2016

# Integration of Micro-Services in The Industrial Internet Era

Sergey Zykov
*National Research University Higher School of Economics*, szykov@hse.ru

Yaroslav Gorchakov
*National Research University Higher School of Economics*, jagorchakov@hse.ru

# Integration of Micro-Services in The Industrial Internet Era

**Sergey Zykov[1], Yaroslav Gorchakov[2]**
[1]Faculty of Computer Science, School of Software Engineering, National Research University — Higher School of Economics, Myasnitskaya Street 20, 101000 Moscow, Russia
[2]Faculty of Business and Management, School of Business Informatics, Department of Modeling and Business Process Optimization, National Research University — Higher School of Economics, Myasnitskaya Street 20, 101000 Moscow, Russia
szykov@hse.ru, jagorchakov@hse.ru

## Abstract

The need for integration arises when companies start considering "boundary-less" information flows across a plateau of multiple enterprise information systems. Adding a new application to the system or replacing one of the existing applications requires the establishment of interfaces with a specific software tool for each of them. The classical way to solve this problem in a large-scale organization is to use a central integration platform based on service-oriented architecture (Service-Oriented Architecture (SOA)). In this work, we forecast the increase of granularity of services and application of micro-services to enhance traditional SOA in the prototypes of industrial internet and Web 3.0.

## Introduction

The benefits of integration platforms include quick and clear response to business requirements by creating flexible user services (Aberdeen Group, 2005). Exchanging data between different applications in a unified form in real time increases efficiency of business management, while services could be hosted from different private vendors. In this paper we extend Service-Oriented Architecture (SOA) (Commonwealth of Kentucky, 2006) principles and investigate the optimal granularity of services that are used for integration purposes.

## Business Case

The main aim of this paper is to propose the concept of data integration using micro-services on the example of bank with dozens of IT Systems and implemented SOA principles:

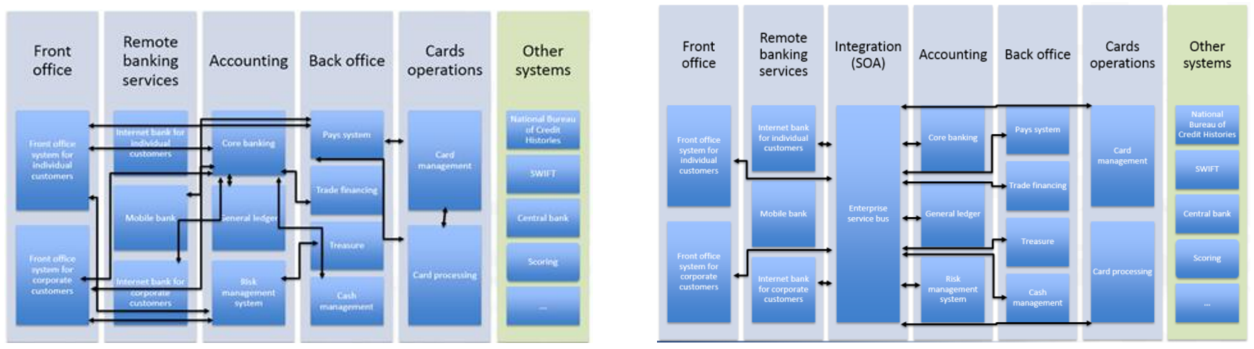

**Figure 1. IT architecture based on "Point to Point" integration versus IT architecture based on SOA principles (sample)**

## Micro-services methodology

The main idea of the micro-services is decentralization and distributed control. Real-time Business Architectures (Gromoff et al. 2012) are more flat, interconnected (in a network) and could quicker be transformed per business requirements. Instant intellectual support expands, and subsequently – dissolves organization borders (Figure 2).

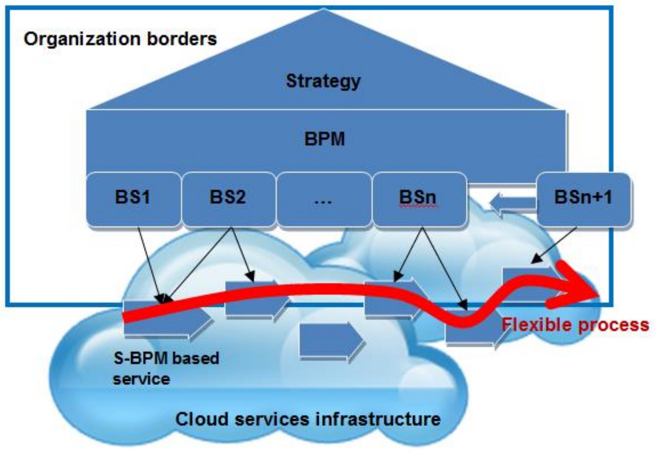

**Figure 2. Real –time process handling in frames of SoEA.**

## Componentization via Services

The ability to dynamically connect new services, as well as the search for these services customers, is also one of the cornerstones of the principles of the system, built on the basis of SOA. Other characteristics are:

| | |
|---|---|
| **Open standards** | Services as components of an information system publish their interfaces (contracts): independent of platform, programming language, operating system and other technical features of the implementation. Services interact and support services through open, widely used standards. |
| **Service performs repetitive** | Each component of the information system service implements a single business function that is logically isolated, repetitive task, which is an integral part of business processes. It is possible to quickly and relatively |

| | |
|---|---|
| **business function** | easy to change business logic, adapting it to the ever-changing market conditions. Changes need to be made only to a single service, and the changes made at the same time used by all client applications. |
| **Loosely coupled service components to build systems** | Services can be implemented regardless of other system services, only required knowledge of interface used services. Low connectivity allows connecting the various components of the information system during its operation with the help of so-called late binding (late binding). |

## Organized around Business Capabilities

Service represents single business capabilities. Micro-service is full-stack- beginning from analysts and finishing with managers.

| | |
|---|---|
| Business features: | Separate creation of business description of the whole system and of each business capability |
| | Business capability micro-service assignment |
| | Forecast diagrams and workflows |
| Managing features: | High –level control of the atomic teams/micro-services |
| | Stack of roles for each micro-service |
| | Size of the micro-service defines the size of the team |
| | Statistics about the teams, their work progress, charts and diagrams |
| Technical features: | Integration between the micro-services – predefined templates, efficient advices and help on integration choice (questionaries') |
| | Graphical representation of the system interfaces |
| | Integration with Version controls systems |

## Products not Projects

A team own a product over its full lifetime. This brings developers into day-to-day contact with how their software behaves in production and increases contact with their users, as they have to take on at least some of the support burden.

| | |
|---|---|
| Managing features: | Provide "Support" tool, which is logical continuation of the development cycle, which supports clients' appearance on the platform (possible integration with Jira, Zendesk, etc.) |

## Smart endpoints and dumb pipes

| | |
|---|---|
| Technical features: | Different integration patterns (possibly on-the-go generated according to the user's needs) |

## Decentralized Data Management

| | |
|---|---|
| Managing features: | Visualize data and system structure, show references between databases<br>Each micro-service may have its own database (or may share it with some other micro-service)<br>Build it / run it –The team is fully responsible for their builds 24/7<br>Build statuses via messaging system |
| Technical features: | Support of different programming languages<br>Build new services with consumer driven contracts-use simple tools that define the contract for a service. This becomes part of the automated build before code for the new service is even written. The service is then built out only to the point where it satisfies the contract - an elegant approach to avoid the 'YAGNI' (Möller, 2016) dilemma when building new software. |

## Infrastructure Automation and evolutionary design

| | |
|---|---|
| Technical features: | Integrate with Infrastructure Automation tools (Jenkins etc.) or develop our own plugin (for personal non-commercial use). |

## Design for failure

Micro-service applications put a lot of emphasis on real-time monitoring of the application, checking both architectural elements (how many requests per second is the database getting) and business relevant metrics (such as how many orders per minute are received). The consequence is that micro-service teams constantly reflect on how service failures affect the user experience.

| | |
|---|---|
| Managing features: | Provide convenient monitoring dashboards, which reflect the statuses, messaging systems (through skype, what's up |
| Technical features: | Provide monitor tools for user and system activity |

## Architecture solution

To clarify the vision of the platform, the Entity –Relationships diagram was created (Figure 3).

