PRINCIPLES OF THE NEWDIMENSIONS SOFTWARE CREATION FOR A CONTROL CENTRE OF THE FUTURE: CLOUD COMPUTING AND SOFTWARE ARCHITECTURE

RÚBEN ARAÚJO[1], JOAQUIM NUNES[1], AFONSO FERNANDES[2] & ROLANDO MARTINS[2]
[1]EFACEC, Portugal
[2]INESC TEC – Instituto de Engenharia de Sistemas e Computadores, Tecnologia e Ciência, Portugal

ABSTRACT
The ongoing technological evolution implies a continuous effort to update OCC (Operation Control Centre) solutions to match market needs and directives. Given that, EFACEC, a long-time player on the transportation business domain, is actively seeking novel solutions to correspond to customer requirements. From this context NewDimensions was born, a project that aims to make use of the current scientific state-of-the-art technologies and architectures for a new generation of company's software solutions for the railway market, through seamless integration of cloud computing, machine learning and cybersecurity. A particular product (INOSS PIS Rail) is becoming a pilot project for the ongoing modernization that is taking place within EFACEC's transportation solutions. A modular microservice oriented approach was chosen to drive the development onwards. Moving from legacy architecture to microservices allows us to build more reliable products, that are easier to scale up and down while having simpler deployment processes. In this paper we reveal the first steps taken to achieve a modern "OCC-as-a-service" solution. It is analysed the impact of the new architectures and technologies implemented, why they were chosen and their practical application. We also address the principles followed during the progressive modernization process. Finally, we present the next steps to continue this modernization and the direction that shall be taken.
Keywords: OCC, cloud computing, microservices, machine-learning, cybersecurity, NewDimensions.

1 INTRODUCTION

Being involved in the transportation business for a long time, EFACEC (with its Transportation Business Unit) is facing challenges coming from its long longevity and legacy systems. With the continuous technological evolution, EFACEC's OCC (Operation Control Centre) solutions are being restructured to respond adequately to customers' needs. These include fast release of new features and use of emerging technologies associated to the cloud and artificial intelligence, without neglecting cybersecurity threats.

From this context NewDimensions emerges. This project aims to take advantage of current scientific state-of-the-art technologies and software architectures for the railway market. Cloud computing, new software architectures, artificial intelligence and cybersecurity are part of the range of subjects in question.

With this modernization, EFACEC intends to present the market with a new and modern generation of OCC solutions and keep up with the latest trends on the business field.

In this paper we intend to reveal the steps that EFACEC is taking in the context of this modernization; explaining the decisions taken and what they aim to achieve after implementation. We also look to the results of the implementation that we achieved at the time of writing the paper.

WIT Transactions on The Built Environment, Vol 199, © 2020 WIT Press
www.witpress.com, ISSN 1743-3509 (on-line)
doi:10.2495/CR200321

2 METHODOLOGY

2.1 Software architecture

The modernization process carried out by the NewDimensions project relies primarily on the implementation of a new software architecture, in this case a microservice architecture. Microservices is a software architecture approach that encourages for many granulated services, each of them responsible for a single business purpose. Each microservice is a self-contained, independently deployable part of a big application that communicate with other elements, usually via well-defined REST APIs. REST, or Representational State Transfer, is a standard for communication between systems via web [1].

Microservices are often recognized as an architecture for building scalable applications running in the cloud. Additionally, they also guarantee high maintainability due to smaller code bases and solid component separation. These characteristics make microservices an interesting option for software modernization [2]. Additionally, the team autonomy promoted by microservices is likely to reduce coordination effort and improve team productivity.

For business applications, which are typically in service for many years, maintainability is also of major importance. Therefore, microservices promise an improvement over conventional monolithic applications. The major reason for modernizing the application architecture is to bring more efficiency to the system and allow the deployment of new features with a proper time-to-market response.

The goals of modernizing the software architecture are:

- To establish a well-defined and independent interface based on the bordered contexts of the underlying platform. The independent interfaces will give the developers the capability of evolving the product without affecting the existing system.
- An incremental migration of programming languages, from C++ to Java.
- An incremental migration from TAO to REST (The ACE ORB (TAO) is a C++ implementation of CORBA).

Microservices is a suitable architecture for achieving these goals due to their organization around business capabilities. The ability to evolve easily, strong component separation, and focus on cross-platform interaction is of extreme importance. Microservices architectures enable a gradual modernization of the legacy applications.

2.2 Modernization process of the software architecture

Modernization of legacy systems creates many challenges due to their size and complexity. Size and complexity issues frequently dictate that these systems are incrementally modernized, and new modules could be deployed before the modernization process is finished. This implies that legacy components work side by side with modernized components in the same system.

The idea of a progressive modernization involves a modernization piece by piece. Adding new modules and modernizing the old ones one by one [3], until the system becomes completely renovated. Progressive modernization of a legacy system is illustrated in Fig. 1.

Since modernized modules and new components are being deployed in a gradual process, it is required to merge components from the legacy system along with the modernized ones. This is necessary to preserve the existing functionality during the development phase. Interface adapters and other wrapping methods may be necessary to provide a communication mechanism between the legacy system and the modernized one. This interaction is shown in Fig. 2.

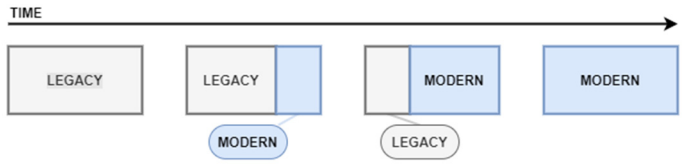

Figure 1: Progressive modernization over time.

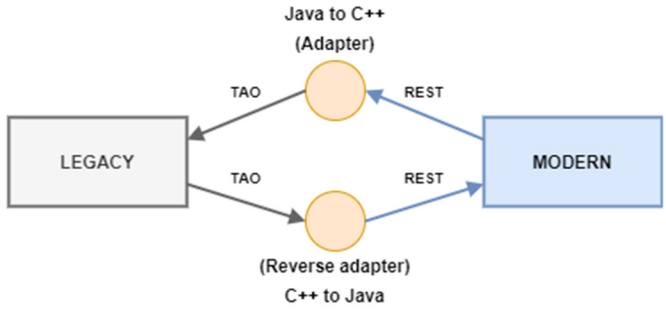

Figure 2: Interoperation of modernized and legacy systems.

A progressive modernization process always aims to keep the system fully operational. This minimizes the amount of rework and technical risk during the upgrade of existing modules or addition of new ones.

2.3 Cloud computing

Today, cloud computing appears as one of the most used and matured technological environment for businesses, hosting and provisioning software systems from all around the world. A continuously growing set of cloud-based solutions in the cloud stack is available to application owners and developers. There are quite a lot of solutions to build applications and exploit the advanced features of this paradigm for elasticity, high availability and performance [4]. These solutions provide many advantages for new applications, but they also create constraints to the modernization and migration of legacy applications. EFACEC's legacy applications, as software not developed with the Cloud in consideration and implemented based on traditional architectural concepts, cannot be as easily scaled and do not share resources beyond infrastructure (e.g. database, memory). A semi-monolithic architecture design style may be deprecated or cannot simply work with the concept of "as a service". Another reason is the fact that being implemented on-premises increases the maintenance costs of the system.

Despite of the reasons for migrating to the cloud, most part of the applications could not profit from the cloud environment, if their main intent is simply to dump the existing architecture to a virtual machine and call it a cloud application. Application scalability would not be achieved without a scalable architecture. Cloud-native architectures like microservices are the ones that have these characteristics, therefore our choices as presented in Section 2.1.

At a high level, cloud native applications should be containerized, segmented into microservices and designed to be dynamically deployed. Consequently, it will allow them to be efficiently managed by orchestration systems like Kubernetes. Kubernetes is a container

orchestrator, designed by Google and maintained by the Cloud Native Computing Foundation (CNCF). Containers are a form of packaging applications with all their required dependencies and libraries in a portable and easily deployable format [5]. When implemented, these packages provide a stable, platform independent and predictable environment for the containerized application.

The steps to approach a cloud native/semi-native system are:

- Start containerizing the various components into portable and discretely deployable pieces in the new modules.
- Decomposing the existing business functions of the application into microservices and containerize them.
- Deploy the containers to the cloud, using a container orchestration tool such as Kubernetes.

3 IMPLEMENTATION

Although the NewDimensions project aims to address numerous technological areas, such as: cloud computing, software architectures, artificial intelligence and cybersecurity; this paper only addresses two of these points: cloud computing and software architectures, with its applicability in the development of the INOSS PIS Rail product.

INOSS PIS Rail (Integrated Network Operations Support System, Public Information System for Rail) is a pilot for the ongoing modernization that is taking place within NewDimensions project. Very briefly, there is a whole process of importing timetables and managing trains' circulation information, together with automatic, semi-automatic and manual modes of operation that needs to be improved. The PIS Rail service, as a new module, is the responsible for implementing the associated business logic.

Due to the high degree of complexity of the legacy system, and based on the principle of progressive modernization, the new modules were developed as individual structures that communicate with the existing system by well-defined interfaces. Therefore, following a modernization process like the one presented in the Section 2.2.

As can be seen in Fig. 3, between the new module and the legacy system two interfaces are placed. These interfaces make a bridge between the modules, adapting the languages and communication protocols, used by the legacy system, with the new module. This architecture allows the communication between modules with different languages and communication protocols.

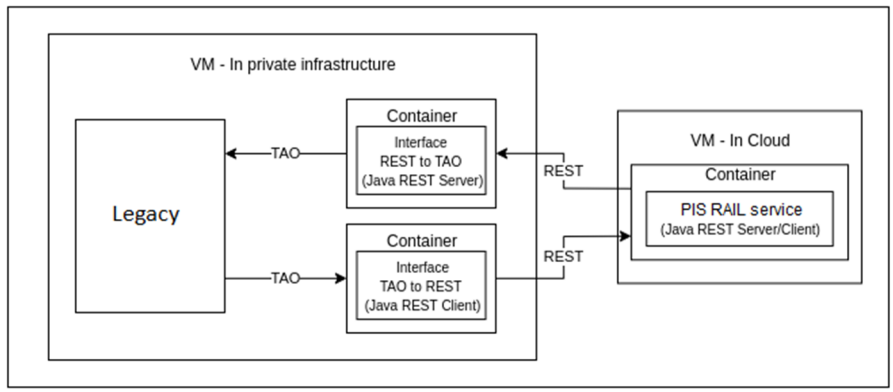

Figure 3: System architecture.

The new modules are packed in containers and, although initially hosted locally, in a final stage they can be hosted in the cloud, as can be seen in Fig. 3. The use of containers allows a better resource efficiency as compared to virtual machines, and an increased flexibility as an application management framework. Containerisation improves development and deployment aspects, such as testing and monitoring of container-based applications. The use of containers also gives us the capability to adjust cluster sizes for the deployment, have an easier maintenance and quicker deployments.

4 DISCUSSION

The implementation described in Section 3, allows the evolution of the legacy system without compromising anything already implemented. Using interfaces like the ones described, we can gradually replace old modules with new ones or keep using them as always.

The adoption of Java as programming language follows the actual development trends. Using microservices, teams can also work on the existing system without having to dive into legacy code. This also facilitates the integration of new developers into the software development cycle.

The product INOSS PIS, and in the scope of the project NewDimensions, is the target of a modernization process focusing on the protocols and technologies used in communication between modules. REST web services are the direct answer to an easy, clean and simpler maintenance communication between the system modules. The use of HTTP (Hypertext Transfer Protocol), in conjunction with REST, transforms the modules into easy pluggable independent services that can be invoked and may evolve easily. Additionally, by using this kind of communication protocols, it enables future integration with external services using APIs (Application Programming Interfaces).

Cloud-based architectures have taken an increasingly important role in building elastic systems with effective fault tolerance. