Incremental and Iterative Agile Methodology (IIAM): Hybrid Approach for Ontology Design towards Semantic Web Based Educational Systems Development

Santhosh John, Nazaraf Shah, and Leonid Smalov

***Abstract*—A number of Semantic Web based system initiatives have been emerging in various fields such as e-governance, healthcare and e-Learning. These initiatives were aimed at incorporating Semantic Web resources for greater adaptability, robustness and seamless services with easy integration of interoperable systems. Ontologies play a vital role in the realization of Semantic Web vision. Semantic Web initiatives based on ontology have emerged as a promising solution to most engineering problems which occur in these fields. However, current efforts in Semantic Web based Educational Systems (SWBES) are focusing mainly on the combination of two popular modalities of learning: web-based educational systems such as e-Learning Management Systems (e-LMS) and Artificial Intelligence in Education Systems (AIED). This research focuses on an ontology driven approach for a SWBES. An Incremental and Iterative Agile Methodology (IIAM) derived from matured software engineering process models have been used for the ontology development. This paper explains the philosophical and engineering aspects of the newly derived methodology. The stages and framework of the methodology have been applied to develop an educational ontology. Ontology organizes the learning hierarchy of Java Programming Language *Protégé* has been used as an ontology editing tool and an appropriate descriptive language, Web Ontology Language (OWL), was used for the customization.**

***Index Terms*—Java programming, methodology, ontology development, semantic web.**

I. Introduction

The last few years have witnessed a paradigm shift in the World Wide Web, from a global information space of connected documents to a Semantic Web where a formal and sharable Knowledge Based system is key. Semantic Web is a mechanism for presenting information over the web in a format that human beings as well as machines can understand. It is a mesh of information which can be linked in a way that can easily be processed by machines and which can produce resources processed by machines [1]. It aims to produce technologies capable of performing reasoning on semi-structured information [2]. Traditionally, data published on the web has been made available in specific formats compromising much of its structure and semantics. In recent years the Web has evolved to one where both data and documents are linked using the structural model of Semantic Web. Semantic Web technologies based on ontologies have emerged as an appropriate engineering solution to the problems of developing systems which ensure the integration and interoperation of data from different sources to provide seamless services to web users.

Formal and sharable Knowledge Based system is the key aspect of Semantic Web. Ontologies in particular fulfill the requirements for knowledge representation for the Semantic Web. The concept of ontology is originally taken from philosophy where it means an explicit specification of a conceptualization. In recent years, however, this concept has been introduced and used in different contexts, even playing a predominant role in knowledge engineering [3]. Mizoguchi summarized the merits of ontology as follows: ‘‘Ontology provides a common vocabulary, and an explication of what has been often left implicit’’. According to Mizoguchi, the linked data, systematization of knowledge and standardization constitute the backbone of knowledge within a knowledge base system. The main idea behind the Semantic Web vision is to provide an opportunity to represent information on the Web in a way that software agents and systems can understand and manipulate and provide environments which are more adaptable, personalized and intelligent [1]. Ontology is the main building block of Semantic Web vision as it facilitates provision of information in machine processable semantic models and produces semantically modeled knowledge representation systems. The visual representation of the generic concepts of a domain best facilitates both syntactic and semantic knowledge [4]. Linked Data in its simplest form is a set of best practices for publishing and connecting structured data on the web. It refers to machine-readable data published on the web in such a way that its meaning is explicitly defined and which can in turn be linked to external data sets [5]. Since ontology is an explicit conceptualization of a domain, it provides the information in machine processable semantic models and produces semantically modeled knowledge representation systems.

Ontology is an explicit specification to interpret the common meanings of the key terms of a domain where conceptual information is spread across knowledge bases on the Web. Ontologies have proved to be beneficial to describe concepts unlike Relational Database Systems. Ontology-based approaches have been proposed by many researchers and attempts have been made by domain and knowledge experts to implement ontologies in the domain of

Manuscript received September 20, 2015; revised November 27, 2015.

Santhosh John is with Middle East College, Oman (e-mail: santhosh@mecit.edu.om).

Nazaraf Shah is with the Department of Computing, Coventry University, UK (e-mail: aa0699@coventry.ac.uk).

