Temporally Enhanced Ontologies in OWL: A Shared Conceptual Model and Reference Implementation

Sven de Ridder   Flavius Frasincar
Erasmus University Rotterdam, the Netherlands
HERE, a Nokia company – Berlin, Germany
sven@svenr.org
frasincar@ese.eur.nl

Outline

Temporal ontologies

Ontologies tend to be *synchronic* (i.e., only represent the domain at a particular instant in time).

- Incorrect or incomplete data.
  "Sam Palmisano was CEO of IBM from October 2000 until January 2012."

Temporally-enhanced data is crucial in many application domains, such as:

- Business and finance
  - Recording financial news events, stock prices
  - Recognizing trends and making predictions
- Biology and medicine
  - Recording patient history
  - Describing disease progression and treatment (both general and patient-specific)
- Computing and telecommunications
  - Service leases and logs

Approaches to temporal ontologies

- Temporal RDF [Gutierrez et al., 2007]
  - But: uses RDF reification, incompatible with OWL DL
- Ontology versioning [Klein and Fensel, 2001]
  - But: data redundancy
  - May still be used to model *transaction-time* instead of *valid-time*
- Extending DL with valid-time [Artale and Franconi, 2000, Artale and Lutz, 2004]
  - But: based on $\mathcal{ALC}$
  - Undecidable with $\mathcal{SHOIN}(\mathcal{D})$ (OWL DL) or $\mathcal{SROIQ}(\mathcal{D})$ (OWL 2) [Artale and Lutz, 2004]
- Representations within OWL DL
  - Reification/n-ary relations approach [Noy et al., 2006]
  - 4D Fluents approach [Welty et al., 2006]

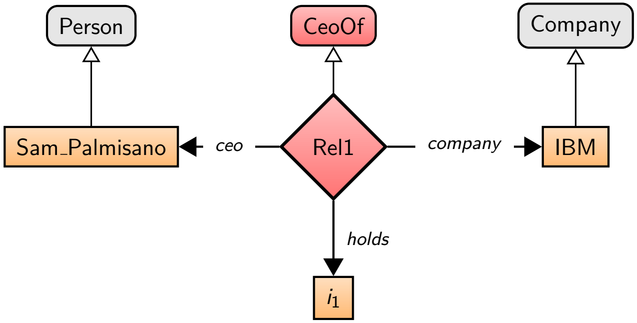
Focus on representations within OWL DL:

- Maximally possible expressiveness while computationally complete and decidable.

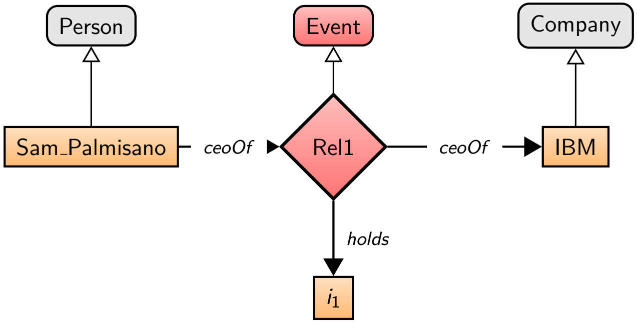
Reification

- General problem: expression of *n-ary* relations.
- Specific problem: relations involving a subject, and object, and a *time interval* (ternary relation).
- Approach: *reification*
    - Convert property assertion into an entity (*reify* the property assertion);
    - Relate subject, object, and time interval instances to reified property assertion.

Reification (cont'd)



Reification (cont'd)



Reification (cont'd)

Drawbacks:

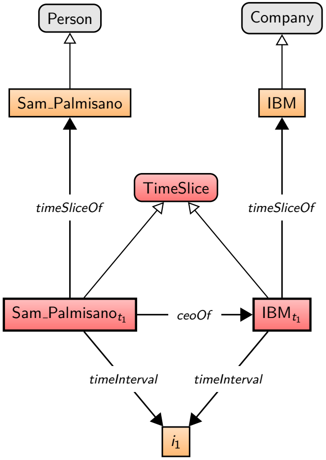
- Unintuitive.
- Proliferation of “strange” entities.
- Less expressive than regular properties (e.g., inverse, transitivity).
- Limited reasoning support.

4D Fluents

Approach: consider *timeslices*.

- A timeslice represents a *temporal facet* of an entity as it "occupies" some interval in time.
- Temporal relations are relations between timeslices.
- Timeslices involved in a relation must be *compatible* with each other; i.e., they must occupy the exact same interval in time.

4D Fluents (cont'd)



4D Fluents (cont'd)

Drawbacks:

- Unintuitive: “space-time worms”.
- In the general case, worse proliferation of objects than the reification approach.

