Journal of Computer Science 7 (3): 320-327, 2011
ISSN 1549-3636
© 2011 Science Publications

Design and Development of Ontology Suite for Software Risk Planning, Software Risk Tracking and Software Risk Control

[1]C.R. Rene Robin and [2]G.V.Uma
[1]Department of Computer Science and Engineering,
Jerusalem College of Engineering, Chennai, 600 100, Tamil Nadu, India
[2]Department of Information Science and Technology,
Anna University, Chennai, 600 025, Tamil Nadu, India

Abstract: Problem statement: Ontology as a conceptual courseware structure may work as a mind tool for effective teaching and as a visual navigation interface to the learning objects. Knowledge visualization is defined as the use of visual representations to transfer knowledge between at least two persons. This study presents the design, development and visualization of ontologies for Software Risk Planning, Software Risk Tracking and Software Risk Controlling. **Approach:** The ontologies are developed using protégé tool, an effective ontology editor and it is represented by the formal knowledge representational language OWL. In order to increase the richness of the knowledge available in the ontologies, its semantic representation is presented using ontology document generator. Finally the ontologies are effectively visualised using OntoViz. **Results:** The ontologies represent the domain knowledge Software Risk Planning, Software Risk Tracking and Software Risk Controlling respectively and is developed with the indention to use it as a knowledge base for effective knowledge representation, Knowledge Management and E-Learning applications. The constructed ontologies are evaluated using quantitative analysis and qualitative analysis. **Conclusion:** Since the average reuse ratio is 0.95, the developed ontologies are highly cohesive. Comparison of concepts and properties used in the ontologies proved that the developed ontologies are concept oriented ontology. The both quantitative and qualitative analysis says, the developed ontologies are ready to use for applications such as E-Learning, Knowledge Management.

Key words: Software risk tracking ontology, e-learning application, software risk planning ontology, visualization approaches, knowledge management, software risk controlling, Software Risk Management Ontology (SRMONTO), ontologies facilitates, visualised using OntoViz, design architecture, Software Risk Management (SRM)

INTRODUCTION

A teacher as the main knowledge provider can preserve the high quality of the knowledge taught, if the knowledge is created in the reusable and sharable form. From an Artificial Intelligence perspective, the knowledge engineering and knowledge representation communities concerned with knowledge elicitation methodologies and the formal notation used to represent knowledge and have dedicated little effort to visualization. Software Risk Management Ontology (SRMONTO) defines common sharable software risk management knowledge. SRMONTO basically provides software risk management concepts-what they are, how they are related and can be related to one another-for representing and communicating over software risk management knowledge. These concepts are widely accepted, thereby facilitating common understanding of the software risk management knowledge by all distributed members. This enables effective ways of sharing and reusing the knowledge for the learner of software risk management. SRMONTO can even assist software engineers to better understand the information. On the other side it creates a better awareness about software risk management among computer science graduates who have been taught the subject. Reaching a harmony of understanding is of benefit to team members in a distributed environment. Ultimately, machines in the form of e-learning applications or software agents provide semantic enabled web service for software risk management; can use the knowledge as well. The common software risk

Corresponding Author: C.R. Rene Robin, Department of Computer Science and Engineering, Jerusalem College of Engineering, Chennai, 600 100, Tamil Nadu, India

management knowledge is semantically shared not only among software engineers, but also among computer based learning systems or software agents.

Building and maintaining software is a risky business (Boehm, 1989). It is the responsibility of the academicians to create awareness among the students about Software Risk Management (SRM). In order to create more meaningful and effective teaching strategies as there is no predefined way to teach SRM, a formal ontology base knowledge base is required. The advantage of the ontology is that it attempts to unify different views on the domain. Knowledge sharing through the software risk management ontology eliminates misunderstandings, miscommunications and misinterpretations. Software risk management ontology presents explicit assumptions concerning the objects referring to the domain knowledge. A set of objects and interrelations and their constraints renders their agreed meanings and properties. SRMONTO consists of Software Risk Identification Ontology, Software Risk Analysis Ontology, Software Risk Planning Ontology, Software Risk Controlling Ontology and Software Risk Tracking Ontology. The authors have developed all the five ontologies and the former two ontologies have been addressed in (Robin and Uma, 2011). The Semantic representation of SRMONTO is described in (Robin and Uma, 2010). The semantic representation of a general ontology is addressed in (Mustapha *et al*., 2010). This study aims to present rest of the three ontolgies such as Software Risk Planning Ontology (SRP ONTO), Software Risk Controlling Ontology (SRC ONTO) and Software Risk Tracking Ontology (SRT ONTO). Its end users are software engineers sharing software risk management domain knowledge as well as learners of software risk management such as teachers and students.

