International Journal of Computer Science Engineering
and Information Technology Research (IJCSEITR)
ISSN (P): 2249-6831; ISSN (E): 2249-7943
Vol. 8, Issue 2, Jun 2018, 31-36


TRANS STELLAR
•Journal Publications •Research Consultancy

# MULTISITE SOFTWARE DEVELOPMENT WITH ONTOLOGY: A REVIEW

**HEMANT H. PATEL[1] & NITESH M. SUREJA[2]**

[1]*Research Scholar, Rai University, Ahmedabad, Guarat, India*

[2]*Director, Om Engineering College, Junagadh, Gujarat, India*

***ABSTRACT***

***Ontology is an important concept of software engineering. It formally expresses knowledge in a way that software can handle knowledge and effectiveness. The software engineering ontology supports the definition of information exchange semantic project the information framework. In this review, we analyze what the software engineering ontology is, its issues and problems, and how to interpret them. The usage scenario used in this review emphasizes the characteristics of software engineering ontology. The software engineering ontology helps define the information for exchanging semantic item information and serves as a communication framework. The end user in a software engineer who shares domain knowledge and knowledge of software engineering examples.***

***KEYWORDS: Software Engineering, Ontology Lifecycle for Software Engineering, Ontology Development, Issue and Challenges of Multisite Software & Multisite Software Development.***

**Received:** Mar 31, 2018; **Accepted:** Apr 20, 2018; **Published:** May08, 2018; **Paper Id.:** IJCSEITRJUN20184

Original Article

## INTRODUCTION

With the advent of the Internet, software development has increasingly focused on the Internet, enabling a multisite environment that allows multiple teams across cities, regions, or countries to develop software distributed throughout the network. The benefits of multisite software development have led large companies to shift software development to countries with relatively low wages. However, distributed settings can cause problems such as communication difficulties, coordination barriers, and language and cultural differences [1].

In the traditional software development model, it is difficult to have people with relevant skills in all these areas of software development [2]. Usually, in order to successfully provide large and complex computer-based information systems and reduce software development costs, development outsourcing and acquire the necessary skills. In other words, in order to achieve a common comprehensive development goal, professional groups need to cooperate in remote areas [3,4].

The following examples demonstrate the need for an MSSD methodology:

- In outsourcing, analysts who create problem definitions, system specifications, or IT solutions may be geographically or internationally far removed from design and programming teams. If the specification is incomplete, fuzzy or evolving, this physical distance can be an important issue.
- If each implementation group is on a different site, different interpretations of component specifications or software requirements may result in incompatibility of components or subsystems.
- Another example is that if a component-based approach is used, components may be developed by multiple

remote teams. Using the same software, but using different terminology for different groups of experts is often difficult and this is overcome without face-to-face communication.

- Another example is that customers and testers may be on different websites with programmers.
- Most existing software process models assume an intensive software development approach. As the example above shows, this assumption is not appropriate in many cases. This assumption also exists in many current software engineering methodologies that do not involve multisite software development. Finally, we can see that many technologies are not mature enough to make multi-point development easier.

This paper is organized as follows like the definition of software Ontology, traditional approach & purpose for ontology, software engineering lifecycle for ontology, issues in global software engineering development, the model of multisite software development and its issues and at the end conclusions are described.

## ONTOLOGY IN SOFTWARE ENGINEERING

- 'Ontology' only provides the standardization and interpretation of machine understandable content. Software engineering is a well-known area, including software development, and it is necessary to be able to record all definitions in a machine interpretable manner.
- The term 'Ontology' is derived from its usage in philosophy where it means the study of being or existence as well as the basic categories [5,6,7,10]. Therefore, in this field, it is used to refer to what exists in a system model.
- An ontology, in the area of computer science, represents the effort to formulate an exhaustive and rigorous conceptual schema within a given domain, typically a hierarchical data structure containing all the relevant elements and their relationships and rules (regulations) within the domain [2,6,7,10].

## TRADITIONAL APPROACH & PURPOSE WHICH ONTOLOGY ARE BEING USED

Traditional software process models does not address the multisite, multi-team situation and the needs of its environment [2]. Most of the examples of existing software process models are based on a strengthened approach to software development. Most project management methods do not consider multisite distributed software development. We have also observed that it is not mature enough to have a lot of technology to promote multisite distributed software development. Process models, methods, techniques, methods, as already suggested in multisite distributed software development, you can't follow linearly [2, 6].

