of the fact that O_i in the version i is satisfiable may be written as G_i @{i}.

Checking satisfiability of a particular concept E taken from an ontology version i in another version, say j, may be executed in several steps. Initially, introduce a *compiled* concept $G_{E,i}$ – conjunction of all explicit and implicit definitions of concepts and individuals, used to define E in O_i . This can be done, e.g. with the help of compilation algorithm, defined in [9]. Then check satisfiability of concept (G_j *intersection* $G_{E,i}$) @{j}.

Analogously, given the whole ontology O_i, checking of the fact that in another version, j (where $j \neq i$), the ontology O_i is satisfiable may be written as (G_j *intersection* G_i) @{j}.

Retrieval queries serve for analysis of the structure of vocabulary, used to define an ontology version, and for analysis of structural changes between ontology versions. Some of the retrieval queries, such as temporal selection, may be formulated without usage of temporal logic. However, retrieval queries benefit from tight relationship with the underlying ontology language. A collection of interesting retrieval query templates, such as "NewChildren", "ObsoleteChildren" etc. in [14], or, ideally, proper ontology of ontology change operators, such as in [3] may serve as the source of various templates. More detailed analysis might require extension of $\mathcal{ALCIO}(\mathcal{MT})$ with role restrictions on temporal concepts.

6 Conclusions and Future Work

The paper proposes an enhancement of the formal apparatus for ontology evolution analysis, and introduces metric temporal description logic $\mathcal{ALCIO}(\mathcal{MT})$, which attempts to release the restrictions of known logical approaches. Reasoning support of OWL-MeT – the extension of Web Ontology language OWL with metric temporal operators proposed in $\mathcal{ALCIO}(\mathcal{MT})$ is now under development on the basis of open-source reasoner Pellet.

Future work is seen in following directions: encoding of interesting change operations, such as "RestrictRange", "ExtendRange", "RestrictDomain", "ExtendDomain", "ModifyEquivalenceToSubclass" etc., taken e.g. from [3], and adopted to OWL DL, in OWL-MeT; development and implementation of optimization strategies of reasoning; and, finally, testing the reasoner services on real cases.

7 Acknowledgements

Authors would like to thank anonymous reviewers for their helpful comments.

