OntoElecting Requirements for Domain Ontologies

The Case of Time Domain

Vadim Ermolayev*,[a]

[a] Department of Computer Science, Zaporizhzhia National University, Ukraine

Abstract. *This paper reports on the use of the OntoElect methodology for evaluating the fitness of an existing ontology to the requirements of the knowledge stakeholders in a domain. It demonstrates, that a thorough routine for indirect elicitation, ensuring completeness, correctness of interpretation, using in ontology evaluation of these requirements is a must for ontology engineering. This is also valid if the requirements for ontology refinement are elaborated by a very high profile expert working groups. The approach used in the reported research is based on the use of OntoElect – the methodology for ontology refinement. The workflow of OntoElect contains three phases: feature elicitation, requirements conceptualization, and ontology evaluation. It elicits the set of terms extracted from a saturated collection of documents in the domain. It further sublimates these terms to the set of required features using the information about term significance in the form of numeric scores. Furthermore, it applies conceptualization and formalization activities to these features yielding their aggregations as ontological fragments interpreted as formalized requirements. Finally, the mappings are specified between the elements in the requirements and ontology elements. The scores are used in the mappings to indicate the strength of positive or negative votes regarding the evaluated ontology. The sum of the votes gives the overall numeric fitness measure of the ontology to the domain requirements. The paper presents the use of OntoElect in the use case of evaluating the W3C OWL-Time ontology against the requirements extracted from the proceedings of the TIME symposia series.*

Keywords. Requirements Elicitation • Automated Term Extraction from Text • Feature Conceptualization • Ontology Engineering • Ontology Refinement • Ontology Evaluation • Ontology Fitness • Domain Knowledge Stakeholder • Vote • OntoElect

1 Introduction

Developing an ontology, in its lifecycle, with an aim to make it meet the requirements of the domain knowledge stakeholders is a complicated task. The State–of–the–Art approaches to ontology engineering still lack a rigorous engineering approach to measure the degree at which an ontology corresponds to the requirements. A major challenge is to elicit these requirements from the expert community around a domain in a way to ensure completeness and correct interpretation. Many popular methodologies, further mentioned in Section 2, suggest that the requirements have to be elicited via direct communication with the subject experts in the domain. Relying on this direct elicitation approach is however unrealistic. Subject experts quite often oppose direct approaches, like interviews, brainstorming sessions, etc., as an undesired overhead to their extensive commitments and prioritise these lowly. In interpreting requirements, the experts and knowledge engineers use different languages with various notations and expressive power. Thus, there is a need to either propose a lingua franca for both groups to share, or to find an indirect way to acquire domain understanding and requirements from the community of experts.

* Corresponding author.
E-mail. vadim@ermolayev.com

Shared languages are elaborated as simple, yet sufficiently expressive lexicons, that are constructed to be equally and easily interpreted by subject experts and ontology engineers in different domains Those could be positioned at a pre-design phase of the lifecycle of an information system or ontology. One notable framework is Klagenfurt Conceptual Pre-design Model (KCPM) by Kop et al. (2004). The alternative path is pursued within the field of Ontology Learning from Texts – see for example Wong et al. (2012) - which is a subfield in Text Mining. The most relevant indirect technique for requirement elicitation is, perhaps, Automated Term Extraction, which provides, together with the terms, the information to assess numerically their significance. However, the output of these extraction routines is just a flat set of terms labelling the required features. Consequently, the questions about the completeness of the requirements set and the correctness of their interpretations are left without an answer.

Even if one is lucky to have a complete and correct set of domain requirements shared by the community of subject experts, there is yet one more complication – the lack of objective quantitative metrics to assess the degree to which an ontology meets these requirements. This is not surprising because the representations of the requirements differ from ontology representations. The requirements to an ontology, to be correctly interpreted, have to be specified in a language which is easily understood by a domain expert who is not a professional ontologist. From the other hand, an ontology, to be properly used, has to be specified in a formal representation language, processable by machines, which reads weirdly to subject experts. Therefore, there is a definite need to have a way: (i) to transform requirements to a form, directly mappable to ontologies; and (ii) measure if the transformed requirements are fully implemented in an ontology.

On this way, a synergy between Conceptual Modelling and Ontology Engineering may help devise a proper engineering approach that answers the outlined questions and attempts to overcome the above mentioned challenging complications. This paper presents OntoElect – the methodology for ontology refinement that offers a rigorously measurable way to: (i) elicit and formalize the statistically representative sets of domain requirements; (ii) formalize these requirements as ontology fragments through conceptualization; and (iii) evaluate the domain ontology against these formalized requirements. To demonstrate that the methodology is valid, the use case in the domain of Time Representation and Reasoning is presented.

The remainder of the paper is structured as follows. Section 2 further explains the motivation toward developing a methodology such as OntoElect. It also briefly reviews the related work in light of seeking the answers to the important questions on assessing completeness and correctness of requirements, and also on evaluating the degree at which the ontology meets these requirements. Section 3 presents OntoElect in terms of describing its workflow, phases, techniques, formalisms, metrics and tools. Section 4 applies OntoElect to eliciting the requirements on time representation from the TIME collection of documents and evaluating the W3C OWL-Time ontology against these requirements. Section 5 summarizes the results and concludes the paper.

2 Motivation and Related Work

Roughly a decade ago, a reviewer of an ontology paper submitted to a Conceptual Modelling community conference remarked, that the paper would have been a good candidate to pass the test of time if the term Ontology had been replaced by the term Conceptual Model, also including related methodological issues. S/he also mentioned that ontolgies are a quick and careless way to sketch out conceptual models. Stimulated by this remark, I started thinking about how to further develop the craft of ontology design into real engineering. It turned out that both conceptual modelling and ontology development were crafts, at least regarding requirements elicitation. Both disciplines always claimed their careful attitude to the requirements of domain knowledge stakeholders. They did not however provide an objective and rigorous way to measure if: (a) all significant requirements were put on the table; (b) all these requirements were correct or correctly interpreted; and (c) the final product met these significant requirements satisfactorily.

The mainstream in ontology engineering methodologies (Gómez-Pérez et al. (2004), Pinto et al. (2004), Schreiber (1999), Suárez-Figueroa et al. (2012), Sure et al. (2004) – to mention alphabetically the few most frequently cited) humbly forwards the answers to the questions on completeness, correctness, and fitness to requirements to the subject experts in the domain, through interviews, brainstorming sessions, or other ways of direct knowledge elicitation. The bottleneck is however that these experts often consider as inefficient the ratio of their own resource to be spent versus the utility of the resulting ontology as an artefact usable in their professional activity. Subject experts are sometimes also careful not to becoming less competitive and valuable to the company or community after making their knowledge available to the public in an explicit form. Therefore, indirect methods for requirements elicitation need to be developed to help provide a rigorous and explicit output in answering the above mentioned important questions.

2.1 Completeness

In Information Retrieval, completeness is often regarded as an indicator for results quality and expressed as recall metric. Recall is also used as one of the basic metrics for assessing quality in Automated Term Extraction which could be used as one of the enabling techniques for building terminology lists, thesauri etc. which could be considered as lightweight ontologies. More details on the related work in Automated Term Extraction, including approaches and implemented tools could be acquired from the review by Kosa et al. (2017).

Perhaps, one of the first mentions of the importance of a solution to the problem of completeness in Automated Term Extraction has been by Chien and Chen (2001) in the context of incremental term extraction from online text resources that are enlarged and extended over time. However, Chien and Chen (2001) looked at the problem from a linguistic perspective only and proposed a solution to analyse if all the term candidates have been extracted from a single textual document. OntoElect, proposed by Tatarintseva et al. (2013), looks at the problem broader and rather from statistical perspective. It suggests a method to measure the terminological completeness of the document collection by analyzing the saturation of terminological footprints of the incremental slices of the document collection, as for example reported by Ermolayev et al. (2014) regarding the domain of time representation and reasoning.

2.2 Correctness

The question on correctness reflects the long standing impedance mismatch between: (i) the requirements specified in a form clear to the domain knowledge stakeholders, but weird for the machine processing or for knowledge engineers; and (ii) ontologies specified in a way suited for machine processing but read weirdly by domain experts. Several approaches to resolve this mismatch could be found in the relevant literature.

One alternative, based on conceptual modelling and its lifecycle, is to propose a lingua franca – an easily understood subset of a conceptual modelling lexicon – that allows sharing and proper interpretation of requirement blueprints between subject experts and knowledge engineers. A notable representative of this approach is the Klagenfurt Conceptual Pre-design Model (KCPM) by Kop et al. (2004) developed in the NIBA project[1].

KCPM has been initially developed to bridge the above mentioned impedance mismatch, within the information system design cycle, between requirements specifications in natural (German) language and abstract conceptual models (for example, UML schemas). In KCPM, requirements are represented in a simplified yet formalized manner with a focus on the structural, functional, and behavioural terminology within an application domain. Regarding KCPM, there are at least two

[1] NIBA (Natürlichsprachige Informationsbedarfsanalyse) project has been funded by the Klaus Tschira Stiftung, Heidelberg.

aspects which are valuable for adoption, perhaps in an adapted form: (i) requirements are formalized to become closer to conceptual models; and (ii) requirements are focused on the terminology elaborated within the expert community in a domain. KCPM has been further developed by incorporating linguistic text processing in the requirements elicitation routine. Fliedl et al. (2005), based on KCPM and shallow text parsing, proposed an approach to bridge the application scenarios, taken in as natural language texts, and conceptual schemas describing the Universe of Discourse within a domain provided by the KCPM framework. These results inspired the development of the Conceptualization and Formalization pipeline in OntoElect presented in Section 3.3.

The approaches in ontology engineering based on the use of ontology design patterns, like those elaborated in frame of the NeOn EU project[2] and some other initiatives (e. g. (Presutti et al. 2012; Vrandecic 2010)] attack the correctness problem by offering reusable best practices in ontology engineering. They propose to shrink the space of opportunities for potential mistakes and mis-interpretations by offering ontology design blocks and patterns which were designed initially by renowned experts and passed the validity test. Further, design patterns may be effectively used for validating the correctness of an ontology, as proposed by Poveda-Villalón (2016). A disadvantage of this approach is that it is applied only to the output of an ontology engineering process, but not also to its input – to the requirements. Accordingly, it does help improve the quality of an ontology but does not help verify if the requirements were correctly met. Nevertheless, the results advancing this ontology engineering strand are quite useful in terms of re-using design patterns as higher-level elements for the specification and correct interpretation of requirements. These patterns are also useful in measuring structural and functional aspects of ontology quality, as proposed for example by Gangemi et al. (2006).

2.3 Ontology Fitness and Ontology Quality

Ontology quality is an issue which is broader than the focus of this paper. Here, one quality aspect – "‘How well does an ontology meet the requirements?"’ – is researched. It is however worth mentioning that there is a spectrum of aspects that need to be measured to assess the quality of an ontology.

