A Temporal Web Ontology Language

Viorel Milea, Flavius Frasincar, and Uzay Kaymak

ERIM Report Series *Research in Management*	
ERIM Report Series reference number	ERS-2009-050-LIS
Publication	September 2009
Number of pages	40
Persistent paper URL	http://hdl.handle.net/1765/16794
Email address corresponding author	kaymak@ese.eur.nl
Address	Erasmus Research Institute of Management (ERIM) RSM Erasmus University / Erasmus School of Economics Erasmus Universiteit Rotterdam P.O.Box 1738 3000 DR Rotterdam, The Netherlands Phone: + 31 10 408 1182 Fax: + 31 10 408 9640 Email: info@erim.eur.nl Internet: www.erim.eur.nl

ABSTRACT AND KEYWORDS	
Abstract	The Web Ontology Language (OWL) is the most expressive standard language for modeling ontologies on the Semantic Web. In this paper, we present a temporal extension of the very expressive fragment SHIN(D) of the OWL-DL language resulting in the tOWL language. Through a layered approach we introduce 3 extensions: i) Concrete Domains, that allows the representation of restrictions using concrete domain binary predicates, ii) Temporal Representation, that introduces timepoints, relations between timepoints, intervals, and Allen's 13 interval relations into the language, and iii) TimeSlices/Fluents, that implements a perdurantist view on individuals and allows for the representation of complex temporal aspects, such as process state transitions. We illustrate the expressiveness of the newly introduced language by providing a TBox representation of Leveraged Buy Out (LBO) processes in financial applications and an ABox representation of one specific LBO.
Free Keywords	Web, time, OWL, fluents, concrete domains
Availability	The ERIM Report Series is distributed through the following platforms: Academic Repository at Erasmus University (DEAR), DEAR ERIM Series Portal Social Science Research Network (SSRN), SSRN ERIM Series Webpage Research Papers in Economics (REPEC), REPEC ERIM Series Webpage
Classifications	The electronic versions of the papers in the ERIM report Series contain bibliographic metadata by the following classification systems: Library of Congress Classification, (LCC) LCC Webpage Journal of Economic Literature, (JEL), JEL Webpage ACM Computing Classification System CCS Webpage Inspec Classification scheme (ICS), ICS Webpage

A Temporal Web Ontology Language

Viorel Milea Flavius Frasincar Uzay Kaymak

Econometric Institute, Erasmus School of Economics

P.O. Box 1738, 3000 DR Rotterdam, the Netherlands

Email: {milea, frasincar, kaymak}@ese.eur.nl

Abstract

The Web Ontology Language (OWL) is the most expressive standard language for modeling ontologies on the Semantic Web. In this paper, we present a temporal extension of the very expressive fragment $\mathcal{SHIN}(\mathcal{D})$ of the OWL-DL language resulting in the tOWL language. Through a layered approach we introduce 3 extensions: i) *Concrete Domains*, that allows the representation of restrictions using concrete domain binary predicates, ii) *Temporal Representation*, that introduces timepoints, relations between timepoints, intervals, and Allen's 13 interval relations into the language, and iii) *TimeSlices/Fluents*, that implements a perdurantist view on individuals and allows for the representation of complex temporal aspects, such as process state transitions. We illustrate the expressiveness of the newly introduced language by providing a TBox representation of Leveraged Buy Out (LBO) processes in financial applications and an ABox representation of one specific LBO.

Keywords

Semantic Web, time, OWL, fluents, concrete domains

1 Introduction

The considerable and ever-increasing need to access the large volume of data present on the World Wide Web today motivates a migration from free-text representations of data to semantically rich representations of information. Endeavors in this direction are being undertaken under a common denominator: the Semantic Web [11]. The state-of-the-art tools and languages provided under this umbrella, such as RDF(S) [12,22] and OWL [35], go beyond the standard Web technology and provide the means for data sharing and reusing outside this platform, i.e., in the form of semantic applications.

Focussed around the inference of implicit knowledge from explicitly represented information, Semantic Web approaches are currently centered around static abstractions of the world. However, conceptualizations lacking a temporal dimension are not only rather artificial, but also impractical in environments that require context-awareness. Examples of such environments can easily be found after a quick browse through highly dynamic domains, such as the financial one. In such a case, one can envision the need for representing ephemeral knowledge, for example contained in news messages (stock price and other financial variables), or more fundamental aspects of the financial domain (mergers & acquisitions and financial processes).

Addressing temporality in abstract representations of the world requires dealing with the aspect of time. One aspect is that of reference system - bringing an order into sequences of events. In this respect, time can be instant-based or interval-based, with instants denoting basic points in time with no duration, and intervals being represented as pairs of distinct instants denoting some period of time.

A second aspect of time as pursued in this paper regards temporal concepts such as, for example, the ephemeral character of relationships between individuals. In this context, we seek to enable representations of change - descriptions of individuals that take variable values for some property at different points in time - and state transitions, enabling the representation of processes and corresponding transition axioms. In this sense, time is somewhat implicit to the representation, i.e., the conceptualization evolves relative to the temporal reference system and requires the latter.

The main goal pursued in this contribution regards an extension of a fragment of OWL-DL with time. The fragment of OWL-DL considered is $\mathcal{SHIN}(\mathcal{D})$, which represents OWL-DL without the use of nominals. In the remainder of this article, we shall denote the fragment of OWL-DL based on the $\mathcal{SHIN}(\mathcal{D})$ description logic as OWL-DL^{-}. We focus on this particular subset due to the fact that this is the most expressive fragment of

OWL-DL extended with concrete domains for which a terminating, sound and complete reasoning algorithm is known [23].

In temporal terms, the extension of OWL-DL$^-$ that we envision addresses time both in the sense of reference system as well as covering more complex temporal aspects, such as change and state transitions. This materializes in a syntactic and semantic extension of OWL-DL$^-$ in the form of a temporal web ontology language (tOWL) [27–30]. The tOWL language is an extension of OWL-DL$^-$ that enables the representation and reasoning with time and temporal aspects. It comes to meet shortcomings of previous approaches, such as [17, 39] that only address this issue to a limited extent.

The approach presented in [17], for example, only deals with the representation of time in the form of intervals and instants. However, ensuring that intervals are properly defined (starting point is always strictly smaller than the ending point) is not possible in this approach. Additionally, no support is offered for reasoning on the temporal constructs introduced other than the standard OWL-DL reasoning. In the OWL-Time it is also not possible to enforce a particular order of state transitions in a process.

The approach taken in [39] builds upon [17] by addressing one of its limitations, namely: the representation of temporal aspects such as change. One of the limitations of the approach in [39] relates to the definition of fluent properties as being symmetric - if the pair (x,y) is the interpretation of a symmetric property, than the pair (y,x) is also an instance of this property. This is more often than not false, as in the very simple example of the *employeeOf* relation: although it holds that x is an employee of y, it certainly is not the case that y is also en employee of x. Of course, restricting the definition of fluent properties could support the definition hereof as symmetric, such as saying that there is an *employee* relation between two concepts. However, for the current goal, we deem this to be insufficient and do not define fluent properties as satisfying symmetry.

Building upon the approach in [39], tOWL enables differentiations between fluents that take values from the *TimeSlice* class and fluents that indicate changing values (datatypes). This is achieved through the use of the *FluentObjectProperty* and *FluentDatatypeProperty* properties, and comes to reduce the proliferation of objects in tOWL ontologies due to the fact that, in the case of datatypes, the number of timeslices that need to be created is reduced to half. For the representation of time the tOWL language relies on an approach based on concrete domains, thus enabling higher temporal expressiveness when compared to the approach in [17], that also stands at the basis of the fluents approach in [39].

The outline of the paper is as follows. In Section 2 we provide an overview of work related to the current endeavor. Section 3 introduces different layers of the tOWL language built on top of OWL-DL$^-$. The RDF/XML serialization of the language is provided in Section 4. An extensive example of how the expressiveness of tOWL can be employed for the representation of Leverage Buy Outs in financial applications is provided in Section 5. Our conclusions and possible future works are given in Section 6.

2 Related Work

World representations may be synchronic or diachronic in the way the temporal perspective is considered within the representation [1]. Synchronic representations consider a single point in time, with no regard for temporal evolution. Diachronic representations take into consideration the existence of a history, and thus take into account change through time. Regardless of the form of representation chosen, one must invariably deal with the problem of identity. Synchronic identity regards identity holding at one single time. Diachronic representations, our current focus, must deal with the problem of diachronic identity, or put differently, establishing how change affects the identity of entities existing at different times.

In considering the issue of identity, two principles formulated by Leibniz provide a different perspective hereon. The first one, regarding the *identity of indiscernibles*, states that entities for which all properties are common and identical are, in turn, identical. Additionally, *indiscernibility of identicals* states that entities being identical implies that the entities have all properties in common, and the values hereof are identical.

Addressing the problem of diachronic identity involves adhering to one of two possible views of the world, 3D vs. 4D. In a 3D view, an explicit distinction is made between *endurants* and *occurants*, or put differently, between physical entities maintaining identity at all times, and events, i.e., things that happen, occur. Following Aristotle's view, endurants are characterized by some essential properties defining their identity, and accidental ones that do not impact the latter. Such a representation comes to conflict with the second principle formulated by Leibniz, *indiscernibility of identicals*, as identity also involves *all* properties to be shared by entities which are identical.

This problem is circumvented in the 4D view by assuming a four-dimensional space in which entities are identified, and in which they exist in the temporal dimension in the same way entities, for as far material, occupy space. This resumes to regarding entities as having different temporal parts that describe the respective entity at different intervals in time. These temporal parts may be assigned different properties, thus ensuring that the *indiscernibility of identicals* principle holds.

Commonly, choices as the ones presented until now concern the field of logics. Due to our current Semantic Web focus, special attention is given to Description Logics (DL) - a fragment of first-order logic consisting of knowledge representation languages with known and desirable computational properties (terminating, sound and complete reasoning algorithm). The issue of temporality has also been addressed in this context, presenting several choices regarding the handling of a temporal dimension.

The main distinction separating these approaches can be made in terms of whether the temporal language offers explicit time, or the temporal dimension is only implicitly present in the language by providing the means to talk about order of events and/or states. Following [6], these different approaches are categorized in *explicit* and *implicit*, respectively.

In the case of formalisms adhering to an *explicit* view on the temporal dimension, a further differentiation may be made between *internal* and *external* methods of incorporating a temporal dimension in the presence of temporal operators. To some extent, the *internal* view is in accordance with the *indiscernibility of identicals* principle of Leibniz, in that individuals are regarded as a set of temporal parts of these individuals. The *external* view on the other hand makes a clear distinction between static and temporal entities. In this latter view, individuals may have different temporal snapshots at different moments in time, describing that specific individual at the respective moment in time. In this approach, the temporal dimension plays the role of relating the different snapshots. Such a world view shares many of the shortcomings inherent to an endurantist view of the world.

