Software Architecture in Distributed Software Development: A Review

Alok Mishra[1] and Deepti Mishra[2]

[1] Department of Software Engineering, Atilim University, Ankara, Turkey
alok@atilim.edu.tr
[2] Department of Computer Engineering, Atilim University, Ankara, Turkey
deepti@atilim.edu.tr

Abstract. This paper presents a literature review of distributed software development (DSD) or global software development (GSD) and software architecture. The main focus is to highlight the current researches, observations, as well as practice directions in these areas. The results have been limited to peer-reviewed conference papers and journal articles, and analysis reports that major studies have been performed in software architecture and global software development, while the empirical studies of interfacing distributed/global software development and software architecture has only received very little attention among researchers up to now. This indicates the need for future research in these areas.

Keywords: Software Architecture**,** Global Software Development, Distributed Software Development, Knowledge Management.

1 Introduction

Global software development is becoming a widely accepted practice in software industry [35]. The advent of the Internet has supported Global Software Development (GSD) by introducing new concepts and opportunities, resulting in benefits such as scalability, flexibility, interdependence, reduced cost, resource pools, and usage tracking. GSD brings challenges to distributed software development activities due to geographic, cultural, linguistic, and temporal distance among project development teams [18]. The number of organizations distributing their software development processes worldwide to attain increased profit and productivity as well as cost reduction and quality improvement is growing [24]. Ensuring quality issues in distributed software development projects is a significant issue [23, 26].

The popularity of GSD is expected to continue growing for many reasons, such as reducing cost, improving quality, shortage of skilled people, and improving time-to-market [5]. Ali et al. [2] further argue that software development companies are discovering new ways of leveraging software development resources that are geographically dispersed and, therefore, there is an increasing need to identify and understand mechanisms for scaling the processes and practices of traditional software development to meet the requirements of GSD.

Y.T. Demey and H. Panetto (Eds.): OTM 2013 Workshops, LNCS 8186, pp. 284–291, 2013.

Software Architecture is a recognized and indispensable part of system development defined in terms of components, connectors, and is usually represented in different views, where each view shows certain concerns to stakeholders [33]. Architecture serves as a blueprint for developers, and it constitutes an abstract description of the application which serves as a basis for verifiable architecture rules. Architecture-based development, thus, facilitates communication by improving comprehension through one common object of work that all participants use and understand [21]. Software architecture can be used to reduce the need for communication in a multi-site development project as the gross structure of the system, the software architecture, can be used to divide work amongst sites [33].

According to Yildiz et al. [34] notably, architecting in GSD has not been widely addressed and key research focus in the GSD seems to have been in specific related to tackling the problems related to communication, coordination and control concerns.

A review of the literature in distributed or global software development shows that little deliberation has been paid on the software architecting process and software architecture as an artifact in the context of distributed software development. Therefore, the motivation of this paper is to provide summary of literature encompassing software architecture in distributed software development.

The paper is organized as follows. The next section presents major research studies in software architecture related with GSD/DSD areas. Finally, section 3 concludes with observations and limitations of the study.

2 Major Studies in Software Architecture in DSD

Distributed software development usually consists of several sites, on which different teams are working to develop a component of a software project. Distributed software development projects have to manage different challenges in different domains including software architecture, eliciting and communicating requirements, setting up suitable environments and tools, and composition of the information system project.

2.1 Software Architecture and DSD

A software architecture design includes components and interfaces [28], which connect different components [2]. A software architecture drives the structure of an organization [4, 15, 16, 20, 31] and is used as a means for coordinating projects [4, 20]. It is used as a coordination mechanism in distributed organizations to allocate tasks and coordinate the distributed teams [16, 19].

An extremely important kind of knowledge that needs to be shared is that concerning the software architecture, where, the global structure of the system to be built is decided upon. This structure, among others, should capture the major architectural decisions that led to it [10]. Capturing these architectural decisions facilitates a better decision-making process in a shorter time, reducing rework and improving the quality of the architecture [7]. In the context of GSD, sound management of architectural knowledge can help overcome the challenges innate to

GSD. Architectural knowledge management can be implemented by performing a series of well-defined practices [14].

Software architecting is a knowledge-intensive process, involving many different stakeholders and, as the size and complexity of system increases, more people get involved, and architecting turns into collaboration [33]. Sharing architectural knowledge is crucial, in particular for reusing best practices, obtaining a more transparent decision-making process, providing traceability between design artifacts, and recalling past decisions and their rationale [33]. All the challenges that face GSD have to do with different forms of distance: temporal distance, geographical distance, and socio-cultural distance [33]. These challenges have to do with [1]:

- Communication and collaboration among team members
- Tasks coordination, and
- Work supervision

Table 1 shows major studies performed in various areas of DSD software architecture during the last decade according to the review performed.

Table 1. Major Relevant papers in various areas of software architecture and GSD/DSD

Software Architecture and GSD	Major Relevant Studies
Knowledge Management	[2], [7], [13], [14], [17], [33]
Process and Quality	[2], [8], [12], [13], [17], [22], [30], [31], [32]
Framework and Tool Support	[5], [6], [27]

2.2 Software Architecture Knowledge Management and Rules

Clerc et al. [15] reported on the use of the so-called architectural rules to handle GSD concerns. Architectural rules are defined as "principles and statements about the software architecture that must be complied with throughout the organization". They have defined four challenges in GSD: time difference and geographical distance, culture, team communication and collaboration, and work distribution.

Part of overcoming these challenges has to do with adequate knowledge management, including the management of architectural knowledge [33]. Important information regarding architecture can be shared among sites in a GSD project, and also the architecture itself may serve as a kind of a proxy for communication, coordination, and control tasks. For instance, less informal meetings are required when all is clearly documented in the architecture documentation, making the coordination of work a lot easier [33].

Nour Ali et al. [2] presented a review of architectural knowledge management in GSD, and suggested that architectural styles and design decisions are important inputs for defining coordination strategies in an organization, while they have to be carefully selected when developing interfaces of components. However, their review is limited to practices associated in a global software environment by identifying the various

constructs and their relationships by summarizing these relationships in a meta model and showing constructs inter-relations. Clerc et al. [14] concluded that architectural knowledge management practices that promote decentralization get much more attention that those promoting centralization at the agile GSD organization.

Rocha de Faria and Adler [30] showed that architecture-centric strategies can soften the impacts of physical distances. They argued that despite cultural barriers, GSD can bring advantages to the business, as long as an organization knows how to coordinate its distributed processes, requiring every person involved to be engaged in the effort - just like any conventional process improvement program - with an established architecture supporting and orienting all the activities. However, the benefits of gaining architectural knowledge that focuses on architecting as a decision-making process (i.e, assumptions, alternatives, and rationale) have not yet been widely accepted in GSD [13]. For example, performance is an important quality criterion, which demands to be addressed by architectural knowledge (according to the auditors, and acknowledged by Bachmann and Bass [8]).

In this respect, Hashmi et al. [18] have proposed cloud paradigm to meet the different challenges posed by GSD. This will result in GSD benefitting from the cloud's infrastructure, platform, and provision of software as a service feature, in which information and data on the cloud is transmitted and shared by means of web services which, in turn work on the underlying Service Oriented Architecture (SOA) principle. They further suggested that cloud can facilitate GSD both as a process and a product. The former could have implications for the GSD business model, in which service providers are organizations, and their services are parts of a GSD process; for instance, requirements, design, coding, and testing. SOA as a product is developed, run, and distributed globally [18].

2.3 Process and Quality

A global software process, based on a well-defined architecture, grants all team members a common language to define tasks and activities, allowing a better understanding of the business domain terms and project milestones in spite of their differences in terms of cultural and organizational settings [32].

Nord et al. [29] proposed a structured approach for reviewing architecture documentation (AD) which also builds on the stakeholders of the artifact and aims to engage and guide them to assure that the architecture documentation meets their quality concerns.

Non-functional requirements (quality issues) have to be satisfied in a software architecture and, once a high-level architecture is designed, the non-functional requirements should become stable [13, 30, 31]. An architect has to collaborate in the requirements elicitation phase to understand and model the non-functional requirements [2].

The distributed nature of software development and the related communication problem make tools and analysis that address these issues necessary to improve software quality and evolution. Del Rosso [17] has studied Social Network Analysis (SNA) by using the Apache web server, and found that metrics such as affiliation

networks, centrality, betweenness, density, cohesion, communities, and brokerage roles allow a software architect to understand cohesion, communities, and communication in large and distributed software development teams [17].

Clerc [13] investigate the relationship between the number of distributed teams and the usefulness of architecture knowledge coordination strategies and practices. They further suggested that a high-level architecture design should be defined through architects from different sites meeting at collocated face-to-face meetings. A practice used for improving coordination is that each distributed team can include an architect. One of the architects from the distributed teams can be selected to be the leading architect from the site close to the customer [12] or at the headquarters of the organization [22].

2.4 Framework and Tool Support

Babar [5] has presented a framework for a groupware-supported software architecture evaluation process, which does not require the simultaneous participation of all the stakeholders; nor do stakeholders need to be physically collocated. This framework can be helpful in software assessment in distributed software development.

Avritzer et al. [3, 4] presented their experience report of the assessment of the coordination implications of software architecture in a global software development project. They observed that in GSD projects, it is especially important to minimize the need for communication among teams that are not collocated, and to maximize such communication within a local team instead.

According to Babar [6] software architecture evaluation is usually performed in a face to face meeting. Collocating large numbers of stakeholders is difficult and expensive to organize, particularly in the context of GSD projects and he proposed a groupware-supported software architecture evaluation process. Ovaska et al. [27] found in their studies that architecture could be used to coordinate distributed development, requiring that the chief architect be capable of maintaining the integrity of the architecture and of communication for all stakeholders.

3 Conclusions

This paper has presented brief summary of the current research themes related with software architecture in the global/distributed software development. There are still very few studies, including empirical ones, which focus in detail on software architecture and distributed or global development issues. The analysis revealed that most research has been done in software process and quality, knowledge management and framework and tool support areas while empirical industrial case studies, designing and modeling of software architecture for global/distributed software development, comparative case studies of different organizations, visualization, security, trust, standards and deployment issues could get too little attention in global/distributed information system development [25].

In near future, the authors intend to extend this literature review to include further dimensions and studies related with architecture and global software development.

Various case and industrial experience report may be included to enrich architecture issues related with GSD. As future work systematic review in this issue with available databases can also be carried out in this area.

References

1. Agerfalk, P., Fitzgerald, B.: Flexible and Distributed Software Processes: Old Petunias in New Bowls? Commun. ACM 49(10), 27–34 (2006)
2. Ali, N., Beecham, S., Mistrik, I.: Architectural Knowledge Management in Global Software Development: A Review. In: 2010 5th IEEE International Conference on Global Software Engineering (ICGSE), pp. 347–352 (2010)
3. Avritzer, A., Paulish, D., Cai, Y., Sethi, K.: Coordination implications of software architecture in a global software development project. J. Syst. Softw. 83(10), 1881–1895 (2010)
4. Avritzer, A., Paulish, D., Yuanfang, C.: Coordination implications of software architecture in a global software development project. In: Seventh Working IEEE/IFIP Conference on Software Architecture (WICSA 2008), pp. 107–116 (2008)
5. Babar, M.A.: A framework for groupware-supported software architecture evaluation process in global software development. J. Softw. Evol. and Proc. 24, 207–229 (2012)
6. Babar, M.A.: A Framework for Supporting the Software Architecture Evaluation Process in Global Software Development. In: Proceedings of the 2009 Fourth IEEE International Conference on Global Software Engineering (ICGSE 2009), pp. 93–102. IEEE Computer Society, Washington, DC (2009)
7. Babar, M.A., de Boer, R.C., Dingsøyr, T., Farenhorst, R.: Architectural Knowledge Management Strategies: Approaches in Research and Industry. In: Second ICSE Workshop on SHAring and Reusing Architectural Knowledge - Architecture, Rationale, and Design Intent 2007 (SHARK ADI 2007). IEEE Computer Society, Minneapolis (2007)
8. Bachmann, F., Bass, L.: Introduction to the Attribute Driven Design Method. In: 23rd International Conference on Software Engineering (ICSE 2001), pp. 745–746. IEEE Computer Society, Toronto (2001)
9. Baldonado, M., Chang, C.-C.K., Gravano, L., Paepcke, A.: The Stanford Digital Library Metadata Architecture. Int. J. Digit. Libr. 1, 108–121 (1997)
10. Bass, L., Clements, P., Kazman, R.: Software Architecture in Practice, 2nd edn. SEI Series in Software Engineering. Addison-Wesley Pearson Education, Boston (2003)
11. Bruce, K.B., Cardelli, L., Pierce, B.C.: Comparing Object Encodings. In: Ito, T., Abadi, M. (eds.) TACS 1997. LNCS, vol. 1281, pp. 415–438. Springer, Heidelberg (1997)
12. Caprihan, G.: Managing software performance in the globally distributed software paradigm in global software engineering. In: International Conference on Global Software Engineering (ICGSE 2006), pp. 83–91 (2006)
13. Clerc, V.: Do Architectural Knowledge Product Measures Make a Difference in GSD? In: 2009 Fourth IEEE International Conference on Global Software Engineering (ICGSE), pp. 382–387. IEEE Computer Society (2009)
14. Clerc, V., Lago, P., van Vliet, H.: Architectural Knowledge Management Practices in Agile Global Software Development. In: Proceedings of the 2011 IEEE Sixth International Conference on Global Software Engineering Workshop (ICGSE-W 2011), pp. 1–8. IEEE Computer Society, Washington, DC (2011)