Burton-Jones et al. (2005) proposed the suite of metrics to assess the quality of an arbitrary ontology, drawing upon semiotic theory. Their metrics assess the syntactic, semantic, pragmatic, and social aspects of quality. The metrics were operationalised and implemented in a prototype software tool called the Ontology Auditor.

A formal model for ontology evaluation and validation based on design patterns was proposed by Gangemi et al. (2006). Their model was based on the O2 meta-ontology and included three types of measures (structural, functional, and usability profiling). Based on this framework, they also elaborated the ontology of ontology validation called oQual. This quality evaluation framework was further extended and refined by Vrandecic (2010). Later, Poveda-Villalón (2016) proposed the set of patterns, metrics, and a tool for validating the correctness of an ontology as a design artefact.

It is known from the above mentioned literature, that one of the aggregate metrics for usability profiling of an ontology is fitness. Gangemi et al. (2006) distinguish fitness to competency questions and organizational fitness. Fitness to competency questions is in fact a way to assess how well an ontology meets the intended requirements, which is measured subjectively, as an opinion of a knowledge engineer and probably a subject expert. In difference to the predecessor work, OntoElect focuses explicitly on measuring fitness to requirements presented in a conceptualized and

[2] NeOn project has been co-funded by the European Commission’s Sixth Framework Programme under grant number IST-2005-027595.

formalized manner. It suggests doing this by specifying mappings between ontological fragments and accounting for required feature significance – as described in Section 3.4.

3 An Outline of OntoElect

OntoElect seeks for maximizing the fitness of the developed ontology to what the domain knowledge stakeholders think about the domain. Fitness is measured as the ratio of stakeholders' positive over negative "votes" – a metric that allows assessing the stakeholders' commitment to the ontology under development – reflecting how well their sentiment about the requirements is met. The more positive votes are collected – the higher the commitment is expected to be. If a critical mass of votes is acquired (say 50%+1, which is a simple majority vote), the ontology is considered to satisfactorily meet the requirements. Votes, and information leading to quantifying votes are collected indirectly – extracted from a statistically representative document collection.

Hence, OntoElect is an ontology refinement methodology. It facilitates, in an unbiased and measured way, to find out what needs to be improved in the domain ontology to better meet the requirements of the domain knowledge stakeholders. OntoElect may also be used to cross-evaluate different ontologies, describing the same domain, by comparing their fitness measurements. Therefore, the inputs to OntoElect are (i) a carefully chosen collection of good quality documents which is deemed, by the knowledge stakeholders, to be representative of the domain; and (ii) an ontology describing this domain. The output of OntoElect is in fact the set of recommendations, based on measurements, on why this or that ontology describes the domain well or not very well and what needs to be improved in it.

3.1 OntoElect Workflow and Tools

The flow of activities in OntoElect is shown in Fig. 1. This workflow involves the two roles: knowledge engineers and subject experts. The major workload and coordination function falls on the knowledge engineers. The subject expert role involves those domain knowledge stakeholders who contributed their professional texts to the document collection describing the domain in question.The phases of the OntoElect workflow are:

- **Feature Elicitation.** Determine the saturated sub-collection of the chosen document collection representative of the domain. Within this sub-collection, determine the documents that provide the highest terminological impact on the domain – the decisive minority subset. Extract the set of multi-word terms from the decisive minority documents. Select the significant terms with their significance scores, further interpreted as required features.

- **Requirements Conceptualization and Formalization.** Categorize and group the required features, build feature taxonomy by elaborating subsumptions, part-whole relationships, feature inheritance, and memberships among the required features. Refine significance scores by accounting for their propagation through inheritance. Develop the feature taxonomy. Collect informal knowledge about the meaning of the required features and transform it to formalized structural contexts, further interpreted as requirements. Aggregate significance scores by giving account of the feature groupings in the requirements.

- **Ontology Evaluation.** Map the requirements to the appropriate ontology contexts. Compute positive and negative votes based on the: (i) similarities or dissimilarities revealed through context mappings; and (ii) aggregated significance scores. Compute the fitness of the ontology as the ratio of positive to negative votes. Make recommendations based on the votes for or against the most significant requirements.

The phases are further described in more detail.

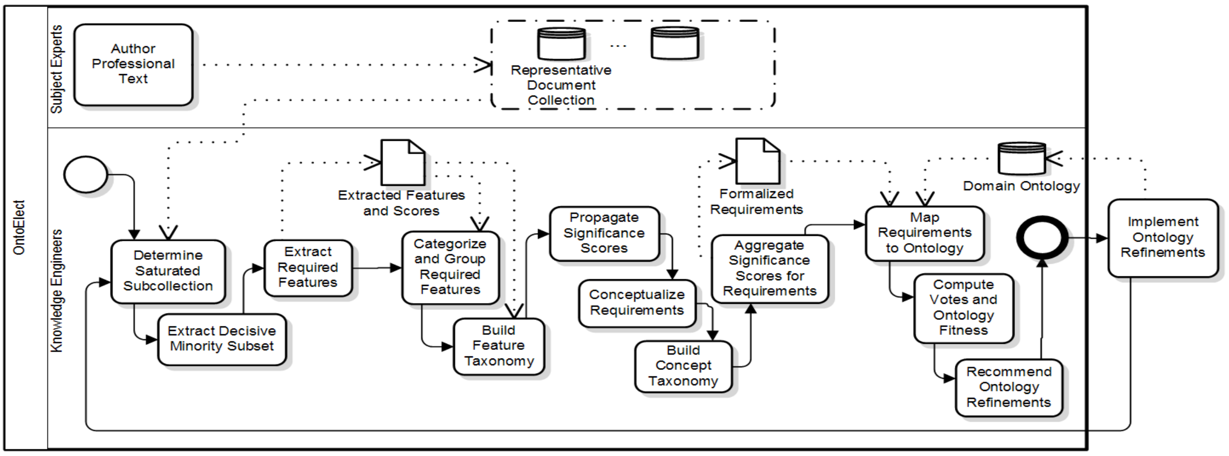

OntoElect
Subject Experts
Knowledge Engineers
Author Professional Text
Representative Document Collection
...
Determine Saturated Subcollection
Extract Decisive Minority Subset
Extract Required Features
Extracted Features and Scores
Categorize and Group Required Features
Build Feature Taxonomy
Propagate Significance Scores
Conceptualize Requirements
Build Concept Taxonomy
Formalized Requirements
Aggregate Significance Scores for Requirements
Domain Ontology
Map Requirements to Ontology
Compute Votes and Ontology Fitness
Recommend Ontology Refinements
Implement Ontology Refinements

Figure 1: OntoElect workflow, further elaborated from Tatarintseva et al. (2013). The phase of Implementing Ontology Refinements is pictured outside of the core workflows as it is beyond the scope of this paper

3.2 Feature Elicitation Phase

As already mentioned in Section 2, direct acquisition of requirements from domain experts is not very realistic as they are expensive and not really willing to do the work falling out of their core activity. In OntoElect, we focus on the indirect collection of the stakeholders' votes by extracting these from high quality and reasonably high impact documents authored by the stakeholders in a domain.

An important feature to be ensured for knowledge extraction from text collections is that a collection needs to be statistically representative to cover the opinions of the domain knowledge stakeholders satisfactorily fully. OntoElect suggests a method to measure the terminological completeness of the document collection *by analyzing the saturation* of terminological footprints of the incremental slices of the document collection. The full texts of the documents from the collection are grouped in datasets in the order of their timestamps. As pictured in Fig. 2a, the first dataset $D1$ contains the first portion *(inc)* of documents. The second dataset $D2$ contains the first dataset $D1$ plus the second incremental slice *(inc)* of documents. Finally, the last dataset Dn contains all the documents from the collection.

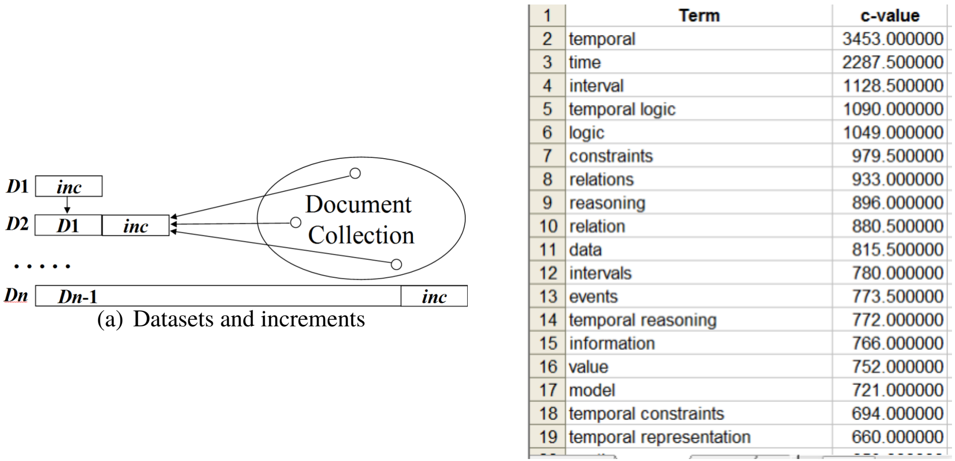
At the next step of the OntoElect workflow, the bags of multi-word terms $B1$, $B2$, ..., Bn are extracted from the datasets $D1$, $D2$, ..., Dn, using UPM Term Extractor[3] software, together with their significance (*c-value*) scores. Those scores correlate to a significant extent to term frequencies – i.e. how often a term was met in the dataset. Please see an example of an extracted bag of terms in Fig. 2b.

At the subsequent step, every extracted bag of terms Bi, $i = 1$, ..., n is processed as follows:

- Individual term significance threshold *(eps)* is computed to cut off those terms that are not within the majority vote. The sum of *c-values* having values above *eps* form the majority vote if this sum is higher than *1/2* of the sum of all *c-values*.
- The cut-off at *c-value* < *eps* is done
- Normalized scores are computed for each individual term: *n-score* = *c-value / max(c-value)*
- The result is saved in *Ti*

After this step only significant terms, whose *n-scores* represent the majority vote, are retained in the bags of terms. *Ti* are then evaluated for

[3] Java software for extracting terms and relations from scientific papers developed at Universidad Politechnica de Madrid in Dr Inventor EU project (https://github.com/ontologylearning-oeg/epnoi-legacy).



