Ontology-based Software Engineering- Software Engineering 2.0

T.S. Dillon FIEEE, E. Chang SMIEEE,P. Wongthongtham

Digital Ecosystems and Business Intelligence Institute,
Curtin University of Technology,
Perth, Western Australia 6845
tharam.dillon{ elizabeth.chang,Pornpit.wongthongtham}@cbs.curtin.edu.au

Abstract

This paper describes the use of ontologies in different aspects of software engineering. This use of ontologies varies from support for software developers at multiple sites to the use of an ontology to provide semantics in different categories of software, particularly on the web. The world's first and only software engineering ontology and a project management ontology in conjunction with a domain ontology are used to provide support for software development that is taking place at multiple sites. Ontologies are used to provide semantics to deal with heterogeneity in the representation of multiple information sources, enable the selection and composition of web services and grid resources, provide the shared knowledge base for multiagent systems, provide semantics and structure for trust and reputation systems and privacy based systems and codification of shared knowledge within different domains in business, science, manufacturing, engineering and utilities. They, therefore, bring a new paradigm to software engineering through the use of semantics as a central mechanism which will revolutionize the way software is developed and consumed in the future leading to the development of software as a service bringing about the dawn of software engineering 2.0.

1. Introduction

Ontology was initially introduced into computing and information technology as a means of providing the semantics in the "Semantic Web". This provided support for the retrieval information based on its meaning rather than just simple string matching. Since this early use of ontologies, they have now grown to provide semantics and mechanisms for communication and structuring of knowledge in a wide variety of uses in IT, business and many other areas of human endeavour. In this keynote paper, we will provide a panoramic vista of some of the ways ontologies are being used in our work and that we hope to stimulate more research in the software engineering community into what you believe to be as the essence of software engineering 2.0.

Software as a service is going to be increasingly the dominant means of delivery and consumption of software. This will mean there must be a good enough characterization of the semantics of the software services to allow one to choose the appropriate software and to compose different components of software that meet particular requirements. In addition, this will introduce new approaches to software development. The use of multisite software development to allow one to cost-effectively access the sources to carry out the development is accelerating. This brings new challenges in communicatiion between the groups working at different sites. Here again, ontologies have an important role to play.

2. Purposes for which ontologies are being used

Ontologies are being used in our research for several purposes. These include:

(1) To provide a strong and unambiguous communication mechanism, and references medium for software engineers working at multiple sites to develop software , for you to solve a software engineering ontology and a project management ontology.
(2) To provide a mediating mechanism for accessing heterogeneous data and information sources, particularly on the Web.
(3) To enable the building of applications on the Web by providing clearly defined semantics for Web services.
(4) To provide a common knowledge base for multi agents working in a particular domain.
(5) To provide clearly defined semantics and confidence for interactions on the Web, more specifically, to build :Trust and Reputation systems[4], Privacy Based systems[14,15]
(6) To provide clearly defined semantics for the knowledge in a number of different domains, including:
    - protein ontology
    - disease ontology
    - manufacturing ontology
    - energy and power systems ontology
    - different financial systems ontologies

It will not be possible in the space of this paper to discuss each of these ontologies in detail and their use in construction of software. I will therefore

1530-0803/08 $25.00 © 2008 IEEE
DOI 10.1109/ASWEC.2008.77

make a selection and discuss the cases one to four above and refer you to additional references or alternatively the websites if you wish to obtain information about the other ontologies and their use.

3. Ontology Definitions

The term "Ontology" is derived from its usage in philosophy where it means the study of being or existence as well as the basic categories [9]. Therefore, it is used to refer to what exists in a system model.

An ontology, in computer science, is an explicit specification of a conceptualisation [11,12]. In such an ontology, definitions associate the names of concepts in the universe of discourse e.g. classes, relations, functions) with describing what the concepts mean, and formal axioms that constrain the interpretation and well-formed used of these terms [10].

People use the word ontology in different ways and to mean different things. However, different definitions provide different and complementary points of view on the word ontology. In the following sections, we compare ontology with data catalogues of glossaries, data dictionaries, thesauri, taxonomies.

We summarise the comparison between ontology and glossary or data dictionary and a taxonomy from [6,19]. Ontology is more than a glossary or data dictionary in whose terms everything else must be well described. An Ontology is more than a taxonomy or classification of terms. Often the term 'ontology' has been used very loosely to label almost any conceptual classification schema. Although a taxonomy contributes to the semantics of a term in a vocabulary, ontologies include richer relations between terms. A true ontology should contain not only a hierarchy of concepts organised by 'is a', 'subtype', or 'subclass' relations, but other 'semantic relations' that specify how one concept is related to another. The terms in ontology are chosen to ensure the representation of the abstract foundational concepts and distinctions within the domain of interest and form a complete set whose relationship one to another is defined using formal techniques which provide the semantic basis for the terminology chosen.

We should also distinguish between knowledge representations in Knowledge Based Systems (KBS) and Ontologies. A knowledge representation in a KBS is solely for the purpose of reasoning within that KBS and the terms do not have to be capable of being shared or understood more widely. In contrast,a key element of anontology is the *shared nature of the conceptualization across the community* which represents the domain.

