RUFORUM Working Document Series (ISSN 1607-9345), 2018, No. 17 (3): 653-659
Available from http://repository.ruforum.org

Research Application Summary

Towards a Scalable Architecture of SME Application: Microservice Integration Approach

*Munezero, I. J. & Balikuddembe, J.

College of Computing and Information Sciences, Makerere University, P.O. Box 7062, Kampala, Uganda
***Corresponding Author:** munejosy@gmail.com

Abstract

Microservice is a scalable architecture that emerges software engineering. Microservices provide several advantages over monolithic systems including scalability and the ability to make rapid functional changes within the system. Existing approaches of microservice are not addressing the issues of Small and Medium Entreprise (SME) in details. This paper examines the challenges that face SME monolithic application for integration strategy that result in high scalable systems. It presents a proposed approach for integrating SME application to microservice architecture.

Keywords: Microservice architecture, scalability, SME application

Résumé

Le microservice est une architecture évolutive qui a émergé du génie logiciel. Les microservices offrent plusieurs avantages par rapport aux systèmes monolithiques, notamment l'échelonnage et la possibilité d'effectuer des changements fonctionnels rapides au sein du système. Les approches existantes de microservices ne traitent pas en détail les problèmes des petites et moyennes entreprises (PME). Cet article a examiné les défis auxquels sont confrontées les applications monolithiques des PME pour la stratégie d'intégration qui se traduisent par des systèmes hautement évolutifs. Il présente une approche proposée pour intégrer l'application PME à l'architecture de microservices.

Mots clés: architecture de microservices, échelonnage, application PME

Introduction

Within the enterprise applications, there has been increased demand for architecture designs that are scalable and give solutions to enterprises to evolve functionally and technicalities (Arshah *et al.,* 2012), especially for monolithic application. The challenges with this architecture is that as enterprises continue to grow functionally and technically, this has a significant impact on the performance of entire system. This makes it more difficult to sustain this monolithic architecture. Equally, in case of an upgrade, other components need to be defined, which add to the impact of the application performance (Bo *et al.,* 2009), and for Small and Medium Entreprise (SME) to scale their monolithic application. A number of suggestions have been proposed to address this particular challenge. Propositions have been made along the line of service oriented architecture (SOA) (Chen *et al.,* 2012); (Faycal *et al., 2*015). The SOA is an approach used to create an architecture based upon the use of services. Services (such as RESTful Web services) carry out small functions, such as requesting an order, registering a customer, or providing simple analytical services.

At a more granular level, other propositions have been made to use microservice architecture. Microservice architecture is an approach for developing single applications as suites of small services, each running in its own process and communicating with lightweight mechanism (Irakli *et al.,* 2016). These services are built around business capabilities and independently deployed by fully automated deployment machinery. Moreover, academic research is supporting microservice on its architecture approach (Alshuqayran *et al.,* 2016). These mechanisms have been adopted quite extensively and widely used for designing more scalable applications (Kalske *et al.,* 2018). Scalability is important for the system to parallel speed up operations without reducing the performance. It is the same for distributed system that require operations to work simultaneously. Microservice have three type of scaling; Vertically scaling is the nature of microservice because the system is separated into different independent services, horizontally scaling consist of making microservice clones when they are needed for example, if microservice is overloaded it can be scaled dynamically, by making different copy of itself, on the time the load is reduced, cloned microservices can be terminated to save resources of other services (Ahmadvand *et al.,* 2016), lastly they can be scaled by partition data so that each server can be responsible of some sort of data.

Based on the need of SME application to be scalable and the capacity of microservice to be a best practice to scale application, we propose to develop a scalable architecture based on integrating microservices architecture style with SMEs applications. We have chosen SME because according to Putra *et al.* (2015), SMEs are different from big enterprises mostly on infrastructure resources they expose to the business and IT (Jan, 2014; Seta *et al.,* 2014), they require minimal investment, and have narrow business domain which determine also the choice of software they use. There is a need for new methods and approaches that can support the communication between monolithic applications and microservices to form a scalable architecture that can work on distributed systems.