(a) Datasets and increments

1	Term	c-value
2	temporal	3453.000000
3	time	2287.500000
4	interval	1128.500000
5	temporal logic	1090.000000
6	logic	1049.000000
7	constraints	979.500000
8	relations	933.000000
9	reasoning	896.000000
10	relation	880.500000
11	data	815.500000
12	intervals	780.000000
13	events	773.500000
14	temporal reasoning	772.000000
15	information	766.000000
16	value	752.000000
17	model	721.000000
18	temporal constraints	694.000000
19	temporal representation	660.000000

(b) An example of an extracted bag of terms

Figure 2: (a) Incrementally enlarged datasets in OntoElect; (b) an example of a bag of terms extracted by UPM Term Extractor

saturation by measuring pair-wise terminological difference between the subsequent bags *Ti* and *Ti*+ 1 , *i* = 0 , ..., *n*-1. It is done by applying the THD algorithm by Tatarintseva et al. (2013). It is provided in Fig. 3 for reader convenience.

In fact, THD accumulates, in the *thd* value for the bag *Ti*+1, the *n-score* differences if there were linguistically the same terms in *Ti* and *Ti*+1. If there was not the same term in *Ti*, it adds the *n-score* of the orphan to the *thd* value of *Ti*+1. After *thd* has been computed, the relative terminological difference *thdr* receives its value as *thd* divided by the sum of *n-scores* in *Ti*+1.

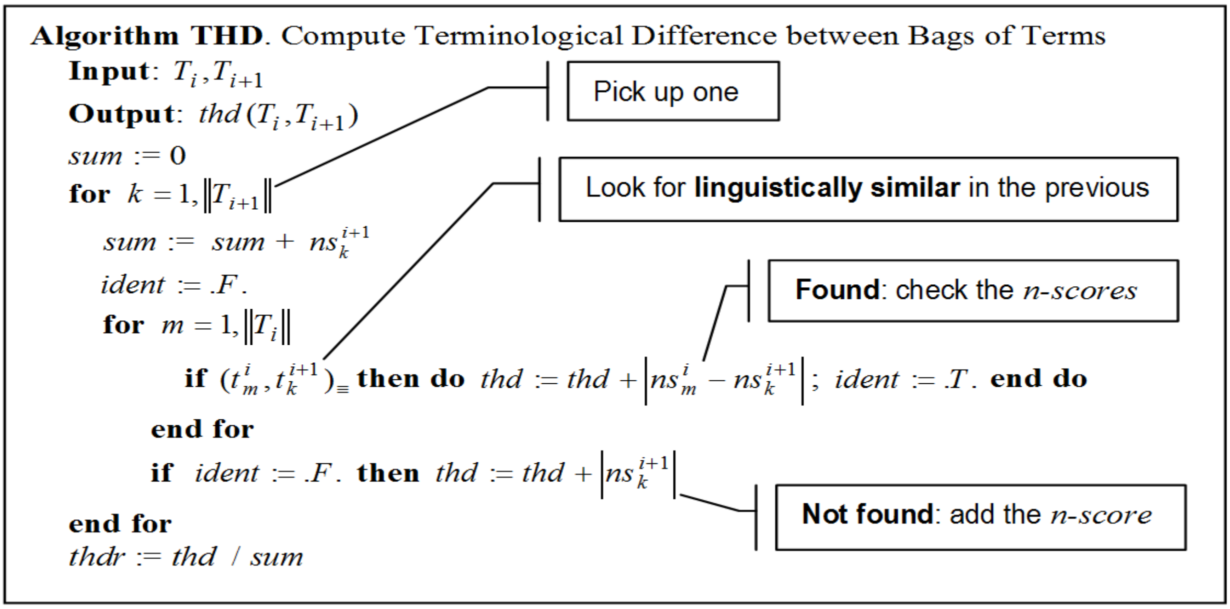
Absolute *(thd)* and relative *(thdr)* terminological differences are computed for further assessing if *Ti*+1 differs from *Ti* more than the individual term significance threshold *eps*. If not, it implies that adding an increment of documents to *Di* for producing *Di*+1 did not contribute any noticeable amount of new terminology. Thus, the subset *Di*+1 of the overall document collection may have become terminologically saturated. However, to obtain more confidence about the saturation, OntoElect suggests that some more subsequent pairs of *Ti* and *Ti*+1 are evaluated. If stable saturation is observed, then the process of looking for a minimal saturated sub-collection could be stopped. Sometimes, however, a terminological peak may occur after saturation has been observed in the previous pairs of *T*. Normally, this peak indicates that a highly innovative document with a substantial number of new terms has been added in the increment.

Additionally, the documents in the saturated sub-collection which have the major terminological impact on domain coverage are found out. As reported in Ermolayev et al. (2014), those are the most frequently cited documents. The numbers of citations for each paper are acquired from Google Scholar[4] using the Catalogue Generator software tool (Kosa et al. 2017). Citation frequencies *cfr* (the number of citations per year) are computed, and the impact of each paper in the collection is computed as:

$$imp = \begin{cases} [0.2 \times cfr] + 1, cfr > 0 \\ 0, cfr = 0 \end{cases} \qquad (1)$$

where the square brackets stand for taking integer part. Hence, the contribution of the frequency of citations to the impact of the paper is weighted by 0.2, while the papers having no citations are filtered out. The documents having their impact value *imp* > *threshold* form the decisive

[4] http://scholar.google.com/



```
Algorithm THD. Compute Terminological Difference between Bags of Terms
  Input: $T_i, T_{i+1}$
  Output: $thd(T_i, T_{i+1})$
  $sum := 0$
  for $k = 1, \|T_{i+1}\|$
     $sum := sum + ns_k^{i+1}$
     $ident := .F.$
     for $m = 1, \|T_i\|$
           if $(t_m^i, t_k^{i+1})_{\equiv}$ then do $thd := thd + |ns_m^i - ns_k^{i+1}|$; $ident := .T.$ end do
        end for
        if $ident := .F.$ then $thd := thd + |ns_k^{i+1}|$
  end for
  $thdr := thd \,/\, sum$
```

Pick up one
Look for **linguistically similar** in the previous
Found: check the *n-scores*
Not found: add the *n-score*

Figure 3: THD algorithm for computing terminological difference in a pair of bags of terms

minority sub-collection of the entire document collection. The *threshold* for filtering out documents from the decisive minority sub-collection is chosen empirically to ensure that the majority of the most significant terms are retained.

The Feature Elicitation Phase of OntoElect outputs the ranked list of the terms which represent the stakeholder sentiment about the domain. This list is extracted from the decisive minority sub-collection of the documents. Finally, the list is examined by a knowledge engineer and the irrelevant terms are withdrawn. These irrelevant terms may appear in the list because these may frequently appear in the collection and are often individuals in terms of ontology structure. For example, if the collection of conference proceedings is used for term extraction, the names of the authors of the frequently cited papers, affiliations, venues, names of the popular datasets used in experiments, etc. may receive high scores and be considered as significant. Among those mentioned above, only the names of the datasets may be somehow relevant for an ontology. The rest have to be withdrawn.

The terms in the cleaned list are further interpreted as the required features to be met by the ontology under evaluation or refinement.

To finalize this brief presentation of the OntoElect feature elicitation phase, it is worth noting that it is domain independent and unsupervised. However, the particular term extraction tool implies that it is able to process only English documents. To compensate this shortcoming, the processing pipeline is architectured in a modular fashion. Thus, it is possible to replace the term extraction tool by another one, for example for a different language.

3.3 Requirements Conceptualization and Formalization Phase

The task for this phase of OntoElect is to transform the ranked list of the required features to formalized ontological fragments (requirements), carrying the positive and negative votes of the involved features in their aggregated significance scores. Requirements are further used in Ontology Evaluation phase to compute the fitness of the ontology. Conceptualization and formalization in OntoElect are done by a knowledge engineer through:

- Grouping and categorizing extracted required features. Individual features could be interpreted as concepts, properties, or individuals.

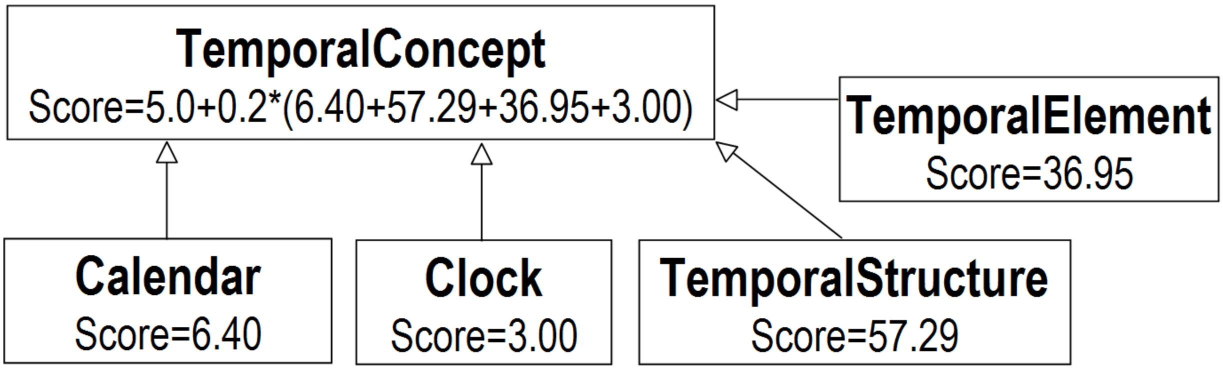


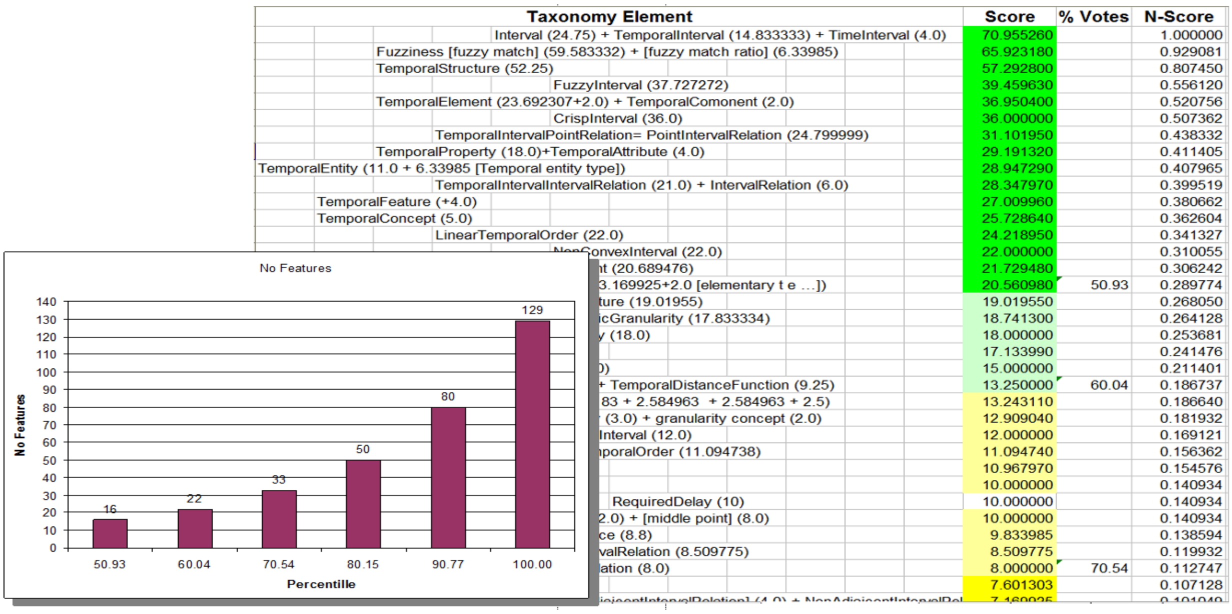
Figure 4: An example of computing score propagation for required features

- Selecting the significant concepts from the list of required features and forming the feature taxonomy, also including property features in the appropriate concept features
- Computing the propagated scores up the concept/property hierarchies
- Selecting the most significant concept features
- Elaborating natural language definitions for the most significant concept features and formalizing these as ontological fragments using UML and OWL
- Documenting requirements

Feature grouping is merging several features which are lexically different but carry equivalent semantics. The relevant cases include: plural and singular forms of the same term, for example "temporal constraints" and "temporal constraint" are the same terms and have to be merged; the terms that had or had not lost two-letter combinations because of the peculiarities of their representation in PDF documents due to the differences in Adobe versions, for example "de nition" and "definition" are also the same terms. The significance scores of the merged terms are added.

