An OWL Ontology for Biographical Knowledge. Representing Time-Dependent Factual Knowledge

Hans-Ulrich Krieger and Thierry Declerck

German Research Center for Artificial Intelligence (DFKI), Universität des Saarlandes, Allgemeine Linguistik
krieger@dfki.de, declerck@dfki.de

Abstract

Representing time-dependent information has become increasingly important for reasoning and querying services defined on top of RDF and OWL. In particular, addressing this task properly is vital for practical applications such as modern biographical information systems, but also for the Semantic Web/Web 2.0/Social Web in general. Extending binary relation instances with temporal information often translates into a massive proliferation of useless container objects when trying to keep the underlying RDF model. In this paper, we argue for directly extending RDF triples with further arguments in order to easily represent time-dependent factual knowledge and to allow for practical forms of reasoning. We also report on a freely available lightweight OWL ontology for representing biographical knowledge that models entities of interest via a tri-partite structure of the pairwise disjoint classes Abstract, Object, and Happening. Even though the ontology was manually developed utilizing the *Protégé* ontology editor, and thus sticking to the triple model of RDF, the meta-modelling facilities allowed us to cross-classify all properties as being either synchronic or diachronic. When viewing the temporal arguments as "extra" arguments that only apply to relation instances, universal biographical knowledge from the ontology can still be described as if there is *no* time.

Keywords: OWL biography ontology, representation of time-dependent information, practical temporal reasoning.

1 Synchronic and Diachronic Relations

Linguistics and philosophy make a distinction between synchronic and diachronic relations in order to characterize statements whose truth values do or do not change over time. *Synchronic* relations, such as dateOfBirth, are relations whose instances stay constant over time, thus there is no direct need to attach a temporal extent to them. Consider, e.g., the natural language sentence:

Tony Blair was born on May 6, 1953.

Assuming a RDF-based N-triple representation (Carothers and Seaborne, 2014), an information extraction system might yield the following set of triples:

```
tb rdf:type Person
tb hasName "Tony Blair"
tb dateOfBirth "1953-05-06"^^xsd:date
```

Since there is only one unique date of birth, this works perfectly well and properly capture the intended meaning.

Diachronic relationships, however, vary with time, i.e., their truth value do change over time. Representation frameworks such as OWL that are geared towards unary and binary relations can not be extended directly by further (temporal) arguments. Consider the following biographical information:

Christopher Gent was Vodafone's chairman until July 2003. Later, Chris became the chairman of GlaxoSmithKline with effect from January 1st, 2005.

From these two sentences, the information extraction system might discover the following underspecified time-dependent facts:

```
cg isChairman vf @ [????-??-??, 2003-07-??]
cg isChairman gsk @ [2005-01-01, ????-??-??]
```

Applying the synchronic representation schema for dateOfBirth from above would give us:

```
cg isChairman vf
cg holdsAt [????-??-??, 2003-07-??]
cg isChairman gsk
cg holdsAt [2005-01-01, ????-??-??]
```

However, the association between the original statements and their temporal extents get lost in the resulting RDF graph:

cg isChairman vf @ [????-??-??, 2003-07-??]
~~cg isChairman vf @ [2005-01-01, ????-??-??]~~
~~cg isChairman gsk @ [????-??-??, 2003-07-??]~~
cg isChairman gsk @ [2005-01-01,????-??-??]

as the second and third association are not supported by the above natural language quotation.

2 Approaches for Representing Time-Dependent Knowledge

Several well-known proposals have been presented in the literature in order to equip (binary) relation instances with time or other kinds of information. The individual rewriting schemas are depicted in Figure 1; see Welty and Fikes (2006) and Krieger (2014) for a closer overview:

1. directly equip the relation instance with additional/temporal arguments (Krieger, 2012);
2. apply a meta-logical predicate as used in the situation calculus (McCarthy and Hayes, 1969);
3. reify the original relation à la RDF, turning the property into a class (Manola and Miller, 2004);
4. employ a fact identifier à la YAGO, implicitly leading to quads (Hoffart et al., 2011);
5. wrap the range arguments in an object, called N-ary relation encoding by W3C (Hayes and Welty, 2006);
6. encode a perdurantist/4D view in OWL (Welty and Fikes, 2006);

approach	**rewriting schema**
1	$\mathit{marriedTo}(p,p') \longmapsto \mathit{marriedTo}(p,p',\underline{s,e})$
2	$\underline{\mathit{holds}}(\mathit{marriedTo}(p,p'),\underline{t}) \longmapsto \exists f\,.\,\mathit{holds}(f,t) \wedge$ $\mathit{type}(f,\mathit{Fluent}) \wedge \mathit{subject}(f,p) \wedge \mathit{predicate}(f,\mathit{marriedTo}) \wedge \mathit{object}(f,p')$
3	$\underline{\mathit{marriedTo}(p,p',s,e)} \longmapsto \exists e\,.\,\mathit{type}(e,\mathit{MarriedToEvent}) \wedge$ $\mathit{person1}(e,p) \wedge \mathit{person2}(e,p') \wedge \mathit{starts}(e,s) \wedge \mathit{ends}(e,e)$
4	$\underline{\mathit{marriedTo}(p,p',s,e)} \longmapsto \exists i\,.\,i := \mathit{marriedTo}(p,p') \wedge \mathit{starts}(i,s) \wedge \mathit{ends}(i,e)$
5	$\mathit{marriedTo}(p,\underline{p',s,e}) \longmapsto \exists o\,.\,\mathit{marriedTo}(p,o) \wedge$ $\mathit{type}(o,\mathit{PersonTime}) \wedge \mathit{person}(o,p') \wedge \mathit{starts}(o,s) \wedge \mathit{ends}(o,e)$
6	$\underline{\mathit{marriedTo}(p,p',s,e)} \longmapsto \exists t,t'\,.\,\mathit{marriedTo}(t,t') \wedge$ $\mathit{type}(t,\mathit{TimeSlice}) \wedge \mathit{hasTimeSlice}(p,t) \wedge \mathit{type}(t',\mathit{TimeSlice}) \wedge \mathit{hasTimeSlice}(p',t') \wedge$ $\mathit{starts}(t,s) \wedge \mathit{ends}(t,e) \wedge \mathit{starts}(t',s) \wedge \mathit{ends}(t',e)$
7	$\underline{\mathit{marriedTo}(p,p',s,e)} \longmapsto \exists t,t'\,.\,\mathit{marriedTo}(t,t') \wedge$ $\mathit{type}(t,\mathit{Person}) \wedge \mathit{hasTimeSlice}(p,t) \wedge \mathit{type}(t',\mathit{Person}) \wedge \mathit{hasTimeSlice}(p',t') \wedge$ $\mathit{starts}(p,s) \wedge \mathit{ends}(p,e) \wedge \mathit{starts}(p',s) \wedge \mathit{ends}(p',e)$
8	$\mathit{marriedTo}(p,p',\underline{s,e}) \longmapsto \mathit{marriedTo\_s\_e}(p,p') \wedge$ $\mathit{marriedTo\_s\_e} \sqsubseteq \mathit{marriedTo} \wedge \mathit{starts}(\mathit{marriedTo\_s\_e},s) \wedge \mathit{ends}(\mathit{marriedTo\_s\_e},e)$
9	$\mathit{marriedTo}(p,p',\underline{s,e}) \longmapsto \mathit{marriedTo}(p,p') \wedge \mathit{starts}(p',s) \wedge \mathit{ends}(p',e)$

Figure 1: Different ways of representing the atemporal statement (the "fluent") marriedTo(p,p') between two people p and p', being true for the time period $t = [s,e]$. "$\longmapsto$" should be read as *rewrite to*. The last representation schema only works if the original property (here: marriedTo) is inverse functional for all relation instances (which needs not to be the case).