Leonid Smalov is with Creative Computing in Department of Computing, Coventry University, UK (e-mail: csx211@coventry.ac.uk).

doi: 10.18178/ijke.2016.2.1.044

Education. Ontology- based Semantic Web-based Educational Systems (SWBES) can add much value to adaptable e-learning environments. The development of a SWBES however is a rather complex and time consuming task which faces challenges in terms of software engineering. One of the reasons behind this complexity is ontology development. Ontology engineering is an emerging field in computer science, which deals with the methods, methodologies and tools for building and managing ontologies. This branch of engineering aims at making explicit knowledge contained within software applications, enterprises and business procedures for a particular domain. Ontology engineering offers a way out of inter-operability problems brought about by semantic obstacles [6]. Any attempt to leverage the matured software engineering process model will bridge the gap between these two complementing engineering branches.

The main purpose of this paper is to explicate the derivation of a hybrid methodology for ontology design for the development of Ontology driven Semantic Web Systems for the education domain. In Section II, an overview of similar and related work is presented to specify the background of the work. The engineering aspects, different stages and framework of the proposed methodology are discussed in Section III. Section IV discusses the application of the proposed methodology for the development of a prototype of an educational ontology for teaching and learning Java programming. Concluding comments and recommendations for further research are given in Section V.

II. Related Work and Background of Research

Ontology engineering is a challenging and dynamic research field, which deals with the methods, methodologies and tools for building and managing ontology. This branch of engineering aims at making explicit knowledge contained within software applications, enterprises and business procedures for a particular domain. Many methodologies have been proposed for ontologies by researchers and ontology experts.

The availability of methodologies for ontology development with tool support is still an area which requires considerable research. One of the reasons for this research gap is the unavailability of a standardized methodology unlike software engineering. One way or the other, this limits large scale ontology development though technologies such as Semantic Web and Linked Data.

The rigorous development process for ontology building requires the use of methodologies and platforms more or less equivalent to software development. A methodology with fewer curves for software engineers can definitely make ontology development appropriate for business users. A software engineering approach to Ontology building (UPON) [7] has been proposed based on a rich set of resemblances between software engineering and ontology engineering in terms of stages and phases. UPON methodology focused on exploiting the possibilities of Unified Process (UP) and Unified Modeling Language (UML). UPON is a novel approach for large scale ontology development that recommends an iterative life for ontology development by leveraging the features of UP and UML. However, the lack of an agile methodology feature in UPON and the resulting complexity make it unsuitable for effective ontology development.

Ontology development methodologies mainly prescribe guidelines for the specification, conceptualization, formalization and implementation of ontology [8]. The specification primarily covers the aims and purposes of the intended ontology along with an indication of its intended users. Domain ontology is built in four major phases. In the conceptualization phase, the objects, concepts and entities of a domain can be represented in a graphical form at its simplest level. The formalization phase transforms the domain ontology to semi-formal representation, which can be done in description logic or UML formalisms. The implementation phase formally represents the semi-formal version of domain ontology in one of the Semantic Web Languages such as RDF or Web Ontology Language (OWL) with the support of ontology editing platforms. The four core phases discussed here have been considered as the foundation of the proposed methodology and detailed stages are built sequentially.

Until recently, work on accepted practices in Systems and Software Engineering has appeared somewhat disjoined from the area of formal information representation on the World Wide Web. However, obvious overlaps between both fields are apparent and many now acknowledge the merit of a hybrid approach to systems development, combining Semantic Web technologies and techniques with more established development formalisms and languages like the Unified Modeling Language (UML) [9]. The literature and practice indicate that the maturity of software engineering has been established due to its well-proven methodologies and their hand in hand support with modeling languages like UML. Though ontology is the back bone of Semantic Web vision, ontology development faces many challenges. The authors strongly believe that a standardized methodology with integrated tool support for domain modeling can make a significant difference in bridging the gap between software engineering and ontology engineering. There is no one correct methodology for developing ontology, since there is no one correct way to model a domain [10].

Many approaches have been proposed by researchers for developing ontologies in the educational domain. The development of an educational Ontology for C-programming [11] proposed by Tatiana Gavrilova followed a five-step algorithm for visual ontology design. Visual form influences both analyzing and synthesizing procedures in ontology development process. Ontology for teaching Java programming was developed later, based on the five step algorithm proposed [12]. Glossary development, Laddering, Disintegration, Categorization and Refinement are the five core phases of the methodology proposed. The published work focused on the knowledge structuring for ontology development, which can be applied to teaching systems where the emphasis is on general understanding rather than factual details. A framework, Java Learning Object Ontology (JLOO) [13], was presented and used as a guideline for the development and organization of learning objects in introductory Java courses in an adaptive learning system. The

classification in JLOO was based on the computing curricula CC2001 of the ACM and IEEE/CS. JLOO followed a purpose oriented model for ontology development. The IBM Research group came up with another approach, *Eclipse Modeling Framework-Based Ontology Engineering System* (EODM), by leveraging Model Driven Architecture (MDA) and Ontology Definition Model (ODM), which enables model transformation [14].

III. Proposed Methodology

The philosophy behind the proposed ontology development methodology originated from the resemblances between ontology engineering and software engineering in terms of stages and activities. The primary concern was to bridge the gap between software engineering and ontology engineering by leveraging the well proven methodologies and process models of software engineering to ontology engineering domain. An attempt has been made to do so by deriving a hybrid methodology for ontology development from two well proven software process models which are linear and iterative in nature. The conventional Waterfall Model and Rational Unified Process (RUP) [15] have been chosen for the purpose due to their maturity level. Both philosophical and engineering aspects of the IIAM have been adopted from the standards. The stages of IIAM originate from the lifecycle proposed by the Methodology [16]. The methodology proposed for ontology construction by the Foundation of Intelligent Physical Agents (FIPA) promotes inter-operability across agent-based applications. The engineering behind the Methodology is the intermediate representation in terms of different models such as specification model-semi-formal specification using a set of intermediate representations, conceptual model and a formalized model (e.g., Description Logic Ontology UML Profile) which will be implemented in an ontology implementation language (e.g., Web Ontology Language).

IIAM classifies the core ontology development phases into three: Planning, Development and Deployment. The planning phase is concerned with the specification phase including feasibility analysis of ontology domain that focuses on assessing the scope of the domain, with a clear definition of boundaries. The development phase involves the conceptualization and formalization phases with the main goal of producing a conceptual/reference model followed by its transformation to semi-formal representation. The final phase, Deployment, is the implementation phase where the semi-formal model of ontology is formally represented in one of the Semantic Web Languages. Every phase delivers specific deliverables with the common goal of creating a functional component-based ontology that can be effectively used by its intended users. Finally, the formal ontology must be deployed on the Web using appropriate programming platforms such as Java, Net, PhP etc.

Feasibility Analysis. Apart from studying the environment in which the ontology is to be deployed, the possibilities of integrating the ontology into other systems also have to be reviewed. This stage includes assessing the scope of the project with a clear definition of boundaries. Domain experts will participate in this phase as agile methodologies ensure their involvement across the planning, design and development phases.

Domain Vocabulary Acquisition. The development of an ontology starts from the definition of concepts related to the scope of domain. The acquisition of domain vocabulary is the key step for further definition of ontology. Classes are the focus of most ontology. These classes are derived as a result of domain vocabulary acquisition. Automated tools/mind mapping tools can be used at this stage for large scale ontology development.

Enumeration of Concepts and Properties. This stage leads to the derivation of both Data properties and Object properties. Concepts make a semantic translation from one source into another possible. Properties and attribute values will help to achieve the needed shared vocabulary. Properties and their values are playing vital role in the Individual instances of concepts

Taxonomy Identification. This stage defines the concept hierarchies. A top-down development process starts with the definition of the most general concepts in the domain and subsequent specialization of the concepts. A bottom-up development process starts with the definition of the most specific classes, the leaves of the hierarchy, with the subsequent grouping of these classes into more general concepts. A combination development process is a combination of the top-down and bottom-up approaches:

Adhoc Binary Relationships. This stage establishes the appropriate semantic/structured relationship among the identified class hierarchies. Organizing the classes into a hierarchical taxonomy based on the generalization principle that an instance of a subclass will necessarily be an instance of the superclass. If a class A is a superclass of class B, then every instance of B is also an instance of A.

Describe Concepts attributes and Relationships. This stage describes the internal structure of the concept. Most of the remaining terms excluded from the class list after the domain vocabulary acquisition stage, are likely to be properties of the classes. These properties become slots attached to classes. This stage also includes the relationships between individual members of the class and other items

Add Complex Restrictions and Rules. This stage describes the value type, its allowed values, the number of the values (cardinality), and other features of the values the slot can take. For example, the value of a name slot is one string; that is, name is a slot with value type string. A slot can have multiple values and the values are instances of the class.

Vocabulary Linking with Data. The last step is creating individual instances of classes in the hierarchy. Defining an individual instance of a class requires choosing a class, creating an individual instance of that class, and filling in the slot values. At this stage the vocabulary is absolutely linked with the real data. In the proposed agile methodology, the ontology development stages fit into the traditional linear Waterfall process model as shown in Fig. 1.

An attempt has been made to map the well-known phases of RUP with ontology development workflows. Mapping has been done such that the ontology development activities are spread across the phases with scope for iteration. Expected

deliverables of each stage are kept as the intended criteria for mapping. Fig. 2 represents the mentioned mapping.

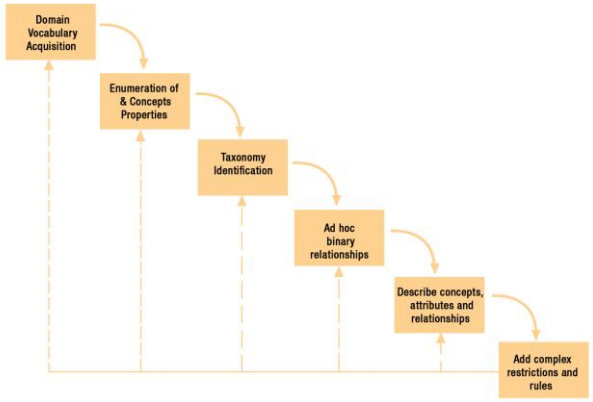

Fig. 1. Linear stages of IIAM.

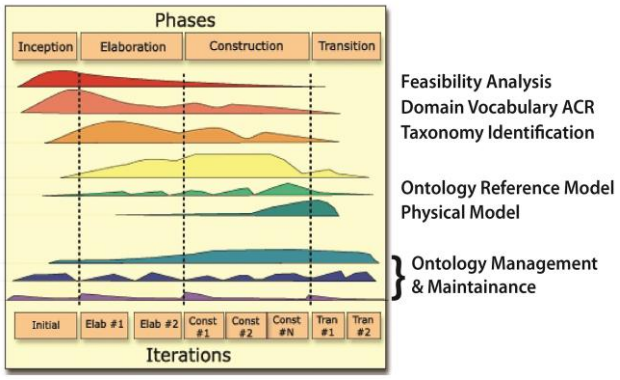

Fig. 2. Mapping with RUP phases and ontology workflows.

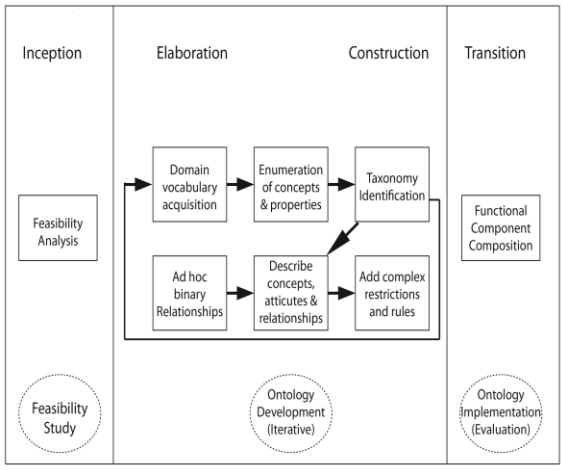

Fig. 3. Framework of proposed methodology.

The final framework of the proposed methodology fits the various ontology development stages into the phases of the incremental and iterative shelf development methodology, RUP. This provides a disciplined approach for assigning tasks and responsibilities within a development team. The Rational Unified Process captures many of the best practices in modern software development in a form that is suitable for ontology development too. The phases of proposed ontology development methodology along with their stages are fitted into RUP phases, Inception, Elaboration, Construction and Transition. The overriding goal of the inception phase is to achieve concurrence among all stake holders on life-cycle objectives for the project and can be mapped to a feasibility study. The purpose of the elaboration phase is to analyze the problem domain, establish a sound architectural foundation, develop the project plan, and eliminate the project's high risk elements. During the construction phase, all components and application features are developed and integrated into the product, and all features are thoroughly tested. These objectives are well mapped to the ontology definition phase. Finally the ontology implementation is mapped towards the transition phase. Fig. 3 illustrates the final framework of the proposed ontology development methodology.

IV. Application of Methodology

The domain chosen for the application of the proposed methodology is taken from education where knowledge sharing and reusability matters a lot. The growth of e-Learning and Semantic Web based educational systems should support the sharing of knowledge in a standard format with common semantics such as a knowledge base. Ontology can be used as a skeletal foundation for such a knowledge base that will allow different applications/software agents to speak in the same language. The proposed methodology is applied to ontology development for teaching and learning Java programming in higher education. One of the motivating factors behind building ontology for Java programming is the attempt to create more effective teaching strategies of an industry language by unifying the different views on the domain. As of now, different teachers introduce Java programming based on many different parameters of their own, such as the order of topics and emphasis on concept. Introduction of ontology ensures uniformity among different views in the domain. This can ensure the basic hierarchical link structure which should not be violated, though the order in which different teachers present the material varies. The different stages of the proposed methodology are applied to the domain chosen and described.

A prototype version of educational ontology was created with *Protégé.* This popular open source editor has been installed and used. The user friendly interface of Protégé is used for the creation of class hierarchy (classes, instances and inheritance structure), slots, domain and range of slots respectively. The model has been derived from the semi-formal UML model generated.

A. Feasibility Analysis

The scope of the ontology focuses on Java 2 Standard Edition (J2SE). It is derived for the domain of Java teaching and learning where different perspectives of Java Programming on higher education have been considered. The feasibility of the ontology was found to be very high as a minimum of seven modules directly dealt with Java programming with 20 different faculty members delivering those modules in the software and computing information system curriculum of Middle East College. A fully developed ontology after evaluation can be integrated with the learning management system, Moodle, as a guideline for different learning paths and making the realization of adaptive learning easy.

B. Domain Vocabulary Acquisition

For the development of ontology, the concepts/classes are

derived from the domain vocabulary acquisition phase. The vocabulary of domain is taken from various resources such as Module Descriptors, surveys and interviews conducted among Java teaching staff and the course materials of the modules: Introduction to Programming, Object Oriented Programming, Distributed programming with Java, Data Structures and Algorithms, Advanced Object Oriented programming and Internet programming. Text books and online resources have been referred for the wide coverage of concepts. Manual and automated mechanism-mind mapping are applied to extract the terms.

The concepts for the first version of ontology used are the core concepts of Java Language, Java OOP, Java GUI, and Java Networking.

C. *Enumeration of Concepts and Properties*

This stage produces an enumeration of concepts and properties. A semantic translation at a very abstract level among concepts is incorporated here. Properties and property values are assigned at this stage to achieve the shared vocabulary. *Protégé 4.3* has been used to model the phase. Fig. 4 shows the general ontology structure based on the concepts derived. The subclass hierarchy with appropriate properties is made for the four core concepts/classes completed during the Taxonomy Identification phase.

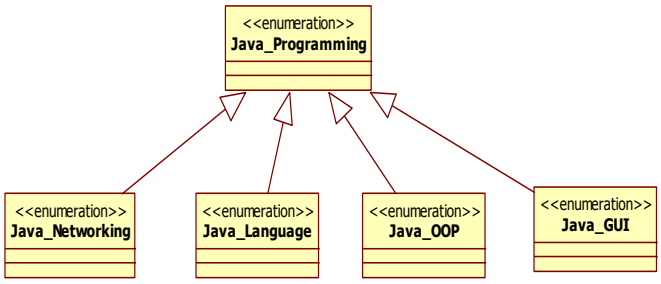

Fig. 4. General structure of ontology.

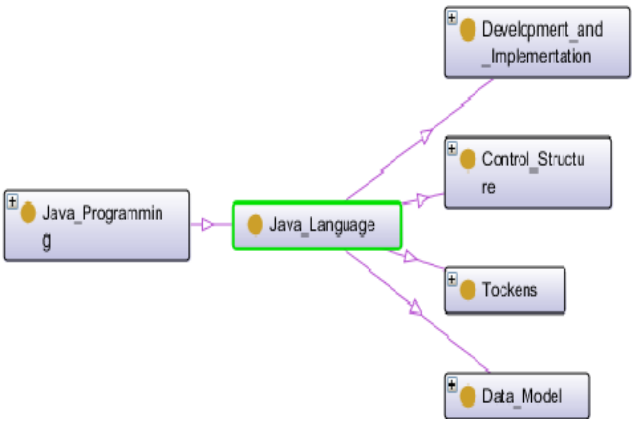

Fig. 5. Abstract level taxonomy of Java language concept.

D. *Taxonomy Identification and Adhoc Binary Relationships*

The core idea behind this stage is concept hierarchy. This stage can be considered as the heart of the proposed methodology. The class hierarchy of the core concepts is the deliverable of the phase. A Top-Down approach has been followed where the process starts with the most general concepts of the domain with subsequent specialization of the concepts. An Abstract level version of taxonomy identification of Java Language concepts is graphically presented in Fig. 5. Hierarchies organize the classes into a hierarchical taxonomy by asking, if by being an instance of one class, the object will necessarily be an instance of some another class. If Class A is a superclass of Class B, then every instance of B is also an instance of A.

Adhoc binary relationships among the concepts belonging to Control Structure Concept are shown in Fig. 6. This has been made for all core concepts identified in the previous stage.

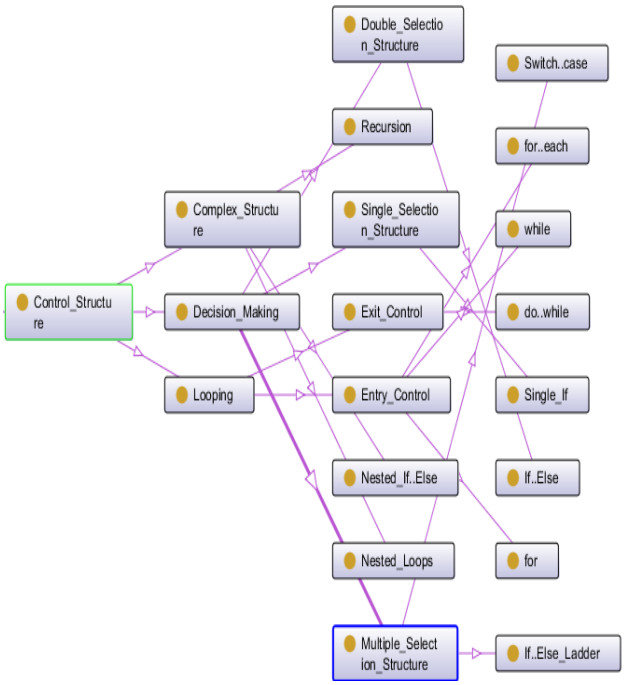

Fig. 6. Binary relationship on control structure concept.

E. *Describe Concept Attributes and Relationships*

This stage describes the internal structure of the concept. These properties become slots attached to classes. This stage also includes the relationships between individual members (instances) of the class. Fig. 7 is a sample screenshot of property values assigned to the concept "**for"** belonging to Entry Control Super Class. Both Object property and Data property are assigned.

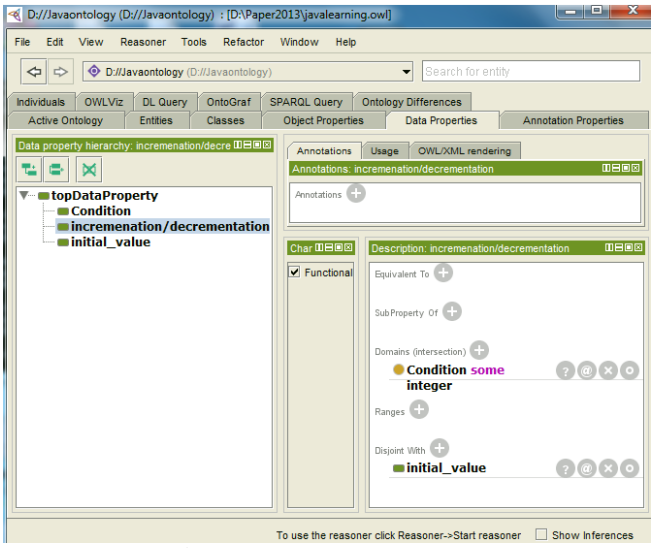

Fig. 7. Screenshot of property values and complex restrictions and rules.

This stage describes the value type, allowed values, the number of the values (cardinality), and other features of the values the slot can take. For example, the value of a name slot is one string. That is, name is a slot with value type string. A slot produced can have multiple values and the values are instances of the class.

F. Vocabulary Linking with Data

This stage creates individual instances of classes in the hierarchy. Defining an individual instance of a class requires choosing a class, creating an individual instance of that class, and filling in the slots' values. At this stage the vocabulary is linked with the real data.

Part of OWL snippets

```
----
  <Declaration>
    <NamedIndividual IRI="#initilization"/>
  </Declaration>
  <EquivalentClasses>
    <Class IRI="#int"/>
    <ObjectSomeValuesFrom>
      <ObjectProperty
abbreviatedIRI="owl:topObjectProperty"/>
      <Class IRI="#byte"/>
    </ObjectSomeValuesFrom>
  </EquivalentClasses>
  <EquivalentClasses>
    <Class IRI="#int"/>
    <ObjectSomeValuesFrom>
      <ObjectProperty
abbreviatedIRI="owl:topObjectProperty"/>
      <Class IRI="#long"/>
    </ObjectSomeValuesFrom>
  </EquivalentClasses>
  <EquivalentClasses>
    <Class IRI="#int"/>
    <ObjectSomeValuesFrom>
      <ObjectProperty
abbreviatedIRI="owl:topObjectProperty"/>
      <Class IRI="#short"/>
    </ObjectSomeValuesFrom>
  </EquivalentClasses>
  <SubClassOf>
    <Class IRI="#Abstract_Classes"/>
    <Class IRI="#Classes"/>
  </SubClassOf>
  <SubClassOf>
    <Class IRI="#Arithmetic_Expresions"/>
    <Class IRI="#Expression"/>
  </SubClassOf>
  <SubClassOf>
    <Class IRI="#Arithmetic_Operator"/>
    <Class IRI="#Operators"/>
  </SubClassOf>
--------
```

V. Conclusions and Future Work

This paper reports the implementation of a new agile hybrid methodology for ontology development derived exclusively from well-known software engineering process models. The proposed methodology is applied to create an educational ontology for teaching and learning Java Programming. Future work should include the validation of this new methodology by evaluation and benchmarking. It can be further enhanced by adding Semantic Web Rule Language (SWRL).The derived ontology produces a solid hierarchical structure of Java topics to be incorporated for a standardized ontology for learning Java programming. This ontology can be integrated with any E-learning system for class room learning purposes as a domain knowledge representation model. The ontology is sharable and reusable for academic institutions and colleges where Java programming is a part of their curriculum. Visualization of Ontology has been made with the support of an ontology editor. Though the general ontology has a good coverage on Java topics, the current version of ontology is limited to Java Language concept.

Future work will complete the comprehensive ontology development of the domain concerned and will integrate with an E-learning system (Moodle). The integration will be experimented in Middle East College for evaluating the viability of the developed ontology.

Acknowledgment

We would like to take this opportunity to express a deep sense of appreciation to Dr. Kiran G.R, Deputy Dean, Middle East College, Sultanate of Oman for his motivation without which this paper would not be possible.