Related work: Software Risk Management (Boehm, 1989) is a discipline whose objectives are to identify, address and eliminate software risk items before they become either threats to successful software operation or major sources of expensive software rework. This work uses, risk management paradigm introduced by Software Engineering Institute as our standard to construct the sub ontologies for the target ontology i.e. software risk management ontology. Risk is omnipresent in each and every step of the software development and all the interactions that software developers carry out. Software development project risk management (Fairley, 2002) should focus on reduction and prevention of risks, continuously assess possible problems, define potential risks, determine what risks are important and deal with them. So a whole project picture is required for successful risk management.

In both computer science and information science, an ontology (Shareha *et al*., 2009) is a data model that represents a set of concepts within a domain and the relationships between those concepts. Basically an ontology (Noy and McGuinness, 2001; Niles and Pease, 2001) defines a common vocabulary for researchers who need to share information in a domain. It includes machine-interpretable definitions (semantics) of basic concepts in the domain and relations among them. Classes are the focus of most ontologies. Classes describe concepts in the domain. A class can have subclasses that represent concepts that are more specific than the superclass. Slots describe properties of classes and instances. So developing an ontology includes defining classes in the ontology, arranging the classes in a taxonomic hierarchy, defining slots and describing allowed values for these slots, filling in the values for slots for instances (Corcho *et al*., 2006). Despite efforts and experience in developing ontologies, there is no agreement on a methodology for building ontologies. The ontology development methodology problem has been investigated in several projects. First attempt was by Gruber (1995) that specifies some principles and criteria for the design of ontologies. We can create a knowledge base by defining individual instances of these classes filling in specific slot value information and additional slot restrictions.

Ontologies can be represented using various methods, in XML (Haw and Lee, 2008), RDF (Brickley and Guha, 2004), OWL (Mousavi *et al*., 2010), (Al-Safadi and Al-Abdullatif, 2010) with Topic Maps. However, these are text-based and usually rather voluminous. In addition, ontologies can also be visualized using UML (Muhairat *et al*., 2010), hyperbolic view, or as a tree, In Mobile Workforce Brokering System (Mousavi *et al*., 2010), the entire domain knowledge is represented semantically using OWL ontology is considered as one of the best examples of OWL ontologies by the authors.

The Protégé project (Musen, 1989a; 1989b; Gennari, 2003) has come a long way since Mark Musen first built the Protégé meta tool for knowledge-based systems in 1987. Protégé is a flexible, well-supported and robust development environment. Using Protégé, developers and domain experts can easily build effective knowledge-based systems and researchers can explore ideas in a variety of knowledge-based domains.

Visualization (Katifori *et al*., 2007) is becoming increasingly important in Semantic Web tools. OWLViz is designed to be used with the Protege OWL plugin. OWLViz integrates with the Protege-OWL plugin, using the same colour scheme so that primitive and defined classes can be distinguished, computed changes to the class hierarchy may be clearly seen and inconsistent concepts are highlighted in red. OWLViz has the facility to save both the asserted and inferred views of the class hierarchy to various concrete graphics formats including png, jpeg and svg. It is used to visualize light-weight ontologies that describe a domain through a set of classes (concepts) and their hierarchical relationships. Also known as taxonomies, such ontologies are frequently used in several domains as classification systems.

MATERIALS AND METHODS

Design architecture of ontologies: Tracking consists of monitoring the status of risks and actions taken to ameliorate risks. Figure 1 shows the design architecture of software risk tracking ontology. In Risk Tracking Ontology there are four properties such as "PartOf", "IsA", "Tracks" and "Has" used to relate the identified concepts. "RiskTracking" concept is at the highest level of the this ontology. Since Risk Manager will do the actual tracking work, "tracks" relationship is used to relate Risk_Manager concept is related with its parent concept "Risk_Tracking".

In Fig. 2, the informal ontology for risk planning has been sketched out manually with the properties "TypeOf", "PartOf", "IsA" and "HasA".