Related works

Usage of Microservice Architecture within Enterprises. Enterprises have used microservices for their applications. The applications need to be developed and deployed fast to allow incrementally functionality for new business requirements. Based on the benefits of microservices over monolithic applications, some big companies like Netflix, Amazon, Elsevier have already migrated to microservices (Waseem and Liang, 2017). Enterprises that need to use microservices, start by developing new modules and integrating these modules with the existing systems. Other films build microservices from scratch by using local servers (e.g., Docker) to replace existing monolithic applications to microservices. The idea behind this practice is for reducing on the cost of switching to different architectures all together (Di Francesco, 2017) Soenen *et al.* (2017). More often, microservices are used in the cloud as IaaS to provide a dynamically scalable application infrastructure for data and file storage (Escobar *et al.,* 2016). They are also used as PaaS to analyze the existing methods for testing cloud applications and identify features that are specific to the microservice architecture (Richardson, 2014). Further, they can be used as Security as a Service to build a flexible monitoring and policy enforcement infrastructure for network traffic to secure cloud applications (Stubbs *et al.,* 2015). More often microservice are used because of its capacity of scalability, flexibility, ease of deployment, upgradability, and maintenance that cannot be found in monolithic applications, because services are small and independent of each other.

SME application integration perspective. Integrating existing application to services in order to increase its capacity and performance it not new in the field of Software Engineering. Different approaches have been proposed to integrate legacy systems of big enterprise and small enterprise to SOA (Khadka e*t al.,* 2012; Ravi *et al.,* 2012). Our approach is targeting integrating SME application to microservice for scaling more. The common method for integration of monolithic and microservices is migration (Di *et al.,* 2017; Elberzhager *et al.,* 2017). Different patterns have been proposed for that migration (Miguel *et al.,* 2017), though, other integration methods like replacement, wrapping and reengineering can be embraced for efficient use of microservice architecture (Putra *et al.,* 2015).

Today's leading economies are led by the good information technologies they adopt. Indeed in recent years adoption of information technology in SME has been increased, however some SME have experienced failure (Jan, 2014). The cause of failures can be divided into two categories; First category is the failure caused by the incapacity of SME to run the application. The second is to use improper applications which are not tailored to SMEs characteristics, by laying on large sized enterprise application where applicability is low, because it lacks the depth of application personalized, and efficiency (Sharif *et al.,* 2014). Furthermore, SME have the ability to grow in business capability. This requires an update in information system that is being used, but updates become difficult for SME that use monolithic application. Software projects have a big task to provide solutions to the challenges and risk that SME face (Jan, 2014). SME are largely diverse, and there is a need for researches to develop successful software product that SME can easily utilize and which take into account their characteristics. General characteristics of SME that distinguish them from big enterprise are based on; number of employees, audience/customers, initial investment and capital.

Approach

Our research strategy will implement the proposed approach in four stages, which involves applying different research methodology in software engineering. There are two kinds of methodology that will be used notably lab experiment and live operation observation. Lab experiment will involve software engineering simulator like Palladio. The choice of Palladio is because it is a model driven simulation software that evaluates and estimates required resource of a system architecture before it is implemented. The construction phase itself will start by estimating required resources, evaluation of scalability by number of supported concurrent users, and by predicting the performance (Dullmann *et al.,* 2017). After Palladio experiment, the architecture will be implemented in live environment of SME pilot to check the applicability of the architecture. Different programming tools will be used and their choice will depend on the result expected.

In order to integrate microservice architecture to SME application for the purpose of increasing performance and scalability of the system, we propose the following approach. This approach in general is concerned with designing architecture pattern, that will perform different works. Notably, Monolithic applications will be split into components for ease of invocation by microservices. Different microservices will be built using different technologies to increase functionality. Lightweight protocols will be used for communication between the services. Decentralization will be used to manage data across diverse data centers (distributed environment). The architecture will implement patterns to handle application failure and provide resilience of the services. The microservices will be deployed continuously on the Docker container to ensure upgradability and incremental evolution. To achieve the result expected, this work proposes an iterative approach composed of five stages described below where the test have to be made at every stage to ensure the relevancy of the proposed architecture.