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

- Reference media providing powerful and clear communication mechanisms and software engineers working on multiple sites develop software.
- Provide heterogeneous data and information sources, especially mediation mechanisms for accessing the Web.
- Provide a common knowledge base for multiple agents working in a specific domain.
- Specifically, in order to provide clear semantics and reliability for interactions on the Web, we have built a trust and reputation system based on privacy [2,6].

## ONTOLOGY IN THE SOFTWARE ENGINEERING LIFECYCLE

- Here we discuss a method of using ontology in software engineering. We discussion the order in which the software engineering life cycle occurs. Is in focus is on the general issues that each method needs to solve.
- Analysis and Design can benefit from ontology from the perspective of knowledge representation and process support. Second, component reuse is selected as a potential application area at design time. In most cases, natural language is used to describe requirements, Like in the form of use cases. The KOntoR system is uses knowledge base approach to solve the problem of component reuse [5].
- The key step in the development process is moving from analysis and design to implementation. Here we discuss about Integration with Software Modelling Languages and Ontology as Domain Object Model. We will also discuss the technology of ontologies that support encoding and code documentation. UML is used for an ontology representation language. Ontology API and Tools are already available to crack the domain object model. The Smart API approach and "Software Information Systems" (SIS) is use Coding Support and Documentation [5].
- Ontology can support Deployment and Run-time with the Semantic Middleware and Business Rules. Component-based and service-oriented ontology is used for the Semantic Middleware. The commercial rules engine provide “ontology “and “knowledge” architecture to deliver a middleware for the Business Rules. Several standards and frameworks like OWL-S or WSMX are currently beneath development for Semantic Web Services [5].
- Maintenance is also a key step in the ontology for project Support, Updating, and Testing. Dhruv is defined a problem-solving processes in web communities for project Support. Dameron describes a framework for automatically updating ontology with the help of Portege editor and its plug-ins. Ontologies could help to generate basic test cases since they encode domain knowledge in a machine processable format [5].

## ISSUES IN EXISTING GLOBAL SOFTWARE ENGINEERING DEVELOPMENT APPROACH

Global software development (GSD) is a sensation of increasing importance, given the constant pressures of the need to remain profitable and competitive in the global landscape. Companies can now leverage the emergence of large multi-skilled labor forces in lower-cost economies thanks to high-speed Internet-based communication links, through which the product (software code) can be quickly transferred between development sites. Computer-Aided Collaborative Work (CSCW) is a well-known method of supporting group design tasks in a manufacturing environment [1].

GSD face some issue like Reduce intensive collaboration, cultural distance and temporal distance [1,13]. Mockus and Weiss [14,15] propose a way to use the code change history to calculate the degree of relevance of work items on two sites with the help of Transfer by functionality, localization, development stage, and maintenance but communication problem between groups and multiple sites cannot solve.

Heeks et al. [15,16] presented three interesting case studies and proposed a successful outsourcing strategy to maximize the value of the team. They consider the relationship with the central office and outsourcing team, developers, and customers. There are not enough details to describe the process, task or action so that the software team can follow these details, strategies, and methods.

## AN ONTOLOGY BASED MODEL FOR MULTISITE SOFTWARE DEVELOPMENT

Figure.1 shows an abstract view of software engineering knowledge. The entire concept of software engineering that represents knowledge in the field of software engineering is incorporated into ontology [10]. Based on a specific problem area, projects or specific software development may use only a portion of the entire series of software projects. Specific software engineering concepts for specific software development projects represent sub-domain knowledge of software engineering and are incorporated into the ontology. General software engineering knowledge represents all concepts of software engineering. Specific software engineering knowledge represents the concept of software engineering in specific problem areas. For example, if the project uses a purely object-oriented approach, the concept of a data flow diagram may not necessarily be included in a particular concept. Instead, it includes concepts such as class diagrams and activity diagrams. Once all domain knowledge, sub-domain knowledge, and instance knowledge are obtained in ontology, it can be shared among software engineers via the Internet. All team members can query semantically linked project information wherever they are and provide common communications for discussion issues, problem analysis, problem analysis, revision proposals, or solution design. It can be used as a knowledge base.

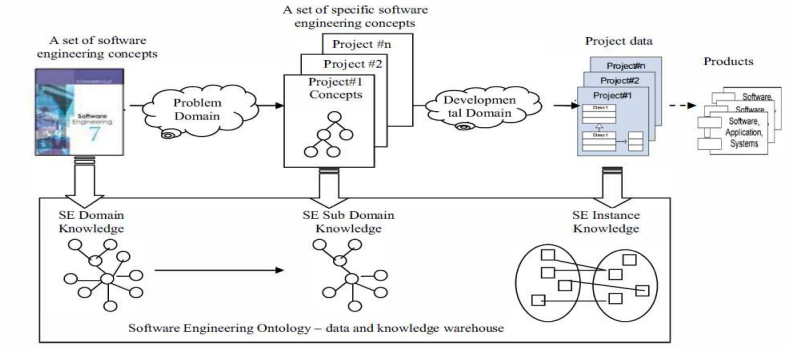


**Figure 1: Multisite Software Development Model [10]**

## DESIGN AND EVALUATION OF ONTOLOGY

The purpose of a given ontology is to agree on common terms and set constraints on the object. We need to agree on the purpose and end use of the ontology. Therefore, we must provide a mechanism to guide the design of the ontology. Such a framework can more accurately evolve ontology proposals by demonstrating the applicability of each proposal to a series of problems that the application generates. Method for designing and evaluating integrated ontology [7]. The proposal including the new ontology and the extension of the existing ontology are shown in Figure.2 below.

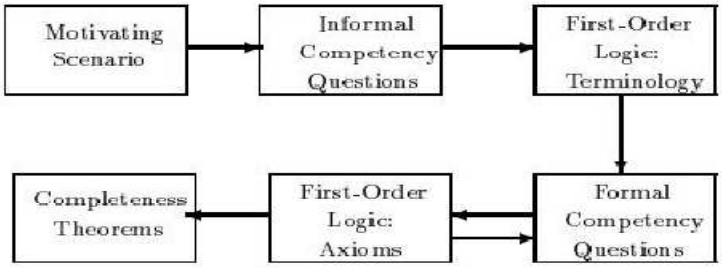


**Figure 2: Design and Evaluation of Ontology [7]**

## KNOWLEDGE SHARING ACTIVITIES IN PROJECT ORGANIZATIONS

The organizational structure is shown in Figure.3. The CEO is responsible for the entire organization made up of multiple departments. The director is responsible for a department composed of multiple departments. The manager is responsible for a multi-person department [9].

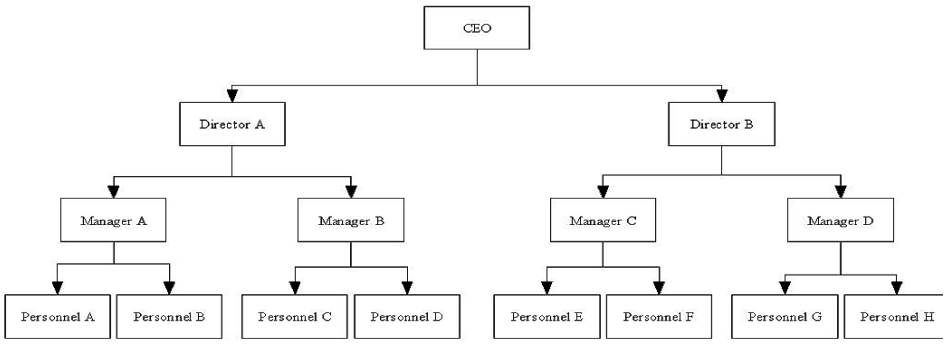


**Figure 3: Knowledge Sharing Activity [9]**

The process of project organization plan can be drawn as like projects are created by one or more department heads. After that, the director handed the draft project to the board of directors consisting of the chief executive officer and the advisor. The Council Board will evaluate the project drafts. If the project drafts are agreed, the projects will be distributed by CEO to the directors who draft the projects.

About the project distribution, normally there are three circumstances which are:

- The CEO assigns a department to a project. The head of the department divides the project into one or more managers. Each manager and related personnel act according to the actual implementation of the task.
- The CEO assigns multiple projects to one department. The director assigns at least one project to each manager. Therefore, each manager is responsible for managing the implementation of the assigned project.
- CEO assigns one project to more than one department, and then the directors of the departments are responsible for the implementation of the project.

