Temporal Optimizations and Temporal Cardinality in the tOWL Language

Viorel Milea*
Flavius Frasincar
Uzay Kaymak

Econometric Institute,
Erasmus University Rotterdam,
P.O. Box 1738, 3000 DR Rotterdam, the Netherlands
Fax: +31-10-4089162
E-mail: {milea,frasincar,kaymak}@ese.eur.nl
*Corresponding author

Geert-Jan Houben

Web Information Systems Group,
Software Technology Department,
Delft University of Technology,
PO Box 5031, 2600 GA Delft, the Netherlands
Email: g.j.p.m.houben@tudelft.nl

Uzay Kaymak is also associated with the
School of Industrial Engineering,
Eindhoven University of Technology,
P.O. Box 513, 5600 MB, Eindhoven, the Netherlands

Abstract: The tOWL language is a temporal Web ontology language based on OWL-DL without nominals. The language enables the representation of time and time-related aspects, such as state transitions. The design choices of the language pose new challenges from a temporal perspective. One such challenge is the representation of temporal cardinality. Another challenge consists of optimizing the temporal representations in order to reduce the number of axioms. One such optimization is temporal coalescing, which merges concepts that are associated with time intervals that either meet or share at least one instant with each other. In this paper we formally introduce these concepts into the tOWL language and illustrate how they can be applied.

Keywords: tOWL, temporal coalescing, temporal cardinality.

Biographical notes: Viorel Milea obtained the M.Sc. degree in Informatics & Economics from Erasmus University Rotterdam, the Netherlands, in 2006. Currently, he is working towards his Ph.D. degree at the Erasmus University Rotterdam, the Netherlands. The focus of his Ph.D. is on employing Semantic Web technologies for enhancing

the current state-of-the-art in automated trading with a focus on processing the information contained in economic news messages and assessing its impact on stock prices. His research interests cover areas such as Semantic Web theory and applications, intelligent systems in finance, and nature-inspired classification and optimization techniques.

Flavius Frasincar obtained the M.Sc. degree in computer science from Politehnica University Bucharest, Romania, in 1998. In 2000, he received the professional doctorate degree in software engineering from Eindhoven University of Technology, the Netherlands. He got the Ph.D. degree in computer science from Eindhoven University of Technology, the Netherlands, in 2005. Since 2005, he is assistant professor of information systems at Erasmus University Rotterdam, the Netherlands. He has published in numerous conferences and journals in the areas of databases, Web information systems, personalization, and the Semantic Web. He is a member of the editorial board of the International Journal of Web Engineering and Technology.

Uzay Kaymak received the M.Sc. degree in electrical engineering, the Degree of Chartered Designer in information technology, and the Ph.D. degree in control engineering from the Delft University of Technology, Delft, the Netherlands, in 1992, 1995, and 1998, respectively. From 1997 to 2000, he was a Reservoir Engineer with Shell International Exploration and Production. He is currently Professor of Intelligence and Computation in Economics with the Econometric Institute, Erasmus University Rotterdam, the Netherlands. He is also affiliated with the School of Industrial Engineering of the Technical University of Eindhoven. Prof. Kaymak is an associate editor of IEEE Transactions on Fuzzy Systems and is a member of the editorial board of several journals.

Geert-Jan Houben holds a doctorate in computer science from the Eindhoven University of Technology (TU/e), the Netherlands (1990). Since then he has been working as an assistant and associate professor at the TU/e, as an IT-consultant with several consultancy firms in the Netherlands, as a guest-professor at the University of Antwerp, Belgium, as a guest-researcher at the Centre for Mathematics and Computer Science (CWI), the Netherlands, and as a full professor in information systems at the Free University of Brussels (VUB), Belgium. Since the summer of 2008 he is working as a full professor in Web-based information systems at Delft University of Technology (TUD), the Netherlands, performing research on large-scale information systems, specifically information systems that involve Web and Semantic Web technology. He is member of editorial boards such as IJWET, IJWS or ACM TWEB.

1 Introduction

tOWL is a temporal Web ontology language based on the $\mathcal{SHIN}(\mathcal{D})$ description logic, a subset of OWL-DL (OWL-DL without nominals) (Milea et al., 2007, 2008; Frasincar et al., 2010). It addresses shortcomings of OWL-DL (Patel-Schneider et al., 2004) by enabling temporal representations in ontologies. The tOWL language is focussed on the representation of concrete time, in the form of instants

and intervals, and change, such as involved in the representation of processes. For the representation of concrete time, the tOWL language relies on a concrete domains extension to the language. For the representation of change, tOWL relies on a four-dimensional representation based on fluents and timeslices.

The language enables different temporal representations in ontologies, from more simple aspects such as the change in the attribute value of a property, to more complex aspects such as processes and the associated state transitions. Such representations are not possible in OWL-DL as this ontology language does not provide any semantics of time. Therefore, any representation of change or processes in OWL-DL will be limited by static semantics, and will not enable any inference related to the temporal aspects of the representation. In a tOWL representation of a process, for example, one could infer the state of the process in which an entity finds itself based on information of the states an entity has already transitioned.

In this paper we develop the tOWL language further by discussing temporal coalescing in this context, and increasing the expressiveness of the language through temporal cardinality. Temporal coalescing is an operation similar to duplicate elimination in databases, applied to a temporal context (Bohlen et al., 1996). In the case of the tOWL language, this relates to merging timeslices that represent the same relation over time intervals that meet or share at least one instant. The motivation for temporal coalescing in tOWL knowledge bases is two-fold. One aspect relates to the proliferation of objects in the knowledge base, inherent to a timeslice approach where new timeslices are created every time something is changing. By temporally coalescing the knowledge base, the number of timeslices can be greatly reduced. Another aspect relates to posing temporal queries upon the knowledge base.

Representing change by means of a 4d approach based on timeslices and fluents in the tOWL language also poses some interesting challenges to some of the static OWL-DL concepts. One such concept is the *cardinality* construct, that is employed in OWL for restricting the number of attribute values that a property is allowed to take. For example, an imaginary *hasBiologicalFather* property in OWL-DL, indicating the individual representing the biological father of another individual, should have a cardinality of exactly one, as any person can have exactly one biological father. Such a cardinality construct is also relevant from a temporal context, where one might want to represent that, *at any point in time*, a person may have exactly one biological father. Introducing the temporal cardinality construct in tOWL increases the expressiveness of the language when regarded from a temporal context and allows for more accurate representations of the world.