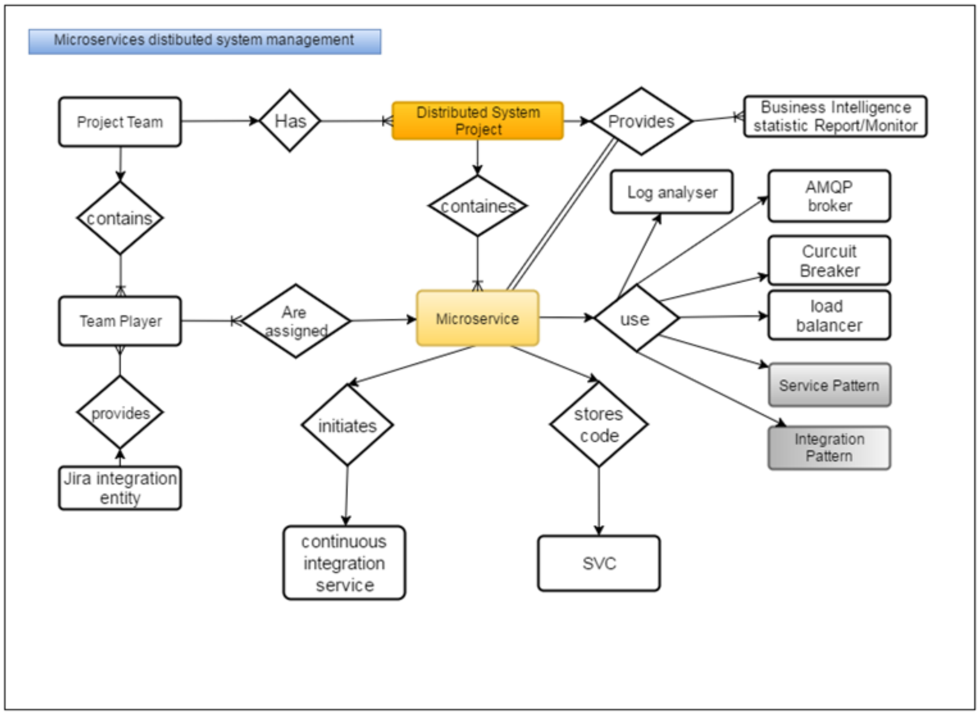

**Figure 3. Entity –Relationships diagram of micro-services decoupling**

Project manager creates new logical unit- new Project. For the project, he assigns the team (team may be also assigned directly to the micro-service). Team player's workload statistics helps manager to define the right team.

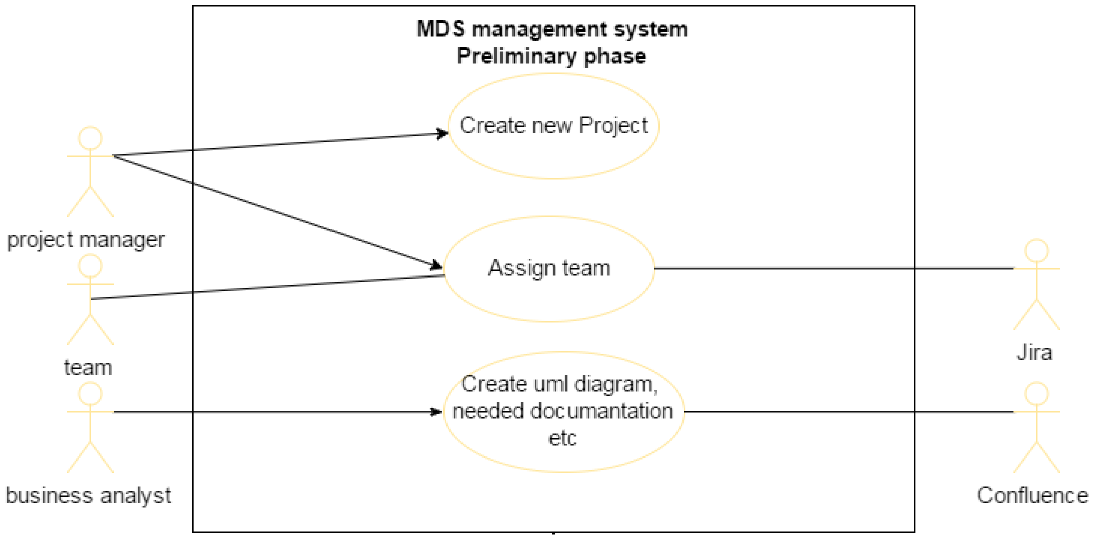

**Figure 4. Distributed system project creation**

Project manager adds micro-services projects to the system project, assigns roles, by following rules and descriptions of the micro-services methodology. Business analyst provides business scope, attaches required documentation. Then the team (probably team lead) begins technical project preparation- chooses patterns or create blank project on the specific language.

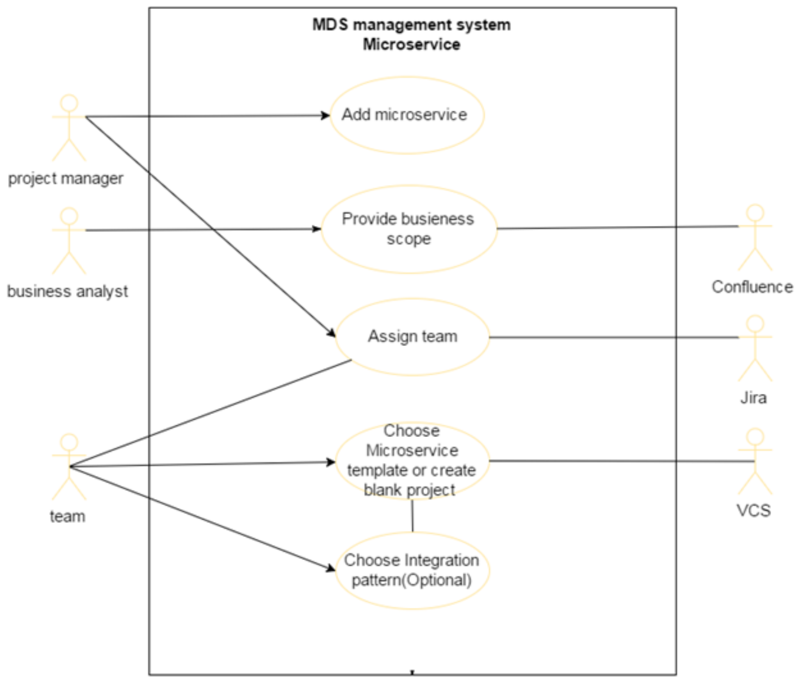

**Figure 5. Micro-service creation**

To monitor project statistics, project manager uses special graphic tools, which allows checking some organizational statistics, as tasks completion, current structure of the system, data flows between micro-services, database structure etc.

Besides, system metrics are obtained from IMPQ broker, which allows understanding the performance of the micro-service, finding vulnerabilities, and making conclusions and suggestions.

## Micro-service Project development, testing and deployment

The concept of Continuous Delivery is used for monitoring a multicomponent system whose components independently change and influent each other. Micro-services project is a good example of such case. We integrate with Travic CI, which will automatically and constantly tests the system and provide the statistics. CodeCov would help to check the code coverage and by the help of DockerHub user will be able to build, ship, run and deploy the microservice.

