1-1-2014

Computationally efficient ontology usage in software requirement planning

Robert B. K Brown Ph.D
University of Wollongong, bobbrown@uow.edu.au

Ghassan Beydoun
University of Wollongong, beydoun@uow.edu.au

Graham Low
University of New South Wales, g.low@unsw.ed.au

William Tibben
University of Wollongong, wjt@uow.edu.au

F Garcia-Sanchez
University of Valencia

See next page for additional authors

Computationally efficient ontology usage in software requirement planning

Abstract

Understanding the needs of stakeholders and prioritizing requirements are the vital steps in the development of any software application. Enabling tools to support these steps have a critical role in the success of the corresponding software application. Based on such a critical role, this paper presents a computationally efficient ontology selection in software requirement planning. The key point guiding the underlying design is that, once gathered, requirements need to be processed by decomposition towards the generation of a specified systems design. A representational framework allows for the expression of high level abstract conceptions under a single schema, which may then be made explicit in terms of axiomatic relations and expressed in a suitable ontology. The initial experimental results indicate that our framework for filtered selection of a suitable ontology operates in a computationally efficient manner.

Disciplines

Engineering | Science and Technology Studies

Publication Details

Brown, R. B. K., Beydoun, G., Low, G., Tibben, W., García-Sánchez, F., Zamani, R. & Martinez-Bejar, R. (2016). Computationally efficient ontology selection in software requirement planning. Information Systems Frontiers: a journal of research and innovation, 18 (2), 349-358.

Authors

Robert B. K Brown Ph.D, Ghassan Beydoun, Graham Low, William Tibben, F Garcia-Sanchez, Reza Zamani, and Rodrigo martinez-Bejar

This article was originally published as:
Brown, R. B. K., Beydoun, G., Low, G., Tibben, W., García-Sánchez, F., Zamani, R. & Martinez-Bejar, R. (2014). Computationally efficient ontology selection in software requirement planning. *Information Systems Frontiers: a journal of research and innovation*, doi: 10.1007/s10796-014-9540-3

Computationally Efficient Ontology Usage in Software Requirement Planning

R.B.K. Brown[1], G. Beydoun[1], G. Low[2], W. Tibben[1], R. Zamani[1], F. García-Sánchez[3], R. Martinez-Bejar[3]

[1]School of Information Science and Technology, Faculty of Engineering and Information Science, University of Wollongong, Australia

[2]Australian School of Business, School of Information Systems Technology and Management, University of New South Wales, Australia

[3]School of Computer Science, University of Valencia, Spain

ABSTRACT

Understanding the needs of stakeholders and prioritizing requirements are the vital steps in the development of any software application. Enabling tools to support these steps have a critical role in the success of the corresponding software application. Based on such a critical role, this paper presents a computationally efficient ontology selection in software requirement planning. The key point guiding the underlying design is that, once gathered, requirements need to be processed by decomposition towards the generation of a specified systems design. A representational framework allows for the expression of high level abstract conceptions under a single schema, which may then be made explicit in terms of axiomatic relations and expressed in a suitable ontology. The initial experimental results indicate that our framework for filtered selection of a suitable ontology operates in a computationally efficient manner.

Keywords: Requirements; Ontologies; Process Models; Retrieval Tool

1. Introduction

The historically poor success rate of information system projects has been often strongly associated with poorly gathered or understood requirements (Ellis & Berry, 2013). To develop acceptable, successful and usable information systems it is necessary to gather reliable requirements and use them to inform the design. To that end, gathered requirements should reflect both the problem state and desired aspects of the desired solution. Typically gathered from domain users (possibly experts), requirements are generally expressed in domain terms and, when gathered from multiple stakeholders, are not guaranteed to adhere to a single explanatory schema. It seems poor requirements are at least partially attributable to ineffective communications between analysts, designers and users (Dawson, 2012). In effect, as has recently been speculated (Brown & Piper, 2013), successful design involves an act of translation between user language and developer language.

Representing requirements in a commonly accessible nomenclature, often graphical, is a strong tradition in the requirements and development discipline. Expressing and translating stakeholder utterances into such forms requires training and experience in one or more notational frameworks (e.g. UML use case models, i* (Yu, 1997), KAOS (Dardenne et al., 1993), NRDR (Beydoun and Hoffmann, 1998). If that framework is not directly deployable at the detailed-design phase, then further translations may well be required, with each extra translation risking a loss of fidelity from the stakeholder's initial conception. Hence, requirement analysis starts from investigating clients' requirements and proceeds through abstraction. Abstraction in this context means breaking these requirements into the mental units that programmers can understand and usefully embody in software. However, experience suggests that rather than making the situation concrete and clear important information and relationships are lost in conventional approaches.

The purpose of this paper is to embrace the apparent ambiguity and multiplicity of requirements by using computing resources to compare possible solutions in order to identify a solution of 'best fit' as a starting point. These possible solutions come by way of existing ontologies. By 'ontology' we mean a formal and accurate hierarchic and taxonomic description of concepts related to each other in a tree-structure format. An Ontology is a structured representation of events and things in a domain (or 'conception'), which can carry with it behaviours and dynamic relations, captured in axioms. Each uses an internally standardised vocabulary and taxonomy to reduce ambiguity and facilitate machine reasoning. As many ontologies are quite specific to problem domains, there exist Library repositories from which analysts and requirements engineers may select one or more ontologies to suit the problem they're investigating. Choosing the best one(s) to fit and cover a target domain however can be an issue in itself.