Stage 1: Acquiring SME application details. In this stage, the emphasis is to know how monolithic applications are designed, what the actual performance is and determine the path in the system that should be exposed to integration. Data on how the system is designed in order to understanding the basic working principle and structure of the systems will be collected in the process of reverse engineering. The critical features will be documented in terms of mandatory functional requirements, design constraints and quality attributes.

Stage 2: Determine microservice integration pattern. Microservice integration pattern such as API gateway, service discovery, load balance, resource manager, and container will be exposed during this stage in the process of architecture risk analysis, to analyze these patterns against functional requirement, design constraint and quality attribute requirements of SME application. The process of architecture risk analysis will prioritize the calculation of performance on each pattern exposed or modified. During this stage all patterns will be studied to determine how patterns can be combined with others for better result. Interpretation of the result to choose the most appropriate pattern to use on integration will subsquently be done.

Stage 3: Generate integration architecture model. The general integration architecture model will be generated based on answering this question: By integrating the microservice architecture pattern with the monolithic application design, can we reduce application performance and scalability challenge for SME applications? The architectural model will be designed according to the result of implementation of pattern in the software architecture simulator. This process is done by Replicating SME application specification obtained in stage 1, that have the same performance metrics as in live environment, then add micro-service patterns to the system simulated obtained in stage 2, and measure the performance following by studying how the system can be modified without interruption of the performance.Subsquently, then propose a scalable integration design.

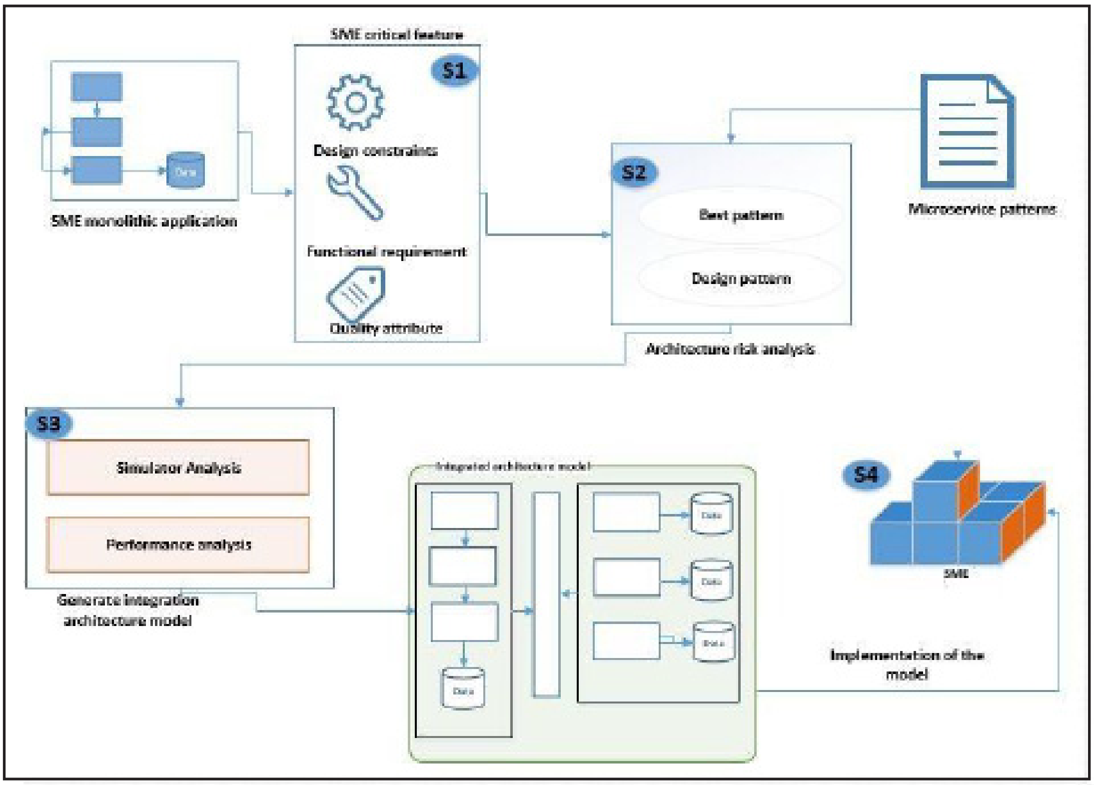

Figure 1. Overview of the proposed integrated architecture

Stage 4: Test and evaluate the performance of the designed integrated architecture

