International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:9, No:1, 2015

The Impacts of Local Decision Making on Customisation Process Speed across Distributed Boundaries: A Case Study

A. M. Qahtani, G. B. Wills, A. M. Gravell

***Abstract*—**Communicating and managing customers' requirements in software development projects play a vital role in the software development process. While it is difficult to do so locally, it is even more difficult to communicate these requirements over distributed boundaries and to convey them to multiple distribution customers. This paper discusses the communication of multiple distribution customers' requirements in the context of customised software products. The main purpose is to understand the challenges of communicating and managing customisation requirements across distributed boundaries. We propose a model for Communicating Customisation Requirements of Multi-Clients in a Distributed Domain (CCRD). Thereafter, we evaluate that model by presenting the findings of a case study conducted with a company with customisation projects for 18 distributed customers. Then, we compare the outputs of the real case process and the outputs of the CCRD model using simulation methods. Our conjecture is that the CCRD model can reduce the challenge of communication requirements over distributed organisational boundaries, and the delay in decision making and in the entire customisation process time.

***Keywords*—**Customisation Software Products, Global Software Engineering, Local Decision Making, Requirement Engineering, Simulation Model.

I. Introduction

THE software industry has shifted its attention to global software engineering. Nevertheless, numerous challenges have arisen. In order to meet the challenges associated with global adoption, changes in software engineering practices are needed. Requirements engineering (RE) serves a very important role in the software development process in both collocated and distributed domains [1]. Managing the communication of customer requirements is a key component of the development production process for the marketplace.

While it is difficult to negotiate and communicate these requirements locally, it is even more difficult to communicate them over distributed boundaries, particularly to multiple customers. This difficulty increases in distributed software development (DSD) projects as well as projects that have multiple distributed customers across organisational and cultural boundaries. In the last two decades, a significant transition from co-located forms of development to global software development has taken place, requiring more communication across organizational boundaries [1], [2]. Requirement engineering in a distributed domain is a complex intersection phenomenon that encompasses numerous technical, social, and organisational aspects [3].

In recent years, the amount of research conducted in the fields of requirement engineering and requirement management have increased [4]. Different aspects of requirement engineering in both distributed and global development environments have been examined in order to identify challenges and propose solutions [1]. Meanwhile, customised software has become commonplace in the software industry, particularly due to the boom in outsourcing software and the offshore development process for many software development vendors [5].

This research aims to identify the requirements of multiple distributed customers in the context of customised software products. To achieve this goal, we began by reviewing the previous research on requirements engineering in the development and outsourcing process in a DSD setting. We propose a model for Communicating Customisation Requirements of Multi-Clients in a Distributed Domain (CCRD). Thereafter, we evaluate that model by presenting the results of a case study of a company with customisation projects for 18 distributed customers. Next, we compare the outputs of the real case process and the outputs of the CCRD model using simulation methods.

We believe that managing customisation requirements in the DSD context is important, and local negotiation of customers' requirements has seen successes in many agile development projects in the DSD domain. In addition, our conjecture is that CCRD model can reduce the challenge of communication requirements over distributed organisational boundaries, and reduce the delay in decision making and in the entire customisation process time.

II. Related Work

Global software development has been of significant interest in the last several years, as it aids software development projects in overcoming certain difficulties associated with recruiting qualified practitioners in their projects. Furthermore, it benefits development projects in terms of cost and productivity [6]. However, many challenges have arisen with the shift to distributed development, such as adequate communication and issues related to coordination

A.M. Qahtani is with School of Electronics and Computer Science, University of Southampton, Southampton, United Kingdom (corresponding author e-mail: amq1u10@ecs.soton.ac.uk).

G.B. Wills and A.M. Gravell are with School of Electronics and Computer Science, University of Southampton, Southampton, United Kingdom (e-mail: gbw@ecs.soton.ac.uk, gbw@ecs.soton.ac.uk).

between distributed teams [7].