For this purpose, Multiple Hierarchical Restricted Domain (MHRD) ontologies, employed by many authors (e.g. (Eschenbach and Heydrich, 1995; Beydoun et al 2006), are well understood and expressive for most domains. MHRD provide sets of inter-related concepts that are defined through a set of attributes, so the presence of axioms between these attributes is not considered. There can be part-of and taxonomic relations among the concepts so that attribute (multiple) inheritance is permitted. If accurate, this formal tree-structure can be used as a reliable knowledge representation scheme. The goal here is to best match user requirements with an ontology that most accurately represents the knowledge of the user as it

relates to the project at hand. As Casu et al. (2013) observed, an ontological approach reduces user bias, and increases abstraction whilst increasing expressive power and extendibility.

In practice the paper draws an area of study that uses ontological approaches and techniques drawn from the semantic web domain in business processes modelling (Veres et al., 2010; Dobson & Sawyer, 2006; Mayank et al., 2004). Semantic techniques have been seen as facilitators in formalizing the complex relationships between the involved entities of a business process model (Mueller, 2012), permitting a consistent representation of corresponding rules and intelligent queries. Mueller further identified the advantages in business process tuning and optimisation made possible by the formal ontologizing of the business process model. The paper benefits from Liang et al. (2011) work which offers guidelines to include human review to increase the reliability of ontology matching. It is based on these studies that this paper seeks to outline an efficient framework for confirming the filtered selection of an existing ontology to match and support requirements expressed in less formal techniques in a representational framework or even in natural language. .

In summary, the paper responds to the need for more effective rapid development approaches in requirements gathering and analysis. In a context, where detailed-design and construction workflows have deemphasised the role of initial analysis and conceptual design (Silva et al., 2011), the paper can be seen to address an important need. The rest of the paper is organised as follows: Section 2 presents the background and related work. Section 3 discusses how an ontology can be used to support requirements analysis and presents a retrieval tool to explore the computational issues in finding the appropriate ontology. Section 4 evaluates the various retrieval approaches described in Section 3 in an actual application domain. Section 5 concludes with a discussion of future work.

2. Background and Related Work

The arcane art of requirements capture and analysis must serve two masters. First, to cope with the different world views of users and analysts, and to adapt to the infinite variety of business scenarios; open, casual and easily understood notations and frameworks are needed. Second, translating user talk into analyst talk is a non-deterministic task.

To promote the design of a solution, an information system which through a higher degree of formality facilitates user *doings*, needs and desires is a major facilitator. In this regard machine executable functionality and machine reasoning across processes, normally best

expressed in natural language, can be implemented for a well crafted, or chosen, ontology framework. A well-engineered ontology can provide "natural language processing, reasoning capabilities, domain enrichment [and] domain validation" (Valiente et al., 2012). Also the formal representation of concepts and relations in any ontology permits the incorporation of formalized semantics into the corresponding business processes (Studer et al., 1998).

Originally an ancient branch of philosophy dealing with the nature and structure of reality, in the information sciences, the term Ontology has come to refer to a machine readable "formal, explicit specification of a shared conceptualisation" (Struder et al., 1998). In this, it is a structured representation of events and or things in some domain, attributing attributes as necessary, using (so far as possible) unambiguous terms and relationships. More than a taxonomy or class graph however, ontologies may also convey axiomatic assertions to describe behaviours as well as the legal transformations and interactions possible within the modelled domain. Guarino et al. (2009) offer the clarification that an Ontology is a set of axioms which set out a logical-theory to capture intended models of a conceptualisation. Further, they clarify the difference between a *top-level* Ontology that provides broad world-views applicable across multiple domains, and *reference* ontologies which define and disambiguate terms within a given domain.

There is, however, a lack of standards. No unambiguous or universal set of definitions, rationales or representations is adequate to serve the task of business process modelling (BMP) and engineering. Despite the rise of several excellent candidate frameworks, there are no generic solutions for matching an ontology to a business process map (Liang et al., 2011). There are also no clear agreed approaches or techniques for modelling with ontologies (Antoniou & van Harmelen, 2004; Allemang & Hendler, 2011). Valiente et al. (2012) observed the complexity and difficulties that arise in attempting to apply formal and machine-processable semantics to process models typically expressed in natural language and rich graphical notations.

Ontologies can gain value in re-use between problems and applications. Collections of ontologies, often domain specific, have become available. Such Libraries of ontologies limit the need to create bespoke ontologies for each new problem or application however the analyst may face challenges to; locate ontologies appropriate to their domain, determine which one(s) provide adequate conceptual coverage for their application and ensure that

selected ontologies are compatible with the technologies and protocols being used (Noy & d'Aquin, 2012).

Ko et al. (2009) and van der Aalst (2013) have surveyed and documented a sizable and rich variety of business modelling standards. Mueller (2012), somewhat despondently interpreted the breadth of BPM choices that has evolved, as a 'jungle'. In a significant step towards disambiguation, van der Aalst (2013) offers a taxonomy of three business process language categories. This includes formal, conceptual and execution languages. Moreover, (Grolinger et al. 2014) notes that the conceptual category, which encompasses the conveniently flexible and often graphical notations, lacks rigorous semantics.

As noted above, an informal and flexible approach suits translation between domains and facilitates achieving consensual understanding of intuitive requirement modelling. These are valuable traits when dealing with business stakeholders whose expertise resides in their own business domains, rather than that of BPM methods. The same flexibilities and informalities, however, may impede ready translation or matching of a business process mapping to a more formally structured ontology.