7. interpret the original entities as time slices (Krieger, 2008);
8. encode the temporal extent through new synthetic properties (Gangemi, 2011);
9. use relation composition applied to the second argument which does not work in general, but only if original relation is inverse functional.

2.1 Discussion

The above approaches are in a certain sense semantically equivalent in that we can rewrite one approach to another one without losing any information. It is worth noting that all approaches invalidate standard OWL reasoning, even though they can be implemented within the RDF framework, and thus at least explicitly stated information can be queried by, e.g., SPARQL engines. Nevertheless, the non-temporal entailment rules for RDFS (Hayes, 2004) and OWL Horst/OWL 2 RL (ter Horst, 2005; Motik et al., 2012) can be adjusted, so that rule-based reasoners that go beyond symbol matching, such as *Jena* (Reynolds, 2006) or *HFC* (Krieger, 2013), are still able to perform extended entailments under these new encoding schemas.

Most of the above approaches require to rewrite the original ontology, sometimes by turning relations into classes. With the exception of approach **1**, all approaches require to introduce one or even two brand-new individuals per time-dependent fact (see Figure 1). As a consequence, reasoning and querying with such representations is extremely complex, expensive, and error-prone. Furthermore, the representation schemas **2–7** bear the potential of a non-terminating closure computation in case the newly introduced individuals are viewed as existentially quantified, i.e., anonymous logic variables (RDF: blank nodes). Luckily, this last danger can often be avoided by generating unique URI names that are deterministically generated from their "parts" (i.e., from information that is accessible through properties from the new individual)—this "trick" reminds us of constructing perfect hash functions over complex objects, as known from computer science.

Approach **1** is pursued in the temporal database community under the heading *valid time* (Snodgrass, 2000). The measurements in Krieger (2012) and Krieger (2014) have shown that this approach easily outperforms all other approaches during querying and reasoning (computation of the deductive closure) in the time domain by *several orders of magnitude*. In some cases, this divergency can make a difference between *doable* and *intractable* applications. Consequently, we think the time now is ripe for allowing n-ary relations, or as Schmolze (1989) once put it in the early days of KL-ONE *"... the advantages for allowing direct representation of n-ary relations far outweigh the reasons for the restriction."*

2.2 Tuples vs. Triples: Representation & Reasoning

We would like to make our preference towards a direct representation of additional (temporal) arguments more clear by looking at concrete examples. Consider the Wikipedia entry for Tony Blair which says he married Cherie Booth on 29th March 1980 (today = 2015-05-08), leading to the *quintuple* representation:

```
tony_blair marriedTo cherie_booth
   "1980-03-29"^^xsd:date "2015-05-08"^^xsd:date
```

A *meaning-preserving triple representation* which adheres to a W3C best practice recommendation, called *N-ary relation encoding* (see rewrite schema **5** in Figure 1) would instead result in five triples, a new individual _:ppt, a new type ValuePlusTime, and the three "accessor" properties hasValue, starts, and ends:

```
tony_blair marriedTo _:ppt
```

```
_:ppt rdf:type nary:ValuePlusTime
_:ppt nary:hasValue cherie_booth
_:ppt nary:starts ”1980-03-29”^^xsd:date
_:ppt nary:ends ”2015-05-08”^^xsd:date
```

Such a representation has a three times larger memory footprint, a slightly more complex structure, and is a bit harder to read. However, as indicated above, the new individual (in our example blank node _:ppt) might turn out to be problematic during entailment reasoning (no longer guaranteed to terminate).

Now let us focus not only on the representation of (static) knowledge, but on the (dynamic) derivation of new knowledge through entailment rules in order to see how much worse a (recommended) triple representation becomes. Consider the following entailment schema for functional diachronic *datatype* properties (in Section 3.3, we will look at the corresponding entailment schema for functional diachronic *object* properties). The original non-temporal schema looks like this (we use the rule syntax of *HFC* (Krieger, 2013) in the examples below):

```
?p rdf:type owl:FunctionalProperty
?p rdf:type owl:DatatypeProperty
?x ?p ?y
?x ?p ?z
→
?x rdf:type owl:Nothing
@test
?y != ?z
```

Such a rule schema is useful, e.g., for detecting contradictory birth dates for one and the same person (famous example: *Louis Armstrong*; right: August 4, 1901, wrongly claimed by him: July 4, 1900). Such a schema matches, for instance,

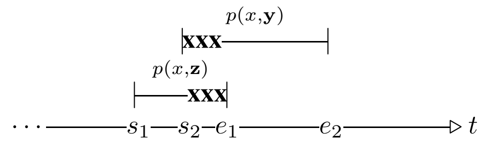
```
louis_armstrong dateOfBirth ”1901-08-04”^^xsd:date
louis_armstrong dateOfBirth ”1900-07-04”^^xsd:date
```

and binds louis_armstrong to ?x, dateOfBirth to ?p, ”1901-08-04”^^xsd:date to ?y, and ”1900-07-04”^^xsd:date to ?z. Having found problematic cases is signaled by assigning the “bottom” type owl:Nothing to the subject element of the triple bound to the logical variable ?x (= Louis Armstrong) on the right hand side of the rule.

Adding time to this rule schema makes it applicable to other functional relations such as hasSalary which do change over time, as indicated by the property characteristics time:DiachronicProperty in the rule below. Extending the rule schema is quite easy by equipping the fourth and fifth left hand side clauses with a temporal extent (things that have been added are underlined):

```
?p rdf:type owl:FunctionalProperty
?p rdf:type time:DiachronicProperty
?p rdf:type owl:DatatypeProperty
?x ?p ?y ?s1 ?e1
?x ?p ?z ?s2 ?e2
→
?x rdf:type owl:Nothing ?s ?e
@test
?y != ?z
IntersectionNotEmpty ?s1 ?e1 ?s2 ?e2
@action
?s = Max2 ?s1 ?s2
?e = Min2 ?e1 ?e2
```

The additional left hand side four-place relation IntersectionNotEmpty from the @test section of the rule simply checks whether the two temporal intervals $[s_1, e_1]$ and $[s_2, e_2]$ have a non-empty intersection, indicated by **xxx** below:



If this is the case, we mark the subject bound to ?x being of type owl:Nothing (same as for the original rule), but this type assignment now only holds for the overlapping observation time, given by the maximum of the starting times (= ?s) and the minimum of the ending times (= ?e), as computed in the @action section of the rule.