Several optimizations are possible in the tOWL language, optimizations that can be related directly to the language or optimizations at the reasoner level. At language level, a possible optimization relates to making a distinction between datatype fluents and object fluents. By not requiring the creation of timeslices for datatypes everytime a concrete value is changing, the number of timeslices is considerably reduced. Another optimization at language level could involve the definition of the time for which timeslices hold as a union of intervals, thus preventing the creation of separate timeslices for each interval in the union and reducing the number of timeslices in the knowledge base. This latter optimization is not considered in the current work. At reasoner level we consider temporal

coalescing as a technique for reducing the number of timeslices in the knowledge base and enabling a large number of queries to be correctly answered when posed on the coalesced knowledge base.

Temporal cardinality and temporal coalescence are also of practical relevance in a Semantic Web context. The concept of temporal cardinality, for example, can be employed in a variety of applications, such as online retailers or corporate knowledge bases, to name a couple. In the case of online stores, one could envision restrictions posed in the store's knowledge base, such as each customer having exactly one customer number at any point in time. In corporate knowledge bases one might want to enforce the restriction that at any point in time, an employee has at least one direct manager. In the same context, temporal coalescing comes to increase the number of queries that can be correctly answered when posed upon a coalesced knowledge base. For example, one employee, e.g., consultant, works for a company for two years, during which period he has a temporary contract. In a tOWL knowledge base, timeslices are instantiated for this fact and the employment relationship is represented. After two years, the employee's contract is extended for an additional period of three years. In the tOWL knowledge base, new timeslices are created to represent this employment relationship for a period of three years. Provided the tOWL knowledge base is uncoalesced, queries posed by, for example, clients of the company on the company's website, seeking to retrieve the consultants with at least four years of experience in the company, will fail to retrieve the employee discussed in this paragraph, since no relationship describes the employment relationship for a period longer than three years. The coalesced knowledge base would enable this consultant to be recongnized as an individual working for the company for five years.

This paper is organized as follows. In Section 2 we discuss work related to the research presented in this paper. In Section 3 we present an overview of the tOWL language and illustrate how the envisioned optimizations and new constructs that we seek to introduce affect the language. In Section 4 we present temporal coalescing in the context of the tOWL language. Section 5 discusses temporal cardinality for the tOWL language. Finally, we conclude in Section 6.

2 Temporal Coalescing and Temporal Cardinality

In the context of the Semantic Web, a number of approaches have already been designed, addressing different temporal aspects in relation to ontology languages. A rather extensive approach towards extending ontology languages with a temporal dimension is Temporal RDF (Gutierrez et al., 2007). This work is similar to the tOWL language as it concerns the ability to represent temporal information in ontologies, but differs in that the language considered is the Resource Description Framework (RDF). Another approach is OWL-Time, which focusses on OWL rather than RDF. The initial purpose behind the design of a time ontology (OWL-Time) (Hobbs et al., 2004) was to represent the temporal content of Web pages and the temporal properties of Web Services. This approach is rather extensive in describing quantitative time and the qualitative relations that may exist among instants and intervals. Being based on OWL-DL, it employs the underlying $\mathcal{SHOIN}(\mathcal{D})$ description logic and thus relies on datatypes rather than

Name	Employer	Time
John	Yahoo	[01/01/2001 14/02/2003)
John	Yahoo	[15/02/2003 16/07/2005)

Table 1 Uncoalesced relation.

Name	Employer	Time
John	Yahoo	[01/01/2001 16/07/2005)

Table 2 Coalesced relation.

concrete domains for the description of instants and intervals, while tOWL uses concrete domains for the representation of concrete time. The problem of change in ontologies has also been addressed in the context of ontology evolution, such as in (Noy et al., 2004) and (Haase et al., 2005). However, the problem of ontology evolution relates to changing TBoxes, while tOWL focusses on representing change at ABox level. These approaches do not attempt to deal with concepts such as temporal cardinality or temporal coalescing.

The concept of temporal coalescing can best be illustrated with the example provided in tables 1 and 2. Table 1 describes John as being an employee of Yahoo over two adjacent intervals. Clearly, queries posed to this knowledge base of the form *employees of Yahoo who have worked for this employer for at least 3 years* cannot be answered correctly in the case of John, since both intervals associated with this description are shorter than 3 years. In such cases, and in cases where the intervals share at least one instant instead of being adjacent, in the presence of equivalent attribute values, the relations can be summarized into a single relation without loss of information. The desired result is presented in Table 2.

A similar definition of coalescence is employed in the XBit datamodel (Wang et al., 2004), while a discussion of temporal coalescence in RDF is provided in (Grandi, 2009). The problem of temporal coalescence was addressed in the context of temporal databases (Bohlen et al., 1996; Dyreson, 2003). In (Bohlen et al., 1996), the authors address the coalescing of timestamped tuples where the attribute values are equal and the timestamps associated with the tuples either meet or overlap (for a formal description of the temporal relations that may exist between time intervals, such as meet and overlap, we refer the reader to (Allen, 1983)). As is the case for this paper, in (Bohlen et al., 1996) the focus is on valid time. It should also be noted that the tOWL language, our current context for temporal coalescing, only focuses on valid time in the representation of temporal relations. The work in (Bohlen et al., 1996) is similar to parts of the work presented in this paper, with the crucial difference that we address temporal coalescence in the context of timeslices connected by fluents, rather than tuples with identical attribute values. Thus, our focus is on those timeslices that represent the same individuals and are connected by equivalent fluents, and hold over temporal intervals that either meet or share at least one instant amongst them. Finally, an SQL implementation of temporal coalescence for Oracle is provided in (Bohlen et al., 1996).

Since tOWL allows different time granularities in the language, the approach described in (Dyreson, 2003) presents some overlap with the current work. The authors discuss the problem of temporal coalescing, and also take into account the possibility that the coalesced relations may be associated with timestamps at different levels of granularity.

Temporal cardinality is a constraint used to limit the number of values an attribute can have over the life-time of its entity (**?**Zimanyi et al., 1997; Spaccapietra et al., 1998). The problem of expressing temporal cardinality in the context of the tOWL language has been preliminarily addressed in (Milea et al., 2008). Here, the authors develop an approach for moving beyond the expression of fluent cardinality, enabled by OWL, and represent cardinality also when overlapping timeslices are involved. The issue of overlapping timeslices is also briefly discussed in (Welty et al., 2006) in the context of a reusable OWL ontology for fluents. A distinction can be made in terms of temporal cardinality between snapshot cardinality, a temporal cardinality that holds over a limited period of time, and lifetime cardinality, a constraint holding across the whole lifetime of an entity (Tauzovich, 1991; Wijsen, 1999). The current work focuses on lifetime cardinality. Some discussion on the concept of snapshot cardinality is also provided in (Artale et al., 2010) in the context of the DL-Lite description logic. The same concept in the context of the Temporal ER model is described in (Gregersen et al., 2002).