2.1. Research Methodology and Limitations

The research underpinning this paper was conducted using the Design Science methodology under which we explore the functioning of an artefact in some particular environment to address an identified problem. Under this framework we create and build an artefact to address a perceived requirement and ask does it work. The construction demonstrates feasibility of the artefact which then becomes the object of study. Such evaluative study however, is limited by the scope of the metrics deployed to define performance goals (March & Smith, 1995). In a novel scenario, the performance of novel or innovative artefacts can be difficult to define for want of comparison artefacts.

Comprehensive evaluation of an artefact's utility and the value it may add may require a series of methodological approaches (Wang et al., 2011). Demonstrating utility under the nature of the problem under investigation within the scope of this study however is best demonstrated with a case study. We acknowledge however that extensive testing with further case studies and larger volumes of quantitative data would be of benefit in the continuing study of our proposed method artefact.

3. Ontologies as a Requirements Bedrock

An ontology comprises a simplified and reduced vocabulary which defines the problem scenario together with a set of axiomatic rules describing relations between and among these entities and concepts (Uschold and Grüninger,1996). Expressing requirements in this way facilitates machine processing. The Web Ontology Language (OWL) (Smith et al., 2004) permits class-based representation of individual entities, where objects within the domain are instances of a class. Binary relations between individuals capture axiomatic rules. Class, property and instance level machine based reasoning is possible within the framework Valiente et al. (2012).

Mueller (2012) has reported that BPM experiments with the OWL and the SPARQL Protocol and Resource Description Framework (RDF) Query Language (SPARQL) (W3C, 2008) were 'promising'. Hsu (2013) found RDF/XML's focus on syntax and format a limitation however, lacking mechanisms to address semantics and knowledge. Casu et al. (2013) observed that SPARQL's semantic capacity to evaluate graph patterns over RDF graphs was a significant point of difference from otherwise similar looking SQL queries. Verma et al. (2005) observed that formal semantics are required to permit reusability of business service processes, whilst interoperability requires formal data descriptions (Nagarajan et al. 2006).

Savvas and Bassiliades (2009) suggested an OWL ontology for business administration, mapped into the semantic-markup variant OWL-S (W3C, 2004). The use of ontology to facilitate transformation between business process representations has been emphasized in Norton et al. (2009) and such an ontology has been characterised as a *transformation ontology*. Later, Liang et al. (2011) proposed OWL-BPC, an OWL based Extensible markup Language (XML) language specifically for business processes customization.

As Grolinger (2014) observed, however, the less formal business process representations do not offer clear categorisation in the manner required in a workable ontology. Even with expert guidance in an initial anthologizing activity, finding the best fit among the myriad ontologies available is a non-trivial and time-consuming task. This necessitates the computationally efficient matching framework presented in the current work.

3.1. Tool Support to Locate Supporting Ontologies

We have used Stanford University's open-source ontology editing package Protégé (http://protege.stanford.edu/), to implement the corresponding ontology. The Protégé screen for task dependency is shown in Figure 1. The implementation is described below, in which two major stages can be identified.

During the first stage, the business process model (i.e. requirement model in our context) is transformed into an ontology as advocated in (Beydoun et al 2014; Tran et al 2007), and during the second stage, this initial ontology is compared with the domain ontologies in a repository, and the closest ones in the domain are returned. Whilst the first stage is currently carried out manually by an expert, a software tool has been developed to deal with the second stage. The architecture of the tool is towards obtaining the domain ontologies that best matches with a given (ontologized) model.

In the second stage, the mechanisms to make the comparison between the two ontologies can range from a mere syntactic approach, based on keyword matching, to a full semantic approach by taking advantage of the formal underpinnings of ontologies. That is why the architecture that constitutes a development support tool is composed of three main components: the *matchmaker*, the *persistence manager* and the *query handler*. As mentioned, the system receives an initial ontology as input and returns some of the ontologies that are stored in the ontology repository as output, with the input ontology containing a formalized representation of the early requirements expressed in the corresponding model. The returned ontologies are determined based on the co-operation of the three aforementioned components and are those among existing ontologies which have most in common with the input ontology. The details for of each of the components are as follows.

The Matchmaker: This module loads the domain ontologies that are stored in the ontology repository by means of the persistence manager, which will be discussed in detail. It then assesses each ontology in terms of the similarity level with respect to the initial ontology, with making use of the query handler, which will be discussed in detail. The input of this module is the initial ontology with the system's early requirements. The output is the set of domain ontologies that exceed a given threshold. At this point, different policies can be applied, including (i) returning only one ontology (the one with the highest score), (ii) returning a given number of ontologies (e.g. the five best matches), or (iii) returning all the ontologies whose score is over a given threshold.