Feature categorization stands for deciding if a feature, due to its semantics, represents a concept, a property, or an individual. Concept features are further used to form subsumption or meronymy hierarchies in the feature taxonomy. The root of the feature taxonomy is the most abstract and general "thing" concept to which the rest of the concept features directly or indirectly subsume. For example, a temporal interval subsumes to temporal thing, etc. Meronymy hierarchies involve concept features which are either parts of a whole, like a weekend is the part of a week, or the wholes for their parts. Both types of these hierarchical relationships are important as they influence the significance of features through property inheritance. Indeed, if a feature subsumes to another feature then it inherits some of its properties – so its significance is formed to a particular extent by these inherited properties. Hence, a parent in a hierarchy may expect that it is rewarded by its children through the propagation of their significance scores. OntoElect suggests (Tatarintseva et al. 2013) that score propagation adds one fifth of the children' scores to their parent's score. An example of computing propagated scores is pictured in Fig. 4.

After propagating the scores in the feature taxonomy, the most important concepts in it, having the potential for high impact on the requirements due to their scores, may be selected. For that, concept features are viewed in a ranked list and the group of features covering the desired proportion of importance is promoted. An example of several (percentile) groups of concept features for the time domain is given in Fig. 5. The promoted concept features are used to form the concept taxonomy and be the central concepts



Taxonomy Element	Score	% Votes	N-Score
Interval (24.75) + TemporalInterval (14.833333) + TimeInterval (4.0)	70.955260		1.000000
Fuzziness [fuzzy match] (59.583332) + [fuzzy match ratio] (6.33985)	65.923180		0.929081
TemporalStructure (52.25)	57.292800		0.807450
FuzzyInterval (37.727272)	39.459630		0.556120
TemporalElement (23.692307+2.0) + TemporalComonent (2.0)	36.950400		0.520756
CrispInterval (36.0)	36.000000		0.507362
TemporalIntervalPointRelation= PointIntervalRelation (24.799999)	31.101950		0.438332
TemporalProperty (18.0)+TemporalAttribute (4.0)	29.191320		0.411405
TemporalEntity (11.0 + 6.33985 [Temporal entity type])	28.947290		0.407965
TemporalIntervalIntervalRelation (21.0) + IntervalRelation (6.0)	28.347970		0.399519
TemporalFeature (+4.0)	27.009960		0.380662
TemporalConcept (5.0)	25.728640		0.362604
LinearTemporalOrder (22.0)	24.218950		0.341327
NonConvexInterval (22.0)	22.000000		0.310055
nt (20.689476)	21.729480		0.306242
3.169925+2.0 [elementary t e ...])	20.560980	50.93	0.289774
ture (19.01955)	19.019550		0.268050
icGranularity (17.833334)	18.741300		0.264128
y (18.0)	18.000000		0.253681
	17.133990		0.241476
0))	15.000000		0.211401
+ TemporalDistanceFunction (9.25)	13.250000	60.04	0.186737
83 + 2.584963 + 2.584963 + 2.5)	13.243110		0.186640
(3.0) + granularity concept (2.0)	12.909040		0.181932
Interval (12.0)	12.000000		0.169121
nporalOrder (11.094738)	11.094740		0.156362
	10.967970		0.154576
	10.000000		0.140934
RequiredDelay (10)	10.000000		0.140934
2.0) + [middle point] (8.0)	10.000000		0.140934
ce (8.8)	9.833985		0.138594
valRelation (8.509775)	8.509775		0.119932
lation (8.0)	8.000000	70.54	0.112747
	7.601303		0.107128

Figure 5: An example of the groups of concept features per their importance. The upper group in the list contains 16 features which cover 50+0.93 percent of the accumulated significance (scores) of all the 129 retained features. The diagram to the left pictures the distribution of the features per percentile groups.

for the formalized requirements. Each of these promoted concept features is conceptualized in a formalized ontological fragment – as a conceptual model and a piece of code in an ontology specification language. Conceptualization means that all the relevant property features and features representing individuals are consolidated in the ontological fragment in a harmonized way to form a coherent piece of a required descriptive theory for the domain.

The subsequent step in conceptualizing a concept feature is elaborating its natural language definition based on the documents in the collection and probably some external resources of high reputation. The task of a knowledge engineer within this step is to ensure that all the required property features are taken into this definition and do not contradict each other. Based on the natural language definition, a conceptual model is developed for this concept feature, including also its properties and relationships to the other relevant concept features.

OntoElect does not enforce any specific working pattern or software tool for a knowledge engineer at this formalization step. In our development practices formalization is a two step process. The first step is updating the conceptual model coded as a UML 1.4 class diagram (Booch et al. 2000) using the ArgoUML editor[5]. Protege ontology editor[6] is used in the second step for coding the ontology in OWL 2 with an account for DL restrictions (Motik et al. 2012). The transformation patterns from UML to OWL follow the recommendations by Schreiber[7]

OntoElect is more specific in recommending a way for documenting the ontology under development. It suggests that the ontology is documented in a set of Semantic MediaWiki[8] pages. Some of those pages provide the overviews of the ontology modules, but the rest, which are the majority, are dedicated to documenting the concepts – one page

[5] ArgoUML is an open source UML modeling tool:http://argouml.tigris.org/

[6] Protege Ontology Editor: https://protege.stanford.edu/

[7] OWL Restrictions: http://www.cs.vu.nl/~guus/public/owl-restrictions/

[8] http://semantic-mediawiki.org/

per concept. A documentation wiki page of a particular concept contains:

- The natural language definition of the concept
- The UML class diagram of the concept's conceptual model
- The description of the concept's properties grouped according to the property types: datatype and object properties

3.4 Evaluation Phase

The objective of this phase is to figure out how well does an ontology (*O*) describe the domain meets the formalized requirements (*R*). This is done by mapping the requirements, as ontological fragments represented by their central concept features, to the semantically corresponding structural contexts within the ontology. The mappings reveal either similarity or dissimilarity and, therefore, either increase or decrease the fitness of *O*. To explain this with a little bit of rigor and present a way to visualize ontology fitness to domain requirements, an allusion of a gravitation field, proposed by Ermolayev (2015), will be further used.

Let us assume that a domain (*D*) is adequately modelled by the set of all relevant requirements (*R*). For building a grid based on these requirements it is assumed, as pictured in Fig. 6a, that:

- All the requirements are placed in the centre of *D*; and
- They are not equal in their significance – i.e. have different spheres of influence around the centre of gravitation, which is quantified using the normalized significance scores $ns \in [0,1]$

Let us suppose now that an ontology (*O*) is positioned in *D* at a distance l from its centre (Fig. 6(b)). This can be any location on the circle of radius l around the centre of the grid (Fig. 6(a)). Let us now reveal what might be the forces influencing *O* in this position.

Let us assume that *O* is checked against the requirements *r* from *R* which spheres of influence reach the position of *O (i.e.* $ns_r \geq l$*)*. The following are the possible outcomes of these checks:

- A particular part of *O*, say a semantic context $o \in O$ (a white coloured circle in Fig. 6(b)), meets the requirement *r*. Therefore, *O* becomes more fitting to *R*. In this case we will consider that the increase in fitness ($\Delta\Phi_0^+$) creates a positive gravitation force $\overrightarrow{G}_o^+$ applied to *O* and directed towards the centre of *D*, as pictured in Fig. 6(b). The absolute value of this force is computed using a direct analogy with the Law of Universal Gravitation by Newton (1999):

$$G_o^+ = \frac{1 \times \Delta\Phi_o^+}{(ns_r)^2}, \qquad (2)$$

  where: "1" in the numerator is the fitness of *r* with respect to *D* – meaning that *r* fits *D* perfectly as one of its requirements; the value of $\Delta\Phi_0^+$ is within [0,1].

- There is no semantic context $o \in O$ that meets the requirement *r* (no circle on the ontology side in Fig. 1(b)) or there is an *o* that contradicts *r* (a black coloured circle in Fig. 6(b)). In both cases *O* becomes less fitting to *R*. Therefore, we will consider that the decrease in fitness ($\Delta\Phi_O^-$ for a missing semantic context; $\Delta\Phi_o^-$ for a context contradictory to *r*) creates a negative gravitation force, $\overrightarrow{G}_O^-$ or $\overrightarrow{G}_o^-$ applied to *O* and directed towards the periphery of *D*, as pictured in Fig. 6(b). Similarly to (2), the absolute values of these forces are computed as:

$$G_O^- = \frac{1 \times \Delta\Phi_O^-}{(ns_r)^2}, G_o^- = \frac{1 \times \Delta\Phi_o^-}{(ns_r)^2} \qquad (3)$$

The overall gravitation force applied to *O* as an influence by *D* is computed as a vector sum:

$$\overrightarrow{G}_O\Big|_D = \sum_{r \in R: ns_r \geq l} \left(\overrightarrow{G}_o^+ + \overrightarrow{G}_O^+ + \overrightarrow{G}_o^-\right) \qquad (4)$$