4. Ontology Based MultiSite Software Development

There have been major shifts from the traditional single-site business environments where people, resources, business, and services are all centrally managed, monitored, and controlled to today's multi-site environments for software development. Multi-site distributed software development requires the implementation of methodologies, technologies, and processes to minimise the potentially negative aspects and to leverage multi-site distributed software development benefits. There are four issues that need to be addressed:

- Communication and coordination
- Unified Knowledge Sharing
- Knowledge Sharing Platform
- Methodology Adaptation and Validation

Failure to identify a clear issue or to correctly interpret an answer, often causes miscommunication, misunderstanding, and misinterpretations during discussion, subsequently followed by lack of coordination of activities and tasks. The physical distance becomes a crucial issue when the specifications are not complete, or ambiguous, or continually evolving, thereby needing more interaction among team members. Failure to share unifying knowledge, which includes domain knowledge, common knowledge, and project information including project data, project agreement, and project understanding, is a key issue. Awareness of the work that is being done according to the plan, the work that is being done co-operatively between teams, the current issues that have been raised, the issues that have been clarified, the means whereby members can conduct a discussion in order to make a decision on issues, all present a challenge in a multi-site distributed environment. Different teams might not be aware of the tasks that are being carried out by others, potentially leading to problems such as two groups overlapping in some work, or other work not being performed due to misinterpretation of the task. Wrong tasks may be carried out due to ignorance of whom to contact in order to obtain the proper details. If everyone working on a certain project is located in the same area, then situational awareness is relatively straightforward.. Over the last three years, we have developed the world's first and only Software Engineering Ontology (SE Ontology) which is available online at www.seontology.org. The SE Ontology defines common sharable software engineering knowledge including particular project information [23,24,25] and typically provides software engineering concepts – what the concepts are, how they are related, and why they are related [23,24,25]. These concepts facilitate common understanding of software engineering project information to all the distributed members of a development team in a multi-site development environment. We have merged Gruber's [11,12], and Studer's [21] definitions of an ontology as a basis to define the

software engineering ontology. Hence, the software engineering ontology is a formal, explicit specification of a shared conceptualisation in the domain of software engineering. 'Formal' implies that the software engineering ontology should be machine-understandable. Software engineering ontology facilitates better communication over software engineering domain knowledge between humans and machines. 'Explicit' implies that the type of software engineering concepts used, and their constraints, are explicitly defined. Software engineering ontology standardises and formalises the meaning of terms in the software engineering through its concepts. 'Shared' shows that the ontology specifies consensual knowledge of software engineering which means it is public and accepted by a group of software engineers. 'Conceptualisation' implies an abstract model that has identified the relevant software engineering concepts. In this work, the input of Professor Ian Sommerville [22,23] from St. Andrews University was critical,

The software Engineering ontology consists of several sub-ontologies namely:

- The software requirements ontology, which consists of the following four sub-ontologies namely: a requirements sub-ontology, a requirements elicitation sub-ontology, a requirements analysis sub-ontology, a requirements specification sub-ontology; See example in Appendix A.
- The software design ontology which consists of a design activitie sub-ontology, an architectural design sub-ontology, a detailed design sub-ontology, and a design strategies and methods sub-ontology; See ample in Appendix B.
- The construction ontology, which consists of construction language sub-ontology, a coding sub-ontology and a re-use sub-ontology; See example in Appendix C.
- The software testing ontology which consists of the following subontologies namely: a test issues sub-ontology, a test targets sub-ontology, a test objectives sub-ontology, a test techniques sub-ontology, a test activities sub-ontology;
- The software tools and methods ontology which consists of a software tools sub-ontology and a software methods sub-ontology.

It is not necessary that an ontology have instances, but software engineering ontology has the instances that represent project information including project data, project understanding, and project agreement. Figure 1 shows a schematic view of the software engineering ontology.

The entire set of software engineering concepts is captured in a generic software engineering ontology as domain knowledge. A particular project or a particular software development probably uses only part of the whole set of software engineering concepts. For example, if a project uses purely object-oriented methodology, then the concept of a data flow diagram might not necessarily be included; instead, it includes concepts like class diagram, activity diagram and so on. The specific software engineering concepts used for the particular software development project are captured in specific software engineering ontology as sub domain knowledge. The generic software engineering ontology represents all software engineering concepts that everyone working on software engineering is agreed upon, while specific software engineering ontology represents some concepts of software engineering for the particular project or particular enterprises need. Then, in each project, there exists project information or actual data including project understanding and project agreement. The project information in particular meets a particular project need and is needed with the software engineering ontology to define instance knowledge. Note that the domain knowledge is separated from instance knowledge. The instance knowledge varies depending on its use for a particular project. The domain knowledge is quite certain, while the instance knowledge varies according to the project. Once all domain knowledge, sub domain knowledge and instance knowledge are created, it is available to be shared among software engineers through the Internet. All team members, regardless of where they are, can query the semantic linked project data and use them as the common communication and knowledge basis for raising discussion matters, questions, analysing problems, proposing revisions or designing solutions, etc.