Research objectives

What is a common conceptual model for temporal OWL ontologies?

- Which concepts should the model introduce, and how are these concepts defined?
- Which existing representation schemes can be converted to and from the conceptual model?
- What are some of the applications of the conceptual model?

Temporal conceptual model: Time layer

Represents the *time domain*: time instants and time intervals, and their relations to each other and to the time axis.

- Orthogonal to the non-temporal ontology.
- Different time models are interchangeable, e.g., simple timestamps, OWL-Time [Hobbs and Pan, 2004].

Temporal primitives:

- `TimeInstant`

```
TimeDeclaration(TimeInstant(_:t1))
TimeInstantRelationAssertion(_:t1 _:t2 <)
InstantTimeAssertion(_:t1
  "2013-06-05T18:42:23.000+02:00"^^xsd:dateTime)
```

- `TimeInterval`

```
TimeDeclaration(TimeInterval(period2013Q1))
IntervalStartAssertion(period2013Q1 _:t1)
IntervalEndAssertion(period2013Q1 _:t2)
TimeIntervalRelationAssertion(_:i1 _:i2 m)
TimeIntervalRelationAssertion(_:i1 _:i3 di)
```

Temporal conceptual model: Fluents layer

Builds on the time layer and the non-temporal ontology.

- Represents fluents, properties (relations and attributes) that change with time.

Fluent properties:

- Declarations

```
FluentsDeclaration(FluentObjectProperty(ceoOf))
FluentsDeclaration(FluentDataProperty(hasTitle))
FluentDataPropertyDomain(hasTitle Person)
FluentDataPropertyRange(hasTitle DataOneOf(
  "Mr."^^xsd:string "Mrs."^^xsd:string "Ms."^^xsd:string))
```

- `Assertions`

```
FluentObjectPropertyAssertion(sam ceoOf ibm _:i1)
FluentDatatypePropertyAssertion(mary hasTitle
  "Ms."^^xsd:string _:i2)
```

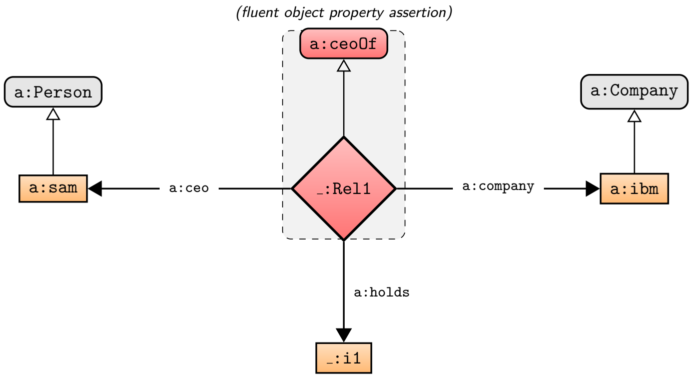
Reference implementation: Kala (Introduction)

- Current focus:
  - Representation of entities and axioms as first-class objects.
  - Storage and retrieval of entities and axioms to/from a temporal ontology.
  - Parsing of existing temporal ontologies in customizable ways.
  - Serialization to customizable representation schemes.
- Developed as a Java library.
- Based on **OWL API** [Horridge and Bechhofer, 2009]
  - *Decorates* core classes (e.g., `Ontology` $\rightarrow$ `TemporalOntology`).
  - Provides temporal primitives (e.g., `TimeInterval`) and temporal axioms (e.g., `DiscreteTimeAxiom`, `IntervalRelationAxiom`).
  - Provides fluent properties and fluent property axioms (e.g., `FluentObjectPropertyDomainAxiom` and `FluentObjectPropertyAssertionAxiom`).
- Open source, under development:
  - Try it out or join the project: `http://www.github.com/owl-kala/owl-kala`

Reference implementation: Kala (Serialization)

- Allow serialization from/to reification and 4D-fluents representations.
  - Customization through configuration objects (e.g., class and property URIs, direction of properties).