Centralized team management is much easier compare to multisite software development because we have the opportunity to face it directly. We can eliminate doubts, confusion or difficulties in time. We have arranged a meeting at any time [8]. This is not the case with multisite distributed team management because of explicit knowledge necessary for multisite software development functions. Deprived Methodology for inter-team workflow planning and management. Cannot define appropriate knowledge sharing management. We cannot discuss any doubts face to face [1].

## CONCLUSIONS

In this review, we discussed multisite software development with its advantages in detail. We have also identified several major issues related to multisite software development. After this, it has been confirmed that there is a need to address the availability of communication and coordination, a unified semantic knowledge sharing and knowledge sharing platform to solve problems related to multisite software development. After that, we showed the requirements of the solution. Ontologies, knowledge-based systems, and knowledge management can be used as part of the conceptual solution for developing multisite software development methods.

***REFERENCES***


1. *P. Wongthongtham, E. Chang, T. S. Dillon, I. Sommerville (2006) "Ontology-based multi-site software development methodology and tools", Journal of Systems Architecture 52, pp. 640–653*
2. *Wongthongtham, P., Chang, E. and Dillon, T. S., (2005)" Towards 'Ontology'-based Software Engineering for Multi-site Software Development", International Conference on Industrial Informatics (INDIN), pp.362-365.*
3. *Udsanee Pakdeetrakulwong, Pornpit Wongthongtham, Naveed Khan (2015) "An Ontology-Based Multi-Agent System to Support Requirements Traceability in Multi-Site Software", ASWEC ' 15 Vol. II, September 28-October 01, 2015, Adelaide, SA, Australia, pp.96-100*
4. *Wu Jianping, (2012) "Novel Software Engineering Knowledge Representation Method for Multi-site Software Development", IEEE, pp.523-526*
5. *Hans-Jorg Happel, and Stefan Seedorf (2001) "Applications of Ontologies in Software Engineering".*
6. *T. S. Dillon FIEEE, E. Chang SMIEEE, P. Wongthongtham (2008) "Ontology-based Software Engineering- Software Engineering 2.0" 19th Australian Conference on Software Engineering, IEEE, pp.13-23.*
7. *Kalyana Chakravarthy Dunuku, Komal, V. Saritha (2013), "Ontology Model of Software Engineering for Multi-Site Software Development", IJSRET, Volume 2 Issue 8, pp 479-485*
8. *Mohamed, S. "New style of software lifecycle strategies–Use Case perspective." International Journal of Management, Information Technology and Engineering 4.3: 99-114.*
9. *Pornpit Wongthongtham, Elizabeth Chang, and Tharam Dillon, (2007) "Multi-site Distributed Software Development: Issues, Solutions, and Challenges" ICCSA 2007, LNCS 4706, Part II, pp. 346–359.*
10. *Hai Dong, Farookh Khadeer Hussain and Elizabeth Chang (2007) "Ontology-based Solutions for Knowledge Sharing Issues in Project Organisations" Inaugural IEEE International Conference on Digital Ecosystems and Technologies , pp.346-351.*
11. *Pornpit Wongthongtham, Elizabeth Chang, Tharam Dillon and Ian Sommerville,(2009) "Development of a Software Engineering Ontology for Multi-site Software Development", IEEE transactions on knowledge and data engineering. p p-1-14.*
12. *Welty, C. A.: Software Engineering. In: Description Logic Handbook (2003) 373-387.*
13. *Ankolekar, A., Sycara, K., Herbsleb, J., Kraut, R., Welty, C.: Supporting Online Problemsolving Communities with the Semantic Web. In Proc. of the 15th Int. Conference on World Wide Web, Edinburgh, Scotland, (2006) pp.575-584.*
14. *A. Crabtree, T. Rodden, S. Benford, Moving with the Times: IT research and the boundaries of CSCW, Compter Supported Cooperative Work (CSCW) The Journal of Collaborative Computing 14 (3) (2005) 217–251.*
15. *E. Carmel, R. Agarwal, Tactical approaches for alleviating distance in global software development, IEEE Software 18 (2) (2001) 22–29.*
16. *A. Mockus, D. M. Weiss, Globalization by chunking: a quantitative approach, IEEE Software 18 (2) (2001) 30– 37.*
17. *R. Heeks et al., Synching or sinking: global software outsourcing relationships, IEEE Software 18 (2) (2001) 54–60.*