Capturing domain knowledge, organising sub domain knowledge, and storing and extending instance knowledge of individual members and teams within a multi-site project, and making the knowledge available to others in the project, are within the research area of knowledge management. The particular research issue of knowledge management requires software engineering knowledge management systems.

The reason for the development of software engineering knowledge management systems is basically to facilitate knowledge sharing, access, update, and exchange. According to this objective, tasks are assigned to the systems containing a number of sub systems. There is a set of systems to facilitate maintaining instantiations of software engineering ontology: safeguard system, ontology system, and decision maker system. The architecture of the whole systems in the multi-site environment is shown in Figure 2.

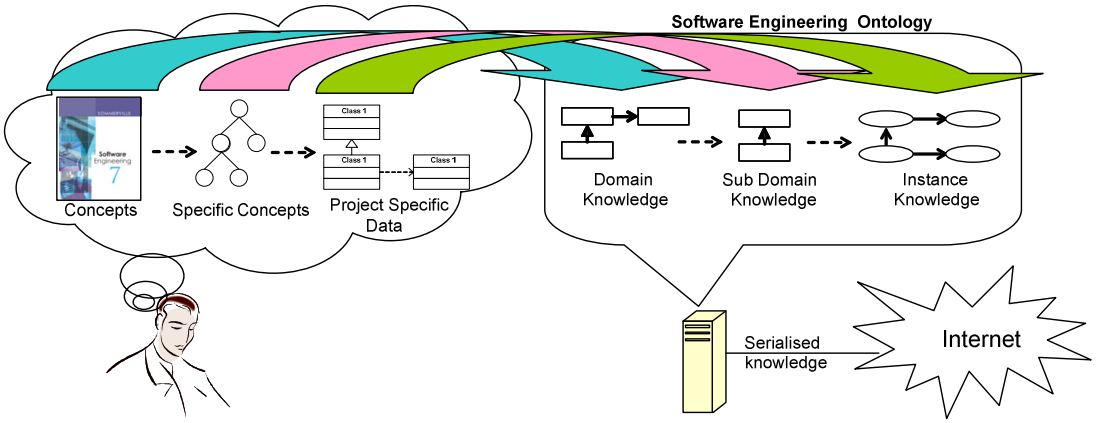


Figure 1 Schematic overview of software engineering ontology

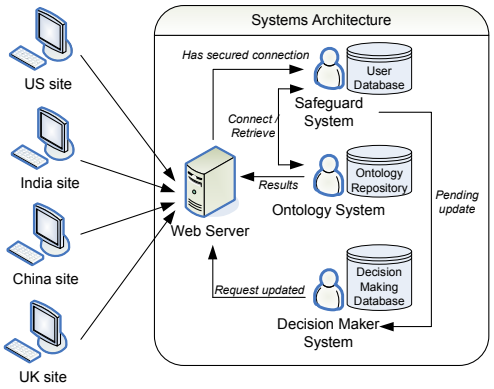


Figure 2Model of Management Systems

Team members, regardless of where they are, connect to the web server via a web browser. This will enable team members to directly use the system without having to download any software or install any application. Each team member is served by the intelligent systems tool as the communication media. This allows direct communication between different team members using a messaging system and also allows monitoring and recording of the activities of the team members. Each team member is provided with a particular set of access privileges that are dependent on the role of that team member in the project. The set of sub systems within the intelligent support systems architecture includes: safeguard system, ontology system and decision maker system.

In addition to the Software Engineering Ontology, the group at DEBII has defined a Multisite Project Management Ontology [3] which consists of 11 sub-ontologies, which are:
A Process sub-ontology based on PMI process structures and definitions; A Software Product sub-ontology which includes software development, PPP management and other PPP work; An Enterprise Architecture sub-ontology; A Software Component sub-ontology; An Actor & Role sub-ontology that defines the interplay of people and computers in PPP management; A PPP Team sub-ontology that represents the allocation of people to portfolio, program and project assignments; A Service Level sub-ontology that defines the product and process performance measures of PPP management; A Quality sub-ontology that defines service quality management to manage product & process performance variances; A Risk sub-ontology that defines the concepts of risks in PPP management enhanced with inclusion of MSPM requirements of PEST risk categories;A Control Structure sub-ontology that defines PMI instances of "Portfolio, Program and Project" as the control framework; A Location sub-ontology that abstracts the geography and building concepts of location, linked to the other appropriate elements of the Risk, Actor + Role, Process and Product sub-ontologies to profile the extension of PPP control structures by location definitions.

5. Ontology Mediated Information Access

In any given field of databases, there are widely varying characteristics using their own categories for storing data. Sometimes, different databases use identical labels but with different meanings; conversely, the same meanings are expressed via different names. Whenever database interoperability becomes a major problem, an ontology has a major role to play in alleviating this situation. For example, in one database whose entity relationship diagram is shown in Figure 3, data about an activity transition (transition between activities) might be encoded for the activity transition together with branch transition (transition between activities through the condition), special transition (transition from an activity to a stop or transition from a start to an activity) and concurrent transition (transition between activities through either a fork or a join). In the ontology, these

transitions are separated, therefore queries about fork transition, for instance, can be directed to the right place.

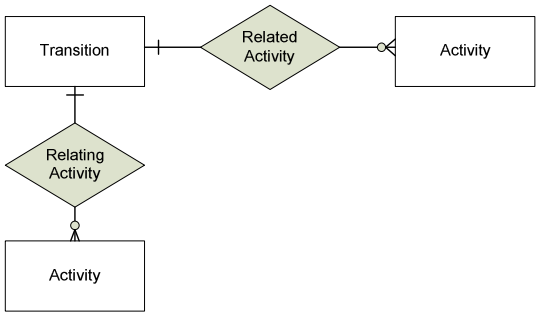


Figure 3. Entity relationship diagram representing activity transition concept

Rules in ontology can be expressed about relationships between concepts or classes and these can be used in query processing that generates all results matching the query according to the specified relationships. Unlike databases, in an ontology, new facts can be generated by inferring or reasoning with the asserted facts.

There has been a Data Explosion of Protein Structure Data which makes it difficult to create explanatory and predictive models that are consistent with the huge volume of data. This difficulty increases when a large variety of heterogeneous approaches to gather data from multiple perspectives and store it with completely different formats in the different protein databases. In order to facilitate computational processing of the data from the multiple data sources we have built the first and only available Protein Ontology (PO) to integrate protein knowledge and provide a structured and unified vocabulary to represent protein synthesis concepts.[20] This PO

- consists of concepts, which are data descriptors for proteomics data and the relationships among these concepts.
- has:
  - a hierarchical classification of concepts represented as classes, from general to specific;
  - a list of attributes related to each concept, for each class;
  - a set of relationships between classes to link concepts in ontology in more complicated ways then implied by the hierarchy, to promote reuse of concepts in the ontology; and
  - a set of algebraic operators for querying protein ontology instances.

More details about Protein Ontology are at: http://www.proteinontology.info

The Protein Ontology is a part of Standardized Biomedical Ontologies available through the National Center for Biomedical Ontologies along with Gene Ontology, Flybase, and others. http://cbioapprd.stanford.edu/ncbo/faces/pages/ontology_list.xhtml

This PO will form a standard on accessing the different protein data sources.

As the ontology in this application can act as a mediator for accessing not only relational data but also semi-structured data such as XML or metadata annotations and unstructured information it is a generalization of the original concept of a mediator proposed by Weiderhold for accessing relational databases. We call this approach Ontology Mediated Information Access (OMIA).

6. Ontology and Semantic Web Services

Web services [1,2,4] are self-contained components applications that can be described, published, located, and invoked over internet.. Web Services can be dynamically composed into applications. And this allows the implementations to be platform independent and programming language-neutral. Web Services systems promote significant decoupling and dynamic binding of components. The independence of different services publishers and the subscribers can formulate the most suited services they want.

The contemporary web services specification models merely focus on the *syntactical* levels, e.g. the Web Service Definition Language (WSDL),the Web Services Flow Language (WSFL), the Business Process Execution Language for Web Services (BPEL4WS), Web Service Capability Description Language (SCDL). Web Service Choreography Service (WSCI). These schemes capture the structural properties of the web components only, using the BPEL and WSCI to weave different Web services into meaningful business processes. However, this still remains a specification at the syntactic level. It is likely that the requirements of a user will often not be met by a single web service but will require the composition of several component web services.

There are several issues that must be addressed for successful application of these web services and these include (1) selection of a suitable architecture –see [7,8] for a discussion of different architectural styles and a proposed new approach;(2) Discovery of suitable services [26](3) selection of a service (4) composition and coordination of the services to meet the requirements. To assist the process particularly of discovery and selection our group like several other researchers have decided that it is necessary to semantically annotate these web services. We use a combination of Ontologies and Web 2.0 philosophy to achieve provision of semantics and composition. The key ideas below are more fully explained in [7,8]

First we define a core concept: a **Service Space** is a supportive environment where a collection of

Web services gather for the purpose of fulfilling user demands. Service space is the 'first class' concept to cope with challenges inherent in distributed Web services. It should be noted that a service space does not host, manage, or run services as do most services containers [5]. Rather, it provides infrastructure to enable service discovery and "mashup" at various levels. Web services within a Service Space are referred to as 'members' of that Service Space. In the Service Space regular Web resources are 'augmented' to Semantic Web Services which are then integrated into various Virtual Organisations in response to user requirements from the application layer. Three major Service Spaces are defined for 'lifting up' Web services, i.e. the Web Service Space, the Semantic Web Space, and the Virtual Organisation Space. In While Web 2.0 technology and 'attitude'[17] are to be entrenched in all three types of service spaces, they are particularly helpful in building Semantic Web Space and Web Service Space respectively.

The **Web Service space** provides fundamental infrastructure that enables the discovery of a large number of basic Web services in a loosely-coupled manner regardless of their locations, categories, and qualities. From the perspective of the Service-Oriented Computing, it resembles a number of contemporary global Web service registries such as public UDDI Business Registry, XMethods, StrikeIron, IBM SOA Catalog, etc. that can facilitate essential keyword-based service discovery. It also supports service subscription that allows potential *users* to track down interesting Web services.