The above natural extension of the non-temporal rule, however, turns into an awfully looking and terribly inefficient rule when being couched in a triple-based setting:

```
?p rdf:type owl:FunctionalProperty
?p rdf:type time:DiachronicProperty
?p rdf:type owl:DatatypeProperty
?x ?p ?blank1
?blank1 rdf:type nary:ValuePlusTime
?blank1 nary:hasValue ?y
?blank1 nary:starts ?start1
?blank1 nary:ends ?end1
?x ?p ?blank2
?blank2 rdf:type nary:ValuePlusTime
?blank2 nary:hasValue ?z
?blank2 nary:starts ?start2
?blank2 nary:ends ?end2
→
?x rdf:type ?new
?new rdf:type nary:ValuePlusTime
?new nary:hasValue owl:Nothing
?new nary:starts ?start
?new nary:ends ?end
@test
?y != ?z
IntersectionNotEmpty ?start1 ?end1 ?start2 ?end2
@action
?start = Max2 ?start1 ?start2
?end = Min2 ?end1 ?end2
?new = MakeUri owl:Nothing ?start ?end
```

Note how the relevant input information is hidden in the two container individuals bound to ?blank1 and ?blank2 and how the output is wrapped in a brand-new individual ?new, generated by MakeUri from the @action section.

2.3 Limitations

Several points are worth mentioning here. Firstly, we are not dealing here with *duration time* in order to resolve expressions like *Monday* or *20 days* against valid time. This needs to be handled by a richer temporal ontology and temporal arithmetic.

Secondly, *temporal quantification*, such as *four hours every week*, needs to be addressed by a richer temporal inventory. Thirdly, even though *underspecified time* is handled by our implementation through wildcards in the XSD dateTime format (e.g., year missing in *Over New Year's Eve, I have visited the Eiffel Tower*), we do not focus on this here. The solution requires to make certain rule tests sensitive to the fact that underspecified time is only partially ordered. These tests then return *true*, *false*, or *don't-know*, whereas only *true* indicates that the test has succeeded, leading to the instantiation of the right hand side of the rule.

Fourthly, coalescing temporal information (i.e., building larger intervals from intervals with overlapping parts) should be addressed in custom rules and should not be regarded as part of the extended RDFS/OWL rule set, since this functionality depends on the (semantic) nature of predicates and the assumption whether temporal intervals are convex (i.e., contain no "holes") or not.

And finally, certain temporal inferences such as $p(\vec{x}, s, t)$ *entails* $p(\vec{x}, s', t')$ in case $s \leq s' \leq t' \leq t$ should *not* be handled in the below rules, since termination of the computation of the deductive closure is no longer guaranteed. Such information can only be obtained on the query level.

3 Ontology for Biographical Knowledge

We already indicated that we favor approach **1** as it is the most perspicuous of the nine approaches presented above, shows the best memory and runtime footprint, and always guarantees a terminating closure computation for extended RDFS (Hayes, 2004) and OWL (ter Horst, 2005) entailment, as shown in Krieger (2012).

In the introduction, we argued that axiomatic knowledge about classes (TBox) and properties (RBox) does *not* need to have a notion of time—this is universal knowledge which we assume to be *static*. For instance, we do *not* assume that the subtype relationship between two classes only holds for some period of time or that an URI should be regarded as a property at time t and as a class at a different time t' (even though this would be possible). The assertional knowledge of an ontology (ABox), i.e., the set of relation instances, however, is what we equip with time (see the various approaches for the marriedTo example in Figure 1), as this is knowledge that has undergone a temporal change.

In this section, we present the schema (the TBox and the RBox) of an ontology that we had developed originally for the TAKE project (http://take.dfki.de) and that was used in the KOMPARSE project (http://komparse.dfki.de) to represent *biographical information* about celebrities (Adolphs et al., 2010). This ontology has been reused and extended in the EU projects MONNET (http://cordis.europa.eu/fp7/ict/language-technologies) and TRENDMINER (http://www.trendminer-project.eu). This biography ontology is now *part* of a larger set of *independently* developed ontologies (called TMO, for TRENDMINER ONTOLOGIES) which are interlinked to one another through the use of *interface axioms* (Krieger and Declerck, 2014). These interface axioms either relates classes (TBox) and properties (RBox) from different sub-ontologies through the use of description logic axiom constructors, e.g.,

$\text{bio:Person} \equiv \text{pol:Person}$

or constrain the domain and range of potentially underspecified properties, e.g.,

$\top \sqsubseteq \forall \text{op:hasHolder} \,.\, \text{bio:Agent}$

The property hasHolder from the opinion ontology (prefix op) is a good example of a property for which only the domain has been specified, viz., op:Opinion:

$\top \sqsubseteq \forall \text{op:hasHolder}^{-} \,.\, \text{op:Opinion}$

However, hasHolder consciously lacks its range, since this information should only be added when several ontologies are brought together.

The above axioms together with the two terminological axioms from the biography (prefix bio) and the politics (prefix pol) ontologies

$\text{bio:Person} \sqsubseteq \text{bio:Agent}$

$\text{pol:Journalist} \sqsubseteq \text{pol:Person}$

guarantee to draw legal inferences, such as *journalists are holders of opinions*, even though the interface axiom above constrain holders of opinions to be of type bio:Agent.

TMO has been assembled from 16 sub-ontologies, some of them also dealing with the representation of biographical knowledge, others describing concepts that can be found in politics and sociology. Especially the *opinion ontology* can be used to model *provenance* information, important for biographical knowledge; for instance, information about the:

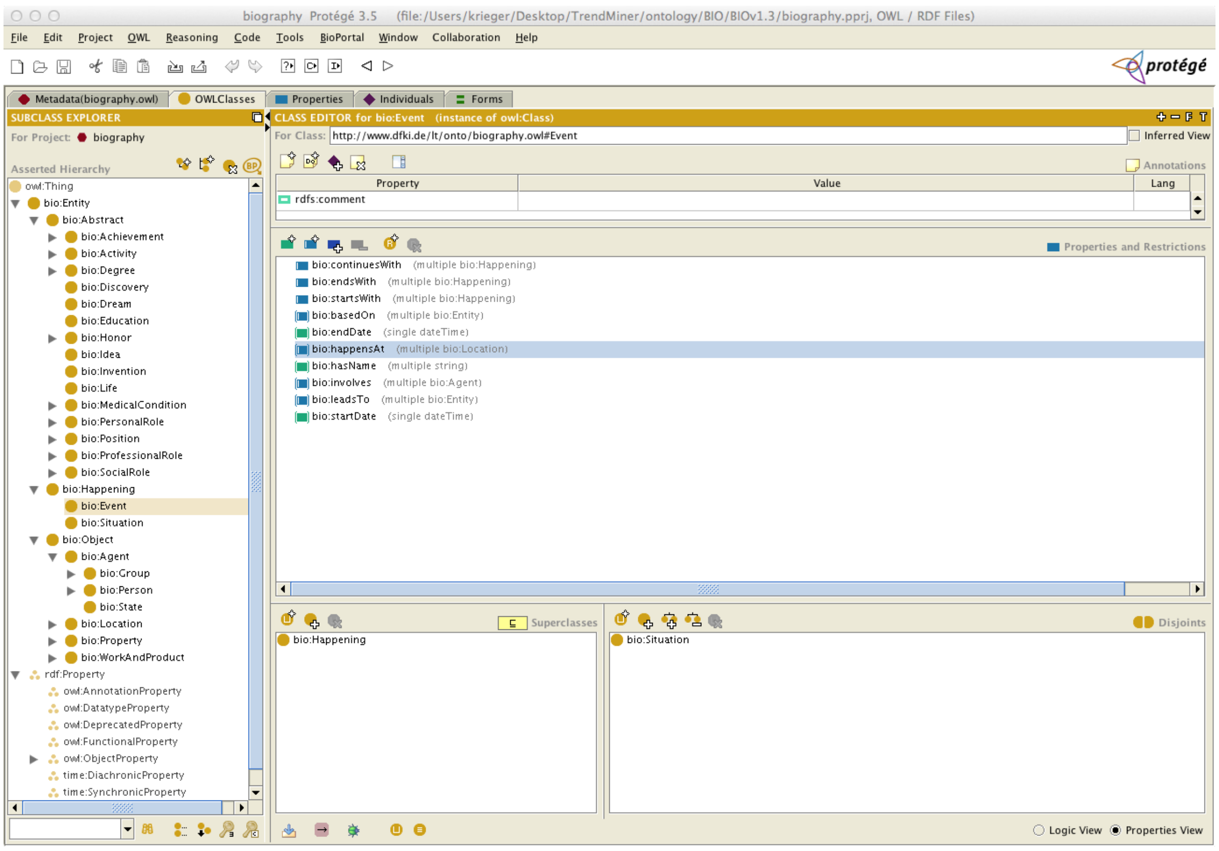
- holder of the opinion: hasHolder;
- source from which the info was taken: extractedFrom;
- time when the opinion was published: utteredAt;
- trustworthiness of the holder: holdersTrust;
- polarity of the opinion: hasPolarity.

