Requirements Elicitation Challenges and Strategies for GSD Based Projects

M. Aqeel Iqbal, and Asadullah Shah

***Abstract*— The geographically distributed software development has attained a significant attraction in software development industry since last few decades due to its wide acceptance across the globe. The main essence of geographically distributed software development is to expand company working by follow-the-sun strategy, get skilled labors at cheaper rates from low economy regions of world and pool-up company resources to raise their utilization graph. The requirements elicitation process is considered as one of the most challenging tasks during software development process in traditional as well as geographically distributed software development projects due to its social and collaborative nature. This article presents a comprehensive insight about major challenges faced by requirements elicitation teams engaged in geographically distributed software development projects. The article also elaborates the possible strategies, which can be adopted to minimize the negative impact of geological distribution of software development activities. The presented study will help the requirements elicitation teams to better plane their requirements elicitation sessions for geological distributed software development projects by encountering the expected hurdles.**

***Index Terms*— Requirements Elicitation, Requirements Elicitation in GSD, Requirements Elicitation Challenges, Distributed Requirements Elicitation, Distributed Software Development.**

I. Introduction

THE requirements engineering is considered as the most critical part of software development process due to its dual nature of engineering work accomplished through social collaborations. The dual nature of requirements engineering process makes it a challenging endeavor for software development teams. The main objective of requirements engineering stage is to collect the product requirements from its intended clients and formally document them as specification document [1-5]. In order to accomplish the requirements engineering objectives, it is further divided into four major phases of requirements elicitation process, requirements analysis and negotiation process, requirements specification process and requirements validation process. Normally, these four phases of requirements engineering process are executed in form of an incremental iterative process. These requirements engineering process is usually known as requirements engineering spiral process as is shown in Fig. 1. The given figure is a self-exploratory figure, which shows that internal phases of requirements engineering are executed as spiral activities. The requirements engineering process starts from requirements elicitation and ends-up after requirements validation [6-10]. The requirements elicitation process starts initially and ends-up with a set of informal statements of requirements as its outcome. After the completion of requirements, elicitation phase, the requirements analysis and negotiation phases is started and it ends-up with a set of agreed requirements as its outcome. After the completion of requirements analysis and negotiation phase, the requirements specification phase starts and ends-up with a draft requirements document as its outcome.

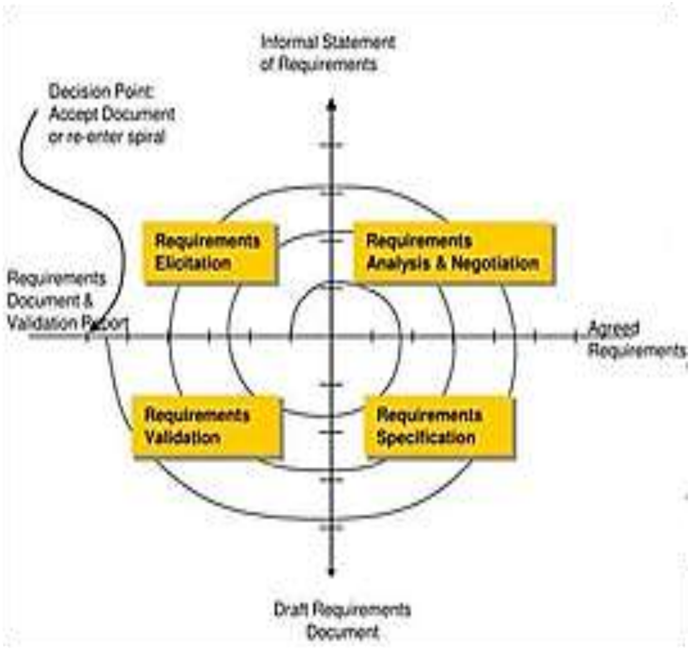
Informal Statement of Requirements
Decision Point: Accept Document or re-enter spiral
Requirements Elicitation
Requirements Analysis & Negotiation
Requirements Document & Validation Report
Agreed Requirements
Requirements Validation
Requirements Specification
Draft Requirements Document

Fig. 1: Requirements Engineering Process.