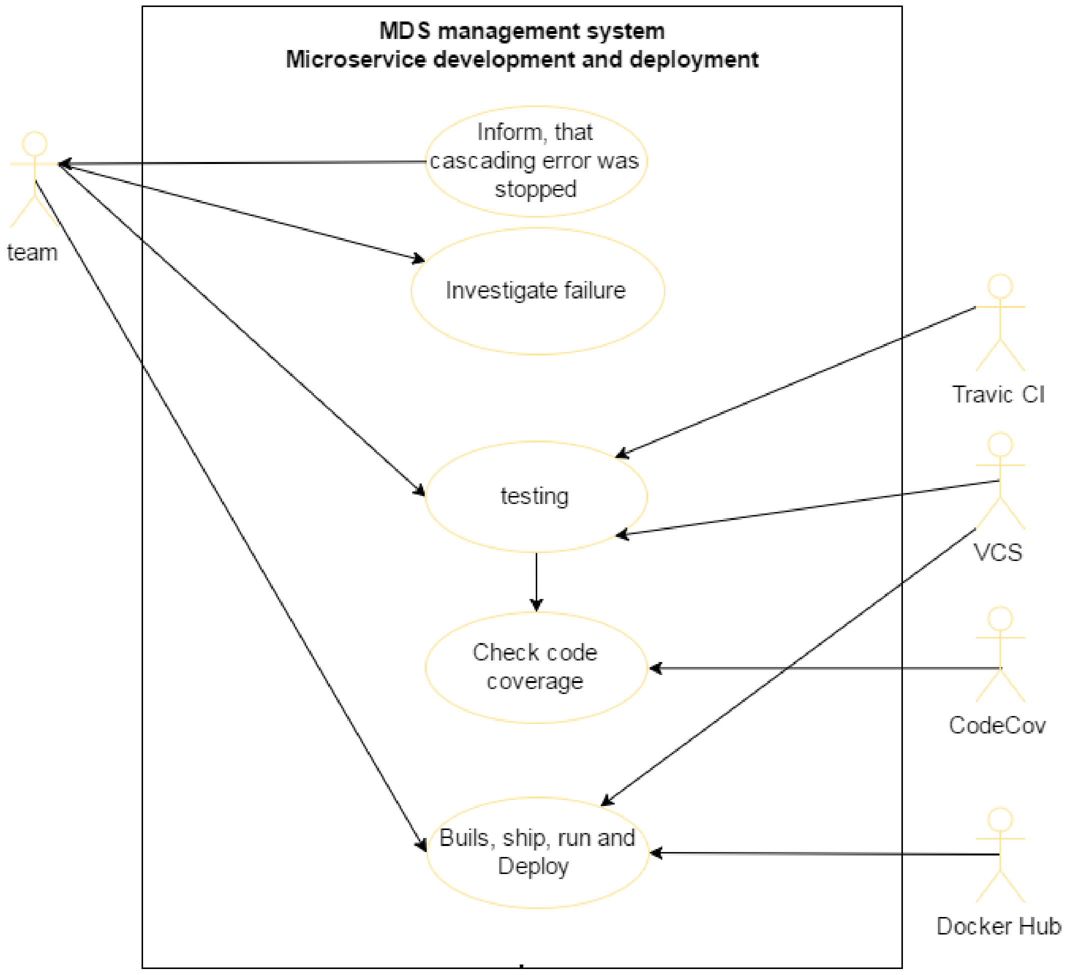
MDS management system
Microservice development and deployment
Inform, that cascading error was stopped
Investigate failure
testing
Check code coverage
Buils, ship, run and Deploy
team
Travic CI
VCS
CodeCov
Docker Hub

**Figure 6. Project development, testing and deployment**

To monitor project statistics, project manager uses special graphic tools, which allows checking some organizational statistics, as tasks completion, current structure of the system, data flows between micro-services, database structure etc.

Besides, system metrics are obtained from IMPQ broker, which allows understanding the performance of the micro-service, finding vulnerabilities, and making conclusions and suggestions.