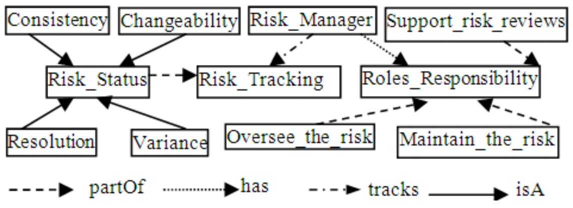

Fig. 1: Software risk tracking ontology

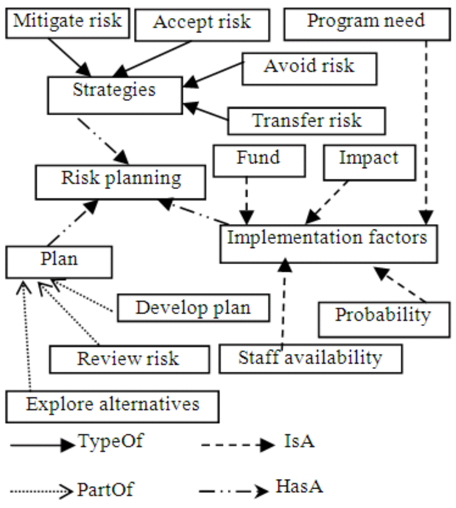

Fig. 2: Software risk planning ontology

Risk planning has four strategies such as "AcceptRisk", "AvoidRisk", "MitigateRisk" and "TransferRisk". Similarly "Fund", "Impact", "Probability", "Needs_Of_Programme" and "Availablity_Of_Staff are various implementation factors of risk planning. And finally "Review_Risk, Develop_Plan and "Explore_alternatives" are part of risk planning.

The goal of risk control is to decide which risk control activities are necessary to take. Figure 3 shows the design architecture of software risk control ontology. In the Software Risk Control Ontology "Risk_Controlling" concept is at top level. It has five subclasses such as Risk_Avoidance, No_Risk_Reducing_Action, Contingency_Plan, Reduce_Loss and Reduce_Event_Probability. All the subclasses are related with main concept by "isA" relationship. In this ontology we used three distinct type of properties such as "isA", "typeOf" and "usedTo". Hence N(Property) is three.

Development of ontologies using protégé: Figure 4 shows the ontology construction process using Protégé. At the left hand side the hierarchal arrangement of risk identification ontology is displayed. When the required concept is highlighted, its semantic, properties and all the other attributes will be displayed at the right side of the editor.

Sir Jorge Cardoso carried a survey on most widely used ontology editors and most widely used domain for ontology development and found that Protégé tool, an effective tool for ontology engineering (Malik *et al.*, 2010) In this study Protégé is used to construct the target ontologies such as SRP ONTO, SRT ONTO and SRC ONTO. It is good to have our ontologies in OWL format to access description logic reasoned and to acquire instances for semantic markup. Figure 5 shows the part of software risk control ontology in OWL format.

Visualization of ontology using ONTOVIZ: The OntoViz Tab configured in protégé tool is used to visualize SRMONTO with the help of a highly sophisticated graph visualization software called Graphviz. The types of visualizations are highly configurable and include picking a set of classes or instances to visualize part of an ontology, displaying slots and slot edges, specifying colors for nodes and edges and when picking only a few classes or instances, you can apply various closure operators (e.g., subclasses, superclasses) to visualize their vicinity. In Fig. 6-8, the constructed ontologies such as software risk planning ontology, software risk tracking ontology and software risk controlling ontology are visualized using protégé editor.