Serialization (reification)



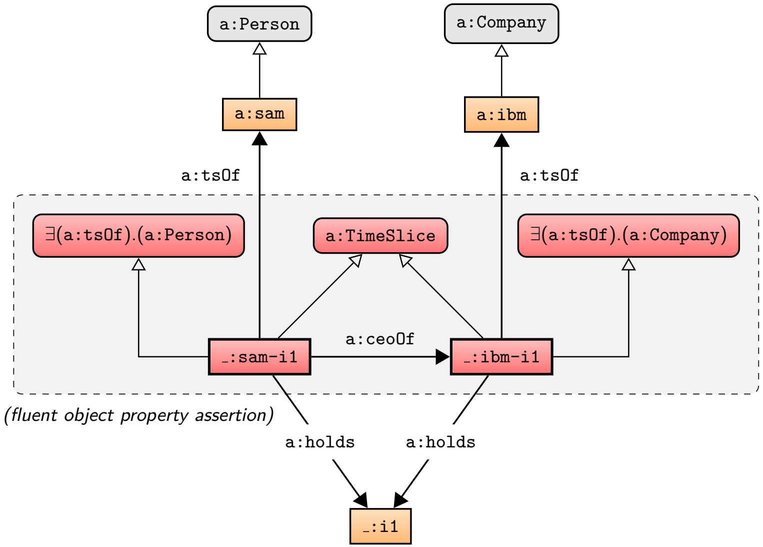
read_reif

```
function READ_REIF($m$, $fluents$)
    $r \leftarrow \emptyset$
    $t \leftarrow \emptyset$
    for all $f \in fluents$ do
        $r_s, r_o \leftarrow$ REIF_CONFIGURATION($f$)
        $D \leftarrow$ RANGE($m, r_s$)
        $R \leftarrow$ RANGE($m, r_o$)
        $f \leftarrow$ FLUENT_PROPERTY($f : D \rightarrow R$)
        $r \leftarrow r \cup f$
        for all $a \in$ INSTANCES($m, f$) do
            $s \leftarrow$ VALUE($m, a.r_s$)
            $o \leftarrow$ VALUE($m, a.r_o$)
            $i \leftarrow$ VALUE($m, a.holds$)
            $a_f \leftarrow$ FLUENT_PROPERTY_ASSERTION($s, f, o, i$)
            $r \leftarrow r \cup a_f$
        end for
        $t \leftarrow t \cup$ TRANSLATED($m, f$)
    end for
    return $r \cup m - t$
end function
```

write_reif

```
1: function $\textsc{write\_reif}(m)$
2:     $r \leftarrow \textsc{nontemporal}(m)$
3:     for all $f[f_D, f_R] \in \textsc{fluent\_properties}(m)$ do
4:         $r_s, r_o \leftarrow \textsc{reif\_configuration}(f)$
5:         $f_c \leftarrow \textsc{as\_class}(f)$
6:         $r \leftarrow r \cup f_c$
7:         $r \leftarrow r \cup \textsc{functional\_property}(r_s : f_c \rightarrow f_D)$
8:         $r \leftarrow r \cup \textsc{functional\_property}(r_o : f_c \rightarrow f_R)$
9:         $r \leftarrow r \cup \textsc{subclass}(f_c \sqsubseteq \exists r_s.f_D \sqcap \exists r_o.f_R \sqcap \exists holds.Interval)$
10:        for all $a[s, o, i] \in f$ do
11:            $ai \leftarrow \textsc{new\_individual\_of}(f_c)$
12:            $r \leftarrow r \cup ai$
13:            $r \leftarrow r \cup \textsc{assert}(r_s(ai\ \ s); r_o(ai\ \ o); holds(ai\ \ i))$
14:            $r \leftarrow r \cup i$
15:        end for
16:    end for
17:    return $r$
18: end function
```

Serialization (4D Fluents)



read_4df

```
function READ_4DF($m$)
    $r \leftarrow t \leftarrow \emptyset$
    for all $ts[s, i] \in$ TIMESLICES($m$) do
        for all $p(ts, o) \in$ PROPERTY_ASSERTIONS($m, ts$) do
            if $p(ts, o) = tsOf(ts, s)$ or $p(ts, o) = holds(ts, i)$ then
                skip
            end if
            if $p \notin r$ then
                $D \leftarrow$ DOMAIN_TS_RESTRICTION($m, p$)
                if IS_OBJECT_PROPERTY($p$) then
                    $R \leftarrow$ RANGE_TS_RESTRICTION($m, p$)
                else
                    $R \leftarrow$ RANGE($m, p$)
                end if
                $r \leftarrow r \cup$ FLUENT_PROPERTY($p : D \rightarrow R$)
            end if
            if IS_OBJECT_PROPERTY($p$) then
                $o \leftarrow$ VALUE($m, o.tsOf$)
            end if
            $r \leftarrow r \cup$ FLUENT_PROPERTY_ASSERTION($s, p, o, i$)
        end for
        $t \leftarrow t \cup$ TRANSLATED($m, ts$)
    end for
    return $r \cup m - t$
end function
```

write_4df

```
function WRITE_4DF($m$)
    $r \leftarrow$ NONTEMPORAL($m$)
    for all $f[f_D, f_R] \in$ FLUENT_PROPERTIES($m$) do
        if IS_OBJECT_PROPERTY($f$) then
            $p \leftarrow$ PROPERTY($f : \exists tsOf.f_D \rightarrow \exists tsOf.f_R$)
        else
            $p \leftarrow$ PROPERTY($f : \exists tsOf.f_D \rightarrow f_R$)
        end if
        $r \leftarrow r \cup p$
        for all $a[s, o, i] \in f$ do
            $ts_s \leftarrow$ TIMESLICE($s, i$)
            $r \leftarrow r \cup ts_s$
            if IS_OBJECT_PROPERTY($f$) then
                $ts_o \leftarrow$ TIMESLICE($o, i$)
                $r \leftarrow r \cup ts_o \cup$ ASSERT($p(ts_s, ts_o)$)
            else
                $r \leftarrow r \cup$ ASSERT($p(ts_s, o)$)
            end if
            $r \leftarrow r \cup i$
        end for
    end for
    return $r$
end function
```

Evaluation: Alliance Boots GmbH LBO

Scenario from tOWL evaluation [Milea et al., 2012]:

- Leveraged Buy-out bidding process.
- Largest LBO in European business history.
- Target: Alliance Boots GmbH (pharmaceuticals/retail)
- Bidding performed by two hedge funds:
    - Kohlberg Kravis Roberts & Co (KKR)
    - Terra Firma
- Data obtained through analysis of financial news messages (source: `http://www.marketwatch.com`)

Evaluation: Results

Kala allowed the expression of any assertional, temporally qualified information relevant to a specific entity, such as:

- LBO Early Stage #1 starts LBO Process #1
- Terra Firma enters the Early Stage of the bidding process on Sunday, March 25, 2007, at 8:42 EDT.

It did not allow (as expected) the expression of generic *temporal constraints*, such as:

- An LBO Process is always started by its associated Early Stage.

Apart from concepts that could not be expressed, the serialization of the conceptual model to tOWL was semantically equivalent to the hand-crafted ontology from the original article.

Future work: temporal conceptual model

Possible avenues for extensions of the temporal conceptual model:

- Additional fluent property semantics:
  - Inverses
  - Transitivity
  - Functional fluent properties
  - Cardinality constraints
- Negative fluent property assertions.
- Lifetime timestamping of entities.
- Class membership/transition timestamping (i.e., *becomes-a*).
- Generation relationships (*produces*).
- Evolution constraints:
  - Fluent properties
  - Entity lifetimes

Future work: Kala

Possible avenues for applications based on Kala:

- A *snapshot reducer* for converting any temporal ontology to a non-temporal ontology that represents the temporal ontology at a particular instant in time.
- Protégé plug-in for viewing and editing temporal ontologies.
- Reasoning based on time model:
  - Advanced queries (supported by a query language).
  - Detection of inconsistencies.
  - Knowledge discovery.
- Temporal visualizations.

Questions

Thank you for listening!
Questions/comments?

References and further reading I

Artale, A. and Franconi, E. (2000).
A survey of temporal extensions of description logics.
Annals of Mathematics and Artificial Intelligence, 30(1-4):171–210.

Artale, A. and Lutz, C. (2004).
A correspondence between temporal description logics.
Journal of Applied Non-Classical Logics, 14(1-2):209–233.

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

Hobbs, J. R. and Pan, F. (2004).
An ontology of time for the Semantic Web.
Transactions on Asian Language Information Processing, 3(1):66–85.

References and further reading II

Horridge, M. and Bechhofer, S. (2009).
The OWL API: A Java API for working with OWL 2 ontologies.
In *5th International Workshop on OWL: Experiences and Directions (OWLED 2009)*.
http://ceur-ws.org/Vol-529/owled2009_submission_29.pdf.

Klein, M. and Fensel, D. (2001).
Ontology versioning on the Semantic Web.
In *International Semantic Web Working Symposium (SWWS 2001)*, pages 75–91, Berlin. Springer-Verlag.

Milea, V., Frasincar, F., and Kaymak, U. (2012).
tOWL: A temporal web ontology language.
IEEE Transactions on Systems, Man and Cybernetics. Part B, Cybernetics (IEEE T-SMC-Part B), 42(1):268–281.

Noy, N., Rector, A., Hayes, P., and Welty, C. (2006).
Defining n-ary relations on the Semantic Web.
W3C Working Group Note, April 2006,
http://www.w3.org/TR/swbp-n-aryRelations/.

References and further reading III

Welty, C., Fikes, R., and Makarios, S. (2006).
A reusable ontology for fluents in OWL.
In Bennett, B. and Fellbaum, C., editors, *4th International Conference on Formal Ontology in Information Systems (FOIS 2006)*, pages 226–236. IOS Press, Amsterdam.