References

1. Research Challenges and Perspectives of the Semantic Web, EC-US NSF Strategic Research Workshop, Sophia Antipolis, France, October 3-5 (2001)
2. Klein M.C.A., Kiryakov A., Ognyanov D., Fensel D.: Finding and Characterizing Changes in Ontologies. In: Proc. of the 21st Int. Conf. on Conceptual Modeling (ER'2002), Tampere, Finland, October 7-11. LNCS 2503 (2002) 79–89
3. Klein M.C.A.: Change Management for Distributed Ontologies. Ph.D. Thesis, Aug. 2004, ISBN 90-9018400-7.
4. Huang Z., Stuckenschmidt H.: Reasoning with Multi-version Ontologies. EU-IST Integrated Project (IP) IST-2003-506826 SEKT, Deliverable D3.5.1 (2005)
5. Klein M.C.A., Noy N.F.: A Component-Based Framework For Ontology Evolution. In: Proc. of the Workshop on Ontologies and Distributed Systems, IJCAI ’03, Acapulco, Mexico.
6. Noy N.F., Musen M.: PromptDIFF: A Fixed-Point Algorithm for Comparing Ontology Versions. In: Proc. of the 19th Nat. Conf. on Artificial Intelligence (AAAI/IAAI 2002), Edmonton, Alberta, Canada, July 28 – August 1. AAAI Press (2002) 744–750
7. Plessers P., De Troyer O.: Ontology Detection Using a Version Log. In: Proc. of the 4th International Semantic Web Conference (ISWC'2005), Galway, Ireland, November 6–10. LNCS 3729 (2005) 578–592
8. Stojanovic L., Maedche A., Motik B., Stojanovic N.: User-driven ontology evolution management. In: Proc. of the 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW02), Siguenza, Spain, October 1-4. LNCS 2473 (2002) 285–300
9. Stuckenschmidt H., Klein M.C.A.: Integrity and change in modular ontologies. In: Proc. of the Intl. Joint Conf. on Artificial Intelligence (IJCAI'2003), Acapulco, Mexico, August 9-15. Morgan Kaufmann (2003) 900–905
10. Haase P., van Harmelen F., Huang Z., Stuckenschmidt H., Sure Y.: A Framework for Handling Inconsistency in Changing Ontologies. In: Proc. of the 4th International Semantic Web Conference (ISWC'2005), Galway, Ireland, November 6–10. LNCS 3729 (2005) 353–367
11. Guarino N.: Formal Ontology and Information Systems. In: Proc. of the 1st Int. Conf. Formal Ontology in Information Systems (FOIS'1998), Trento, Italy, June 6–8. IOS Press/Ohmsha (1998) 3–15
12. Prior A.N.: Past, Present and Future. Oxford University Press, Oxford (1967)
13. Keberle N. G.: Properties of Propositional Metric Temporal Calculus for Description of Evolving Conceptualizations. In: Theses of the 3rd Int. Conf. on Mathematical Support and Software for Artificial Intelligence (MSSAI’2005), Dnepropetrovsk, Ukraine, November 16–18 (2005) 65–66
14. Huang Z., Stuckenschmidt H.: Reasoning with Multi-version Ontologies: A Temporal Logic Approach. In: Proc. of the 4th International Semantic Web Conference (ISWC'2005), Galway, Ireland, November 6–10. LNCS 3729 (2005) 398–412
15. Chien S.-Y., Tsotras V., Zaniolo C.: Efficient Management of Multiversion Documents by Object Referencing. In: Proc. of the 27th VLDB Conference, Roma, Italy, (2001)
16. Areces C., Blackburn P., Marx M.: A roadmap on the complexity of hybrid logics. In: Flum J. and Rodriguez-Artalejo M. (eds.): Computer Science Logic, LNCS 1683. Springer (1999) 307–321
17. Blackburn P.: Representation, Reasoning, and Relational Structures: a Hybrid Logic Manifesto. In: Areces C., Franconi E., Goré R., de Rijke M. and Schlingloff H. (eds.): Special Issue of the Logic Journal of the IGPL 8:3 (2000) 339–625
18. Prior A.N.: Stratified Metric Tense Logic. Theoria 33 (1967) 28–38

19.Clifford J.E.: Tense and Tense Logic. Mouton Publishers. The Hague, The Netherlands (1975)
20.Kutz O., Wolter F., Zakharyaschev M.: A Note on Concepts and Distances. Working Notes of the 2001 Intl. Description Logics Workshop (DL-2001), Stanford, CA, USA, August 1-3 (2001)113–121
21.Artale A., Franconi E.: Temporal Description Logics. In: Vila L. et al, van Beek P., Boddy M., Fisher M., Gabbay D., Galton A. and Morris R. (Eds.): Handbook of Time and Temporal Reasoning in Artificial Intelligence. MIT Press (1999)
22.Gasquet O., Herzig A., Sahade M.: Terminating Modal Tableaux with Simple Completeness Proof. In: Advances in Modal Logic, 6 (2006) 167-186
23.Horrocks I., Sattler U.: A Tableaux Decision Procedure for SHOIQ. In: Kaelbling L.P., Saffiotti A. (eds.): Proc. of the 19th Int. Joint Conf. on Artificial Intelligence, Edinburgh, Scotland, UK, July 30-August 5, 2005. Professional Book Center (2005) 448–4533.
24.Franceschet M., de Rijke M., Schlingloff B.-H.: Hybrid Logics on Linear Structures: Expressivity and Complexity. In: Proc. of 10th Int. Symposium on Temporal Representation and Reasoning and 4th Int. Conf. on Temporal Logic (TIME-ICTL'2003), Cairns, Queensland, Australia, July 8-10. IEEE Computer Society (2003) 166–173
25.Baader F., Calvanese D., McGuinness D., Nardi D. and Patel-Schneider P.F.: The Description Logic Handbook: Theory, Implementation and Applications. Cambridge University Press (2003)