Requirement engineering is an important area in the software development process, which is affected by communication challenges in a distributed domain [8]. The main challenge lies in accommodating the needs of distributed stakeholders for either development or customised software. Many researches have discussed different aspects of that challenge and have proposed solutions for requirements engineering in a distributed domain. Hayat et al. [9] proposed a model to manage requirement changes during the development process. That model has since been extended to serve as a framework for global development projects, called RCM, by Khan et al. [10]. The RCM framework emphasizes knowledge sharing by adding a central repository in order to increase awareness between distributed teams, thereby reducing the need for communication across distributed boundaries. In terms of global software development projects addressing the challenges of communication across different cultures, Damian [2] discussed the challenges of global development projects across various organizational and cultural boundaries. Moreover, this research explores how knowledge acquisition and sharing between developers and stakeholders can help requirements engineering in the GSD domain.

On the other hand, several researchers have discussed and taken an interest in managing and communicating customers' requirements in a distributed domain. Damian and Zowghi [1] investigated the impact of multi-distributed customers on requirement management and communication and proposed a model addressing several factors that affect requirements management in distributed development projects, such as remote communication and knowledge management, cultural diversity and time differences. Other researchers have emphasised the most frequent challenge of cross-sites problems, namely, a delay in resolving customers' issues. This delay means that customers' requirements take up additional time in communications between distributed teams [11]. Gopal et al. [12] conducted a study on the impacts of coordination and communication of requirements across the global software process. They emphasised the fact that speed and productivity are the consequences of the distributed boundaries, and they found that local negotiation of customers' needs and requirements reduce the delay in resolving time.

III. Background and Problem Area

This research discusses communication requirements, as this issue is significant in the customisation process of distributed development projects. Our research target is software vendor organisations, which customise software products for different distributed customers. They then appoint representative customisation teams to install the software and deal with customisation requests and changes (Fig. 1). The problem is how to reduce the challenges of communicating customisation requirements between a software vendor and customers' locations; also, to reduce the implications of those issues on the customisation process speed. This paper also presents a model for the customisation process of distributed customers' requirements, which enhanced the concept of locality for making decisions at customers' locations (i.e., the CCRD model, Fig. 3).

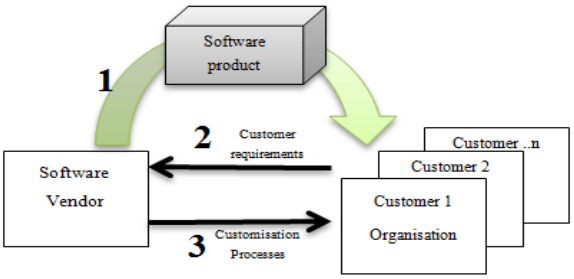


Fig. 1 Distributed customization software domain

A. Research Objectives and Questions

The objectives of this research are:

- Discuss customization requirements in the context of DSD across distributed boundaries.
- Applying the concept of locality on decision making for the customisation process by presenting the CCRD model, and evaluating it to see how much that would impact the productivity and speed of the customisation process.

To achieve the above objectives, the following research question is formulated:

RQ1. What are the impacts of making decisions locally on the total decision making time, and the total customisation time of the software customisation process in a distributed domain?

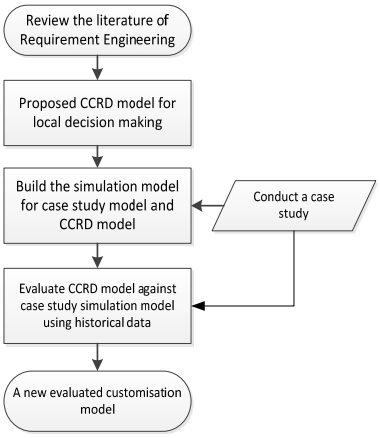


Fig. 2 Flow chart of the research method

A. Research Methodology

The methods used in this research (as shown in Fig. 2) started by reviewing the literature in order to investigate the success of locality in software development and how that applied in some approaches, such as the Agile development method. Then, customisation model was designed, which located the decision making at customers' locations and the rest of the customisation process at the software vendors' distributed locations (Fig. 3). Afterward, in order to evaluate

the proposed model (CCRD model), we conducted a case study of a company that has 18 distributed customers using one software product, as the company deals with their customisation requirements across distributed boundaries. The customisation scenario model of the selected company and the CCRD model have been transformed into simulation models in order to evaluate the impacts of local decision making on both models, and to examine how much the CCRD model improves the productivity and speed of the customisation process and overcomes the distribution challenges.

IV. Communicating Customization Requirements Model (CCRD)

The CCRD model was designed to model the communication of customisation requirements of distributed clients. This model was designed to enhance local decision making in order to overcome the challenges of communicating clients' requirements across multiple sites. The designed model relies on the main practices of a software development life cycle for the customisation process, starting from collecting clients' requirements, to resolving and delivering them. Also, it relies on theories which emphasize the benefits of local negotiation and decision making and on distributed development in the productivity and speed of the software development process in distributed domains [12]. Through the CCRD model, this research aims to investigate the impacts of decision making on clients' customization requirements at clients' locations, in order to reduce the challenges of communicating customization requirements for multi-distributed clients.

This scenario of this CCRD model aims to provide a mechanism to manage and communicate customisation requirements for multi-clients across distributed boundaries, and shows the importance of communication and coordination among the three main groups in software development projects in a distributed domain, which are the client, the onshore customisation team (at the client's location) and the offshore customisation team (at the vendor location), in order to deal with clients' customisation requirements [13]. This model enhances the practice of local decision making at clients' locations in order to overcome the challenges of communicating and coordinating clients' requirements in a distributed domain and to reduce the implications of these challenges.

The design of this model, as mentioned before, relies on the main software development life cycle process and practices such as requirement gathering and analysis, design, implementation and testing [14], and the benefits and features of a collocated team and the successful experience of agile software development through local negotiation and decision making at the client's location [15]. Thus, the CCRD model components are designed to start with the requirements analysis process. In this model, the collection of customisation requirements occurs at the client's location. Then, the development process includes design, implementation, testing and evaluation at the offshore site of the software vendor. All these components come to work in the designed model as follows:

The flow of the customisation process starts from collecting clients' requirements. The customisation requirements in this case come as two types. The type first is bugs, which represent the problems of the working system. This requirement demands investigation to find the issues and debug them. The second type of clients' requirements is new features (when clients request new features that need to be added to the system). This need requires analysis and development in most cases. Both types require some communications between clients and software vendors in order to understand their collective requirements. In typical offshore projects, there is a small vendor team located in a client's location to undertake these activities and act as a proxy for the main software vendor [16]. The second step is making a decision and then negotiating on the request. The decision takes three forms:

- Cancellation: This form of decision happens when the client changes his mind about the requests before or after some discussions regarding his requests.
- Rejection: The rejection decision is made by a decision maker after investigation of that request. It is rejected because it is outside the project scope, it would create more implications for the project, or the request does not make sense.
- Acceptance: Acceptance happens when the initial investigation has been conducted to make sure the request make sense, and accounts for bugs or new features. Then, it is sent to the central customisation team in order to apply the required development or debug any issues.

After that, the request goes through a check with the offshore customisation team at the vendor's location to make sure the requests are understood and scheduled into the team tasks. If the request does not exist, in the case of bugs, or if the new features requested are not clear, it returns to the client's location for more investigation and another decision. The last process in the customisation life cycle, in which the requests go through it, is the development process, which includes implementation, testing and verifying, and delivery of the completed requests to client's location. It is worth noting that the offshore team at the vendor's location deals with multi-distributed clients at the same time, which makes communication and coordination more challenging. However, most communication in this type of project happens at the level of decision making and negotiation, which is moved in this model to the client's location in order to reduce those challenges.

Communications is one of the main challenges of distributed development projects, and the customisation process for multi-clients involves struggling with that challenge. On the other hand, locality in some software development and management processes has achieved success in different development approaches and contexts. This model adopts the concept of local decision making in the customisation process for multi-clients in order to provide a model for communicating customisation requirements for different clients and overcoming the challenges of distributed domains.