Next to *explicit* incorporations of the temporal dimension, [6] discusses approaches adhering to an *implicit* view on time. In such approaches, the focus is mainly on describing worlds in a state-like manner. From this perspective, the temporal dimension is only implicitly present in the language by enabling the description of orders of events/states. In this approach, no a direct reference is made to the time associated with the event/states. The main drawback of such approaches regards the inability of determining what holds true at a certain point in time, i.e., constructing partial worlds from what holds true across or during a given period.

Both the explicit and the implicit view of the world have different implementations in the form of formalisms and/or systems. For the current purpose, we focus on the explicit approach as we deem implicit temporal representations inappropriate for our current purpose, for two main reasons: i) in representations with an implicit temporal dimension, one cannot say exactly what holds true at one specific time, and ii) the representational power of representations based on implicit temporal representations can be captured by formalisms relying on an explicit time dimension. Of particular interest are the explicit approaches relying on description logics, as this type of logics are the underlying formalism of the OWL-DL$^-$ language, our current focus.

The interval-based $\mathcal{TL}$-$\mathcal{ALCF}$ description logic [4, 5] enables the representation of temporal interval networks, through Allen's interval temporal logic, in the context of the static $\mathcal{ALCF}$ description logic. The resulting logic is an aggregation of a temporal and static logic, thus making this approach external as the temporal dimension is external to the $\mathcal{ALCF}$ description logic. The $\mathcal{TL}$-$\mathcal{ALCF}$ is decidable, and is currently the most expressive DL extended in such a way that a terminating, sound and complete algorithm is known. Returning to our current focus, OWL-DL$^-$ and the very expressive underlying description logic $\mathcal{SHIN}(\mathcal{D})$, it can be concluded that an approach not moving beyond the expressiveness of $\mathcal{ALCF}$ is insufficient for the current goal.

Approaches similar to $\mathcal{TL}$-$\mathcal{ALCF}$, but relying on a point-based temporal structure rather than an interval-based one, provide means to represent temporal dependencies between entities. An example of one such logic consists of the $\mathcal{DLR}$ description logic extended with the temporal operators *Until* and *Since*, resulting in the $\mathcal{DLR}_{\mathcal{US}}$ temporal description logics [7]. Similarly, the $\mathcal{ALCT}$ temporal description logic is an extension of $\mathcal{ALC}$ with the temporal connectives of tense logic, such as existential and universal future

[6,37]. These approaches present drawbacks for our current purpose, in that extending the point-based approach with an interval based one, as well as adding to the expressiveness of the static DL that is extended, both severely impact the complexity of the logic, and thus decidability.

Time can also be incorporated in the formalism by making it part of the latter, in what constitutes an internal approach. One such approach consists of extending a DL formalism with concrete domains. Initially proposed in [8], concrete domains allow abstract concepts to be related to concrete values through functional roles. Description logics extended with concrete domains maintain decidability, provided that the concrete domain satisfies the property of admissibility or ω-admissibility [8, 25]. For a number of constraint systems, special types of concrete domains based on binary domain predicates that are jointly-exhaustive and pairwise disjoint, ω-admissibility has been proved in [25], such as for example a constraint system based on a domain consisting of intervals, and Allen's 13 interval relations that may hold between pairs of intervals. This constraint system approach to introducing time in DL-based formalisms is, for the current context, the one that is less restricted by the expressiveness of the static DL which it extends. Results are known for $\mathcal{SHIQ}(\mathcal{C}+)$ description logic for which a terminating, sound and complete reasoning algorithm is known [23].

Different aspects regarding the representation and management of time-varying data have also been addressed within the broad area of temporal databases. A common way of regarding time in such a context relates to the type of time that is addressed by the system. This has resulted in 3 types of time [19] that may be considered, individually or combined, in a temporal database: i) valid time, the time when a fact is true in the real world, ii) transaction time, the time when the fact is known in the database, and iii) user-defined time, which can represent any temporal attribute for which the temporal semantics is only known to the user and has no particular meaning in the database. When valid and transaction time are considered together, this results in bitemporal datamodels [19]. Regarding the structure of the time domain, a further distinction may be made between linear time - one timeflow from past, through present, to future - and branching time, where the representation of possible, alternative futures is allowed [33]. Different temporal extensions of standard database models are known [33], from which a vast majority is focussed on the relational database model, see for example [3, 10, 13, 14, 20], but considerable research has been performed in extending object-oriented databases with a temporal dimension, see for example [9, 36, 38, 40]. Although some of the concepts and problems encountered in temporal databases have a certain overlap with the issues of temporal extensions of Semantic Web ontology languages, some crucial differences apply when ontology change is considered in the light of database schema evolution, as discussed in [31].

In the context of the Semantic Web, a number of approaches have already been designed, addressing different temporal aspects in relation to ontology languages. In this final part of the overview regarding work related to our current aim, we discuss temporal RDF [15], OWL-Time [17], OWL-MeT [21] and an OWL ontology for fluents [39].

A rather extensive approach towards extending ontology languages with a temporal dimension is reported in [15]. This work is similar to our current goal as it concerns the ability to represent temporal information in ontologies, but differs in that the language considered is the Resource Description Framework (RDF). The result of this approach consists of temporal RDF graphs with underlying temporal semantics allowing temporal entailment on these graphs. The extension presented in [15] with regard to RDF consists of a vocabulary extension focused on temporal labeling. This vocabulary extension enables the labeling of triples, thus allowing for the representation of the time period during which the triple was true. These labels are represented as intervals (ordered pairs of instants - point-like moments in time). This allows the introduction of graph slices at a certain time t, a subgraph consisting of all temporal triples with temporal label t, and graph snapshots at a certain time t, all triples holding true at or during t. In other words, in temporal RDF, triples are considered in association with their valid time.

The OWL-Time approach focusses on the Web Ontology Language rather than RDF. The initial purpose behind the design of a time ontology was to represent the temporal content of Web pages and the temporal properties of Web Services (DAML-Time) [17]. The time ontology is built around the *TemporalEntity* class and the relations describing its individuals. Temporal entities may be of two types: *Instant* (point-like moments in time) and *Interval* (time descriptions having a duration, represented as ordered pairs of *Instant* individuals). Additionally, the ontology contains classes meant to describe other temporal concepts, such as duration (*DurationDescription*), dates and times (*DateTimeDescription*), temporal units (*TemporalUnit*), and alternative representations for the days of the week (*DayOfWeek*).

The main relations describing individuals of type *Instant* are the functional properties *begins* and *ends*. The range of these properties are objects of type *Instant*. The actual time belonging to these individuals can be expressed as of the internal type *CalendarClockDescription* or as *dateTime* of XML Schema. Intervals are also present in this time ontology, including the different properties that may describe them. The property *inside* may be defined in the case of individuals of type *Interval*, describing a relation between an *Instant* and an *Interval*, equivalent to asserting that some individual of type *Instant* is within the bounds of the individual of type *Interval*.

Despite being rather extensive in describing quantitative time and the qualitative relations that may exist among instants and intervals, the OWL-Time ontology is represented in OWL-DL, and thus employs the underlying $\mathcal{SHOIN}(\mathcal{D})$ description logic, which has a limited expressivity for the purpose pursued in this paper. This DL only provides limited support for datatypes, and is far from employing a fully fledged concrete domain or constraint system as described in [25]. For this reason, the actual meaning of, for example, an interval, is limited to the OWL-DL semantics. Representing proper intervals, i.e., intervals for which the starting point is strictly smaller than the ending point, as shown in Equation 1, is not possible.

$$\texttt{ProperInterval} \equiv \exists(\texttt{begin, end}).< \tag{1}$$

Axioms describing all possible relations between intervals are present in OWL-Time. More complex constructs can be defined from the foundation offered by OWL-Time, from which the most important category is represented by temporal aggregates. An example of a temporal aggregate expressible in OWL-Time is: "Every other week on Monday, Wednesday and Friday, until December 24th 1997, but starting on Tuesday, September 2nd, 1997" [34].

One final remark should be made regarding the expressiveness of the OWL-Time approach, and that relates to support for time zones. This relies on an external time zone ontology where the characteristics of time zones are explicitly modeled. Information regarding time zones in the US as well as across the whole world is present in this time zone ontology and is able to handle all the usual time zones and daylight saving cases.

The problem of ontology evolution has been considered in an extension of OWL with metric time, resulting in the OWL-MeT language [21]. The language extension concerns the addition of the constructs of the $\mathcal{ALCIO}(\mathcal{MT})$, such as *somefuture* and *allfuture*, for example. The main focus of this research is however different from the goal we pursue in this paper. OWL-MeT is solely concerned with ontology evolution, and thus the main focus is on dealing with multiple ontology versions. This resumes to issues of compatibility and querying over multiple versions of an ontology.

An approach related to incorporating perdurants through the use of timeslices and fluents in OWL-DL is presented in [39], where the authors develop a reusable ontology for fluents in OWL-DL. The fundamental building blocks of this representation are time slices and fluents. Time slices represent the temporal parts of a specific entity at clear moments in time and the concept itself is then defined as all of its timeslices. Fluents are nothing more than properties that hold at a specific moment in time, may this time be an instant (point-like representation of time) or an interval. This approach is an OWL implementation of a perdurantist, i.e. four-dimensional, view of the world, that does not stand in violation with the Leibniz Law. One of the drawbacks of this approach consists of a proliferation of objects in the ontology, due to the creation of 2 timeslices each time something is changing, that in turn must be associated to the static individual they represent and linked to each other by a fluent. Additionally, no solution is provided for the temporal equivalent of the cardinality construct, which cannot be modeled in the case of overlapping timeslices, as also argued in [39].

3 The Temporal Web Ontology Language

Devising a temporal extension of OWL-DL^{-} begins with a clarification of what is understood under the general, common denominator *time*. For the current purpose, we consider a couple of fundamental aspects hereof, namely: i) temporal infrastructure, and ii) change.

The first aspect, *temporal infrastructure*, regards the representation of time in the form of instants and/or intervals. From this perspective, we consider desirable an approach that incorporates both a point-based as well as an interval-based time representation. Such an approach should provide not only the temporal entities that constitute the temporal infrastructure of the language, but also the relations that may hold between these entities, e.g., the *before* relation that may hold between intervals.