Semantic Web Space (SWS) refers to a focused *Service Space* where a group of related Web services forms a domain-specific Web service community in order to facilitate dependable collaboration through trust-driven service selection and semantic-based service discovery. Domain here refers to areas with limited boundaries such as a specific geographical region, a particular industry, etc. Semantic Web Space shall provide sufficient elements for the establishment and enforcement of trust for users [4] and 'sense of community' for member Web services. We have recently observed that numerous Web 2.0 communities (e.g. 43things, Youtube, MySpace, del.icio.us) prosper for various reasons that can be studied in a number of disciplines including economy, social science, biology, and information science. The Semantic Web Space respects this phenomenon. Moreover, it utilises and extends such 'collective intelligence' by providing formal semantic-enabled and semantic-aware instruments that help to build long-lasting Web service communities beneficial for all Web service providers and consumers.

Transient Virtual Organisation (VO) is a demand-driven *Service Space* that allows a small group of Web services to form an ad hoc team working collectively in order to fulfil particular user demands during a given period of time. The main reasons for spawning such a transient VO lies in the gap between the complexity of actual user requirements and the limitation of each individual Web service obtained from both Web Service Space and Semantic Web Space. In addition,we believe ad-hoc **Web service mashup** – Web service mediation, expansion, customisation, and integration are essential for a VO to satisfy real-world user requirements. Presumably, VO members often come from the same Semantic Web Space so that most collaboration grounding – trust establishment, shared mission and value, agreed-upon business protocol, and essential technical interfaces, etc. – has been addressed by the semantic-based augmentation prior to the SWS formation. This well-established SWS is defined as the **enclosing SWS** of the VO. During the VO member selection, preferences are given to enclosing SWS members. It is however possible that external Web services are sometimes 'invited' to join a VO in case that appropriate Web services cannot be found solely from a single *enclosing SWS*. It is also possible that a SWS member is engaged in several VOs. In this case, the proportion of its commitment to a particular VO becomes an important criterion for the VO member selection.A group of End Users or a Broker conducts in-depth search in the Web Service Space and selectively collects Web services from various providers into several Semantic Web Spaces based on interests and the semantics of these Web services. During this process, a great number of anarchic Web services are 'clustered' into a well-organised Semantic Web Space dedicated in one specific domain. In general, we envisage that one can apply two approaches to semantically enrich existing Web services. The first top-down approach is based on the concept of ontology engineering, where scientists and domain experts manually annotate relevant Web services using specific domain ontologies and/or knowledge databases. The second empirical approach builds on practical methods such as data/text mining, business intelligence, machine learning that can be carried out (semi-) automatically without intensive human involvement. The Semantic Web Space nurtures Web services mainly through three means: semantic enrichment, semantic classification, and semantic discovery. A Broker directly deals with End User's demands and selects appropriate Web services from existing Semantic Web Space to conduct **Web Service Mashup** – a process where related Web services are rapidly integrated, customised, expanded, and mediated in an ad-hoc manner – in order to form a Virtual Organisation fulfilling the customer requirements. Our previous work in [4] has made the first endeavours to address service assessment and selection using the trust model and methodology. Instead of relying exclusively on the Service Broker, end users can also track down constantly-changing Web services in any Service

Spaces through the user-centred **Web Service Portal** (WSP). A WSP refers to a locally-accessible and highly-customisable user interface that provides a personalised view of activities and information essential to performing Service Space functions. In other words, WSP acts as a proxy on behalf of the end users to maintain a list of communication channels to involved Service Spaces. Unlike a traditional HTTP proxy server shared by a group of corporate users, a WSP is dedicated to serve only one user, thus creating the 'user-centred' view. WSP also reveals the notion of 'User Mashup'– a core concept underpinning the attitude of Web2.0 [16]. **User Mashup** in the context of WSP refers to an activity in which the user can 'hack' standard Service Space communication protocols, and hence extensively customises user interface or features based on his own preferences. User Mashup has a far-reaching influence on the development of the user-centred Service Space. It endows users with a broader control over the information flow across the Service Space as well as a refined user experience seamlessly integrated with end user applications in a loosely-coupled manner. Most significantly, User Mashup provides a powerful yet simple mechanism by which infinite 'virtual' syndications of Service Spaces can be created for each WSP. A **Virtual Syndication** of Service Spaces is a fresh, highly filtered, and combinatory view of several Services Spaces within a WSP. It is created, customised, and solely owned by each individual SSP user and does not affect other users or existing Service Spaces in any ways.

These ideas have also been extended to semantic Grid Services in [8].

