A Task Orientated Requirements Ontology for Cloud Computing Services

Richard Greenwell[1], Xiaodong Liu[1], Kevin Chalmers[1] and Claus Pahl[2]
[1]Institute for Informatics and Digital Innovation, Edinburgh Napier University, Colinton Road, Edinburgh, U.K.
[2]School of Computing, Dublin City University, Glasnevin, Dublin, Republic of Ireland

Keywords: Cloud, Service, Description, Semantic, Requirements, Engineering, Ontology.

Abstract: Requirements ontology offers a mechanism to map requirements for cloud computing services to cloud computing resources. Multiple stakeholders can capture and map knowledge in a flexible and efficient manner. The major contribution of the paper is the definition and development of an ontology for cloud computing requirements. The approach views each user requirement as a semantic intelligence task that maps and delivers it as cloud services. Requirements are modelled as tasks designed to meet specific requirements, problem domains that the requirements exist in, and problem-solving methods which are generic mechanisms to solve problems. A meta-ontology for cloud computing is developed and populated with ontology fragments on to which cloud computing requirements can be mapped. A critical analysis of the usage of ontologies in the requirements process is made and a case study is described that demonstrates the approach in a real-world application. The conclusion is that problem-solving ontologies provide a useful mechanism for the specification and reuse of requirements in the cloud computing environment.

1 INTRODUCTION

Wind and Schrödl (Wind and Schrödl, 2011) describe a number of approaches to Requirements Engineering (RE) in cloud computing, which were found to be unsuitable in a number of key areas, such as architecture selection, legal issues and pricing. Cloud services require semantics to express functionality derived from many service providers.

Semantic web-services have successfully used ontologies (Fensel et al., 2003), as have a number of RE approaches (Happel and Seedorf, 2006). An ontological approach can address some of the shortcomings seen in the current cloud computing RE process, such as lack of completeness, consistency and conflicts between requirements.

Ontologies have been used for modelling requirements for various aspects of information systems. Farfeleder et al (Farfeleder et al., 2011) describe ontologies using natural language for formalising and verifying requirements in embedded systems. Jureta et al.(Jureta et al., 2008) discuss the use of ontologies in stakeholder communication.

A particularly useful ontological RE approach is described by Bogg et al (Bogg et al., 2011). Bogg explores the use of Problem-Solving Methods (PSMs) expressed as an ontology in RE. PSM are reusable methods or approaches to problems that can be used across a number of knowledge domains. The approach is seen as cogent for cloud computing, as large compute clouds can be seen in a service brokerage process, which could provide access to a large number of PSMs instances to solve problems across a number of knowledge domains.

In this paper, we advocate the view that the cloud computing environment can be seen as a problem-solving environment. Users have problems which can be tackled using a cloud computing, at a given quality of service and cost. Requirements ontology is used to support this problem-solving approach. The requirements are modelled as tasks designed to meet specific requirements, problem domains that requirements exist in, and as problem-solving methods which are generic mechanisms to solve problems and bridges between the three elements. The approach enables each user requirement to be considered as a semantic task, which can be implemented as a cloud service.

The remaining parts of the paper are organised as follows: section two presents an overview of the important work and aspects of ontology usage in RE. The design of the ontology is defined in section three. Section four provides a specification of the

Greenwell, R., Liu, X., Chalmers, K. and Pahl, C.
A Task Orientated Requirements Ontology for Cloud Computing Services.
In *Proceedings of the 6th International Conference on Cloud Computing and Services Science (CLOSER 2016) - Volume 1*, pages 121-128
ISBN: 978-989-758-182-3

requirements ontology. A case study is presented in section five which demonstrates and verifies the proposed RE approach. Section six provides discussion of the approach. Conclusions are drawn in section seven, with future work identified.