The TMO ontology suite is freely available for academic research and to other sites upon request (see http://www.dfki.de/lt/onto/). Parts of the taxonomic structure of the biography ontology is depicted in Figure 2.

3.1 Overall Guidelines

TMO, and thus the biography ontology, implements several "guidelines" that we have found useful in many projects which have dealt with the representation of time-dependent knowledge (some of the arguments have already been presented):

1. model the TBox and RBox axioms of an ontology as if there is *no* time, since the ontology schema is regarded to be immutable; consequence: standard ontology editors, such as *Protégé* can be used for this task.
2. cross-classify all properties as being either *synchronic* or *diachronic*; advantage: these property characteristics can be used, amongst other things, as distinguishing marks in entailment rules (see examples).
3. populate the ABox of an ontology *with* extended relation instances, i.e., with quintuples whose fourth and fifth argument encode the temporal extent of the preceding atemporal statement (the triple).
4. extend the RDFS/OWL entailment rules by a temporal dimension; example: use XSD's date or dateTime format to implement an interval-based calendar time (used by the examples in this paper).


biography Protégé 3.5 (file:/Users/krieger/Desktop/TrendMiner/ontology/BIO/BIOv1.3/biography.pprj, OWL / RDF Files)
File Edit Project OWL Reasoning Code Tools BioPortal Window Collaboration Help
protégé
Metadata(biography.owl) OWLClasses Properties Individuals Forms
SUBCLASS EXPLORER
CLASS EDITOR for bio:Event (instance of owl:Class)
Inferred View
Annotations
Properties and Restrictions
Superclasses
Disjoints
Logic View
Properties View

For Project: biography

Asserted Hierarchy

```
owl:Thing
▼ bio:Entity
    ▼ bio:Abstract
        ▶ bio:Achievement
        ▶ bio:Activity
        ▶ bio:Degree
          bio:Discovery
          bio:Dream
          bio:Education
        ▶ bio:Honor
          bio:Idea
          bio:Invention
          bio:Life
        ▶ bio:MedicalCondition
        ▶ bio:PersonalRole
        ▶ bio:Position
        ▶ bio:ProfessionalRole
        ▶ bio:SocialRole
    ▼ bio:Happening
          bio:Event
          bio:Situation
    ▼ bio:Object
        ▼ bio:Agent
            ▶ bio:Group
            ▶ bio:Person
              bio:State
        ▶ bio:Location
        ▶ bio:Property
        ▶ bio:WorkAndProduct
▼ rdf:Property
      owl:AnnotationProperty
      owl:DatatypeProperty
      owl:DeprecatedProperty
      owl:FunctionalProperty
    ▶ owl:ObjectProperty
      time:DiachronicProperty
      time:SynchronicProperty
```

For Class: http://www.dfki.de/lt/onto/biography.owl#Event

Property	Value	Lang
rdfs:comment		

```
bio:continuesWith   (multiple bio:Happening)
bio:endsWith   (multiple bio:Happening)
bio:startsWith   (multiple bio:Happening)
bio:basedOn   (multiple bio:Entity)
bio:endDate   (single dateTime)
bio:happensAt   (multiple bio:Location)
bio:hasName   (multiple string)
bio:involves   (multiple bio:Agent)
bio:leadsTo   (multiple bio:Entity)
bio:startDate   (single dateTime)
```

Superclasses: bio:Happening

Disjoints: bio:Situation

Figure 2: The class subsumption hierarchy of the biography ontology. Note the two subclasses time:DiachronicProperty and time:SynchronicProperty of class rdf:Property that are used to cross-classify (i.e., to type) the properties of the biography ontology; see Figure 4.

3.2 Tri-Partite Structure

The biography ontology assumes a tri-partite structure, defining a most general class Entity, having pairwise disjoint subclasses Abstract, Happening, and Object. TMO is a lightweight ontology that consists of 146 classes and 80 properties, and is of expressivity $\mathcal{SHIN}(D)$, according to the *Ontology metrics* pane of *Protégé*, version 4.3.0. A partial view of the three subclasses and properties linking them is given in Figure 3.

3.2.1 Abstract

Ontological categories that do not fit into Happening or Object are regarded to be of type Abstract, thus this class is a kind of "remainder" category. Abstract things can be used to describe *literal* concepts, e.g., activities, academic degrees, ideas, inventions, the life, or personal, professional, and social roles. An abstraction manifestsIn real-world happenings, whereas the outcome of a happening leadsTo virtually everything (= Entity). For example: a specific military activity (the invasion of Poland) manifested in World War II. The outcome of WW-II has led to military inventions (Abstract), has led to the Cold War (Happening), and has led to the building of 86 U2 aircrafts (Object).

3.2.2 Happening

Happenings are things that "happen" or "unfold" and are disjointly categorized as being either static atomic Situations or dynamic decomposable Events. They come with a (possibly underspecified) startDate and endDate. A happening is basedOn or leadsTo entities (i.e., either abstract things, further happenings, or concrete objects), thus these properties can be used to encode pre- and post-conditions of a happening. An instance of this class also involves Agents and happensAt a Location. Situations help to "terminate" the decomposition of a Happening. The other subclass Event can be used to model simple unordered processes, as it comes with three relational properties of its own, viz., startsWith, continuesWith, and endsWith, all mapping to Happening (see Figure 2).

3.2.3 Object

Objects are "physical" things and mostly deal with Agents (an exhaustive disjoint partition between Person, Group, and political State) and other categories that we think are relevant for biographical information, e.g., Location, material Property, or WorkAndProduct. A Person isAwareOf a Happening: (s)he "owns" it, can be part of it, or learns about a happening. As isAwareOf is a *diachronic* property, awareness of a happening might even turn into oblivion.

