The Nucleus **51**, No. 2 (2014) 291-301

The Nucleus

A Quarterly Scientific Journal of Pakistan Atomic Energy Commission

NCLEAM, ISSN 0029-5698

AN ONTOLOGICAL FRAMEWORK FOR REQUIREMENT CHANGE MANAGEMENT IN DISTRIBUTED ENVIRONMENT

A. KHATOON, Y. HAFEEZ, *S. ASGHAR and T. ALI

University Institute of Information Technology, PMAS University of Arid Agriculture, Rawalpindi, Pakistan

(Received December 31, 2013 and accepted in revised form April 30, 2014)

Global Software Development (GSD) is getting fame in the software industry gradually. However, in GSD, multiple and diverse stakeholders are involved in the development of complex software systems. GSD introduces several challenges, i.e. physical distance, time zone, culture difference, language barriers. As requirements play a significant role in any software development. The greatest challenge in GSD environment is to maintain a consistent view of the system even if the requirements change. But at the same time single change in the requirement might affect several other modules. In GSD different people use terms and have different ways of expressing the concepts for which people at remote sites are unable to get uniformity regarding the semantics of the terms. In a global environment requires effective communication and coordination. However, to overcome inconsistencies and ambiguities among the team members and to make the team members aware of the consistent view, a shared and common understanding is required. In this paper an approach beneficial to software industry has been proposed, focusing on changing requirements in a Global Software Development environment. A case study has been used for the evaluation of the proposed approach. Therefore, Requirements change management process has been improved by applying the approach of the case study. The proposed approach is beneficial to the software development organizations where frequent changes occur. It guided the software industry to provide the common understandings to all the development teams residing in remote locations.

Keywords: Requirement Change Management (RCM), Global Software Development (GSD), Ontology, Protégé

1. Introduction

RE is the most considerable phase in the software development life cycle. Requirements engineering (RE) addresses gathering, organization and documenting client's requirements so that the product could meet client's demands [1]. When team members are disseminated anywhere round the globe, RE worth improves. Therefore, main aim is to successfully manage the requirements in global software development (GSD) environment where the development team is distributed at multiple locations and to satisfy customer needs.

RE can be divided into two main activities, i.e. requirements development and requirements management. Requirements development includes requirement elicitation, analysis, documentation and validation. Therefore, requirements development deals with the creation and analysis of the customer demands. Requirements management includes requirement change management and traceability [2]. In software development requirement management is a collaboration intensive activity; moreover more difficulties arise when stakeholders are disseminated [3].

An ontological framework for requirement change management in distributed environment.

A software development when team members are at located at remote sites is termed as Distributed Software Development (DSD). Hence network of distant sub teams, establishes. The sub teams may either be members of the same organization or involvement of different organizations in software development may exist. Distance among the sub teams can vary from a few meters to continents. A type of software development when team members are disseminated across national boundaries is known as global software development (GSD) [4].

Now-a-days, GSD is acting as a model for the software companies [5]. Trends in software development are shifting from collocated development towards GSD in order to achieve benefits; low labor cost, latest technology, right and good quality personal, etc. These benefits are correlated with some of the challenges which are cultural diversity, inadequate communication, temporal difference and knowledge management. Due to these all software project activities are affected, but requirements engineering affected the most. Stakeholders belong to different background and they complete the requirements related activity altogether [6].

In different cities and countries software engineering practices are somehow diverse. Therefore, if software engineering principles are not well realized and followed, communication among the team members becomes a challenging task. When proper communication is not defined, this might give rise to inconsistencies among the team members in distributing

* Corresponding author : sohail.asghar@uaar.edu.pk

knowledge about the changed requirements [7].

Organizations residing in remote locations use different ways to express terms and concepts. Knowledge of various software components must be shared across the organizations, therefore global environment requires collaboration and knowledge sharing [8]. Lack of shared understanding of requirements is the major hurdle for successful requirement engineering in GSD. Shared understanding helps in reducing inconsistencies and ambiguities. It can be achieved when all the participants working on the project has the same recognition of every requirement [5]. There must be a mechanism to handle changing requirements because poorly tackled changes effect quality of products and unsatisfactory results [3].

The literature has proposed numerous techniques to cope with the problem of changing requirements in a GSD environment. But to reduce the inconsistencies and to make the collaboration confident, a mechanism must be introduced which certifies to share knowledge among all the involved stakeholders [9]. To fulfill the client’s need and demands GSD projects requires efficient knowledge management techniques. In successful GSD projects ontological knowledge management techniques are becoming active. Knowledge sharing and knowledge management in the global software development environment can be reduced by means of software engineering ontology [10]. Hence ontology can be used as an infrastructure for knowledge management, which provides a vocabulary related to the domain. So, in this paper, we are concerned about the solution of RCM for GSD. where team members are distributed at remote locations, it is a very challenging task that how requirements are conveyed to the team members, and changes made in requirements at any stage during the software development life cycle would be visible to persons involved in the project.

The remaining paper is divided into following sections. Section 2 defines problem statement. Section 3 discusses related work. Section 4 describes the proposed solution. Section 5 describes an evaluation of the proposed solution. In section 6, we discuss conclusions and future work.

