Temporal Classes and OWL

Natalya Keberle

Chair of IT, Zaporozhye National University, Ukraine
nkeberle@gmail.com

Abstract. Temporal class as a design primitive for knowledge engineering allows for natural representation of evolving concepts of a domain. The paper proposes arguments to employ temporal entities (classes, axioms, ontologies) as first-class citizens in the Semantic Web applications. The paper presents an informal description of one temporal extension of a subset of OWL to deal with temporal classes — OWL-MeT, based on the combination of metric temporal logic and basic description logic with nominals, $\mathcal{MT}$-$\mathcal{ALCO}$. Described are the solutions proposed and the open issues of such a temporal extension.

Introduction and Use Cases

Temporal logics are successfully used for verification of the dynamic systems, e.g. in software and hardware development. Application of temporal logics to the Semantic Web applications is now limited due to the lack of standards on temporal ontology languages and of the reasoning support.

The palette of the use cases utilizing temporal description logics is naturally divided into those cases where usage of temporal concepts allows capturing more (or even all) semantics implicit, and those where temporal concepts and/or axioms serve as a new solution of a known problem.

Use case 1. Natural semantics of a dynamic concept. As noted in [1] it is a common situation when an obvious dynamic concept is modeled statically, hiding or (worse) loosing substantial information about a dynamic concept behavior.

A lot of theoretical examples, including the discussion on the precise definition of a Mortal in [2] show that adding temporal concept constructors to the ontology language enable definition of dynamic concepts.

Use case 2. Ontology evolution analysis. There are several complementing approaches for the problem of ontology evolution analysis. First of all, such kind of analysis can be focused on structural changes occurring in the elements of the ontology during its lifetime. Well-known tools and plug-ins, including PROMPTDIFF (and Prompt Tab in Protégé), OntoView, Changes Tab in Protégé, allow to store, view, seek and analyze the consequences of structural changes. The critique of structural change analysis is also known, it was discussed in [3] and [4]. To sum up, if

there is no version log, usage of heuristics does not provide 100% correct detection of changes, and if the version log exists, the results of change detection procedures can be interpreted differently, and it will be better to write own procedures to look for particular changes across the version log.

Another approach is the analysis of logical changes. The discussion of logical compatibility of ontology versions was initiated in [5]. Here we envisage two use cases. First, the user may want to see the logical difference in addition to / instead of structural one. The steps in this direction are already made, for example, CEX algorithm [6] that calculates the logical difference between two acyclic EL-terminologies. Second, the user may want to pose (via some Web service) a query to an ontology provider server, involving old and new terms and see the result. Such a service might be an external part of the ontology version control system. Types of queries may include subsumption, equivalence, disjointness of old and new terms, entailment of facts from the new ontology with respect to the old terms etc. It's important to say that such queries are naturally extended not only to two ontology versions, but also to several versions. And here we face with the formalism to describe such queries – it should at least address versions, support the elements of the ontology language, support the definition of the axioms, having elements from different versions. One known solution for such type of logical analysis of evolution is realized in MORE framework [7], where queries on different versions of ontology were encoded in XML using temporal tags like "previous version" or similar.

In the light of temporal logic approach, the formal description of the task of logical analysis of ontology evolution can be as follows.

Given a set of ontology versions $\{O_{t_k}\}$ and a time structure $\langle T, \prec \rangle$, the temporalized ontology can be defined as $\left\langle \{O_{t_k}\}_{t_k \in T}, \langle T, \prec \rangle \right\rangle$. In the simplest case a set of version is linearly ordered, and the correspondent time structure is isomorphic to $\langle Z, \leq \rangle$. Let L be a temporal description logic language, able to describe temporalized ontology. A model of L is $\mathrm{M} = < \Delta, \{R_F, R_P\}, I >$, where $\Delta = \{\Delta^k, k \in Z\}$, each Δ^k represents the domain of the interpretation of O_{t_k}, R_F and R_P are temporal accessibility relations and interpretation $I(k) = < \Delta^k, \cdot^{I(k)} >$. Each ontology version has a model in Δ^k. Then answering a general type query $\varphi \in WFF(L)$ over the set of ontology versions is the model checking problem, $\mathrm{M} \models \varphi$.

Temporal concept constructors in the temporal description logic enable the following queries for logical evolution analysis.

Type 1. Presence of an atomic concept *A* or complex (non-temporalized) concept *E* in a version t_k (positive answer means *A* (or *E*) has a model in the ontology version).

$$\mathrm{M} \models A@\{t_k\} \qquad \mathrm{M} \models E@\{t_k\},$$

where *E* is constructed with the help of $\sqcap$, $\sqcup$, $\neg$.

Due to the open world assumption, applicable to OWL as ontology language, straightforward interpretation of such query allows arbitrary concept *A* be satisfiable

in the ontology, even if that concept is not defined in the ontology. Additional actions should be taken to "close" the domain Δ^k .

Type 2. Presence of a complex (non-temporalized) concept *E* in a version defined relatively to a version t_k (*n* versions before/after, in some version before/after, in all versions before/after).

$$\mathrm{M} \models \textit{modality}\ E\ @\{\ t_k\ \},$$

where *modality* is one of temporal operators like "*somepast*", "*allfuture*" etc.

Type 3. Truth of a temporal formula of the general form relative to the ontology versions set.

$$\mathrm{M} \models \varphi$$

Use Case 3. Temporal conceptual modeling. Temporal description logics are known to be an instrument for conceptual modeling of dynamic information [8], [9]. Investigated are several tractable logics, among them is *TDL-Lite*$_{bool}$ [9], expressive enough to translate temporal conceptual models into the set of logic formulae.

It was shown in [10] that UML class diagrams and ER models can be translated to the description logic *DL-Lite*$_{bool}$. Entities can be mapped into concept names, hierarchy of entities into subsumption axioms, *n*-ary relationships are reified etc. It should be pointed that *DL-Lite*$_{bool}$ enables general concept inclusions (GCI) that allow complex concepts be subsumed by or equivalent to other concepts. *TDL-Lite*$_{bool}$ converts a subsumption axiom $C \sqsubseteq D$, where *C* and *D* are time-independent, to the form $\textit{allfuture}(C \sqsubseteq D) \sqcap \textit{allpast}(C \sqsubseteq D)$.

The three use cases described show different required levels of interoperation between temporal and non-temporal part of a temporal description logic language.

OWL+Time=?

Time-related issues in the Semantic Web applications are usually modeled using different formalisms, RDF-based or OWL-based [11]: reification, versioning, 4D-fluents, and temporal logics.

Known solutions on the level of OWL are versioning and fluents [12], on the RDF level – temporal RDF graphs [13] reified using OWL-Time [14].

The semantics of practical temporal description logic language enable non-redundant definition of a dynamic concept. Non-redundancy here means that all the temporal semantics is encoded in such language constructs, supported with the reasoning engine, and no additional action is needed, except for proper definition of the dynamic concept.

Steps towards practical implementation of temporal ontology languages based on temporal logic are made recently. Temporal description logics known in the literature explore interval-based or point-based, linear or branching time structures, assume domain of time be concrete or abstract. The detailed survey of decidability and complexity of reasoning for various temporal extensions of description logics for point-based time structure is presented in [15].

Depending on the time structure, already known are at least three different temporal languages, based on the combination of temporal and description logics, described in the abstract syntax.

TL-OWL – is the implementation of the known language $\mathcal{TL}$-$\mathcal{SHOIN}(\mathcal{D})$, based on $\mathcal{TL}$-$\mathcal{ALCF}$ [16] for interval-based time structure is presented in [17]. The language has the abstract and the exchange syntax defined, together with the model semantics. The authors have also proved the decidability of $\mathcal{TL}$-$\mathcal{SHOIN}(\mathcal{D})$.

TOWL [18] proposes to present time as a concrete domain over real numbers with binary predicates $=, \neq, \leq, <$.

OWL-MeT [4] – is the implementation of the language $\mathcal{MT}$-$\mathcal{ALCO}$ for point-based time structure. The language has the abstract and the exchange syntax[1], RDF semantics defined.

Both TL-OWL and OWL-MeT do not apply temporal operators to the axioms and provide various temporal restrictions, although different for interval-based and point-based time structures. For example, OWL-MeT allows application of temporal operators to concepts (thus making temporal concepts from non-temporal ones), whereas in TL-OWL temporal relations ("before", "during" etc.) are applicable only to temporal variables, which in turn are binded to non-temporal concepts.

Without claiming to be the best solution, OWL-MeT was realized in practice. The details and open issues of the realization are discussed in the next session.

Realization

The experimental realization of a reasoning support for OWL-MeT was made as the extension of Pellet [19] reasoning engine, called Pellet-MeT[2]. The logic underlying OWL-MeT, $\mathcal{MT}$-$\mathcal{ALCO}$, is defined over linear infinite to the future and to the past time line. Correspondent time structure is $\langle Z, \leq \rangle$, where Z is the set of integers, and $\leq$ is reflexive and transitive precedence relation.

The DL-part of the logic is weaker then OWL Lite, particularly in axiomatization of roles. The roots of such simplification are as follows: the first lies in the complexity/decidability issues related to the usage of roles in temporal description logics, and the second is the attempt to incrementally complicate the interoperation of temporal and description logic and to analyze the behavior of the reasoner, having in mind the results of others [20].

Let A denote atomic non-temporal concepts, R – atomic role, E, F – complex non-temporal concepts, C, D – complex temporal concept, $\{o\}$ – object nominal (denoting an individual in some possible world), $\{a\}$ – temporal nominal (denoting possible world, e.g. a point on a time line). Then the rules presented in the Fig.1 generate complex concepts.

[1] The complete syntactical definitions can be found at http://ermolayev.com/owl-met/

[2] Pellet-MeT is available at http://ermolayev.com/owl-met/reasoner.htm

$E, F \rightarrow A \mid top \mid bottom \mid E \sqcap F \mid E \sqcup F \mid \neg E \mid \exists R.E \mid \forall R.E \mid \{o\}$
$C, D \rightarrow E \mid \{a\} \mid C\ intersection\ D \mid C\ union\ D \mid not\ C \mid C@\{a\} \mid future\ n\ C \mid$
$\mid past\ n\ C \mid somefuture\ C \mid somepast\ C \mid allfuture\ C \mid allpast\ C$

Fig. 1. Syntax rules for concepts/roles construction.

If C and D are temporal concepts, then $C\ equivalent\ D$, $C\ subclassof\ D$ are temporal formulae. If φ and ψ are temporal formulae, then $\varphi\ union\ \psi$, $\varphi\ intersection\ \psi$, $not\ \varphi$ are also temporal formulae.

$\mathcal{MT}$-$\mathcal{ALCO}$ is interpreted over Kripke model $M =< \Delta, dist, \{R_F, R_P\}, I, V >$, 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 $\mathcal{ALCO}$-interpretation $I(k) =< \Delta^k, \cdot^{I(k)} >$. Function $den : \{a\} \rightarrow Z$ encodes temporal nominals into integers. For a temporal nominal $\{a\}$, hybrid valuation V assigns $\{a\}$ a unique world $\Delta^{den(a)}$ - singleton subset of Δ.

RDF serialization of OWL-MeT. Temporal classes and temporal restrictions (unnamed classes) are defined with:

```
owlmet:TClass        rdf:type rdf:resource .
owlmet:TRestriction  rdf:type rdf:resource;
                     rdfs:subClassOf owlmet:TClass .
```

All non-temporal classes are actually also temporal (see Fig.1):

```
owl:Class   rdfs:subClassOf   owlmet:TClass .
```

This statement needs some clarification. In OWL-MeT each non-temporal class *A* can be defined as *future 0 A* (at 0 moments to the future *A*). Temporal nominals, defining particular time moments on a time line are presented as owlmet:Instant:

```
owlmet:Instant   rdf:type rdf:resource;
                 rdfs:subClassOf   owlmet:TClass .
```

Temporal operators, such as owlmet:allfuture, owlmet:at, owlmet:happens are defined as instances of rdf:property:

```
owlmet:allfuture   rdf:type      rdf:property;
                   rdfs:domain   owlmet:TRestriction;
                   rdfs:range    owlmet:TClass .
owlmet:at          rdf:type      rdf:property;
                   rdfs:domain   owlmet:TRestriction;
                   rdfs:range    owlmet:Instant .
```

owlmet:happens	rdf:type	rdf:property;
	rdfs:domain	owlmet:TRestriction;
	rdfs:range	owlmet:TClass .

Reasoning support for OWL-MeT. The main reasoning task for Pellet-MeT is checking the consistency of a temporalized ontology. Temporalized ontologies here are considered as sets of TBox axioms having temporal concepts both on the left and the right parts. Additionally, a time structure can be defined as a finite set of temporal nominals, ordered with precedence order. When a time structure is defined, the reasoner uses the temporal nominals and its order during the completion procedure, and may stop the completion if some completion rule requires introduction of a new time moment. Absence of the time structure forces the reasoner to check the consistency of a temporalized ontology over the infinite time line.