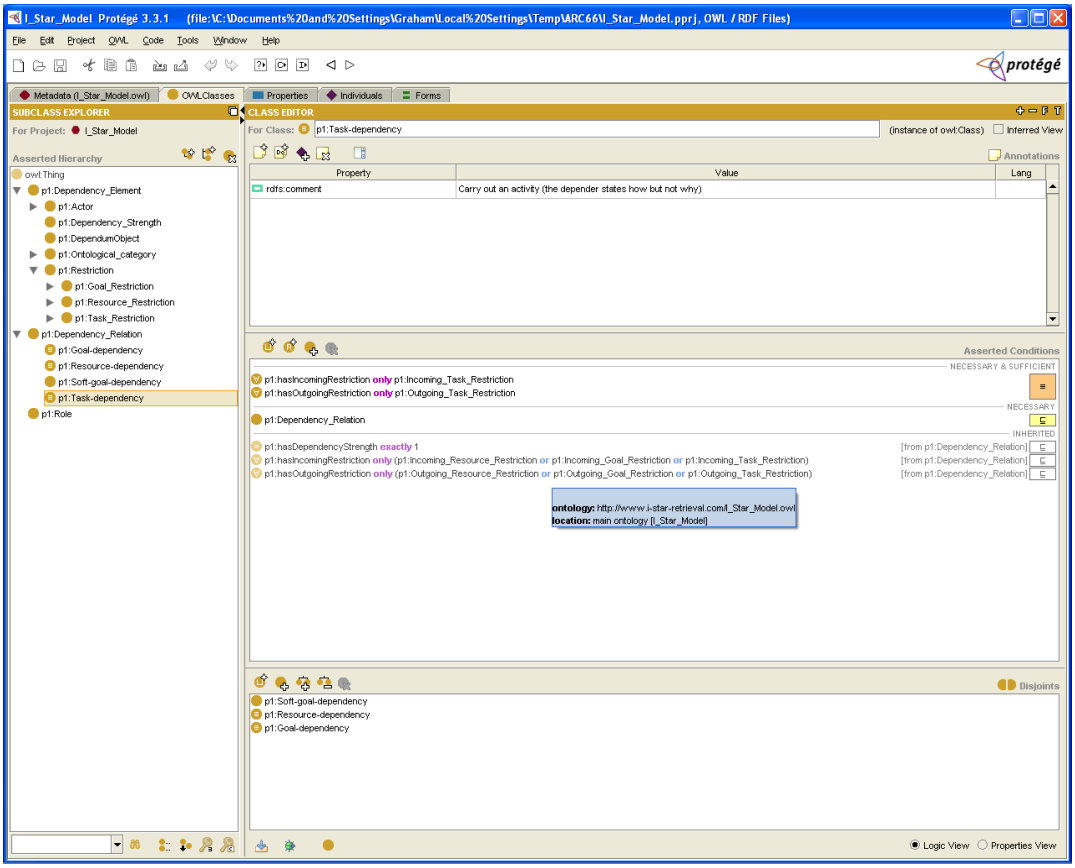

Figure 1. Protégé screen for task relationship

The Persistence Manager: This module makes use of the Jena Framework (McBride, 2002) to retrieve the initial ontology that is stored in a particular location on the hard disk. Two approaches for managing the domain ontologies have been evaluated, namely (i) using Jena to load these ontologies into memory from the hard disk, and (ii) storing the ontologies in the Sesame RDF repository (Broekstra et al., 2002). In the second approach, the repository is backed up by a MySQL database and accessing is provided through the Java API. Jena enables the developer to work with the document at the ontology level (OWL) while, with Sesame, the developer must work with the ontology at the RDF level, which is not as powerful as OWL. However, the drawback of the first approach is that having to load all the ontologies into memory reduces its scalability. With Sesame, on the other hand, it is possible to issue SPARQL queries to the repository without having to load the ontologies, which makes it perfectly suitable for our purposes.

The Query Handler: This module provides three different approaches for assessing the similarity between ontologies. Since the factory method pattern (Gamma et al., 1995) has been used to this end, the matchmaker can choose what method to apply to compare the

ontologies at run time with minimal effort. The first approach is *"keyword-based filtering"*. It consists of providing a set of keywords to be matched against the keywords used to describe the model. Despite the fact that it is the most efficient approach, it seems to be the less accurate among the others (Rao et al., 2012). The second approach is *"controlled vocabulary filtering"* (Macgregor & McCulloch, 2006). It makes use of controlled vocabularies with explicit, formal semantics. Ontologies are prominent conceptual means for this purpose. This approach is well-balanced in trading efficiency with accuracy. The final approach is *"semantic matchmaking"* (Di Noia et al., 2007). Based on rich semantics to define the compared models, this approach is the most accurate but the less efficient approach. In its current state, the proposed system makes use of a "simple semantic" approach. Whereas the elements that are being compared are in the form of ontologies, these ontologies are just exploited as controlled vocabularies.

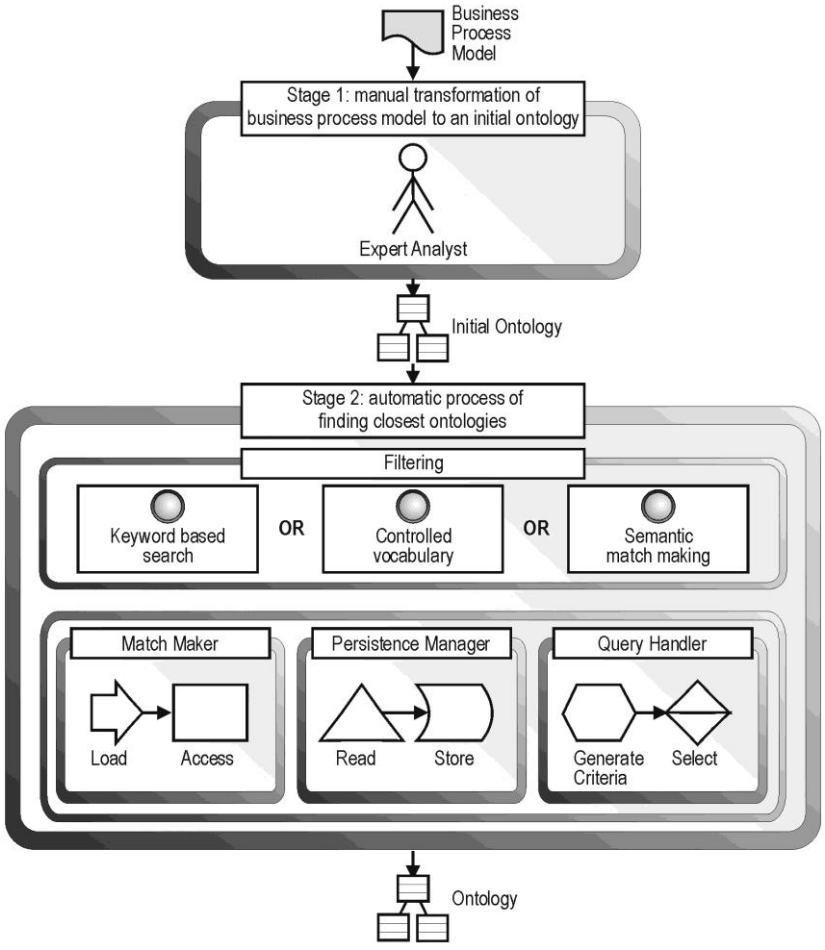

Figure 2. Architecture of the proposed system

As a first step, the system gathers the keywords available within the requirements and compares them with the contents of the domain ontologies. Each domain ontology obtains a score that is determined by the number of occurrences of the selected keywords in the domain ontology under question. Let's suppose O is the input requirement models and DO is one of the domain ontologies stored in the repository. The architecture of the system used to assess the similarity between the requirement models and each domain ontology is shown in Figure 2.

4. Case Study: the meeting scheduler

In this section, the way the proposed methodology and tool can assist in retrieving relevant domain ontologies in a meeting scheduler scenario is described. Its computational efficacy is then evaluated. The requirement models are first converted into a structured form to enable systematic identification of keywords. In this paper, the requirement models themselves are converted into an ontology. In Figure 3, a portion of the ontology that represents the requirements of an application known as the 'meeting scheduler' is depicted. Three actors are involved in the requirements, namely (i) the 'Meeting Initiator', (ii) the 'Meeting Participant', and (iii) the 'Important Participant'. A total of seven relations have been identified among goals, resources and tasks. Each dependency relation has both incoming and outgoing restrictions. Three dependum objects were also considered closely related to the actors identified. The ontology also included ten different tasks, four resources and eleven goals.

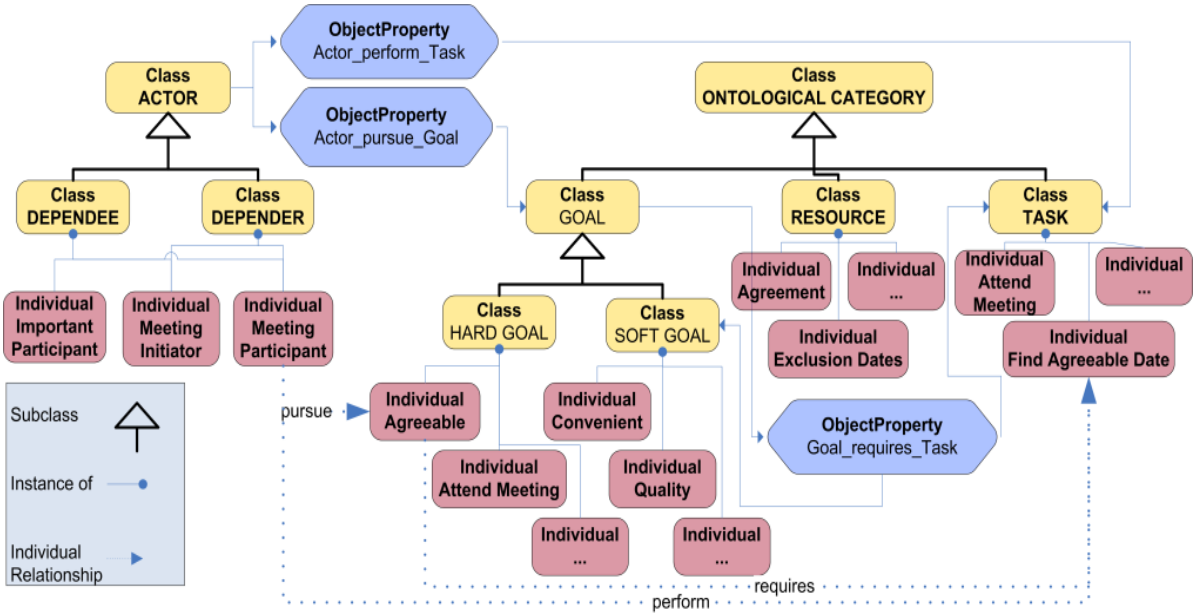

Figure 3. Excerpt of the meeting scheduler initial ontology

The tool that compares ontologies is initiated, and the initial ontology representing the models is matched against a set of sample domain ontologies. By following the manual *expert* procedure, the model with the early requirements for the meeting scheduler scenario is translated into an ontology.

For testing purposes in this case study, a set of ten random, non-related domain ontologies were stored in the ontology repository. The chosen ontologies vary in both size and in the nature. It is worth mentioning that the tests have been performed using an Intel Core 2 Quad 2.40GHz processor with 3 GB RAM.

Table 1. Description of the ontologies in the repository.

Ontology	Scope	Metrics
Agenda-ont.owl	Meeting agenda ontology	8 classes, 3 object properties, 11 data-type properties and 32 restrictions
Cyc.owl	OpenCyc is the open source version of the Cyc technology, the world's largest and most complete general knowledge base and commonsense reasoning engine	2948 classes, 1243 object properties, 2 data-type properties and 7573 individuals
e-commerce.owl	Elements concerning commercial transactions	20 classes, 7 object properties, 7 data-type properties and 7 restrictions
Event.owl	This ontology describes concepts for modeling events in an intelligent meeting room environment.	12 classes, 28 object properties and 2 data-type properties
Finances.owl	An ontology about the stock exchange domain	147 classes, 20 object properties, 38 data-type properties and 139 restrictions
Jobrecruitment.owl	An ontology for the employment domain describing applicants' profiles and employers' offers	69 classes, 14 object properties, 50 data-type properties and 5 individuals
Otasks.owl	It represents information about events that take place in an office	524 classes, 67 object properties and 148 data-type properties
Pizza.owl	An example ontology that contains information regarding the elaboration of pizza	99 classes, 10 object properties, 4 data-type properties and 5 individuals
Portal.owl	The ontology represents the knowledge used in the CS AKTive Portal testbed: people, projects, publications, geographical data, etc.	169 classes, 108 object properties, 29 data-type properties and 75 individuals
Travel.owl	An example ontology for tutorial purposes about tourism related issues.	30 classes, 15 object properties, 25 data-type properties and 50 individuals

The comparison tool was executed to evaluate the similarity between the initial ontology described in the previous section and the ontologies in the repository. The two aforementioned approaches of JENA and Sesame for managing the ontologies in the repository have been separately tested under identical experimental conditions. The results of the experiments are shown in Table 2.

Jena and Sesame yield similar outcomes in terms of the similarity with respect to the provided initial ontology. Slight differences can be noted in the final score for each ontology. This is due to the ability of JENA to load the ontologies that are imported by the ones it meant to access. Moreover, a significant increase is noticeable in the time required by the Sesame approach. The other point worth mentioning is that enabling the tool to access remotely hosted RDF repositories backed up by relational databases seems to help the system to gain further effectiveness at the cost of losing processing speed.

Table 2. Results of the experiments.

Approach	**JENA**	**Sesame**
Time (ms) / *Ontology*	**45.203**	**120.797**
Agenda-ont.owl	0.27 (5)	0.34 (5)
Cyc.owl	1.00 (1)	0.94 (1)
e-commerce.owl	0.09 (7)	0.00 (8)
Event.owl	0.68 (3)	0.37 (4)
Finances.owl	0.09 (8)	0.03 (7)
Jobrecruitment.owl	0.23 (6)	0.14 (6)
Otasks.owl	0.73 (2)	0.74 (2)
Pizza.owl	0.09 (9)	0.00 (9)

Portal.owl	0.66 (4)	0.54 (3)
Travel.owl	0.04 (10)	0.00 (10)

5. Conclusion and Future Work

There is a long history of significant waste in information technology (IT) projects (Keil et al., 1998) and any improvement in the success rate would yield considerable savings. A high failure rate for IT projects has been well documented over an extended period (Standish, 1995; OASIG 1995; Cooke et al., 2001; Ellis, 2007 and Crear, 2009) which routinely results in significant economic loss across the entire IT industry. Poor collection and analysis of client requirements in the earliest phases of an IT Project has been identified as a significant cause of IT project failure. (Whittaker 1999, Tichy and Bascom 2008). In the requirement phase of any software project, initial requirements can be often ambiguous, incomplete, inconsistent and usually expressed informally. An improvement to the efficiency of selecting appropriate ontologies can only positively impact the cost of IT project development.

It has been argued that model-driven techniques can reduce development and maintenance costs (Hermida et al., 2013) and that ontologies are a key component of such techniques. An Ontology can provide the common language for argumentation needed to facilitate the operation of heterogeneous agents across a given domain (Heras et al. 2014). In the field of distributed development, where communications gaps may emerge (especially under rapid and agile approaches) the use of well-chosen ontologies for modelling user requirements can assist in detecting and resolving ambiguities and contradictions. The agile approach can benefit significantly from this mechanism for rapidly testing requirements in some automatable manner (Carrera et al. 2014).

To enable automatic processing of initial requirements at the knowledge level, we organized them into an ontology. A tool is then used to retrieve related ontology that can be used to enhance the quality of the requirements. This paper evaluated the computational efficiency and effectiveness of the tool. The architecture of the presented tool is towards obtaining the domain ontologies that best matches with a given (ontologized) model.

The operations are performed in two stages, and automatic processing is performed in the second stage. Whereas during the first stage, the business process model is manually

transformed into an ontology, during the second stage, this initial ontology is compared with the domain ontologies in a repository, and leads to returning the closest ones. Despite the fact that the initial ontologies may be ambiguous, incomplete and/or inconsistent, the early requirements, expressed informally as a business process map. Our use of an expert manual translation generated a consistent formal ontology with index to those initial ontologies. This was performed by defining formal ontology with an index back to initial informal ontologies, preserving the mapped relationships.

In future work, we will evaluate the quality of the requirements-based retrieval ontology and adapt the retrieval function accordingly. In (Beydoun et al 2013) we formulated the guidelines to evaluate MHRD ontologies for this purpose. The retrieval function will become a set of various retrieval functions. Out of this set, one will be chosen according to examine both the quality of the ontology, as well as some key features of an application domain.

References

Allemang, D., & Hendler, J. (2011). Semantic Web for the Working Ontologist (2nd Ed.). Morgan Kaufmann

Antoniou, G., & van Harmelen, F. (2004). A Semantic Web Primer. MIT Press.

Beydoun, G. and Hoffmann, A. (1998). Simultaneous Modelling and Knowledge Acquisition using NRDR. 5th Pacific Rim Conference on Artificial Intelligence (PRICAI98), Singapore, Springer-Verlag.