2. Problem Statement

This section discusses the problem statement. Requirements act as a backbone in any type of software development. During software development when development teams as well as clients are disseminated at a remote location, a consistent view of the system must be maintained among the stakeholders even when the requirements change. But GSD issues, i.e. physical distance, temporal difference, cultural difference etc, act as huge obstacles to make the software development inconsistent. Moreover give rise to ambiguities as different people belongs to different locations and express the terms and concepts in different ways. Summarizing all, an effective communication and coordination is required to make the changes available to all the stakeholders. That might be achieved by affirming knowledge management. That could be achieved by providing a shared understanding. Hence ontology could be used for providing a shared understanding and a common vocabulary.

3. Related Work

In this section we have analyzed literature covering requirements related issues in global software development and found that the area of a requirement change in a GSD environment is given less importance. In ref. [3] author proposed a model for requirement change management for the global environment (GRCM) and evaluated this model by using a case study, which helped to improve the RE process by analyzing common RE activities for GSD organization. But the limitation of this model is that for communication no mediated technology was suggested, moreover the model is applicable on the pre-development phases of the software development.

In Ref [10] author presented a framework for requirement management in GSD environment. Author used ontology to manage the project knowledge and also proposed constraints to control requirement management process. However, the limitation in the proposed framework is how to communicate and requirements repository is omitted. Ref. [11] stated some requirement management practices in a software organization. These practices include creation of software teams, overcoming cultural issues, providing a glossary of key words, defining roles and responsibilities, etc. weakness in the proposed work was authored hypothesizes determined the practices that might be included for requirement management in GSD.

In Ref. [2] author suggested ontology for knowledge management to minimize ambiguities and to provide a shared understanding. However, ontology in the framework is defined in an abstract level. Another author in ref. [12] proposed ontological solution to requirement management has been proposed. The approach helped the stakeholders by establishing requirements repository, by performing traceability and establishing a communication agenda among the development team. However, change management process might be elaborated further in the ontology.

4. Proposed Solution

For managing changing requirements in global software development an effective process is needed. As knowledge sharing and knowledge management in the global software development environment can be achieved by means of software engineering ontology. If knowledge management is achieved, then communication and coordination problem could be overcome. Therefore, ontological solution is provided for RCM in GSD. RCM-GSD is application ontology, as it covers software requirement change management domain for global software development environment. Several methodologies for ontology development are available, but no one is considered the standard for ontology development [9].

To develop ontologies, its development process needs to be understood. Ontology development was based on following phases [13],

Specification: The aim of the specification is to get informal knowledge of the domain

Conceptualization: aim of conceptualization is to assemble informal knowledge using some notations independent of formal language; usually UML Class diagram notations are used.

Implementation: In implementation ontology development tool protégé is used to develop ontologies.

Scope and the application of the ontology can be determined by means of competency questions, moreover developed ontology can be evaluated by means of competency questions [14]. Before designing and development of the ontology, some competency needs to be designed. Developed ontology must answer some of the competency questions to assure the correctness of the ontology.

i. What are the activities in a change management process in a GSD environment?

ii. What are the roles involved in the Change management process?

iii. What artifacts are stored in a change management process?

iv. What are the factors which affect the GSD project?

Understanding of the concepts in an informal way is captured from [15.-18] of RCM and [12, 18- 21] for GSD. Figure 1 shows the conceptual model for RCM in GSD environment, elaboration of the conceptual model will give the following detail.

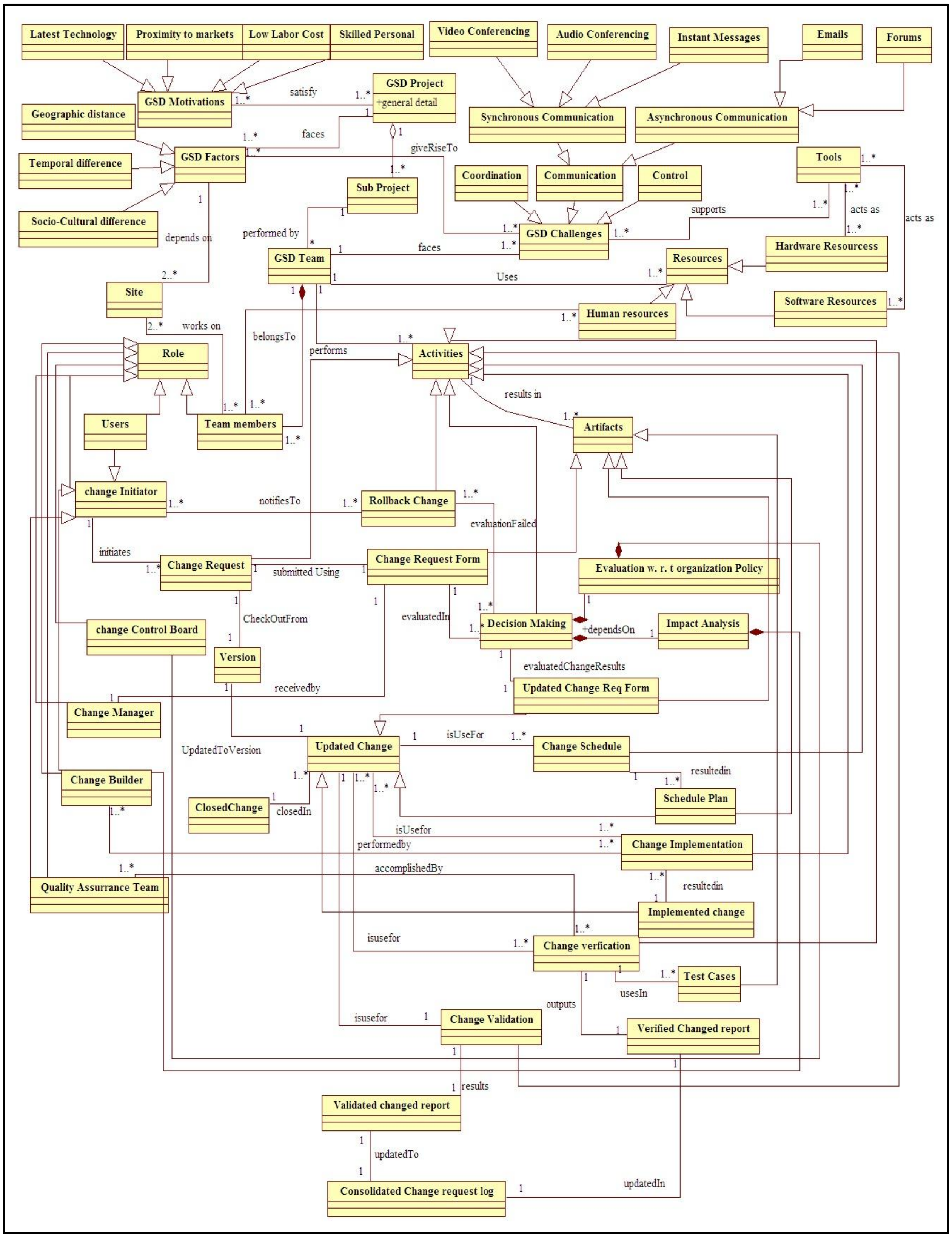
GSD project is accomplished by GSD teams. GSD teams are composed of one or more team members residing at more than one location performing different activities. The scope of the research is limited to requirement change management (RCM) in GSD therefore, RCM activities taken into account are change request, evaluation, change schedule, change implementation, change verification, change validation and finally change closure. Team members' i.e. change manager, change builders, QA team and Change control board are responsible for the development of the software product. For the successful completion of the software product involvement of the client must be active. Team members and clients are specialized for higher entity "Role".

Moreover, Change initiator initiates the change by using a change request form. Change initiator is the person who initiates the change. Change placed might be internal and external, external change is the change initiated by the customers or clients whereas internal change is the change placed by the team members i.e. change builders or QA Team. GSD project largely provides some motivations due to which several organizations are adopting it. GSD motivations include low labor cost, access to skilled or talented personal proximity to market or customers, etc. Some of the GSD factors which negatively affect a GSD project also exist. These factors are geographic distance, time zone difference and cultural and language difference as they causes communication, coordination, delay problems.

All these are the major sources of inconsistencies, misunderstandings and ambiguities. GSD team performs activities resulting in different artifacts uses resources. Some of the resources can be used to reduce GSD threats, i.e. communication, coordination, control, etc. Change management process starts when change initiator initiates change request from version. Change request performed by change initiator when submitted through the change request form must be evaluated.

As the change request is an activity describing the change that is requested. Change request is submitted to the change manager who then forwards the request to change the control board and change builders for evaluation. Evaluation is the most important and decisive activity; in which decision is made whether to implement change or not. The decision is taken by the change control board and change builder. The Change Control Board evaluates the technical aspects of the project, whereas change builders check the impact of the changed requirement on the other requirements.

Updated change request form resulted from the evaluation (Decision Making). This updated change request form is used by the change manager to schedule the requested change. A schedule plan is created or providing the guidelines to the change builders. Change builders residing in more than one location are communicated to follow the schedule plan. After


Latest Technology
Proximity to markets
Low Labor Cost
Skilled Personal
Video Conferencing
Audio Conferencing
Instant Messages
Emails
Forums
GSD Motivations
satisfy
GSD Project
+general detail
Geographic distance
Synchronous Communication
Asynchronous Communication
faces
giveRiseTo
GSD Factors
Temporal difference
Tools
Coordination
Communication
Control
Sub Project
Socio-Cultural difference
acts as
supports
acts as
depends on
performed by
GSD Challenges
Hardware Resourcess
faces
GSD Team
Resources
Uses
Site
Human resources
Software Resources
works on
belongsTo
performs
Role
Activities
results in
Users
Team members
Artifacts
change Initiator
notifiesTo
Rollback Change
evaluationFailed
initiates
Change Request
submitted Using
Change Request Form
Evaluation w. r. t organization Policy
CheckOutFrom
evaluatedIn
Decision Making
+dependsOn
Impact Analysis
change Control Board
Version
evaluatedChangeResults
Updated Change Req Form
receivedby
Change Manager
Updated Change
isUseFor
Change Schedule
UpdatedToVersion
resultedin
Change Builder
ClosedChange
closedIn
Schedule Plan
isUsefor
performedby
Change Implementation
accomplishedBy
resultedin
Quality Assurance Team
Implemented change
isusefor
Change verfication
Test Cases
usesIn
outputs
isusefor
Change Validation
Verified Changed report
results
Validated changed report
updatedTo
Consolidated Change request log
updatedIn

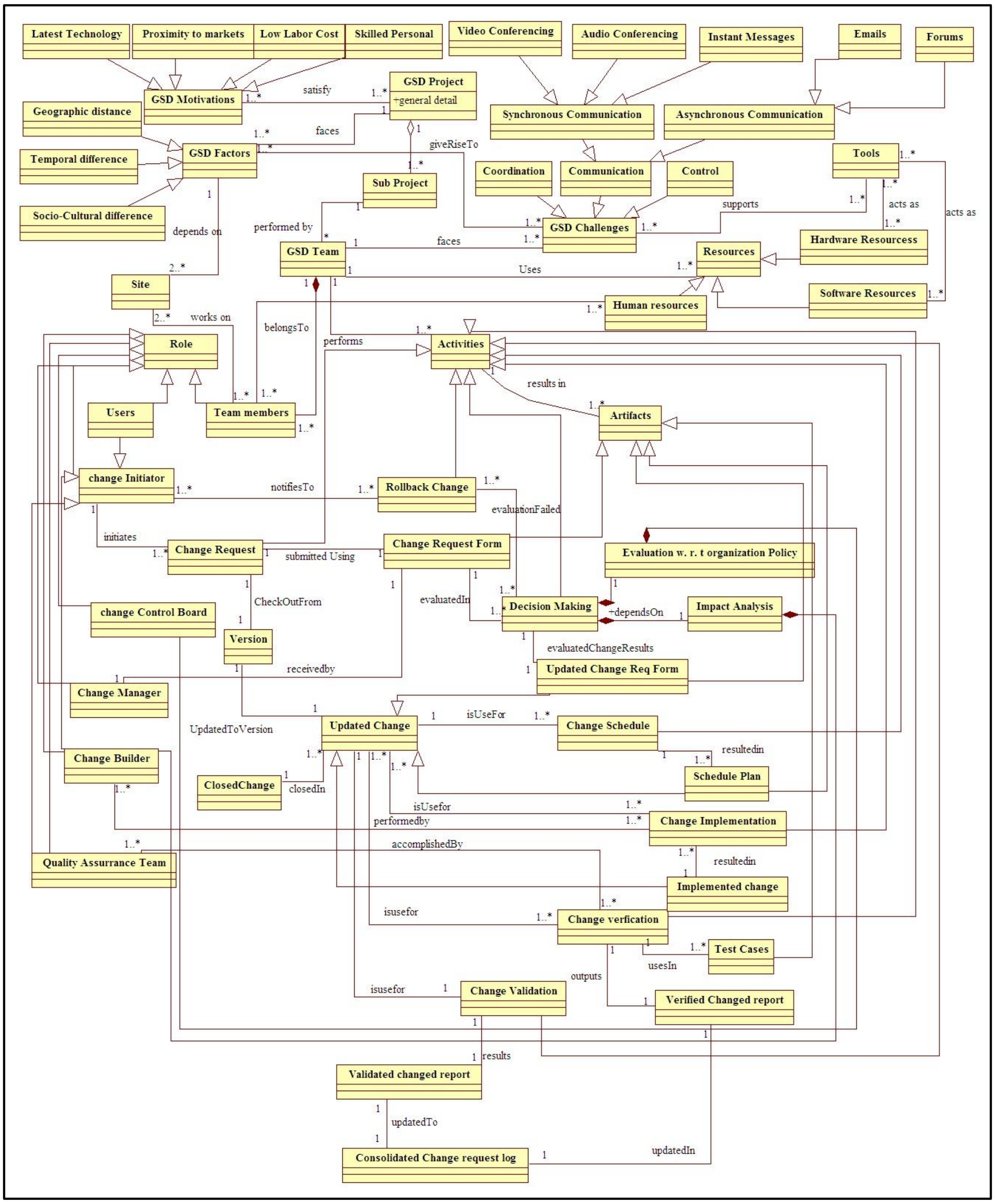


Figure 1. Conceptual model for RCM in GSD

receiving the guideline, team lead at each location will acknowledge by using synchronous and asynchronous communication. Change builders are responsible for implementing the change after performing the checkout from the version in which changes are to be incorporated. Change builders implement the change and apply modifications to the checked out. Implemented change is updated to the version (named Updated Change, it is a temporary version for the artifacts).

After implementing the change it must be verified. Verification responsibility is assigned to the QA team. QA team checks whether the implemented change is error free or implemented correctly. Verified change report is generated and updated in the consolidated change request log. Consolidated change request log is the place where details of all the changes with their status are recorded; moreover it is maintained by the change manager. Finally, change is validated by the change initiator; therefore, implemented changes are validated by using verified change reports and updated to the consolidated change request log. Finally the validated implemented change is updated to the version maintaining the project data.

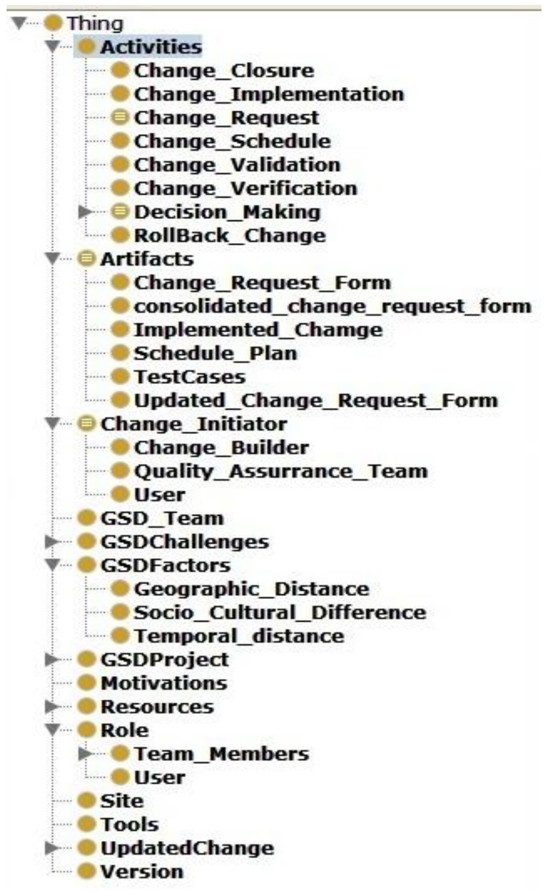
It is worthwhile to mention that some concepts used in RCM and GSD must have super types and sub types. For instance, client and team members merged in a super type "Role". Change can be initiated by client, change builder or QA team; therefore, "Change initiator" can be represented as a super type. Similarly a GSD project satisfies GSD motivations (namely low labor cost, access to skilled personnel, proximity to markets and customers) are the sub types of GSD motivation concept. Yellow blocks in the Figure1 represent classes or concepts, association is shown by a single line, whereas arrow indicates inheritance among the concepts or classes. Hence, integration of task ontology with the domain ontology gives rise to application ontology. Therefore, application ontology is being developed.

To analyze knowledge sharing, reducing ambiguities and inconsistencies in requirement change management process in GSD environment ontology is constructed in OWL using protégé editor. Developed ontology must answer the competency question sketched at the start of the proposed solution. Ontology comprises of classes, properties and constraints or axioms. An ontology consists of a set of entities representing domain concepts that can be organized hierarchically also referred to concepts or classes. All entities have properties and features that describe them. Moreover, these properties take part in applying constraints. Each entity consists of a set of individual having the same properties and constraints [9]. Moreover, the definition of some of the core concepts used in the Ontology is described in Table 1.

In protégé "Thing" is the super class of all the classes or concepts added. The Protégé interface of the concepts or classes added is shown in Figure 2. After extraction of the concepts association between the concepts was were mapped by object properties. For example, Change Request submitted using Change Request Form. In this proposition "Change Request"

Table 1. Definition of the core concepts mapped in requirement change management in global software development ontology.

Core Concepts	Definition of the Core Concepts
GSD Project	General detail of the project assigned by the clients or customers.
GSD Motivations	In a GSD environment, motivations exist, that attracts the organization to perceive GSD environment.
GSD Team	Detail of team members involved in a GSD project.
Activities	What activities are followed in a requirement change management process
Role	Involves team members as well as clients that have certain responsibilities to fulfill the project.
Artifacts	Involves the deliverables developed by different "Roles"
Change initiator	Person or personal responsible for specifying change request.
Consolidated Change Request Log	Details of all the changes with their status exist.
Version	Accomplished changes, different modules reside.
Updated Change	A special type of version where artifacts are updated temporarily.



```
Thing
  Activities
    Change_Closure
    Change_Implementation
    Change_Request
    Change_Schedule
    Change_Validation
    Change_Verification
    Decision_Making
    RollBack_Change
  Artifacts
    Change_Request_Form
    consolidated_change_request_form
    Implemented_Chamge
    Schedule_Plan
    TestCases
    Updated_Change_Request_Form
  Change_Initiator
    Change_Builder
    Quality_Assurance_Team
    User
  GSD_Team
  GSDChallenges
  GSDFactors
    Geographic_Distance
    Socio_Cultural_Difference
    Temporal_distance
  GSDProject
  Motivations
  Resources
  Role
    Team_Members
    User
  Site
  Tools
  UpdatedChange
  Version
```

Figure 2. Protégé interface showing list of concepts or classes.

Table 2. Elaboration of some object properties.

Object Property	Domain	Range	Description
Works On	Team members	Site	Team members “works on” Site
Faces	GSD Projects	GSD Factors	GSD Projects “faces” GSD Factors
Give Rise To	GSD Factors	GSD Challenges	GSD Factors “giveRiseTo” GSD Challenges
Initiates	Change Initiator	Change Request	Change Initiator “initiates” the Change Request.
Submitted Using	Change Request	Change Request form	Change Request is “submittedusing” Change Request Form
Evaluated In	Change Request Form	Decision Making	Change Request Form is “evaluatedIn” Decision making Activity
Evaluated Change Results	Decision Making	Updated Change Request Form	Decision Making “evaluatedChangeResults” in Updated Change Request Form
Check Out From	Change Request	Version	Change request is “checkedOut” from Version
Received By	Change Request Form	Change Manager	Change Request Form is “receivedBy” Change Manager.
Is Use For	Updated Change Request Form	Change Schedule	Updated Change Request Form “isUseFor” Change Schedule
Resulted In	Change Schedule	Schedule Plan	Change Schedule” resultedIn” Schedule Plan
Is Use For	Schedule Plan	Change Implementation	Schedule Plan “isUseFor” Change Implementation

and “Change Request Form” are the concepts, whereas, “submitted using” determines the object property or predicate. Predicates or properties are mapped by using restrictions as domain and range where domain indicates the concept from where association starts and range recommends the concept showing the end of the association. For instance, Change Request” and “Change Request Form” propose domain and range respectively. Table 2 explains some of the object properties. Some of the restrictions (can be termed as axioms or constraints) are also applied to the classes. Axioms can be disjoint (concepts must be distinct) or equivalent (concepts can be equal).

5. Results and Discussion

To analyze the effectiveness of the developed ontology a case study of the campus management system (CMS) of a university is used. A public sector university X planned for a campus management system (CMS). University officials assigned the development of the system to a software organization. After clear analysis of the system according to the requirements placed, CMS consisted of a number of modules but some modules were top priority modules. These modules were examination system (ES), attendance system (AS) and library management system (LMS). Therefore, software organization distributed the modules at remote locations; ES was assigned to the organization in Pakistan whereas the other two were allotted to a GSD team in China. At the time of analysis instances were mapped onto the developed ontology. So that disseminated team member may use the ontology.

As team members were in Pakistan and China, therefore GSD factors which affect the project most were cultural and language difference and up to some extent geographical difference. ES was assigned to the team members in Pakistan and the other were assigned to the development teams of an organization. They used the developed ontology to understand the structure of the project and to reduce the ambiguities in RCM process. Frequent changes were placed from the university in some ES for which AS module was also affected. Appropriate instances related to the case study were added by the organization according to their requirement and work distribution. Developed ontology answered the competency questions when evaluated by using reasoner plug-in present in protégé by the software organizations. Reasoner is a tool which acts as a plug-in used for querying and reasoning in ontology. Moreover, it is also helpful to process the inference rules, to deduce more rules and to make consistency certain.

Figure 3 shows the comprehensive structure of the ontology after inferring the rules applied. Moreover, it also acted as a project map regarding the RCM process in a GSD environment. Therefore, taking into account the whole proposed work and evaluation, firstly, terms (concepts and properties) are enumerated. Secondly a conceptual map was graphically represented with the necessary conditions and constraints were applied. By means of ontology development tool protégé conceptual framework was implemented. An evaluation was performed by the protégé tool to show the structure of the ontology. Evaluation by using tool protégé did not show inconsistencies as the conditions applied were consistent.

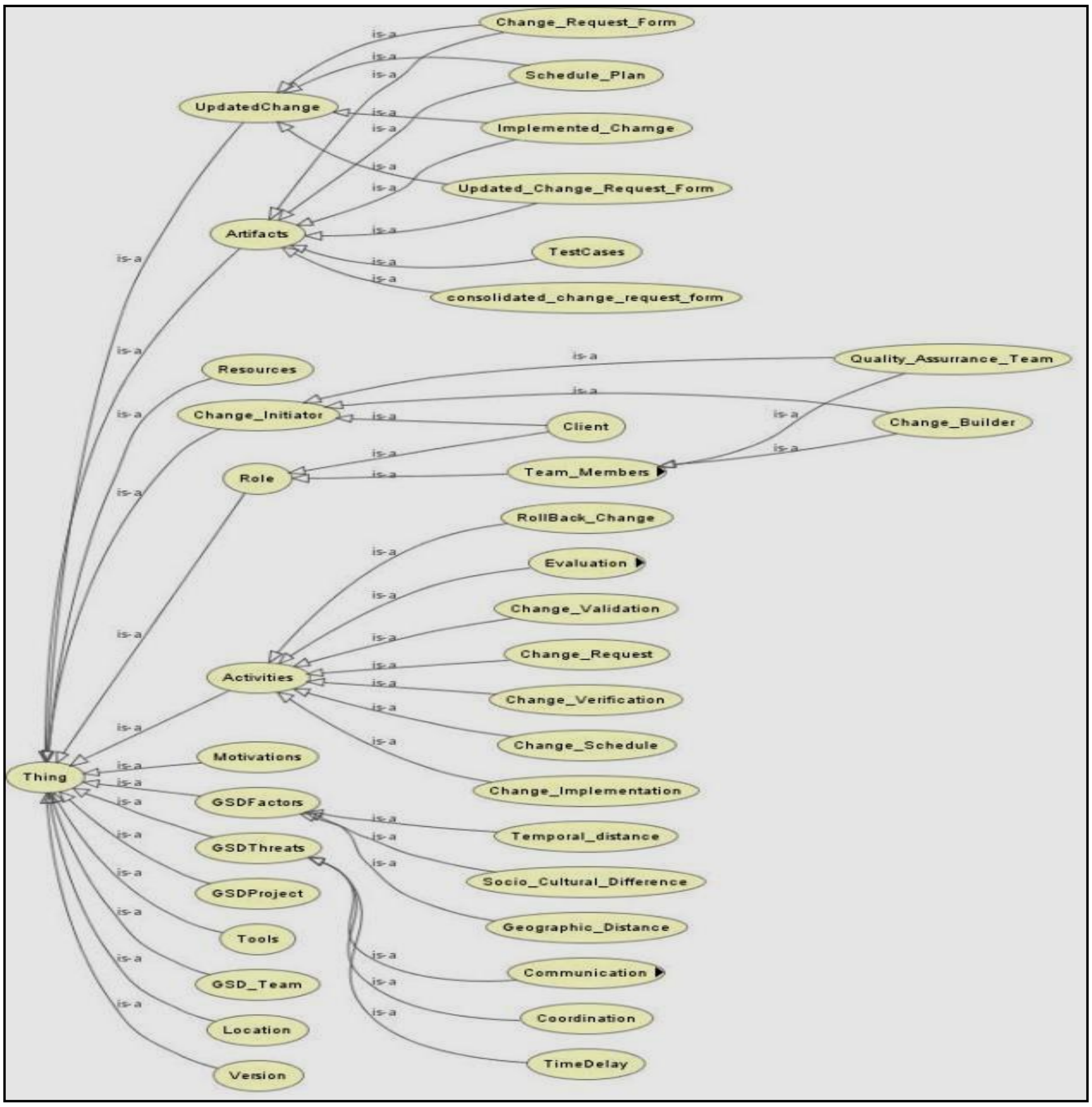


Figure 3. Inferred structure of the developed ontology.

Table 3. Expert opinions.

Parameters	Change Manager (Pakistan)	Change Builder (Pakistan)	QA Team (Pakistan)	Change Builder (China)	QA Team (China)	Change Control Board
Knowledge Management	√	√	√	√	√	√
Knowledge Sharing	√	√	√	√	√	√
Reusability	√	∞	∞	∞	√	×
Reduction of Ambiguities	√	√	√	√	√	√
Communication	√	√	√	√	√	√
Coordination	√	√	√	√	√	√
Easy To Use	∞	√	√	√	√	∞

Strongly Agree= √ Partially Agree = ∞ Not Agree =×

Table 3 and Figure 4 contain expert reviews and graphical representation of reviews after adoption of ontology. Experts were the (Team members), who used ontology in the campus management system (CMS) of a university. After the use of the ontology team members (onshore and offshore) involved in the project expressed their views for some of the parameters which were satisfied by the ontology. Parameters were knowledge management, knowledge sharing, reusability, reduction of ambiguities, communication, coordination and easy to use. Reviews were taken from the Change Manager, Change Builder (Pakistan) who is working as a Team Lead in Pakistan, Change Builder (China) who is working in China as a Team Lead, two Quality Assurance personnel one from Pakistan and one from China and CCB (Change Control Board) representative.

They recorded their reviews after using an ontology to show the hierarchy and a project map of the RCM process in GSD paradigm. Moreover, determine the correctness of the conceptual model by using the ontology practically to find the consistency among all the concepts. Knowledge Management and knowledge sharing was improved that helped in the improvement of communication and coordination among the stakeholders. Therefore, majority of the participants showed higher satisfaction level, whereas some participants showed partial satisfaction for reusability and ease of use.

Experts' opinion shows that the conceptual model and proposed ontology showed consistency and gave better results than the traditional method they usually use. RCM process in a GSD environment can be improved by making the knowledge management confident. Knowledge management helps to overcome communication and collaboration by reducing ambiguities and inconsistencies.

6. Conclusion and Future Work

Requirement management is the collaboration intensive activity. When a change in requirements is frequent it is necessary to manage them to satisfy customers' needs. When participants are distributed at remote locations a challenging situation arises to manage the change in requirements. As GSD team members residing at different locations belong to different background and culture. It might enhance inconsistencies and ambiguities because communication and coordination is not achieved.

To fulfill the client's demands GSD projects requires efficient knowledge management techniques. Ontological knowledge management techniques are becoming active in GSD projects. Ambiguities and inconsistencies in the global software development environment can be reduced by providing a common and shared understanding. Therefore, an ontological solution for RCM in GSD environment is provided. The solution was evaluated by implementing a case study and expert's opinions were collected to analyze the effectiveness of the solution. Proposed framework provided a guide to the GSD team to manage the changing requirements by reducing ambiguities and inconsistencies. In near work we will refine the change management process and will evaluate ontology by implementing it on more case studies.