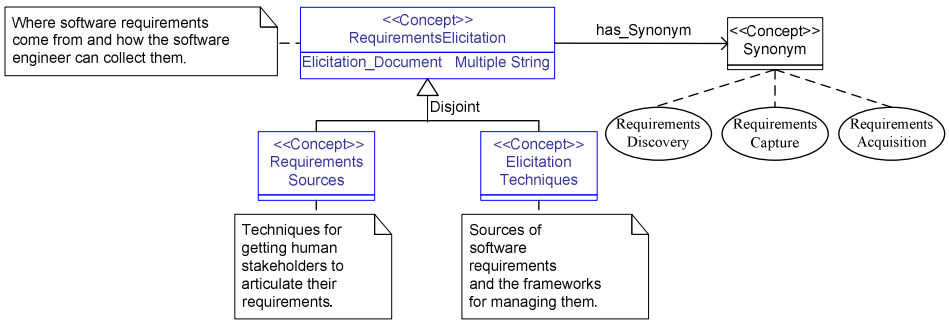
7. Ontology Based Multi Agent Systems

Agents are software entities capable of autonomous action. To solve more complex problems, a collection of agents that collaborate to solve problems in a given domain are employed and these systems are referred to as Multi Agent Systems. Frequently these agents have a small knowledge base to endow them with some intelligence. The problem always remains of ensuring that the knowledge bases of the different agents are coherent and consistent with one another. One solution to this is to have an ontology which is shared by all the agents in a given domain. The collection of agents in the Multi Agent System could then utilise this ontology as their common knowledge base. This will considerably facilitate communication and coordination between the agents when they are collaborating to solve a problem. However one of the problems that has remained until recently is that while there are methodologies for developing an ontology and methodologies for developing Multi Agent systems they are quite separate and do not have any link or connection with one another. As the key aspect is putting the Multi Agent System and Ontology together to leverage of each other it is important that the methodology for developing one takes account of the other. This issue has led to the group at DEBII developing a Methodology for Integrated Multi Agent and Ontology Development and the research is reported in [13].

9. Conclusions

We discussed several different uses of ontologies which varied from supporting software engineers to develop software to their use in providing semantics in very different settings. This use of ontologies particularly when coupled with the philosophy of Web 2.0 is likely to have a profound effect on the nature of , consumption of and development of software. It is therefore important that the software engineering community takes this on board and plays a leading role in the developments that are taking place.

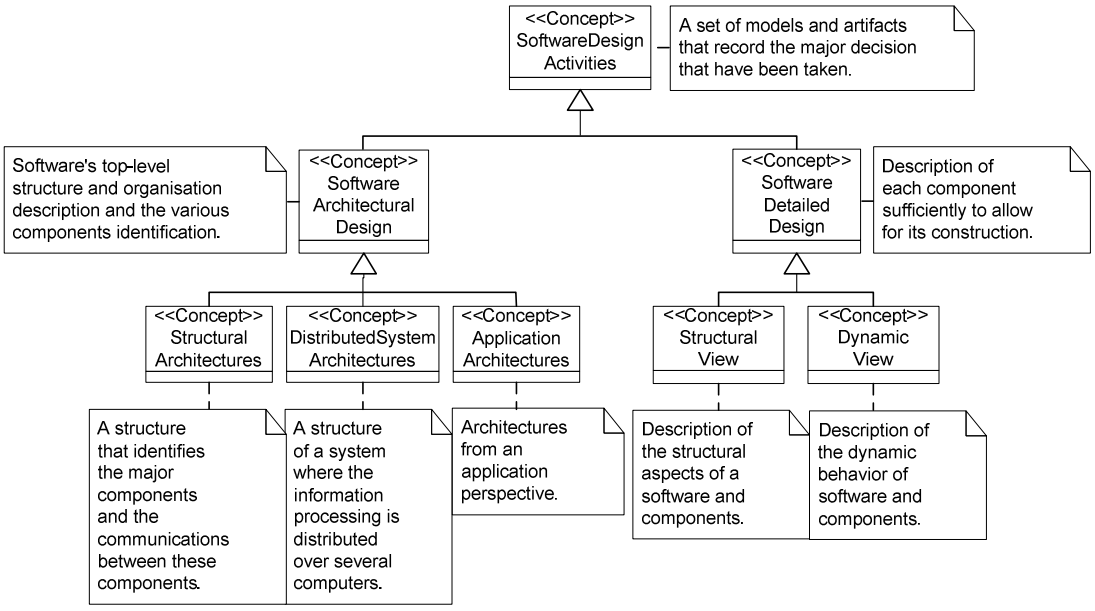
Appendix A: Ontology Representation for *Requirements Elicitation*



