Volume 4, Issue 2, February 2014 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

Challenges Involved in Adding a New Team to an Ongoing Cloud Based Distributed Project

Ram Ji, Devanand Padha

Department of Computer Science and IT,
Central University of Jammu, India

Abstract—a distributed development project is a research & development project that is done across multiple business worksites or locations. It is a form of R&D where the project members may not see each other face to face, but they are all working collaboratively toward the outcome of the project. Distributed Software Development (DSD) in the cloud is rapidly increasing. Ramping up cloud infrastructure is inherent to cloud-based services, but adding a new offshore team to the existing cloud-based DSD project may pose several technical and non-technical challenges. In particular, effective ways to set up and integrate new teams for DSD projects on cloud-based platforms is insufficiently understood. This paper presents various challenges involved in adding a new team to an ongoing cloud based distributed project. The study reveals that ramping up cloud infrastructure in the cloud-based DSD with a new team is particularly challenging from three perspectives: 1) exchanging cloud-based project's architecture with a new team, 2) determining the right cloud access controls for a newly added team and 3) developing scalable team design parallel to cloud infrastructure for future team ramp-up.

Keywords— Cloud-based Software Development, Distributed Software Development, Global Software Development, Empirical Software Engineering, Offshore Software Development, Cloud Computing.

I. INTRODUCTION

Distributed software development (DSD) is constantly adopting new ways of working (i.e. new processes) and new technologies (such as cloud computing) to achieve faster speed of development, lower total cost and better-quality software. Cloud computing, being a recent and ever-increasing technology adoption in DSD, requires an in-depth understanding of its risks and challenges [1, 17, 19, and 20]. Companies are increasingly utilizing cloud platforms to develop software across global teams [14]. Cloud platforms, which provide the central and continuous access to codebase, promote better coding environments, easier continuous integration, and rapid development. Salesforce.com and Force.com are key examples of how developers can create software faster than traditional software development, as database and other services are built into the platform. Cloud-based software development promises to be extremely rapid, i.e. slashing time-to-market by manifold [2, 3]. As another example, the Android platform brings a whole new set of opportunities to deliver software-led solutions from the cloud. As reported in [22], at present software developers create architectures and support strategies that fit the business model in which software products and services will be embedded. Often, companies no longer develop software solutions in isolation. They have to leverage and build onto open ecosystems of development. These examples pose very different sets of challenges for software development and open ecosystems [23]. Such ecosystems are often established when companies want to involve external developers or external development teams in their own development projects or programs. This requires that a common development environment is offered to a community of external developers. In addition, mechanisms need to be established to motivate and integrate such external developers or development teams in the software ecosystem.

II. BACKGROUND

There are several definitions of cloud computing and related stakeholders [7, 8 and 9]. Based on the literature, we convey the simple meaning of cloud. Cloud computing refers to the delivery of a stack of hardware or software residing in the data centre as a utility-like service over the network. We particularly focus on DSD in the context of cloud-based computing environments. We refer to DSD in the cloud as Developing software on the cloud-based platform across multiple geographically distributed sites. For example, Phaphoom et al. [21] present key perceived benefits of cloud-based software development and consequently deduce empirical propositions. Riungu et al. [15] present how cloud-based software testing influences application, management and legal issues. Hashmi et al. [3] and Yara et al. [4] investigate the role of the cloud-based platform in facilitating global software development. The call for research to gain a deeper understanding of DSD in the cloud—especially from the perspective of managing cloud-based DSD—is evident in the literature [e.g. 3, 4, 21]. We specifically focus on integration of new teams in the DSD projects using cloud-based development infrastructure.

III. CHALLENGES

a) Knowledge exchange. The new team, in particular, needs more exchanges of information than an on-going team. The challenge of distribution became more severe due to cloud-based resources. The new team joining the project perceived that exchange of project know-how suffered both in terms of distribution of team and an access to all product material, but without proper understanding of the same. One of the participants in our interviews said: *"We got access to the remote server and lots of material about the product, but that's not enough, as it did not help us to really understand and use the platform".*

b) Synchronizing the work capacity. The new team's work capacity must have right balance with the current team and overall project workload. In other words, the work allocation must happen jointly, involving the new team. New team may lack information on available work capacities when the project started. It turned out that the new team had much more work time available than the project had planned. The new team's work capacity was not considered fully by the project during sprint planning. One of the participants said: *"We had almost double the work time available as the Spanish site just because our team size was much larger and we were allocating more hours".*

c) Communication during team ramp-up. The cloud-based project brings another dimension to the overall team communication, especially when a new team is added. In particular, everything in the cloud makes it bit harder for the new team to use it as a point of reference and in parallel goes through typical DSD challenges. The typical DSD challenges include cultural, temporal and geographic distances related issues. Hurriedly approaching the development causes delays and overheads, and more importantly, weakens the integration of a new team in the project. One of the participants said: *"The first two weeks were pretty much spent on sharing and understanding the environment".* The other major reason for the slower start was attributed to work-time differences. This was believed to be causing challenges in knowledge exchange between teams, as they were getting very little overlapping time for active dialogue to understand the cloud-based platform and the product in development.

d) Understanding the code base. The learning curve for a new team to understand the existing code base gets even longer when the codebase is centrally positioned in the cloud and there is no dedicated anchor to immerse the new team in the large code base. Having a shared codebase in the cloud does not lead directly to expected benefits such as more efficient expandability of the modifiability of the codebase. **T**he common codebase in the cloud requires that the code itself is written with wider architectural perspectives of future development scenarios. It was extremely difficult for the new team to understand it.

e) Access controls. Another challenge of adding a new team to a DSD project is the need to assign appropriate access rights to different teams and team members. In particular, it is important to consider the fact that both, the codebase in the cloud and the changes made, might be related to other systems that are outside the cloud platform. Only a limited number of project participants or only project-external persons might have access to such kinds of systems. In the case of this project the background knowledge and the resulting product were considered strategic for the product owner. In any case, this situation was real and can arise frequently in a cloud-development scenario. Access rights to the cloud-based platform varied for teams in the project depending on their work assignments and needs. The study indicated that determining access controls requires additional support and coordination when access to the cloud platform must be scaled up or down.

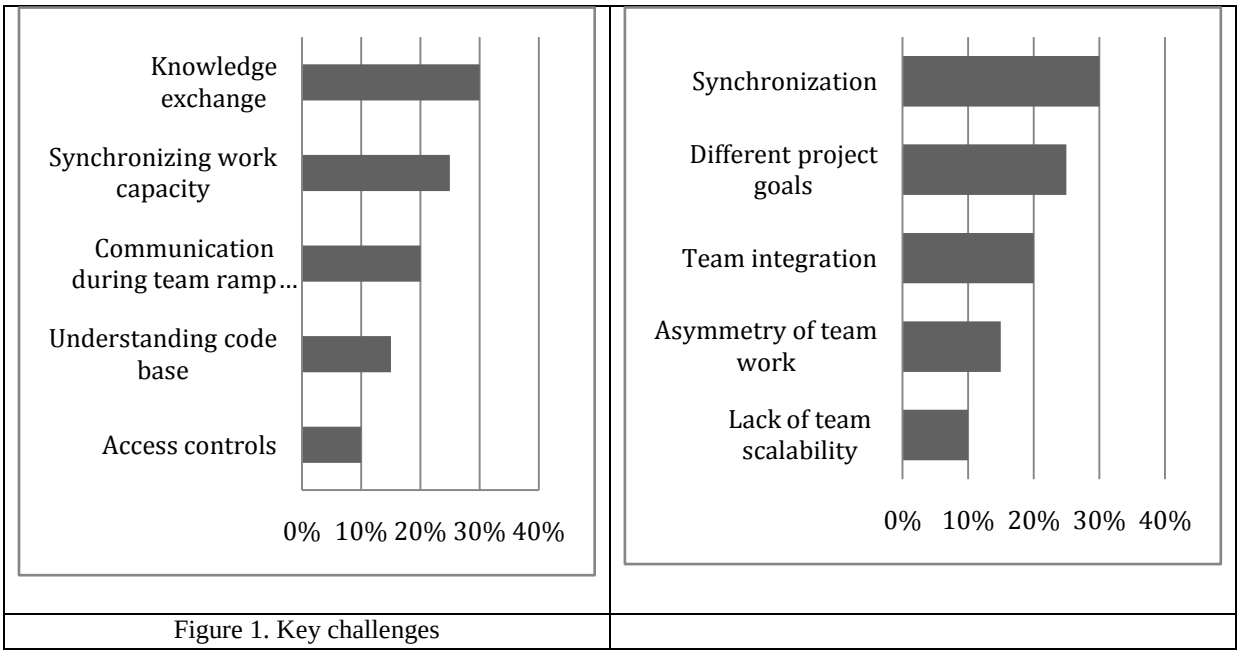

Figure 1. Key challenges

f) Synchronization. The new team may lack enough participation in the collaborative development. In particular, the differences in daily working pattern of the new team and existing teams brought up several challenges.

g) Different Project Goals. The existing team may felt that the new team had a different project goal.

h) Team Integration. This addresses two main challenges: being part of the team and asymmetry of teamwork in the cloud. One of the major concerns is about whether the new team really felt and worked to 'be part of the team' as a whole. Working as one big team may not really achieved, even when video cameras were available to facilitate direct communication with other team members.

i) Asymmetry of teamwork. The existing teams may work more in synch and in more similar ways than the newly integrated team. The problem is not in the different style of work but in the resulting asymmetry of teamwork, which will rely on the same, shared, cloud-based infrastructure. The lack of common rules and guidelines may consider one of the prime reasons behind the asymmetrical work pattern across teams.

j) Lack of team scalability in project design. Finally, another related challenge is about lack of scalability in team design. The new team may face difficulties during their initial cooperation.. Although they started with 'one whole team' spirit when the new team was integrated, several challenges still surfaced and their learning from previous experience did not help directly.

They believed that the scalable team design should have been given more thought when adding a new team to the on-going project. The team also believed that the integration of the new team started too fast and that the new team was brought into the project too quickly. The on-going project had distinct design in terms of technical adaptation, language used, codebase and so on. Integrating a new team with all these parameters being somewhat different created more challenges.

IV. LIMITATIONS

Cloud-based DSD is a promising area that offers concrete benefits towards easier resource sharing, increasing the speed of development and coordinating complex development with a central codebase. However, just having access to cloud-based platforms in DSD is insufficient. In fact, it may lead to increased technical debt and slower development if knowledge about the platform is not properly exchanged with the teams. Here, we discuss results from cross-case analysis and observations in the study.

a) Scalable design. When a new team comes on board, it also brings its own baggage of dos and don'ts. The needs and expectations of a new remote team must be properly understood. This can happen if the overall project design supports scalability. The cloud-based development requires that the new team is not only get proper exchange on process or way of working but more importantly on cloud-based development. The new team should get proper guidance on how to access, use and contribute to the cloud-based code as the changes in it affects overall product development.

b) Dependencies. One of the factors observed is the new team's dependency on the existing team. In particular, the access to cloud-based resources, understanding of the shared infrastructure environment and developing capabilities to commit to the central codebase without creating system-wide problems creates several dependencies on the teams who already have experience. We expected fewer dependencies, as we thought the cloud would provide the central shared access and working platform for DSD teams. If something did not work in the cloud, the new team also became dependent on the team that had control over the cloud platform. We also argue that if the teams are working on multiple DSD projects, they may have to coordinate and work across different cloud platforms with varying access controls and development needs.

c) Cloud culture. Cloud-based development accumulates into larger code base, with more dependencies affecting in real time, compared to conventional development. The cloud-based code became difficult for the new team to grasp and contribute to, especially due to its design, structure and language used for coding and insufficient exchanges across teams. Adding a new team with full access to cloud-based code may result into a longer learning curve. In addition to sufficient exchanges on how the new team can access, use, share and contribute to cloud-based code, code dependencies should be properly documented. We can argue that DSD in the cloud, at this stage, has to not only develop more a robust technical-development environment but also resolve operational challenges, even some others related to strategic knowledge ownership, to reap the benefits of cloud-based development. Many of these problems are not new, but in DSD in the cloud, where resources are shared all over the process, they become more prominent.

One of the major limitations of our findings is their generalizability. The findings we propose in this paper are based on one DSD in the cloud. We also recognize the limitation of subjectivity in the collected qualitative data and the relatively small sample size. However, we countered this limitation by using rich and cross-discussions of researcher observations in the project. The results should be considered an indication rather than affirmative or tested hypotheses

V. CONCLUSIONS AND FUTURE WORK

In this paper, we presented a qualitative study conducted to examine DSD in the cloud. Based on the case objectives, an assessment of its industry relevance and existing literature, we proposed a research question examining the implications of adding a new remote team in a cloud-based DSD project. We collected qualitative data during focus group interviews and further supplemented them with indirect data, including participant observations during the project. Based on the results. We can conclude that adding a new team to the cloud-based DSD project poses specific challenges. In particular,

synchronising work capacity of different teams, understanding the central codebase, knowledge transfer of shared infrastructure and codebase, deciding access controls and committing to cloud-based code could pose specific challenges in the cloud-based DSD context. Nonetheless, people and process-relevant challenges are yet to be overcome. We can say that the elasticity of cloud does not necessarily equate to elasticity of teams working in the cloud! We will continue our empirical inquiry on DSD in the cloud with further analysis of our data at other project sites. Based on our repeatable research design, we call for several case studies to examine DSD in the cloud.

ACKNOWLEDGMENTS

We sincerely thank to management of central university of Jammu, especially department of computer science and information technology and all of its staff members for their moral support.

REFERENCES

[1] Garrison, G., Kim, S., and Wakefield. R.L.: Success factors for deploying cloud computing. Communications of ACM. 55, 9. (September 2012) 62-68.

[2] Arimura, Y. and Ito, M. Cloud Computing for Software Development Environment. In-house Deployment at Numazu Software Development Cloud Center. *Fujitsu Sci. Tech. J*, 47(3)325-334. (2011).

[3] Hashmi, S.I., Clerc, V., Razavian, M., Manteli, C., Tamburri, D.A., Lago, P., Nitto, E.D. and Richardson, I.: Using the Cloud to Facilitate Global Software Development Challenges. Global Software Engineering Workshop (ICGSEW) - 2011 Sixth IEEE International Conference on. (Aug 2011) 70-77. doi: 10.1109/ICGSE-W.2011.19.

[4] Yara, P., Ramachandran, R., Balasubramanian, G., Muthuswamy, K. and Chandrasekar, D. Global Software Development with Cloud Platforms. Software Engineering Approaches for Offshore and Outsourced Development Lecture Notes in Business Information Processing. In O. Gotel, M. Joseph, and B. Meyer (Eds.): SEAFOOD 2009, LNBIP 35, 81–95. Springer-Verlag Berlin Heidelberg (2009).

[5] Erdogmus, H.: Cloud Computing: Does Nirvana Hide behind the Nebula? IEEE Software. 26(2). 4-6. (March-April 2009) doi: 10.1109/MS.2009.31.

[6] Shroff. G. Dev 2.0: model driven development in the cloud. In Proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of software engineering (SIGSOFT '08/FSE-16). ACM, New York, NY, USA, 283-283. (2008). DOI=10.1145/1453101.1453139.

[7] Dillon, T., Wu, C. and Chang, E.: Cloud Computing: Issues and Challenges. 24th IEEE International Conference on Advanced Information Networking and Applications. (2010).

[8] Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I. and Zaharia, M. A view of cloud computing. Communications of the ACM, 53 (4), 50–58. (2010).

[9] Mell, P. and Grance, T. Draft - NIST working definition of cloud computing - v15. (Aug 2009). [URL: http://www.nist.gov/itl/cloud/upload/cloud-def-v15.pdf].

[10] Kniberg, H.: Kanban and Scrum - Making the Most of Both. Lulu.com. 2010.

[11] Runeson P. and Höst, M.: Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering.* 14(2). (2009). 131-164. DOI: 10.1007/s10664-008-9102-8.

[12] Krueger, R.A and Casey, M.A.: *Focus groups: a practical guide for applied research*. Pine Forge Press. (2009).

[13] Kim, W., Kim, D.S., Lee, E. and Lee, S.: Adoption issues for cloud computing. In: Proceedings of iiWAS2009, pp. 3-6. (Dec 2009).

[14] Hattori, L. and Lanza, M.: Syde: a tool for collaborative software development. In: International Conference on Software Engineering (ICSE), Cape Town, South Arfrica, May 2-8, pp. 235-238. (2010).

[15] Riungu, L.M., Taipale, O. and Smolander, K.: Research issues for software testing in the cloud. In: Proceedings of 2nd IEEE International Conference on Cloud computing Technology and Science, pp. 557-564. (2010).

[16] Rellermeyer, J.S., Duller, M. and Alonso, G.: Engineering the cloud from software modules. In: CLOUD'09 ICSE'09 Workshop International Conference on Software Engineering (ICSE), Vancouver, Canada, 32-37. (May 2009).

[17] Buyya, R., Yeo, S., Venugopal, S., Broberg, J. and Brandic, I. Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility, *Future Generation Computer Systems.* 25(6). 599-616. (June 2009). ISSN 0167-739X, 10.1016/j.future.2008.12.001.

[18] Zhang, Q., Cheng, L. and Boutaba, R. Cloud computing: state-of-the-art and research challenges. *Journal of Internet Services and Applications.* 1(1). (2010). 7-18. DOI: 10.1007/s13174-010-0007-6.

[19] Rimal, B.P., Choi, E. and Lumb, I.: A Taxonomy and Survey of Cloud Computing Systems. NCM '09. Fifth International Joint Conference on. 44-51. (Aug. 2009). doi: 10.1109/NCM.2009.218.

[20] www.softwarefactory.cc

[21] Phaphoom, N., Oza, N., Wang, X. and Abrahamsson. P. Does cloud computing deliver the promised benefits for IT industry?. In *Proceedings of the WICSA/ECSA 2012 Companion Volume* (WICSA/ECSA '12). ACM, New York, NY, USA, 45-52. (2012). DOI=10.1145/2361999.2362007.

[22] Favaro, J. and Pfleeger, S.L. Guest Editors' Introduction: Software as a Business. *IEEE Softw.* 28, 4 (July 2011), 22-25. DOI=10.1109/MS.2011.77.

[23] Draxler, S. and Stevens, G. Supporting the collaborative appropriation of an open software ecosystem. *Computer Supported Cooperative Work: CW: An International Journal*, 20(4-5), 403-448. (2001) DOI 10.1007/s10606-011-9148-9.