The proposed architecture will be implemented and tested at SME pilot, and the result will be monitored over a period of time to study quality attributes such as performance and scalability capacity of the architecture. Automated test will be implemented to check performance parameter such as response time, loading time and throughput. With stress testing test, infrastructure breaking point will be observed with emphasis on the ability to satisfy the end users established. To test scalability capacity, a set of microservices will be introduced by using different containers (distributed environment). The new microservice will be observed and monitored to determine the infrastructure and system breaking point of the architecture. The experiments will be repeated by changing the existing or altering the number of microservice in the system. The overview of the proposed approach is presented in Figure 1.

Conclusion

Applying microservice architecture in SME that is using monolithic is still a challenge, because it lacks applicability and efficiency due to infrastructure resource constraints. From literature microservice architecture offer the ability to develop applications that are more scalable and lightweight compared to monolithic style. This study will consider SME limitation and design architecture patterns that are tailored to SME. Microservice aspects will be considered to obtain a lightweight microservice integration architecture for SME applications; this will enable performance optimization and ensure system scalability.

Aknowledgement

We thank the Regional Universities Forum for Capacity Building in Agriculture (RUFORUM) for funding this research in partnership with Carnegie Mellon. This paper is a contribution to the Sixth Africa Higher Education Week and the RUFORUM Biennial Conference, 22-25 October 2018, Nairobi, Kenya.

References

Ahmadvand, M. and Ibrahim, A. 2016. Requirements and reconciliation for scalable and secure microservice (De) composition. In: IEEE 24th International Requirements Engineering Conference Workshop (p. 6). https://doi.org/10.1109/REW.2016.14

Alshuqayran, N., Ali, N. and Evans, R. 2016. A systematic mapping study in microservice architecture. pp. 44-51. In: IEEE 9th International Conference on Service-Oriented Computing and Applications (SOCA). IEEE.

Arshah, R. A., Desa, M. I. and Hussin, A. R. C. 2012. Establishing important criteria and factors for successful Integrated Information System. *Research Notes in Information Science* 10 (0): 22–32. https://doi.org/10.4156/rnis.vol10.issue0.4

Bo, W. and Ruizhong, W. 2009. Zero-knowledge trust negotiation. pp. 390–395. In: Proceedings of the 2009 13th International Conference on Computer Supported Cooperative Work in Design, CSCWD 2009. https://doi.org/10.1109/CSCWD.2009.4968090

Chen, S. W., Tseng, Y. T. and Lai, T. Y. 2012. The design of an ontology-based service-oriented architecture framework for traditional Chinese medicine healthcare. pp. 353–356. In: IEEE 14th International Conference on E-Health Networking, Applications and Services, Healthcom 2012, https://doi.org/10.1109/HealthCom.2012.6379435

Di Francesco, P. 2017. Architecting microservices. pp. 224–229. In: IEEE International Conference on Software Architecture Workshops, ICSAW 2017: Side Track Proceedings https://doi.org/10.1109/ICSAW.2017.65

Düllmann, T.F., Heinrich, R., van Hoorn, A., Pitakrat, T., Walter, J. and Willnecker, F. 2017. CASPA: A platform for comparability of architecture-based software performance engineering approaches. pp. 294–297. In : IEEE International Conference on Software Architecture Workshops, ICSAW 2017: Side Track Proceedings. https://doi.org/10.1109/ICSAW.2017.26

Elberzhager, F. and Arif, T. 2017. From Agile development to DevOps : Going towards faster releases at high quality – experiences from an industrial context. Springer International Publishing AG. 1: 33–44. https://doi.org/10.1007/978-3-319-49421-0

Escobar, D., Cárdenas, D., Amarillo, R., Castro, E., Garcés, K., Parra, C. and Casallas, R. 2016. Towards the understanding and evolution of monolithic applications as microservices. pp. 1-11. In: XLII Latin American Computing Conference (CLEI) IEEE.