3 The tOWL Language

In this section we provide a general description of the tOWL language. This description is not meant to be exhaustive, and for more in-depth discussions of tOWL we refer the reader to (Milea et al., 2007, 2008; Frasincar et al., 2010). This description is presented in Section 3.1. A discussion of the issues that we address in this paper relative to the tOWL language is presented in Section 3.2.

3.1 General Description of the Language

The tOWL language (Milea et al., 2007, 2008; Frasincar et al., 2010) is a temporal Web ontology language based on the $\mathcal{SHIN}(\mathcal{D})$ description logic, an expressive subfragment of OWL-DL (Patel-Schneider et al., 2004). An overview of the different layers introduced by the tOWL language on top of OWL-DL is provided in Figure 1. The language enables the representation of time and time-related aspects, such as change. For the representation of time, the tOWL language relies on concrete domains, and enables both instant-based as well as interval-based representations, as well as the relations that may exist between instants and intervals (such as Allen's 13 interval relations (Allen, 1983) in the case of intervals). For the representation of more complex aspects, such as change, the tOWL language is designed around a 4-dimensional view of the world. In this view, timeslices are employed to represent otherwise static OWL individuals across temporal intervals, and fluents, a type of temporal property, are employed to indicate what is changing. This design enables the representation of, for example, processes, and the associated state transition axioms. An example of how a

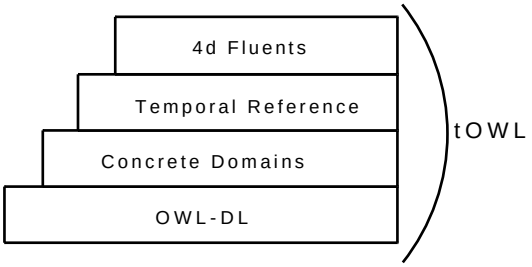

Figure 1 tOWL layer cake.

complex process, in this case a leveraged buyout process, can be represented in the tOWL language is given in (Frasincar et al., 2010). The focus of the language is solely on valid time, i.e., the time when an axiom is true in the real world.

The timeslices-based representation has the ability to determine, at any point in time, what holds true. In order to employ this representation, one has to create timeslices for the static individuals that are involved in a relation that is ephemeral in nature. For example, if one wants to represent the changing CEO of a company, and the ontology contains static individuals that represent both the person that is a CEO, as well as the company, then timeslices have to be instantiated for both these static concepts. Upon having done this, the two timeslices can be connected by a fluent, such as the *hasCEO* fluent, to indicate that, over the time interval associated with the timeslices, the two timeslices are in the *hasCEO* relationship. This is illustrated in Figure 2.

In the example presented in Figure 2, two OWL classes have been defined, namely *Company* and *Person*. For each of these classes, one individual is instantiated, namely *iGoogle*, representing the company Google, an instance of the *Company* class, and *iEricSchmidt*, representing the individual with the name Eric Schmidt, an instance of the *Person* class. For each of these individuals, a timeslice is instantiated, namely *iGoogle_TS1* and *iEricSchmidt_TS1*, respectively. These timeslices both hold over the same interval, *iInterval1*, a consequence of the design of the tOWL language (fluents can only connect timeslices that hold over the same interval), and thus represent the static individuals with which they are associated over that interval. To denote that Eric Schmidt is the CEO of Google over the period denoted by *iInterval1*, we connect the two timeslices by the *hasCEO* fluent. In this way, we represent that Eric Schmidt was the CEO of Google over the interval *iInterval1* by employing two timeslices and a fluent property.

3.2 Representational and Reasoning Issues in the tOWL Language

Temporal coalescing is an operation similar to duplicate elimination in databases, applied to a temporal context (Bohlen et al., 1996). In the case of the tOWL language, this relates to merging timeslices that represent the same relation over temporal intervals that either meet or share at least one instant with each other. Both the case of meeting timeslices, as well as the case of overlapping timeslices, are depicted in Figure 3.

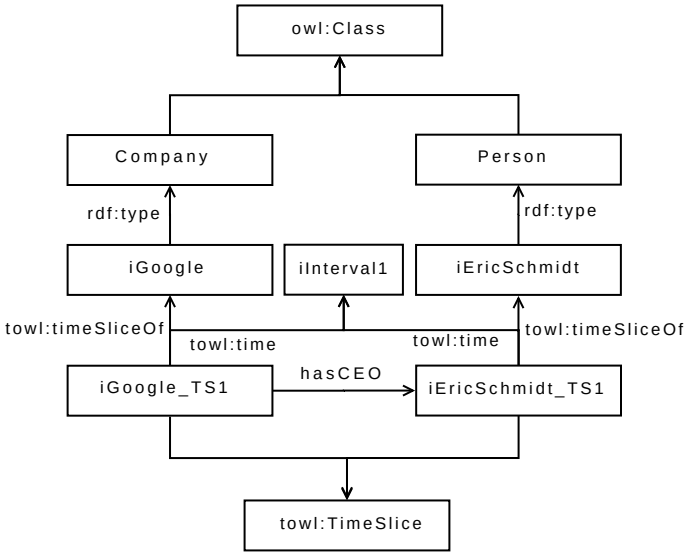

Figure 2 Representing change in tOWL.

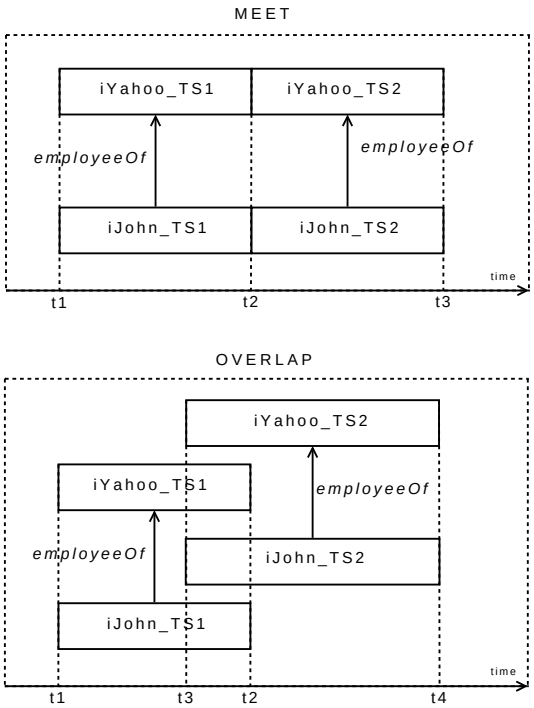

Figure 3 tOWL coalesce candidates.