After the completion of the requirements specification phase, the requirements validation phase starts and ends-up with a validated requirements document and a requirements validation report as its outcomes. After the completion of requirements validation phase, decision is made about either accept the requirements document and release it or re-enter the spiral again for additional iterations [11-15].

Kulliyyah of ICT, International Islamic University Malaysia (correspondence e-mail: maqeeliqbal21@hotmail.com)

Received: 5-02-2021, Revised: 25-03-2021, Accepted: 14-04-2021

The requirements elicitation is considered as the frontline task performed during requirements engineering process [1]. The requirements elicitation task is full of formal and informal communications between requirements analyst's team and product user's team to reach the product conception [2, 16-23]. Therefore, the requirements elicitation is considered as a socially enriched task of whole software development life cycle. The requirements elicitation task is further sub-divided into four major phases including the objective establishment, the background understanding, the knowledge organization and the requirements gathering as is shown in Fig. 2.

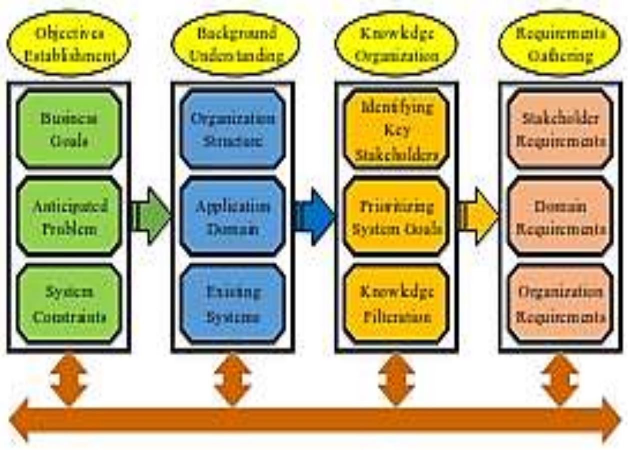

Fig. 1: Phases in Requirements Elicitation

In the phase of objectives establishment, three activities are performed including business goals achievements, anticipated problem scoping and system constraints fixing. After the completion of objective establishment, background-understanding phase starts. In this phase, three activities are performed including organization structure understanding, application domain understanding and existing systems understanding [24-26]. After the completion of background-understanding phase, the knowledge organization phase starts. In this phase, three activities are performed including identifying key stakeholders, prioritizing system goals and knowledge filtration. After the completion of knowledge organization phase, the requirements gathering phase starts. In this phase, three activities are performed including stakeholder's requirements gathering, domain requirements gathering and organization requirements gathering. All these four phases of requirements elicitation are performed iteratively/incrementally to elicit the product requirements from its intended users and meet the vision and scope of the product defined at early stages of software development life cycle.

II. GLOBALLY DISTRIBUTED SOFTWARE DEVELOPMENT

Globally Distributed Software Development commonly known as global software development (GSD) has become the key trend in the area of software development. It is motivated by the opportunities of reaching mobility in resources, obtaining extra knowledge, getting better-skilled labor, speeding time-to-market and increasing operational efficiency [3]. The globally distributed software development is based on the collaboration of client teams, onsite development team and offshore development team as is shown in Fig. 3.

Fig. 3: Globally Distributed Software Teams

Therefore, globally distributed software development is accompanied with many challenges along with its associated benefits [3]. The software development companies need to manage the inherited challenges of global software development to gain the business level benefits generated from it. The majority of the software development activities become challenging in global software development [4]. The most important activity influenced by global software development context include the requirements elicitation process running between distributed user's teams and analyst's teams [5]. The main challenges faced by requirements elicitation process in global software development projects include issues raised due to geographical distribution, temporal diversity, cultural diversity and linguistic diversity. The geographical distribution becomes the main cause of diversity in time zones, diversity in cultures and diversity in linguistic aspects [4]. High geographical distribution raises these diversities while low geographical distributions reduces these diversities and vice versa. In order to manage the effects of geographical distribution on the work products, the software development companies manage these aspects by giving trainings to their employees about client's organization cultures and manage their works by adjusting working hours according to the overlapping time intervals [6].

III. REQUIREMENTS ELICITATION CHALLENGES IN GSD

Consider the given Table-I, which shows contextual factors involved in global software development projects. The table shows that there are nine major contextual factors involved in globally distributed software development projects. These nine contextual factors include cultural diversity, language variations, customized user interfaces, varied local standards, different local regulations, regional issues, educational backgrounds, work context issues and environmental conditions [5, 7, 8].

Table-I: Contextual Factors in RE-GSD

S#	Contextual Factor	Effects on System Development
1	Temporal Diversity	The temporal diversity may drive the changes in the development schedules and working modes. It also affects communicational modes of development teams.
2	Cultural Diversity	The cultural diversity may drive the changes in system functions, features and interactional behavior. It also affects the design and aesthetics of user interface.
3	Language Variation	Customer may prefer to use their local languages, which might be different from the languages used by developers.
4	Customized User Interface	The variations in social and cultural norms may drive the variations in design of user interfaces including aesthetics, design and interaction modes.
5	Varied Local Standards	The customer may have varied local standards including measurement standards, calendars. This variation can affect their working patterns.
6	Different Local Regulations	The customer may have different rules and regulations imposed by their local authorities and country policy matters.
7	Regional Issues	The multiple users belonging to different regions may require different types and levels of customizations in product development.
8	Educational Backgrounds	The customers may have different educational standards and backgrounds. Their level of understanding for software may vary.
9	Work Context Issues	Different cultures, languages, local regulations and political contexts may change the working contexts of customers.
10	Environmental Conditions	The different geological locations may cause different environmental conditions for system use and maintenance.

A. Temporal Diversity

The temporal diversity may drive the changes in the development schedules and working modes. It also affects the communicational modes of development teams. The main source of temporal diversity is geographical distribution around the different time zones of the world. The variations of time zones push the software development teams to shift their working hours and working modes by considering clients overlapping work-hours.

The temporal overlap directly affects the communicational modes/styles used by software development teams to interact with their clients. If the temporal overlap between software developer's team and product client's team is low, then development teams are mostly recommended to use asynchronous communication modes like emails and asynchronous interactive white boards. If the temporal overlap between software developer's team and product client's team is high, then development teams are mostly recommended to use the synchronous communication modes like chats and synchronous interactive white boards. If the temporal overlap between software developer's team and product client's team is medium, then development teams are mostly recommended to use mixed communication modes (synchronous as well as asynchronous).

B. Cultural Diversity

The cultural diversity may drive the changes in system functions, features and interactional behavior. It also affects the design and aesthetics of user interface. The cultural diversity may affect the design of user interfaces as well as functionalities and features of the product. The main effect of cultural diversity on user interfaces may appear in the form of variations in the aesthetics of GUI designs including variations in styles, variations in coloring schemes and variations in text displaying modes. Therefore, the cultural diversity directly affects the aesthetical design layouts of the user interfaces of the product.

C. Language Variations

Customer may prefer to use their local languages, which might be different from the languages used by developers. The customer may prefer to use their local languages during communications with developers about product requirements. In addition, it quite possible that, customer/users may prefer to use their local languages on user interfaces like GUI. In such scenarios, we may need to use language translation plugins provided by third party venders in product development.

D. Customized User Interface

The variations in social and cultural norms may drive the variations in design of user interfaces including aesthetics, design and mutual interaction modes. The variations in client's cultural norms may drive the changes in the aesthetical design aspects of graphical user interfaces of the product. The interaction modes and scenarios may also vary due to the variations in social and cultural norms of the clients. In addition, the working norms of client organizations may also affect the design of customized user interfaces of the product for some specific clients.

E. Varied Local Standards

The customer may have varied local standards including measurement standards, calendars. This variation can affect their working patterns. This variation in local standards would drive the implementation of system and design of user interfaces of the anticipated product. The software development companies need to keep aligned with the local regulations of the client's organizations. This is because client's organization is going to use the product as per their native regulations.

F. Different Local Regulations

The customer may have different rules and regulations imposed by their local authorities and country policy matters. This variation in local regulations would also drive the implementation of system and design of user interfaces of the anticipated product. The software development companies need to keep aligned with the local regulations of the client's organizations. This is because client's organization is going to use the product as per their native regulations.

G. Regional Issues

The multiple users belonging to different regions may require different types and levels of customizations in product development. These customizations may include customizations in user interfaces, main product utilities, product features and product quality attributes. Hence, it is mandatory for the requirements elicitation teams to consider the regional issues of the client's organization to capture their software needs with more accuracy and relevancy.

H. Educational Backgrounds

The customers may have different educational standards and backgrounds. Their level of understanding for software may vary. The variations in educational background may directly affect the design of user interfaces. Hence, it is mandatory for the requirements elicitation teams to consider the educational backgrounds of the client to capture their software needs with more accuracy and relevancy.

I. Work Context Issues

Different cultures, languages, local regulations and political contexts may change the working contexts of customers. The working context is mostly a key indicator about the working environment of the client's organization. Hence, work context consideration would be traced from the background understanding stage of the requirements elicitation phase.

*J. Environmental Condition*s

The different geological locations may cause different environmental conditions for system use and maintenance. This would cause changes to user interface designs and functionality variations specific to different environmental conditions. The variations in environmental conditions may also be caused due to the temporal dispersions.

IV. REQUIREMENTS PROBLEMS OCCURRED DUE TO RE-GSD CONTEXT

We presents a set of problems or issues raised in system/software requirements due to the requirements engineering in global software development (RE-GSD) context. The different problems raised in requirements may include incomplete requirements, ambiguous requirements, unstable requirements, inconsistent requirements, incomplete domain analysis, requirements omission, requirements misconception, requirements different understandability, less requirements due to less communications and incorrect requirements [8, 9, 10].

1. Incomplete Requirements

The main cause of this problem is the poor understandability of requirements. The poor system understandability is considered as the one of the most critical issues faced by developers working in the global software development [11, 12]. The poor understandability of requirements is because in GSD environments there is a lack of informal communications during project development between developers and clients.

2. Ambiguous Requirements

The requirements ambiguity may occur due to the unclear system understandability and poor domain understandability [12, 13]. The requirements ambiguity can be removed by writing them in a way in which each requirement may have only one interpretation. The requirements ambiguity may also occur due to the wrong interpretations of the textual communications performed using synchronous or asynchronous methods.

3. Unstable Requirements

This issue occurs due to the frequent requirements change requests caused by poor coordination and understandability [14]. The frequent changes in system requirements affect the project schedules and deadlines. The requirements change request made by clients push the developers to keep changing the product implementations.

4. Inconsistent Requirements

It may occur due to the poor clarity of system caused by poor coordination among key stakeholders [15]. The requirements inconsistency is also considered as one of the main reasons of system changes. The requirements inconsistency may also occur due to the variations in stakeholders of product on client side during requirements communication and negotiations.

5. Incomplete Domain Analysis

This imperative issue may occur when stakeholders are reluctant to reply [12, 13]. The poor participation of stakeholders in project conversations and meetings causes the incomplete domain analysis. The poor accessibility of developers to client organizations creates a major hurdle for them to understand the client domain and organization completely. The main reason is the poor interactions and involvements of key stakeholders in product development process.

6. Requirements Omission

It may occur due to the poor communications among key stakeholders [13, 14]. The poor communication is considered as the one of the most critical issues faced by developers working in the global software development projects. The poor communication of requirements is because in GSD environments there is a lack of informal communications during project development between developers and clients.

7. Requirements Misconception

It may occur due to the unavailability of communication and contact among developers and users [14, 15]. The poor system understandability is considered as the one of the most critical issues, which may also lead to requirements misconception. The poor understandability of requirements is because in GSD environments there is a lack of informal communications during project development between developers and clients.