3.3 Practical Temporal Reasoning

For a larger non-trivial example, let us again turn our attention to the marriage of *Tony Blair* and *Cherie Booth*. marriedTo is at the same time a *symmetric*, a *diachronic*, a *functional*, and an *object* property (see the *Types* pane at the bottom of Figure 4).

We mentioned that we have cross-classified every property from the biography ontology as being either synchronic

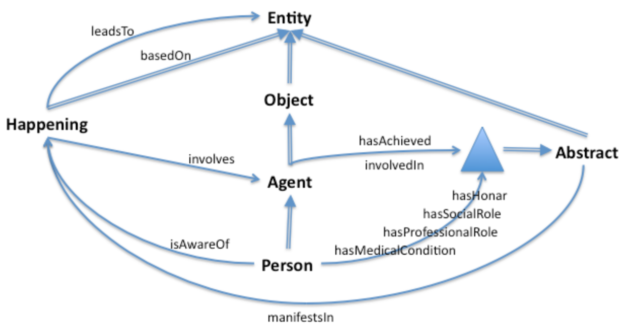


Figure 3: Properties of the biography ontology which relate the three disjoint classes Happening, Object, and Abstract. The solid blue triangle on the right side should indicate subclasses of the class Abstract, such as Achievement.

or diachronic and have already discussed the temporal extension of the entailment rule for functional diachronic *datatype* properties in Section 2.2. Let us now focus on the complementary rule for functional diachronic *object* properties which is applicable to the marriedTo relation:

```
?p rdf:type owl:FunctionalProperty
?p rdf:type time:DiachronicProperty
?p rdf:type owl:ObjectProperty
?x ?p ?y ?s1 ?e1
?x ?p ?z ?s2 ?e2
→
?y owl:sameAs ?z
@test
IntersectionNotEmpty ?s1 ?e1 ?s2 ?e2
```

Here, as in the former example, the additional left hand side test IntersectionNotEmpty checks whether the two temporal intervals $[s_1, e_1]$ and $[s_2, e_2]$ have a non-empty intersection. Assuming that a person is not married to more than one partner at the same time, such a rule is able to identify individuals/URIs bound to ?y and ?z for two properly overlapping observations through the use of owl:sameAs.

Consider again the Wikipedia entry for the marriage of Tony Blair and Cherie Booth that we used in the example from Section 2.2:

```
tony_blair marriedTo cherie_booth
   "1980-03-29"^^xsd:date "2015-05-08"^^xsd:date
```

and furthermore assume that the Economist article *The loneliness of Tony Blair* from December 2014 mentioned that Cherie Blair is Blair's wife (quintuple again):

```
tony_blair marriedTo cherie_blair
   "2014-12-20"^^xsd:date "2014-12-20"^^xsd:date
```

Now it is safe to assume that Cherie Booth and Cherie Blair are in fact the same person, according to the successful application of the above temporal entailment rule:

```
cherie_booth owl:sameAs cherie_blair
```

It is worth noting that sameAs statements will not be equipped with a temporal extent—commonsense dictates that once we do identify individuals, they will never fall apart.

At every moment in time, we never know how long a person is married to his/her partner in advance. That is why we introduced another property divorcedFrom, being the temporal disjoint object property to marriedTo (see the owl:disjointObjectProperty pane in Figure 4). As the Economist article does not specify the date of marriage, we better opt for a moment in time, when Blair and Booth were definitely married (actually a day: start = end). Luckily, the right hand side sameAs inference from above, together with another extended OWL entailment rule, called rdfp11 (ter Horst, 2005), makes sure that even

```
tony_blair marriedTo cherie_blair
   "1980-03-29"^^xsd:date "2015-05-08"^^xsd:date
```

is a valid entailment, exactly what we expect.

3.4 Temporal Arguments as Extra Arguments

So far, our approach has argued for a direct encoding of the temporal extent through two further arguments, turning a binary relation, such as marriedTo ⊆ Person × Person into a quaternary one: marriedTo ⊆ Person × Person × date × date. Given the original relation signature, the *non*-temporal entailment rule schema for *symmetric* binary relations from ter Horst (2005) thus leads to the following instantiation:

$$\mathit{marriedTo}(p, p') \rightarrow \mathit{marriedTo}(p', p)$$

as *symmetric* relations swap their domain and range arguments (p, p' being two people).

Now, if we add time (b = begin; e = end), we obtain:[1]

$$\mathit{marriedTo}(i; j, b, e) \rightarrow \mathit{marriedTo}(j, b, e; i)$$

Clearly, something has gone wrong here because symmetric relations assume the same number of arguments in domain and range position. One solution would be to reduplicate the starting and ending points, so we would end up in sexternary relation:

$$\mathit{marriedTo}(i, b, e; j, b, e) \rightarrow \mathit{marriedTo}(j, b, e; i, b, e)$$

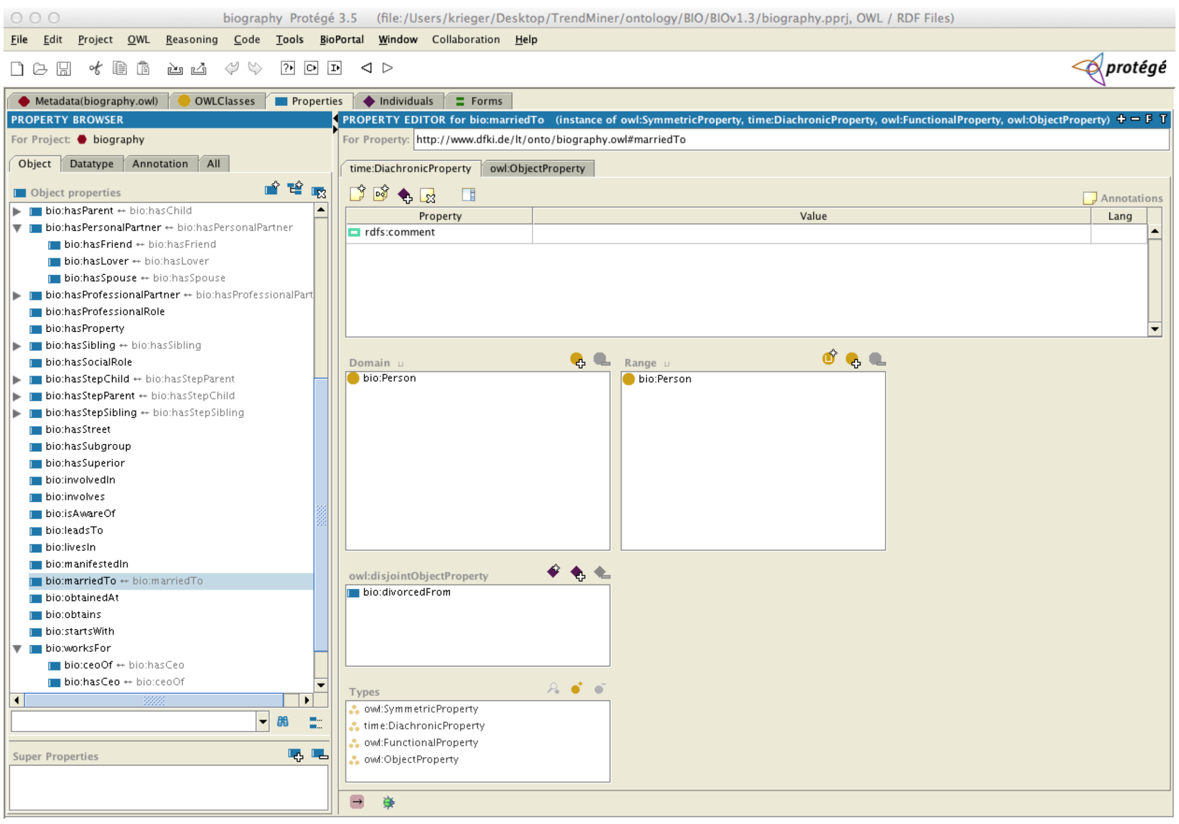
This is *not* an appealing solution as the structures become larger, and rules and queries are harder to formulate, read, debug, and process. What we would like to see is something like:

$$\mathit{marriedTo}(i; j; b, e) \rightarrow \mathit{marriedTo}(j; i; b, e)$$

whereas the *second* semicolon should indicate that the additional temporal arguments are *extra* arguments, belonging to the relation instance as such (a kind of relation instance annotation, not possible in OWL). Thus with this idea in mind, we can still keep the idea of having only binary relations, *without* introducing any new identifier (contrary to the rewrite schemas **2–7** from Figure 1).

Nevertheless, we are *not* arguing against arbitrary n-ary relations as we are convinced that many binary relations in today's ontologies are ignoring additional arguments (e.g., properties oriented towards ditransitive verbs or having additional modifiers/adjuncts) or come along with unsatisfactory means to encode the additional arguments (relation composition, by taking the object of a binary relation instance into account). The current biography ontology, for instance, poorly models the property obtains as a relation

[1]For better readability, we separate the domain and range arguments from one another by using a semicolon.



```
biography  Protégé 3.5    (file:/Users/krieger/Desktop/TrendMiner/ontology/BIO/BIOv1.3/biography.pprj, OWL / RDF Files)

PROPERTY BROWSER
For Project: biography

Object properties
▶ bio:hasParent ↔ bio:hasChild
▼ bio:hasPersonalPartner ↔ bio:hasPersonalPartner
     bio:hasFriend ↔ bio:hasFriend
     bio:hasLover ↔ bio:hasLover
     bio:hasSpouse ↔ bio:hasSpouse
▶ bio:hasProfessionalPartner ↔ bio:hasProfessionalPart
  bio:hasProfessionalRole
  bio:hasProperty
▶ bio:hasSibling ↔ bio:hasSibling
  bio:hasSocialRole
▶ bio:hasStepChild ↔ bio:hasStepParent
▶ bio:hasStepParent ↔ bio:hasStepChild
▶ bio:hasStepSibling ↔ bio:hasStepSibling
  bio:hasStreet
  bio:hasSubgroup
  bio:hasSuperior
  bio:involvedIn
  bio:involves
  bio:isAwareOf
  bio:leadsTo
  bio:livesIn
  bio:manifestedIn
  bio:marriedTo ↔ bio:marriedTo
  bio:obtainedAt
  bio:obtains
  bio:startsWith
▼ bio:worksFor
     bio:ceoOf ↔ bio:hasCeo
     bio:hasCeo ↔ bio:ceoOf

PROPERTY EDITOR for bio:marriedTo   (instance of owl:SymmetricProperty, time:DiachronicProperty, owl:FunctionalProperty, owl:ObjectProperty)
For Property: http://www.dfki.de/lt/onto/biography.owl#marriedTo

time:DiachronicProperty | owl:ObjectProperty

Property        Value        Lang
rdfs:comment

Domain ⊔                     Range ⊔
bio:Person                   bio:Person

owl:disjointObjectProperty
bio:divorcedFrom

Types
owl:SymmetricProperty
time:DiachronicProperty
owl:FunctionalProperty
owl:ObjectProperty
```

Figure 4: The property subsumption hierarchy of the biography ontology.

between people and (academic) degrees. In order to obtain the educational organization where the degree was obtained, we employ relation composition at the moment, using an additional property $\mathsf{obtainedAt}$ between degree and education:

$$\mathsf{obtainedAt} \circ \mathsf{obtains} \subseteq \mathsf{Person} \times \mathsf{EducationalOrganization}$$

This way of representing the additional argument is related to approach **9** from Figure 1 and only works because $\mathsf{obtains}$ is *inverse functional* (a characteristics applicable to properties in OWL). Ideally, $\mathsf{obtains}$ should be modeled as a quinternary relation, having one domain argument, two range arguments, and two extra temporal arguments:

$$\begin{aligned} \mathsf{obtains} \subseteq \mathsf{Person} \times & & \text{// domain} \\ & \mathsf{Degree} \times \mathsf{EducationalOrganization} \times & \text{// range} \\ & \mathsf{xsd{:}dateTime} \times \mathsf{xsd{:}dateTime} & \text{// extra} \end{aligned}$$

In order to easily define such non-binary relations, ontology editors need to be extended by *Cartesian* types. In Krieger and Willms (2015), we described ×-*Protégé*, an extension of the *Protégé* ontology editor that provides means to define such Cartesian types and to use them to type the domain, range, and extra arguments of non-binary relations. A first public version of ×-*Protégé* will be available in mid 2015.

4 Relation vs. Event Representation

The approaches considered in Section 2 were investigated on how well they perform w.r.t. *binary* relations whose two arguments are considered to be *obligatory*. Such a kind of relation is the default case in today's popular knowledge resources, such as YAGO, DBpedia, BabelNet, or Google's Knowledge Graph.

In case more (e.g., time) and especially *optional* arguments are investigated, our verdict concerning the different approaches might turn into a different direction, so the representation format needs to be updated (in the best case) or changed (in the worst case). Consider the following example, taken from (Davidson, 1967, p. 83):

Jones buttered the toast $\underline{\textit{in the bathroom}}$ $\underline{\textit{with a knife}}$ $\underline{\textit{at midnight}}$.

The binary base relation butter (we assume a direct mapping of the transitive verb to the relation name here) now needs to be split and/or extended by further optional arguments, as the following sentences are perfectly legal:

Jones buttered the toast.
Jones buttered the toast $\underline{\textit{in the bathroom}}$.
Jones buttered the toast $\underline{\textit{with a knife}}$.
Jones buttered the toast $\underline{\textit{at midnight}}$.
Jones buttered the toast $\underline{\textit{in the bathroom}}$ $\underline{\textit{with a knife}}$.
Jones buttered the toast $\underline{\textit{in the bathroom}}$ $\underline{\textit{at midnight}}$.
..... etc.

In principle, the number of adjuncts is *not* bounded, thus adding a large number of potentially underspecified direct relation arguments is probably a bad solution. Today's technologies often address such hidden arguments through a kind of *relation composition* as we have seen above for the $\mathsf{obtains}$ example from the last section and listed as approach **9** in Figure 1. We think that this modeling "trick" is *unsatisfactory* as it operates on the object of the binary relation instance, but *not* on the relation instance itself (besides being only correct if the original relation is inverse functional, as explained before).