The loading procedure of a temporalized ontology includes parsing of temporal constructs in addition to OWL parsing, search for a time structure, construction of a TBox and an ABox. The TBox is then undergoing the standard procedures of normalization and internalization. Specific features of hybrid and metric temporal operators of OWL-MeT allow normalization of some combinations of temporal operators (see [21]).

Consistency checking of a temporalized ontology starts with satisfiability checking of every concept found in the temporalized ontology. For each concept C an Abox $x:\neg C$ is introduced. Predefined temporal nominal $\{now\}$ corresponds to the initial time moment. Tableau rules are divided into description logic tableau rules and temporal/hybrid rules. Rules for $\mathcal{ALCO}$-part remain the same as for description logic.

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 is presented as movement across the tableaux sequence using the accessibility relations, as well as creation of a particular tableau. Completion procedure for an ABox stops in a standard way: either finding a clash in some time moment, or in obtaining consistent and complete tableau.

Open issues. There were discovered several open issues that require specific attention. They are mainly connected with the classification procedure and time structure definition.

At first, subsumption axioms for temporal concepts define the hierarchy of temporal concepts. Every hierarchy needs a root node, and in general, a root node for all temporal classes. The choices can be owlmet:TemporalThing, having owl:Thing as a subclass, and vice versa, owl:Thing can have owlmet:TemporalThing as the subclass. If (and this is proposed in OWL-MeT) every owl:Class is a subclass of owlmet:TClass, then owlmet:TemporalThing will subsume owl:Thing. The same issue arises for the concepts owl:Nothing and owlmet:TemporalNothing.

At second, depending on the choice of the root for all temporal classes, the classification procedure may behave differently, trying to classify a temporalized

ontology where both temporal and non-temporal classes are defined. Temporal class has a model in several static domains, whereas non-temporal classes have models in a static domain. With point-based time structure defined explicitly, the classification may be done after splitting of temporal classes into groups of non-temporal classes belonging to the same (static) domain, addressed with a temporal nominal. Then each group can be classified separately. Another way is to follow the classification procedure for OWL, with owlmet:TemporalThing instead of owl:Thing as a root node.

At third, researchers in the temporal description logics field outline the importance of the properties of a time structure such as linearity/branching, infinity, density etc. The behavior of the underlying logic depends heavily on the combination of these properties. It may be possible to declaratively define the properties of a time structure in the temporalized ontology.

Conclusions

Temporal description logic without application of temporal operators to the axioms and allowing temporal classes may verify the hierarchies of time-dependent entities of a temporal conceptual model, describe dynamic concepts of a domains, serve as a language for evolution analysis. Continuous research in the theoretical backgrounds of temporal description logics gives certainty that practical implementation of different OWL-based temporal languages is not far off. Starting from simple and obvious languages and reasoning strategies the common grounding for consistent and practical research in the field can be find.

The directions for future research include the development of the formalism for the description of temporalized axioms, alternative sets of temporal operators (*until/since*, *until/next* and others) in addition to metric ones, more rigorous description of a time structure.

References

[1] Baader, F., Ghilardi, S., Lutz, C.: LTL over Description Logic Axioms. In: Baader, F., Lutz, C., Motik, B. (eds.) 21st International Workshop on Description Logics (DL'08)

[2] Artale, A., Franconi, E.: Temporal Description Logics. In: Fisher, M., Gabbay, D., Vila, L. (eds.) Handbook of Time and Temporal Reasoning in Artificial Intelligence, 1, pp. 375--388. Elsevier, Amsterdam (2005)

[3] Plessers, P., de Troyer, O.: Ontology Change Detection using a Version Log. In: Gil, Y. et al (eds.) The Semantic Web – ISWC 2005. LNCS, vol. 3729, pp. 578--592. Springer, Berlin/Heidelberg (2005)

[4] Keberle, N., Litvinenko, Y., Gordeyev, Y., Ermolayev, V.: Ontology Evolution Analysis with OWL-MeT. In: Flouris, G., d'Aquin, M. (eds.) Workshop on Ontology Dynamics (IWOD 2007), pp. 1--12, http://kmi.open.ac.uk/events/iwod/papers/paper-05.pdf