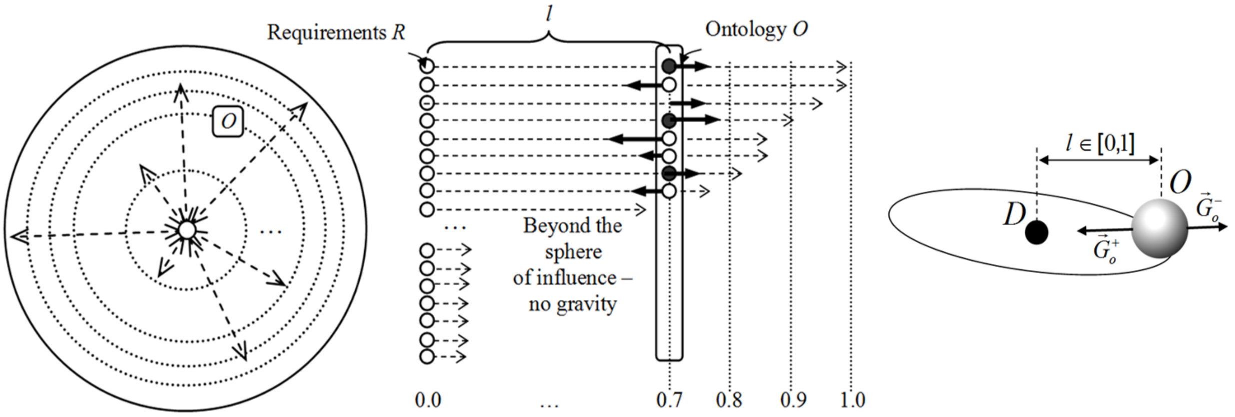


Figure 6: Domain requirements: (a) their spheres of influence; (b) gravitation forces; and (c) the equilibrium state of ontology O in D – adopted from Ermolayev (2015)

O is considered as properly positioned within *D* when it reaches its *equilibrium state* (Fig. 6(c)) with respect to the gravitation field in *D*, i.e. appears at a distance l from the centre of *D* at which $\overrightarrow{G}_O\big|_D = \overrightarrow{0}$. This distance could be interpreted as an integral measure of the semantic difference between what does *O* describe and what is required to be described for *D* by its knowledge stakeholders. If *O* is not in an equilibrium state regarding *D*, $\overrightarrow{G}_O\big|_D$ will cause it to move either towards the centre of *D* or towards its periphery.

Equivalence mappings are created to measure, based on significance scores, the degree of (dis-)similarity at the schema level between the required features and the elements in the ontology under evaluation. A mapping is a relationship between a concept feature and a concept in the ontology, or a property feature and the property in the ontology:

$$\mu(f, e) = \langle \equiv; l; e; r; score; n - score[; cf] \rangle, \qquad (5)$$

where: ≡ is the signature (equivalence mapping); f is the required feature; e is the corresponding ontology element; r is the ratio of similarity of f to e; *score* is the aggregated score of f; *n-score* is its normalized aggregated score to determine its sphere of influence; and cf is an optional confidence factor provided by the knowledge engineer and equal to 1 by default.

Each mapping (5) is a way to specify a positive vote in the sense of (2) or a negative vote in the sense of (3-4). Thus, the task of a knowledge engineer at this OntoElect phase is, for every formalized requirement, to specify the set of mappings of the features aggregated in this requirement to the elements of the evaluated ontology. When done, (s)he may compute the values of positive and negative votes and, further the value for the overall ontology fitness.

4 Evaluating OWL-Time against Time Domain Requirements

The progress in understanding the World and its data in their dynamics is based on having an adequately expressive model of time and, therefore, pushes forward the refinement of time models. The developments in Philosophy, Artificial Intelligence, Databases, Distributed Systems, etc. in the last two decades have brought to life several prominent theoretical frameworks dealing with temporal aspects. Some parts of these theories gave boost to research in logics – yielding a family of temporal logics comprising temporal description logics. Based on this foundation, knowledge representation languages have received their capability to represent time, and several ontologies of time have been implemented by the Semantic Web community. It is however important to find out if this plenty is enough to meet the demand in Semantic Data Management.

4.1 The Use Case of OWL-Time

One of the most widely used temporal ontologies is W3C OWL-Time initially developed by Cox et al. (2006) as W3C Working Draft dated 27 September 2006. Since that the ontology has been stalled as recognized by the Consortium. In its current shape, however, OWL-Time has noticeable shortcomings. For example, as articulated by the experts in the W3C Spatial Data on the Web Working Group (SDW WG)[9] , "... one of the shortcomings of OWL-Time is that it is unclear how to use OWL-Time in practice, especially how you query temporal data in ISO 8601 via OWL-Time." The members of the SDW WG committed to deliver a refined ontology, based on OWL-Time, in a year time frame.

The WG members, based on their expertise, also articulated and discussed the important requirements to this refined ontology. Among these requirements mentioned by the experts were:

- Non-Gregorian calendars
- Other (than currently in OWL-Time) time (TimeStamp) formats
- Approximate time instants (TimeStamp)
- Periods like Cretaceous period
- Leap seconds

Valid questions regarding this use case for OntoElect is if these requirements articulated and discussed by the SDW WG members are complete, accurate, and important.

As it is demonstrated in the subsequent sections, OntoElect allows answering these questions. Completeness is checked by comparing the list to the requirements elicited and conceptualized from the TIME paper collection presented in Section 4.2. The result of extracting required features from this document collection is presented in Section 4.3.

The TIME community has been chosen for the use case as the members of the SDW WG outlined the need to query temporal data via an ontology. Consequently, Time Representation and Reasoning looks like a very relevant community.

4.2 TIME Document Collection and Datasets

To assess the sufficiency of domain coverage, the consensual set of the features of time has to be extracted and appropriately structured. A way to do that is to analyze the document corpus produced by the appropriately chosen professional community and extract the required features from there – the TIME community (http://time.di.unimi.it/) in this case. The document corpus for required features extraction has been formed of the proceedings papers of the TIME Symposia series published by IEEE. The collection contained all the papers published in the TIME symposia proceedings between 1994 and 2013, which are 437 full text documents in total. The papers of this collection have been pre-processed manually, including their conversion to plain texts and cleaning of these texts. Accordingly, the resulting datasets were not very noisy. The datasets have been generated using Dataset Generator[10] module from the OntoElect Instrumental Toolset by Kosa et al. (2017). We have chosen the increment for generating the datasets to be 20 papers. Moreover, based on the available texts, we have generated 22 incrementally enlarged datasets $D1, D2, ..., D22$[11] . For generating the datasets the chronological order of adding documents has been used.

[9] Here and below in this section, we cite and use the facts from the minutes of the W3C SDW WG meeting on the 09 February 2016. The document is available at https://www.w3.org/2016/02/08-sdw-minutes#item07.

[10] The Dataset Generator is available at: https://github.com/bwtgroup/SSRTDC-PDF2TXT. More details, also on the other software modules of the OntoElect Instrumental Toolset, are given in Kosa et al. (2017).

[11] The TIME collection in plain text and the datasets generated of these texts are available at: https://www.dropbox.com/sh/64pbodb2dmpndcy/AAAzVW7aEpgW-JrXHaCEqg2Sa/TIME?dl=0.

4.3 Feature Elicitation

For extracting terms from TIME datasets, the UPM Term Extractor software has been deliberately chosen by Kosa et al. (2017) as an appropriate tool for automated term extraction from plain text with respect to measuring terminological saturation. The results of measuring terminological difference and detecting terminological saturation are presented in Table 1 and pictured in Fig. 7.

The saturation measurements **revealed stable saturation** starting from $D11 - D12$ – as presented in Table 1 by bold values and pictured in Fig. 7 by the vertical dashed line. The saturation curve has terminological peaks hinting about the appearance of documents with higher terminological contributions. Saturation is detected at *eps* equal to 23.774. The number of retained terms in $T12$ is 7110, which is only 2.47% of the total number of extracted terms in the corresponding bag of terms $B12$.

Following Ermolayev et al. (2014), we selected the decisive minority sub-collection using the information about the frequency of citations, as described in Section 3. The paper terminological impact *threshold* has been chosen as equal to 2. The decisive minority sub-collection contained 24 papers. A single dataset was formed of these 24 papers and terms extracted using UPM Term Extractor. The resulting bag of terms contained 686 terms after withdrawing the irrelevant entries and grouping.

4.4 Conceptualizing and Formalizing Requirements

For easier cleaning, the ranked list of retained terms has been classified as indicated in Table2[12]. Each term has been put into only one category.

This classification helped withdraw the terms attributed to the groups deemed as not fully relevant – all except Features. Furthermore, cleaning reduced the set of feature candidates to 175 items. The terms in the candidate list have been grouped – yielding 129 features.

The results of grouping are pictured in Fig. 5. For instance, it shows in the first row of the table that the feature of TimeInterval has been grouped by merging the features of Interval (with significance score of 24.75), TemporalInterval (14.83), and TimeInterval (4.0). Moreover, the significance score of a TimeInterval became 43.58. Significance score propagation has then been done for all 129 required features. Fig. 5 shows for example that the TimeInterval feature has received the additions in its score at least from FuzzyInterval (39.45), CrispInterval (36.0), and NonConvexInterval (22.0)[13]. After adding the propagated scores, the significance of TimeInterval became equal to 70.96, which made it the top scoring required feature.

As it may also be seen in Fig. 5, some features, like TimeInterval or TemporalStructure, could be categorized as concept features. Many other features, for example TemporalIntervalRelation or Fuzziness, read as properties and were categorized as property features. There were also a few features that read as individuals, those however were quite modestly scored in significance and were further neglected as not really important.

The taxonomy of temporal features has been further developed as shown in Fig. 8. The authenticity of the names was preserved from the above mentioned list (Fig. 5) to a maximal extent.

Already at this stage, it is possible to check if the requirements for refining OWL-Time discussed at the W3C SDW WG meeting (Section 4.1) are significant compared to the sentiment of the TIME community. The correspondences are presented in Table 3.

The analysis of Table 3 reveals that the requirements discussed by the experts in the SDW WG do not correspond to the most significant required features. The reason might be that these top ranking required features were already properly implemented in OWL-Time. This hypothesis will

[12] The complete table may be accessed at http://ermolayev.com/TimeOnto/ClassifiedTerms.zip.