The second aspect of time considered for the current purpose, *change*, requires further clarification. This clarification is required, on one side, by the fact that there are two types of changes in OWL ontologies: changes at the terminological level (TBox), and changes at the assertional level (ABox). For the tOWL language, the focus is solely on changes that concern individuals; in other words, tOWL enables the representation of change at ABox level. Having isolated the focus of change, we also pinpoint its meaning to relate to the change of any attribute value that describes an individual. For the current purpose, we deem such changes to be one of three considered types: i) change in a concrete attribute value of an individual, such as a change of hair color, ii) a change in the relationship between entities, such as a product that belongs/is produced by a company, and iii) state transitions in processes, such as the transition from the liquid state to a bankruptcy state in the case of companies. In this context, we refer only to valid time, as known from temporal databases, rather than transaction time, i.e., we seek to represent when certain changes take place in the actual world rather than the time when they are represented in the ontology.

The vast amount of research in temporal logic, temporal databases and even the Semantic Web, as discussed in the previous section, comes to motivate reusing existing results. A discussion on the choices made when considering the language design is provided in Section 3.1. The individual tOWL layers are presented, one by one, in Sections 3.3 through 3.5. The conceptual presentation of the language is concluded with a discussion on reasoning in the tOWL language, in Section 3.7.

3.1 Design Choices

Designing the tOWL language concerns a number of choices regarding the most suitable approach(es) for the representation of the two temporal aspects considered for the current scope: temporal infrastructure and change.

At the level of temporal infrastructure, we seek to enable point-based as well as interval-based representations. Additionally, we seek to extend the expressiveness of OWL-DL$^-$ and the underlying $\mathcal{SHIN}(\mathcal{D})$ description logic without constraining the latter. From the approaches investigated in Section 2, the only approach suitable for these goals is one based on concrete domains. The temporal infrastructure thus becomes internal to the language, and covers both the point-based time and the interval-based time. For a point-based representation of time, we rely on a concrete domain based on the set $\mathbb{Q}$ of rational numbers and the set of binary concrete domain predicates $\{<, \leq, =, \neq, \geq, >\}$. Results are known for such an extension to the description logic $\mathcal{SHIQ}$, where the concrete domain is also extended with an additional unary predicate $=_q$ for denoting equality with $q \in \mathbb{Q}$, resulting in the $\mathcal{SHIQ}(\mathcal{C}^+)$, for which concept satisfiability and subsumption with regard to general TBoxes are both ExpTime-complete [23, 24]. Introducing such a concrete domain in the language has the advantage of not only enabling the representation of dates and times in terms of a translation between the *xsd:dateTime* XML datatype and rational numbers, but enables the description of any numerical attribute through a direct reference to the concrete domain.

Still considering the temporal infrastructure of the language, we seek to enable an interval-based representation of time satisfying the previously mentioned constraints. For this purpose, we seek to add intervals and Allen's 13 interval relations [2] to the tOWL language. As known from [2], all 13 Allen's interval relations may be translated in terms of equivalent relations on the intervals' endpoints. For this reason, the concrete domain based on the set $\mathbb{Q}$ of rational numbers and the set of binary concrete domain predicates $\{<, \leq, =, \neq, \geq, >\}$ is sufficient for such representations. Thus, intervals and Allen's 13 interval relations are not introduced in the language by means of a concrete domain, but rather as syntactic sugaring over the concrete domain $\mathbb{Q}$ with the respective relations. By only introducing one concrete domain into the language, we build upon known decidability results for description logics extended with concrete domains and ensure the language decidability.

The representation of change in a temporal ontology language poses several problems that need to be addressed. For the current goal we consider diachronic representations that take history into account rather than synchronic ones, and are thus faced with the problem of diachronic identity, as discussed in Section 2. The second principle of Leibniz, indiscernibility of identicals, poses an additional restriction on the choice of representation and perspective on identity when change is involved. Finally, as the temporal language we develop is aimed at the Semantic Web, one must invariably be able to say what holds true at a certain moment in time. The Semantic Web, and OWL-DL^{-} in this context, further restrict the flexibility of designing an approach for the representation of change due to the restriction of the underlying $\mathcal{SHIN}(\mathcal{D})$ description logic.

The straight-forward approach of associating a valid time to the binary predicate, similar to solutions from temporal databases and temporal RDF, is not suited in the current case, as ternary predicates are not directly supported in OWL-DL. The W3C Semantic Web Best Practices working group provides 3 alternative ways of representing n-ary relationships on the Semantic Web [32], namely: i) representing a relationship as a class rather than as a property, ii) representing the individuals participating in the relation in the form of a collection or ordered list, and iii) RDF reification. The first two approaches share the drawbacks of proliferation of objects and the reduced meaning of the actual representation of instances, especially in the case of OWL-DL. Regarding the third, it should be noted that RDF reification is not appropriate when "the intent is to talk about instances of a relation, not about statements about such instances." [32]. Besides the fact that the RDF "reification of a triple does not entail the triple, and is not entailed by it" [16], reification is not supported at all in OWL-DL^{-}, the language that we are extending, thus making such an approach unsuitable.

Another approach for associating valid time with a binary relation relates to the addition of a meta-logical [26] predicate that takes as arguments the binary relationship and the time when this relationship holds, respectively. However, as also discussed in [39], such predicates are not supported in any of the OWL species.

The fluents approach presented in [39] and discussed in Section 2 is consistent with the second principle of Leibniz and enables the maintenance of identity through change by introducing a 4D view of the world in OWL ontologies. By moving the temporal argument to the level of timeslices rather than the fluent itself, it circumvents the issue of n-ary relationships, while still enabling the determination of what holds true at a particular time. This approach also has the advantage of not restricting the expressiveness of the

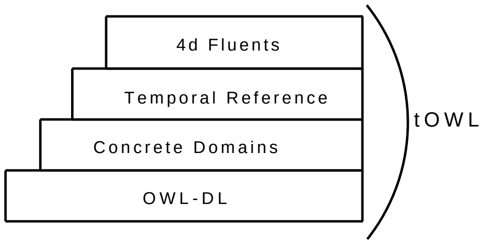

Figure 1: The tOWL Layer Cake

description logic it extends, as it is more concerned with syntactic sugaring rather than being a semantic extension. As introduced in [39], this 4D approach is achieved in the form of an OWL ontology, which although insufficient for our current goal, extending the OWL-DL$^-$ language, should prove a good starting point in addressing the representation of change in the tOWL language.

For the design of the language we chose a layered approach. On top of the foundational OWL-DL$^-$ layer, we add a concrete domains layer, a temporal reference layer and a 4d fluents layer, as described in the following sections.

3.2 Layered Approach

The language extension concerns a number of different components, from which a couple have already been identified in the form of the concrete domains extension and the fluents extension. Additionally, we introduce an extension of the general expressiveness of the language that is necessary both for concrete domains and fluents, but provides for more expressiveness even outside a temporal context. This extension concerns mainly the representation of functional role chains in the language. Thus, from an abstract perspective, OWL-DL is extended with 3 main components.

The division of the extension into different, though interdependent components, motivates a layered approach for the design of the temporal language. The tOWL layercake presented in Figure 1 provides an overview of the different layers introduced by tOWL on top of the basic OWL-DL$^-$ layer - the language we extend.

The first extension introduced by tOWL concerns the expressiveness of the language in a broader sense, rather than being restricted to a temporal domain. The *Concrete Domains (CD) Layer* enables the representation of restrictions based on role compositions and binary concrete domain predicates.

Partly enabled by the CD Layer, the *Temporal Reference (TR) Layer* adds a temporal reference system to the language, in the form of concrete time (instants and intervals) and concrete temporal relations. The TR layer employs the CD layer for the representation of temporal restrictions between time-bounded entities, such as statements regarding the relative temporal incidence of events.

Upon enabling temporal reference in the language, the representation of change and state transitions is provided through the *4d Fluents (4dF) Layer*. This extension enables the representation of temporal parts of individuals that may differ from one another in different aspects (attribute values of properties) at various moments in time. The strong relation between the CD and TR layers and the 4dF layer initiates in the latter being enabled by the previous layers, and moving temporality beyond a simple reference system.

In the following three sections, we provide a more detailed presentation of each of the three layers.

3.3 Concrete Domains Layer

The representation of complex restrictions, regardless of whether they describe some temporal aspect, or rather relate to some static expression, can be achieved through the composition of roles. In what follows, we denote by *feature chain* a composition of features (functional roles). Following common denomination from Description Logics and the Semantic Web, we make a distinction between abstract features, that point to something in the abstract domain, and concrete features, that take values from the concrete domain.

Additionally, in tOWL we allow the feature chains to be composed with one concrete feature g, giving birth to what is commonly denoted as a *concrete feature path* (CFP), and which is mathematically equivalent to the following composition:

$$f_1 \circ f_2 \circ ... \circ f_n \circ g,$$

where $n \in \mathbb{N}$. Note that for $n = 0$, by convention, the set of abstract features is empty.

Letting u_i denote a CFP, we allow existential and universal quantification of the following form in tOWL, where p_d denotes a binary concrete domain predicate:

$$\exists(u_1, u_2).p_d$$

$$\forall(u_1, u_2).p_d$$

For such constructs, u_i may arbitrarily denote a CFP of length m, with $m \in \mathbb{N}^*$ (note that a path of length 1 consists solely of a concrete feature g).

We summarize the semantics introduced by this layer in Figure 2, with reference to the tOWL abstract syntax constructs we introduce.

tOWL Abstract syntax	**Model-Theoretic Semantics**
`ConcreteFeatureChain`$(f_1\ f_2\ \dots\ f_n\ g)$	$\{a_1 \in \Delta^\mathcal{I} \mid \exists! a_2 \in \Delta^\mathcal{I}, \dots, \exists! a_{n+1} \in \Delta^\mathcal{I} \wedge$ $\wedge\ \exists! b \in \Delta_\mathcal{D} : (a_1, a_2) \in f_1^\mathcal{I}, \dots$ $(a_n, a_{n+1}) \in f_n^\mathcal{I} \wedge g^\mathcal{I}(a_{n+1}) = b\}$.
`dataSomeValuesFrom` $(u_1\ u_2\ p_d)$	$\{x \in \Delta^\mathcal{I} \mid \exists! q_1 \in \Delta_\mathcal{D}, \exists! q_2 \in \Delta_\mathcal{D} :$ $u_1^\mathcal{I}(x) = q_1 \wedge u_2^\mathcal{I}(x) = q_2 \wedge\ (q_1, q_2) \in p_d^\mathcal{I}\}$.
`dataAllValuesFrom` $(u_1\ u_2\ p_d)$	$\{x \in \Delta^\mathcal{I} \mid \forall q_1 \in \Delta_\mathcal{D}, \forall q_2 \in \Delta_\mathcal{D} :$ $u_1^\mathcal{I}(x) = q_1 \wedge u_2^\mathcal{I}(x) = q_2 \wedge\ (q_1, q_2) \in p_d^\mathcal{I})\}$.

Figure 2: Semantics for the *concrete domains* layer.

3.4 Temporal Reference Layer

The concrete domain in the tOWL context, as presented in the previous section, enables the representation of new restrictions in the language. Under the *Temporal Reference* layer we include basic representations of time, both point-based and interval-based, as well as a number of temporal relations between instants and intervals, as discussed in Section 3.1. This forms the basis for our approach, as it allows the definition of complex restrictions, such as the ones described in the previous section, but this time presenting a temporal character. The concrete domain employed for the current purpose is a concrete domain based on the set $\mathbb{Q}$ of rational numbers and the set of binary concrete domain predicates $\{<, \leq, =, \neq, \geq, >\}$,

This concrete domain also enables the representation of intervals and Allen's 13 interval relations through a translation scheme between interval relations and equivalent relations in terms of the intervals' endpoints. Rather than being a concrete domain, this extension

Allen Relation	**Translation**
$\exists(i_1, i_2)$.`equal`	$\exists(i_1 \circ start, i_2 \circ start).= \ \sqcap \ \exists(i_1 \circ end, i_2 \circ end).=$
$\exists(i_1, i_2)$.`before`	$\exists(i_1 \circ end, i_2 \circ start).<$
$\exists(i_1, i_2)$.`after`	$\exists(i_2 \circ end, i_1 \circ start).<$
$\exists(i_1, i_2)$.`meets`	$\exists(i_1 \circ end, i_2 \circ start).=$
$\exists(i_1, i_2)$.`met-by`	$\exists(i_1 \circ start, i_2 \circ end).=$
$\exists(i_1, i_2)$.`overlaps`	$\exists(i_1 \circ start, i_2 \circ start).< \ \sqcap \ \exists(i_2 \circ start, i_1 \circ end).< \ \sqcap \ \exists(i_1 \circ end, i_2 \circ end).<$
$\exists(i_1, i_2)$.`overlapped-by`	$\exists(i_2 \circ start, i_1 \circ start).< \ \sqcap \ \exists(i_1 \circ start, i_2 \circ end).< \ \sqcap \ \exists(i_2 \circ end, i_1 \circ end).<$
$\exists(i_1, i_2)$.`during`	$\exists(i_2 \circ start, i_1 \circ start).< \ \sqcap \ \exists(i_1 \circ end, i_2 \circ end).<$
$\exists(i_1, i_2)$.`contains`	$\exists(i_1 \circ start, i_2 \circ start).< \ \sqcap \ \exists(i_2 \circ end, i_1 \circ end).<$
$\exists(i_1, i_2)$.`starts`	$\exists(i_1 \circ start, i_2 \circ start).= \ \sqcap \ \exists(i_1 \circ end, i_2 \circ end).<$
$\exists(i_1, i_2)$.`started-by`	$\exists(i_1 \circ start, i_2 \circ start).= \ \sqcap \ \exists(i_2 \circ end, i_1 \circ end).<$
$\exists(i_1, i_2)$.`finishes`	$\exists(i_2 \circ start, i_1 \circ start).< \ \sqcap \ \exists(i_1 \circ end, i_2 \circ end).=$
$\exists(i_1, i_2)$.`finished-by`	$\exists(i_1 \circ start, i_2 \circ start).< \ \sqcap \ \exists(i_1 \circ end, i_2 \circ end).=$

Figure 3: Translation scheme between Allen's relations and a representation based on the concrete domain.

is achieved by means of syntactic sugaring at language level, while at reasoner level we rely on the concrete domain $\mathbb{Q}$ and the respective relations for dealing with representations based on intervals. A complete translation scheme between Allen's interval relations and equivalent relations based on the intervals' endpoints is presented in Figure 3.

A final issue regarding time in this context relates to its representation in tOWL ontologies. The actual representation of time in tOWL ontologies is based on XML Schema datatypes, namely *dateTime* as enabled by the concrete domain based on rational numbers and relations that may exist between these numbers.

In the current case, relationships such as the *starts* relationship in the previous example are enabled by the introduction of a constraint system based on intervals and Allen's 13 interval relations.

Finally, it should be noted that the definition of intervals as introduced by tOWL goes beyond the expressiveness of OWL-DL^{-} by relying on the concrete domain predicate $<$ and the two concrete features *start* and *end* for stating that the starting point of an interval should always be strictly smaller than its ending point:

$$\texttt{ProperInterval} \ \equiv \ \exists(\texttt{begin, end}).< \tag{2}$$

3.5 4d Fluents Layer

The concrete domain approach that enables a temporal infrastructure in ontologies as presented in the previous sections forms the basis for our approach. Building further upon these blocks, we seek to represent temporal aspects of entities other than timespan. In this context, the final level of expressiveness that we enable in tOWL regards different aspects of change, as identified in the introductory part of Section 3: i) change in a concrete attribute value of an individual, such as a change of hair color, ii) a change in the relationship between entities, such as a product that belongs/is produced by a company, and iii) state transitions in processes.

A perdurantist approach forms the foundation of this type of features. Up to a certain level, it can be argued that the fluents and timeslices employed for the representation of temporal information do not go beyond the expressiveness of OWL-DL$^-$. Rather, fluents and timeslices represent a kind of vocabulary employed for the representation of temporal parts of individuals that change some property in time. However, the semantics of fluents as envisioned for tOWL enforces a number of restrictions on tOWL specific concepts, and most importantly on fluents and timeslices. Some interesting features fuel the interdependence between the concrete domain and the timeslices/fluents approach and relate mostly to the restrictions this approach imposes on the very concepts it introduces.

One such restriction relates to the fact that fluents only relate timeslices that hold over the same time interval. Representing such a restriction involves the concept of equality of concrete values, and such a representation can thus only be enabled through the use of a concrete domain. We illustrate this idea through an example that we graphically depict in Figure 4. Here, we assume the existence of two OWL classes, namely *Company* and *Product*, each with one static individual - *iGoogle* and *iChrome*, respectively. What we seek to model is the fact that, over some period of time, the Chrome web-browser, represented here by the *iChrome* individual, is a product of the Google company, represented here by the *iGoogle* individual. In order to represent such a fact, we first instantiate a timeslice for each of the two concepts involved in this relation - *iGoogle_TS1* and *iChrome_TS1*, each linked through the *towl:timeSliceOf* property to the static individual it represents, as depicted in Figure 4. Each timeslice is described by an interval through the *towl:time* property. Finally, the *iGoogle_TS1* and *iChrome_TS1* timeslices are related by the *hasProduct* fluent of the type *towl:FluentObjectProperty.* The restriction that fluents only relate timeslices that hold over the same interval translates in this concrete case the *equals* Allen relation holding between the two intervals that are associated to the two timeslices that participate in the fluent relation. This is of course equivalent to a representation in terms of interval endpoints, as illustrated by the translation presented in Figure 3.

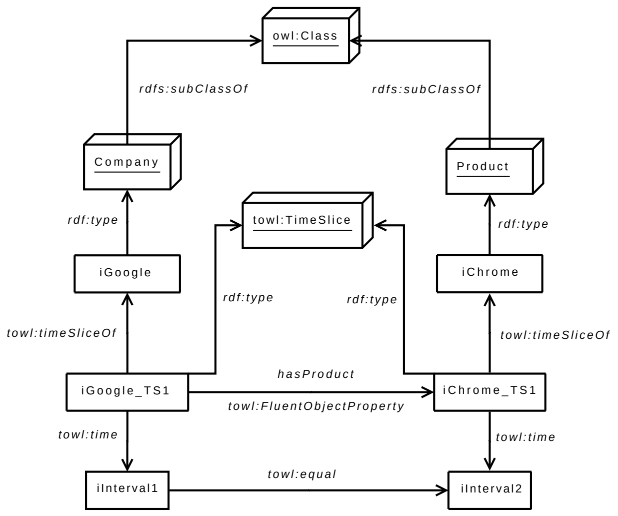

Figure 4: Temporal restrictions on timeslices connected by fluents.

In Figure 5 we present an overview of the tOWL TBox axioms corresponding to the timeslices/fluents layer.

tOWL 4dFluents Construct	**tOWL Axioms in OWL-DL**
Class(TimeSlice)	$\exists \mathtt{time.Interval} \sqcap \exists \mathtt{timeSliceOf}.\neg(\mathtt{TimeSlice} \sqcup \mathtt{Interval} \sqcup \mathtt{rdfs{:}Literal})$
Class(Interval)	$\exists(\mathtt{start}, \mathtt{end}).< \sqcap \exists \mathtt{start.dateTime} \sqcap \exists \mathtt{end.dateTime}$
Class(FluentProperty)	$\mathtt{FluentProperty} \sqsubset \mathtt{rdf{:}Property}$
Class(FluentObjectProperty)	$\mathtt{FluentObjectProperty} \sqsubset \mathtt{FluentProperty}$
Class(FluentDatatypeProperty)	$\mathtt{FluentDatatypeProperty} \sqsubset \mathtt{FluentProperty}$
Property(timeSliceOf)	$\geq 1\ \mathtt{timeSliceOf} \sqsubseteq \mathtt{TimeSlice}$ $\top \sqsubseteq \forall \mathtt{timeSliceOf}.\neg(\mathtt{TimeSlice} \sqcup \mathtt{Interval} \sqcup \mathtt{rdfs{:}Literal})$
Property(time)	$\geq 1\ \mathtt{time} \sqsubseteq \mathtt{TimeSlice}$ $\top \sqsubseteq \forall \mathtt{time.Interval}$
Property(start)	$\geq 1\ \mathtt{start} \sqsubseteq \mathtt{Interval}$ $\top \sqsubseteq \forall \mathtt{start.dateTime}$
Property(end)	$\geq 1\ \mathtt{end} \sqsubseteq \mathtt{Interval}$ $\top \sqsubseteq \forall \mathtt{end.dateTime}$

Figure 5: tOWL axioms for the *4DFluents* layer.

The fluents approach enables the determination of what holds true or false at a specific moment in time, but also what is unknown. The Open World Assumption (OWA) common on the Semantic Web, and for the OWL-DL language, is consistent with such an approach due to the fact that it uses strong negation. As opposed to a Closed World Assumption (CWA) which relies on the weak negation and thus associates a positive truth value even when it is not known what holds at a certain moment, the OWA does not associate any truth value when facts are not known. An overview of weak vs. strong negation is given in Figure 6.