15. Clerc, V., Lago, P., Van Vliet, H.: Global Software Development: Are Architectural Rules the Answer? In: Second IEEE International Conference on Global Software Engineering (ICGSE 2007), August 27-30, pp. 225–234 (2007)
16. Clerc, V., Lago, P., Van Vliet, P.: Assessing a Multi-Site Development Organization for Architectural Compliance. In: Sixth Working IEEE/IFIP Conference on Software Architecture. IEEE Computer Society (2007)
17. Del Rosso, C.: Comprehend and analyze knowledge networks to improve software evolution. J. Softw. Maint. Evol.: Res. Pract. 21, 189–215 (2009)
18. Hashmi, S.I., Clerc, V., Razavian, M., Manteli, C., Tamburri, D.A., Lago, P., Di Nitto, E., Richardson, I.: Using the Cloud to Facilitate Global Software Development Challenges. In: Proceedings of the 2011 IEEE Sixth International Conference on Global Software Engineering Workshop (ICGSE-W 2011), pp. 70–77. IEEE Computer Society, Washington, DC (2011)
19. Herbsleb, J.D., Grinter, R.E.: Architectures, Coordination, and Distance: Conway's Law and Beyond. IEEE Software 16(5), 63–70 (1999)
20. Herbsleb, J.D.: Global software engineering: the future of socio- technical coordination. In: Future of Software Engineering (FOSE 2007), pp. 188–198 (2007)
21. Kornstadt, A., Sauer, J.: Tackling Offshore Communication Challenges with Agile Architecture-Centric Development. In: The Working IEEE/IFIP Conference on Software Architecture (WICSA 2007), January 6-9, p. 28 (2007)
22. Laredo, J.A., Ranjan, R.: Continuous improvement through iterative development in a multi-geography. In: Third IEEE International Conference on Global Software Engineering 2008, pp. 232–236 (2008)
23. Mishra, D., Mishra, A.: A Global Software Inspection Process for Distributed Software Development. J. UCS 18(19), 2731–2746 (2012)
24. Mishra, D., Mishra, A.: A review of non-technical issues in global software development. International Journal of Computer Applications in Technology 40(3), 216–224 (2011)
25. Mishra, D., Mishra, A.: Research Trends in Management Issues of Global Software Development: Evaluating the Past to Envision the Future. Journal of Global Information Technology Management 14(4), 48–69 (2011)
26. Mishra, D., Mishra, A.: A software inspection process for globally distributed teams. In: Meersman, R., Dillon, T., Herrero, P. (eds.) OTM 2010. LNCS, vol. 6428, pp. 289–296. Springer, Heidelberg (2010)
27. Ovaska, P., Rossi, M., Marttiin, P.: Architecture as a coordination tool in multi-site software development. Software Process: Improvement and Practice 8(4), 233–247 (2003)
28. Perry, D.E., Wolf, A.L.: Foundations for the study of software architecture. SIGSOFT Software Engineering Notes 17(4), 40–52 (1992)
29. Nord, R., Clements, P., Emery, D., Hilliard, R.: A Structured Approach for Reviewing Architecture Documentation. Technical Note, CMU/SEI-2009-TN-0302009, SEI-CMU (2009)
30. Rocha de Faria, H., Adler, G.: Architecture-Centric Global Software Processes. In: International Conference on Global Software Engineering (ICGSE 2006), pp. 241–242 (2006)
31. Salger, F.: Software architecture evaluation in global software development projects. In: Meersman, R., Herrero, P., Dillon, T. (eds.) OTM 2009 Workshops. LNCS, vol. 5872, pp. 391–400. Springer, Heidelberg (2009)

32. Vanzin, M., Ribeiro, M.B., Prikladnicki, R., Ceccato, I., Antunes, D.: Global Software Processes Definition in a Distributed Environment. In: 29th Annual IEEE/NASA Software Engineering Workshop, April 7, pp. 57–65 (2005)
33. van Vliet, H.: Software Architecture Knowledge Management. In: van Vliet, H. (ed.) 19th Australian Conference on Software Engineering (ASWEC 2008), pp. 24–31 (2008)
34. Yildiz, B.M., Tekinerdogan, B., Cetin, S.: A Tool Framework for Deriving the Application Architecture for Global Software Development Projects. In: IEEE Seventh International Conference on Global Software Engineering (ICGSE 2012), pp. 94–103 (2012)
35. Yu, L., Mishra, A.: Risk Analysis of Global Software Development and Proposed Solutions. Automatika 51(1), 89–98 (2010)