tOWL: A Temporal Web Ontology Language

Viorel Milea, *Graduate Student Member, IEEE,* Flavius Frasincar, and Uzay Kaymak, *Member, IEEE*

***Abstract*—Through its interoperability and reasoning capabilities, the Semantic Web opens a realm of possibilities for developing intelligent systems on the Web. The Web Ontology Language (OWL) is the most expressive standard language for modeling ontologies, the cornerstone of the Semantic Web. However, up until now no standard way of expressing time and time-dependent information in OWL has been provided. 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 three extensions: i) *Concrete Domains*, which allows the representation of restrictions using concrete domain binary predicates, ii) *Temporal Representation*, which introduces timepoints, relations between timepoints, intervals, and Allen's 13 interval relations into the language, and iii) *TimeSlices/Fluents*, which 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 using an example from the financial domain.**

***Index Terms*—machine communication, intelligent systems, Semantic Web, time representation, OWL, fluents, concrete domains.**

I. Introduction

THE considerable and 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 [1]. The state-of-the-art tools and languages provided under this umbrella, such as Resource Description Framework (RDF), RDF Schema (RDFS) [2], [3] and OWL [4], 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.

Focused on 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 are also impractical in environments that require temporal awareness. Examples of such environments are the financial domain, scheduling, marketing, etc. Within the financial domain, for example, one can envision the need for representing ephemeral knowledge, contained for instance in news messages (e.g., stock price and other financial variables), or more fundamental aspects of the financial domain (e.g., mergers and acquisitions, financial processes, etc.).

The authors are with the Econometric Institute, Erasmus School of Economics, Erasmus University Rotterdam, the Netherlands. Uzay Kaymak is also affiliated with the School of Industrial Engineering, Eindhoven University of Technology, the Netherlands. E-mail: {milea,frasincar}@ese.eur.nl, u.kaymak@ieee.org.

Consider, for example, the temporary relation between a person and a company in which that person is the Chief Executive Officer (CEO) of that company. Such a relation is described by a temporal interval across which a person fulfills the function of CEO for a company. For example, until 16 October 2008, Jack Dorsey was the CEO of Twitter. On that date Jack Dorsey stepped down and Evan Williams became the new CEO of the company. In the standard Semantic Web approach based on OWL-DL once the CEO of the company changes, there is no way to represent both CEOs and the times associated with them. Furthermore, one would like to reason with temporal information. For example, assuming that Evan Williams is only CEO for a limited amount of time, and that the ending point of him being a CEO of Twitter is known, we would like to consistently represent all this information in our temporal language, without any loss of information regarding the *ceoOf* relationship. In graphical terms, what we would like to represent in a temporal Semantic Web language is illustrated in Figure 1. For this representation, we would like to be able to define temporal constraints such as that the starting point of a time interval should be before the ending point of the interval (in our example $t_1 < t_2$ and $t_2 < t_3$). Such knowledge and constraints cannot be be enforced using OWL-DL semantics. In this paper, we propose a temporal extension to the OWL language that allows us to represent and reason with temporal information in the Semantic Web.

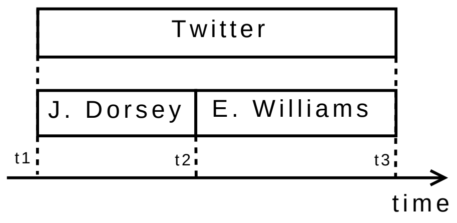

Fig. 1. Change of CEO in the Twitter example.

In general, 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 regards temporal concepts such as the ephemeral character of relationships between individuals. In this context, representations of change should be possible. These representations include 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 context, 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 paper is 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 it is the most expressive fragment of OWL-DL extended with concrete domains for which a terminating, sound, and complete reasoning algorithm is known [5].

In temporal terms, the extension of OWL-DL^{-} that we envision addresses time in the sense of a 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) [6]–[10]. Hence, the tOWL language is an extension of OWL-DL^{-} that enables the representation of and reasoning with time and temporal aspects.

The extension of OWL that we present in this paper is mainly aimed at extending the communication between machines to contexts that require temporality. Building upon the main goals of Semantic Web languages, tOWL is not only aimed at enabling the inference of implicit knowledge when a temporal dimension is involved, but aims at representing information, especially information that is temporal in nature, in an unambiguous fashion, in a unified way that ensures the preservation of meaning across different machines. Due to the desirable computational properties of the language, which is based on a decidable description logic extended with concrete domains, we enable temporal reasoning in tOWL knowledge bases that extends well beyond the capabilities of any of the Semantic Web languages developed until now.

Generally speaking, ontologies are separated into a TBox and an ABox. The TBox contains the terminological knowledge in the ontology, and refers to classes and properties. The ABox is the assertional part of the ontology, and relates to individuals - instances of the classes described in the TBox. The issue of time is also relevant in the context of TBoxes and ABoxes of ontologies. Rather than focusing on the evolution of ontologies, i.e., changes at TBox level, we solely focus on changes in the ABox (we assume that the domain structure is known).

The outline of the paper is as follows. In Section II we provide an overview of work related to the current endeavor. Section III introduces different layers of the tOWL language built on top of OWL-DL^{-}. An extensive example of how the expressiveness of tOWL can be employed for the representation of a financial process is provided in Section IV. We give a discussion Section V, where we place our proposed language into a broader context. Finally, we conclude in Section VI.

II. Temporal Representations

World representations may be synchronic or diachronic in the way the temporal perspective is considered within the representation [11]. 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, which are our current focus, must deal with the problem of diachronic identity, or put differently, establish how change affects the identity of entities existing at different times.

Leibniz provided two principles regarding the issue of identity [12]. 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 thereof are identical. Such choices are mainly concerned with the field of logics. As our focus is on the Semantic Web, special attention is given to approaches related to Description Logics [13]. The issue of temporality has also been addressed in this context, presenting several choices regarding the handling of the temporal dimension. The main distinction separating approaches in temporal description logics can be made in terms of whether the temporal language offers explicit time, or whether the temporal dimension is only implicitly present in the language by providing the means to talk about an order of events and/or states. Following [14], these different approaches are categorized as *explicit* and *implicit*, respectively.

Additionally, philosophy presents us with two main theories regarding the persistence of objects through time: endurantism and perdurantism. Endurantism refers to the view that objects are three-dimensional, and persist through time, i.e., are always present. Perdurantism, or four-dimensionalism, regards objects as being composed of temporal parts. The identity of a four dimensional object then consists of all the temporal parts of that object, i.e., all instances of that object through time. An approach related to incorporating perdurants through the use of timeslices and fluents is presented in [15], where the authors develop a reusable ontology for fluents in OWL-DL. In this approach, timeslices represent the temporal parts of a specific entity at intervals in time and the concept itself is then defined as all of its timeslices. Fluents are properties that hold at a specific moment in time, or at a specific interval in time. One of the drawbacks of this approach is the proliferation of objects in the ontology due to the creation of two timeslices each time something is changing, which, in turn, must be associated to the static individuals they represent and linked to each other by a fluent. Further, no solution is provided for the temporal equivalent of the cardinality construct, which cannot be modeled in the case of overlapping timeslices [15]. Finally, the time associated to timeslices relies on the OWL-Time ontology, rather than on a more expressive approach based on concrete domains.

The expressiveness of description logics is usually denoted with a series of letters, such as $\mathcal{SHOIN}(\mathcal{D})$, where each letter stands for a level of expressiveness. A language that allows functional properties will contain the letter $\mathcal{F}$ in its name, while $\mathcal{S}$ stands for the $\mathcal{ALC}$ language (attributive language with complement) with transitive roles, $\mathcal{H}$ stands for role hierarchy, $\mathcal{O}$ for nominals, $\mathcal{I}$ for inverse roles, and $\mathcal{N}$ for number restrictions. The $\mathcal{SHOIN}$ language thus is the language that incorporates the expressiveness associated with each letter as described above, and $\mathcal{SHOIN}(\mathcal{D})$ additionally provides support for data types, as indicated by $\mathcal{D}$.

When logics, and especially description logics, are considered, concepts such as decidability, concept satisfiability, and subsumption play an important role. Decidability relates to whether a method exists such that formulas validity can always be determined in a finite number of steps. Concept satisfiability consists of checking whether an assertion has a model, i.e., its interpretation is non-empty. Finally, subsumption relates to being able to determine whether one concept is more general than some other concept [13]. Decidability is highly relevant, especially in the context of the Semantic Web, where machines must be endowed with the capability to reason on the knowledge that is being presented to them. The focus of the Semantic Web on description logics comes from the fact that the DL community has always placed emphasis on the decidability of the logics introduced. As our temporal extension of the web ontology language is also intended for the Semantic Web, special attention is given to the decidability of the language. When considering relevant literature on temporal extensions to description logics, we also focus on the decidability of the temporal extension.

The interval-based $\mathcal{TL}$-$\mathcal{ALCF}$ description logic [16], [17] 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. 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 our goal. This relates mostly to the fact the $\mathcal{ALCF}$ description logic is much less expressive than $\mathcal{SHIN}(\mathcal{D})$, which is our main focus for the temporal extension. In the temporal language that we propose, role hierarchy ($\mathcal{H}$) and inverse roles ($\mathcal{I}$) play an important role. For example, the fluent *ceoOf* is a more specific variant of the *worksFor* fluent, which can only be represented in the knowledge base when role hierarchy is enabled. Additionally, the inverse of the *ceoOf* fluent, *hasCEO*, can only be represented in the knowledge base when inverse roles are enabled by the language.

Approaches similar to $\mathcal{TL}$-$\mathcal{ALCF}$, but relying on a point-based temporal structure rather than an interval-based one, provide the 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 [18]. 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 [14]. These approaches are not sufficient for our current purpose mainly because extending the expressiveness of the static DL in this way easily leads to undecidability.

Time can also be incorporated in a 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 [19], concrete domains allow abstract concepts to be related to concrete values through functional roles (roles that take exactly one value). Description logics extended with concrete domains maintain decidability, provided that the concrete domain satisfies the property of admissibility or ω-admissibility [19], [20]. 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 [20], such as 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 less restricted by the expressiveness of the static DL which it extends. Indeed, results are known for $\mathcal{SHIQ}(\mathcal{C})$, description logic for which a terminating, sound and complete reasoning algorithm is known [5].

Linear time temporal logic (LTL) has also been considered as a temporal extension of DLs [21], particularly with regard to the $\mathcal{ALC}$ description logic, resulting in the $LTL_{\mathcal{ALC}}$ temporal DL. However, extensions of this TDL to more expressive DLs, such as $\mathcal{SHIQ}$, easily results in undecidability [21], thus making such an approach unsuitable due to the limited expressiveness of the underlying language.

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 three types of time [22] that may be considered 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. Combinations of these types are also possible. When valid time and transaction time are considered together, this results in bitemporal data models [22]. Regarding the structure of the time domain, a further distinction may be made between linear time - one time flow from past, through present, to future - and branching time, where the representation of possible, alternative futures is allowed [23].

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 [24]. This work is similar to our approach 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 [25], which focuses on the Web Ontology Language rather than RDF. The initial purpose behind the design of a time ontology (OWL-Time) was to represent the temporal content of Web pages and the temporal properties of Web Services (DAML-Time) [25]. 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 data types rather than concrete domains for the description of instants and intervals. Due to this fact, proper intervals, i.e., intervals for which the starting point is strictly smaller than the end point, cannot be represented in this approach. Semantic Web approaches similar to ours also include [26], [27], relating to a 4d fluents approach for representing change, and [28] focusing on the representation of valid time in RDF and OWL. In the following, we discuss our proposed temporal extension of the web ontology language.

III. The Temporal Web Ontology Language

Designing a temporal extension of OWL-DL$^-$ begins with a clarification of what is understood under the general, common denominator *time*. 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 aim for 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.

Regarding the second aspect, *change*, we note 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 the ABox level. We allow three types of change: 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 is built 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. Therefore, we seek to represent when certain changes take place in the actual world rather than the time when they are represented in the ontology.

In the following we discuss the details of our proposed tOWL language. The design uses the results from temporal logic, temporal databases, and Semantic Web research where possible. The design choices are explained in Section III-A. The individual tOWL layers are presented, one by one, in Sections III-B through III-D. A discussion on reasoning in the tOWL language is presented in Section III-E.

A. Design Choices

For the tOWL language there are a number of choices regarding the most suitable approach(es) for the representation of the two temporal aspects considered above. 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 known in the literature, the only method suitable for our goals is the one based on concrete domains. The temporal infrastructure then 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 $\mathcal{SHIQ}(\mathcal{C}^+)$ [5], [29]. 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 data type and rational numbers, but enables also the description of any numerical attribute through a direct reference to the concrete domain.