World	**Weak negation**	**Strong negation**
true	true	true
unknown	true	unknown
false	false	false

Figure 6: Truth values in the case of weak vs. strong negation.

In the current case, this assumption translates to not associating any truth values to time intervals for which nothing is known from the temporal perspective, i.e., time intervals for which no timeslice describes what holds true over that interval. This case is illustrated in Figure 7. Here we represent timeslices for two individuals *iC* and *iD*, and choose a point in time, *t1*. While for the individual *iC* it can be determined what holds true through the timeslice *iC_TimeSlice1*, this is not the case for individual *iD* for which no timeslice is present at the specified point *t1*. In such a case, the OWA is equivalent to the world representation described in Figure 7, while in a CWA formalism everything would be true across the unknown period.

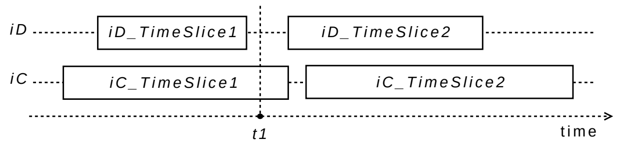

Figure 7: Determining what holds true at a point in time.

3.6 The tOWL Language

The goal of this section is to provide a self-contained description of the tOWL language which includes the OWL-DL$^-$ constructs allowed in tOWL for descriptions, axioms and facts.

We begin by providing an overview of the tOWL description enabled by the language in Figure 8. Here, C, D are used to denote class names, R is an object property, U is a datatype property, n is a positive integer, u_1, u_2 are constructs of type

tOWL Abstract Syntax	DL Syntax	Semantics
A (URI Reference)	A	$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
`towl:Thing`	$\top$	$\Delta^{\mathcal{I}}$
`towl:Nothing`	$\bot$	$\{\}$
`intersectionOf`$(C_1\ \ C_2 ...)$	$C_1 \sqcap C_2$	$C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}}$
`unionOf`$(C_1\ \ C_2 ...)$	$C_1 \sqcup C_2$	$C_1^{\mathcal{I}} \cup C_2^{\mathcal{I}}$
`complementOf`(C)	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
`restriction`$(R$ `someValuesFrom`$(C))$	$\exists R.C$	$\{x \mid \exists y.\langle x, y\rangle \in R^{\mathcal{I}} \text{ and } y \in C^{\mathcal{I}}\}$
`restriction`$(R$ `allValuesFrom`$(C))$	$\forall R.C$	$\{x \mid \forall y.\langle x, y\rangle \in R^{\mathcal{I}} \rightarrow y \in C^{\mathcal{I}}\}$
`restriction`$(R$ `minCardinality`$(n))$	$\geq n\ R$	$\{x \mid \sharp(\{y.\langle x, y\rangle \in R^{\mathcal{I}}\}) \geq n\}$
`restriction`$(R$ `maxCardinality`$(n))$	$\leq n\ R$	$\{x \mid \sharp(\{y.\langle x, y\rangle \in R^{\mathcal{I}}\}) \leq n\}$
`restriction`$(U$ `someValuesFrom`$(D))$	$\exists U.D$	$\{x \mid \exists y.\langle x, y\rangle \in U^{\mathcal{I}} \text{ and } y \in D^{\mathcal{D}}\}$
`restriction`$(U$ `allValuesFrom`$(D))$	$\forall U.D$	$\{x \mid \forall y.\langle x, y\rangle \in U^{\mathcal{I}} \text{ and } y \in D^{\mathcal{D}}\}$
`restriction`$(U$ `minCardinality`$(n))$	$\geq n\ U$	$\{x \mid \sharp(\{y.\langle x, y\rangle \in U^{\mathcal{I}}\}) \geq n\}$
`restriction`$(U$ `maxCardinality`$(n))$	$\leq n\ U$	$\{x \mid \sharp(\{y.\langle x, y\rangle \in U^{\mathcal{I}}\}) \leq n\}$
`ConcreteFeatureChain(`$f_1\ \ ...\ \ f_n\ \ g$`)`	$f_1 \circ ... \circ f_n \circ g$	$\{a_1 \in \Delta^{\mathcal{I}} \mid \exists! a_2 \in \Delta^{\mathcal{I}}, ..., \exists! a_{n+1} \in \Delta^{\mathcal{I}} \wedge$ $\wedge\ \exists! b \in \Delta_{\mathcal{D}} : (a_1, a_2) \in f_1^{\mathcal{I}}, ...$ $(a_n, a_{n+1}) \in f_n^{\mathcal{I}} \wedge g(a_{n+1}) = b\}$.
`restriction`$((u_1, u_2)$ `someValuesFrom`$(p_d))$	$\exists(u_1, u_2).p_d$	$\{x \in \Delta^{\mathcal{I}} \mid \exists! q_1 \in \Delta_{\mathcal{D}}, \exists! q_2 \in \Delta_{\mathcal{D}} : u_1^{\mathcal{I}}(x) = q_1 \wedge$ $\wedge\ u_2^{\mathcal{I}}(x) = q_2 \wedge\ (q_1, q_2) \in p_d^{\mathcal{I}}\}$
`restriction`$((u_1, u_2)$ `allValuesFrom`$(p_d))$	$\forall(u_1, u_2).p_d$	$\{x \in \Delta^{\mathcal{I}} \mid \forall q_1 \in \Delta_{\mathcal{D}}, \forall q_2 \in \Delta_{\mathcal{D}} : u_1^{\mathcal{I}}(x) = q_1 \wedge$ $\wedge\ u_2^{\mathcal{I}}(x) = q_2 \wedge\ (q_1, q_2) \in p_d^{\mathcal{I}})\}$

Figure 8: tOWL Descriptions

`towl:ConcreteFeatureChain` and p_d denotes a binary concrete domain predicate. The language description in Figure 8 is an adaptation of the summary of OWL-DL descriptions found in [18].

An overview of the tOWL axioms and facts that are enabled by the language is given in Figure 9. This is an adaptation of the summary of the OWL-DL axioms and facts found in [18], that has been extended with the tOWL specific constructs introduced by the language. Additionally, concepts related to the use of nominals have been excluded from this summary due to our current focus on the OWL-DL$^-$ language.

3.7 Reasoning

The tOWL language extends OWL-DL$^-$ through the addition of constructs that support the representation of time and temporal aspects. The $\mathcal{SHIN}(\mathcal{D})$ description logic, on which OWL-DL$^-$ is based, proves insufficient for the expressiveness introduced by the tOWL layers.

tOWL Abstract Syntax	DL Syntax	Semantics
Class(A partial $C_1 ... C_n$)	$A \sqsubseteq C_1 \sqcap ... \sqcap C_n$	$A^{\mathcal{I}} \subseteq C_1^{\mathcal{I}} \cap ... \cap C_n^{\mathcal{I}}$
Class(A complete $C_1 ... C_n$)	$A = C_1 \sqcap ... \sqcap C_n$	$A^{\mathcal{I}} = C_1^{\mathcal{I}} \cap ... \cap C_n^{\mathcal{I}}$
SubClassOf (C_1 C_2)	$C_1 \sqsubseteq C_2$	$C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$
EquivalentClasses ($C_1 ... C_n$)	$C_1 = ... = C_n$	$C_1^{\mathcal{I}} = ... = C_n^{\mathcal{I}}$
DisjointClasses ($C_1 ... C_n$)	$C_i \sqcap C_j = \bot, i \neq j$	$C_i^{\mathcal{I}} \cap C_j^{\mathcal{I}} = \{\}, i \neq j$
Datatype(D)		$D^{\mathcal{I}} \subseteq \Delta_D^{\mathcal{I}}$
DatatypeProperty(U super(U_1)...super(U_n)	$U \sqsubseteq U_i$	$U^{\mathcal{I}} \subseteq U_i^{\mathcal{I}}$
domain(C_1)...domain(C_m)	$\geq 1U \sqsubseteq C_i$	$U^{\mathcal{I}} \subseteq C_i^{\mathcal{I}} \times \Delta_D^{\mathcal{I}}$
range(D_1)...range(D_l)	$\top \sqsubseteq \forall U.D_i$	$U^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times D_i^{\mathcal{I}}$
[Functional])	$\top \sqsubseteq \ \leq 1U$	$U^{\mathcal{I}}$ is functional
SubPropertyOf(U_1 U_2)	$U_1 \sqsubseteq U_2$	$U_1^{\mathcal{I}} \subseteq U_2^{\mathcal{I}}$
EquivalentProperties($U_1 ... U_n$)	$U_1 = ... = U_n$	$U_1^{\mathcal{I}} = ... = U_n\mathcal{I}$
ObjectProperty(R super(R_1)...super(R_n)	$R \sqsubseteq R_i$	$R^{\mathcal{I}} \subseteq R_i^{\mathcal{I}}$
domain(C_1)...domain(C_m)	$\geq 1R \sqsubseteq C_i$	$R^{\mathcal{I}} \subseteq C_i^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
range(C_1)...range(C_l)	$\top \sqsubseteq \forall R.C_i$	$R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times C_i^{\mathcal{I}}$
[inverseOf(R_0)]	$R = (^{-}R_0)$	$R^{\mathcal{I}} = (R_0^{\mathcal{I}})^{-}$
[Symmetric]	$R = (^{-}R)$	$R^{\mathcal{I}} = (R^{\mathcal{I}})^{-}$
[Functional]	$\top \sqsubseteq \ \leq 1R$	$R^{\mathcal{I}}$ is functional
[InverseFunctional]	$\top \sqsubseteq \ \leq 1R^{-}$	$(R^{\mathcal{I}})^{-}$ is functional
[Transitive])	$\mathrm{Tr}(R)$	$R^{\mathcal{I}} = (R^{\mathcal{I}})^{+}$
SubPropertyOf(R_1 R_2)	$R_1 \sqsubseteq R_2$	$R_1^{\mathcal{I}} \subseteq R_2^{\mathcal{I}}$
EquivalentProperties($R_1 ... R_n$)	$R_1 = ... = R_n$	$R_1^{\mathcal{I}} = ... = R_n\mathcal{I}$
AnnotationProperty(S)		
FluentDatatypeProperty(U^{FD} super(U_1^{FD})...super(U_n^{FD})	$U^{FD} \sqsubseteq U_i^{FD}$	$(U^{FD})^{\mathcal{I}} \subseteq (U_i^{FD})^{\mathcal{I}}$
domain(C_1^{TS})...domain(C_m^{TS})	$\geq 1U^{FD} \sqsubseteq C_i^{TS}$	$(U^{FD})^{\mathcal{I}} \subseteq (C_i^{TS})^{\mathcal{I}} \times \Delta_D^{\mathcal{I}}$
range(D_1)... range (D_l)	$\top \sqsubseteq \forall U^{FD}.D_i$	$(U^{FD})^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times D_i^{\mathcal{I}}$
FluentObjectProperty(R^{FO} super(R_1^{FO})...super(R_n^{FO})	$R^{FO} \sqsubseteq R_i^{FO}$	$(R^{FO})^{\mathcal{I}} \subseteq (R_i^{FO})^{\mathcal{I}}$
domain(C_1^{TS})...domain(C_m^{TS})	$\geq 1R^{FO} \sqsubseteq C_i^{TS}$	$(R^{FO})^{\mathcal{I}} \subseteq (C_i^{TS})^{\mathcal{I}} \times \Delta_D^{\mathcal{I}}$
range(C_1^{TS})...range(C_l^{TS})	$\top \sqsubseteq \forall R^{FO}.C_i^{TS}$	$(R^{FO})^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times (C_i^{TS})^{\mathcal{I}}$
Individual(o type (C_1)... type (C_n)	$o \in C_i$	$o^{\mathcal{I}} \in C^{\mathcal{I}}$
value(R_1 o_1)... value (R_n o_n)	$\langle o, o_i \rangle \in R_i$	$\langle o^{\mathcal{I}}, o_i^{\mathcal{I}} \rangle \in R_i^{\mathcal{I}}$
value(U_1 v_1)... value (U_n v_n))	$\langle o, v_i \rangle \in U_i$	$\langle o^{\mathcal{I}}, v_i^{\mathcal{I}} \rangle \in U_i^{\mathcal{I}}$
SameIndividual($o_1 ... o_n$)	$o_1 = ... = o_n$	$o_1^{\mathcal{I}} = ... = o_n^{\mathcal{I}}$
DifferentIndividuals($o_1 ... o_n$)	$o_i \neq o_j, i \neq j$	$o_i^{\mathcal{I}} \neq o_j^{\mathcal{I}}, i \neq j$
TimeSlice(o^{TS} type (C_1^{TS})... type (C_n^{TS})	$o^{TS} \in C_i^{TS}$	$(o^{TS})^{\mathcal{I}} \in (C^{TS})^{\mathcal{I}}$
value(timeSliceOf o)	$\langle o^{TS}, o \rangle \in$ timeSliceOf	$\langle (o^{TS})^{\mathcal{I}}, o^{\mathcal{I}} \rangle \in \mathtt{timeSliceOf}^{\mathcal{I}}$
value(R_1^{FO} o_1^{TS})... value (R_n^{FO} o_n^{TS})	$\langle o^{TS}, o_i^{TS} \rangle \in R_i^{FO}$	$\langle (o^{TS})^{\mathcal{I}}, (o_i^{TS})^{\mathcal{I}} \rangle \in (R_i^{FO})^{\mathcal{I}}$
value(U_1^{FD} v_1)... value (U_n^{FD} v_n))	$\langle o^{TS}, v_i \rangle \in U_i^{FD}$	$\langle (o^{TS})^{\mathcal{I}}, v_i^{\mathcal{I}} \rangle \in (U_i^{FD})^{\mathcal{I}}$

Figure 9: tOWL Axioms and Facts

Currently, a reasoner has been implemented for the Lite version of the tOWL language. The tOWL-Lite language is based on the $\mathcal{ALC}(\mathcal{C})$ description logic, and is thus limited in expressiveness. However, this logic is sufficient for representing even complex cases, such as the Leveraged Buy Out example in Section 5. The reasoner is based on the algorithm described in [25], extended with a number of optimizations techniques meant to enhance the efficiency of the algorithm. The implemented optimizations are:

- Normalization and Simplification Normalization,
- TBox Absorption,
- RBox Absorption,
- Lazy Unfolding,
- Dependency-directed Backjumping, and
- Top-Bottom Search for Classification.

Rather than extending existing reasoners, the tOWL-Lite reasoner consists of a C++ implementation containing the tableau algorithm for the unrestricted version of the $\mathcal{ALC}(\mathcal{C})$ description logic as described in [25]. The execution of algorithms based on tableaux as an inference procedure for expressive logics requires a massive use of dynamic structures thus motivating the implementation of a new reasoner from scratch using C++.

4 RDF/XML Serialization

The main focus of this section is to present the serialization of the tOWL abstract syntax. We introduce the new constructs in RDF/XML by means of examples. Each of the following subsections is focussed on one tOWL construct.

4.1 towl:ConcreteFeatureChain

Concrete chains are represented through the `towl:ConcreteFeatureChain` construct. In order to enforce an ordering on the represented chain, this construct is defined as a subclass of *rdf:List*, and thus inherits its characteristics. In this fashion, it is possible to represent a functional role chain in RDF/XML as:

```
<towl:ConcreteFeatureChain
 rdf:ID="iEarlyStageChain">
  <rdf:first rdf:resource="#earlyStage" />
  <rdf:rest>
   <towl:ConcreteFeatureChain>
    <rdf:first rdf:resource="#time" />
    <rdf:rest rdf:resource="&rdf;nil" />
   </towl:ConcreteFeatureChain>
  </rdf:rest>
</towl:ConcreteFeatureChain>
```

Here, the RDF/XML code shown above translates to the following DL representation:

$$\texttt{earlyStage} \circ \texttt{time}$$

which represents the composition of two features, `earlyStage` and `time`, for individuals for which the `earlyStage` feature is defined.

4.2 Restrictions Based on Chains

We enable complex restrictions based on the`towl:ConcreteFeatureChain` construct or on the `towl:Concrete2RoleChain` construct in the tOWL language by extending `owl:Restriction` and by adding the `towl:onPropertyChains` construct. In order to enforce an ordering on the chains contained by the restriction, we represent these type of expressions as lists. Additionally, we employ the `towl:dataSomeValuesFrom` construct for representing an existential quantification:

```
<owl:Class rdf:ID="LBOProcess_TS">
 <rdfs:subClassOf>
  <towl:Restriction>
   <towl:onPropertyChains>
    <rdf:List>
     <rdf:first
      rdf:resource="#iEarlyStageChain" />
     <rdf:rest>
      <rdf:List>
       <rdf:first rdf:resource="#time" />
       <rdf:rest rdf:resource="&rdf;nil" />
      </rdf:List>
     </rdf:rest>
    </rdf:List>
   </towl:onPropertyChains>
   <towl:dataSomeValuesFrom
    rdf:resource="#starts"/>
  </towl:Restriction>
 </rdfs:subClassOf>
</owl:Class>
```

This translates to the following DL representation:

$$\exists(\mathtt{earlyStage} \circ \mathtt{time}, \mathtt{time}).\mathtt{starts}$$

4.3 towl:TimeSlice

Timeslices are available in tOWL through the *towl:TimeSlice* construct that represents the class of all timeslices. Individuals of this class are represented in the same way regular OWL class members are represented. Following this specification, a minimal introduction of a *towl:TimeSlice* individual is of the form:

```
<towl:TimeSlice rdf:ID="iEarlyStage1_TS1"/>
```

4.4 towl:Interval

In the same fashion timeslices are defined, one can also represent individuals of the *towl:Interval* class, as follows:

```
<towl:Interval rdf:ID="t1"/>
```

4.5 towl:FluentProperty

Two types of fluents are available in tOWL ontologies, namely fluents for which the range is a timeslice (*towl:FluentObjectProperty*) and fluents pointing to datatypes (*FluentDatatypeProperty*). Both are defined as subproperties of *towl:FluentProperty*, which in turn is a subclass of *rdf:Property*

4.5.1 towl:FluentObjectProperty

Fluents that link solely timeslices are represented as instances of the *towl:FluentObjectProperty*. This is achieved as follows:

```
<towl:FluentObjectProperty rdf:ID="inStage"/>
```

4.5.2 towl:FluentDatatypeProperty

Fluents that point to datatypes are represented as subproperties of the *towl:FluentDatatypeProperty*, in the same way this is achieved for *towl:FluentObjectProperty* relations:

```
<towl:FluentDatatypeProperty
        rdf:ID="stageImpact">
```

4.6 towl:timeSliceOf

The *towl:timeSliceOf* construct is used to link timeslices to the static individuals they represent across some temporal interval. Declarations involving this type of construct can be achieved by following the OWL-DL specification, as follows:

```
<towl:timeSliceOf rdf:ID="iAllianceBoots"/>
```

4.7 towl:time

The *towl:time* property is used to associate timeslices to the temporal interval across which they hold true. Declarations involving this type of construct can be achieved as follows:

```
<towl:time rdf:ID="t1"/>
```

4.8 towl:start

The *towl:start* property is used to indicate the starting point of individuals of type *towl:Interval*. This is achieved by making a direct reference to objects of type XML Schema *dateTime*. Declarations involving this type of construct can be achieved by following the OWL-DL specification, as follows:

```
<towl:start rdf:datatype="&xsd;dateTime">
            2007-12-25T12:30:00</start>
```

4.9 towl:end

The *towl:end* property is used to indicate the ending point of individuals of type *towl:Interval*. This is again achieved by making a direct reference to objects of type XML Schema dateTime. Declarations involving this type of construct can be achieved as follows:

```
                    <end rdf:datatype="&xsd;dateTime">
                             2007-12-26T12:30:00</end>
```

5 A Practical Application

The main focus of this section is to illustrate the use of the tOWL language in a temporal context. For this purpose, we focus on a complex process - Leveraged Buy Outs (LBO) in financial applications. In Section 5.1 we present LBO processes in general, and introduce the particular LBO, the Alliance Boots LBO, that is employed for the purpose of this section. In Section 5.2 we illustrate how such a process can be represented in the tOWL language.

5.1 Leveraged Buy Outs in General

A Leveraged Buy Out is a special type of an acquisition of a company by another company by relying mostly on loans for the price of the acquisition. Additionally, often enough the assets of the company that is to be acquired are used, partly of wholly, as collateral for the loans.

This type of process is of particular interest in the current case for 2 reasons: i) its high complexity is adequate for illustrating the main features of the tOWL language, and ii) the ability to deal with such a process in an automated fashion is also of interest in the economic domain, due to the high impact that the different stages have on the share prices of the involved companies.

An LBO process consists of a number of main stages, namely:

1. Early stage
2. Due diligence
3. Bidding
4. Acquisition

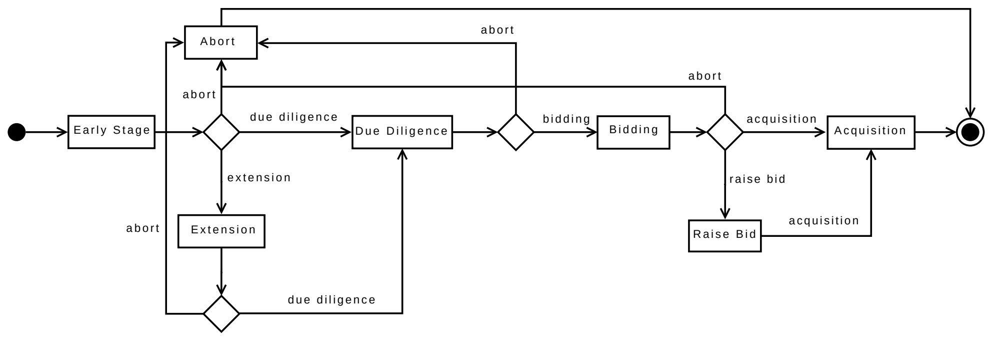

Figure 10: Stages of an LBO process.

The transition between stages is however not straightforward, as after nearly each stage the process can be aborted. Additionally, some of the stages may be extended before the transition into a different stage. In the bidding stage, this extension materializes in a raise of the current bid. We illustrate an LBO process in the light of its stage transitions by means of an activity diagram in Figure 10. The initial state of an LBO process is the *Early Stage*. From this stage, a transition can be made into the next state - *Due Diligence*, or this state may be extended, or the whole process can be aborted. Whether an extension or not is granted, the process may evolve to the *Due Diligence* stage. In case the process is not aborted in this stage, the LBO can continue with the *Bidding* phase. Again, besides the process being aborted, the LBO can continue with a *Raise Bid* phase in which the companies involved increase the amount they are prepared to lay down for the target. When the final bid is made and accepted, even in the case when no counter bids are made, the process slides into the *Acquisition* phase and ends.

The example introduced in this document describes the biggest LBO acquisition in Europe. In the March and April of 2007, two hedge funds competed for the acquisition of one target company. From the two hedge funds (KKR and Terra Firma), the first won the bidding and acquired target company Alliance Boots.

5.2 The Alliance Boots LBO in tOWL

The focus of this section is to illustrate how the information regarding an LBO process, both at an abstract level as well as in the particular example presented here, can be represented in Description Logics and tOWL abstract syntax, respectively. The main focus is on illustrating the main concepts that are relevant from a language perspective rather than providing a representation of the full example, thus avoiding repetition.

5.2.1 The LBO Example in DL Notation

In this Section we represent the LBO example in Description Logics. We illustrate the main concepts, both at TBox and Abox level, by means of examples.

TBox At TBox level we represent conceptual information that is known about LBO processes in general. In this context, two types of companies that take part in an LBO are known: `HedgeFund` and `Target`, which we define as subclasses of the `Company` class, as follows:

$$\begin{aligned} \texttt{HedgeFund} &\sqsubseteq \texttt{Company} \\ \texttt{Target} &\sqsubseteq \texttt{Company} \end{aligned}$$

This translates to the following representation in tOWL abstract syntax:

```
Class(Company)
Class(HedgeFund partial Company)
Class(Target partial Company)
```

The different stages of an LBO process are represented as subclasses of the `Stage` class, such as for example in the case of the `Bidding` stage:

$$\texttt{Bidding} \sqsubseteq \texttt{Stage}$$

This translates to the following representation in tOWL abstract syntax:

```
Class(Bidding partial Stage)
```

All stages are pairwise disjoint, which we represent as follows, for each unique pair of ordered stages:

$$\texttt{EarlyStage} \sqsubseteq \neg\texttt{DueDiligence}$$

This translates to the following representation in tOWL abstract syntax:

```
DisjointClasses(EarlyStage, DueDiligence,
      Bidding, RaiseBid, Acquisition,
      Abort, Extension)
```

We define the class of all timeslices of an LBO Process as follows:

$$\texttt{LBOProcess\_TS} \equiv \texttt{TimeSlice} \sqcap \exists\texttt{timeSliceOf.LBOProcess}$$

This translates to the following representation in tOWL abstract syntax:

```
Class(LBOProcess_TS complete
      restriction(timeSliceOf(someValuesFrom
         LBOProcess)))
```

In similar fashion, we define, for each stage, the class of all timeslices of that stage. For the `EarlyStage` this is achieved as follows:

$$\texttt{EarlyStage\_TS} \equiv \texttt{TimeSlice} \sqcap \exists\texttt{timeSliceOf.EarlyStage}$$

This translates to the following representation in tOWL abstract syntax:

```
Class(EarlyStage_TS complete
      restriction(timeSliceOf(someValuesFrom
        EarlyStage)))
```

For each stage, we define a functional property that links a particular LBO process timeslice to the timeslice of the stage belonging to it:

$$\begin{array}{l}\texttt{functional(earlyStage)}\\ \texttt{earlyStage} : \texttt{LBOProcess\_TS} \times \texttt{EarlyStage\_TS}\end{array}$$

This translates to the following representation in tOWL abstract syntax:

```
ObjectProperty(earlyStage
     domain(LBOProcess_TS)
     range(EarlyStage_TS))

Func(earlyStage)
```

Next, we move on to define the `inStage` fluent, that for each timeslice of a company points to the stage in which the company finds itself.

$$\begin{array}{lcl}\texttt{inStage} & : & (\exists\texttt{timeSliceOf.Company}) \times \\ & & (\exists\texttt{timeSliceOf.Stage})\end{array}$$

This translates to the following representation in tOWL abstract syntax:

```
FluentObjectProperty(inStage
   domain(
      restriction(timeSliceOf(someValuesFrom
         Company)))
   range(
      restriction(timeSliceOf(someValuesFrom
         Stage)))
```

Timeslices of an LBO process are defined by the sequence of stages that a company may follow in this process. Representing such sequences relies on functional role chains, and reduces to assessing the order of the intervals associated with the different stages. For example, representing that the `EarlyStage` always starts an LBO process can be represented as follows:

$$\mathtt{LBOProcess\_TS} \sqsubseteq \exists(\mathtt{earlyStage} \circ \mathtt{time}, \mathtt{time}).\mathtt{starts}$$

This translates to the following representation in tOWL abstract syntax:

```
Class(LBOProcess_TS partial
   restriction(
      dataSomeValuesFrom(
         ConcreteFeatureChain(earlyStage time),
             time, starts))))
```

ABox At ABox level we represent particular information that is known about the specific LBO process presented in this section. We start off by instantiating the relevant individuals that are known to play a role in the LBO process.

First, we represent the participating companies:

```
iAllianceBoots:Target
iKKR:HedgeFund
iTerraFirma:HedgeFund
```

This translates to the following representation in tOWL abstract syntax:

```
Individual(iAllianceBoots type(Target))
Individual(iKKR type(HedgeFund))
Individual(iTerraFirma type(HedgeFund))
```

For each of the hedgefunds involved, we instantiate a process and define its stages, such as in the case of the TerraFirma:

```
iLBOProcess1:LBOProcess
(iLBOProcess1_TS1,iLBOProcess1):timeSliceOf

(iLBOProcess1_TS1,iEarlyStage1_TS1):earlyStage
(iLBOProcess1_TS1,iDueDiligence1_TS1):dueDiligence
(iLBOProcess1_TS1,iBidding1_TS1):bidding
(iLBOProcess1_TS1,iAbort1_TS1):abort
```

This translates to the following representation in tOWL abstract syntax:

```
Individual(iLBOProcess1_TS1 type(LBOProcess_TS)
   value(timeSliceOf iLBOProcess_1))

LBOProcess_TS(iLBOProcess1_TS1
   value(earlyStage iEarlyStage1_TS1)
   value(dueDiligence iDueDiligence1_TS1)
   value(bidding iBidding1_TS1)
   value(abort iAbort1_TS1))
```

Next, we represent the information contained by the individual news messages associated with the LBO process. We illustrate this by employing the first news message that describes the hedgefund `TerraFirma` entering the `EarlyStage` phase. Here, we only present a summary of the actual news message and indicate the stage that is signaled by it. The date and time associated to the news message is the one as specified on `http://www.marketwatch.com/`, and represents the time when the news message was issued and thus became available to the wide public.

Buyout firm Terra Firma mulls Boots bid

Sun Mar 25, 2007 8:42am EDT

This news message signals the beginning of the LBO, mentioning that Terra Firma is considering a bid for Alliance Boots (*EarlyStage*)

For representing the information contained in the news message we create a timeslice for the hedgefund and the target, respectively, a time interval associated to the stage, and employ the `inStage` fluent to associate the companies to the stage:

```
t1:Interval
iEarlyStage1:EarlyStage

iEarlyStage_TS1:EarlyStage_TS
(iEarlyStage_TS1,iEarlyStage1):timeSliceOf
(iEarlyStage_TS1,t1):time

iAllianceBoots_TS1:TimeSlice
(iAllianceBoots_TS1,iAllianceBoots):timeSliceOf
iAllianceBoots_TS1,t1:time
(iAllianceBoots_TS1,iEarlyStage_TS1):inStage

iTerraFirma_TS1:TimeSlice
(iTerraFirma_TS1,iTerraFirma):timeSliceOf
(iTerraFirma_TS1,t1):time
(iTerraFirma_TS1,iEarlyStage_TS1):inStage
```

This translates to the following representation in tOWL abstract syntax:

```
Individual(t1 type(Interval))

Individual(iEarlyStage1 type(EarlyStage_TS))

Individual(iEarlyStage1_TS1 type(TimeSlice)
  value(timeSliceOf iEarlyStage1)
```

```
    value(time t1))


Individual(iAllianceBoots_TS1 type(TimeSlice)
    value(timeSliceOf iAllianceBoots)
    value(time t1)
    value(inStage iEarlyStage1_TS1))
Individual(iTerraFirma_TS1 type(TimeSlice)
    value(timeSliceOf iTerraFirma)
    value(time t1)
    value(inStage iEarlyStage1_TS1))
```

6 Conclusion & Further Research

The tOWL language is an extension of OWL-DL$^-$ that enables the representation and reasoning with time and temporal aspects. It comes to meet shortcomings of previous approaches, such as [17, 39] that only address this issue to a limited extent.