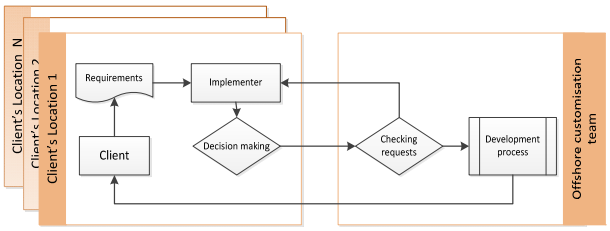


Fig. 3 CCRD model — local decision making

V. Case Study Design and Evaluation Method

A. Case Study

The case study is suitable methodology for software engineering research, as it studies different phenomena in their natural context [17]. Therefore, this research used a real case study of the customisation process in order to build and validate a simulation model for that case, and then used it as a baseline in the evaluation process for this research. To this end of building a baseline model of the selected company, the selected company was working in customisation software and dealing with customers' requirements in a distributed domain. The selected company has significant experience in customisation software products for distributed customers. The company has developed a variety of software solutions for different sectors, such as academic systems, healthcare systems and business intelligence systems. These products have been developed and customised based on customers' needs.

In this study, we have collected the historical data of 18 customers, which contains 2,479 customisation requirements observed in 1,290 working hours. In addition, the collected data includes the processes applied on customers' requests starting from arrival time, decision making time and development. Also, the decisions made on those requirements are studied. All this information is used as trace data to drive the design of a simulation of the case study in order to use it as a baseline model [18]. Furthermore, the arrival requirements are used as input for both CCRD and baseline simulation models in the evaluation process.

B. Baseline Simulation Model

This section describes the importance of the modelling and simulation process in research and evaluation studies, and how it reduces the effects of conducting experiments on real cases. According to Robinson (2007), simulation is defined as, "Experimentation with a simplified imitation (on a computer) of an operations system as it progresses through time, for the purpose of better understanding and/or improving that system." Modelling and simulation become very popular methods in research and industry for many subjects such as engineering, business and medical science. However, the software development field has increasingly used modelling and simulation to support the process of software development and software project management [19]. In addition, Kellner et al. [19] state that software process simulations are currently being used for different issues such as strategic software development, process improvement, and control and operational management of software engineering.

In this research, simulation has been selected as the evaluation method for the research hypothesis. According to Abdel-Hamid [20], although it is easy to propose a hypothesis in software engineering, it is very difficult to test it. There are many reasons for the difficulty in testing a software engineering hypothesis. Applying and controlling software engineering experiments in the real world demand cost and time [21]. The challenge and difficulty increase in large, complex and dynamic projects like distributed development projects [22]. Therefore, simulation models enable researchers to control software engineering experiments, and also allow them to identify the factors that impact the outputs of the simulation model with less cost and time [21]. In terms of the evaluation software development process, Martin & Ra [23] state that changes in the software development process can be evaluated by simulation models. However, that evaluation should consider the context of the project environment.

We used Simul8 as a software simulation package, which validates and is used in many simulation projects [24]. In terms of the simulation model settings and the fitting of activity distribution, all activities are set and fit based on the collected empirical data of the real system. We applied the Black Box validation comparison method, which examines the validity of the simulation model of real system by using the same inputs of the real system and compares the outputs by using the Wilcoxon signed-rank test. All results of the comparison between both simulation models were not significantly different, which means that the simulation model is valid enough. Table I shows the results of the Wilcoxon test for comparing real system outputs and simulation model outputs. It examines all activity in order to test the validity of the baseline simulation model of a real system. The results showed there is no significant difference between real system outputs and the baseline simulation model, as all P values are greater than 0.05 at the 95% level of confidence, and the mean of the proceed requirements through these activates is closer in both models. Fig. 3 illustrates the simulation of the CCRD model, which has allocated decision making teams at customers' locations. In that model, holding and rejection activities have been removed as there is no need for them in the CCRD model.