In our approach, we seek to enable an interval-based representation of time satisfying the previously mentioned constraints. For this purpose, we aim to add intervals and Allen's 13 interval relations [30] to the tOWL language. As known from [30], 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 [19], [29] 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. We consider diachronic representations that take history into account rather than synchronic ones, and are thus faced with the problem of diachronic identity, as mentioned in Section II. The second principle of Leibniz, indiscernibility of identicals, poses an additional restriction on the choice of representation and the 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 our case, as ternary predicates are not directly supported in OWL-DL. The W3C Semantic Web Best Practices working group provides three alternative ways of representing n-ary relationships on the Semantic Web [31], 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" [31]. Besides the fact that the RDF "reification of a triple does not entail the triple, and is not entailed by it" [32], reification is not supported at all in OWL-DL$^-$. Since we are extending OWL-DL$^-$, such an approach is not suitable.

Another approach for associating valid time with a binary relation relates to the addition of a meta-logical [33] predicate that takes as arguments the binary relationship and the time when this relationship holds. However, as also discussed in [15], such predicates are not supported in any of the OWL species. The fluents approach presented in [15] and discussed in Section II 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 description logic it extends, as it is more concerned with syntactic sugaring rather than being a semantic extension. As introduced in [15], this 4D approach can be achieved in the form of an OWL ontology, which although insufficient for 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 choose 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.

B. Concrete Domains Layer

The representation of complex restrictions, regardless of whether they describe some temporal aspect, or 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, forming what is commonly denoted as a *concrete feature path* (CFP), and which is mathematically equivalent to the composition:

$$f_1 \circ f_2 \circ ... \circ f_n \circ g, \tag{1}$$

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

An example of such a CFP could consist of the composition of the *time* abstract feature and the *start* concrete feature, resulting in a composition of type $f_1 \circ g$, where f_1 is the *time* feature and g is represented by the *start* feature, as follows:

$$\textit{time} \circ \textit{start}. \tag{2}$$

A construction such as the one in (2) would denote the starting point of an interval by first applying the *time* abstract feature to obtain the interval associated with an individual and then the *start* concrete feature to obtain the starting point of that interval.

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:

$$\exists(u_1, u_2).p_d, \tag{3}$$
$$\forall(u_1, u_2).p_d. \tag{4}$$

For such constructs, u_i may arbitrarily denote a CFP of length m, with $m \in \mathbb{N}^*$. Such constructs are useful for defining, for example, that the starting point of an interval should be strictly smaller than the ending point of that interval. Such a definition of an interval would take the following form:

$$\exists(\textit{time} \circ \textit{start}, \textit{time} \circ \textit{end}). < \tag{5}$$

where we employ the $<$ concrete domain predicate (p_d) to state that the starting point of some interval is strictly smaller than its ending point.

We summarize the semantics introduced by this layer in Table I, with reference to the tOWL abstract syntax constructs we propose. In Table I, f_n denotes an abstract feature, g a concrete feature, u_i is a concrete feature chain, a_i and x are individuals from the abstract domain, b, q_1, and q_2 are concrete values, and p_d is a concrete domain predicate. For a complete overview of the tOWL language we refer the reader to the appendix.

The first definition in this table, that of a `ConcreteFeatureChain`, states that the interpretation of such a concept consists of all those pairs of individuals of the abstract domain and the concrete domain, respectively, such that each of these abstract individuals is in the interpretation of the f_1 abstract feature together with exactly one other abstract individual, a_2, which in turn is in the interpretation of f_2, together with exactly one other individual, a_3, and so on to a_{n+1}. Finally, the interpretation of the concrete feature g on the individual a_{n+1} should be defined and should take on exactly one concrete value, namely b.

The `dataSomeValuesFrom` construct, states that the interpretation of such a concept consists of all those individuals from the abstract domain such that, when the two concrete feature chains u_1 and u_2 are interpreted over these individuals, the result consists of the q_1 and q_2 unique concrete values that are in the interpretation of the p_d concrete domain predicate. Hence, they can be described by p_d. Since this is an existential quantification, the values involved should exist, i.e., be explicitly defined.

TABLE I
SEMANTICS FOR THE CONCRETE DOMAINS LAYER.

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

The `dataAllValuesFrom` construct is similar to the `dataSomeValuesFrom`, with the exception that this time a universal quantification is involved, which means that the p_d relation should hold true for all values of q_1 and q_2. The difference between these two constructs is that, in the case of `dataAllValuesFrom`, the relationship can hold true when q_1 and q_2 are missing.

C. 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. In 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 III-A. 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 [34]. Rather than being a concrete domain, this extension is achieved by means of syntactic sugaring at language level, while at reasoner level we rely on the concrete domain $\mathbb{Q}$ and the corresponding relations for dealing with representations based on intervals.

Another 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 data types, namely *dateTime* as enabled by the concrete domain based on rational numbers and relations that may exist between these numbers. 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:

$$\texttt{ProperInterval} \equiv \exists(\texttt{begin, end}).< \qquad (6)$$

Although not directly enforced on the user, this restriction on proper definitions of temporal intervals is checked by the reasoner. All intervals that do not satisfy this restriction are not considered proper intervals, which will be indicated to the user through the reasoning service following a consistency check.

D. 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 this, we seek to represent temporal aspects of entities other than timespan. In this context, the final level of expressiveness that we enable in tOWL considers different aspects of change: i) change in a concrete attribute value of an individual, ii) a change in the relationship between entities, 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 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 emphasize 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 imposes 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 2. In this example, we define two OWL classes, namely *Company* and *Product*. For each of these classes, we instantiate an individual, namely *iGoogle* and *iChrome*, respectively, representing the company Google and Chrome, the web-browser from Google. For each of these individuals, we instantiate a timeslice, namely *iGoogle_TS1* and *iChrome_TS1*, respectively, representing the static individuals over the periods *iInterval1* and *iInterval2*. Here, the two timeslices *iGoogle_TS1* and *iChrome_TS1* share the same time interval, i.e., *iInterval1* is equal to *iInterval2*, as denoted by the *towl:equal* relationship. Finally, the two timeslices are connected by the fluent *hasProduct* that indicates that over the period *iInterval1* (equivalent to the period *iInterval2*) Google Chrome is a product of Google.

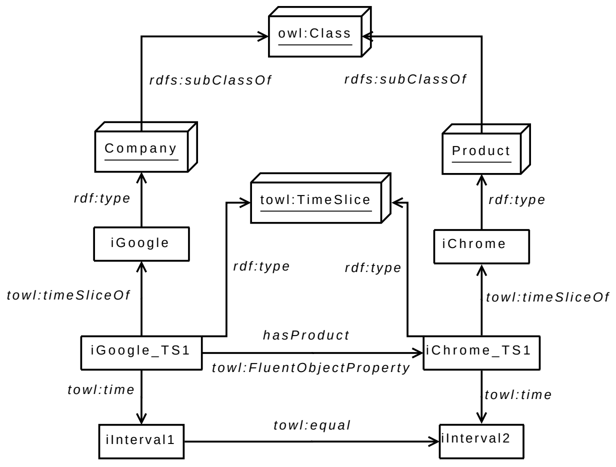

Fig. 2. Temporal restrictions on timeslices connected by fluents.

In Table II we present an overview of the tOWL TBox axioms corresponding to the timeslices/fluents layer. The tOWL axioms and facts are described in the appendix.

In Table II, the concept of `TimeSlice` is defined as all those individuals for which the `time` property is defined and takes a value of type `Interval`, and for which the `timeSliceOf` property is defined and takes a value that is not an `Interval`, a `TimeSlice`, or a `Literal`. The concept of `Interval` is defined as all those individuals for which the `start` and `end` properties are defined and take a value from XML Schema `dateTime` such that the value associated to the starting point is smaller than the value associated to the ending point. The concept of `FluentProperty` is defined as a subclass of the RDF `Property` class, and is in turn a superclass for the `FluentObjectProperty` and `FluentDatatypeProperty` constructs. The `timeSliceOf` property is defined as that property that can be applied to timeslices and that only takes values that are not timeslices, intervals, or literals. The `time` property is defined as that property that takes values only of type `Interval` and can be applied to individuals of type `TimeSlice`. The `start` and `end` properties are defined as those properties that are defined for intervals and that take values from XML Schema `dateTime`.

E. 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, is insufficient for the expressiveness introduced by the tOWL layers. 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 fairly complex cases, such as the Leveraged Buy Out example in Section IV. The reasoner is based on the algorithm described in [20], extended with a number of optimization 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 [35].

The complexity of ontology entailment in $\mathcal{SHIQ}(\mathcal{C})$ and thus also of tOWL is ExpTime-complete [5], [29], and for $\mathcal{ALC}(\mathcal{C})$ and tOWL Lite it is as well ExpTime-complete [29], [34], provided that the satisfiability in $\mathcal{C}$ (the concrete domain) can be decided in ExpTime. Additionally, the timeslices/fluents extension proposed for the tOWL language (the 4d fluents layer) is merely syntactic sugaring, and does not incur reasoning cost when regarded from the perspective of language complexity.

Rather than extending existing reasoners, the tOWL-Lite reasoner consists of a new C++ implementation containing the tableau algorithm for the unrestricted version of the $\mathcal{ALC}(\mathcal{C})$ description logic as described in [20]. 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++. The decision to implement a new reasoner from scratch has been taken due to the lack of documentation, or very poor documentation when present, of existing reasoners, thus not fostering extensions and making the choice for the design of a new reasoner necessary.

The tOWL reasoner enables different temporal inferences on tOWL knowledge bases. For example, given a time instant, we can determine what holds true at the moment in time based on the *inside* relationship between an instant and an interval. In this way, it can be determined, at any point in time, which timeslices hold true, since each timeslice has an interval associated with it. Thus, we can determine what facts are true at any moment in the knowledge base. Additionally, based on the relationships between intervals, we can, for example, determine, how intervals relate to each other in temporal terms, and thus the facts that we represent in the knowledge base. More concrete examples of reasoning in a practical application are provided in Section IV-C.

The correctness of the reasoner has been tested by using the benchmark suite proposed for Description Logics systems [36]. The test procedure consists of four categories of tests, as outlined in [36]: concept satisfiability, artificial TBox classification, realistic TBox classification, and synthetic ABox tests. The concept satisfiability tests are focused on the performance of computing satisfiability of large concept expressions without reference to a TBox. The artificial TBox classification tests investigate the performance of classifying an artificially generated TBox, while the realistic TBox classification tests perform the same investigation but on knowledge bases related to the GALEN medical terminology knowledge base [37]. Finally, the synthetic ABox tests look at the system's performance when realising a synthetic ABox.

TABLE II
tOWL AXIOMS FOR THE *4DFluents* LAYER.

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

F. RDF/XML Serialization

We present the RDF/XML serialization of the tOWL abstract syntax as a separate document available as an electronic attachment to this paper. The serialization is relevant as the RDF/XML syntax is the lingua franca of Semantic Web applications. By providing the serialization we enable different users to employ the tOWL language in their applications. These applications can export the tOWL data for reuse in interoperability scenarios.

IV. EXAMPLE APPLICATION

In this section we 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 IV-A we present LBO processes in general, and introduce the Alliance Boots LBO. In Section IV-B we illustrate how such a process can be represented in the tOWL language. We conclude this section by providing some reasoning examples for the LBO application in Section IV-C.

A different implementation of the tOWL language, next to the one presented in this paper, is described in [8]. Here, we employ the tOWL language for the representation of company market recommendations in a system that aggregates these recommendations for the generation of buy/hold/sell advices for the stock market. In this example, tOWL proves valuable in representing the different recommendations that may hold at different points in time, and overlap each other, and in determining which advices hold true at any point in time.

A. 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 the assets of the company that is to be acquired are used, partly or wholly, as collateral for the loans. This type of process is of particular interest in the current case for two reasons: i) its 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 activity diagram of an LBO process is presented in Figure 3.

An LBO process can be divided into 4 main stages:

1) Early Stage.
2) Due Diligence.
3) Bidding.
4) Acquisition.

The transition between stages is 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, the extension leads to a raise of the current bid. 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 is granted or not, 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 company. When the final bid is made and accepted, even in the case when no counter bids are made, the process moves into the *Acquisition* phase and ends.

In the following we consider a model of 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 the target company Alliance Boots.

B. The Alliance Boots LBO in tOWL

The focus of this section is to illustrate how the information regarding the LBO process can be represented in tOWL abstract syntax, both at an abstract level as well as in the particular example presented here. The main focus is on illustrating the main concepts that are relevant from a language perspective.

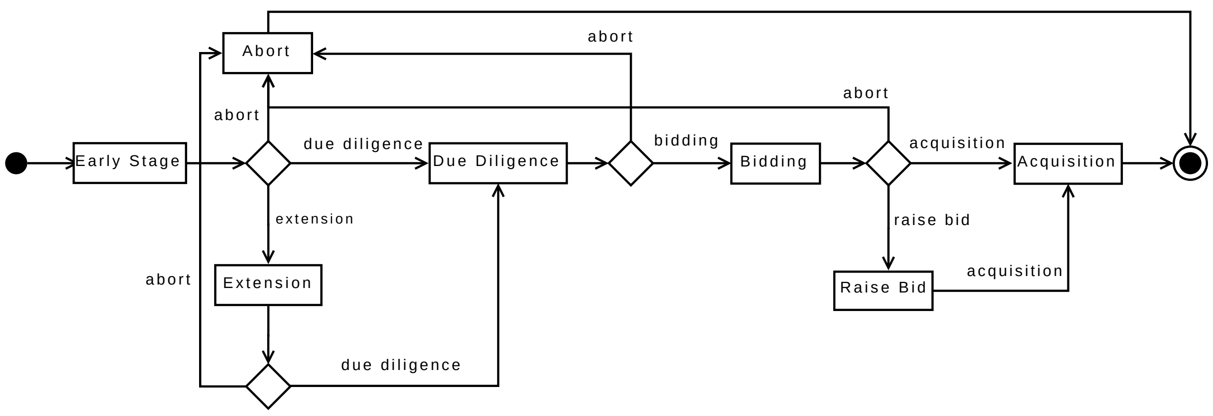

Fig. 3. Stages of an LBO process.

1) 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.

```
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.

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

All stages are pairwise disjoint, which we represent as follows.

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

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

```
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.

```
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.

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

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

```
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.

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

2) ABox: At ABox level we represent particular information that is known about the specific LBO process presented in this section. We start by instantiating the relevant individuals that are known to play a role in the LBO process. First, we represent the participating companies.

```
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 TerraFirma.

```
Individual(iLBOProcess1_TS1
   type(LBOProcess_TS)
   value(timeSliceOf iLBOProcess_1)
   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 a news message that describes the hedge fund `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 hedge fund and the target, respectively, a time interval associated to the stage, and employ the `inStage` fluent to associate the companies to the stage.

```
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))
```

In this section we have shown how a dynamic process, in the form of an LBO process, can be represented in the tOWL language. Due to the temporal expressiveness of the language, we were able to define the order of stages of the considered process, as well as the transitions that the companies involved make through this process.

C. Reasoning Examples

In this section we present an example of how reasoning can be employed in the LBO application previously described. We show how it can be inferred that a company is in a certain stage based on information on other stage transitions in which a company was involved.

The different paths that a company may follow when involved in an LBO process have been described in Figure 3. A set of axioms, as presented in Section IV-B, have been used to represent this in a tOWL knowledge base, exhaustively describing all acceptable stage sequences in an LBO process. Based on this knowledge, and in the presence of incomplete information, one can, in a certain number of cases, infer this missing information from the facts already present in the knowledge base. In this section we illustrate this by means of an example.

Assuming the existence of a news message, N_1, reporting that the company KKR, seeking to acquire Alliance Boots, has entered the Due Diligence stage of an LBO process, we can represent the following information in the tOWL knowledge base, illustrated here by means of tOWL abstract syntax.

```
Individual(t3 type(Interval))
Individual(iDueDiligence1
   type(DueDiligence_TS))
Individual(iDueDiligence1_TS1
   type(TimeSlice)
   value(timeSliceOf iDueDiligence1)
   value(time t3))
Individual(iKKR_TS3
   type(TimeSlice)
   value(timeSliceOf iKKR)
   value(time t3)
   value(inStage iDueDiligence1_TS1))
Individual(iAllianceBoots_TS3
   type(TimeSlice)
   value(timeSliceOf iAllianceBoots)
   value(time t3)
   value(inStage iDueDiligence1_TS1))
```

The representation just introduced defines a new time interval, $t3$, that is associated with timeslices of the two companies involved in the Due Diligence phase as well as with a timeslice of this stage. Additionally, we associate the timeslices of the two companies with the timeslice of the Due Diligence stage through the *inStage* fluent, thus indicating that, over interval $t3$, KKR and Alliance Boots find themselves in the Due Diligence phase.

Following the N_1 news message, another news message is issued, N_2, reporting that KKR and Alliance Boots have

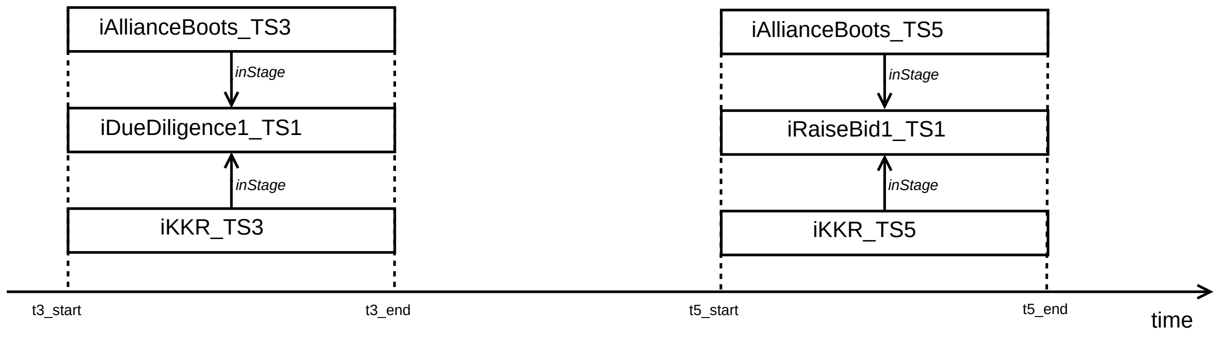

Fig. 4. Overview of explicit knowledge on LBO stages.

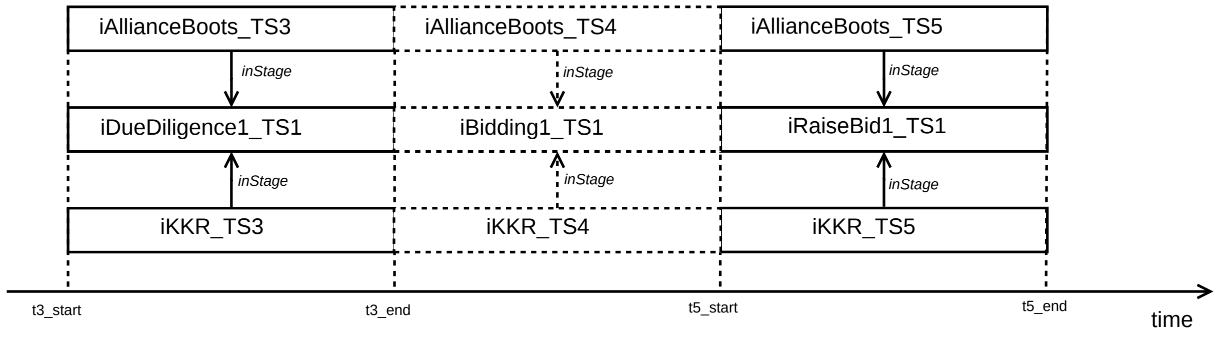

Fig. 5. Overview of explicit and implicit knowledge on LBO stages.

entered the Raise Bid stage of the LBO process, over some interval $t5$. The knowledge relating to these two stages in the tOWL knowledge base is depicted in Figure 4.

Having previously described all the possible paths through which an LBO process can be instantiated for a certain company, it is apparent that a direct transition from the Due Diligence phase to the Raise Bid phase is not possible. We derive that, between the two stages, the two companies must have transitioned the Bidding phase before moving on to Raise Bid, since no other path is possible between the two stages described in the news messages N_1 and N_2, respectively. We depict the new, relevant, snapshot of the knowledge base in Figure 5, where the inferred knowledge is represented in dotted lines.

V. Discussion

We have seen in Section IV that the tOWL language can be used to represent rather complicated processes in which temporal aspects such as time and change play an important role. The tOWL language meets shortcomings of various previous approaches, such as OWL-Time [25] and the OWL ontology for fluents [15] that only address temporality to a limited extent. The approach presented in [25], for example, only deals with the representation of time in the form of intervals and instants. However, ensuring that intervals are properly defined (e.g., 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. For example, in OWL-Time it is also not possible to enforce a particular order of state transitions in a process.

The approach in [25] is limited in the representation of temporal aspects such as change. The approach taken in [15] builds upon [25] by addressing this limitation, namely: the representation of temporal aspects such as change. However, it is limited in another sense which relates to the definition of fluent properties as being symmetric, i.e., if the pair (x,y) is the interpretation of a symmetric property, then 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 is certainly not the case that *y* is also an employee of *x*. Therefore, enforcing symmetry on fluent properties is usually too restrictive.

Building upon the approach in [15], tOWL enables differentiations between fluents that take values from the *TimeSlice* class and fluents that indicate changing values (data types). This is achieved through the use of the *FluentObjectProperty* and *FluentDatatypeProperty* properties. 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 approaches in [25] and [15].

The approach proposed in this paper can also provide a strong logical base for temporal extensions of conceptual languages. In the ER model [38], for example, the relationship between concepts can be represented as a fluent in order to denote a time-varying relationship. Adaptations of the fluent approach presented in this paper, such as the differentiation between fluents relating to objects and fluents relating to data types, can help refine existing conceptual models where time is taken into account. For example, the approach in [39], presenting the TERC+ temporal conceptual model, could incorporate such a refinement in the language specification. Additionally, the approach taken in tOWL for the representation of time could be incorporated in temporal conceptual models when the representation of processes and state transitions is envisioned.

From an application perspective, our work comes to enable temporal representations in systems where this was not previously possible, such as often the case in designs based on computational intelligence methods combined with Semantic Web approaches. Many such applications have been developed, as for example the application of a fuzzy ontology to news summarization [40], an ontology-based computational intelligent multi-agent system applied to Capability Maturity Model Integration assessment [41], and project monitoring [42]. Ontology-based approaches have also been applied in the development of systems based on computing with words approach [43]. Fuzzy concept networks and their evolution are analysed in [44], while the fuzzy matchmaking of Semantic Web services is described in [45]. A fuzzy markup language (FML) based on XML is applied in the context of an adaptive domotic framework in [46], and in the more general context of Ambient Intelligence, together with other fuzzy technologies, in [47]. tOWL can enhance these systems by providing a formalism for the representation of time and change.

VI. Conclusions

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 [15], [25] that only address this issue to a limited extent. 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 used for the representation of the different aspects of change, such as state transitions. Building on the approach presented in [15], it extends the latter by making a difference between fluents that point to data types 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 data type fluents. The tOWL language can be employed for representation and reasoning in a wide variety of dynamic domains, such as the financial one as shown in this paper.

Acknowledgments

The authors are partially supported by the EU funded IST STREP Project FP6 - 26896: Time-determined ontology-based information system for real-time 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. The authors would like to thank Alessandro Artale, Tommaso di Noia, and Kurt Sandkuhl for their valuable reviews and suggestions on parts of this paper, and the anonymous referees for their valuable comments.

Appendix
The TOWL Language Summary

We begin by providing an overview of the tOWL descriptions enabled by the language in Table III. Here, C, D are used to denote class names, R is an object property, U is a data type property, n is a positive integer, u_1, u_2 are constructs of type `towl:ConcreteFeatureChain` and p_d denotes a binary concrete domain predicate. The language description in Table III is an extension of the summary of OWL-DL constructs found in [48].

An overview of the tOWL axioms and facts that are enabled by the language is given in Table IV. This is an adaptation of the summary of the OWL-DL axioms and facts found in [48], 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 focus on the OWL-DL$^-$ language.

References

[1] T. Berners-Lee, J. Hendler, and O. Lassila, "The semantic web," *Scientific American*, vol. 284, no. 5, pp. 28–37, 2001.

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

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

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

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

[6] V. Milea, F. Frasincar, and U. Kaymak, "The tOWL web ontology language," in *The 20th Belgian-Dutch Conference on Artificial Intelligence (BNAIC 2008)*, 2008, pp. 343–344.

[7] 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)*. Springer, 2008, pp. 457–466.

[8] 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)*. IEEE Computer Society Press, 2008, pp. 65–74.

[9] 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)*. Tapir Academic Press, 2007, pp. 791–802.

[10] F. Frasincar, V. Milea, and U. Kaymak, "tOWL: Integrating time in OWL," in *Semantic Web Information Management: A Model-Based Perspective*, R. D. Virgilio, F. Giunchiglia, and L. Tanca, Eds. Springer, 2010, ch. 11, pp. 225–246.

[11] "The stanford encyclopedia of philosophy," 2003. [Online]. Available: http://plato.stanford.edu/

[12] G. Leibniz, *Philosophical papers and letters*. D. Reidel, 1969.