We would also like to take this opportunity to express our gratitude to Dr. Arun N.S, Head of Strategic Initiatives, Department of Middle East College, Sultanate of Oman for his continuous support and guidance.

References

[1] L. L. Bittencourt, E. Costa, M. Silva, and E. Soares, "A computational model for developing semantic web-based educational systems," *Knowledge-Based Systems*, vol. 22, no. 4, pp. 302–315, 2009.

[2] Z. Ahmed. (August 1992). Web to semantic web and role of ontology in its development [Online]. Available: http://www.semantic-web-journal.net/sites/default/files/swj99.pdf

[3] B. Chandrasekaran, J. R. Josephson, and R. Benjamins, "what are ontology, and why do we need them?" *IEEE Trans Intelligent Systems*, vol. 14, no. 2, pp. 20-26, 1999.

[4] D. Fensel, *Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce*, Springer, 2001.

[5] C. Bizer, T. Heath, and T. Berners-Lee, "Linked data-the story so far," *International Journal on Semantic Web and Information systems*, vol. 17, no. 3, pp. 30-56, 2003

[6] Semantic web Activity. [Online]. Available: http://www.w3.org/2001/sw/

[7] A. D. Nichola, M. Missikoff, and R. Navigli, "A software engineering approach to ontology building," *Science Direct Trans. Information Systems*, vol. 34, pp. 258-275, Jan. 2009.

[8] F. Ruiz, C. Calero, and M. Piattini, *Ontologies for Software Engineering and Software Technology*, Springer, 2006.

[9] P. Tetlow, J. Z. Pan, D. Oberle, E. Wallace, M. Uschold, and E. Kendall, "Ontology driven architectures and potential uses of the semantic web in systems and software engineering," *World Wide Web Consortium, Tech. Rep*., February 2006.

[10] D. Gasevic, D. Djuric, and V. Devedzic, *Model Driven Architecture and Ontology Developmen*t, Springer, July 2006.

[11] S. Sosnovsky and T. Gavrilova, "Development of educational ontology for C-programming," *International Journal Information Theories and Applications*, vol. 13. pp. 303-308, 2006

[12] G. Ganapathi, R. Lourdusamy, and V. Rajaram, "Towards ontology development for teaching programming language," in *Proc. the World Congress on Engineering*, London, UK, 2011.

[13] M. C. Lee, Ye, D.Y. Ze, and T. I. Wang, "Java learning object ontology," *IEEE International Conference on Advanced Learning Technologies*, pp. 538-542, 2005.

[14] An MDA-based software development environment for ontology engineering. (2005). [Online]. Avialble:

http://domino.research.ibm.com/library/cyberdig.nsf/papers/0D884EF43842DFD8852570BD0060E974/$File/rc23795.pdf
[15] B. MacIsaac. An overview of the rup as a process engineering platform. (2015). [Online]. Avialble: http://www.researchgate.net/publication/265917353
[16] A. Gómez-Pérez, M. Fernández-López, and O. Corcho, *Ontological Engineering*, Berlin: Springer, 2004.

Santhosh John completed his masters from University of Madras, India (1998) in computer applications and is currently pursuing his PhD in computer science. He has published over 10 refereed publications in books, journals and conference proceedings. He has over sixteen years of experience as a faculty in computer science in premier IT institutions of India, Malaysia and the Sultanate of Oman. His other accomplishments include being a grant recipient of Microsoft Knowledge Capital Centre-Malaysia (2003). His research interests are ontology development for education, semantic web systems for education, software engineering for education and cross platform mobile applications development.

Nazaraf Shah is a senior lecturer in the Department of Computing at Coventry University, UK. His research interests include intelligent agents, service-oriented computing, cloud computing and big data and Dynamic Scheduling. He published over 50 refereed publications in books, journals and conference proceedings. Dr. Shah is a member of editorial broad for a number of international journals. Dr. Shah has been involved a number of EU funded projects such as MOSIACA, DEHEMS and GREENet. Dr. Shah served as a number of international conferences by taking different responsibilities such publication chair, special issue chair and workshop chair and served as guest editor and lead guest editor of two international journals.