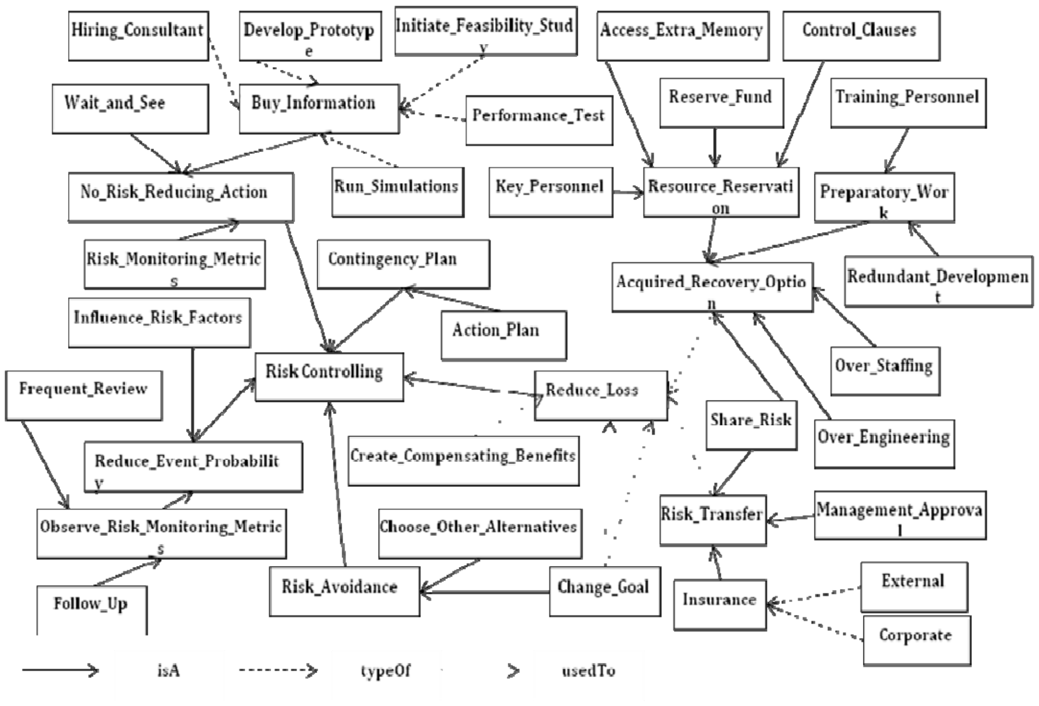

Fig. 3: Software risk control ontology

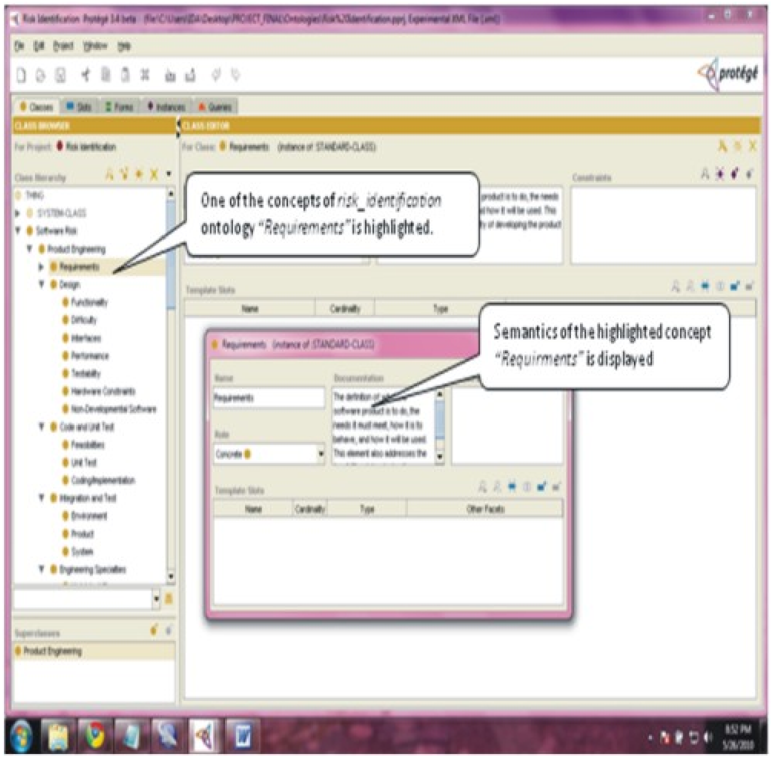
protégé
One of the concepts of risk_identification ontology "Requirements" is highlighted.
Semantics of the highlighted concept "Requirments" is displayed

Fig. 4: Concepts and semantics representation of risk identification ontology using protégé

```
<?xml version="1.0"?>
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   xmlns:owl="http://www.w3.org/2002/07/owl#"
   xmlns="http://www.owl-ontologies.com/unnamed.owl#"
  xml:base="http://www.owl-ontologies.com/unnamed.owl">
  <owl:Ontology rdf:about=""/>
  <owl:Class rdf:ID="Reduction">
    <rdfs:comment
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >taking action to minimise either the likelihood of the risk
developing, or its effects.</rdfs:comment>
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Risk_Control"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Transference">
    <rdfs:subClassOf>
      <owl:Class rdf:about="#Risk_Control"/>
    </rdfs:subClassOf>
    <rdfs:comment
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >transferring the risk to a third party, for example with an insurance
policy.</rdfs:comment>
  </owl:Class>
  <owl:Class rdf:about="#Risk_Control">
    <rdfs:comment
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Risk control is about the methods you can use to get rid of or
manage a risk:</rdfs:comment>
  </owl:Class>
  <owl:Class rdf:ID="Contingency">
    <rdfs:comment
                                ……

  </owl:Class>
  <owl:Class rdf:ID="Prevention">
    <rdfs:comment
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >putting in place measures to stop a problem from occurring or
having impact on a work area or organisation.</rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Risk_Control"/>
  </owl:Class>
</rdf:RDF>
<!-- Created with Protege (with OWL Plugin 3.4, Build 125)
http://protege.stanford.edu -->
```

