Application of Mixed Distributed Software Architectures for Social-Productive Projects Management in Peru

Jorge Valenzuela Posadas
INTILab, Universidad de Ciencias y Humanidades-UCH
Lima, Perú
jvalenzuela@uch.edu.pe

***Abstract*— FONCODES is the public body in charge of financing and managing the execution of social-productive projects throughout the Peruvian territory. The objective of social-productive projects within the framework of the National Strategy of the Ministry of Development and Social Inclusion is to generate economic opportunities for rural households living in poverty and extreme poverty. The key processes to achieve these national strategic objectives are related to the management of portfolios and projects at all stages. The key information of these processes is that related to budget, expenditure authorizations, accountability, liquidation and financial records since it allows evaluating the efficiency of the obtained management results. There are serious limitations during the transmission of information in real time generating bottlenecks in the respective processes. This paper presents the development and application of mixed distributed software architectures to improve the flow and exchange of key information of social-productive projects in the Peruvian territory. A comparison is made of the developed mixed architectures, the monolithic architecture with SOA services and the micro-services architecture as a REST API on Cloud. The mixed distributed software architecture satisfies the need to interoperate legacy systems with a monolithic web application and SOA services however also is required satisfies the need to support high peaks of requests in certain processes where microservices deployed on the Cloud are the best choice. The performance tests results obtained on both architectures are presented, allowing a performance comparison of monolithic architecture and microservice architecture. The paper allow to evaluate at high technical level the benefits and challenges that agencies government can face when implement monolithic web applications and microservices on cloud related to execution perfomance and development team management.**

Keywords*— *microservices architecture, monolithic, service oriented architecture, SOA, API, REST, digital e-government, cloud computing, scalability, agile development, infraestructure as a service, IaaS, platform as a service, PaaS, portafolio financial management, project financial management

I. INTRODUCTION

At present, citizens and companies expect a government to offer public services with reduced costs, more efficient, with administrative transparency, fast delivery and quality. There are many initiatives in the world to establish the Cloud Government [1]. The Cloud has characteristics of agglomerating resources and offer great computing capacity at low cost and high levels of security, and it is recommended as studies show to build a Government Cloud [2] and especially a Government Private Cloud [3] to deliver services more efficient with the use of IT resources and expand capabilities to quickly scale the applications and services available.

Cloud Computing is a model that enables companies to deploy enterprise applications with capabilities to scale their computing resources on demand. The options to deploy own applications on the cloud are Infrastructure as a Service (IaaS) [4] or Platform as a service (PaaS) [5], the issues to take into account in the deployment of applications on the Cloud are: Continuous delivery, hot deployment, high availability, dynamic monitoring, among others. [6] Companies attempting to deploy applications in IaaS or Paas generally attempt to deploy a monolithic application.

A monolithic application is an application with a simple large codebase that provides tens or hundreds of services using different interfaces such as HTML pages, Web services and / or REST services. Scaling up on-demand applications to support peak periods by gaining operational efficiencies forces companies to move to the Cloud. A monolithic architecture agglomerates widely used services with services that are not. Scaling the monolithic application does not make efficient use of computational resources.

Continuous delivery [7] and agile methodologies allow you to enable capabilities to innovate, change and update your applications in production continuously seeking to satisfy better product experiences and new features. This continuous improvement of the applications is more easily achieved lately through deployments in the Cloud.

Ensuring that all services continue to work is the concern of developers in monolithic applications, as the number of services increases complexity and limits the ability of companies to innovate with new features. A restoration of a monolithic application restarts all the services contained in it, generating a bad experience for users. A monolithic display represents a single point of failure. If the application fails the full set of services stops working. These difficulties have been faced by large companies on the Internet, which have been forced to create new strategies, mechanisms and technologies. Microservices are a response to this. The microservice

978-1-5090-6363-5/17/$31.00 ©2017 IEEE

architecture enables nimble innovation, reduced complexity, scaling computing resources efficiently and growth of development teams in a controlled manner. To understand these benefits in this paper develops and tests a case study where the processes of development, testing, deployment, scaling, operation and updating are experienced.

The efficiency of project management can be achieved through project portfolio management [8]. The projects represent a significant investment for organizations, in this context, project portfolio management efforts are considered as a main efficiency and effective mechanism to align the execution of projects with the organizational strategy. This is given by measuring, ranking and prioritizing the components to pre- established criteria. On the other hand, the implementation of the project portfolio management process with agile practices is applied in small and medium scale projects [9], seeking to reduce the difficulties and excessive documentation required in a traditional process. Unlike the perspective of project management that focuses on a single project, the perspective of portfolio management and programs, is focused on managing a set of related projects through sharing common objectives or clients, or projects that have Common resources and interdependencies optimizing resources.

In FONCODES throughout its existence 37 programs and 76 portfolios social-productive have been created related to water and sanitation infrastructure, social intervention, access to organized community to market local products, strengthening of housing for climatic events. With the main motivation at generating economic opportunities for rural households living in poverty and extreme poverty. The volume of projects from 1991 to 2017 by FONCODES amounts to 133,269. There are 324,419 people forming project execution centers in areas of poverty or extreme poverty, grouped in 414 central executing centers. This functional agents transmit information to 26 territorial units. Thus this units collect the information of a portable system and replicate the information to the headquarters located in Lima.

This process of information replicates is executed by a software agent that initiates the transfer every 10 minutes from the territorial units to headquarters and updates the information in a similar time. It is in this perspective that the implementation of service oriented software architectures (SOA) is necessary to allow the transfer of information in real time from remote locations with Internet access to the headquarters in Lima. At first, the SOA architecture seems to be sufficient to achieve the objectives of integrating the information of the organization. However, characterized processes where there are an undetermined number of requests, which happen to be tens to thousands and require to support peaks of concurrent requests. Then is when SOA begins to generate bottlenecks. Giving place to the architecture of microservices or Cloud-ready to solve this need.

This paper is organized as follows: Section 2 presents the evolution of application development with SOA and micro-services. In Section 3 we present the case study developed. The details of implementation of the application using mixed distributed software architecture are described in section 4. Finally in Section 5 we present the performance results, conclusions and line of research from the implementation.

II. Related work

Applications that need to support huge amounts of concurrent users are becoming more common in governments. Due to ease of Internet access, massive use of mobile devices, provision of public cloud services, exponential growth of Startups and e-business; and the initiative of governments to provide more online services to citizens and companies. The purpose is seeking greater transparency within the framework of the concept of Open Government. Organizations move on the roadmap from traditional Service Oriented Architecture (SOA) toward Enterprise Service Bus (ESB) and microservices.

Usually the traditional SOA architecture has been used to bridge the monolithic applications of legacy system that maintain information in isolation. However for peaks of requests from thousands to millions of users, ESB generates bottlenecks since it has not been built for that purpose. In that sense the microservice architecture emerges as a light subset of SOA, companies that have adopted microservices as part of their business architecture are Amazon [10], Netflix [11], Ubber [12], among many others. Microservices are intended to achieve simplicity at the business and technical level; and agility of development tasks [13] and deployment, allowing development teams to scale and agile deploy over the Cloud.

From the high-level technical viewpoint, microservices divide an application into a set of small, self-contained, expandable and scalable services, containing a business layer and a persistence layer on the back-end and a presentation layer on the front end, within a separate and independent technology stack, each one offers a sub-set of services provided by one application that is usually deployed in a gateway. Each microservice is called by a set of Web applications deployed on a Gateway. Being the Gateway application as a gateway for microservices, it exposes those using different interfaces and protocols such as HTTP, SOAP and REST. Gateways have no persistence layer.

From the teamwork pointview, each microservice is independently developed and tested by a development team using the most appropriate technology stack given the requirements. The microservice development team is responsible for deploying, scaling, operating, and updating on IaaS or PaaS. In the presentation layer the microservices are deployed using the REST (Representational State Transfer) architecture style [14] because it uses HTTP methods directly and its simplicity.

In this paper we implement both styles of distributed architectures and support the use of both because they respond to the current requirements of the government office of management of Peruvian social-productive projects. We highlight technical differences and teamwork in the development process Software, in the maintenance and scalability of the respective applications, and based on this we project the benefits of the use of each one of them.

III. Monolithic and Microservices Case Study

In order to evaluate the implications of using microservice style, we have developed the Project Execution Information System used the monolithic architecture for the Financial Execution Web module that supports the processes of Expenditure Authorization, Accountability, Liquidation and Financial Records [15]. We have used the microservice architecture for project information queries through the provision of REST API Institutional Services [16] consumed by a mobile application built for that purpose. Being the most concurrent process the first days of the month to register the authorizations of expenditure, the last days of the month in

terms of accountability and each end of the quarter pre-liquidations in monolithic application. While the mobile application architecture with microservices supports a greater number of concurrences during the last 15 days of the month when project supervision is carried out at national level and is consulted by Headquarters supervisors, regional supervisors and local supervisors. Teams responsible for projects execution as well as the beneficiaries (5000 consultations during the first hours of the day).

The two architectures defined in the Project Execution Information System are described below:

A. *Financial Execution Web Module (Monolithic Architecture)*

A typical monolithic architecture must have a single codebase and this can be developed using an MVC (Model View Controller) web application framework such as JEE, Dot Net, Play, Laravel among many others. The architecture is illustrated in Fig.1.

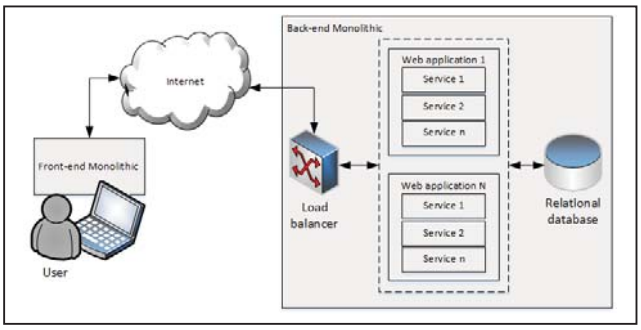

Fig. 1. Deployed Web Application Monolithic Architecture

B. *Institutional REST API (Microservices Architecture)*

The Gateway is developed as a light web application that receives request from end-users, gets the result consuming one or more microservices, and returns the results. At the presentation layer, the Gateway exposes to Internet two services to end-users: Institutional news and project execution information. At the business logic layer, Gateway consumes the services offered by the microservices through REST using JSON. The architecture is illustrated in Fig.2.

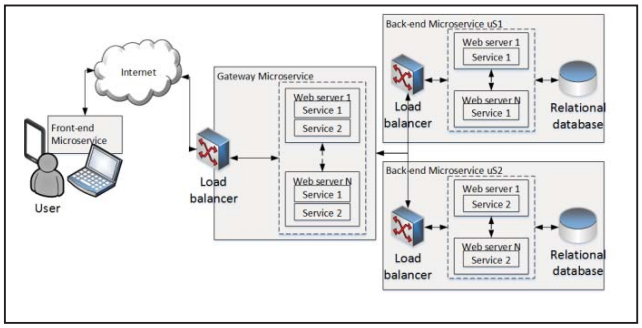

Fig. 2. Deployed Web Application Microservices Architecture

The microservice architecture would be deployed on a Cloud solution. The microservice web application is deployed using a load balancer and multiple web servers and use relational database to store information. In a common scenario the microservice application would be developed by four micro teams, a micro- team developing the Gateway, other the microservice 1, other the microservice 2 and other the front-end application.

IV. Implementation

To implement both architectures we use the .Net framework, We selected ASP.Net Web API [17] to developing microservices because it provides the option to deploy in On-Premise Server and after, the option to deploy on Cloud migrating to ASP.Net Core or using pure microservice framework Azure Service Fabric, being the last option the best way to deploy to Cloud. For monolithic application we selected ASP.Net MVC [18] because allow create traditional one codebase and use WCF (based XML) services through proxy service contracts. Both application was deployed on IIS servers and the microservices architecture was deployed on Azure Cloud Services to purpose this paper. MySQL and Oracle was used as relational database en microservice architecture and only Oracle was used in monolithic architecture. The gateway application is an API REST that was deployed on IIS and Azure Cloud Services to purpose this paper.