[13] F. Baader, D. Calvanese, D. McGuinness, P. Patel-Schneider, and D. Nardi, *The Description Logic Handbook: theory, implementation, and applications*. Cambridge Univ. Press, 2003.

[14] A. Artale and E. Franconi, "A survey of temporal extensions of description logics," *Annals of Mathematics and Artificial Intelligence*, vol. 30, no. 1, pp. 171–210, 2000.

[15] 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)*. IOS Press, 2006, pp. 226–336.

[16] A. Artale and E. Franconi, "A temporal description logic for reasoning about actions and plans," *Journal of Artificial Intelligence Research*, vol. 9, no. 2, pp. 463–506, 1998.

TABLE III
TOWL CONSTRUCTS

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($C_1 \ C_2 \ldots$)	$C_1 \sqcap C_2$	$C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}}$
unionOf($C_1 \ C_2 \ldots$)	$C_1 \sqcup C_2$	$C_1^{\mathcal{I}} \cup C_2^{\mathcal{I}}$
complementOf(C)	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
restriction(R someValuesFrom(C))	$\exists R.C$	$\{x \in \Delta^{\mathcal{I}} \mid \exists y.\langle x, y\rangle \in R^{\mathcal{I}}$ and $y \in C^{\mathcal{I}}\}$
restriction(R allValuesFrom(C))	$\forall R.C$	$\{x \in \Delta^{\mathcal{I}} \mid \forall y.\langle x, y\rangle \in R^{\mathcal{I}} \rightarrow y \in C^{\mathcal{I}}\}$
restriction(R minCardinality(n))	$\geq n \ R$	$\{x \in \Delta^{\mathcal{I}} \mid \sharp(\{y.\langle x, y\rangle \in R^{\mathcal{I}}\}) \geq n\}$
restriction(R maxCardinality(n))	$\leq n \ R$	$\{x \in \Delta^{\mathcal{I}} \mid \sharp(\{y.\langle x, y\rangle \in R^{\mathcal{I}}\}) \leq n\}$
restriction(U someValuesFrom(D))	$\exists U.D$	$\{x \in \Delta^{\mathcal{I}} \mid \exists y.\langle x, y\rangle \in U^{\mathcal{I}}$ and $y \in D^{\mathcal{D}}\}$
restriction(U allValuesFrom(D))	$\forall U.D$	$\{x \in \Delta^{\mathcal{I}} \mid \forall y.\langle x, y\rangle \in U^{\mathcal{I}}$ and $y \in D^{\mathcal{D}}\}$
restriction(U minCardinality(n))	$\geq n \ U$	$\{x \in \Delta^{\mathcal{I}} \mid \sharp(\{y.\langle x, y\rangle \in U^{\mathcal{I}}\}) \geq n\}$
restriction(U maxCardinality(n))	$\leq n \ U$	$\{x \in \Delta^{\mathcal{I}} \mid \sharp(\{y.\langle x, y\rangle \in U^{\mathcal{I}}\}) \leq n\}$
ConcreteFeatureChain($f_1 \ \ldots \ f_n \ g$)	$f_1 \circ \ldots \circ f_n \circ g$	$\{(a_1, b) \in \Delta^{\mathcal{I}} \times \Delta_{\mathcal{D}} \mid \exists! a_2 \in \Delta^{\mathcal{I}}, \ldots, \exists! a_{n+1} \in \Delta^{\mathcal{I}} \wedge$ $\wedge \ \exists! b \in \Delta_{\mathcal{D}} : (a_1, a_2) \in f_1^{\mathcal{I}}, \ldots$ $(a_n, a_{n+1}) \in f_n^{\mathcal{I}} \wedge g^{\mathcal{I}}(a_{n+1}) = b\}.$
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))	$\forall(u_1, u_2).p_d$	$\{x \in \Delta^{\mathcal{I}} \mid \forall q_1 \in \Delta_{\mathcal{D}}, \forall q_2 \in \Delta_{\mathcal{D}} : u_1^{\mathcal{I}}(x) = q_1 \wedge$ $\wedge \ u_2^{\mathcal{I}}(x) = q_2 \wedge \ (q_1, q_2) \in p_d^{\mathcal{I}})\}$

TABLE IV
TOWL AXIOMS AND FACTS

tOWL Abstract Syntax	DL Syntax	Semantics
Class(A partial $C_1 \ldots C_n$)	$A \sqsubseteq C_1 \sqcap \ldots \sqcap C_n$	$A^{\mathcal{I}} \subseteq C_1^{\mathcal{I}} \cap \ldots \cap C_n^{\mathcal{I}}$
Class(A complete $C_1 \ldots C_n$)	$A = C_1 \sqcap \ldots \sqcap C_n$	$A^{\mathcal{I}} = C_1^{\mathcal{I}} \cap \ldots \cap C_n^{\mathcal{I}}$
SubClassOf ($C_1 \ C_2$)	$C_1 \sqsubseteq C_2$	$C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$
EquivalentClasses ($C_1 \ldots C_n$)	$C_1 = \ldots = C_n$	$C_1^{\mathcal{I}} = \ldots = C_n^{\mathcal{I}}$
DisjointClasses ($C_1 \ldots C_n$)	$C_i \sqcap C_j = \bot, i \neq j$	$C_i^{\mathcal{I}} \cap C_j^{\mathcal{I}} = \{\}, i \neq j$
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($U_1 \ldots U_n$)	$U_1 = \ldots = U_n$	$U_1^{\mathcal{I}} = \ldots = U_n \mathcal{I}$
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])	$R^+ \sqsubseteq R$	$R^{\mathcal{I}} = (R^{\mathcal{I}})^+$
SubPropertyOf($R_1 \ R_2$)	$R_1 \sqsubseteq R_2$	$R_1^{\mathcal{I}} \subseteq R_2^{\mathcal{I}}$
EquivalentProperties($R_1 \ldots R_n$)	$R_1 = \ldots = R_n$	$R_1^{\mathcal{I}} = \ldots = R_n^{\mathcal{I}}$
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)	$o \in C_i$	$o^{\mathcal{I}} \in C_i^{\mathcal{I}}$
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($o_1 \ldots o_n$)	$o_1 = \ldots = o_n$	$o_1^{\mathcal{I}} = \ldots = o_n^{\mathcal{I}}$
DifferentIndividuals($o_1 \ldots o_n$)	$o_i \neq o_j, i \neq j$	$o_i^{\mathcal{I}} \neq o_j^{\mathcal{I}}, i \neq j$
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 ($U_n^{FD} \ v_n$))	$\langle o^{TS}, v_i\rangle \in U_i^{FD}$	$\langle (o^{TS})^{\mathcal{I}}, v_i^{\mathcal{I}}\rangle \in (U_i^{FD})^{\mathcal{I}}$

[17] ——, "A computational account for a description logic of time and action," in *The 4th Conference on Principles of Knowledge Representation and Reasoning (KR 1994)*. Morgan Kaufmann, 1994, pp. 3–14.

[18] 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)*. CEUR Workshop Proceedings, 2001, pp. 96–105.

[19] 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)*. Morgan Kaufmann, 1991, pp. 452–457.

[20] C. Lutz and M. Milicic, "A tableau algorithm for description logics with concrete domains and general Tboxes," *Journal of Automated Reasoning*, vol. 38, no. 1–3, pp. 227–259, 2007.

[21] C. Lutz, F. Wolter, and M. Zakharyashev, "Temporal description logics: A survey," in *15th International Symposium on Temporal Representation and Reasoning (TIME 2008)*. IEEE, 2008, pp. 3–14.

[22] C. Jensen, J. Clifford, R. Elmasri, S. Gadia, P. Hayes, and S. Jajodia, "A consensus glossary of temporal database concepts," *SIGMOD Record*, vol. 23, no. 1, pp. 52–64, 1994.

[23] G. Ozsoyoglu and R. Snodgrass, "Temporal and real-time databases: A survey," *IEEE Transactions on Knowledge and Data Engineering*, vol. 7, no. 4, pp. 513–532, 1995.

[24] C. Gutierrez, C. Hurtado, and A. Vaisman, "Introducing time into RDF," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 2, pp. 207–218, 2007.

[25] J. Hobbs and F. Pan, "An ontology of time for the semantic web," *ACM Transactions on Asian Language Information Processing*, vol. 3, no. 1, pp. 66–85, 2004.

[26] S. Batsakis and E. Petrakis, "SOWL: spatio-temporal representation, reasoning and querying over the semantic web," in *6th International Conference on Semantic Systems*. ACM, 2010.

[27] ——, "Representing temporal knowledge in the semantic web: The extended 4d fluents approach," *Combinations of Intelligent Methods and Applications*, pp. 55–69, 2011.

[28] B. Motik, "Representing and querying validity time in rdf and owl: a logic-based approach," *International Semantic Web Conference (ISWC 2010)*, pp. 550–565, 2010.

[29] C. Lutz, "Description logics with concrete domains–a survey," in *Advances in Modal Logics*, vol. 4. King's College Publications, 2003, pp. 265–296.

[30] J. Allen, "Maintaining knowledge about temporal intervals," *Communications of the ACM*, vol. 26, no. 11, pp. 832–843, 1983.

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

[32] P. Hayes and B. McBride, "RDF semantics," 2004, W3C Recommendation.

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

[34] C. Lutz, "Interval-based temporal reasoning with general tboxes," LuFG Theoretical Computer Science, RWTH Aachen, LTCS-Report LTCS-00-06, 2000.

[35] D. Maggiore, "Deliverable 4.3: Design of a TOWL temporal reasoner," TOWL Project, Technical Report, 2008.

[36] I. Horrocks and P. Patel-Schneider, "DL systems comparison," in *The 1998 Description Logics Workshop (DL 1998)*, ser. CEUR-WS, vol. 11, 1998, pp. 55–57.

[37] I. Horrocks, "The FaCT system," *Automated Reasoning with Analytic Tableaux and Related Methods*, pp. 307–312, 1998.

[38] P. P. Chen, "The entity-relationship model–toward a unified view of data," *ACM Transactions on Database Systems*, vol. 1, no. 1, pp. 9–36, 1976.

[39] E. Zimanyi, C. Parent, S. Spaccapietra, and A. Pirotte, "TERC+: a temporal conceptual model," in *International Symposium on Digital Media Information Base*, 1997.

[40] C. Lee, Z. Jian, and L. Huang, "A fuzzy ontology and its application to news summarization," *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 35, no. 5, pp. 859–880, 2005.

[41] C. Lee and M. Wang, "Ontology-based computational intelligent multi-agent and its application to CMMI assessment," *Applied Intelligence*, vol. 30, no. 3, pp. 203–219, 2009.

[42] C. Lee, M. Wang, and J. Chen, "Ontology-based intelligent decision support agent for CMMI project monitoring and control," *International Journal of Approximate Reasoning*, vol. 48, no. 1, pp. 62–76, 2008.

[43] R. Marek and C. Ly, "Ontological Approach To Development Of Computing With Words Based Systems," *International Journal of Approximate Reasoning*, vol. 50, no. 1, pp. 72–91, 2009.

[44] S. Calegari and F. Farina, "Fuzzy ontologies and scale-free networks analysis," *International Journal of Computer Science & Applications*, vol. 4, no. 2, pp. 125–144, 2007.

[45] G. Fenza, V. Loia, and S. Senatore, "A hybrid approach to semantic web services matchmaking," *International Journal of Approximate Reasoning*, vol. 48, no. 3, pp. 808–828, 2008.

[46] G. Acampora and V. Loia, "Fuzzy control interoperability and scalability for adaptive domotic framework," *IEEE Transactions on Industrial Informatics*, vol. 1, no. 2, pp. 97–111, 2005.

[47] ——, "Using FML and fuzzy technology in adaptive ambient intelligence environments," *International Journal of Computational Intelligence Research*, vol. 1, no. 2, pp. 171–182, 2005.

[48] 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*, vol. 1, no. 1, pp. 7–26, 2003.

Viorel Milea obtained the MSc degree in Informatics & Economics from Erasmus University Rotterdam, the Netherlands, in 2006. Currently, he is working towards his PhD degree at the Erasmus University Rotterdam, the Netherlands. The focus of his PhD is on employing Semantic Web technologies for enhancing the current state-of-the-art in automated trading with a focus on processing 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 master 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 PhD degree in computer science from Eindhoven University of Technology, the Netherlands, in 2005. Since 2005, he is assistant professor in 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 and holds the chair of information systems in the healthcare at the School of Industrial Engineering, Eindhoven University of Technology, the Netherlands. Prof. Kaymak is a member of the editorial board of several international journals, such as Fuzzy Sets and Systems, and Soft Computing.