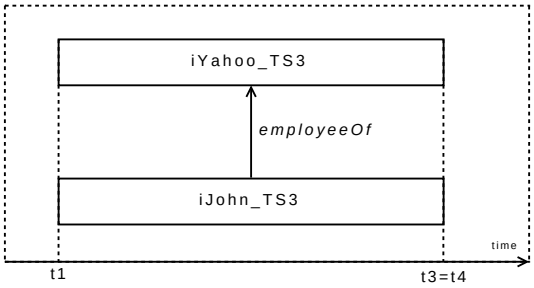

Figure 4 tOWL coalesced timeslices.

The desired result of temporal coalescing in the case of the tOWL language is depicted in Figure 4. Here we depict the cases of meeting as well as overlapping timeslices, and the resulting, merged timeslice. For the case of meeting timeslices, a new timeslice is created holding from the beginning timestamp of the timeslice that appears chronologically first (t_1), until the ending timestamp of the timeslice that appears chronologically second (t_3). Upon removal of the two original timeslices, the knowledge base is simplified with no loss of information. The lack of loss of information comes from the design of the language, since timeslices are not reused in the representation of temporal information. Thus, when removing coalesced timeslices, we do not encounter references to other timeslices (such as a fluent relation between the removed timeslice and another timeslice in the knowledge base) that might be lost once the coalesced timeslices are removed. This is due to the fact that timeslices are created everytime something is changing, thus not relying on the reuse of timeslices for representation purposes. For this reason, we do not consider the problem of referential integrity (Widagdo, 2007; Steiner et al., 1997) in our current approach, and assume that the timeslices that are coalesced have no references to/from other timeslices.

The same procedure applies to the case of overlapping timeslices, where the resulting, merged timeslice again holds from the beginning of the timeslice that appears chronologically first (t_1) until the end of the timeslice that appears chronologically second (t_4). A similar mechanism is applied when the intervals can be described by the equal, starts, finishes, and during Allen relationships, as well as the inverses of the latter three relationships.

The motivation for temporal coalescing in tOWL knowledge bases is two-fold. One aspect relates to the proliferation of objects in the knowledge base, inherent to a timeslice approach where new timeslices are created every time something is changing. By temporally coalescing the knowledge base the number of timeslices can be greatly reduced. Another aspect relates to posing temporal queries upon the knowledge base. In the case of the *employeeOf* relationship where the two timeslices meet, one could query whether John was an employee of Yahoo during the $[t_1, t_3)$ time interval. Since, in the original state, no relationship describes the link between John and Yahoo over the whole time interval, but rather subintervals hereof, this query would be evaluated as the empty set. The temporally coalesced version of this knowledge base correctly describes John as being an employee of

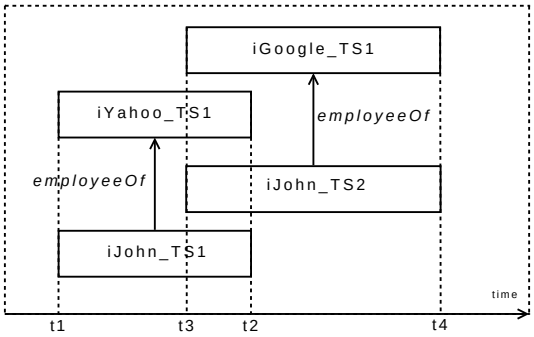

Figure 5 tOWL temporal cardinality example.

Yahoo during the whole time interval, thus enabling more queries on the knowledge base than previously.

Representing change by means of a 4d approach based on timeslices and fluents in the tOWL language also poses some interesting challenges to some of the static OWL-DL concepts. One such concept is the *cardinality* construct, that is employed in OWL for restricting the number of attribute values that a property is allowed to take. For example, an imaginary *hasBiologicalFather* property in OWL-DL, indicating the individual representing the biological father of another individual, should have a cardinality of exactly one, as any person can have exactly one biological father. Such a cardinality construct is also relevant from a temporal context. We might want to state, returning to our employee example, that John may be an employee of not more than one company, at a time. With this assumption in mind, we depict an example violating this temporal cardinality constraint in Figure 5. As can be observed from this depiction, there exists an interval $[t_3, t_2)$ over which John is both an employee of Yahoo, as well as Google, thus violating our temporal cardinality constraint stating that John may be an employee of at most one company, at any time (assuming that Google is different from Yahoo).

The static cardinality construct that tOWL inherits from OWL-DL can be applied directly to the fluent *employeeOf*, stating that this fluent may take at most one value. However, as depicted in Figure 5, our cardinality constraint may be violated in a temporal context without violating the static cardinality constraint attached to the fluent, thus rendering the OWL-DL cardinality construct not expressive enough in a temporal setting.

The motivation for the representation of temporal cardinality in tOWL arises mainly from the increased expressiveness that this construct has to offer. Returning to the employee example, many such situations can be pictured where the concept of cardinality is not only relevant from the static perspective, but also from the temporal one. Increasing the expressiveness of the language with temporal cardinality will provide for more precise descriptions of the world, at least when regarded from a temporal point of view.

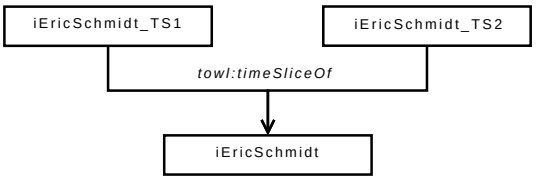

Figure 6 Individual equivalence in tOWL.

4 Temporal Coalescing in tOWL

In this section we introduce temporal coalescing in the tOWL language. Temporal coalescing implies merging timeslices that represent the same individual and are connected to timeslices of the same individual by the same fluent, over temporal intervals that either meet or share at least one instant amongst them. This operation is similar to duplicate elimination in databases (Bohlen et al., 1996). Temporal coalescing in tOWL has two desirable consequences, namely the reduction of the proliferation of objects in the knowledge base, and resolving temporal queries in an appropriate way.

We introduce the concept of *individual equivalence*, which is illustrated in Figure 6, where the timeslices *iEricSchmidt_TS1* and *iEricSchmidt_TS2* are individual equivalent.

Definition 1 (Individual equivalence)
Two timeslices are individual equivalent (`ie`) if they are connected to the same static individual, i.e., they represent the same individual over arbitrary temporal intervals.

$$\begin{aligned}\texttt{ie}(m,n) \equiv m,n \in \texttt{towl:TimeSlice}^{\mathcal{I}} \wedge (\exists p,q \in (\neg(\texttt{towl:TimeSlice} \sqcup \\ \texttt{towl:Interval} \sqcup \texttt{rdfs:Literal}))^{\mathcal{I}}, (m,p) \in \texttt{towl:timeSliceOf}^{\mathcal{I}} \\ \wedge (m,q) \in \texttt{towl:timeSliceOf}^{\mathcal{I}} \wedge (p,q) \in \texttt{owl:sameAs}^{\mathcal{I}})\end{aligned}$$

We also introduce the concept of *fluent equivalence* and illustrate this concept in Figure 7, where *iEricSchmidt_TS1* and *iEricSchmidt_TS2* are fluent equivalent.