The main goal of these activates in the baseline model was for negotiation and discussion with distributed customers in order to make an accurate decision, which is done in the CCRD model face-to-face by the local decision making team.

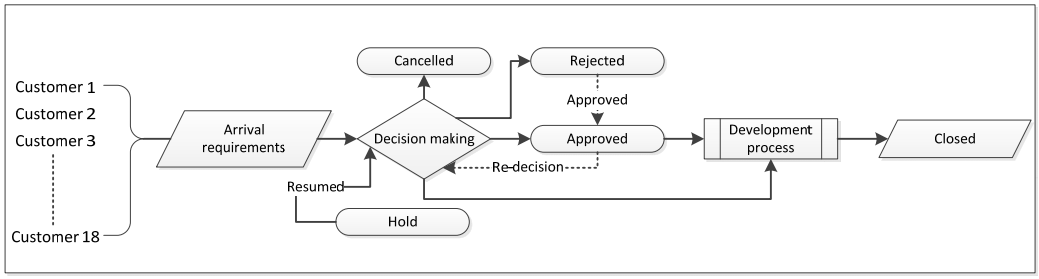


Fig. 4 Baseline model for the real case study

TABLE I
RESULTS OF WILCOXON TEST FOR REAL SYSTEM OUTPUTS VS. SIMULATION OUTPUTS

Activity	Model	*Mean*	*T*	*P*
Decision process	Real system	100.05	7	0.167
	Simulation	116.40	13	
Holding process	Real system	0.92	7	0.953
	Simulation	1.00	8	
Rejection process	Real system	1.20	5	0.395
	Simulation	1.60	2	
Approval process	Real system	58.50	8	1.00
	Simulation	63.31	16	
Development process	Real system	124.53	6	0.449
	Simulation	124.53	10	

VI. EVALUATION RESULTS

The evaluation experiments in this study have been designed in order to examine the locality in the CCRD model and how much applying that concept would reduce the implications of communication across distributed boundaries by reducing customisation time and decision making time.

A. CCRD Simulation Model

The evaluation experiments used a Black-Box comparison approach between outputs of the CCRD simulation model and the baseline simulation model (Fig. 5). This experiment was run on three different groups of customer requirements. The first customer group had up to 50 requests during the observation time (1,290 working hours). The second group had seven customers, each of them having between 51 and 100 requests in the observation time. The final group had five customers with 101 or more requests. The arrival time of the three groups' requirements was inserted into both simulations and the outputs were compared using the Wilcoxon signed-rank test.

Fig. 6 illustrates the comparison between outputs of simulations of the real-world model and the CCRD model. Fig. 6 (a) shows the difference in the mean of decision making time between the baseline model and CCRD model for all three groups of requirements, which indicates a significant reduction in the decision time from more than two hours to less than 30 minutes. Fig. 6 (b) illustrates the impact of local decision making on the entire customisation process mean time for the three groups, which were between 2.52 and 2.79 in the baseline model, and decreased to between 1.13 and 1.65 in the CCRD model.

In terms of statistical results of the experiment, Table II displays the inferential results of the Wilcoxon signed-rank test, which refers to the significant difference for all results when P values are less than 0.05 at 95% confidence level. In this case, the null hypothesis rejected and the alternative hypothesis accepted the difference between the simulation model of baseline and CCRD outputs in decision making time and in the entire customization process time.

