Design and Development of SRIONTO: An Educational Ontology Representing Software Risk Identification Knowledge

C.R.Rene Robin
Assistant Professor & Head
Department of Computer Science and Engineering
Jerusalem College of Engineering, Chennai-100
Tamil Nadu, India

G.V.Uma
Professor & Head
Department of Information Science and Technology
Anna University, Chennai – 600 025
Tamil Nadu, India

ABSTRACT
Knowledge can be captured and made available to both machines and humans by an ontology. Ontology can be served as a structured knowledge representation scheme, capable of assisting the construction of a personalized learning path. This paper describes the processes of conceptualization and specification, or building of, an ontology. The domain for which the ontology has been constructed is software risk identification. The required concepts, the semantic description of the concepts and the interrelationship among the concepts along with all other ontological components have been collected from various literatures and experience of the people from software industry. From which, a taxonomy has been constructed by using the property 'isA' and the design architecture for the required ontology has also been sketched out manually with nearly four different types of properties. In order to reduce implementation efforts, the Protégé platform, a scalable and integrated framework for ontological engineering, has been used to construct the ontology. The constructed ontology has been represented in owl format, which makes it more machine understandable. Then the semantic representation of the knowledge has been made using the OWL document generator, which automatically generates a set of documents from the ontology. In order to understand the knowledge in more detailed way again the ontology has been visualized using ontoviz tool.

General Terms
Knowledge Management, Software Engineering and Artificial Intelligence.

Keywords
Ontology, Protégé, Software Risk Identification Ontology (SRIONTO), Knowledge Management, E-learning, OWL, and Visualization.

1 INTRODUCTION
Any e-learning or knowledge management application needs the required knowledge to be represented good enough to help its users according to their present and expected competency state. Without a structural model for a domain, target knowledge will be unstructured pieces of text, which are difficult to use in applications like knowledge management and e-learning systems. The knowledge domain in an e-learning or knowledge management application can be structured in many ways: dictionaries, thesaurus, book summary, library catalog, indexes and metadata, knowledge graphs, ontologies, etc.

The tree organization of a knowledge domain is an important property that can significantly reduce the processing, but it is insufficient to describe the rich network of relations that ties the concept structures. It needs to be complemented with relations between concepts and axioms in order to sustain more refined mechanism of conceptual matching and inference. With the advancement of artificial intelligence technologies, ontology technologies enable a linguistic infrastructure to represent conceptual relationships between course materials. Ontology technology is considered to be a highly suitable means of supporting educational-technology systems [Mizoguchi & Bourdeau, 2000].

Drawing on the previous work [Marvin J. Carr, 1993] and the experience gained in numerous projects, this paper presents an ontology for software risk identification (SRIONTO) that combines the concepts of knowledge, semantic description of each concepts, and properties. It provides ways to annotate semantically resources in e-learning and knowledge management environments, in particular to define semantics of individual concepts, prerequisites and goals for activities and resource content, for e-learning and knowledge management applications.

The rest of the paper is organized as follows. Section 2 of this paper discusses on the related work that have been done on this area. Section 3 describes the taxonomical arrangement of the concepts. Section 4 describes the design architecture of SRIONTO. Section 5 starts with ontology construction process using protégé software Section 6 presents the OWL representation of the developed ontology. Section 7 describes the semantic knowledge representation of the developed ontology. Section 8 describes the visualization of software risk identification ontology. The evaluation result of SRIONTO is given in section 9 and section 10 presents the conclusion of the paper with future work.

2 RELATED WORK
Knowledge engineering addresses the structuring and representation of knowledge [Sowa, 2000]. Ontologies have emerged as a central technique [Daconta et al., 2003] for knowledge integration, sharing and reuse. Ontologies help us to make the knowledge that is represented in learning content explicit. Knowledge is central in learning; learners consume content to acquire knowledge. Knowledge is also important for the content developer, as content can be an elaboration of explicitly represented knowledge, and therefore a central ingredient for the development of content. Ontologies can fill the gap between authors and content and instruction representations in authoring systems [Mizoguchi & Bourdeau, 2000]. Ontology [Huan Wang, et. al., 2010] is not just a hierarchical collection of concepts with parent-child relation. In the recent years Ontologies [Christopher Brewster, et. al., 2007]

have become the knowledge representation medium of choice for a range of computer science specialities including the Semantic Web, Agents, and Bio-informatics.Ontologies can be used to represent knowledge about content, supporting instructors in creating content or learners in accessing content in a knowledge-guided way.