Definition 2 (Fluent equivalence)
Two timeslices are fluent equivalent (`fe`) if the values of any object fluent that is connected to these timeslices are individual equivalent and the values of any datatype fluent are the same.

$$\begin{aligned}\texttt{fe}(m,n) \equiv m,n \in \texttt{towl:TimeSlice}^{\mathcal{I}} \wedge (\forall f \in \texttt{towl:FluentObjectProperty}^{\mathcal{I}} \\ \exists p,q \in \texttt{towl:TimeSlice}^{\mathcal{I}}\ (m,p) \in f^{\mathcal{I}} \wedge (n,q) \in f^{\mathcal{I}} \wedge \texttt{ie}(p,q)) \wedge \\ (\forall f \in \texttt{towl:FluentDatatypeProperty}^{\mathcal{I}}\ \exists p,q \in \texttt{owl:Datatype}^{\mathcal{I}} \\ (m,p) \in f^{\mathcal{I}} \wedge (n,q) \in f^{\mathcal{I}} \wedge (p=q))\end{aligned}$$

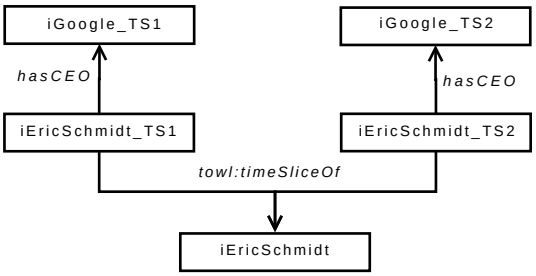

Figure 7 Fluent equivalence in tOWL.

Finally, we introduce the concept of *temporal relatedness* as a temporal relationship between timeslices. This concept has already been illustrated in Figure 3, where, for example, the timeslices *iJohn_TS1* and *iJohn_TS2* are temporally related.

Definition 3 (Temporal relatedness)
Any pair of timeslices is temporally related (`tr`) if the intervals over which they are defined can be described by one of following Allen's interval relations: equal, meet, overlaps, starts, during, or finishes. We use only the direct Allen relations and not their inverses, as the order of timeslices m and n, and thus of variables p and q, is not important here (we can always swap these to obtain the direct relations if needed).

$$\begin{aligned}\texttt{tr}(m,n) \equiv m,n \in \texttt{towl:TimeSlice}^{\mathcal{I}} \wedge (\exists p,q \in \texttt{towl:TimeInterval}^{\mathcal{I}} \\ (m,p) \in \texttt{towl:time}^{\mathcal{I}} \wedge (n,q) \in \texttt{towl:time}^{\mathcal{I}} \wedge \\ (p,q) \in \texttt{allen:meets}^{\mathcal{I}} \vee (p,q) \in \texttt{allen:overlaps}^{\mathcal{I}} \vee \\ (p,q) \in \texttt{allen:starts}^{\mathcal{I}} \vee (p,q) \in \texttt{allen:finishes}^{\mathcal{I}} \vee \\ (p,q) \in \texttt{allen:during}^{\mathcal{I}} \vee (p,q) \in \texttt{allen:equal}^{\mathcal{I}})\end{aligned}$$

Having defined individual equivalence, fluent equivalence and temporal relatedness, we can move on to introduce the concept of timeslices that are apt for temporal coalescence, i.e., timeslices that can be merged into a new timeslice representing the previous two timeslices over the union of the intervals over which the latter are defined.

Definition 4 (Binary aptness for temporal coalescence)
Any pair of timeslices is apt for temporal coalescing (`coal2`) if the timeslices are individual equivalent, fluent equivalent, and temporally related.

$$\texttt{coal2}(m,n) \equiv m,n \in \texttt{towl:TimeSlice}^{\mathcal{I}} \wedge \texttt{ie}(m,n) \wedge \texttt{fe}(m,n) \wedge \texttt{tr}(m,n)$$

Having defined aptness for temporal coalescence when two timeslices are involved, we can define this aptness for each individual timeslice, such that, given any timeslice in the knowledge base, it can be determined whether this timeslice can be merged with any other timeslice that satisfies the given conditions.

Definition 5 (Unary aptness for temporal coalescence)
Any individual timeslice is apt for temporal coalescing (`coal1`) if there exists some other timeslice such that the pair consisting of these two timeslices is apt for temporal coalescing.

$$\texttt{coal1}(m) \equiv m \in \texttt{towl:TimeSlice}^{\mathcal{I}} \wedge (\exists n \in \texttt{towl:TimeSlice}^{\mathcal{I}}\ \texttt{coal2}(m, n))$$

Provided that two timeslices, m and n satisfy the $\texttt{coal2}(m, n)$ relationship, we perform temporal coalescing according to the algorithm presented in Algorithm 1. Starting from these 2 timeslices, we begin by creating a new timeslice that will represent the coalesced relation over the merged time intervals, and attach the static individual describing m and n to the new timeslice. Additionally, we merge the intervals associated to m and n and attach the thus obtained interval to the new timeslice. Finally, we attach all fluents describing the 2 timeslices to the newly created timeslice, and complete the process by deleting the timeslices m and n that initiated the process.

Algorithm 1
($*$ Temporal coalescing algorithm $*$)
Input: m and n timeslices which are apt for temporal coalescing
Output: p timeslice which is obtained by coalescing m and n
1. create new timeslice *ts*
2. attach the static individual linked to m and n as static individual of *ts*
3. start(i) = min(start(time(m)), start(time(n)))
4. end(i) = max(end(time(m)), end(time(n)))
5. attach interval i as interval of *ts*
6. **for** all fluents f attached to m
7. **do** attach fluents f to *ts*
8. delete timeslices m and n

5 Temporal Cardinality in tOWL

The concept of cardinality in OWL-DL, which we denote as static cardinality, relates to the number of values that may be assigned to a property. It is used for representing the number of attribute values that may describe, through some static property, an OWL individual. In this static context, one might for example want to represent that a bicycle has exactly two wheels, namely the front wheel and the back wheel. Thus, when assigning attribute values to a property *hasWheels*, one must invariably assign both a front wheel and a back wheel to the individual being described. Static cardinality comes in three flavours. In addition to describing

the exact cardinality one might also define the minimum cardinality, i.e., the minimum number of attribute values describing an individual through some property, or, conversely, the maximum cardinality. Formally, these three concepts can be described as in (Bechhofer et al., 2004):

- *minCardinality*: if stated to have the value a on a property P, with respect to a class C, then any instance of C will be related through P to at least a semantically distinct values (individuals or data values) (of which the type may further be restricted by the range of P);
- *maxCardinality*: if stated to have the value a on a property P, with respect to a class C, then any instance of C will be related through P to at most a semantically distinct values (individuals or data values) (of which the type may further be restricted by the range of P);
- *cardinality*: if stated to have the value a on a property P, with respect to a class C, then any instance of C will be related through P to exactly a semantically distinct values (individuals or data values) (of which the type may further be restricted by the range of P). In other words, both a *minCardinality* of a and a *maxCardinality* of a are simultaneously satisfied.