Fig. 5: A portion of software risk control ontology in OWL format

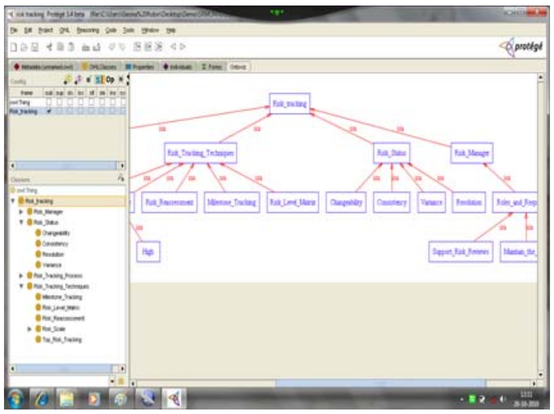

Fig. 6: Risk tracking ontology visualized using OntoViz in protégé editor

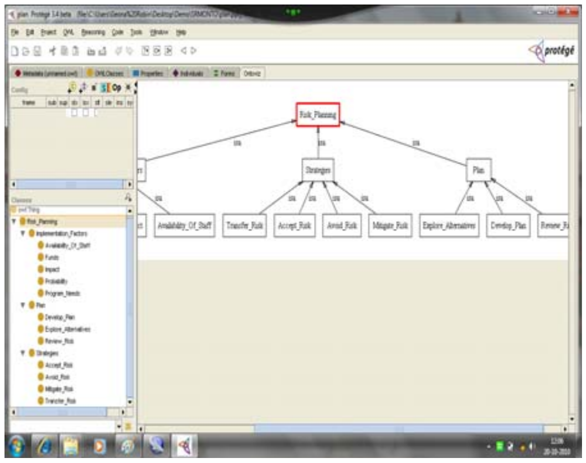

Fig. 7: Risk planning ontology visualized using OntoViz in protégé editor

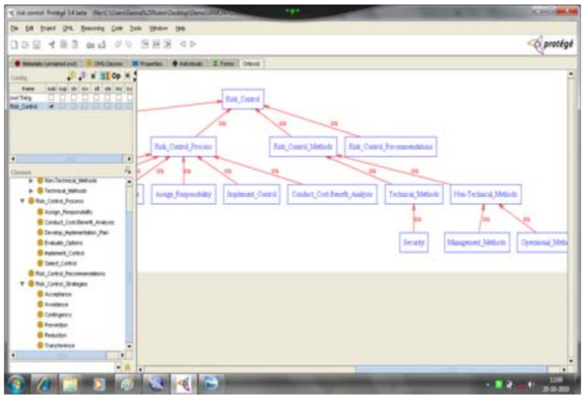

Fig. 8: Risk Control Ontology visualized using ontoviz in protégé editor

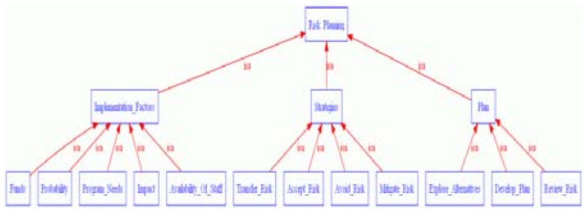

Fig. 9: Visualization of software risk planning ontology

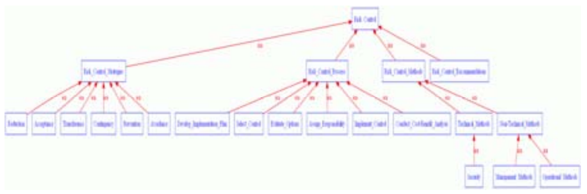

Fig. 10: Visualization of software risk control ontology

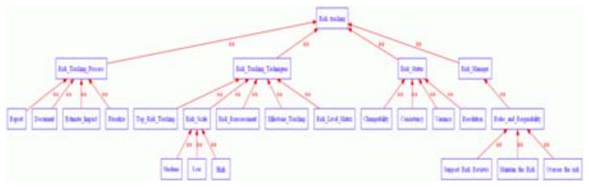

Fig. 11: Visualization of software risk tracking ontology

The visualizations are captured by OntoViz embedded in Protégé. The entire portion of the visualized ontology can be stored as image files by specifying the required path. The Fig. 9-11 show the visual representation of the three ontologies.