2 REQUIREMENTS ENGINEERING ONTOLOGY

Ontologies provide a structured framework for modelling the concepts and relationships of a domain of expertise. Ontologies support the creation of repositories of domain-specific reference knowledge (Crubézy and Musen, 2003). Ontologies have been used for requirements engineering for a number of years. Zave and Jackson (Zave and Jackson, 1997) described "core" ontology as solving the "requirements problem". The core ontology established the minimum set of information required for engineering requirements as:

S, W ├R

Given:

R are given requirements
S is a complete specification
W are domain assumptions

Proof of Obligation requires that the specification and domain assumptions to be satisfied by the requirements (Classen, 2007). This points to a "pure" but simplistic approach to RE that only specification and domain assumptions are required in the RE process. The approach is criticised, by Jureta et al (Jureta et al., 2008), who state that partial requirements cannot be described in Zave and Jackson's model, and only a complete specifications can be created. The requirements specifications cannot be ranked in terms of better or worse requirements for a given specification. Non-core requirements cannot be defined and, nice to have requirements may be lost.

Castanada et al (Castaneda et al., 2010) identify a number of benefits in using ontology in the RE process. A requirements model is imposed enabling the structuring of requirements and the knowledge domain in question. The Interrelationships between requirements can be defined.

A number of attempts have been made to specify an ontology to describe the components of cloud computing, a typical example being Youseff et al (Youseff et al., 2008). These ontological approaches suffer from viewing cloud computing as a continuation of Software as a Service and concentrate on low level virtualisation.

A cloud computing ontology has been developed and enhanced for cloud computing using the work of Norton et al (Norton et al., 2008). The ontology development is the major contribution of this paper, along with ontology 'Fragments'. The ontology can be seen as a meta-ontology for RE in cloud computing environments, building on more generic ontologies for problem-solving.

Each user requirement can be defined as a semantic task, this facilitates enhanced capability in the validation of specification, the discovery of services and composition of cloud services. Cloud computing can be seen as more complex than traditional Information (IT) environments. User requirements are expressed at a high level, a brokerage layer or service will find and price these requirements from a number of cloud computing resources. Cloud computing resources will then execute tasks for these brokered requirements.

Expressing requirements using a problem-solving ontology allows the requirements engineer to utilise an approach that is well suited to the cloud computing environment. Tasks can be seen as a unit of work that is well understood by users. Problem Solving Methods (PSMs) can be seen as reusable specifications for solving the problems posed by tasks. Domain models can be built as an ontology so it can be understood by users and verified using ontological reasoning tools. The requirements ontology can be seen as a specialisation of more generalised problem-solving ontology such as the Unified Problem-solving Method Development Language (UPML) which will be described.

Fensel et al (Fensel et al., 2003) describe (UPML) which is a framework for developing knowledge-intensive reasoning systems based on libraries of generic problem-solving components. They go onto describe the UPML architecture as *tasks* that defines the requirements for the problem that is to be solved. *Problem-solving methods (PSM)* define the reasoning process and, *domain models* that describe domain knowledge. Bridges are used to map and define the relationship and transformation between the task and PSM.

Crubézy and Musen (Crubézy and Musen, 2003) describe how Problems Solving Methods (PSMs) and domain ontologies are combined to produce knowledge systems. Musen (Musen, 2001) describes domain ontologies as a "Characterisation of concepts and relationships in an application area, providing a domain of discourse". Domain ontologies define problem specific knowledge.

3 REQUIREMENTS ONTOLOGY DESIGN

A detailed requirement ontology can be mapped to a number of 'knowledge components' for implementation within ontology modelling tools. The knowledge component provides a base selection of properties such as description and requirement pragmatics. Elements of the ontology then inherit properties from the knowledge component. Specialist PSMs such as problem decomposers (PSMs that can split a task into subtasks) can be developed for specific purposes. Requirements engineers can develop their own specialist tasks, PSMs and domain models for a specific requirements problem using powerful mechanisms such as inheritance and set operations. The usage of tasks, PSMs and domain models will lead to greater reuse as a generic method PSM.

Figure 1 (below) describes a model into which requirements can be tailored. This machine readable model is used directly in cloud computing environments.

The ontology provides a checklist of 'what' requirements are needed and is specified in terms of tasks, PSMs and Domain Models. The model provides representation for elements of requirements. Requirements are expressed in terms of semantics and, concepts such as tasks can be expressed in terms of rich semantics, as can relationships between tasks, PSMs and Domain Models. This allows the requirements engineers' greater expressive power, and the ability to carry out fuzzy searches and to map new knowledge and requirements via the reasoning tools seen in ontology management tools.