[13] The other subsumed features are not visible in Fig. 5 – for example InfiniteInterval (12.0)

Table 1: Saturation measurements for the TIME bags of terms extracted by UPM Term Extractor

Datasets Pair	**No of Terms**		**Cut-off threshold *eps***	**Retained Terms *(c-value >eps)***	***thd*, value**	***thdr*,%**
	in the Bag of Terms *Bi*	**With *c-value* >1**				
empty-D1	53478	13775	28.000000	1379	112.240776	100.000000
D1-D2	91701	23816	24.000000	2473	72.425797	59.389624
D2-D3	114061	32419	21.500000	3028	24.265441	17.312132
D3-D4	129896	39643	19.651484	3997	32.879384	20.295700
D4-D5	145796	46702	19.651484	4466	32.622249	17.809632
D5-D6	162746	54629	20.000000	4587	44.646245	27.027091
D6-D7	190263	63684	21.000000	5133	38.071510	24.076680
D7-D8	200176	69097	22.000000	5413	26.869088	18.598430
D8-D9	217461	76315	22.000000	5855	18.776156	13.110501
D9-D10	245967	84664	23.219281	6453	26.914239	18.281013
D10-D11	263034	91132	24.000000	6428	24.164533	16.688847
D11-D12	287887	99231	**23.774438**	**7110**	**18.109566**	**12.737127**
D12-D13	298367	104398	23.774438	7383	12.573733	9.144105
D13-D14	320500	112898	24.000000	7723	13.334954	9.624406
D14-D15	333975	119787	23.774438	8298	14.403930	10.698614
D15-D16	350741	127257	24.000000	8426	16.428110	13.135633
D16-D17	369316	135085	24.000000	8877	9.642629	7.638542
D17-D18	389022	143452	24.000000	9617	11.416546	8.784302
D18-D19	399553	148896	24.000000	10005	8.042102	6.136623
D19-D20	420464	158179	24.000000	10574	11.655716	8.652365
D20-D21	435075	165519	26.000000	9751	9.781677	7.297311
D21-D22	449719	171135	26.000000	10139	6.926144	5.109224

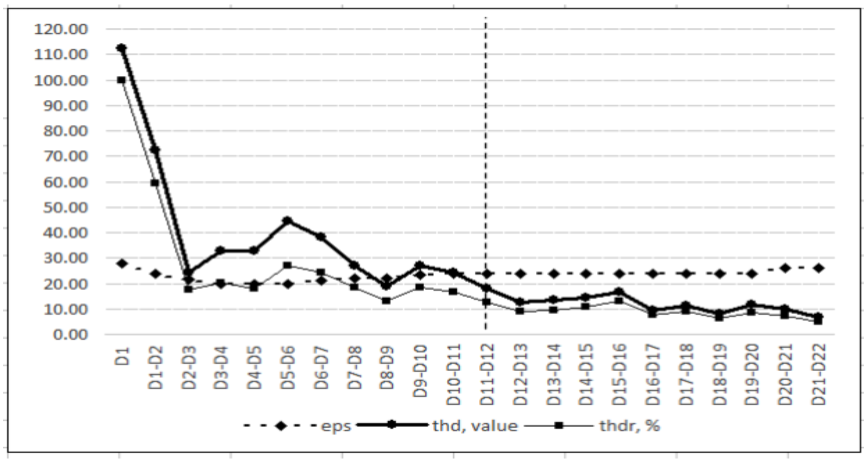
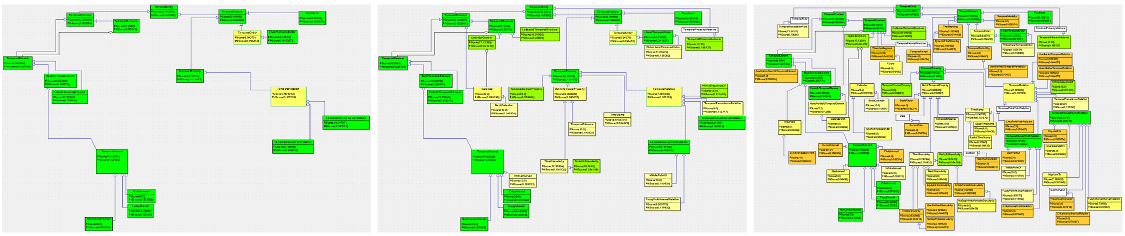


Figure 7: Saturation measurements on the TIME datasets based on the bags of terms extracted using UPM Term Extractor

Table 2: The first fourteen terms extracted from the decisive minority sub-collection dataset and their classification. The numbers under the categories indicate the total quantities of the terms under each category.

Score	Term	**Logic**	**Problem**	**Formula**	**Formalism**	**Operator**	**Method**	**Model**	**Reasoner**	**Domain**	**Language**	**Feature**	**Constraint**	**Instance**	**Pattern**	**Application**	**Project**	**Author**
	Total No of terms: **686**	44	27	6	36	8	22	24	1	4	8	175	28	1	13	110	1	178
147.11	temporal logic	✓																
100.11	calendar pattern														✓			
86.54	temporal constraint												✓					
68.73	temporal operator					✓												
59.58	fuzzy match											✓						
52.25	temporal structure											✓						
49.83	calendar schema											✓						
46.25	temporal representation				✓													
41.00	temporal reasoning						✓											
40.00	freeze quantifier				✓													
37.73	fuzzy interval											✓						
36.36	xml document															✓		
36.00	crisp interval											✓						
34.00	satisfiability problem		✓															



(a) 50 percentile – 16 features (b) 70 percentile – 33 features (c) 90 percentile – 80 features

Figure 8: Required feature taxonomy. The numbers of features in percentiles (a) – (c) and the colours of the included required features correspond to the diagram in Fig. 5.

Table 3: The correspondences between the required features extracted using OntoElect and requirements by W3C SDW WG

Required Features Extracted from TIME	Significance	W3C SDW WG Requirements
TimeInterval	70.96	
Fuzziness	65.92	
TemporalStructure	57.29	
FuzzyInterval	39.46	
TemporalElement	36.95	
CrispInterval	36.0	
PointIntervalRelation	31.10	
TemporalProperty	29.19	
TemporalEntity	28.95	
IntervalIntervalRelation	28.35	
TemporalFeature (high-level)	27.01	Approximate time instants
TemporalConcept	25.73	
LinearTemporalOrder	24.22	
NonConvexInterval	22.0	
PeriodicTemporalElement	21.73	Periods like Cretaceous period
BasicTemporalElement	20.56	
...		
TimeStamp	10.97	Other (than currently in OWL-Time) time (TimeStamp) formats
...		
Calendar	6.4	Non-Gregorian calendars
...		
Clock	3.0	Leap seconds
...		

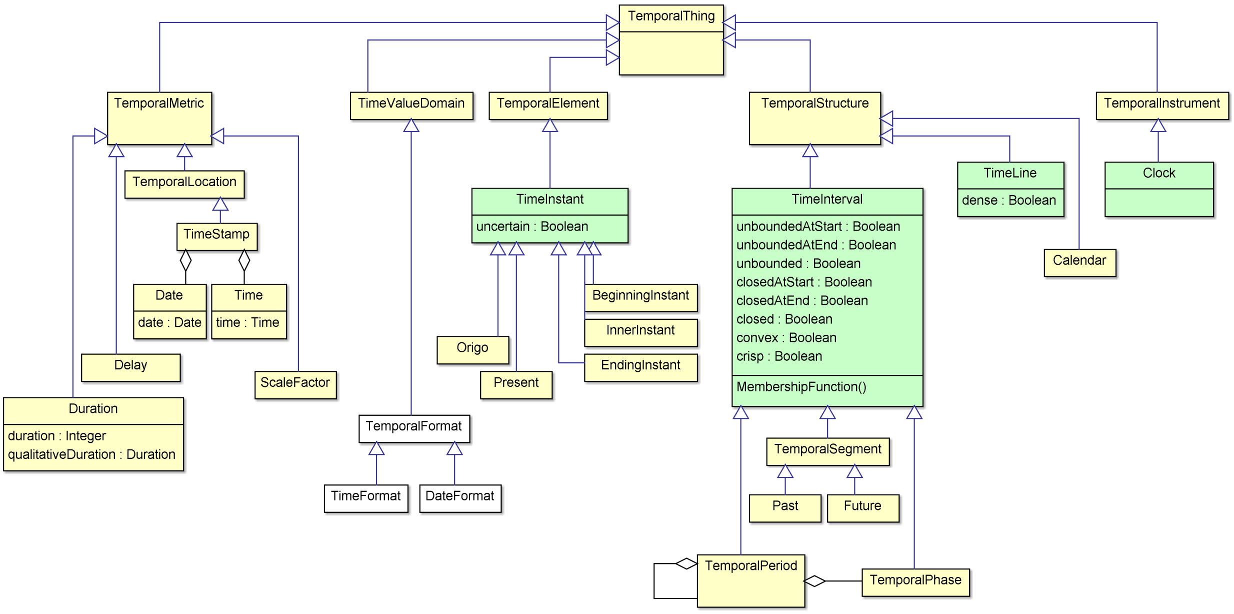


Figure 9: The concept taxonomy of TIME requirements. Significant concept features, that constitute, together with their subsumed hierarchies and aggregated property features, the 70 percentile (Fig. 8(b)), are coloured green.

be checked in Section 4.5 by building the equivalence mappings between the ontological fragments of OWL-Time and formalized requirements.

The hierarchy of the concept features presented in a tabular form in Fig. 5 has been then transformed into the concept taxonomy pictured in Fig. 9. This transformation also included adding the relevant property features to the concept features. Based on the groupings of the relevant features within the concepts of the taxonomy, ontological contexts for the most significant concept features (coloured green in Fig. 9) have been developed. This conceptualization work has been performed by:

- Developing a fragment of the descriptive theory of time around the concept feature and property features grouped with this concept
- Developing a conceptual model of this fragment as a UML class diagram
- Transforming the conceptual model to the formalized requirement specification in OWL (and SWRL in case a rule is needed to represent a feature)
- Documenting the requirement in a Wiki page

The results of these activities are illustrated below using the example of a TimeInstant concept feature.

The fragment of the descriptive domain theory elaborating the context of a time instant is presented below. It contains the basic definition of the concept feature, its place in the subsumption hierarchy, and also the description of the properties associated with this concept.

A time instant is a temporal element representing a point in time which has no duration. Having no duration is a qualifying feature of a time instant in difference to a time interval which has duration. From the other hand, a time instant is qualified by its position on the time line (temporal location), measured using the time stamp. There is only one time line on which a time instant is positioned. The time instants having equal timestamps but positioned on different time lines are regarded as different; before and after relations also do not work in this case.

An Origo is a specific kind of a time instant which, if exists for a particular time line, is placed at the beginning of times for this time line. Accodringly, there does not exist any other time instant on this time line which is positioned before the Origo. The segment of the Past on this time line is bounded at its beginning by the Origo.

A Present is a specific kind of a time instant which stands for now or current moment. By so saying, a Present is the boundary between the segments of the Past and Future, but not belonging neither to the Past, nor to the Future.

The basic property of a time instant is its (absolute) position on the time line with which the time instant is associated. This position is measured by the value (or the structured collection/bag of values) being the time stamp of the time instant. The format for the value / structured bag of values of a time stamp is specified by the time value domain.

A time stamp of a time instant may not be known. In this case, the (absolute) position of this time instant on a time line could be pointed to, with uncertainty, using qualitative temporal relations. For example, if it is known that an event occurred after 01/12/2017 and before 01/01/18 then the two time instants, t_a and t_b, with these timestamps, can be added to a knowledge base. A time instant t_e pointing to the temporal location of the event can then be asserted without time stamp, but having properties $after(t_e, t_a)$ and $before(t_e, t_b)$.

