# General Domain Ontology in Enterprise Software Development Process

**Marzanah A. Jabar, Mustafa S.Khalefa**

***Abstract:*** ***In software development, it is usually difficult to find the proper software component and customize or integrate it into the body of software. To reduce time and costs of errors produced in development processes, a systematically enriched representation of components is necessary. The main objective of this paper is to propose a general concept of domain ontology to express formally a shared understanding of information; we argue that they can be used to improve knowledge flow in enterprise software development process (SDP). In order to achieve this, we collect and analyse the concepts that have been used in knowledge flow ontology during enterprise software development process. We justify the need of the domain ontology in enterprise software development process. We further implemented the knowledge flow framework in domain ontology and validate it using a model of knowledge flow in software development. We performed a five-step procedure to measure the quality of the proposed knowledge flow framework model indicating that our proposed ontology for knowledge flow improves effectiveness of knowledge acquisition and sharing in software development organizations.***

***Keywords—*** ***Ontology, knowledge flow, software development process.***

## I. INTRODUCTION

Ontology is an accepted technique for capturing and conceptualizing the knowledge of an enterprise (Á. E. Prieto and A. Lozano-Tello. 2009). (Linda Anticoli and Elio Toppano. 2011, Pisanelli, D.Met all. 2002) , defined ontology as a specific and a clear picture of an abstract world to be presented in a clear simplified view for some purpose whereby vocabularies are extracted from a domain. Having a clear picture may lead to the concepts and their relations to be extracted from the real world (Gruber, T. 2003). Since ontologies use similar shared vocabularies when describing ideas and relations they can also be used as tools in specifying the meaning of terminology systems in a clear defined manner. The authors in (Guarino, N. 2009) identified three major significant uses of ontologies which are, (i) to help in communication between human beings, (ii) to make communication among software systems possible, and (iii) to improve the Enterprise system (ES) design and quality of software development process. The new domain should be designed using the following procedures as identified by (Y. Kalfoglou and M. Schorlemmer., 2002) for the purpose, to capture, perform the coding, integrate with any existing ontology, perform evaluation and complete the documentation of the ontology. There is a convention in the ontology group that the incorporation of current ontologies is a more valuable way to make a new ontology and thus cut down on the time, price, and attempt done by the work of (Linda Anticoli and Elio Toppano 2011). Ensuring semantic interoperability among different ontologies for incorporation is a key aspect for building ontologies effectively and guaranteeing semantic interoperability across the domain (Jeongsoo Lee et all 2006).

In software development, a systematically representation of knowledge related to the activities requires extensive knowledge related to the process. An ontology-based approach can facilitate those needs by providing a systematic and structuralized description of knowledge in the development process. We use a combination of Top-Down and Bottom-Up approaches to develop the ontology. In Top-down approach, development starts with finding the general concept class and we perform subsequent specialization of those concepts..

## II. RESEARCH BACKGROUND

The researchers in (Ricardo de Almeida and Giancart Guizzardi. 2002) addressed that the engineering domain analysis activities are distributed across processes. They tested the Ontology Based Domain Engineering (ODE) on the proposed ontology-based approach to domain engineering that considers two main phases: building ontologies and deriving object frameworks from them. However, the technique does not provide explicit concept, which is a common problem in engineering domain. In (Liana Razmerita et all 2003), they identified the aspects of user modeling relevant to Knowledge Management System (KMS) and integrated them in a generic framework based on ontologies. They showed a good example of the use of the Ontology Based user Modeling (OntobUM) framework in an ontology-based KMS. On the otherhand, this framework can be adapted to different ontology-aware environments, therefore, the user ontology is implemented using semantic web technology and it is structured on extended Information Management Systems Learner Information Package Specifications (IMS LIP). Besides that, the most significant part of the user ontology is common and it could be reused in another application domains. The drawback of the presented work is a slow search process when large database applied for KMS.

**Manuscript published on 28 February 2019.**
* Correspondence Author (s)

**Marzanah A. Jabar,** Department Of Software Engineering & Information System Faculty Of Computer Science &Information Technology, 43400 Selangor, Serdang, Malaysia.(marzanah@upm.edu.my)
**Mustafa S.Khalefa,** Education College for Pure Science, Basra University, Iraq.(mustafakhalefa@gmail.com)

In (Pornpit Wongthongtham et all 2004) issues raised and discussed are about how people cannot share their knowledge when they do not speak a common language. They presented an ontology model of software engineering to solve this problem by using special ontology characteristic for software engineering and model it using graphical notion unified modeling language (UML) to share the knowledge. Work of (Polala Niranjan and Kukatlapalli Pradeep. 2010), discussed on the problem of how to communicate with team from different site location, and how to solve this problem using ontology shared system. The development of their project as windows application was tested by using visual studio framework with C# program language and they used a practical case study for their implementation. however they did not cover all aspect of software engineering such as testing and quality and new approaches affecting every single aspect in software engineering. In this paper we examine the role of ontology based SDP to investigates the possible integration of ontology, as well as to develop the taxonomy as framework for understanding Ontology Based Knowledge Management for Enterprise system (OKES). The motivatation of this paper will be to investigate in depth the OKES in order to present understanding of software engineering project information. We will conceptualize ontology model as instance knowledge, in order to derive the software engineering knowledge ontology.

## III. METHODOLOGY

The ontology-based approach provides a systematic and structuralized description of knowledge in the development process. We adapt the work in by (A. Maedche and S. Staab. 2001) to define an ontology as O=(l,f,g,c,root,h,s) which consists of lexicon, reference function, concepts, taxonomy, template slots. In other definition, ontology is defined as O=(c,r) where c is a set of ontology concepts and r is relationship between knowledge concepts (G. Zhang, et all. 2010). C is a class which includes class concept Cc, attribute concept Cp, hence C can be shown as a set of C=<Cc, Cp>. R is a finite set of assertions that can be expressed as r={r┤|r ∈C}. After defining classes (c) and class hierarchy (r), we use a combination of Top-Down and Bottom-Up approaches to develop the ontology in (A. Heredia, et all, 2013). In Top-down approach, development we starts by finding the general concept class and we perform subsequent specialization of those concepts. For example, we start building the ontology with two general concepts of "implementation" and "business modeling". We specialize the implementation class by generating appropriate subclasses such as: source code, log history and error tracking. This relationship can be expressed as S={C_x |C_x is an argument of C}.

In Bottom-Up development we start with more specific concepts and grows upward to the most general concepts that are super classes of concepts. For example, we start with subclasses of "Java" and "Git" and "Trac" and then we generalize to super classes of "source code", "log history" and "error tracking", respectively.

This relationship can be demonstrated as G={C_x |C_x is a child of C} (M.cSensoy and P. Yolum. 2009). Presenting knowledge concepts based on their concept classes, helps us to build the knowledge flow framework. This happens by implementing two methods of specialization and generalization of knowledge concepts.

Figure 1 shows the domain ontology of an enterprise software development processes. It starts with enterprise software development processes as the basis and we expand to general knowledge concepts of specifications, Design and implementation, Integration and testing, Operation and maintenance. Then we use specialization to identify subclass concepts. For example, a super class of "Specifications" can be expanded to subclasses of "Enterprise business modeling", "Portfolio management", "People management", "Enterprise administration", "Business analysis" and finally "Requirement analysis".

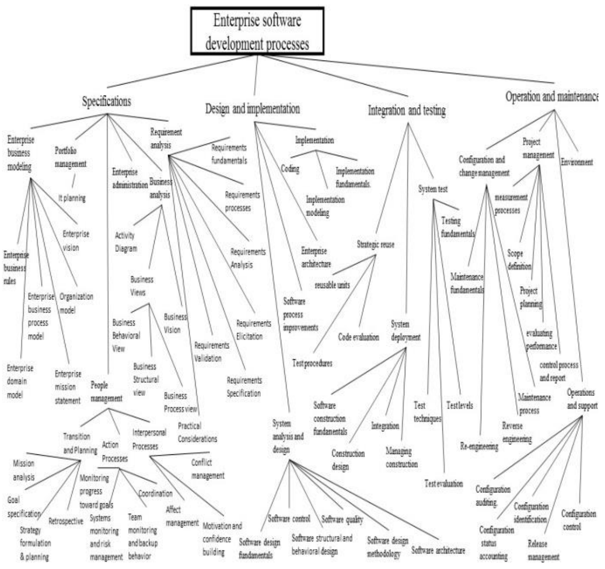

**Figure 1. Domain Ontology Of ESD**

Similarly, the subclass of Enterprise business modeling can be extended to "Enterprise business rules", "Enterprise domain model", "Enterprise business process model", "Enterprise mission statement", "Organization model", and "Enterprise vision" in (H.-E. Eriksson and M. Penker. 2000). In other words, S Enterprise business modeling={ Enterprise business rules, Enterprise domain model, Enterprise business process model, Enterprise mission statement, Organization model, Enterprise vision } and G Enterprise business modeling=(Specifications). In the domain ontology (Y. Ding and S. Foo. 2002), a subclass of "specification" named people management can be expanded to transition and planning, action processes and interpersonal processes which can be expanded further to S Transition and Planning ={Mission analysis, Goal specification, Strategy formulation & planning, Retrospective}, S Action Processes ={Monitoring progress toward goals, Systems monitoring and risk Management, Coordination, Team monitoring and backup behavior}, and S Interpersonal Processes ={Conflict management, Motivation and confidence building, Affect management} (S. Bourdeau and H. Barki. 2013).

The work (H.-E. Eriksson and M. Penker. 2000), suggested that Gbusiness analysis=(Specification) can be specialized to Sbusiness analysis={Activity diagram, Business view, Business vision}. Similarly the rest of relationships have been gathered from (A. Abran, et all 2001

Using the provided domain ontology of enterprise software development processes, we are able to infer knowledge requirements of each knowledge concept at any level. For example, project managers have prior knowledge about operation and management and the involved process in a project or manage project planning activities. In this case, knowledge requirements can be presented by enterprise employee in configuration, project management or environment management.

## IV. DOMAIN ONTOLOGY FOR SDP

Domain ontology term can be explained in many different ways such as a categorical specification of a group of entities that are objects, concepts and relationship (Jain, V., and Singh, M., 2013). In the field of software development, it provides a common ground for software developers to understand the concepts as well as the systems to interpret definitions of entities. Thus in general every ontology contains important concepts in a specific domain along with relationships and constraints of each concept Protégé in used as the Ontology development tool as it is customizable and effective to create ontologies using an extensible knowledge model. It has an extensible architecture that enables integration with other applications such as graphical representation of domain ontology. Figure 2 shows a view of our proposed domain ontology of enterprise software development processes.

Some of the most popular ontology languages are DARPA Agent Markup Language (DAML) which is based on the semantic web and incorporates XML markups that enhances interoperability of the language Hendler, J., & McGuinness, D. L. (2000). Semantic web rule language syntax SWRL which includes semantics and abstract syntax (Horrocks, I., et all, 2004), and Web Ontology Language OWL which presents entities and their relationship using formal XML vocabulary (McGuinness, D. L., and Van Harmelen, F., 2004).

Protégé is a tool use in our study for the creation and modification of domain ontologies (Jain, V., and Singh, M., 2013). The following figures described the ontology implementation of one part of activity in a software development process selected from Figure 2, as an example of how we implement ontoly to describe the knowledge flow among the people responsible in this activity. Figure 3, depicted the Design and Implementation classes, and it's relationship described using protégé tools in SDP domain ontology. Figure 4 shows the roles of individual involved during the Design and Implementation phase that includes developer, analyst, designer and project manager. While Figure 5 shows the individuals relation between the roles during the design and implementation phase in enterprise domain ontology using protégé tools.

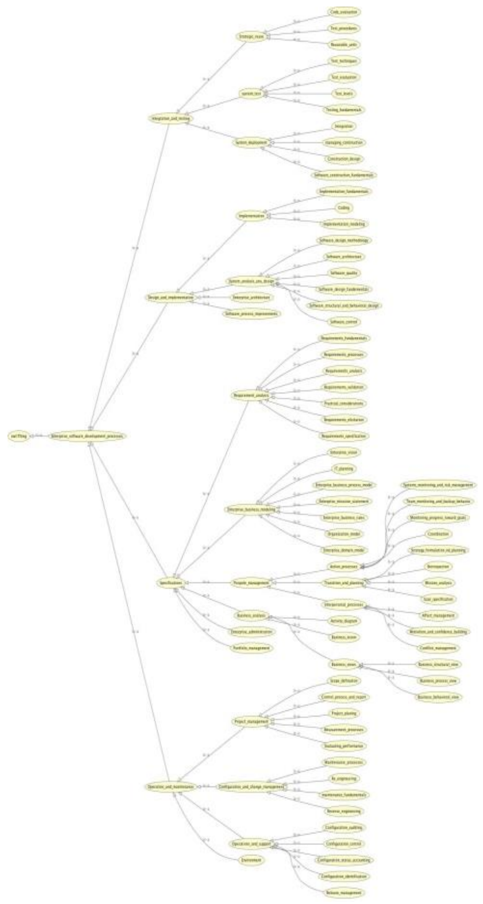

Figure 2. Model View of Complete Domain Ontology

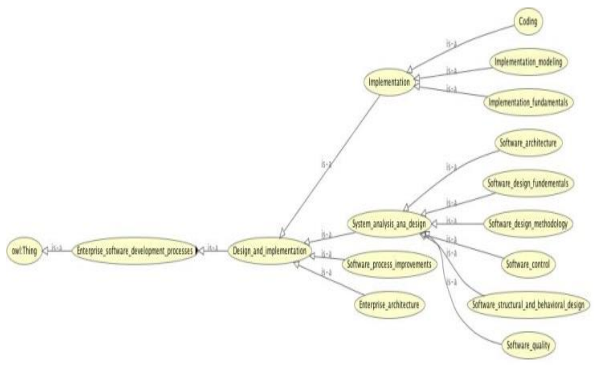
owl:Thing
is-a
Enterprise_software_development_processes
Design_and_implementation
Implementation
Coding
Implementation_modeling
Implementation_fundamentals
System_analysis_ana_design
Software_architecture
Software_design_fundementals
Software_design_methodology
Software_control
Software_structural_and_behavioral_design
Software_quality
Software_process_improvements
Enterprise_architecture

Figure 3.Design And Implementation Classes

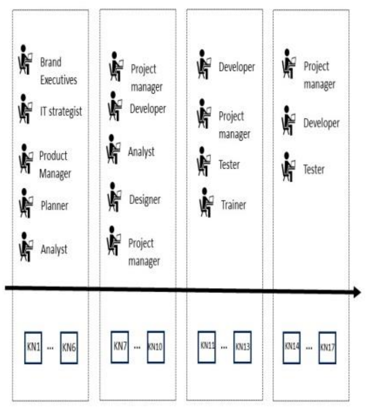

Figure 4: Design And Implementation Role

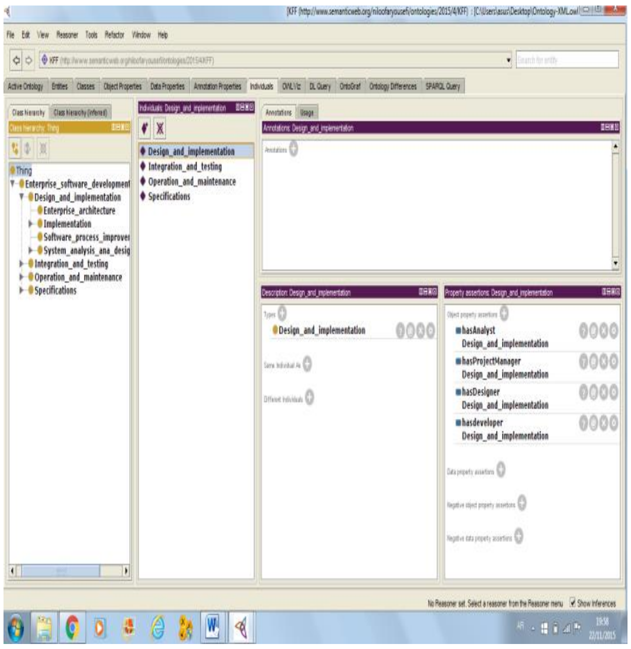
Thing
Enterprise_software_development
Design_and_implementation
Enterprise_architecture
Implementation
Software_process_improvement
System_analysis_and_design
Integration_and_testing
Operation_and_maintenance
Specifications
hasAnalyst Design_and_implementation
hasProjectManager Design_and_implementation
hasDesigner Design_and_implementation
hasdeveloper Design_and_implementation

**Figure 5. Design And Implementation Roles In Individual Relation**

## V. RESULTS AND DISCUSSION

In assessing the ontology, we propose an ontology based knowledge flow model in SDP and proved the construct model is valid. In our evaluation we proposed 17 independent variables and 4 latent variables derive from Figure 2, according to the abstraction of knowledge flow for software development domain ontology. The variables have been measured using a five point Likert scale ranging from 1 representing "strongly disagree" to 5 indicating "Strongly agree", as described. We have taken a five step regression analysis method to examine the goodness of fit for the model. This method consists of assessment of R-squared value (R2), path coefficient values, F-squared values (F2), predictive relevance (Q2), and finally goodness of fit. In the second step, we performed path coefficient value assessment. The relationship between latent variables of "Specification" and "Design and implementation" was moderately strong with path coefficient value of 0.688. Next we measure the effect size of f2 variables in the third step. In the fourth step, we performed predictive relevance model using Partial Least Square (PLS) to show relevancy of the proposed model constructs. Finally, we calculated goodness of fit for the proposed model. In conclusion, all five step of structural assessment of model resulted positive that can be concluded as substantially reliable and adequate of the ontology based model for knowledge flow in SDP.

## VI. CONCLUSION

Software development teams require knowledge to support their tasks and task-related performance, especially in enterprise software development. A knowledge flow framework is a structure to identify, share and document task related knowledge of development teams thus providing a basis for knowledge support tool that benefits team members. The purpose of knowledge flow model is to represent direction of team member's knowledge requirements and referencing flow of documents that are necessary to perform task related activities. Hence an effective ontology knowledge flow framework supports the knowledge requirements in software development processes. The proposed model is based on the proposed software development domain ontology that covers different aspects of enterprise software development. We performed a five-step procedure to measure the quality of the proposed knowledge flow framework model indicating that our proposed ontology for knowledge flow improves effectiveness of knowledge acquisition and sharing in software development organizations.

## VII. ACKNOWLEDGEMENT

The authors would like to express gratitude for the financial support provided under the Fundamental Research Grant Scheme (FRGS) Grant Cost Centre : 5524961.

## REFERENCES

1. A. Abran, P. Bourque, R. Dupuis, and J. W. Moore. 2001. Guide to the software engineering body of knowledge-SWEBOK. IEEE Press.
2. Á. E. Prieto and A. Lozano-Tello. 2009. "Use of ontologies as representation support of workflows oriented to administrative management," J. Netw. Syst. Manag., vol. 17, no. 3, pp. 309–325.
3. A. Heredia, J. Garcia-Guzman, A. Amescua, and M.-I. Sanchez-Segura, 2013. "Interactive knowledge asset management: Acquiring and disseminating tacit knowledge," J. Inf. Sci. Eng., vol. 29, no. 1, pp. 133–147,
4. A. Maedche and S. Staab. 2001. Comparing ontologies-similarity measures and a comparison study. AIFB.
5. G. Zhang, S. Jia, and Q. Wang. 2010. "Construct ontology-based enterprise information metadata framework," J. Softw., vol. 5, no. 3, pp. 312–319.
6. Gruber, T. 2003. "A Translation Approach to Portable Ontology Specifications", Knowledge Acquisition. Volume 58, Issue 1, January, Pages 89–123.
7. Guarino, N. 2009."Formal Ontology in Information Systems", Proceedings of FOIS'98, Formal Ontology in Information Systems, Handbook on Ontologies, Information Systems, Springer-Verlag Berlin Heidelberg Page 2. 202 N. Guarino and CA Welty.
8. H.-E. Eriksson and M. Penker. 2000. "Business modeling with UML," Bus. Patterns Work. John Wiley Sons, New York, USA.
9. Hendler, J., & McGuinness, D. L. (2000). The DARPA agent markup language. IEEE Intelligent systems, 15(6), 67-73.
10. Horrocks, I., Patel-Schneider, P. F., Boley, H., Tabet, S., Grosof, B., & Dean, M. (2004). SWRL: A semantic web rule language combining OWL and RuleML. W3C Member submission, 21, 79.
11. Jain, V., and Singh, M. (2013). Ontology development and query retrieval using protégé tool. International Journal of Intelligent Systems and Applications (IJISA), 5(9), 67
12. Jeongsoo Lee, Heekwon Chae, Kwangsoo Kim, Cheol-Han Kim. 2006. "An Ontology Architecture for Integration of Ontologies" The Semantic Web – ASWC 2006 Lecture Notes in Computer Science Volume 4185, , pp 205-211.
13. Liana Razmerita, Albert Angehrn ,and Alexander Maedche.2003. "ontology based user modeling for knowledge management system" INSEAD,CALT-Centre of Advanced Learning Technologies, 77300 Fontainebleau, France.

14. Linda Anticoli and Elio Toppano. 2011. "The Role of Culture" in Collaborative Ontology Design International Conference, - dl.acm.org.
15. McGuinness, D. L., & Van Harmelen, F. (2004). OWL web ontology language overview. W3C recommendation, 10(10), 2004.
16. Polala Niranjan, Kukatlapalli Pradeep. 2010. "An Efficient software engineering ontology tool for knowledge sharing.
17. Pornpit wongthongtham , Elizabeth chane, tharam Dillon ,Ian sommerville. 2004. "Development of software engineering ontology for multi-site software development.
18. Pisanelli, D.M., Gangemi, A. And Steve, G. 2002. "Ontologies and Information Systems: The Marriage of the Century", Proceedings of the LYEE Workshop, Paris.
19. Ricardo de Almeida, Giancart Guizzardi. 2002. "An ontology approach to domain engineering " Computer Science Department, Federal University of Espírito Santo , Fernando Ferrari Avenue, CEP 29.060-900.
20. S. Bourdeau and H. Barki. 2013 "Toward a Typological
21. Y. Ding and S. Foo. 2002. "Ontology research and development. Part 1-a review of ontology generation," J. Inf. Sci., vol. 28, no. 2, pp. 123–136.
22. Y. Kalfoglou and M. Schorlemmer. , 2002 "Information-flow-based ontology mapping". In On the Move to Meaningful Internet Systems 2002: CoopIS, DOA, and ODBASE, volume 2519 of Lecture Notes in Computer Science, pages 1132–1151. Springer.