Faycal, H., Hakima, M. and Habiba, D. 2015. An agent based encapsulator system: For integrating and composing legacy system functionalities. pp. 84–87. In: IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT). https://doi.org/10.1109/WI-IAT.2015.41

Irakli, N., Ronnie, M., Matt, M. and Mike, A. 2016. Microservice Architecture. Bauer, H.

(Ed.), 1st Edition. O'Reilly Media Inc.

Jan, D. 2014. Information Systems for Small and Medium-Sized. In: van Landeghem, H. and Jan, D. (Eds.) (1st edition). Springer-Verlag Berlin Heidelberg, Dirk Deschoolmeester.

Kalske, M., Mäkitalo, N. and Mikkonen, T. 2018. Challenges when moving from Monolith to Microservice Architecture. In: Current trends in Web Engineering ICWE 2017 International Workshops, Liquid Multi-Device Software and EnWoT, practi-O-web, NLPIT, SoWeMine, Rome, Italy, June 5-8, 2017. Springer, Cham.

Khadka, R., Saeidi, A., Jansen, S. and Hage, J. 2013. A structured legacy to SOA migration process and its evaluation in practice. pp. 2-11. In: 2013 IEEE 7th International Symposium on the Maintenance and Evolution of Service-Oriented and Cloud-Based Systems. IEEE.

Khadka, R., Saeidi, A., Idu, A., Hage, J. and Jansen, S. 2013. Legacy to SOA evolution: a systematic literature review. pp. 40-70. In: Migrating Legacy Applications: Challenges in Service Oriented Architecture and Cloud Computing Environments. IGI Global.

Miguel, Z., Insfran, E., Abrah, S. and Cano-Genoves, C. 2017. Automation of the Incremental Integration of Microservices Architectures. pp. 51–68. In: Springer International Publishing Switzerland https://doi.org/10.1007/978-3-319-52593-8

Putra, P. O. H. and Hasibuan, Z. A. 2015. E-business framework for small and medium enterprises: A critical review. pp. 516–521. In: 3rd International Conference on Information and Communication Technology, ICoICT 2015. https://doi.org/10.1109/ICoICT.2015.7231478

Richardson, C. 2014. Monolithic Architecture pattern. Retrieved April 5, 2016, from file:///G:/New folder/micro2/web file/Monolithic Architecture pattern.html.

Seta, P.A.K. and Arman, A. A. 2014. General service oriented architecture (SOA) for Small Medium Enterprise (SME). pp. 217–221. In: Proceedings- 2014 International Conference on ICT for Smart Society: "Smart System Platform Development for City and Society, GoeSmart 2014", ICISS. https://doi.org/10.1109/ICTSS.2014.7013176

Sharif, A.M. and Basri, S. 2014, June. Risk assessment factors for SME software development companies in Malaysia. pp. 1-5. In: 2014 International Conference on Computer and Information Sciences (ICCOINS), IEEE.

Soenen, T., Tavernier, W., Colle, D. and Pickavet, M. 2017. Optimising microservice-based reliable NFV management and orchestration architectures. In: Proceedings of 2017 9th International Workshop on Resilient Networks Design and Modeling, RNDM 2017. https://doi.org/10.1109/RNDM.2017.8093034

Stubbs, J., Moreira, W. and Dooley, R. 2015. Distributed systems of microservices using docker and serfnode. pp. 34-39. In: 2015 7th International Workshop on Science Gateways. IEEE.

Waseem, M. and Liang, P. 2017. Microservices Architecture in DevOps. pp. 13-14. In: 2017 24th Asia-Pacific Software Engineering Conference Workshops (APSECW). https://doi.org/10.1109/APSECW.2017.18