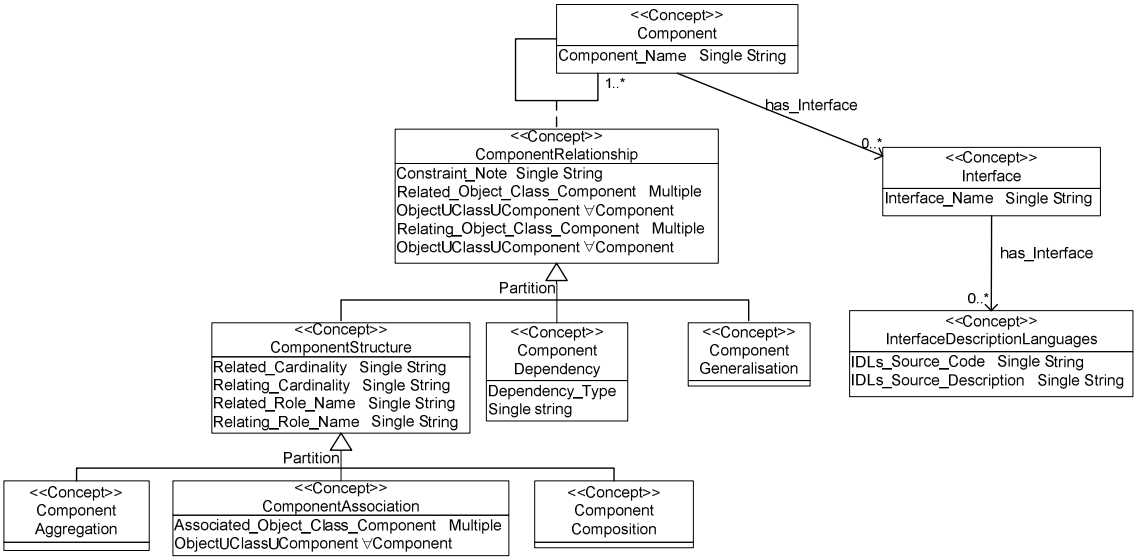
9.0 References

1. Alonso G., F. Casati, H. Kuno and V. Machiraju, Web Services: Concepts, Architectures and Applications, Springer-Verlag Heidelberg New York, 2004
2. Shadbolt N., W. Hall, and Tim Berners-Lee, The semantic Web revisited, IEEE Intelligent Systems, Volume 21, Issue 3, Jan.-Feb. 2006 Pp. 96 - 101
3. Chan C. 2007 Multi-Site Project Management (MSPM) -An Ontology Supported Methodology PhD Thesis, Curtin University of Technology.
4. Chang, E, Dillon, T & Hussain, FK 2006, *Trust and Reputation for Service Oriented Environment: Technologies For Building Business Intelligence And Consumer Confidence*, John Wiley and Sons.
5. Dhesiaseelan A and V. Ragunathan, "Web Services Container Reference Architecture (WSCRA)," ICWS'04, 2004.
6. Dillon, T 1993, *Object Oriented Conceptual Modeling*, Prentice Hall
7. Dillon Tharam S., Chen Wu, Elizabeth Chang 'Reference Architectural Styles for Service-Oriented Computing' Keynote , IFIP NPC 2007 Dalian, China
8. Dillon Tharam S., Chen Wu, Elizabeth Chang 'GRIDSpace: Semantic Grid Services on the Web — *Evolution towards a SoftGrid,* Keynote*,* IEEE Semantics, Knowledge and Grid Conf 2007, Xian,China
9. Fensel, D 2001, *Ontologies: Silver Bullet for Knowledge Management and Electronic Commerce*, SpringerVerlag.
10. . Klein, M, Fensel, D, Harmelen, Fv & Horrocks, I 2001, 'The relation between ontologies and XML schemas', in *Link oping Electronic Articles in Computer and Information Science*, vol. 6.
11. Gruber, TR 1993a, 'Toward principles for the design of ontologies used for knowledge sharing', *International Workshop on Formal Ontology in Conceptual Analysis and Knowledge Representation*, eds. G N & P R, Kluwer Academic Publishers, Deventer, The Netherlands, Padova, Italy.
12. Gruber, TR 1993b, 'A translation approach to portable ontology specification', *Knowledge Acquisition*, pp. 199-220.
13. Hadzic M., Wongthongtham P. , Chang E., Dillon T.S. 'Integrated MultiAgent and Ontology Development' Springer To Appear 2008
14. Hecker M., Dillon Tharam S., Elizabeth, Chang Elizabeth, Privacy Ontology Support for E-Commerce IEEE Internet Computing To Appear 2008
15. Hecker M., and Dillon T.S., "Ontological privacy support for the medical domain," in *eHPass National e-Health Privacy and Security Symposium*, Brisbane, Australia, 2006.
16. Högg R., M. Meckel, K. Stanoevska-Slabeva, and R. Martignoni, "Overview of business models for Web 2.0 communities," GeNeMe 2006.
17. Lin, K.-J. "Serving Web 2.0 with SOA, (Keynote Presentation)," ICEBE, Shanghai ,China, 2006.
18. Musser J. and T. O'Reilly, *Web 2.0 Principles and Best Practices*: O'REILLY RADAR, 2006.
19. NOY, NF & KLEIN, M 2003, 'Ontology Evolution: Not the Same as Schema Evolution', *Knowledge and Information Systems,* vol. 5.
20. Sidhu A.S., Dillon T.S, Chang E., "Integration of Protein Data Sources through PO," 17th International Conference on Database and Expert Systems Applications (DEXA 2006), Poland, 2006, pp. 519-527.
21. Studer, R, Benjamins, V & Fensel, D 1998, 'Knowledge Engineering: Principles and Methods', IEEE Transactions on Data and Knowledge Engineering, pp. 161-97.
22. Sommerville, I 2004, Software Engineering, 7th edn, Pearson Education Limited.
23. Sommerville, I 2007, Software Engineering, 8th edn, Pearson Education Limited.
24. Wongthongtham, P 2006, A methodology for multi-site distributed software development, PhD Thesis, Curtin University of Technology
25. Wongthongtham, P, Chang, E, Dillon, T & Sommerville, I 2006, 'Ontology-based multi-site software development methodology and tools', Journal of Systems Architecture, vol. 52, no. 11, pp. 640 - 53.
26. Wu C., E. Chang, "Aligning with the Web: An Atom-based Architecture for Web Services Discovery," *Service-Oriented Computing and Applications*, vol 1, 2007.