The static cardinality implies the ability to determine when timeslices are equal, thus avoiding counting the latter more than once. Equality of timeslices can be determined by the following formula, where two timeslices m, n are considered to be equal if they hold over the same interval and describe the same static individual.

$$\begin{aligned}\texttt{equal}(m,n) \equiv (\exists o,p \in \texttt{towl:TimeInterval}^{\mathcal{I}}\ (m,o) \in \texttt{towl:time}^{\mathcal{I}} \wedge \\ (n,p) \in \texttt{towl:time}^{\mathcal{I}} \wedge (o,p) \in \texttt{allen:equal}^{\mathcal{I}}) \wedge \\ (\exists q,r \in \texttt{owl:Class}^{\mathcal{I}}\ (m,q) \in \texttt{towl:timeSliceOf}^{\mathcal{I}} \wedge \\ (n,r) \in \texttt{towl:timeSliceOf}^{\mathcal{I}} \wedge (q,r) \in \\ (\neg(\texttt{towl:TimeSlice} \sqcup \texttt{towl:Interval} \sqcup \texttt{rdfs:Literal}))^{\mathcal{I}})\end{aligned}$$

In a temporal context in the tOWL language, the concept of static cardinality can be employed to describe the cardinality of a fluent. Returning to the example presented in Figure 2, the concept of static cardinality can be applied to the fluent *hasCEO*, stating that this fluent can point to exactly one individual, since a company may have only one CEO. This restriction would thus be violated when we assign, for example, two different CEO's to the same fluent holding for the same timeslice. This is graphically depicted in Figure 8, where the left-hand side represents a non-violating example of the static cardinality applied to a fluent, and the right-hand side represents a violation of this restriction.

Thus, a straight-forward extension of the static cardinality concept might be applied directly to fluents in a temporal setting, instead of static properties, and maintain its semantics in the context of the tOWL language. However, due to the timeslices representation employed in tOWL, this use of the static cardinality

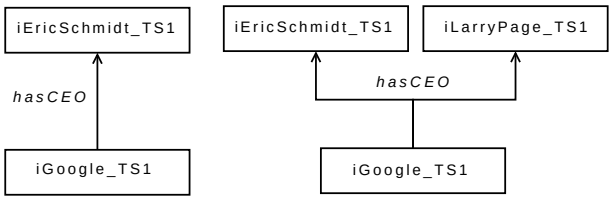

Figure 8 Static cardinality applied to fluents.

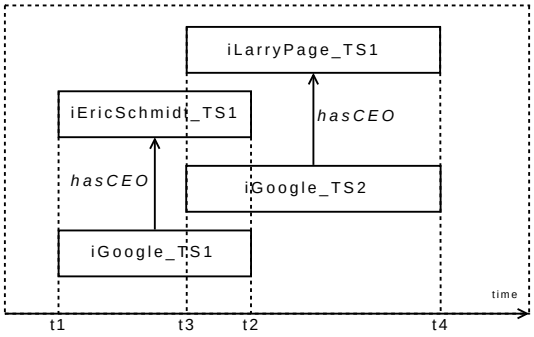

Figure 9 Violation of *hasCEO* temporal cardinality constraint.

concept proves insufficient. For illustrating why such a concept is insufficient, we consider the example already presented in Figure 5. Translating this to our current *hasCEO* example results in the illustration presented in Figure 9.

As can be seen from this figure, the fluent cardinality constraint is not violated, as, at all times, the *hasCEO* fluent is pointing to exactly one individual. However, we can clearly see that over the time interval $[t_3, t_2)$, the company Google has two CEOs, which is against the spirit of the cardinality contraint that we envision (a company may have exactly one CEO at a time). Thus, when discussing the concept of cardinality in a temporal context, we deem it necessary to make a distinction between two types of temporal cardinality.

1. *Fluent cardinality*: the (static) cardinality of the *hasCEO* fluent should be equal to one, following the description above. In other words, the *hasCEO* fluent must be associated to exactly one timeslice (of a static individual of type *Person*) each time it is defined for a timeslice of an individual of type *Company*. This issue can easily be addressed by employing the OWL-DL cardinality construct, as shown in (Welty et al., 2006).

2. *Overlapping timeslices*: the (temporal) cardinality of the *hasCEO* fluent should be equal to 1. In other words, at any point in time, the *hasCEO* relation must be described by one timeslice of a static individual of type *Person*.

Clearly, simply addressing fluent cardinality in a temporal context is insufficient for expressing truly temporal cardinality constraints in tOWL. Therefore, we seek

to extend the static cardinality constructs presented in this section in a temporal setting, and thus introduce the concepts of *temporalMinCardinality*, *temporalMaxCardinality*, and *temporalCardinality*. We define these concepts as follows.

Definition 1 (temporalMinCardinality)
Given a fluent property f, and a value a such that $a \in \mathbb{N}$, we represent by *temporalMinCardinality(f,a)* the restriction on timeslices of an arbitrary individual so that f is defined such that, at any point in time, the number of individuals that are associated to timeslices referred to by the fluents f is **at least** a.

Definition 2 (temporalMaxCardinality)
Given a fluent property f, and a value a such that $a \in \mathbb{N}$, we represent by *temporalMaxCardinality(f,a)* the restriction on timeslices of an arbitrary individual so that f is defined such that, at any point in time, the number of individuals that are associated to timeslices referred to by the fluents f is **at most** a.

Definition 3 (temporalCardinality)
Given a fluent property f, and a value a such that $a \in \mathbb{N}$, we represent by *temporalCardinality(f,a)* the restriction on timeslices of an arbitrary individual so that f is defined such that, at any point in time, *temporalMinCardinality(f,a)* and *temporalMaxCardinality(f,a)* simultaneously hold.

We next focus on giving a formal semantic representation of the three types of temporal cardinality we introduced. In achieving this, we first define a function g that, given a fluent f, a static individual i and a point in time t, returns the number of different individuals j, for which f refers from a timeslice of i to a timeslice of j, for an interval that includes t. The result of this function is a natural number, obtained by counting the unique individuals obeying the above constraints, as shown next.

$$
\begin{aligned}
g_{(f,i,t)} = |\{j \in (\neg(&\texttt{towl:TimeSlice} \sqcup \texttt{towl:Interval} \sqcup \texttt{rdfs:Literal}))^{\mathcal{I}} \mid \\
&\forall x \in \texttt{towl:TimeSlice}^{\mathcal{I}} \exists y \in \texttt{towl:TimeSlice}^{\mathcal{I}} \wedge \\
&s \in \texttt{xsd:dateTime}^{\mathcal{I}} \wedge e \in \texttt{xsd:dateTime}^{\mathcal{I}} \wedge \\
&p \in \texttt{towl:TimeInterval}^{\mathcal{I}} \wedge \\
&(x,i) \in \texttt{towl:timeSliceOf}^{\mathcal{I}} \wedge (y,j) \in \texttt{towl:timeSliceOf}^{\mathcal{I}} \wedge \\
&(x,y) \in f^{\mathcal{I}} \wedge (y,p) \in \texttt{towl:time}^{\mathcal{I}} \wedge (p,s) \in \texttt{towl:start}^{\mathcal{I}} \wedge \\
&(p,e) \in \texttt{towl:end}^{\mathcal{I}} \wedge s \leq t < e\}|
\end{aligned}
$$

This function enables the definition of the three temporal constructs we seek to introduce in the language, namely *temporalMinCardinality*, *temporalMaxCardinality*, and *temporalCardinality*. In the following, $\geq_{\mathcal{T}}$, $\leq_{\mathcal{T}}$, and $=_{\mathcal{T}}$ denote the three constructs we introduce, a is a natural number larger than 0, f denotes a fluent and t denotes a point in time. It should be noted that the definition of temporal cardinality includes (is stronger than) the definition of static cardinality.

```
Class(Company)
Class(Person)

Class(Person_TS partial TimeSlice
  restriction(timeSliceOf someValuesFrom(Person)))

Class(Company_TS partial TimeSlice
  restriction(timeSliceOf someValuesFrom(Company))
  restriction(hasCEO someValuesFrom(Person_TS))
  restriction(hasCEO temporalCardinality(1)))
```

Figure 10 Using temporal cardinality in tOWL abstract syntax.

$$
\begin{aligned}
(\geq_{\mathcal{T}} a\ f)^{\mathcal{I}} = \{x \in \texttt{towl:TimeSlice}^{\mathcal{I}} \mid \exists i \in (\neg(\texttt{towl:TimeSlice} \sqcup \\
\texttt{towl:Interval} \sqcup \texttt{rdfs:Literal}))^{\mathcal{I}}, \\
\exists p \in \texttt{towl:TimeInterval}^{\mathcal{I}}, \exists s, e \in \texttt{xsd:dateTime}^{\mathcal{I}} \wedge \\
(x, i) \in \texttt{towl:timeSliceOf}^{\mathcal{I}} \wedge (x, p) \in \texttt{towl:time}^{\mathcal{I}} \wedge \\
(p, s) \in \texttt{towl:start}^{\mathcal{I}} \wedge (p, e) \in \texttt{towl:end}^{\mathcal{I}} \wedge \\
\forall t \in \texttt{xsd:dateTime}^{\mathcal{I}}, s \leq t < e, g_{(f,i,t)} \geq a\}
\end{aligned}
$$

$$
\begin{aligned}
(\leq_{\mathcal{T}} a\ f)^{\mathcal{I}} = \{x \in \texttt{towl:TimeSlice}^{\mathcal{I}} \mid \exists i \in (\neg(\texttt{towl:TimeSlice} \sqcup \\
\texttt{towl:Interval} \sqcup \texttt{rdfs:Literal}))^{\mathcal{I}}, \\
\exists p \in \texttt{towl:TimeInterval}^{\mathcal{I}}, \exists s, e \in \texttt{xsd:dateTime}^{\mathcal{I}} \wedge \\
(x, i) \in \texttt{towl:timeSliceOf}^{\mathcal{I}} \wedge (x, p) \in \texttt{towl:time}^{\mathcal{I}} \wedge \\
(p, s) \in \texttt{towl:start}^{\mathcal{I}} \wedge (p, e) \in \texttt{towl:end}^{\mathcal{I}} \wedge \\
\forall t \in \texttt{xsd:dateTime}^{\mathcal{I}}, s \leq t < e, g_{(f,i,t)} \leq a\}
\end{aligned}
$$

$$(=_{\mathcal{T}} a\ f)^{\mathcal{I}} = (\geq_{\mathcal{T}} a\ f)^{\mathcal{I}} \cap (\leq_{\mathcal{T}} a\ f)^{\mathcal{I}}$$

Returning to the *hasCEO* example presented in this section, the newly introduced concepts can be employed for representing the fact that a company may only have one CEO at a time. For representing this example we rely on tOWL abstract syntax. Upon defining the two static OWL classes, `Company` and `Person`, we define the class of all timeslices of a person, namely `Person_TS`. The class `Company_TS` is defined as the class of all timeslices of an individual of type `Company` for which the `hasCEO` fluent takes a value that is a timeslice of `Person`. Finally, the temporal cardinality of the `hasCEO` fluent is defined to be equal to 1, i.e., one company may have exactly one CEO at a time. It should be noted that, since the formal definition of temporal cardinality includes static cardinality, there is no need to additionally define the static cardinality of the `hasCEO` fluent.

6 Conclusions

In this paper we have introduced two novel concepts in the tOWL language: temporal coalescing and temporal cardinality. The first concept, temporal coalescing, ensures a reduction of the proliferation of objects in the knowledge base, while also ensuring that a larger number of temporal queries can be resolved than previously. Temporal cardinality comes to address the limitations of the concept of static (OWL-DL) cardinality when timeslices are involved. Rather than focussing on the range of a property, the temporal cardinality introduced in tOWL involves overlapping timeslices that may violate a cardinality constraint, such as a company with two CEOs at a moment in time.

Temporal coalescence, as introduced in this paper in the context of the tOWL language, is a reasoner-based optimization aimed at reducing the number of timeslices contained in the knowledge base. Although this does not impact the representational power available to the user, we deem this optimization relevant in a practical context, where the size of the knowledge is of crucial importance in determining the speed of the applications based on the tOWL language. Also, it allows to answer correctly more temporal queries than using tOWL without this optimization.