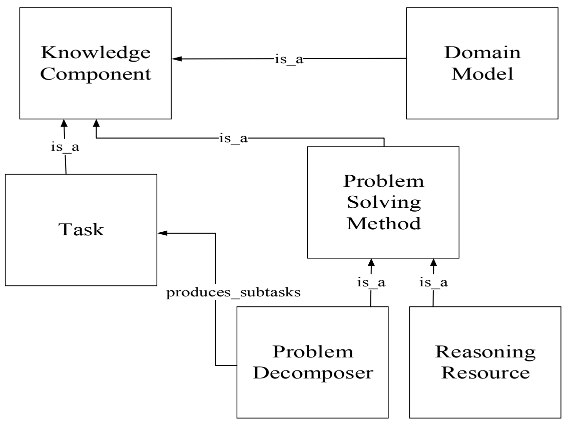

Figure 1: Requirements Ontology Implementation.

The architecture of the ontology is described in Figure 2 (below). The highest layer deals with problem-solving for cloud computing. Users will have tasks which use the PSMs and domain ontology. The brokerage layer defines elements in terms of ontology, tasks will be executed at a strategic level across the cloud environment dealing with issues such as cost and quality of service. The low level layer deals with operational requirements.

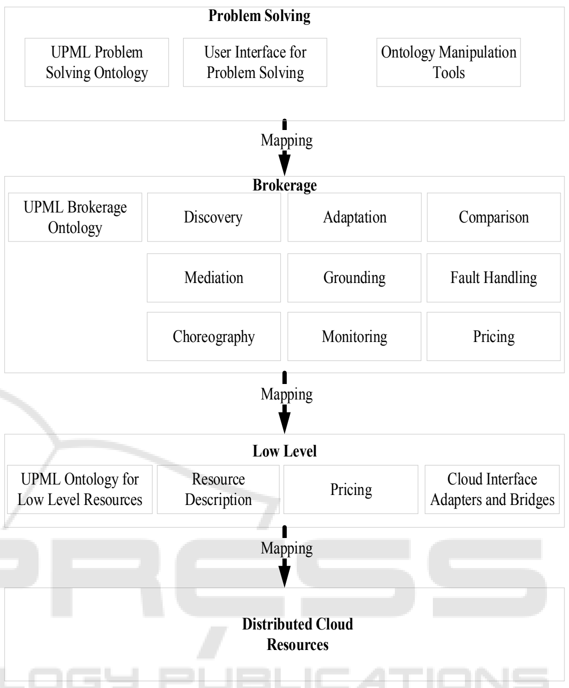

Figure 2: The hierarchical Stricture of Ontology.

The details of each element of the requirements ontology from figure 2 (above) will now be described.

3.1 Problem Solving

The problem-solving layer relates to the high level requirements of users expressed as ontology and, they describe 'what' is required; which may be full requirements or partial requirements expressed as ontology fragments. The requirements must be matched to low level requirements through the brokerage process via semantic or fuzzy matching.

The separation of requirements into tasks, PSMs and domain knowledge and representation as UPML provides ease of mapping to low level requirements through the brokerage process.

3.2 Brokerage

The brokerage of requirements map high level requirements to lower level requirements. This is carried out by semantic searching, fuzzy matching and negotiated processes. Discovery can be driven by the Quality of Service (QOS) requirements. These requirements are carried forward from user requirements expressed in the high level layers of the requirements ontology. Monitoring can use the QOS requirements to define the requirements for service failure. Pricing requirements can also be related to QOS.

Discovery requirements describe what and how cloud services are found across a set of cloud resources. Adaptation requirements relate to how defined requirements can be adapted to meet new user requirements. The adaptation process is made easier by the use of ontology as sets of requirements that that can be adapted by recombination through semantic relationships provided in the UPML ontology. Closely related to adaptation, composition requirements describe how sets of cloud resources are combined to meet high level user requirements.