8. Requirements Different Understandability Primarily this issue may occur due to the language barrier between developers and users [15]. The main source of this issue is the ambiguity in the requirements statements. In GSD projects, there is lack of informal communications, which becomes cause of misunderstanding of written requirements when interpreted by different stakeholders. The physical contact of development teams with client teams enables them to resolve this issue by performing informal communications.

9. Less Requirements

This issue may occur due to the Time zone differences because in such scenarios, developers assume requirements. The poor communication is considered as the one of the main reasons of less requirements gathering. The poor communication of requirements is because in GSD environments there is a lack of informal communications during project development between developers and clients.

10. Incorrect Requirements

It may occur due to the assumptions made by developers during product requirements gathering, system design and implementation. The incorrect requirements refer to the invalid requirements. The invalid requirements become a major cause of not meeting the product scope defined in scope and vision document. The probability of occurrence of incorrect system requirements is more in the development environments where there is lack of informal communications.

V. Conclusion

The requirements elicitation is full of informal and formal communications between developer teams and client teams during product conception stage. The requirements elicitation is affected by the poor communications and interactions appeared in the global software development projects. There are a large number of challenges faced by requirements elicitation teams during global software development projects. The main cause of all these challenges are the traditional GSD constraints of geographical distribution, temporal dispersion, cultural diversity and linguistic diversity. The better management of GSD major constraints improves the requirements elicitation process and its outcomes in working environments where development teams are geographically distributed around the world.

References

[1] Alsahli A. A., & Hameed U. K. (2015) “FreGsd: A framework for global software requirement engineering”. Journal of software, 10 (10), 1189-1198.

[2] Mighetti, J. P., & Hadad, G. D. (2016). A Requirements Engineering Process Adapted to Global Software Development. CLEI Electronic Journal, 19(3), 181-209.

[3] Ali, N., & Lai, R. (2017). A method of requirements elicitation and analysis for Global Software Development. Journal of Software: Evolution and Process, 29(4), e1830.

[4] Ali, N., & Lai, R. (2018). Requirements Engineering in Global Software Development: A Survey Study from the Perspectives of Stakeholders. Journal of Software, 13(10), 520-533.

[5] Carrillo de Gea, J. M., Nicolás, J., Fernández-Alemán, J. L., & Toval, A. (2017). Automated support for reuse-based requirements engineering in global software engineering. Journal of Software: Evolution and Process, 29(8), e1873.

[6] Usmani, N., Hassan, R., & Mahmood, W. (2017). Impediments to Requirement Engineering in Distributed Team. International Journal of Information Engineering and Electronic Business, 9(6), 10.

[7] Yaseen, M., Baseer, S., & Sherin, S. (2015, December). Critical challenges for requirement implementation in context of global software development: A systematic literature review. In 2015 International Conference on Open Source Systems & Technologies (ICOSST) (pp. 120-125). IEEE.

[8] Shafiq, M., Zhang, Q., Akbar, M. A., Khan, A. A., Hussain, S., Amin, F. E., & Soofi, A. A. (2018). Effect of project management in requirements engineering and requirements change management processes for global software development. IEEE Access, 6, 25747-25763.

[9] Khan, H. H., Mahrin, M. N. R. B., & Malik, M. N. (2016). Situational Requirement Engineering Framework for Global Software Development: Formulation and Design. Bahria University Journal of Information & Communication Technologies, 9(1), 74-84.

[10] Nicolás, J., De Gea, J. M. C., Nicolás, B., Fernández-Alemán, J. L., & Toval, A. (2018). On the risks and safeguards for requirements engineering in global software development: systematic literature review and quantitative assessment. IEEE Access, 6, 59628-59656.

[11] Katina, P. F., Keating, C. B., & Ra’ed, M. J. (2014). System requirements engineering in complex situations. Requirements engineering, 19(1), 45-62.

[12] Jamil, Mohsin, Asim Waris, Syed Omer Gilani, Bilal A. Khawaja, Muhammad Nasir Khan, and Ali Raza. "Design of Robust Higher-Order Repetitive Controller Using Phase Lead Compensator." IEEE Access 8 (2020): 30603-30614.