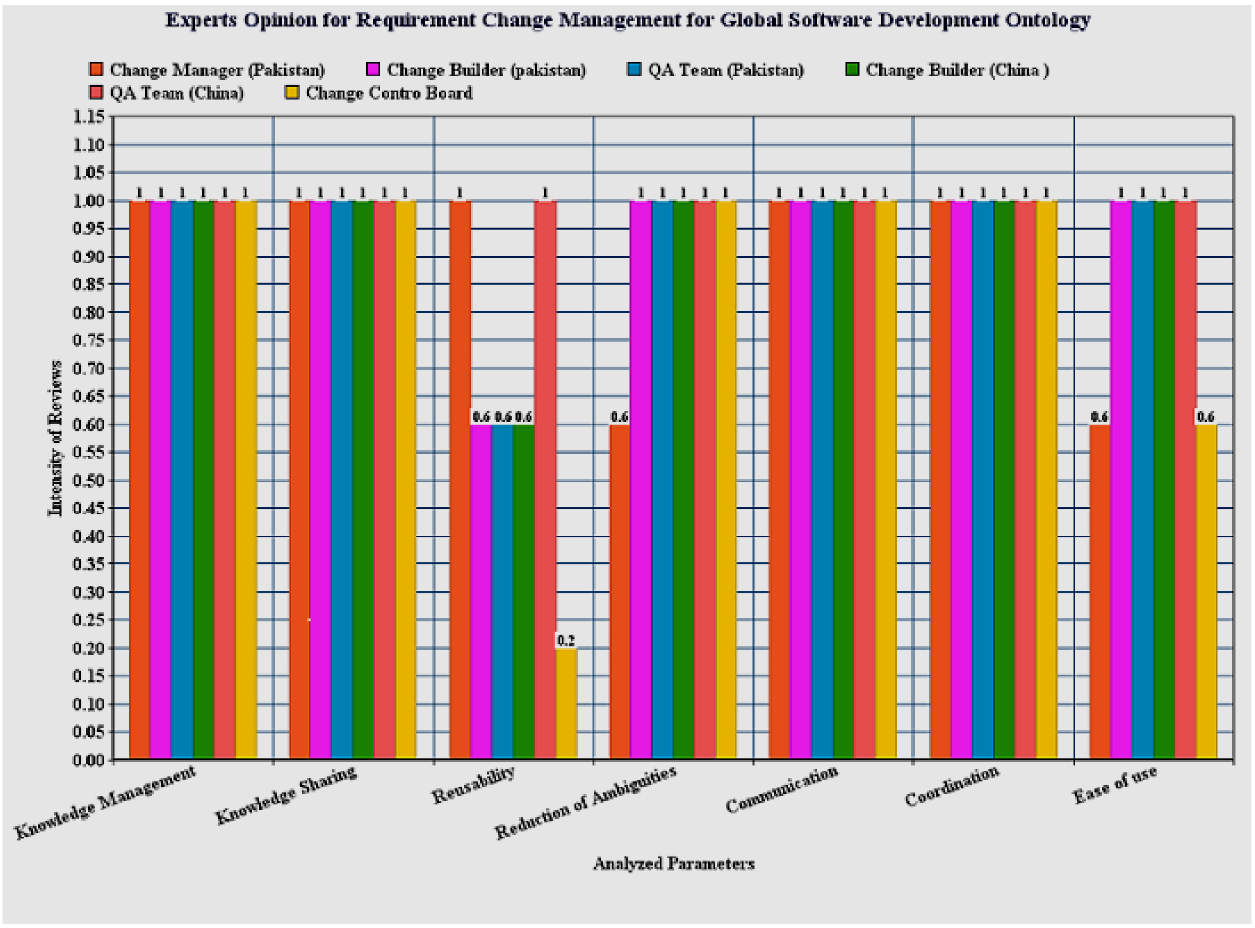


Figure 4. Experts opinion.

References

[1] H. Naz, Y. Hafeez, S. Asghar, M.A. Abbas and A. Khatoon, The 5th International Conference on Computer Science and Information Technology, Jordan, ISBN 978-9957-8592-1-3 (2013).

[2] Y. Hafeez, M. Riaz, S. Asghar, H.Naz, S.M. Gilani, A. Batool, M. Ahmed and M.S. Hassan, 7th International Conference on Computing and Convergence Technology, 944 – 948, 978-1-4673-0894-6 (Dec. 3-5, 2012).

[3] W. Hussain and T. Clear, 2nd International Requirements Engineering Efficiency Workshop held at, Essen, Germany, http://hdl.handle.net/10292/4565, (2012-03-19).

[4] M. Jim´enez, M. Piattini and A.Vizca´ıno, Hindawi Publishing Corporation, Advances in Software Engineering, Volume 2009, Article ID 710971, 14 pages, doi:10.1155/2009/710971.

[5] J.D. Herbsleb, A. Mockus, A. Thomas, F. Rebecca and E. Grinter, ICSE Conference (2001) pp. 81-90, 10.1109/ICSE.2001.919083 (12-19 May 2001

[6] M. Humayun and C. Gang, Int. J. of Software Engg. and its Applications **7**, No. 1 (2013) 79.

[7] P. Wongthongtsham, I. Sommerville, E. Chang and T. Dillon, IEEE Transactions on Knowledge and Data Engineering **21**, No. 8 (2009) 1205.

[8] S.G. Shiva, S.B. Lee, L.A. Shala and C.B. Simons, International Journal of Distributed Sensor Networks **5**, No. 1 (2009) 6.

[9] P.C. Ana, I. Steinmacher and G.L. Camila, CLEI Electronic Journal **14**, No.2 (2011) 12.

[10] S.A. Kumar and T.A. Kumar, International Journal of Software Engineering & Applications **2**, No. 4 (2011) 107.

[11] D. Smite, Journal of Universal Knowledge Management **1**, No. 2 (2006) 69.

[12] R. Lai and N. Ali, Advances in Information Sciences **1**, No. 1 (2013) 38.

[13] G. Brusa, M.L. Caliusco and O. Chiotti, Proceedings of the Second Australasian Workshop on Advances in Ontologies **72** (2006) 7.

[14] G.L. Simmons and T.S. Dillon, IFIP International Federation for Information Processing Volume 203, Open Source Systems, Eds. E. Damiani, B. Fitzgerald, W. Scacchi, M. Scotto, G. Succi (Boston: Springer) (2006) pp. 65-75.

[15] S. Imtiaz, N.Ikram and S.Imtiaz, Proceedings of the Fourth IASTED International Conference on Advances in Computer Science and Technology (2008) pp. 121-128, ACTA Press Anaheim, CA, USA ©(2008).

[16] S. Ramzan and N.Ikram, Proc. of the 10th IEEE International Multitopic Conference, Islamabad, Pakistan, (2006).pp 219

[17] S.V. Nagabhushan and K.S. Swarnalatha, Global Journal of Management and Business Research **10**, No. 9 (December 2010) 20.

[18] D.E. Damian and D. Zowghi, Requirements Engineering Journal **8** (2003) 149.

[19] M.Tanner, Journal of Information Technology, and Organizations **4** (2009) 57.

[20] Y.H. Shah, M.Raza and S.U. Haq, International Journal of Advanced Science and Technology **40**, (March, 2012).1.

[21] A. Vizcaı´no, F.I. Garcı´a, J.C. Caballero and V.M. Piattini, The Institution of Engineering and Technology Published in IET Software doi: 10.1049/iet-sen.2011.0087 (2012).