Mediation requirements define how high level problem-solving requirements will be translated into low level requirements by a process of iteration. Many of the mediation requirements are concerned with QOS, Rimal et al (Rimal et al., 2011) describe the need for quality of service requirements in cloud computing. Quality of service provides a guarantee of the availability and performance of tasks inside the cloud. Requirements are supported by service elements such as security, reliability and dependability. Stakeholder groups will place value on service elements, for example low latency short burst resources will be required by some users, whereas other users will require long running resource pools. Grounding requirements link the execution of the requirements with how the requirement is to be executed at a low level. Fault handling requirements provide actions that are necessary when errors occur at low levels in cloud resources.

Choreography requirements provide the approach required for coordinating higher level requirements so they are performed correctly at a low level. Monitoring requirements specify the information required as tasks are executed and choreographed. Pricing requirements at the brokerage level deal with pricing estimation for a given high level tasks and aggregate pricing for packages of low level tasks.

The requirements ontology can draw upon many leading research concepts seen in the literature to represent concepts in the requirements ontology as a complete ontology or ontology fragments. Robinson (Robinson, 2003) describes service monitoring, which is a brokerage component within the requirements ontology. The high level requirements for monitoring can be represented as a PSM:

1. Define the design-time model
   a. Define goals and requirements
   b. Define obstacles and monitors
2. Define the run-time model
3. Monitor the running program

It should be noted that this PSM can be used by a number of tasks as the PSM can act on a number of problem domains. The requirements are represented in the brokerage layer. A primary goal can be decomposed by a specialist PSM called a problem decomposer. Tasks such as monitor will have inputs, outputs, competencies and formal definitions seen in Figure 2 (above). Lower level representation can also be represented as ontology.

The discovery and monitoring processes can use a similar service discovery and monitoring approach in cloud computing. High level requirements are used to drive the service discovery of web-services. Users can then select cloud services that match their QOS requirements.

Service adaptation can be seen in the requirements ontology. Higher level requirements goals and services categories can be represented as domain models; these are measured using a 'measure' PSM. In the brokerage layer service definitions are domain models used by a monitor definition PSM. The lower service domain models are monitored by PSM.

Ontological representation could provide many of the features required by researchers, such as fuzzy searching and the matching and representation of partial requirements using ontological fragments.

3.3 Low Level Requirements

Resource description requirements of each low level cloud resource are required so that they can be brokered. Examples of a resource description could be maximum CPU capacity, storage capacity, response time and spare CPU capacity.

Sun et al (Sun et al., 2012) point out that cloud computing has seen vendors offering a number of cloud computing platforms. Ontology can be used to describe vendors' offerings and can be used to abstract models from the integration of disparate offerings. Pricing requirements at a low level deal with areas such as the cost of CPU capacity and storage capacity. The requirements of cloud adaptors and bridges provide information for brokerage requirements.

4 SPECIFICATION OF THE REQUIREMENTS ONTOLOGY

The three levels of the requirements ontology (problem-solving, brokerage and low level) are all described in terms of UPML. This allows mapping, stepwise refinement, interaction and reasoning to be carried out between the layers. The usage and processing of ontology fragments has been described by a number of researchers (Nebot and Berlanga, 2009), (Kalfoglou et al., 2008) and (Packer et al., 2010).

The high level problem-solving requirements are specific to each individual requirements domain or process. They are still defined and structured in a UPML and the example of high level problem-solving is shown in the case study (below). The requirements ontology concentrates on the brokerage and low level aspects of RE. Brokerage fragments will take the problem-solving requirements and consider the requirements for their fulfilment. An example of a brokerage fragment will be given for discovery. Discovery is the process of finding resources for the fulfilment of a high level requirement. In Figure 3 (below), the RE ontology fragment for discovery is shown. The UPML ontology provides the framework for ontology fragments, which in turn guide the subsequent RE process. The discovery process is driven by two tasks, the discover resources search the cloud resource model and the cloud technology to build a catalog of resources and will search the catalog with a query string to allow the resources to be discovered.

Table 1 (below) shows how properties can be defined for the "Discover Resources" task.

Table 1: Discover Resources Task.

Input/Output Class	Cardinality	Type
Input	Exactly 1	Cloud Resource Model
Input	Exactly 1	Cloud Topology
Output	Exactly 1	Catalog

Now the requirements ontology has been described in detail a case study will demonstrate how the requirements can be defined for each requirements ontology element.

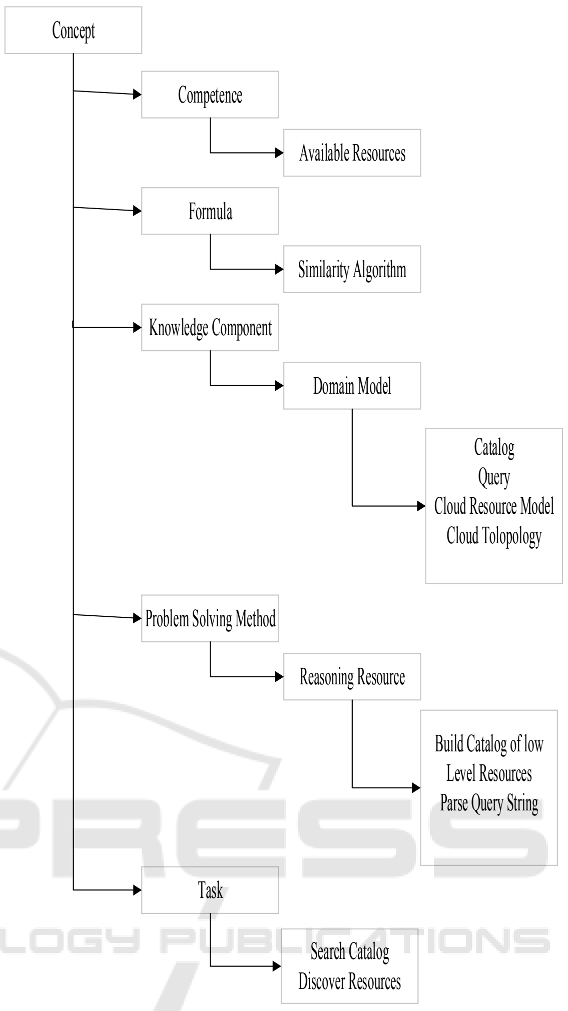

Figure 3: Discovery Element Ontology Fragment.

5 CASE STUDY

The case study shows how RE can utilise a UPML based ontology using the concepts described in the requirements ontology. The case study describes the requirements for a document similarity framework, which allows documents such as academic texts to be compared to the papers they reference. Manning et al (Manning et al., 2008) outline an approach for document similarity.

- Collect the documents to be indexed
- Tokenise the text
- Carry out linguistic pre-processing of tokens
- Index the documents that each term occurs in
- Use similarity measures based on mathematical measures, such as Cosine Similarity

- Report or carry out further processing such as clustering

The case study is particularly suited to cloud computing as large amounts of parallel processing are required to process documents. In single processor machines finding and comparing thousands of documents can take several hours. There is also scope to expand the application to recursively find referenced documents from the documents referenced from a study text.

5.1 High Level Problem-solving Requirements Ontology.

The high level tasks required for the case study are described Table 2 (below). The requirements describe a workflow of tasks which need to be executed to carry out document similarity for a document from a student course.

Table 2: Case Study: High Level Requirements.

Task Requirements	Description
Find academic references in course Documents	Parsing to find document references.
Create structured references and import into reference management system	Format references so they are machine readable.
Find academic papers for references	Find references automatically using cloud-services
Extract plain text from the PDF files, break into pages and tokenise text	Use off-the-shelf cloud software libraries
Pre-process tasks and indexation	Use off-the-shelf software libraries
Create similarity measures and match documents	Suited to Cloud computing Burst of processor bound tasks
Reporting	Report document similarity

These task requirements are then converted into UPML ontology. The requirements are split into tasks, PSM and Domain Models.

5.2 Brokerage

The brokerage ontology matches the high level requirements to the low level requirements ontology. Each aspect of the ontology will now be discussed.

Discovery can be seen as requirements which use a ‘find low level requirement for a high level requirement’ PSM. The high level requirement task ‘evaluate_corpus’ requires the low level formulas such as ‘Ratio Distance’ and will discover ‘match_papers_to_study_text_papers’.

Adaptation is the process of adapting low level requirements to meet a new or existing high level requirement. Composition defines the ordering of requirements tasks to complete the goal of producing document similarity for a corpus of documents. The tasks in the case study are self-organising as output from one low level resource feeds the input of another low level resource.