While ontologies exist for many subject domains, their quality and suitability for the educational context might be unclear. For numerous subjects, ontologies do not exist. Ontology for C Programming [Sergey Sosnovsky et. al, 2005], Cryptography Ontology [Yoshihito Takahashi, 2005], Software Testing Ontology [Hong Zhu et.al., 2004], E-R Model Ontology [Boyce S., 2007], Computer Networks Ontology [Ling Jiang, 2008], Upper Level Ontology for Chemistry [Colin Batchelor, 2008] and Ontology for Software Engineering [Pornpit Wongthongtham et.al, 2009] are few exiting educational ontologies. [Marko Grobelnik, et.al., 2008] demonstrated one possible scenario how contextual information can be exploited during semi-automatic ontology construction from text corpora. Boyce, S., & Pahl, C. [2007] presented a method for domain experts rather than ontology engineers to develop ontologies for use in the delivery of courseware content.

Software Risk Identification Ontology (SRIONTO) is like other OWL ontologies in other domains, which consist of instances, properties and classes. It consists of instances representing specific risk factors, properties representing binary relations held among software risk identification concepts and classes representing the software risk identification concepts. The key ingredients that make up the software risk identification ontology are a vocabulary of basic software risk identification terms and a precise semantic description of what those terms mean.

3 TAXONOMY

Taxonomy is a way of classifying or categorizing a set of things using a hierarchical structure, which is a tree like structure, with the most general category as the root of the tree. Each node, including the root note, is information entity that represents some object in the real world that is being modeled. In this paper, the hierarchical structure i.e. the taxonomical arrangement of concepts has been structured by considering the following rules

Rule 1 : Concepts of one level should be linked to their parent concept by the relationships "is-a". This means that concepts of one layer should have similar nature and level of granularity.

Rule 2 : Cross-links should be avoided as much as possible.

4 DESIGN OF SRIONTO

While designing the software risk Identification ontology, the design criteria such as clarity, coherence, extendibility, minimal encoding bias and minimal ontological commitment has been considered and intended to use this ontology for knowledge-sharing. Fig. 1 shows the overview of design architecture of risk identification ontology that has been sketched out manually with 86 concepts and four different properties. In SRIONTO, the concept "Software_Risk" resides at the top layer and the three main basic components of risk identification [M. J.Carr, 1993] such as product engineering, development environment and program constraints along with identification techniques form four subclasses namely "Product_Engineering", "Development_Environment", "Program_Constraints" and "Risk_Identifcation_Techniques".

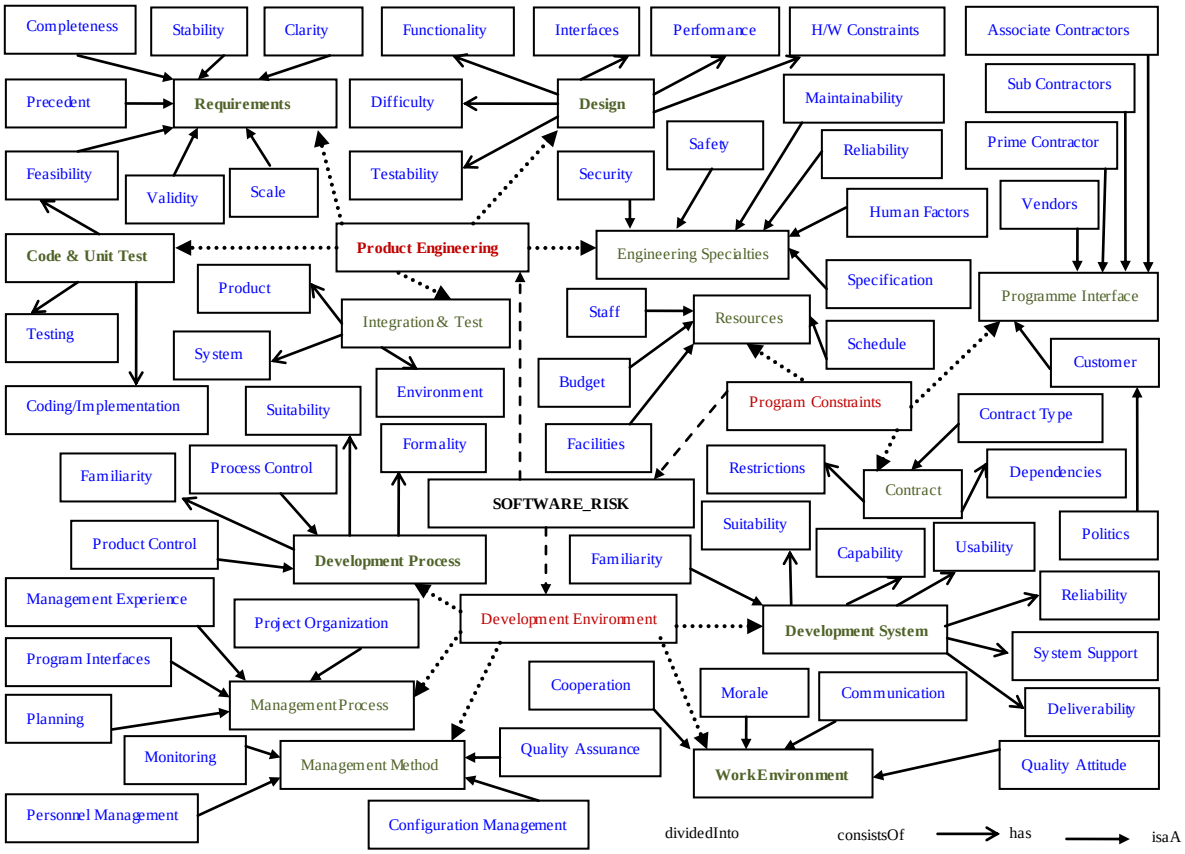

Fig.1: Design Architecture of SRIONTO

There are five subclasses for "Product_Engineering" concept that are used to define complex concepts in other classes of the ontology namely "Requirements", "Integration_Test", "Design", "Code_UnitTest" and "Engineering_Specialist". Similary subclasses of "Development_Environment" concept that are used to define complex concepts in other classes of the ontology namely "Development_Process", "Management_Process", "Management_Method", "Work_Environment" and "Development_System". In the next level "Suitability", "Familiarity", "Product_Control", "Process_Control" and "Formality" are presented as the subclasses of "Development_Process".

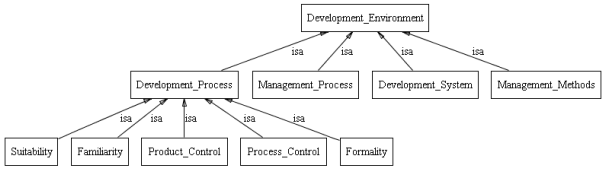

Fig.2: Ontological Model for Development_Environment Concept

The Ontological representation of the concept "Development_Environment" is shown in the Figure 2. The parameters of each concept in Ontology can be represented by the tuple C[P_1, P_2, P_3,…,P_n]. Where C is the concept's name, P_1, P_2, P_3,…,P_n are it parameters or sub classes of C and n is the total number of subclasses. Since "Software_Risk" is the main concept of SRIONTO, at the first level its parameters are represented by the following tuple.

Software_Risk[Product_Engineering, Development_Environement, Program_Constraints, Risk_Indentification_Techniques]

The paraments of the "Product_Engineering" concept are represented as follows.

Product_Engineering[Code&Unit_Test, Integration&Test, Requirements, Design, Engineering-Specialties]

The sample concepts selected randomly from all the three levels of SRIONTO and its corresponding semantic descriptions are displayed in table 1.

Table 1. Sample Semantic Description for Existing Concepts

Concepts	Semantic Description
Product Engineering	Product engineering refers to the system engineering and software engineering activities involved in creating a system that satisfies specified requirements and customer expectations.
Development Environment	It addresses the project environment and the process used to engineer a software product.
Program Constraints	Factors that may be outside the control of the project but can still have major effects on its success or constitute sources of substantial risk.
Requirements	Attributes of the requirements element cover both the quality of the requirements specification and also the difficulty of implementing a system that satisfies the requirements.
Completeness	Missing or incompletely specified requirements such as a requirements document with many functions or inadvertently omitted requirements.
Stability	The stability attribute refers to the degree to which the requirements are changing and the possible effect changing requirements and external interfaces will have on the quality, functionality, schedule, design, integration, and testing of the product being built.
Clarity	This attribute refers to ambiguously or imprecisely written individual requirements that are not resolved until late in the development phase.

Apart from the existing risk factors some more risk factors have also been identified and they are added in SRIONTO to enrich it. A set of newly identified concepts, its super class, the semantic description of the concept along with its graphical representation are tabulated in Table 2.

Table 2. Semantic Description of Newly Identified Concept

Identified Concepts	SubClassOf	Description	Graphical Representation
Wrong_Time_Estimation (WTS)	Schedule	The assumed duration to complete the project was not accurate	Schedule ← willAffect ← WTS
Improper_Resource_Tracing (IRT)	Schedule	The way in which the resource to be fetched was not in proper method	Schedule ← willAffect ← IRT
Complex_Functionalities_Identification (CFI)	Schedule	To find the intricate modules	Schedule ← willAffect ← CFI
Unexpected_Project_Scope (UPS)	Schedule	The main functionalities of the project was not analysed.	Schedule ← willAffect ← WTS
Wrong_Budget_Estimation (WBE)	Budget	Estimated cost out bounded	Budget ← willAffect ← WTE
Cost_Overruns (CO)	Budget	The actual cost exceeded the estimated cost.	Budget ← willAffect ← CO
Project_Scope_Expansion (PSE)	Budget	When the scope of the project expands the development cost will automatically increased.	Budget ← willAffect ← PSE

5 DEVELOPMENT OF SRIONTO USING PROTÉGÉ

Ontology is comprised of four main components: concepts, instances, relations and axioms. Figure 3 shows the home page of the protégé editor and the creation of the software risk identification ontology. Various tabs like the classes tab, slots tab, forms tab, instances tab and the queries tab are used for the ontology development . Classes or concepts are abstract groups, sets, or collections of objects. They may contain individuals, other classes, or a combination of both. Classes represent concepts in the domain and not the words that denote these concepts. Here top down development process is handled which starts with the definition of the most general concepts in the domain and subsequent specialization of the concepts.

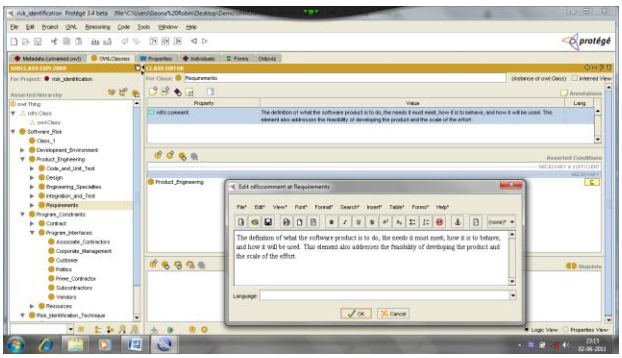

Fig.3: Creating Software Risk Identification Ontology using Protégé.

6 OWL REPRESENTATION OF SRIONTO

Web Ontology Language (OWL) [A. Grigoris, et.al., 2003], [D. McGuinness, et.al., 2003], [Dean M., et.al., 2003], [M. Smith et. al., 2003], [P. Patel-Schneider, et.al., 2003] and [Rajiv Pandey, et.al., 2011] which is a language for processing web information and provides a richer integration and interoperability of data among communities and domains.

All classes in software risk identification ontology are subclasses of class 'Thing'. Its notation description is the same with an ontology class notation. OWL documents are usually called OWL Ontologies, and the elements of which are Namespaces, Housekeeping, Classes, Properties, Property restrictions, Enumerations and Instances. Because OWL is written in RDF, and RDF is written in XML, so OWL documents start with several namespace declarations using RDF, XML Namespace, and URIs. rdf:RDF is the root element of a OWL Ontology, and also specifies a number of namespaces. Fig. 4 shows the Namespace declaration of the developed ontology.

```
<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#"
```

Fig.4: Representation of Namespace Declaration

After rdf:RDF, some declarations to identify namespaces associated with this Ontology could be added. The effect of all of these namespaces is that such prefixes as owl and rdf should be understood as referring to things drawn from following namespaces, as for example http://www.w3.org/2002/07/owl#.

In OWL, classes are defined by using an owl:Class element that is a subclass of rdfs:Class. Figure 5 shows the owl representation of Super Class and Sub Class relationship. The concept "Design" which is the subclass of the concept "Product_Engineering" and its data type i.e. string are represented.

```
<owl:Class rdf:about="#Design">
<rdfs:subClassOf rdf:resource="#Product_Engineering"/>
<rdfs:comment
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
>The translation of requirements into an effective design within
project and operational constraints</rdfs:comment>
</owl:Class>
```

Fig.5: Sample Representation for SuperClass-SubClass Relationship

One of the power elements of OWL is a "owl:disjointWith", which is missing from RDFS, and is used to disjoint one class from others. "owl:equivalentClass" is another element that could be used to establish equivalence between classes. Along with the elements discussed above, there are two more predefined classes, such as owl:Thing (which defines everything) and owl:Nothing, which is empty set. A fragment of OWL representation of SRMONTO is shown in figure 6.

```
<?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="Dependencies">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Contract"/>
    </rdfs:subClassOf>
    <rdfs:comment
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >This attribute refers to the possible contractual
dependencies on outside contractors or vendors, customer-
furnished equipment or software, or other outside products and
services.</rdfs:comment>
  </owl:Class>
                                  …
<owl:FunctionalProperty rdf:about="#Risk_Factors">
    <rdfs:label
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Risk Factors</rdfs:label>
    <rdfs:comment
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >a. Stability, Quality, Functionality, Schedule, Integration,
Design, Testing </rdfs:comment>
    <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProper
ty"/>
    <rdfs:domain>
      <owl:Class>
        <owl:unionOf rdf:parseType="Collection">
          <owl:Class rdf:about="#Stability"/>
          <owl:Class rdf:about="#Completeness"/>
          <owl:Class rdf:about="#Validity"/>
        </owl:unionOf>
```

```
    </owl:Class>
  </rdfs:domain>
  <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
 </owl:FunctionalProperty>
</rdf:RDF>
<!-- Created with Protege (with OWL Plugin 3.4, Build 125)
http://protege.stanford.edu -->
```

Fig.6: Portion of SRIONTO Represented in OWL Format

In OWL it is possible to talk about Boolean combinations such as union, intersection, or complement of classes. For example it can be said that "Stability", "Completeness", and "Validity" are collection of "Risk_Factors".

7 SEMANTIC KNOWLEDGE REPRESENTATION OF SRIONTO

A total of 86 source files have been automatically generated by OWL document generator and each represents a concept in the ontology. The semantic representation [C.R.Rene Robin et.al., 2010] of software risk identification ontology not only gives the hierarchal structure but gives five different types of knowledge. After selecting a particular concept, the name of the concept is displayed at the top as 'class' name followed by the semantic description of the concept is displayed. Then the taxonomical arrangement of the domain starting from the top level concept to the selected concept is displayed from the ontology followed by the parent concept is displayed and at the end, the descendents of the selected concept is displayed.

With software risk identification ontology, the software risk identification terms can be annotated or parsed with SRM concepts and can recall the necessary details and relevant information. Figure 7 shows that the insurance terminology "Software_Risk" is annotated as class along with its semantic description or axiom precisely. The taxonomical arrangement is annotated with two levels. Because the annotated concept is present in the first level of the ontology and its super class is "owl:Thing", which is the root class. The term 'owl:Thing' is annotated as super class of the highlighted concept "Software_Risk". Finally the sub terms "Development_Environment", "Product_Engineering", "Program_Constraints", and "Risk_Identfication_Technique" are annotated as sub classes of "Software_Risk" concept.

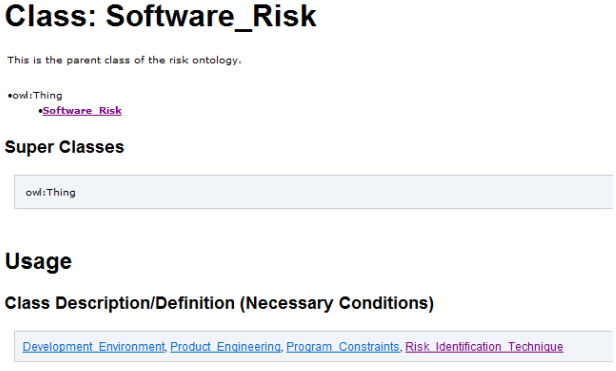

Fig.7: Representation of Software_Risk Concepts

Similarly in Figure 8, the insurance terminology "Engineering_Specialties" is annotated as class along with its semantic description or axiom precisely. The taxonomical arrangement is annotated with four levels. Because the annotated concept is the sub class of "Product_Engineering", which is the sub class of "Software_Risk", which is the sub class of the root class "owl:Thing". Further, the term 'Product_Engineering" is annotated as super class of the insured concept "Engineering_Specialties". Finally the sub terms "Human_Factor", "Maintainability", "Reliability", "Safety", "Security", and "Specifications" are annotated as different sub classes of the insured concept "Engineering_Specialties".

Fig.8: Representation of Engineering_Specialties Concept

8 VISUAL REPRESENTATION OF SRIONTO

The OntoViz Tab configured in protégé tool is used to visualize SRIONTO with the help of 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. Figure 9 shows the visualization of the SRIONTO using OntoViz in Protégé editor.

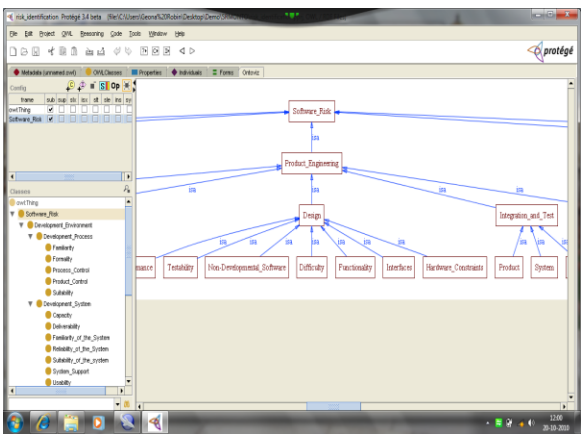

Fig.9: SRIONTO Visualized using ONTOViz in Protégé Editor

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. Figure 10 shows a portion of the visualization of the SRIONTO.

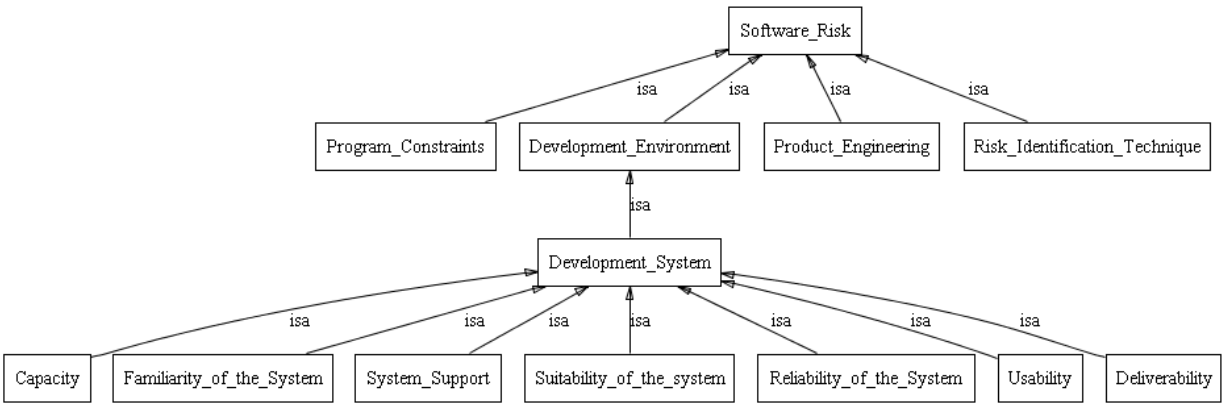

Fig.10: A portion of the Visualization of Software Risk Identification Ontology

9 EVALUATION RESULTS

The content of ontologies should be evaluated before using it in applications. In general ontologies can be evaluated with the help of several metrics. In this paper, the class metrics, property metrics and ratio metrics are considered for quantitative evaluation of SRIONTO. The metrics addressed in semiotics approach [Stamper et al., 2000] and earlier work [Burton-Jones et. al., 2005] are taken for this qualitative evaluation. Further the SRIONTO is also been compared with three different existing educational ontologies.

The number of classes (NoC), number of leaf classes (NoLC), number of properties (NoP), maximum depth of inheritance (MxDoI), number of sub classes and the reuse ration are the various parameters considered for the quantitative analysis.

The reuse ratio of the SRIONTO is calculated by the following formula.

Reuse Ratio = Number of Sub Classes / Number of Classes

Hence Reusability rate of SRIONTO is calculated as 96 %.

The evaluation result of quantitative analysis using above said metrics is shown in Table 3.

Table 3. Quantitative Analysis of SRIONTO

NoC	NoLC	NoP	MxDoI	NoSbC	Reuse Ratio
86	17	4	3	22	0.96

The next analysis says, SRIONTO has concepts twenty times than the number of properties. Since NoC is higher than NoP, SRIONTO presumed to be a concept oriented ontology.

The SRIONTO is again compared with existing educational ontologies such as Cryptography Ontology, Software Testing Ontology and E-R Model Ontology. The comparison result of SRIONTO with above said educational ontologies is presented in Table 4.

Table 4 : Comparison between SRIONTO with three existing educational ontologies.

Quantitative Metrics	Crypto Onto	Software Testing Onto	E-R Model Onto	SRI Onto
NoC	21	39	17	86
NoSpC	9	10	7	22
NoSbC	20	38	16	55
AvDoI	2.91	2.35	3.80	4.14
MxDoI	4	4	6	5
Reuse Ratio	0.95	0.97	0.94	0.98
Specialization Ratio	2.22	3.8	2.28	2.50

The following inferences have been observed from the above mentioned comparison. The number of classes, subclasses and superclasses in SRIONTO is considerably higher than the existing educational ontologies. The maximum and average depth of inheritance is higher in SRIONTO, which is an indication of the ontology being more detailed and accurate to its domain. SRIONTO presents a higher level of reusability than the existing three educational ontologies taken for comparison. Finally, SRIONTO makes better use of specialization of its classes, indicated by its higher specialization ratio.

As the educational ontology, SRIONTO is again evaluated by a set of 55 students, who are the primary users of this knowledge base. A set of metrics mentioned in the semiotics approach and earlier work are taken for the qualitative evaluation. The list of metrics and the responses received from the students are shown in the Table 5. Further the result is also represented using the bar chart in Fig. 10. The sample evaluation sheet is given in Appendix – I.