If a time stamp has a structure, this structure is commensurate to the available time units. The time units are chosen regarding to the precision (granularity) of the scale generated by the clock.

Relationships between the time instants on the same time line are specified to reflect their relativist positioning. Let t_1 and t_2 be any two time instants positioned on the same time line (T). Then, one and only one of the following three statements holds true reflecting the total linear ordering on the set of time instants: $before(t_1, t_2)$, $equals(t_1, t_2)$,

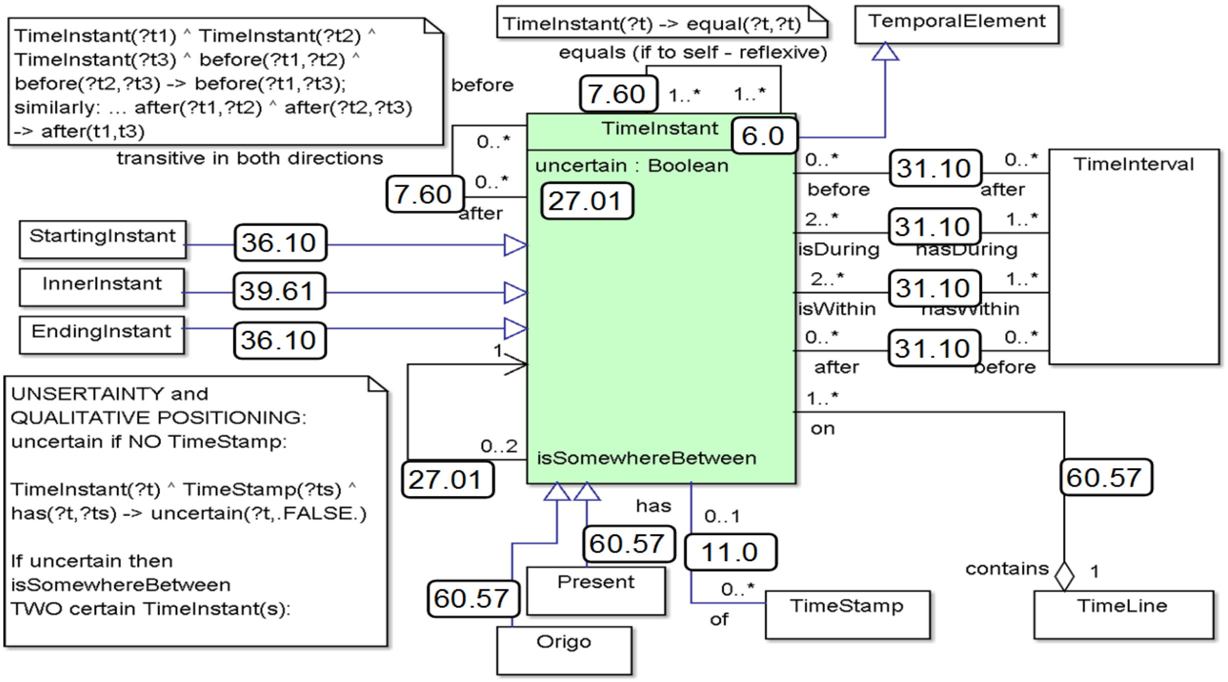


Figure 10: The UML model of a TimeInstant. Feature significance scores are shown as the numbers in rounded rectangles. SWRL rules associated with the properties as restrictions are shown as comments. The overall significance score of the TimeInstant requirement is 509.96.

$after(t_1, t_2)$. The total linear ordering imposes that the following hold true:

- Anisotropy:
  $\forall t_1, t_2 \in \mathrm{T} \;\; before(t_1, t_2) \leftrightarrow \; after(t_2, t_1)$
- Anisotropy: $\forall t_1 \in \mathrm{T} \;\; equals(t_1, t_1)$
- Transitivity:
  $\forall t_1, t_2, t_3 \in \mathrm{T} \; before(t_1, t_2) \wedge \; before(t_2, t_3) \rightarrow before(t_1, t_3)$
  $\forall t_1, t_2, t_3 \in \mathrm{T} \;\; before(t_1, t_2) \wedge \;\; after(t_2, t_3) \rightarrow after(t_1, t_3)$

It is considered that there could be no uncertainty in the relations between time instants. However, there could be uncertainty in the comparison of the timestamps of different time instants.

Relationships between the time instants positioned on different time lines can not be specified (directly) as there is no ordering established between the elements of different time lines.

Therefore, currently within this framework temporal relations can be inferred for time instants / intervals on the same time line. A rule / set of rules accounting for the above mentioned complications has to be set to infer if a time point on one timeline is before, after, or equal to a time point on the other time line in future work.

One possible way to reason about the relativist relationship between the time instants positioned on different time lines is to compare the values of their timestamps. This comparison is complicated by at least:

- The differences in the presence and relative positioning (offset) of the Origo points on these time lines
- The difference in the velocity of the time flow for different time lines
- The difference in the time units chosen for structuring the timestamps

The UML model of a time instant and the significance scores of its features are pictured in Fig. 10.

4.5 Evaluating the Fitness of OWL-Time to the Elicited Requirements

As suggested in Section 3.4, the mappings between the required features grouped in the formalized requirements and the corresponding elements of OWL-Time were specified for every significant requirement (Fig. 9): a TimeInstant and a TimeInterval. It was not possible to specify the mappings for a TimeLine and a Clock as these requirements were not implemented in OWL-Time. The mappings of TimeInstant are pictured in Fig. 11. In the figure, the mappings are given in a condensed form, without mentioning the names of f and e, as sources and tagets are easily identified by the arrows. Confidence factors are also not provided and are all equal to 1. Furthermore, as we are not interested in computing the gravitation forces in this paper, the spheres of influence of the features *(n-scores)* were neglected. The scores are given in round brackets as pairs of positive and negative votes.

The results of collecting votes and computing the fitness of OWL-Time to the elaborated TIME requirements are summarized in Table 4.

It may be stated that partially implemented features in an ontology raise more concern than the features that were not implemented at all. Indeed, if a feature has been implemented in part then the knowledge engineer has been made aware about the necessity of this feature by a requirement. Therefore, the missing bits of the feature semantics need to be added to the ontology. The features that are fully missing have not been required from the ontology before. Thus, it would be good to focus on the analysis of these features and the feasibility of their implementation.

As a straightforward recommendation, it may be mentioned that implementing the following five features with top significance scores will help to improve the fitness of OWL-Time by 233.41 points:

- Fuzziness – 65.92 + FuzzyInterval – 39.46
- TemporalStructure – 57.29
- TemporalFeature (high-level) – 27.01, including Uncertainty, (Un)Boundedness, Openness/ Closeness, Density (Discrete, Dense, Continuous)
- NonConvexInterval – 22.0
- PeriodicTemporalElement – 21.73

This will increase the fitness of OWL-Time to TIME requirements to 0.5768 by 17.97 percentage points.

It needs to be mentioned to conclude the discussion of this use case that the presented results have limited validity. Indeed, the fitness of OWL-Time was measured against the requirements reflecting the sentiment about time representation and reasoning by the TIME community. The results may have been different if another community and their representative document collection has been chosen.

5 Concluding Remarks

This paper showcased how the synergy of text mining, conceptual modelling, and ontology engineering techniques collected in one methodology helps transform the craft of ontology refinement to engineering. It reported on the use of the OntoElect methodology for evaluating the fitness of an existing ontology to the requirements of the knowledge stakeholders in the domain of Time Representation and Reasoning. It demonstrated, that a thorough routine for indirect elicitation, ensuring completeness, correctness of interpretation, using in ontology evaluation of these requirements is a must for Ontology Engineering.

In its motivating Section 2, the paper argued that both conceptual modelling and ontology engineering were in fact crafts to a substantial extent. The disciplines always claimed their careful attitude to the requirements of domain knowledge stakeholders. They did not however provide an objective and rigorous way to measure if: (a) all

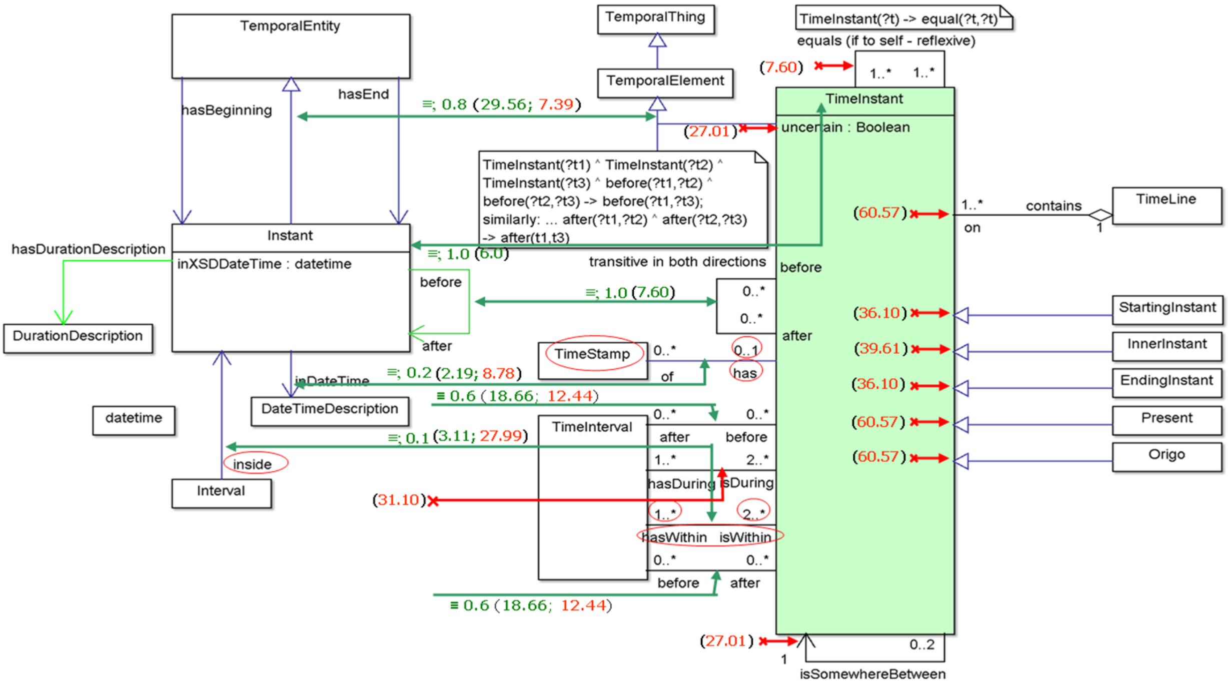


Figure 11: The mappings of the TimeTnstant requirement to OWL-Time

Table 4: Fitness of OWL-Time to the 4 most significant TIME Requirements

Key Element	**Fully Implemented Features**	**Partially Implemented Features**	**Missing Features**
	Cumulative Significance Count		
TimeInterval context (**1231.04**)	**351.59**	**97.05 / 103.15**	**679.25**
TimeInstant context (**509.96**)	**13.60**	**53.52 / 56.60**	**386.24**
TimeLine (TemporalStructure) (**57.29**)	—	—	**57.29**
Clock (TemporalDistanceMeasure + Clock) (**16.25**)	—	—	**16.25**
Total:	**515.76**		**1298.78**
Fitness:		**0.3971**	

significant requirements were put on the table; (b) all these requirements were correct or correctly interpreted; and (c) the final product met these significant requirements satisfactorily. After looking at the related work facing these important research questions, the paper arrived at understanding that a synergy of different techniques and approaches from Automated Term Extraction, Conceptual Pre-design, Ontology Evaluation is required to enable a coherent processing pipeline with enough maturity and rigour to answer these questions.