The application executed in the browser on monolithic application was developed with ASP Net MVC and jquery, ajax, jqueryUi and bootstrap. While the application front-end monolithic was developed on Android and support request on HTTP protocol on Web application Asp.Net Web API.

Both architectures may be developed using other back-end frameworks such as JEE, Play, Laravel among others, however the goal of this work is to present the mixed: Monolithic and Microservices Architecture used to support the information flow of portfolio and project management of the public organization and improve work agility with both architectures. Although of differences, the monolithic architecture was developed to meet requirements of transition from legacy client-server applications to SOA and the microservice architecture was developed to meet requirements of on-demand scalability and easy deploy on Cloud. One of the benefits the microservices is the capability of using multiple technological stacks as units specialized getting the better technological options.

V. Results

With the presentation of the case study, different aspects are evaluated to get a better criteria of selection in future development web application and services on Peruvian government. About the performance, the average response time on monolithic and microservices was tested with JMeter [19][20]. It was defined two services. S1: Register a authorization of expenditure with Monolithic Architecture and S2: Get a financial execution state of project with Microservices Architecture. The performance requirements was considered and differentiated by each one, Jmeter was configured to execute 40 request per minute to S1 and 1,500 request per minute to S2 during 20 minutes to simulate the workload. It is illustrated in Fig.3.

It show that both architectures meet the performance business requirements for services. The impact on latency is related to using more computational resources in both architectures. Managing Cloud instance types with microservices is more easy that Monolithic, because software granularity help to scalability and reduce costs. Conway's Law [21] is verified with microservices, because require a change in the way the organization develop software,

each team work on small applications and maintain agile communication and collaboration.

TABLE I. REQUIREMENTS AND JMETER TESTING RESULTS

Requirements			Jmeter Testing results	
Service	*Request per Minute*	*Average Response Time (ms)*	*Request per Minute*	*Average Response Time (ms)*
Register a authorization of expenditure	40	3000	40	1650
Get a financial execution state of project	1500	400	1500	175

Fig. 3. Requirements and Jmeter testing results.

This change in the software development organization complemented with automation tools facilitate the adoption of Devops practices, where teams work on development, deployment, operation and monitoring on the cloud is realized agile, frequently and more reliably. Each microservices enables horizontal scaling to support users request peak periods, it represents saving in IT infrastructure costs, efficient pay per computational resources use and on-demand benefits. For applications with a small number of users, the monolithic architecture may be a more practical way, usually microservices applications start as monolithic and after with requirements of increment infrastructure scalability become to microservices. REST API gateway web application is important to allow that microservices can be easily consumed. To deploy and scale microservices and gateways on Cloud environments, it is recommended to use lightweight servers.

VI. CONCLUSIONS AND FUTURE WORK

The paper allowed to evaluate at high technical level the benefits and challenges that agencies government can face when implement applications with monolithic or microservices architectures, the test realized on-premise and cloud allowed to get important conclusions related to execution perfomance and development team management. In this case study the mixed architecture allow to bring and share information in massive scale of remote places of Peru where realize social-productive projects in several sectors where poverty and extreme-poverty exists. Microservices and Monolithic Architectures with SOA are used to meet Peruvian government requeriments actual, however when there is demand peaks on specificity time periods, the microservices approach response improves infrastructure costs saving and response more agile way by your granular scalability, overcoming for this use case as a better option.

As future work, we are going to evaluate the automatic deployment of microservices and gateways managed by cloud services on Government services, adoption of agile methodologies and Devops practices on Government software development processes and adoption Cloud Model for costs efficiencies in Government.

ACKNOWLEDGMENT

The author would like to thank to Information Technology Unit of FONCODES Peruvian Government Agency. They gave us valuable support to conduct this work.