Summarizing, tOWL is a new temporal web ontology language that enables the representation of dynamic worlds. It extends the OWL-DL$^-$ language with concrete domains, and enables class axioms that rely on binary concrete domain predicates that can also be employed in combination with property chains. The language provides a concrete domain based on the set $\mathbb{Q}$ of rational numbers and the set of binary concrete domain predicates $\{<, \leq, =, \neq, \geq, >\}$. By means of syntactic sugaring we also introduce intervals and Allen's 13 interval relations that may hold between intervals in the language. Additionally, a fluents approach is employed for the representation of the different aspects of change considered relevant for the tOWL language. Building on the approach presented in [39], it extends the latter by making a difference between fluents that point to datatypes and fluents that point to objects, thus limiting the proliferation of objects inherent to this approach, since less timeslices are created in the case of datatype fluents. The tOWL language can be employed for representation and reasoning in a wide variety of dynamic domains, such as the financial one as exemplified in this paper.

Acknowledgment

The authors are partially supported by the EU funded IST STREP Project FP6 - 26896: Time-determined ontology-based information system for realtime stock market analysis. More information on the project is available on the official project website at `http://www.semlab.nl/towl`.

The authors would like to thank Alessandro Artale, Tommaso di Noia and Kurt Sandkuhl for their valuable reviews and suggestions regarding (parts of) the current work.

References

[1] The Stanford Encyclopedia of Philosophy, 2003.

[2] J. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843, 1983.

[3] G. Ariav. A temporally oriented data model. *ACM Transactions on Database Systems*, 11(4):499–527, 1986.

[4] A. Artale and E. Franconi. A computational account for a description logic of time and action. In *The 4th Conference on Principles of Knowledge Representation and Reasoning (KR 1994)*, pages 3–14. Morgan Kaufmann, 1994.

[5] A. Artale and E. Franconi. A temporal description logic for reasoning about actions and plans. *Journal of Artificial Intelligence Research*, 9(2):463–506, 1998.

[6] A. Artale and E. Franconi. A survey of temporal extensions of description logics. *Annals of Mathematics and Artificial Intelligence*, 30(1):171–210, 2000.

[7] A. Artale, E. Franconi, M. Mosurovic, F. Wolter, and M. Zakharyaschev. The $\mathcal{DLR}_{\mathcal{US}}$ Temporal Description Logic. In *The 2001 Description Logic Workshop (DL 2001)*, pages 96–105. CEUR Workshop Proceedings, 2001.

[8] F. Baader and P. Hanschke. A scheme for integrating concrete domains into concept languages. In *The 12th International Joint Conference on Artificial Intelligence, (IJCAI 1991)*, pages 452–457. Morgan Kaufmann, 1991.

[9] D. Beech and B. Mahbod. Generalized version control in an object-oriented database. In *The Fourth International Conference on Data Engineering (ICDE 1988)*, pages 14–22. IEEE Computer Society, 1988.

[10] J. Ben-Zvi. The time relational model. 1982.

[11] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, 284(5):28–37, 2001.

[12] D. Brickley and R. Guha. RDF vocabulary description language 1.0: RDF schema. *W3C Recommendation*, 2004.

[13] J. Clifford and A. Croker. The Historical Relational Data Model (HRDM) Revisited. *Temporal Databases: Theory, Design, and Implementation*, pages 6–26, 1993.

[14] S. Gadia. A homogeneous relational model and query languages for temporal databases. *ACM Transactions on Database Systems*, 13(4):418–448, 1988.

[15] C. Gutierrez, C. Hurtado, and A. Vaisman. Introducing time into RDF. *IEEE Transactions on Knowledge and Data Engineering*, 19(2):207–218, 2007.

[16] P. Hayes and B. McBride. RDF Semantics. W3C Recommendation, 2004.

[17] J. Hobbs and F. Pan. An ontology of time for the semantic web. *ACM Transactions on Asian Language Information Processing*, 3(1):66–85, 2004.

[18] I. Horrocks, P. Patel-Schneider, and F. Van Harmelen. From SHIQ and RDF to OWL: The making of a web ontology language. *Web semantics: science, services and agents on the World Wide Web*, 1(1):7–26, 2003.

[19] C. Jensen, J. Clifford, R. Elmasri, S. Gadia, P. Hayes, and S. Jajodia. A consensus glossary of temporal database concepts. *SIGMOD Record*, 23(1):52–64, 1994.

[20] C. Jensen, M. Soo, and R. Snodgrass. Unifying temporal data models via a conceptual model. *Information Systems*, 19(7):513–547, 1994.

[21] N. Keberle, Y. Litvinenko, Y. Gordeyev, and V. Ermolayev. Ontology evolution analysis with OWL-MeT. In *International Workshop on Ontology Dynamics (IWOD-07)*, pages 1–12, 2007.

[22] G. Klyne and J. Carroll. Resource description framework (RDF): Concepts and abstract syntax. *W3C Recommendation*, 2004.

[23] C. Lutz. Adding numbers to the SHIQ description logic: First results. *The Eighth International Conference on Principles of Knowledge Representation and Reasoning (KR 2002)*, pages 191–202, 2002.

[24] C. Lutz. Description logics with concrete domains–a survey. In *Advances in Modal Logics*, volume 4, pages 265–296. King's College Publications, 2003.

[25] C. Lutz and M. Milicic. A tableau algorithm for description logics with concrete domains and general tboxes. *Journal of Automated Reasoning*, 38(1–3):227–259, 2007.

[26] J. McCarthy and P. Hayes. Some philosophical problems from the standpoint of artificial intelligence. *Machine Intelligence*, 4:463–502, 1969.

[27] V. Milea, F. Frasincar, and U. Kaymak. Knowledge engineering in a temporal semantic web context. In *The Eighth International Conference on Web Engineering (ICWE 2008)*, pages 65–74. IEEE Computer Society Press, 2008.

[28] V. Milea, F. Frasincar, and U. Kaymak. The tOWL web ontology language. In *The 20th Belgian-Dutch Conference on Artificial Intelligence (BNAIC 2008)*, 2008.

[29] V. Milea, F. Frasincar, U. Kaymak, and T. di Noia. An OWL-based approach towards representing time in web information systems. In *The 4th International Workshop of Web Information Systems Modeling Workshop (WISM 2007)*, pages 791–802. Tapir Academic Press, 2007.

[30] V. Milea, M. Mrissa, K. van der Sluijs, and U. Kaymak. On temporal cardinality in the context of the TOWL language. In *The 5th International Workshop of Web Information Systems Modeling Workshop (WISM 2008)*, pages 457–466. Springer, 2008.

[31] N. Noy and M. Klein. Ontology evolution: Not the same as schema evolution. *Knowledge and Information Systems*, 6(4):428–440, 2004.

[32] N. Noy and A. Rector. Defining n-ary relations on the semantic web. *Working Draft for the W3C Semantic Web best practices group*, 2005.

[33] G. Ozsoyoglu and R. Snodgrass. Temporal and real-time databases: a survey. *IEEE Transactions on Knowledge and Data Engineering*, 7(4):513–532, 1995.

[34] F. Pan and J. Hobbs. Temporal Aggregates in OWL-Time. *Proceedings of the 18th International Florida Artificial Intelligence Research Society Conference (FLAIRS)*, pages 560–565, 2005.

[35] P. Patel-Schneider, Hayes, and I. P., Horrocks. Web ontology language (OWL) abstract syntax and semantics. *W3C Recommendation*, 2004.

[36] E. Rose and A. Segev. TOODM: A temporal object-oriented data model with temporal constraints. In *The International Conference on the Entity Relationship Approach (ER 1991)*, pages 205–229. ER Institute, 1991.

[37] K. Schild. Combining Terminological Logics with Tense Logic. In *The 6th Portugese Conference on Artificial Intelligence*, pages 105–120. Springer, 1993.

[38] M. Stonebraker, L. Rowe, and M. Hirohama. The implementation of POSTGRES. *IEEE Transactions on Knowledge and Data Engineering*, 2(1):125–142, 1990.

[39] C. Welty and R. Fikes. A reusable ontology for fluents in OWL. In *The Fourth International Conference on Formal Ontology in Information Systems (FOIS 2006)*, pages 226–336. IOS Press, 2006.

[40] G. Wuu and U. Dayal. A uniform model for temporal object-oriented databases. In *The Eighth International Conference on Data Engineering (ICDE 1992)*, pages 584–593, 1992.

Publications in the Report Series Research* in Management

ERIM Research Program: “Business Processes, Logistics and Information Systems”

2009

How to Normalize Co-Occurrence Data? An Analysis of Some Well-Known Similarity Measures
Nees Jan van Eck and Ludo Waltman
ERS-2009-001-LIS
http://hdl.handle.net/1765/14528

Spare Parts Logistics and Installed Base Information
Muhammad N. Jalil, Rob A. Zuidwijk, Moritz Fleischmann, and Jo A.E.E. van Nunen
ERS-2009-002-LIS
http://hdl.handle.net/1765/14529

Open Location Management in Automated Warehousing Systems
Yugang YU and René B.M. de Koster
ERS-2009-004-LIS
http://hdl.handle.net/1765/14615

VOSviewer: A Computer Program for Bibliometric Mapping
Nees Jan van Eck and Ludo Waltman
ERS-2009-005-LIS
http://hdl.handle.net/1765/14841

Nash Game Model for Optimizing Market Strategies, Configuration of Platform Products in a Vendor Managed Inventory (VMI) Supply Chain for a Product Family
Yugang Yu and George Q. Huang
ERS-2009-009-LIS
http://hdl.handle.net/1765/15029

A Mathematical Analysis of the Long-run Behavior of Genetic Algorithms for Social Modeling
Ludo Waltman and Nees Jan van Eck
ERS-2009-011-LIS
http://hdl.handle.net/1765/15181

A Taxonomy of Bibliometric Performance Indicators Based on the Property of Consistency
Ludo Waltman and Nees Jan van Eck
ERS-2009-014-LIS
http://hdl.handle.net/1765/15182

A Stochastic Dynamic Programming Approach to Revenue Management in a Make-to-Stock Production System
Rainer Quante, Moritz Fleischmann, and Herbert Meyr
ERS-2009-015-LIS
http://hdl.handle.net/1765/15183

Some Comments on Egghe’s Derivation of the Impact Factor Distribution
Ludo Waltman and Nees Jan van Eck
ERS-2009-016-LIS
http://hdl.handle.net/1765/15184

The Value of RFID Technology Enabled Information to Manage Perishables
Michael Ketzenberg, and Jacqueline Bloemhof
ERS-2009-020-LIS
http://hdl.handle.net/1765/15412

* A complete overview of the ERIM Report Series Research in Management:
https://ep.eur.nl/handle/1765/1

ERIM Research Programs:

LIS	Business Processes, Logistics and Information Systems
ORG	Organizing for Performance
MKT	Marketing
F&A	Finance and Accounting
STR	Strategy and Entrepreneurship