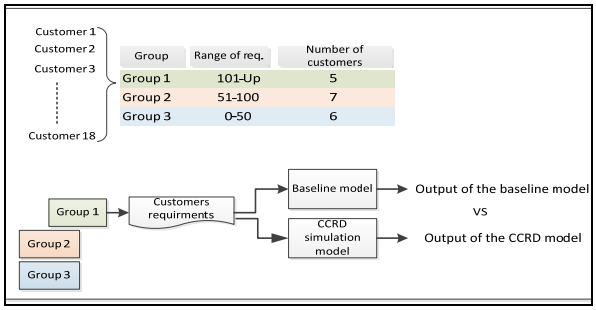
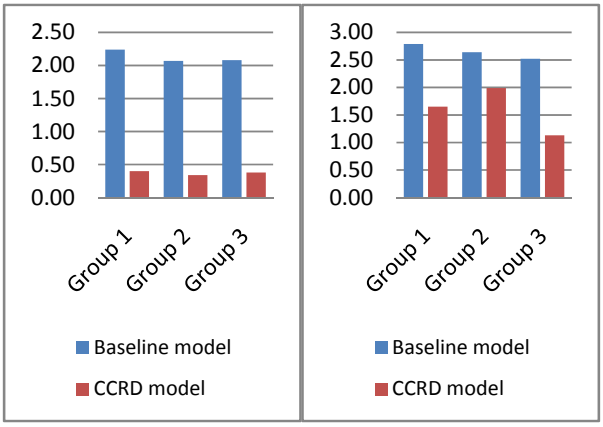


Fig. 5 Evaluation of experiment design using the Black-Box comparison approach



(a) (b)

Fig. 6 Comparing results of baseline and CCRD models

VII. DISCUSSION

Results presented in the previous section indicate that making decisions for customisation requirements at customers' locations in distributed domains reduces the time need to make decisions, and therefore it reduces the entire

customisation process time. This reduction refers to decision making which includes requirements for negotiation to overcome the challenges of communicating customisation requirements across distributed domains, as most communications in the customisation process involve discussing and understanding customers' requirements [12]. It was eliminated in the proposed model (CCRD model) by discussing all issues face-to-face at customers' locations.

TABLE II A
STATISTICAL RESULTS OF EVALUATION EXPERIMENTS OF THE CCRD MODEL

Mean time of decision making		Baseline model		CCRD model		*P* value
		Mean	Std. Dev	Mean	Std. Dev	
	Group1	2.24	5.7	0.4	0.32	0.001
	Group2	2.07	5.45	0.34	0.33	0.001
	Group3	2.08	5.53	0.38	0.37	0.001

TABLE II B
STATISTICAL RESULTS OF EVALUATION EXPERIMENTS OF THE CCRD MODEL

Mean time of Entire process		Baseline model		CCRD model		*P* value
		Mean	Std. Dev	Mean	Std. Dev	
	Group1	2.79	5.69	1.65	2.14	0.001
	Group2	2.64	5.51	1.99	2.87	0.028
	Group3	2.52	5.62	1.13	1.3	0.001

VIII. CONCLUSION AND FUTURE WORK

This study aims to discuss the benefits of locality in reducing the implications of communication across distributed boundaries. It proposed a customisation model for multiple customers, which enhances the locality concept by locating decision making at customers' locations. Furthermore, this study has evaluated the CCRD model by using a real case study from a company that customised software for 18 distributed customers. The evaluation results refer to the significant difference in reducing the mean time of decision making and the entire customisation process time for the CCRD model.

The future purpose of this research is to examine locality on the development process. In addition, using local decision making in distributed projects gives rise to the issue of awareness between distributed teams, so it is useful to investigate the knowledge and awareness of management when decisions are taken locally.

REFERENCES

[1] D. Damian and D. Zowghi, "Requirements Engineering challenges in multi-site software development organizations," Requir. Eng. J., vol. 8, pp. 149–160, 2003.
[2] D. Damian, "Stakeholders in Global Requirements Engineering : from Practice," IEEE Softw., vol. 24, no. 2, pp. 21–27, 2007.
[3] D. E. Damian, "The study of requirements engineering in global software development: as challenging as important," in International Workshop on Global Software Development, 2002.
[4] J. R. Jiao and C. Chen, "Customer Requirement Management in Product Development: A Review of Research Issues," Concurr. Eng. Res. adn Appl., vol. 14, no. 3, pp. 1–25, 2006.
[5] A. M. Qahtani, G. B. Wills, and A. M. Gravell, "Customising software products in distributed software development A model for allocating customisation requirements across organisational boundaries," in International Conference on Information Society, i-Society 2013, 2013, pp. 92–98.
[6] F. Q. B. da Silva, C. Costa, a. C. C. Franca, and R. Prikladinicki, "Challenges and Solutions in Distributed Software Development Project Management: A Systematic Literature Review," 2010 5th IEEE Int. Conf. Glob. Softw. Eng., pp. 87–96, Aug. 2010.
[7] M. Jiménez, M. Piattini, and A. Vizcaíno, "Challenges and Improvements in Distributed Software Development: A Systematic Review," Adv. Softw. Eng., vol. 2009, pp. 1–14, 2009.
[8] D. Damian, S. Marczak, and I. Kwan, "Practice: Requirements Engineering in Global Teams," in Global Software and IT A Guide to Distributed Development, projects, and OUtsourcing, First Edit., John Wiley & Sons, Inc., 2012, pp. 257–267.
[9] F. Hayat, N. Ehsan, A. Ishaque, S. Ahmed, and E. Mirza, "A methodology to manage the changing requirements of a software project," 2010 Int. Conf. Comput. Inf. Syst. Ind. Manag. Appl., pp. 319–322, Oct. 2010.
[10] A. A. Khan, S. Basri, and P. D. D. Dominic, "A propose framework for requirement Change Management in Global Software Development," 2012 Int. Conf. Comput. Inf. Sci., pp. 944–947, Jun. 2012.
[11] J. D. Herbsleb, A. Mockus, T. A. Finholt, and R. E. Grinter, "An Empirical Study of Global Software Development: Distance and Speed," pp. 81–90, 2001.
[12] A. Gopal, J. A. Espinosa, S. Gosain, and D. P. Darcy, "Coordination and Performance in Global Software Service Delivery: The Vendor's Perspective," vol. 58, no. 4, pp. 772–785, 2011.
[13] J. Espinosa, S. Slaughter, R. Kraut, and J. Herbsleb, "Team Knowledge and Coordination in Geographically Distributed Software Development," Journal of Management Information Systems, vol. 24. pp. 135–169, 2007.
[14] I. Sommerville, Software Engineering, 6th ed. Essex: Pearson Education Limited, 2001.
[15] K. Sureshchandra and J. Shrinivasavadhani, "Adopting Agile in Distributed Development," 2008 IEEE Int. Conf. Glob. Softw. Eng., pp. 217–221, Aug. 2008.
[16] K. V. P. Y. S. Gopalakrishnan S., "Offshore model for software development: the infosys experience," in Proceedings of the ACM SIGCPR Conference, 1996, pp. 392–393.
[17] P. Runeson and M. Höst, "Guidelines for conducting and reporting case study research in software engineering," Empir. Softw. Eng., vol. 14, no. 2, pp. 131–164, Dec. 2008.
[18] A. M. Law, Simulation Modeling and Analysis, Fourth Edi. New York: McGraw-Hill, 2007.
[19] M. I. Kellner, R. J. Madachy, and D. M. Ra, "Software process simulation modeling: Why? What? How?," J. Syst. Softw., vol. 46, pp. 91–105, 1999.
[20] T. K. Abdel-Hamid, "The Economics of Software Quality Assurance: A Simulation-Based Case Study," MIS Q., vol. 12, pp. 395–411, 1988.
[21] S. Setamanit, W. Wakeland, and D. Raffo, "Using Simulation to Evaluate Global Software Development Task Allocation Strategies," Softw. Process Improv. Pract., vol. 12, no. May, pp. 491–503, 2007.
[22] R. L. Glass, "Modern Programming practices: A Report from industry," Englewood Cliffs, 1982.
[23] R. Martin and D. Ra, "Application of a hybrid process simulation model to a software development project," vol. 59, 2001.
[24] K. H. Concannon, K. I. Hunter, and J. M. Tremble, "SIMUL8-Planner simulation-based planning and scheduling," Proc. 2003 Int. Conf. Mach. Learn. Cybern. (IEEE Cat. No.03EX693), pp. 1488–1493, 2003.