REFERENCES

[1] WangNing, Xie Xiaoshan, et al. “Survey of Application and Research on Government Cloud Computing In China”, in Proc. IEEE/WIC/ACM, 2015, pp.140-143.

[2] Jin Tian, “Research on the Application of Cloud Computing in E-Government”, in Proc. International Conference on Computer Science and Service System (CSSS), 2011, pp.1630-1632.

[3] Rui-Hua Di, Hai Lv, et al. “Research on the Impact of Cloud Computing Trend on E-government Framework”, in Proc. International Conference on E-Business and E-Government (ICEE), 2011, pp.1-4.

[4] Dadang Sunandar, Albarda. “Design of Software as a Service (SaaS) for regional data service: Case study: Statistics Indonesia”, in Proc. International Conference on Information Technology Systems and Innovation (ICITSI), 2016, pp.1-6.

[5] Giuseppina Cretella and Beniamino Di Martino. "An Overview of Approaches for the Migration of Applications to the Cloud.". Springer, 2014, pp.67-75

[6] Mario Villamizar, Oscar Garces et al. “Evaluating the Monolithic and the Microservice Architecture Pattern to Deploy Web Applications in the Cloud”, in Proc. 10th Computing Colombian Conference (10CCC), 2015, pp.583-590.

[7] Carmine Vassallo, Fiorella Zampetti et al. "Continuous Delivery Practices in a Large Financial Organization", in Proc. IEEE International Conference on Software Maintenance and Evolution (ICSME), 2016, pp.519-528.

[8] Ana Lima, Gabriela Fernandes et al. “Project and Program Management Implications in the Portfolio Management of IT Projects in Applied R&D Organizations”, in Proc. 10th International Conference on the Quality of Information and Communications Technology (QUATIC), 2016, pp.224-229.

[9] Lílian da Silva, Sandro Bezerra et al. “A Process Framework with Agile Practices for Implementation of Project Portfolio Management Process”, in Proc. 10th International Conference on the Quality of Information and Communications Technology (QUATIC), 2016, pp.146-149.

[10] Charith Tangirala, “Microservices Architecture for Enterprises”, [Online], Internet: https://www.thoughtworks.com/insights/blog/microservices-architecture-for-enterprises , Jul 2015.

[11] Josh Evans, “Mastering Chaos - A Netflix Guide to Microservices”, [Online], Internet: https://www.infoq.com/presentations/netflix-chaos-microservices , Dec. 2016.

[12] Emily Reinhold, ”Rewriting Uber Engineering: The Opportunities Microservices Provide”, [Online], Internet: https://eng.uber.com/building-tincup/ , Apr.2016.

[13] George Lawton. “How microservices bring agility to SOA”, [Online], Internet: http://searchcloudapplications.techtarget.com/feature/Howmicroservices-bring-agility-to-SOA, Jan. 2015.

[14] Roy Fielding, “Architectures Styles and the design of network-based softwae achitectures”, Doctor of Philosophy Thesis Dissertation, University of California, Irvine, USA, 2000.

[15] Foncodes- Peruvian Government Agency, “Financial Execution Web Module.” Internet: http://sistemas.foncodes.gob.pe/Ejecucion, Aug. 2016

[16] Foncodes- Peruvian Government Agency, “Institutional REST API”, Internet: http://sistemas.foncodes.gob.pe:8081/ApiRestFoncodes, Jul. 2016

[17] Tugberk Ugurlu, Alexander Zeitler et al. “Pro ASP.NET Web API”, Springer, 2013, pp 57-74

[18] Adam Freeman. “Pro ASP.NET MVC 5”, Springer, 2015, pp 51-66

[19] Emily Halili, “Apache JMeter”, Packt publishing, 2008

[20] Srinivasa.Shenoy, Nur Asyikin, et al. “An Adaptive Framework for Web Services Testing Automation Using JMeter”, in Proc. 2014 IEEE 7th International Conference on Service-Oriented Computing and Applications, 2014, pp.314.318.

[21] Kyle L. Blatter; T. J. Gledhill, et al. in Proc. 3rd International Workshop on Replication in Empirical Software Engineering Research, 2013, pp. 25-33..