Beydoun, G. and A. Hoffmann. Building Problem Solvers Based on Search Control Knowledge. *11th Banff Knowledge Acquisition for Knowledge Base System Workshop*, Calgary, SRDG publications, pp Share1/1-Share1/16 (1998)

Beydoun, G. *Incremental Knowledge Acquisition for Search Control Heuristic*. PhD thesis, School of Computer Science and Engineering, University of New South Wales, 223 (2000)

Beydoun, G., Tran, N., Low, G. and Henderson-Sellers, B. (2006) Foundations of Ontology-Based Methodologies for Multi-agent Systems. *Proceedings of AOIS2005* (eds. M. Kolp, P. Bresciani, B. Henderson-Sellers and M. Winikoff), LNAI 3529, Springer-Verlag, Berlin, 111-123

Beydoun, G., García-Sánchez, F., Vincent-Torres, C.M, Lopez-Lorca, A.A., and Martínez-Béjar, R. (2013), Providing metrics and automatic enhancement for hierarchical taxonomies, *Information Processing and Management*, 49(1): 67-82.

Beydoun, G., Low, G., García-Sánchez, F., Valencia-García, R., and Martínez-Béjar, R. (2014), Identification of ontologies to support information systems development, *Information Systems*, 46 (12), 45-60.

Broekstra, J., Kampman, A. & van Harmelen, F. (2002). Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema. *First International Semantic Web Conference*, 54-68. Italy.

Brown, R.B.K., & Piper, I.C. (2013). What Users Do: SA&D with the ATSA Method, In R. Pooley, et.al (Eds.), *Information Systems Development, Reflections, Challenges and New Directions* (pp. 305-316). Springer.

Carlsson, S. A., Henningsson, S., Hrastinski, S., & Keller, C. (2011). Socio-technical IS design science research: developing design theory for IS integration management. *Information Systems and e-Business Management*, 9(1), 109-131.

Carrera, Á., Iglesias, C. A., & Garijo, M. (2014). Beast methodology: An agile testing methodology for multi-agent systems based on behaviour driven development. *Information Systems Frontiers*, 16(2), 169-182.

Casu, M., Cicala, G., & Tacchella, A. (2013). Ontology-based data access: An application to ontological logistics. *Information Systems Frontiers*, 15(5), 849-871.

Constantine, L.L., & Lockwood, L.A.D. (1999). *Software for Use - A Practical Guide to the Models and Methods of Usage-Centred Design*. ACM Press. Addison Wesley, Reading MA. (pp. 23)

Cooke, D., Gelman, L. and Peterson, W. J. (2001). ERP Trends. Available at: <http://www.conferenceboard.ca/documents.aspx?DID=869> Date accessed 15 Feb. 2010.

Crear, J. (2009), CIO Standish Group. Available at: <http://www1.standishgroup.com/newsroom/chaos_2009.php>. Date Accessed 24 March 2010.

d'Aquin, M., & Noy, N. F. (2012). Where to publish and find ontologies? A survey of ontology libraries. *Web Semantics: Science, Services and Agents on the World Wide Web*, 11, 96-111.

Dardenne, A., Lamsweerde, A., and S. Fickas. Goal-directed requirements acquisition. Science of computer programming, 20(1-2):3–50, 1993.

Dawson, L. (2012, August). A Social-Creative-Cognitive (SCC) model for requirements engineering. *21st International Conference on Information Systems Development*, Prato, Italy.

Di Noia, T., Di Sciascio, E., & Donini, F.M., (2007). Semantic Matchmaking as Non-Monotonic Reasoning: A Description Logic Approach .*Journal of Artificial Intelligence Research 29*, 269-307.

Dobson, G., & Sawyer, P. (2006). Revisiting Ontology- Based Requirements Engineering in the age of the SemanticWeb, In *Dependable Requirements Engineering of Computerised Systems at NPPs, Institute for Energy Technology (IFE)*, Halden.

Ellis, K., (2007). The Executive Briefing on the Issues Surrounding Getting Business Requirements Right. Available at: <http://www.scribd.com/doc/6766319/Business-Requirements> Date accessed 21 Apr 2011.

Ellis, K., & Berry, D.M. (2013). Quantifying the impact of requirements definition and management process maturity on project outcome in large business application development, *Requirements Engineering*, 18, 223–249.

Eschenbach, C. & Heydrich, W. (1995) Classical mereology and restricted domains. International Journal of Human-Computer Studies 43**,** 723-740.

Gamma, E., Helm, R., Johnson, R. and Vlissides, J., (1995). *Design Patterns. Elements of Reusable Object-Oriented Software*, Addison-Wesley, Reading, MA, USA, 395pp.

Grolinger, K., Capretz, M.A.M., Cunha, A., & Tazi, S. (2014). Integration of business process modeling and Web services: a survey. *Service Oriented Computing and Applications, 8*(2). 105-128.

Gruber, T., (1993). A Translation Approach to Portable Ontology Specifications. *Knowledge Acquisition, 5*. 199-220.

Guarino, N., Oberle, D., & Staab, S., (2009), What is an Ontology? In A. Staab, & R. Studer (eds.*) Handbook on Ontologies*, 2nd edn., pp.1–17. Springer, Heidelberg.

Heras, S., Botti, V., & Julián, V. (2014). An ontological-based knowledge-representation formalism for case-based argumentation. *Information Systems Frontiers*, 1-20. Available at: <http://link.springer.com/article/10.1007/s10796-014-9524-3/fulltext.html>. Date Accessed 10 Sep. 2014.

Hermida, J. M., Meliá, S., Montoyo, A., & Gómez, J. (2013). Applying model-driven engineering to the development of Rich Internet Applications for Business Intelligence, *Information Systems Frontiers*, 15(3), 411-431.

Hsu, I-C. (2013). Personalized web feeds based on ontology technologies. *Information Systems Frontiers*, 15(3), 465-479.