Mediation is driven by the high level requirements specification to find the most appropriate low level resource by stepwise refinement.

Grounding is a simple mapping of high level tasks requirements to individual software modules.

Fault handling requirements deal with actions that occur in low level programming language modules, virtual machines and physical machines.

An off the shelf chorography model was used. Yazir et al (Yazir et al., 2010) describe the PROMETHEE methodology for chorography across multiple cloud resources where a number of physical machines (PM) are allocated including Virtual Machines (VM) across a set of cloud resources. Monitoring requirements concern the information required being used to review the progress of the low level execution of tasks.

An existing pricing model was used in this case study. Henzinger et al. (Henzinger et al., 2010) discuss the Flexprice model for pricing across multiple cloud resources. In a commercial implementation of a document similarity framework high level task requirements will be priced across a number of cloud providers and the most cost effective solution will be selected

5.3 Low Level Requirements Ontology

A number of formulas are required to calculate document similarity. UPML allows individual software components to be described. Tasks describe the operations required to meet requirements high level requirements. UPML can describe both high and low level requirements in a structured way.

6 DISCUSSION

Requirements engineering ontology provides a three layer framework for RE in the cloud computing environment built on UPML.

The ontology can be checked for correctness and reasoning and can map new knowledge from the ontology that can be relayed to users. Requirements can be inserted into the ontology and used at a later date. Requirements can be found using semantic or fuzzy searching as well as syntactical searching.

The requirements ontology environment can be used to develop meta-services. These meta-services support two key features that are new to cloud computing self-service and on-demand provision. The high level and brokerage requirements seen in the requirements ontology allow customers to access on-demand self-service via meta-services.

The case study has demonstrated the requirements ontology built on UPML. The three layers of the requirements ontology provide guidance for the definition of a document similarity framework for study texts and the papers referenced from the study text. High level requirements, brokerage and low level requirements are expressed as textual requirements and, then as a UPML ontology. Ontology mapping and reasoning tools can be used to match each layer of the model, so that high level requirements can be executed by appropriate resources in the cloud. The use of ontology leads to a greater reuse of requirements and the generation of new requirements by reasoning.

The reuse of requirements is a key advantage of using a UPML based ontology. A PSM can be used in many knowledge domains and knowledge domains can be re-used for new requirements. Problem-solving ontologies are seen as useful for cloud computing as it can be seen as a problem-solving paradigm, as opposed to an extension of SaaS or virtualisation of existing applications.

7 CONCLUSIONS AND FUTURE WORK

This paper has described an ontology driven approach to requirements engineering for cloud computing. This is embodied in the requirements ontology which was built on a specialised form of ontology based on a UPML, which is well suited to service specification. A key aspect of the approach is the examination of the brokerage requirements, which bridge high level and low level requirements specifications.

The requirements engineering problem is broken down into three sets of concepts: tasks which describe the work that is to be done, problem-solving methods which describe the solutions to problems, and a problem domain which describes concepts for a given requirements scenario. The requirements ontology builds on a UPML structured ontology approach across the three distinct levels in cloud computing RE. Ontology mapping is seen as a key tool for linking requirements at different levels in the requirements ontology.

Future work will see the implementation being expanded to allow for a simpler specification of knowledge components such as tasks, domain knowledge, problem-solving methods and bridges. In future case studies, more complex brokerage will be used. Security will be included in the future version of the requirements ontology as it is a major emerging area in cloud computing.

REFERENCES

Bogg, P., Beydoun, G., Low, G., 2011. Problem-solving methods in agent-oriented software engineering. Information Systems Development, *Springer New York*} 4, 243–254.

Castaneda, V., Ballejos, L., Caliusco, M.L., Galli, M.R., 2010. The Use of Ontologies in Requirements Engineering. *Global Journal of Research Engineering 10*.

Classen, A., 2007. Problem-Oriented Feature Interaction Detection in *Software Product Lines*.

Crubézy, M., Musen, M.A., 2003. Ontologies in Support of Problem Solving, in: *Handbook on Ontologies. Springer*, pp. 321–342.