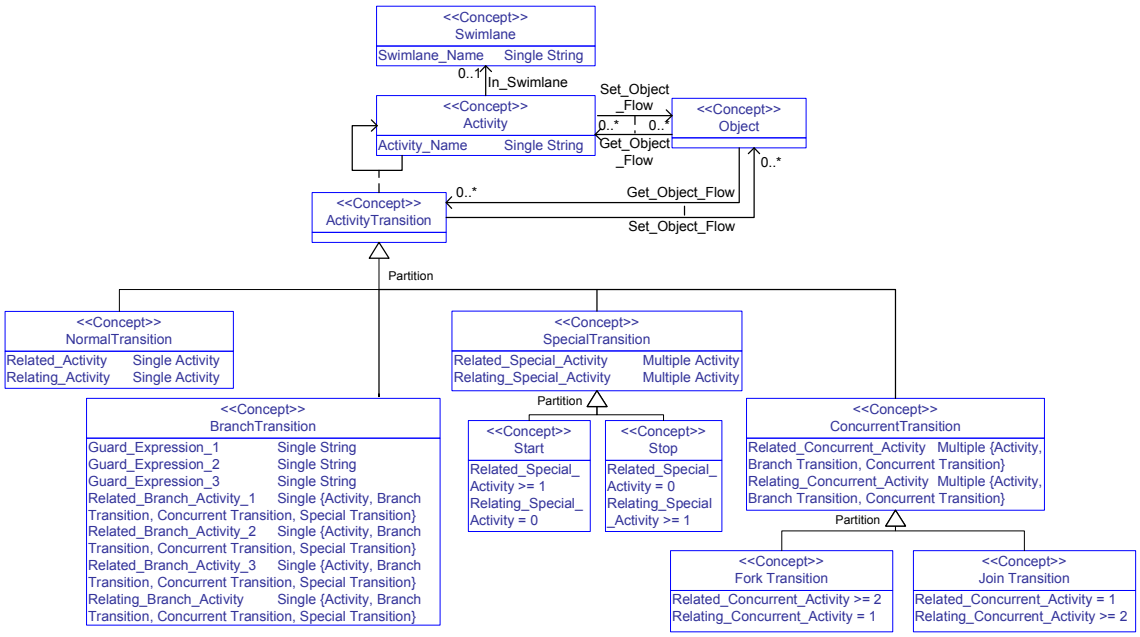
Appendix B1- Software design activities ontology



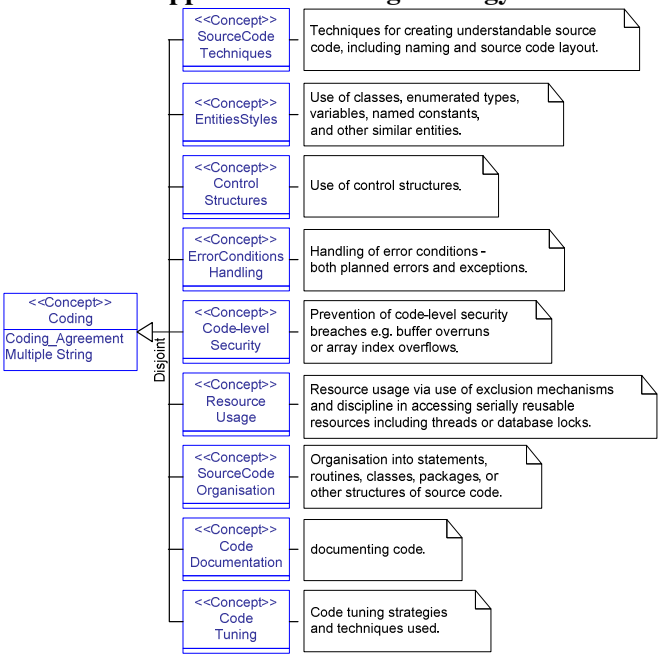
Appendix B2- Component diagrams ontology



Appendix B3 - Activity diagrams ontology



Appendix C- Coding ontology



Professor Tharam S. Dillon is a Fellow of the IEEE, ACS and IE(aust). He is Chairman of the Technical Committee of IEEE IES on Industrial Informatics, and ADCOM member for IEEE IES. He is also Chair of IFIP working Group in Web Semantics. His current research interests include Web semantics, ontologies, Internet computing, e-commerce, hybrid neurosymbolic systems, neural nets, software engineering, database systems, and data mining. He has more than 650 papers published in international conferences and journals and is the author of five books and has another 5 edited books. He is the Editor-in-Chief of the *International Journal of Computer Systems Science and Engineering* as well as the *Engineering Intelligent Systems*. He is the Co-Editor of the *Journal of Electric Power and Energy Systems*.