[5] Klein, M.C.A., Fensel, D.: Ontology versioning on the Semantic Web. In: 1st Semantic Web Working Symposium, pp. 75--91 (2001)

[6] Konev, B., Lutz, C., Walther, D., Wolter, F.: Logical Difference and Module Extraction with CEX and MEX. In: Baader, F., Lutz, C., Motik, B. (eds.) 21st International Workshop on Description Logics (2008)
[7] Huang, Z., ten Teije, A., van Harmelen, F.: MORE2: An Extended Reasoning and Management System for Multi-version Ontologies. Deliverable D 3.5.3 (WP3.5), EU-IST Integrated Project (IP) IST-2003-506826 SEKT. Amsterdam, Vrije University (2007)
[8] Artale, A., Franconi, E., Mandreoli, F.: Description Logics for Modeling Dynamic Information. In: Chomicki, J., van der Meiden, R., Saake, G. (eds.) Logics for Emerging Applications of Databases 2003 [Outcome of a Dagstuhl seminar], pp. 239--275. Springer, Berlin/Heidelberg (2003)
[9] Artale, A., Kontchakov, R., Lutz, C., Wolter, F., Zakharyaschev, M. Temporalising Tractable Description Logics. In: 14th International Symposium on Temporal Representation and Reasoning (TIME'07), pp. 11--22. IEEE Computer Society Press (2007)
[10] Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R.: DL-Lite: Tractable description logics for ontologies. In: Veloso, M. M., Kambhampati, S. (eds.) 20th National Conference on Artificial Intelligence (AAAI 2005), pp. 602--607. AAAI Press / The MIT Press (2005)
[11] Baratis, E., Petrakis, E.G.M., Batsakis, S., Maris, N., Papadakis, N.: TOQL: Temporal Ontology Querying Language. In: Mamoulis, N., Seidl, Th., Pedersen, T. B., Torp, K., Assent, I. (eds.) SSTD-2009. LNCS, vol. 5644, pp. 338--354. Springer, Berlin/Heidelberg (2009)
[12] Welty, C., Fikes, R.: A Reusable Ontology for Fluents in OWL. Frontiers in Artificial Intelligence and Applications, 150, 226--236 (2006)
[13] Gutierrez, C., Hurtado, C.A., Vaisman, A.: Introducing Time into RDF. IEEE Trans. Knowledge and Data Engineering, 19 (2), 207--218 (2007)
[14] Time Ontology in OWL, http://www.w3.org/TR/owl-time
[15] Lutz, C., Wolter, F., Zakharyaschev, M.: Temporal Description Logics: A Survey. 15th International Symposium on Temporal Representation and Reasoning (TIME 2008), pp. 3--14. IEEE Society Press (2008)
[16] Artale, A., Franconi, E.: A Temporal Description Logic for Reasoning about Actions and Plans. J. Artificial Intelligence Research, 9, 463--506 (1998)
[17] Kim, S.-K., Song, M.-Y., Kim, C., Yea, S.-J., Jang, H.C., Lee, K.-C.: Temporal Ontology Language for Representing and Reasoning Interval-Based Temporal Knowledge. In: 3rd Asian Semantic Web Conference on the Semantic Web. LNCS, vol. 5367, pp. 31—45. Springer, Berlin/Heidelberg (2008)
[18] Milea, V., Frasincar, F., Kaymak, U., Noia, T.: An OWL-based Approach Towards Representing Time in Web Information Systems. In: Frasincar, F., Houben, G.-J., Thiran, P. (eds.) 4th International Workshop of Web Information Systems Modeling (WISM 2007), pp.791--802,
http://people.few.eur.nl/frasincar/workshops/wism2007/wism2007proceedings.pdf
[19] Sirin, E., Parsia, B., Cuenca Grau, B., Kalyanpur, A., Katz, Y.: Pellet: A Practical OWL-DL Reasoner. J. of Web Semantics: Science, Services and Agents on the World Wide Web, 5(2), 51--53 (2007)
[20] Günsel, C. A Tableaux-Based Reasoner for Temporalised Description Logics. Doctoral Thesis of the Univ. of Liverpool (2005)
[21] Keberle, N.G.: Properties of Propositional Metric Temporal Calculus for Description of Evolving Conceptualization. Problems of Applied Mathematics and Mathematical Modelling, pp.80--99. Dniepropetrovsk, DNU Press (2006), http://ermolayev.com/owl-met/owl-met-pubs/PTC-MT-mssai-05.pdf