Table 5 : Evaluation based on semiotic approach

Qualitative Metrics	Very High	High	Medium	Low
Lawfulness	40	14	1	0
Richness	37	16	2	0
Interpretability	34	19	2	0
Consistency	37	16	2	0
Clarity	36	15	4	0
Comprehensiveness	39	13	3	0
Accuracy	38	15	2	0
Relevance	36	16	3	0

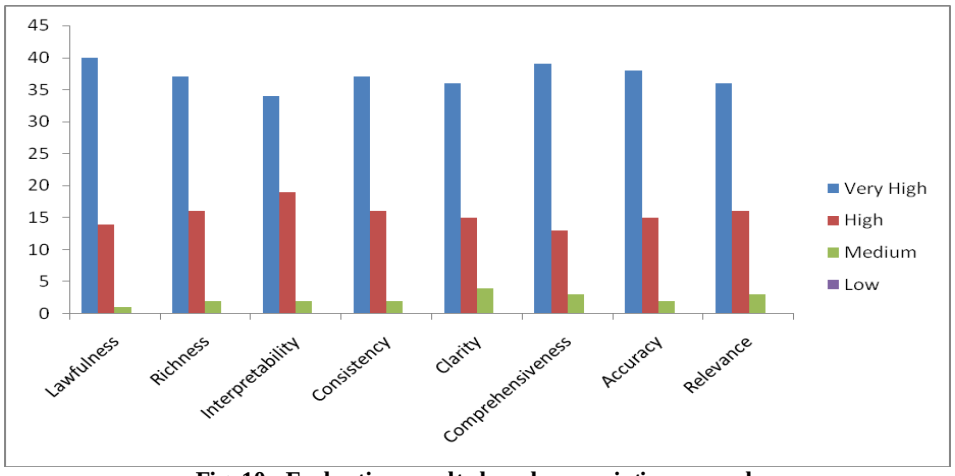

Fig. 10 : Evaluation results based on semiotic approach

10 CONCLUSION

The ontology is designed based on the domain knowledge of software risk identification to mediate the communications between the agents. It was represented in OWL to codify the knowledge of the domain for agents' processing of messages. During the testing and validation of the prototype system, it is realized that OWL representation is not very readable for domain experts to validate the ontology. So the developed ontology has been processed again and semantically represented using OWL document generator and visualized using OntoViz. The software risk identification knowledge, formed into SRIONTO, helps communications among team members and learner and provides consistent understanding of the domain knowledge. Finally the ontology is evaluated both qualitatively and quantitatively by a set of metrics. The evaluation result shows that the reusability of concepts of SRIONTO is 96%. It says that the ontology is ready to use for the intended applications. In another follow on paper, the authors will describe the creation of a student interface for an educational system that is one of the primary goals of the authors'.

11 REFERENCES

[1] Boyce, S., and Pahl, C. 2007, Developing Domain Ontologies for Course Content, Educational Technology & Society, 10 (3), 275-288.

[2] Burton-Jones, A., Storey, V.C., Sugumaran, V., and Ahluwalia, P. 2005, "A semiotic metrics suite for assessing the quality of ontologies," Data & Knowledge Engineering (55:1), PP. 84-102.

[3] Christopher Brewster, Kieron O'Hara, 2007, Knowledge representation with ontologies: Present challenges—Future possibilities, Int. J. Human-Computer Studies, 65, 563–568.

[4] Colin Batchelor. 2008, An Upper Level Ontology for Chemistry, Formal Ontology in Information Systems, Proceedings of the Fifth International Conference. FOIS, ISSN 0922-6389.

[5] Daconta, M. C., Obrst, L. J., & Smith, K. T. 2003, The Semantic Web – a Guide to the Future of XML, Web Services, and Knowledge Management, Indianapolis, USA: Wiley & Sons.

[6] Dean. M, G. Schreiber (eds), et al, "OWL Web Ontology Language Reference "http://www.w3.org/TR/2003/WD-owl-ref-20030331/

[7] A. Grigoris, H.Frank van. 2003. Web Ontology Language: OWL. Handbook on Ontologies in Information Systems. Springer-Verlag.

[8] Hong Zhu and Qingning Huo, 2004, Developing A Software Testing Ontology in UML for A Software Growth Environment of Web-Based Applications, available at http://cms.brookes.ac.uk/staff/HongZhu/Publications/SEUMLXML.pdf.