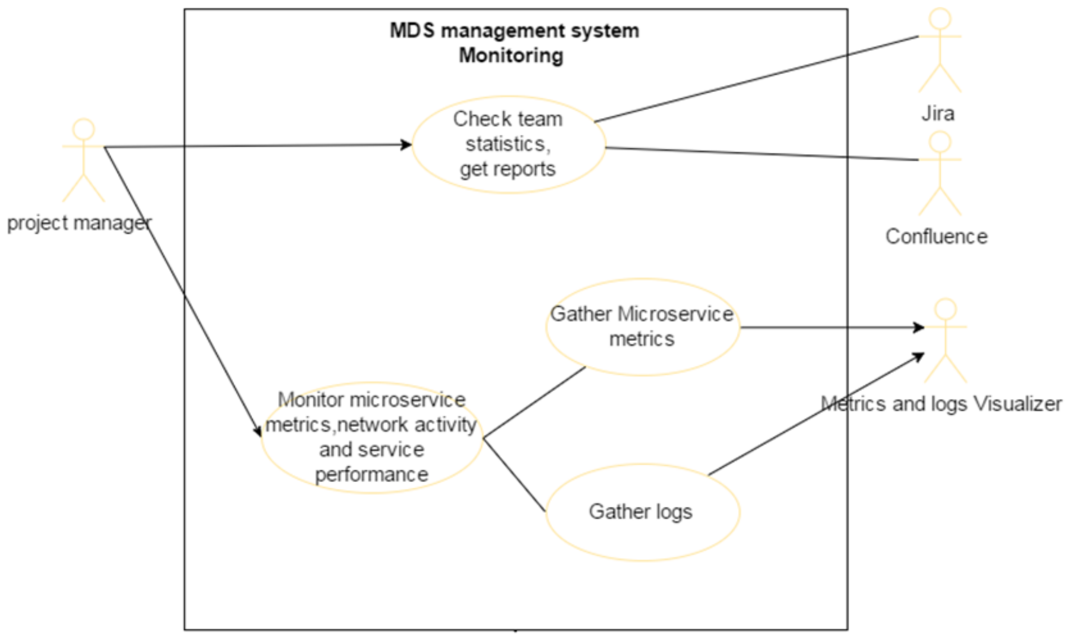
MDS management system
Monitoring
project manager
Check team statistics, get reports
Jira
Confluence
Gather Microservice metrics
Monitor microservice metrics,network activity and service performance
Gather logs
Metrics and logs Visualizer

**Figure 7. Project Monitoring**

## Conclusion

The proposed approach of micro-services handling for enterprise IS integration raises questions and is expected to be discussed during workshop.

## References

WHITE PAPER, Oracle SOA vs. IBM SOA Customer Perspectives on Evaluating Complexity and Business Value, A CRIMSON CONSULTING GROUP, 2011.

"Solving Integration Problems using Patterns" http://www.enterpriseintegrationpatterns.com/patterns/messaging/Chapter1.html

Enterprise Service Bus and SOA Middleware, Boston: Aberdeen Group, June 2006, p. 5

Commonwealth of Kentucky., NASCIO Recognition Award Application: Enterprise Service Bus, June 2006

Washington DC., NASCIO Recognition Award Application: Enterprise Architecture District Enterprise Integration Stack, June 2006

"The ESB in the Land of SOA", Information Technology Research: Enterprise Strategies, Boston:Aberdeen Group, December 7, 2005.

Howard, Philip, SOA and Information Services, Bloor Research, March 2006.

Gromoff A., Kazantsev, N., Kozhevnikov, D., Ponfilenok, M. and Stavenko, Y. (2012). Newer Approach to Create Flexible Business Architecture of Modern Enterprise. Global Journal of Flexible Systems Management. 13(4), Springer-Verlag, 207-215

Möller, D. P. (2016). Introduction to Cyber-Physical Systems. In Guide to Computing Fundamentals in Cyber-Physical Systems (pp. 81-139). Springer International Publishing.

Corcoran, P. (2016). Advance Romance at ICCE 2017 [Conference Reports]. IEEE Consumer Electronics Magazine, 5(3), 27-34.

Stavenko, Y., Kazantsev, N., & Gromoff, A. (2013). Business process model reasoning: From workflow to case management. Procedia Technology, 9, 806-811.

Gromoff, A., Kazantsev, N., Schumsky, L., & Konovalov, N. (2014, June). Business transformation based on cloud services. In Services Computing (SCC), 2014 IEEE International Conference on (pp. 844-845). IEEE.