RESULTS

The metrics used for quantitative evaluation (Vijay and Manoharan, 2009), (Longo and Oreste, 2010), are classified into four categories such as class metrics property metrics ratio metrics and axiom metrics. Each category has a set of parameters. Table 1 shows the results of quantitative analysis made for class metrics. Similarly the results for property metrics, ratio metrics and axiom metrics are shown in Table 2-4 respectively.

Table 1: Statistics of class metrics

METRICS	SRP ONTO	SRT ONTO	SRC ONTO
NoC	19.00	25.00	40.00
NoLC	14.00	18.00	16.00
NoRC	1.00	1.00	1.00
NoSpC	5.00	7.00	6.00
NoSbC	18.00	24.00	21.00
NoHvR	7.00	6.00	9.00
AvDoI	2.21	2.33	2.13
MxDoI	3.00	3.00	3.00

Table 2: Statistics of property metrics

Property metrics	SRP ONTO	SRT ONTO	SRC ONTO
NoOp	4	4	3

Table 3: Statistics of ratio metrics

Ratio metrics	SRP ONTO	SRT ONTO	SRC ONTO
Specialization ratio	3.60	3.43	3.50
Reuse Ratio	0.95	0.96	0.95

Table 4: Statistics of axiom metrics

Axiom Metrics	SRP ONTO	SRT ONTO	SRC ONTO
Has Value Restrictions	7	6	9

Table 5 : Comparison between SRIONTO with three existing

Quantitative Metrics	Crypto Onto	Software Testing Onto	E-R Model Onto	Developed Suite
NoC	21.00	39.00	17.00	84.00
NoSpC	9.00	10.00	7.00	18.00
NoSbC	20.00	38.00	16.00	63.00
MxDoI	4.00	4.00	6.00	3.00
Reuse Ratio	0.95	0.97	0.94	0.95
Specialization Ratio	2.22	3.80	2.28	3.51

Class metrics: The following metrics are identified as class metrics:

NoC:	No of Classes
NoLC:	Number of Leaf Classes
NoRC:	Number of Root Classes
NoSpC:	Superclasses
NoSbC:	Number of Subclasses
NoEqC:	Number of Classes with Equivalent Class Expressions
NoHvR:	Number of Classes with 'HasValue' restriction Axioms
AvDoI:	Average Depth of Inheritance
MxDoI:	Max Depth of Inheritance

Property metrics: The following metrics are identified as property metrics:

NoDtP:	Number of Datatype Properties
NoOP:	Number of Object Properties
NoAP:	Number of Annotation Properties

Ratio metrics: The following metrics are identified as ratio metrics. This metrics is depends on class metrics and it is a type of indirect metrics:

Specialization ratio = NoSbC/NoSpC

Reuse ratio = NoSbC/NoC

Axiom metrics: There are two metrics are identified as axiom metrics:

Equivalent class

Has value restrictions

The constructed ontology suite is compared with three existing educational ontologies such as Cryptography Ontology (Takahashi *et al*., 2005), Software Testing Ontology (Zhu and Huo, 2004) and E-R Model Ontology (Boyce and Pahl, 2007). The comparison result of SRIONTO with above said educational ontologies is presented in Table 5.

DISCUSSION

The statistical analysis says SRP ONTO has concepts four times than the number of its properties. SRT ONTO has the number of concepts more than three times higher than the number of its properties. Similarly SRC ONTO has the number of concepts more than five times higher than the number of its properties. Hence in the resultant ontology, NoC will be higher than NoP and is presumed to be a concept oriented ontology.

The concept and property analysis shown in Fig. 12 says SRP ONTO has concepts five times than its number of properties, SRT ONTO has concepts six times than the number of its properties and SRC ONTO has concepts thirteen times than its number of properties. Since in all cases, NoC is higher than NoP,

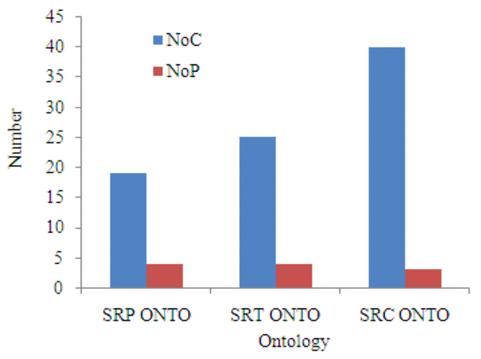

Fig. 12: Concept and property analysis

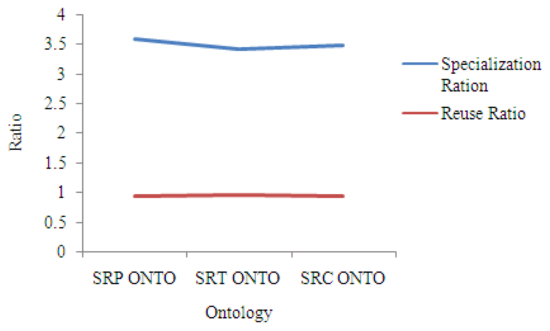

Fig. 13: Analysis of ratio metrics

the constructed ontology suite presumed to be a concept oriented ontology. Figure 13 shows the analysis of ratio metrics. The reusability ratio of the constructed ontologies is linear because all the sub ontologies have the reuse ratio more that 95%. It shows that the constructed ontologies are highly cohesive. The cohesion metrics (Yao *et al*., 2005) examine the fundamental quality of cohesion as it relates to ontologies.

The following inferences have been made from the comparison presented in Table 5. The number of classes, subclasses and superclasses of the developed ontology suite is considerably higher than the existing educational ontologies. The developed ontology suite has a higher level of reusability than the existing three educational ontologies taken for comparison. Finally, it makes better use of specialization of its classes, indicated by its higher specialization ratio.

CONCLUSION

The complex, abstract and interrelated contents yet represented verbally are transformed into visual representations through the proposed work. It includes design, development and visualization of three sub ontologies of SRMONTO which serves as the knowledge repository of software risk management. The ontology suite has been constructed based on the information available in the literature and the experience of various software developers. The main objective of this ontology suite is to use it as a content ontology to express formal domain model in e-learning to teach software risk management subject. The general e-learning system is discuss in (Jabr and Omari, 2010). As the knowledge representation mechanism this ontology suite facilitates visualizing intellectual structures based on widely available sources and augments knowledge visualization approaches that focus on documents and concepts. Users can use this visual representation to discover patterns and make valuable connections between data. In another follow on study we will describe the creation of a student interface for an educational system that is one of the authors' primary goals. The identified metrics are categorized into class metrics, property metrics, ratio metrics and axiom metrics. The reusability of each sub ontology is calculated separately and found that the ontology is ready to use for the intended applications.

REFERENCES

Al-Safadi, L.A.E. and N.A.O. Al-Abdullatif, 2010. Educational advertising ontology: A domain-dependent ontology for semantic advertising networks. J. Comput. Sci., 6: 1070-1077. DOI: 10.3844/jcssp.2010.1070.1077

Boehm, B., 1989. Software risk management. Lect. Notes Comput. Sci., 387: 1-19. DOI: 10.1007/3-540-51635-2_29

Boyce, S. and C. Pahl, 2007. Developing domain ontologies for course content. Edu. Technol. Soc., 10: 275-288.

Brickley, D. and R.V. Guha, 2004. RDF Vocabulary Description Language 1.0: RDF Schema. W3C Working Draft. http://duboc.me/fr/node/179

Corcho, O., M. Fernández-López and A. Gómez-Pérez, 2006. Ontological engineering: Principles, methods, tools and languages. Ontolog. Software Eng. Software Technol. 2006: 1-48, DOI: 10.1007/3-540-34518-3_1

Fairley, R., 2002. Risk management for software projects. IEEE Software 11: 57-67. DOI: 10.1109/52.281716

Gennari, J.H., M.A. Musen, R.W. Fergerson, W.E. Grosso and M. Crubezy *et al*., 2003. The evolution of Protégé: An environment for knowledge-based systems development. Int. J. Hum. Comput. Stud., 58: 89-123. DOI: 10.1016/S1071-5819(02)00127-1

Gruber, T.R., 1995. Toward principles for the design of ontologies used for knowledge sharing. Int. J. Hum.-Comput. Stud., 43: 907-928.

Haw, S.C. and C.S. Lee, 2008. TwigINLAB: A decomposition-matching-merging approach to improving xml query processing. Am. J. Applied Sci., 5: 1199-1205. DOI: 10.3844/ajassp.2008.1199.1205

Jabr, M.A. and H.K.A. Omari, 2010. E-learning management system using service oriented architecture. J. Comput. Sci., 6: 285-295. DOI: 10.3844/jcssp.2010.285.295

Katifori, A., C. Halatsis, G. Lepouras, C. Vassilakis and E. Giannopoulou, 2007. Ontology visualization methods-a survey. ACM Comput. Surveys, DOI: 10.1145/1287620.1287621

Longo, S. and P. Oreste, 2010. Ceppo morelli block-falls probability study to support the decision of excavating a by-pass tunnel. Am. J. Eng. Applied Sci., 3: 723-727. DOI: 10.3844/ajeassp.2010.723.727

Malik, S.K., N. Prakash and S.A.M Rizvi, 2010. Developing an University Ontology in Education Domain using Protégé for Semantic Web, Int. J. Eng. Sci. Technol., 2: 4673-4681.

Mousavi, A., M.D.J. Nordin and Z.A. Othman, 2010. An ontology driven, procedural reasoning system-like agent model, for multi-agent based mobile workforce brokering systems. J. Comput. Sci., 6: 557-565. DOI: 10.3844/jcssp.2010.557.565

Muhairat, M.I., R.E.A. Qutaish and A.A. Abdelqader, 2010. UML diagrams generator: A new CASE tool to construct the use-case and class diagrams from an event table. J. Comput. Sci., 6: 253-260. DOI: 10.3844/jcssp.2010.253.260

Musen, M.A., 1989a. Automated Generation of Model-Based Knowledge-Acquisition Tools. 1st Edn., Hyperion Books, USA., ISBN-10: 0273088122, pp: 300.

Musen, M.A., 1989b. Automated support for building and extending expert models. Mach. Learn., 4: 347-375. DOI: 10.1007/BF00130719

Mustapha, A., M.N. Sulaiman, R. Mahmod and M.H. Selamat, 2010. Corpus-based analysis on cross-domain experiments in classification-and-ranking generation. J. Comput. Sci., 6: 1326-1333. DOI: 10.3844/jcssp.2010.1326.1333

Niles, I. and A. Pease, 2001. Towards a Standard Upper Ontology. Proceedings of the International Conference on Formal Ontologies in Information Systems, (FOIS'01), ACM Press, New York, USA., pp: 2-9. DOI: 10.1145/505168.505170

Noy, N and D.L. McGuinness, 2001. Ontology development 101: A guide to creating your first ontology. Technical Report KSL-01-05, Stanford Medical Informatics, Stanford. DOI: 10.1.1.136.5085

Robin, C.R.R. and G.V. Uma, 2010. Ontology based semantic knowledge representation for software risk management. Int. J. Eng. Sci. Technol., 2: 5611-5617.

Robin, C.R.R. and G.V. Uma, 2011. Development of educational ontology for software risk analysis. Proceedings of the International Conference on Communication, Computing and Security, Feb. 12-14, ACM New York, NY, USA., pp: 610-615. DOI: 10.1145/1947940.1948067

Shareha, A.A.A., M. Rajeswari and D. Ramachandram, 2009. Multimodal integration (image and text) using ontology alignment. Am. J. Applied Sci., 6: 1217-1224. DOI: 10.3844/ajassp.2009.1217.1224

Takahashi, Y., T. Abiko, E. Negishi, G. Itabashi and Y. Kato *et al*., 2005. An ontology-based e-learning system for network security. Proceedings of the 19th International Conference on Advanced Information Networking and Applications, (AINA'05), IEEE Computer Society Washington, DC, USA., pp: 197-202. DOI: 10.1109/AINA.2005.116

Vijay, J.F. and C. Manoharan, 2009. Initial hybrid method for analyzing software estimation, benchmarking and risk assessment using design of software. J. Comput. Sci., 5: 717-724. DOI: 10.3844/jcssp.2009.717.724

Yao, H., A.M. Orme and L. Etzkorn, 2005. Cohesion metrics for ontology design and application. J. Comput. Sci., 1: 107-113. DOI: 10.3844/jcssp.2005.107.113

Zhu, H. and Q. Huo, 2004. Developing A Software Testing Ontology in UML for A Software Growth Environment of Web-Based Applications. Developing a Software Testing Ontology. http://cms.brookes.ac.uk/staff/HongZhu/Publications/SEUMLXML.pdf