[9] Huan Wang, Xing Jiang, Liang-Tien Chia and Ah-Hwee Tan, 2010. Wikipedia2Onto – Building Concept Ontology Automatically, Experimenting with Web Image Retrieval, Informatica, 34, 297-306.

[10] Ling Jiang, Chengling Zhao, Haimei Wei, 2008. The Development of Ontology-Based Course for Computer Networks, International Conference on Computer Science and Software Engineering, Vol. 5, PP.487-490.

[11] Marko Grobelnik, Janez Brank, Blaž Fortuna and Igor Mozetič, 2008, Contextualizing Ontologies with OntoLight: A Pragmatic Approach, Informatica, Vol. 32, PP 79–84.

[12] Marvin. J.Carr, S. L. Konda, I. Monarch, F. C. Ulrich, C. F. Walker, 1993. Taxonomy-Based Risk Identification, Software Engineering Institute Technical Report, Carnegie Mellon University, Pittsburgh, Pennsylvania.

[13] D. McGuinness and F. van Harmelen, 2003, OWL Web Ontology Language Overview, available at http://www.w3.org/TR/2003/WD-owlfeatures-20030331/

[14] Mizoguchi, R. & Bourdeau, J. 2000. Using ontological engineering to overcome common AI-ED problems. International Journal of Artificial Intelligence in Education, Vol. 11(2), PP 107-121.

[15] P. Patel-Schneider, P. Hayes, I. Horrocks. 2003. OWL Web ontology Language Semantics and Abstract Syntax, available at http://www.w3.org/TR/2003/WD-owlsemantics-20030331/

[16] Pornpit Wongthongtham, Elizabeth Chang, Tharam Dillon, Ian Sommerville, 2009. Development of a Software Engineering Ontology for Multisite Software Development, IEEE Transactions on Knowledge and Data Engineering, Vol. 21, No. 8, PP. 1205-1217.

[17] Rajiv Pandey and Sanjay Dwivedi, Ontology Description using OWL to Support Semantic Web Applications, International Journal of Computer Applications 14(4):30–33, January 2011, Published by Foundation of Computer Science.

[18] C.R.Rene Robin, G.V.Uma, 2010. Ontology Based Semantic Knowledge Representation for Software Risk Management, International Journal of Engineering Science and Technology, Vol. 2, No.10, PP. 5611-5617.

[19] Sergey Sosnovsky, Tatiana Gavrilova, 2005. Development of Educational Ontology for C-Programming, Proceeding of XI-th International Conference on Knowledge - Dialogue - Solution, Volume 1, Pg. 127-131.

[20] M. Smith, C.Welty, D. McGuinness, 2003. OWL Web Ontology Language Guide, available at http://www.w3.org/TR/2003/WD-owl-guide-20030331/.

[21] Sowa, J. F. 2000. Knowledge Representation – Logical, Philosophical, and Computational Foundations, Pacific Grove, CA, USA: Brooks/Cole.

[22] Stamper R., Liu K., Hafkamp M., and Ades Y. 2000. Understanding the role of signs and norms in organisations – a semiotic approach to information systems design, Behaviour & Information Technology (19:1), PP. 15-27.

[23] Y.Wand, V.C.Storey, and R.Weber. 1999. An Ontological Analysis of the Relationship Construct in Conceptual Modeling, ACM Transaction on Database Systems, Vol. 24, No. 4, PP. 495-528.

[24] Yoshihito Takahashi, Tomomi Abiko, Eriko Negishi, Goichi Itabashi, Yasushi Kato, Kaoru Takahashi, Norio Shiratori, 2005. An Ontology-Based e-Learning System for Network Security, 19th International Conference on Advanced Information Networking and Applications, Vol.1, PP.197-202.

APPENDIX - I

Evaluation Sheet

SOFTWARE RISK IDENTIFICATION ONTOLOGY (SRIONTO)

Metrics	Attributes / Criteria	Description	Degree
Syntactic Quality (The way it is written)	Lawfulness	SRMONTO is syntactically correct	Very High High Medium Low
	Richness	Proportion of features in the ontology language (unionOf, intersectionOf)	Very High High Medium Low
Semantic Quality (Meaning of terms in ontology Library)	Interpretability	Meaning of terms are checked with WorkNet	Very High High Medium Low
	Consistency	Terms used are has its own specific meaning	Very High High Medium Low
	Clarity	Context of term should be clear when interpreting a term at the lower level	Very High High Medium Low
Pragmatic Quality (Usefulness for users)	Comprehensiveness	Size of ontology	Very High High Medium Low
	Accuracy	Claim made is true or false determined by domain experts	Very High High Medium Low
	Relevance	It satisfies the user's specific requirements	Very High High Medium Low

Name :
Degree :
Year :

Signature