Farfeleder, S., Moser, T., Krall, A., Stalhane, T., Zojer, H., Panis, C., 2011. DODT: Increasing requirements formalism using domain ontologies for improved embedded systems development, proceedings of the 14th IEEE International Symposium on Design and Diagnostics of Electronic Circuits & Systems (DDECS). *IEEE*, pp. 271–274.

Fensel, D., Motta, E., Benjamins, V.R., Crubezy, M., Decker, S., Gaspari, M., Groenboom, R., Grosso, W., Harmelen, F. van, Musen, M., Plaza, E., Schreiber, G., Studer, R., Wielinga, B., 2003. The Unified Problem-solving Method Development Language UPML. Knowledge and Information Systems, *Springer-Verlag London Limited 5*, 83–131.

Happel, H.J., Seedorf, S., 2006. Applications of ontologies in software engineering. Presented at the Proc. of Workshop on Sematic Web Enabled Software Engineering"(SWESE) on the *ISWC, Citeseer*, pp. 5–9.

Henzinger, T.A., Singh, A.V., Singh, V., Wies, T., Zufferey, D., 2010. FlexPRICE: Flexible Provisioning of Resources in a Cloud Environment, in: Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on. Presented at the Cloud Computing (CLOUD), 2010 *IEEE 3rd International Conference on*, pp. 83–90. doi:10.1109/CLOUD.2010.71.

Jureta, I.J., Mylopoulos, J., Faulkner, S., 2008. Revisiting the core ontology and problem in requirements engineering. Presented at the Proceedings of the 16th *IEEE International Conference on Requirements Engineering (RE'08)*, 2008, *IEEE*, pp. 71–80.

Kalfoglou, Y., Smart, P.R., Braines, D., Shadbolt, N., 2008. POAF: *Portable ontology aligned fragments*.

Manning, C., Raghavan, P., Schutze, 2008. Introduction to Information Retrieval. *Cambridge University Press*.

Musen, M., 2001. Building Systems with Ontologies and Problem-Solving Methods.

Nebot, V., Berlanga, R., 2009. Efficient retrieval of ontology fragments using an interval labeling scheme. Information Sciences 179, 4151 – 4173. doi:10.1016/j .ins.2009.08.012.

Norton, C., Domingue, J., Zaremba, M., 2008. Semantic Execution Environments for Semantics-Enabled SOA, in: IT - Methods and Applications of Informatics and Information Technology. Presented at the Special Issue in *Service-Oriented Architectures*, pp. 118–121.

Packer, H.S., Gibbins, N., Jennings, N.R., 2010. Collaborative learning of ontology fragments by cooperating agents.

Rimal, B., Jukan, A., Katsaros, D., Goeleven, Y., 2011. Architectural Requirements for Cloud Computing Systems: An Enterprise Cloud Approach. *Journal of Grid Computing 9*, 3–26.

Robinson, W.N., 8. Monitoring Web service requirements. Requirements Engineering Conference, 2003. Proceedings. 11th *IEEE International 65–74*. doi:10.1109/ICRE.2003.1232738.

Sun, Y., Harmer, T., Stewart, A., Wright, P., 2012. Mapping application requirements to cloud resources, in: Euro-Par 2011: *Parallel Processing Workshops*. pp. 104–112.

Wind, S., Schrödl, H., 2011. Requirements engineering for cloud computing: a comparison framework, in: Proceedings of the International Conference on Web Information Systems Engineering, WISS'10. *Springer-Verlag*, Berlin, Heidelberg, pp. 404–415.

Yazir, Y.O., Matthews, C., Farahbod, R., Neville, S., Guitouni, A., Ganti, S., Coady, Y., 2010. Dynamic resource allocation in computing clouds using distributed multiple criteria decision analysis, in: Cloud Computing (CLOUD), 2010 *IEEE 3rd International Conference on*. pp. 91–98.

Youseff, L., Butrico, M., Silva, D.D., 2008. Towards a unified ontology of cloud computing, in: In Proc. of the Grid Computing Environments Workshop (GCE08). *IEEE,* Austin, Texas.

Zave, P., Jackson, M., 1997. Four dark corners of requirements engineering. *ACM Transactions on Software Engineering and Methodology (TOSEM) 6*, 1–30.