Keil, M., Cule, P. E., Lyytinen, K. and Schmidt, R.C. (1998), A Framework for Identifying Software Project Risk, *Communications of the ACM*, 41(11).

Ko, R.K.L., Lee, S.S.G., & Lee, E.W. (2009). Business process management (BPM) standards: a survey. *Business Process Management Journal 15*(5).744–791.

Liang, Q., Wu, X., Park, E.K., Khoshgoftaar, T.M., & Chi C-H., (2011). Ontology-Based Business Process Customization for Composite Web Services. *IEEE Transactions on Systems, Man, and Cybernetics—Part A: Systems and Humans, 41*(4), 717-729

March, S. T., & Smith, G. F. (1995). Design and natural science research on information technology. *Decision support systems*, 15(4), 251-266.

Mayank, V., Kositsyna, N., & Austin, M. (2004). Requirements Engineering and the Semantic Web, Part II. Representaion, Management, and Validation of Requirements and System-Level Architectures. *Institute for Systems Research Technical Reports 2004*(14) http://drum.lib.umd.edu/bitstream/1903/6421/1/TR_2004-14.pdf

McBride, B. (2002). Jena: a semantic web toolkit. *IEEE Internet Computing 6*(6), 55-59.

Macgregor, G., & McCulloch, E. (2006). Collaborative tagging as a knowledge organisation and resource discovery tool. *Library Review, 55*(5): 291-300

Mueller, R., (2012). Enterprise applications of semantic technologies for business process management. *Journal of Zhejiang University-SCIENCE C (Computers & Electronics) 14*(4), 308-310

Nagarajan, M., Verma, K., Sheth, A., Miller, J., & Lathem, J. (2006). Semantic Interoperability of Web services: challenges and Experiences, *4th IEEE International Conference on Web Services*, IEEE CS Press, (pp. 373–382).

Norton, B., Cabral, L., & Nitzsche, J. (2009). Ontology-based translation of business process models. *The International Conference on Internet and Web Applications and Services*, (pp 481- 486).

OASIG (1995) Available at: <http://www.it-cortex.com/Stat_Failure_Rate.htm#The%20OASIG%20Study%20> Date accessed 15 Aug 2010.

Rao, W., Chen, L., Hui, P., & Tarkoma, S. (2012, June). Move: A Large Scale Keyword-based Content Filtering and Dissemination System. *32nd International Conference on Distributed Computing Systems (ICDCS)*, Macau, China.

Rittel, H., & Webber, M. (1973). Dilemmas in a General Theory of Planning, *Policy Sciences 4*, 155-169.

Silva, T.S., Martin, A., Maurer, F., & Silveira, M. (2011 August). User-Centered Design and Agile Methods: A Systematic Review. *Agile Conference*, Salt Lake City, UT, (pp. 77-86).

Smith, K., Welty, C., & McGuinness, D.L. (2004). *OWL Web Ontology Language Guide, W3C Recommendation*. Available at: <http://www.w3.org/TR/owlguide/>.

Standish Group, The (1995). The CHAOS Report (1994). Available at: <http://www.standishgroup.com/sample_research/chaos_1994_1.php>. Date accessed 24 Mar 2010.

Studer, R., Benjamins, V.R., & Fensel, D. (1998). Knowledge engineering: principles and methods. *Data Knowledge Engineering*, *25*(1–2), 161–197.

Tichy, L., and Bascom, T., (2008). The Business End of IT Project Failure, Mortgage Banking. Available at: <http://www.stratmorgroup.com/PDF/The_Business_End_of_IT_Project_Failure_-_March_2008.pdf> Date accessed 4 Aug 2012.

Tran, Q.N.N., Beydoun, G. and, Low, G. (2007). Design of a peer-to-peer information sharing MAS using MOBMAS (ontology-centric agent oriented methodology). In *Advances in Information Systems Development*, Springer pp. 63-76.

Uschold, M., & Grüninger, M. (1996). Ontologies: principles, methods, and applications, *Knowledge Engineering Review 11*(2), 93–113.

Valiente, M.-C., Garcia-Barriocanal, E., & Sicilia, M.-A., (2012). Applying an ontology approach to IT service management for business-IT integration. *Knowledge-Based Systems, 28*, 78-87.

Veres, C., Sampson, J., Cox, K., Bleistein, S., & Verner, J., (2010). An Ontology-Based Approach for Supporting Business-IT Alignment In F. Xhafa et al. (Eds.), *Complex Intelligent Systems and Their Applications, Springer Optimization and Its Applications 41*, 21-42.

Verma, K., Sivashanmugam, K., Sheth, A., Patil, A., Oundhakar, S., & Miller, J. (2005). Meteor-S WSDI: a scalable infrastructure of registries for semantic publication and discovery of Web services, *Journal of Information Technology and Management 6* (1), 17–39.

Whittaker B. (1999) What went wrong? Unsuccessful information technology projects, *Information Management & Computer Security*, 7(1), 23-29.

Yu, E. S. (1997, January). Towards modelling and reasoning support for early-phase requirements engineering. In *Requirements Engineering, 1997., Proceedings of the Third IEEE International Symposium on* (pp. 226-235).

IEEE. World Wide Web Consortium (W3C) SPARQL working group, (2008). SPARQL Query Language for RDF W3C Recommendation. Available at < http://www.w3.org/TR/rdf-sparql-query/>. Date Accessed 14 Mar. 2011.

World Wide Web Consortium (W3C). (2004). *OWL-S: Semantic Markup for Web Services. W3C Member Submission*, Available at: <http://www.w3.org/ Submission/OWL-S/>.