Our *personal* solution would model the *obligatory* arguments, including (under- or unspecified) *time* and perhaps space, as *direct* arguments of the corresponding relation instance or tuple (approach **1**). A further argument, an *event* identifier, also takes part in the relation. Optional arguments, however, would be addressed through binary relations, now working on the *event* argument. Applying this kind of *Davidsonian* or *event* representation to the above example gives us (informal relational notation):

$$\exists e \,.\, \mathit{butter}(e, \mathit{Jones}, \mathit{toast}, \mathit{at\ midnight}) \wedge \mathit{location}(e, \mathit{bathroom}) \wedge \mathit{instrument}(e, \mathit{knife})$$

It is worth noting that two of the approaches from Figure 1 are related to such an event representation, viz., **3** and **4**.

Approach **3** (*internal* reification) can be seen as a kind of "owlfication" of *Neo-Davidsonian* semantics (Parsons, 1990), as the original relation is always turned into an *event* (an OWL class). Here the event identifier e from above directly corresponds to a URI, referring to an instance of the OWL class. For instance, the marriedTo relation is turned into an event class, say Marry; thus:

tony_blair marriedTo cherie_booth
"1980-03-29"^^xsd:date "2015-05-08"^^xsd:date

needs to be expressed by (we use *VerbNet* terminology):

e rdf:type Marry
e agent tony_blair
e co-agent cherie_booth
e starts "1980-03-29"^^xsd:date
e ends "2015-05-08"^^xsd:date

Approach **4** (fact identifier) is a kind of *external* reification. YAGO uses its own extension of the N3 plain triple format, called N4, which associate unique identifiers i with each time-dependent fact. However, the association $i := \mathit{marriedTo}(p, p')$ has the disadvantage of *not* being part of the triple repository, as it is a quadruple technically. So we guess that there exists a separate extendable mapping table outside of the semantic repository, storing the triples.

Luckily, the biography ontology presented in Section 3 both allows for extended relation instances (as shown before), but also Davidsonian-like events through the class Happening and its subclasses Event and Situation (see Figure 2). As there does not exist a Marry event class so far (but only the marriedTo property), such a class needs to be introduced as a subclass of class Event, if needed.

5 Related Ontologies

Several ontologies addressing the representation of biographical information, cultural heritage information, and news-related information exist today, all building on Description Logics and Semantic Web technology standards. These include *ESO* (Segers et al., 2015), *Wikidata* (Erxleben et al., 2014), the *BiographyNet* ontology (Ockeloen et al., 2013), the *BBC Storyline Ontology* (Wilton et al., 2013), *SEM* (van Hage et al., 2011), *FRBR*$_{OO}$ (Le Bœuf, 2010), *LODE* (Shaw et al., 2009), or *Event-Model-F* (Scherp et al., 2009). Some of these ontologies make use of other resources, such as *WordNet*, *FrameNet*, *Wikipedia*, *SUMO*, *DOLCE*, or *CIDOC CRM*. In order to represent time-dependent knowledge, these approaches always need to stick to an event-like representation in which all information is hidden in an object and time is accessible through properties, similar to approach **3** in Figure 1. None of them are able to encode time as direct arguments of a relation instance (approach **1**). A comparison of some of these event ontologies is presented in (Shaw et al., 2009, section 2) and (van Hage et al., 2011, section 5).

As we have indicated in the beginning of Section 3 (Journalist example), OWL axiom constructors and domain/range restrictions allow us to manually interface our biography ontology with other ontologies, may they be complimentary domain ontologies (*opinion, politics, sociology*), overlapping *biography* event ontologies (see above), or even OWL versions of *upper* ontologies (if desired), such as *DOLCE+DnS* (Gangemi et al., 2002), *SUMO* (Niles and Pease, 2001), or *Cyc* (Reed and Lenat, 2002). For instance, if we would like to interface the *BBC storyline ontology*, the following single axiom suffices:

bio:Happening $\equiv$ nsl:Event

Connecting with *LODE* essentially reduces to:

bio:Happening $\equiv$ lode:Event
bio:happensAt $\equiv$ lode:atPlace
bio:involves $\equiv$ lode:involvedAgent
bio:basedOn $\sqsubseteq$ lode:involved
bio:leadsTo $\sqsubseteq$ lode:involved

Other properties from *LODE* either do not have a direct counterpart (lode:illustrate) or need to be decomposed (lode:atTime onto bio:startDate and bio:endDate). The sub-properties bio:startsWith, bio:continuesWith, and bio:endsWith from the class bio:Event would even allow us to decompose *LODE* events into smaller units, a feature partially available in the *SEM* ontology:

bio:startsWith $\sqsubseteq$ sem:hasSubEvent
bio:continuesWith $\sqsubseteq$ sem:hasSubEvent
bio:endsWith $\sqsubseteq$ sem:hasSubEvent

As our ontology comes with the class bio:Happening, it is possible to take advantage of the great effort invested in the definition of event types in the *ESO* ontology. We finally note that some of the mappings are not expressible through simple OWL axiom constructors, because they involve a translation from n-ary relation instances to sets of triples (and vice versa). This would require to apply *HFC migration rules*, similar to the rewrite rule of approach **3** in Figure 1 which mediates between the quaternary marriedTo relation and its event representation MarriedToEvent.

6 Summary and Conclusion

In this paper, we have presented an overview of nine approaches to the representation of time-dependent knowledge and have favored the direct encoding of the temporal information as extra arguments of the original relation instance. Nevertheless, allowing at the same time for an event-based representation of situations, happening in the real world, is profitable as a knowledge engineer might choose the representation which fits her/his needs. For instance, a marriage ceremony between two people is probably modeled best as an event, whereas the fact that these two people are married for a specific time period is better represented as a quaternary relation. The lightweight biography ontology, presented in this paper, allows both views through the very general class Happening and relations defined between classes which are extended by a starting and ending time, expressing the temporal extent in which the atemporal fact is true (called *valid time* in temporal databases).

Our debate on the right representation format can even be viewed as the more *general quest* on how to integrate/add important (meta) information that has been neglected in the past for practical matters, but has gained a lot of attention recently; see the W3C recommendation for the *provenance* data model *PROV-DM* (Moreau and Missier, 2013). This additional information might include the *holder* of a time-dependent statement or event (person, website, program/service), the spacial *location* of the holder, the *time* when the statement/event was communicated by the holder or made public on the Web (related to *transaction time* in temporal databases), the *trustworthiness* of the holder, and the attitude of the holder w.r.t. the statement/event (sentiment/opinion). Ontologies for all these different aspects already exist today (for instance, the *BiographNet* ontology (Ockeloen et al., 2013) which incorporates a multi-level, multi-perspective model for provenance), but a unified standard is still missing. As a short-/mid-term workaround, we suggest to manually interface these different sources of information, as indicated in Section 5, thus making it possible to incorporate work carried out by other researchers.

Acknowledgements

The work described in this paper has been partially carried out in the *ALL SIDES* project, funded by the German Federal Ministry of Education and Research under contract number FKZ 01IW14002 and the *PHEME* project, funded by the European Union's 7th Framework Programme under grant agreement no 611233. The authors would like to thank our three reviewers and Antske Fokkens for their useful comments, Hans Uszkoreit for giving the initial impetus for the development of the biography ontology several years ago, and Bernd Kiefer for always having an open ear in many discussions.

7 References

Peter Adolphs, Xiwen Cheng, Tina Klüwer, Hans Uszkoreit, and Feiyu Xu. 2010. Question answering biographic information and social network powered by the semantic web. In *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC)*, pages 2764–2768.

Gavin Carothers and Andy Seaborne. 2014. RDF 1.1 N-Triples. a line-based syntax for an RDF graph. Technical report, W3C.

Donald Davidson. 1967. The logical form of action sentences. In Nicholas Rescher, editor, *The Logic of Decision and Action*, pages 81–95. University of Pittsburgh Press.

Fredo Erxleben, Michael Günther, Markus Krötzsch, Julian Mendez, and Denny Vrandečić. 2014. Introducing Wikidata to the Linked Data Web. In *Proceedings of the 13th International Semantic Web Conference (ISWC)*, pages 50–65.

Aldo Gangemi, Nicola Guarino, Claudio Masolo, Alessandro Oltramari, and Luc Schneider. 2002. Sweetening ontologies with DOLCE. In *Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW)*, pages 166–181.

Aldo Gangemi. 2011. SuperDuper schema: an OWL2+RIF DnS pattern. In *KCAP 2011* Deep Knowledge Representation Challenge Workshop.

Patrick Hayes and Chris Welty. 2006. Defining N-ary relations on the Semantic Web. Technical report, W3C.

Patrick Hayes. 2004. RDF semantics. Technical report, W3C.

Johannes Hoffart, Fabian M. Suchanek, Klaus Berberich, Edwin Lewis Kelham, Gerard de Melo, and Gerhard Weikum. 2011. YAGO2: Exploring and querying world knowledge in time, space, context, and many languages. In *Proceedings of the 20th International World Wide Web Conference (WWW 2011)*, pages 229–232.

Hans-Ulrich Krieger and Thierry Declerck. 2014. TMO—the federated ontology of the TrendMiner project. In *Proceedings of the 9th edition of the Language Resources and Evaluation Conference (LREC)*.

Hans-Ulrich Krieger and Christian Willms. 2015. Extending OWL ontologies by Cartesian types to represent N-ary relations in natural language. In *Proceedings of the IWCS Workshop on Language and Ontologies*.

Hans-Ulrich Krieger. 2008. Where temporal description logics fail: Representing temporally-changing relationships. In *KI 2008: Advances in Artificial Intelligence*, volume 5243 of *Lecture Notes in Artificial Intelligence*, pages 249–257. Springer.

Hans-Ulrich Krieger. 2012. A temporal extension of the Hayes/ter Horst entailment rules and an alternative to W3C's n-ary relations. In *Proceedings of the 7th International Conference on Formal Ontology in Information Systems (FOIS)*, pages 323–336.

Hans-Ulrich Krieger. 2013. An efficient implementation of equivalence relations in OWL via rule and query rewriting. In *Proceedings of the 7th IEEE International Conference on Semantic Computing (ICSC)*, pages 260–263.

Hans-Ulrich Krieger. 2014. A detailed comparison of seven approaches for the annotation of time-dependent factual knowledge in RDF and OWL. In *Proceedings of the 10th Joint ACL-ISO Workshop on Interoperable Semantic Annotation (ISA)*.

Patrick Le Bœuf. 2010. From FRBR to FRBR_{OO} through CIDOC CRM. Slide presentation, October. International Symposium on the Future of Information Organization Research, Taipei, Taiwan.

Frank Manola and Eric Miller. 2004. RDF primer. Technical report, W3C.

John McCarthy and Patrick J. Hayes. 1969. Some philosophical problems from the standpoint of artificial intelligence. In B. Meltzer and D. Michie, editors, *Machine Intelligence 4*, pages 463–502. Edinburgh University Press.

Luc Moreau and Paolo Missier. 2013. PROV-DM: The

PROV data model. Technical report, W3C. W3C Recommendation 30 April 2013.

Boris Motik, Bernardo Cuenca Grau, Ian Horrocks, Zhe Wu, Achille Fokoue, and Carsten Lutz. 2012. OWL 2 web ontology language profiles. Technical report, W3C. W3C Recommendation 11 December 2012.

Ian Niles and Adam Pease. 2001. Towards a standard upper ontology. In *Proceedings of the 2nd International Conference on Formal Ontology in Information Systems (FOIS)*, pages 2–9.

Niels Ockeloen, Antske Fokkens, Serge ter Braake, Piek Vossen, Victor de Boer, Guus Schreiber, and Susan Legêne. 2013. BiographyNet: Managing provenance at multiple levels and from different perspectives. In *Proceedings of the 3rd International Workshop on Linked Science (LISC) at ISWC*, pages 59–71.

Terence Parsons. 1990. *Events in the Semantics of English. A Study in Subatomic Semantics*. MIT Press, Cambridge, MA.

Stephen L. Reed and DouglaS B. Lenat. 2002. Mapping ontologies into Cyc. In *Proceedings of the AAAI 2002 Conference Workshop on Ontologies For The Semantic Web*, pages 1–6.

Dave Reynolds. 2006. Jena rules tutorial. In *Jena User Conference*. PowerPoint presentation.

Ansgar Scherp, Thomas Franz, Carsten Saathoff, and Steffen Staab. 2009. F—a model of events based on the foundational ontology DOLCE+DnS Ultralite. In *Proceedings of the 5th International Conference on Knowledge Capture (K-CAP)*, pages 137–144.

James G. Schmolze. 1989. Terminological knowledge representation systems supporting n-ary terms. In *Proceedings of the 1st International Conference on Principles of Knowledge Representation and Reasoning (KR)*, pages 432–443.

Roxane Segers, Piek Vossen, Marco Rospocher, Luciano Serafini, Egoitz Laparra, and German Rigau. 2015. ESO: a frame based ontology for events and implied situations. In *Proceedings of MAPLEX*.

Ryan Shaw, Raphaël Troncy, and Lynda Hardman. 2009. LODE: Linking open descriptions of events. In *Proceedings of the 4th Asian Semantic Web Conference (ASWC)*, pages 153–167.

Richard T. Snodgrass. 2000. *Developing Time-Oriented Database Applications in SQL*. Morgan Kaufmann, San Francisco, CA.

Herman J. ter Horst. 2005. Completeness, decidability and complexity of entailment for RDF Schema and a semantic extension involving the OWL vocabulary. *Journal of Web Semantics*, 3:79–115.

Willem Robert van Hage, Véronique Malaisé, Roxane Segers, Laura Hollink, and Guus Schreiber. 2011. Design and use of the simple event model (SEM). *Journal of Web Semantics*, 9(2):128–136.

Christopher Welty and Richard Fikes. 2006. A reusable ontology for fluents in OWL. In *Proceedings of 4th FOIS*, pages 226–236.

Paul Wilton, Jeremy Tarling, and Jarred McGinnis. 2013. Storyline ontology: An ontology to represent news storylines. http://www.bbc.co.uk/ontologies/storyline, May. Contributors: Paul Rissen, Helen Lippell, Matt Chadburn, Tom Leitch, Dan Brickley, Michael Smethurst, Sebastien Cevey.