The paper reported on the research which has been inspired by the State–of–the–Art contributions in the above mentioned areas. The ideas and techniques that facilitated the development of OntoElect methodology in its current shape were:

- For an indirect approach to feature elicitation and answering the completeness question, a method to measure the terminological completeness of the document collection by analysing the *saturation* of terminological footprints of the incremental slices of the collection proposed by Tatarintseva et al. (2013).
- For ensuring the correct interpretation of the features as requirements, at least two ideas of KCPM by Kop et al. (2004) were valuable for reuse: (i) requirements are formalized to become closer to conceptual models; and (ii) requirements are focused on the terminology elaborated within the expert community in a domain
- For developing a metric of how well an ontology meets the requirements, the proposal by Gangemi et al. (2006) to measure ontology fitness for usability profiling appeared to be quite inspiring as it has been used to develop a gravitation-based technique to measure fitness against formalized requirements through the use of mappings.

The paper in its overview of OntoElect in Section 3 reported on how these motivating ideas have been further developed and refined in each of the processing phases of the methodology: Feature Elicitation, Requirements Conceptualization, and Ontology Evaluation.

With respect to the **feature elicitation** pipeline, the paper explained the technique used for discovering a saturated sub-collection of documents. Here, terminological difference *(thd, thdr, eps)* was used as a metric to detect saturation. It argued about the way to discover the documents with the highest terminological impact – the decisive minority sub-collection – using the citation frequency as the basic metric. It explained how to sublimate the extracted terms to the set of required features using the information about term significance in the form of numeric scores.

Regarding **requirements conceptualization and formalization**, the paper offered the sequence of activities to create formalized ontological fragments for requirements. It proposed a way to categorize and group the required features. Based on these groupings, it explained how to build the feature taxonomy using subsumptions, part-whole relationships, and memberships among the required features. Furthermore, it explained why and how significance scores of the required features ought to be refined by accounting for their propagation through inheritance. As a next step, the development of the feature taxonomy was suggested which was helpful for prioritizing the features based on their refined significance scores. The guidelines have further been given on how to group and aggregate the required features in the proper ontological fragment and in a harmonized way.

For **ontology evaluation**, the paper proposed to use the allusion of a gravitation grid and field for measuring the difference of the ontology to the requirements. Equivalence mappings, incorporating feature significance scores, were regarded as atomic gravitation forces denoted as positive and negative community votes with respect to the ontology. It has been proposed to measure the overall fitness of the ontology to the requirements as the ratio of positive to negative votes.

In Section 4, the methodology has been evaluated by applying it to measuring the fitness of the W3C OWL- Time ontology to the requirements elicited from the representative collection of research papers of the TIME symposia series. The three phases of the methodology were applied to this collection. As a result, it has been shown – in numbers – that OWL-Time meets the TIME community requirements only marginally. The paper suggested 5 major refinements to the ontology which may substantially increase its fitness. Interestingly, these proposed refinements differ noticeably from those elaborated by the experts in the W3C SDW WG by brainstorming.

Finally, for indicating the plans for the future work on OntoElect, it may be noted that the instrumental support for conceptualization and evaluation phases of the methodology is under development. Having this instrumental support will allow to lower the effort of a knowledge engineer in checking if (s)he really met the requirements of the knowledge stakeholders in the domain of the ontology under refinement.

At the time of writing, the W3C Recommendation of the Time ontology in OWL has appeared (Cox et al. 2017). Therefore, one more plan for the future work is to evaluate this updated revision against the TIME requirements.

6 Acknowledgements

The author would like to express his gratitude to Prof. Heinrich C. Mayr and the members of his Conceptual Predesign (KCPM) taskforce for the inspiration on the high-level idea of OntoElect – using stakeholder contributions to elicit requirements indirectly; and also for their approach to ensure that the collected requirements are interpreted correctly.

The research leading to this paper has been done in part in in frame of FP7 Marie Curie IRSES SemData project (http://www.semdata-project.eu/), grant agreement No PIRSES-GA-2013-612551.

The author would also like to acknowledge generous help of several colleagues who made achieving the reported results possible. Olga Tatarintseva contributed to the development of the OntoElect methodology at an early stage. Victoria Kosa helped quite significantly with extracting the required features. Dr. Andreas Harth helped a lot with choosing and substantiating the W3C SDW WG use case as the member of this expert group. Dr. Natalya Keberle helped in transforming the conceptual (UML) models to OWL 2 DL. Dr. Sotiris Batsakis and Prof. Frederic Mallet contributed their time and expertise in the discussions of the fragments of the descriptive theories of timelines, time intervals, time instants, and clocks. The author would also like to thank the anonymous reviewer who suggested to change ontologies to conceptual models to pass the test of time.

References

Booch G., Jacobson I., Rumbaugh J. (2000) OMG Unified Modeling Language specification

Burton-Jones A., Storey V. C., Sugumaran V., Ahluwalia P. (2005) A semiotic metrics suite for assessing the quality of ontologies. In: Data and Knowledge Engineering 535(1) October 2005, pp. 84–102

Chien L.-F., Chen C.-L. (2001) Incremental extraction of domain-specific terms from online text resources. In: Bourigault D., Jacquemin C., L'Homme M.-C. (eds.) Recent advances in computational terminology. Natural Language Processing Vol. 2. John Benjamins Publishing, Amsterdam/Philadelphia, pp. 91–109

Cox S., Little C., Hobbs J. R., Pan F. (2006) Time Ontology in OWL. W3C working draft.. Ontology Engineering Patterns Task Force. Semantic Web Best Practices and Deployment Working Group. World Wide Web Consortium (W3C). http://www.w3.org/TR/2006/WD-owl-time-20060927/

Cox S., Little C., Hobbs J. R., Pan F. (2017) Time Ontology in OWL. W3C Recommendation. OGC 16-071r2. Ontology Engineering Patterns Task Force. Semantic Web Best Practices and Deployment Working Group. World Wide Web Consortium (W3C). https://www.w3.org/TR/2017/REC-owl-time-20171019/

Ermolayev V. (2015) The law of gravitation for ontologies and domains of discourse. In: Computer Science Journal of Moldova 23(2), pp. 209–236

Ermolayev V., Batsakis S., Keberle N., Tatarintseva O., Antoniou G. (2014) Ontologies of time: review and trends. In: International Journal of Computer Science and Applications 11(3), pp. 57–115

Fliedl G., Kop C., Mayr H. C. (2005) From textual scenarios to a conceptual schema. In: Data and Knowledge Engineering 535(1), pp. 20–37

Gangemi A., Catenacci C., Ciaramita M., Lehmann J. (2006) Modelling ontology evaluation and validation. In: Sure Y., Domingue J. (eds.) Proc. ESWC 2006. LNCS Vol. 4011. Springer-Verlag, Berlin/Heidelberg, pp. 140–154

Gómez-Pérez A., Fernández-López M., Corcho O. (2004) Ontological engineering. Springer-Verlag, London

Kop C., Mayr H. C., Zavinska T. (2004) Using KCPM for defining and integrating domain ontologies. In: et al C. B. (ed.) Proc. WISE Workshops. LNCS Vol. 3307. Springer-Verlag, Berlin/Heidelberg, pp. 190–200

Kosa V., Chaves-Fraga D., Naumenko D., Yuschenko E., Badenes-Olmedo C., Ermolayev V., Birukou A. (2017) Cross-evaluation of automated term extraction tools. TS-RTDC-TR-2017-1. Dept of Computer Science, Zaporizhzhia National University, Ukraine

Motik B., Patel-Schneider P. F., Parisa B. (2012) OWL 2 Web Ontology Language. Structural specification and functional-style syntax (Second Edition).. http://www.w3.org/TR/2012/REC-owl2-syntax-20121211/

Newton I. (1999) The principia, mathematical principles of Natural Philosophy, a new translation. By I Bernard Cohen and Anne Whitman, preceded by ”A guide to Newton’s principia” by I Bernard Cohen. University of California Press

Pinto H. S., Tempich C., Staab S., Sure Y. (2004) DILIGENT: towards a fine-grained methodology for distributed, loosely-controlled and evolving engineering of ontologies. In: de Mántaras R., Saitta L. (eds.) 16th European Conf. on Artificial Intelligence (ECAI). IOS Press

Poveda-Villalón M. (2016) Ontology evaluation: a pitfall-based approach to ontology diagnosis. PhD thesis, Universidad Politécnica de Madrid

Presutti V., Blomqvist E., Daga E., Gangemi A. (2012) Pattern-based ontology design In: Ontology engineering in a networked world. Suárez-Figueroa M. C., Gómez-Pérez A., Motta E., Gangemi A. (eds.) Springer-Verlag, Berlin/Heidelberg, pp. 35–64

Schreiber G. (1999) Knowledge engineering and management: the CommonKADS methodology. the MIT Press, Cambridge, Massachusetts

Suárez-Figueroa M. C., Gómez-Pérez A., Motta E., Gangemi A. (2012) Ontology engineering in a networked world. Springer-Verlag, Berlin/Heidelberg

Sure Y., Staab S., Studer R. (2004) On-To-Knowledge methodology (OTKM) In: Handbook on ontologies Staab S., Studer R. (eds.) Series of Handbooks in Information Systems Springer-Verlag, Berlin/Heidelberg, pp. 117–132

Tatarintseva O., Ermolayev V., Keller B., Matzke W.-E. (2013) Quantifying ontology fitness in OntoElect using saturation- and vote-based metrics. In: et al. V. E. (ed.) Revised selected papers of ICTERI 2013, CCIS. CCIS Vol. 412. Springer-Verlag, Berlin/Heidelberg, pp. 136–162

Vrandecic D. (2010) Ontology evaluation. PhD thesis, Fakultät für Wirtschaftswissenschaften, KIT, Karlsruhe, Germany

Wong W., Liu W., Bennamoun M. (2012) Ontology learning from text: a look back and into the future. In: ACM Comput. Surv. 44(4) Article 20, 36 pages