As future work we focus on further extending the expressiveness of the tOWL language in a temporal context, as well as the optimizations that may be envisioned in such a context. A possible research direction is introducing multi-dimensional time (e.g., transaction time, user-defined time, etc.) in the language. An example of an optimization that we plan to investigate is the ability to relate multiple intervals to the same timeslice.

Acknowledgements

This work has partially been supported by the EU funded IST STREP Project FP6 - 26896: Time-determined ontology-based information system for realtime stock market analysis and by the European Science Foundation through COST Action IC0702 Combining Soft Computing Techniques and Statistical Methods to Improve Data Analysis Solutions.

References

Allen, J.F. (1983) 'Maintaining Knowledge about Temporal Intervals', *Communications of the ACM*, vol. 26, no. 11, pp. 832–843, ACM.

Artale, A., Kontchakov, R., Ryzhikov, V. and Zakharyaschev, M. (2010) 'Temporal Conceptual Modelling with DL-Lite', *23rd International Workshop on Description Logics (DL 2010)*, CEUR-WS series, vol. 573. Online: `http://ceur-ws.org/Vol-573/paper_28.pdf`.

Bechhofer, S., Van Harmelen, F., Hendler, J., Horrocks, I., McGuinness, D.L., Patel-Schneider, P.F. and Stein, L.A. (2004) 'OWL Web Ontology Language Reference', *W3C Recommendation*, 2004.

Bohlen, M.H. and Snodgrass, R.T. (1996) 'Coalescing in Temporal Databases', *Twenty-Second International Conference on Very Large Data Bases (VLDB 1996)*, pp. 180–191, Morgan Kaufmann Publishers.

Dyreson, C.E. (2003) 'Temporal Coalescing with Now Granularity, and Incomplete Information', *ACM SIGMOD International Conference on Management of Data (SIGMOD 2003)*, pp. 169–180, ACM.

Frasincar, F., Milea, V. and Kaymak, U. (2010) 'tOWL: Integrating Time in OWL', *Semantic Web Information Management*, ch. 11, pp. 225–246, Springer.

Fokoue, A., Kershenbaum, A., Ma, L., Schonberg, E. and Srinivas, K. (2006) 'The Summary ABox: Cutting Ontologies Down to Size', *5th International Semantic Web Conference (ISWC 2006)*, pp. 343–356, Springer.

Gutierrez, C. and Hurtado, C. and Vaisman, A. (2007) 'Introducing Time into RDF', *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 2, pp. 207–218, IEEE.

Grandi, F. (2004) 'Multi-temporal RDF Ontology Versioning', *International Workshop on Ontology Dynamics (IWOD 2009)*. Online: `www-db.deis.unibo.it/~fgrandi/papers/IWOD09.pdf`.

Gregersen, H. and Jensen, C.S. (2002) 'Temporal Entity-Relationship Models - A Survey', *IEEE Transactions on Knowledge and Data Engineering*, vol. 11, no. 3, pp. 464–497.

Haase, P. and Stojanovic, L. (2005) 'Consistent Evolution of OWL Ontologies', *2nd European Semantic Web Conference (ESWC 2005)*, pp. 182–197, Springer.

Hobbs, J.R. and Pan, F. (2004) 'An Ontology of Time for the Semantic Web', *ACM Transactions on Asian Language Information Processing*, vol. 3, no. 1, pp. 66–85, ACM.

Jensen, C.S., Clifford, J., Elmasri, R., Gadia, S.K., Hayes, P.J. and Jajodia, S. (1994) 'A Consensus Glossary of Temporal Database Concepts', *SIGMOD Record*, vol. 23, no. 1, pp. 52–64, ACM.

Milea, V., Frasincar, F., Kaymak, U. and di Noia, T. (2007) 'An OWL-Based Approach Towards Representing Time in Web Information Systems', *4th International Workshop of Web Information Systems Modeling (WISM 2007)*, pp. 791–802, Tapir Academic Press.

Milea, V.,Frasincar, F. and Kaymak, U. (2008) 'Knowledge Engineering in a Temporal Semantic Web Context', *International Conference on Web Engineering (ICWE 2008)*, pp. 65–74, IEEE Computer Society Press.

Milea, V., Mrissa, M., Sluijs, K. and Kaymak, U. (2008) 'On Temporal Cardinality in the Context of the TOWL Language', *Fifth International Workshop on Web Information Systems Modeling (WISM 2008)*, pp. 457–466, Springer-Verlag.

Noy, N.F. and Klein, M. (2004) 'Ontology Evolution: Not the Same as Schema Evolution', *Knowledge and Information Systems*, vol. 6, no. 4, pp. 428–440, Springer.

Patel-Schneider, P.F. and Hayes and P., Horrocks, I. (2004) 'Web Ontology Language (OWL) Abstract Syntax and Semantics', *W3C Recommendation*, W3C.

Spaccapietra, S., Parent, C. and Zimanyi, E. (1998) 'Modeling Time from a Conceptual Perspective', *Seventh International Conference on Information and Knowledge Management (CIKM 1998)*, pp. 432–440, ACM.

Steiner, A. and Norrie, M.C. (1997) 'Implementing Temporal Databases in Object-Oriented Systems', *Fifth International Conference on Database Systems for Advanced Applications (DASFAA 1997)*, vol.6, pp. 381–390, World Scientific Press.

Tauzovich, B. (1991) 'Toward Temporal Extensions to the Entity-Relationship Model', *10th International Conference on the Entity Relationship Approach (ER 1991)*, pp. 163–179, ER Institute.

Wang, F. and Zaniolo, C. (2004) 'XBiT: An XML-Based Bitemporal Data Model', *23rd International Conference on Conceptual Modeling (ER 2004)*, pp. 810–824, Springer.

Welty, C., Fikes, R. and Makarios, S. (2006) 'A Reusable Ontology for Fluents in OWL', *Fourth International Conference on Formal Ontology in Information Systems (FOIS 2006)*, pp. 226–336, IOS Press.

Widagdo, T.E. (2007) 'Managing Referential Integrity in Bitemporal Databases', *International Conference on Electrical Engineering and Informatics (ICEEI 2007)*, pp. 305–308.

Wijsen, J. (1999) 'Temporal FDs on Complex Objects', *ACM Transactions on Database Systems*, vol. 24, no. 1, pp. 127–176, ACM.

Zimanyi, E., Parent, C., Spaccapietra, S. and Pirotte, A. (1997) 'TERC+: A Temporal Conceptual Model', *International Symposium on Digital Media Information Base (DMIB 1997)*. Online: `cs.ulb.ac.be/publications/P-97-12.pdf`.