[13] Raza A, Akhtar A, Jamil M, Abbas G, Gilani SO, Yuchao L, Khan MN, Izhar T, Dianguo X, Williams BW. A protection scheme for multi-

terminal VSC-HVDC transmission systems. IEEE Access. 2017 Dec 25;6:3159-66.

[14] Bashir N, Jamil M, Waris A, Khan MN, Malik MH, Butt SI. Design and Development of Experimental Hardware in Loop Model for the Study of Vibration Induced in Tall Structure with Active Control. Indian Journal of Science and Technology. 2016 Jun;9:21.

[15] Jamil M, Arshad R, Rashid U, Ayaz Y, Khan MN. Design and analysis of repetitive controllers for grid connected inverter considering plant bandwidth for interfacing renewable energy sources. In2014 International Conference on Renewable Energy Research and Application (ICRERA) 2014 Oct 19 (pp. 468-473). IEEE.

[16] Khan MN, Jamil M, Gilani SO, Ahmad I, Uzair M, Omer H. Photo detector-based indoor positioning systems variants: A new look. Computers & Electrical Engineering. 2020 May 1;83:106607.

[17] Kashif H, Khan MN, Altalbe A. Hybrid Optical-Radio Transmission System Link Quality: Link Budget Analysis. IEEE Access. 2020 Mar 18;8:65983-92.

[18] Zafar K, Gilani SO, Waris A, Ahmed A, Jamil M, Khan MN, Sohail Kashif A. Skin Lesion Segmentation from Dermoscopic Images Using Convolutional Neural Network. Sensors. 2020 Jan;20(6):1601.

[19] Uzair M, D DONY RO, Jamil M, MAHMOOD KB, Khan MN. A no-reference framework for evaluating video quality streamed through wireless network. Turkish Journal of Electrical Engineering & Computer Sciences. 2019 Sep 18;27(5):3383-99.

[20] Khan MN, Gilani SO, Jamil M, Rafay A, Awais Q, Khawaja BA, Uzair M, Malik AW. Maximizing throughput of hybrid FSO-RF communication system: An algorithm. IEEE Access. 2018 May 25;6:30039-48.

[21] Khan MN, Jamil M, Hussain M. Adaptation of hybrid FSO/RF communication system using puncturing technique. Radioengineering. 2016 Dec 1;25(4):12-9.

[22] Khan MN, Jamil M. Adaptive hybrid free space optical/radio frequency communication system. Telecommunication Systems. 2017 May 1;65(1):117-26.

[23] Iqbal, M. A., Shah, A., & Khan, T. K. (2019). Predicting Most Productive Requirements Elicitation Teams using MBTI Personality Traits Model. International Journal of Engineering and Advanced Technology. Regular Issue, 9(1), 3809-3814. doi:10.35940/ijeat.a9833.109119

[24] Iqbal, M. A., Aldaihani, A. R., & Shah, A. (2019). Big-Five Personality Traits Mapped with Software Development Tasks to Find Most Productive Software Development Teams. International Journal of Innovative Technology and Exploring Engineering. Regular Issue, 8(12), 965-971. doi:10.35940/ijitee.j9755.1081219

[25] Iqbal, M. A., Ammar, F. A., Aldaihani, A. R., Khan, T. K., & Shah, A. (2019). Building Most Effective Requirements Engineering Teams by Evaluating Their Personality Traits Using Big-Five Assessment Model. 2019 IEEE 6th International Conference on Engineering Technologies and Applied Sciences (ICETAS). doi:10.1109/icetas48360.2019.9117561

[26] Iqbal, M. A., Ammar, F. A., Aldaihani, A. R., Khan, T. K., & Shah, A. (2019). Predicting Most Effective Software Development Teams by Mapping MBTI Personality Traits with Software Lifecycle Activities. 2019 IEEE 6th International Conference on Engineering Technologies and Applied Sciences (ICETAS). doi:10.1109/icetas48360.2019.9117370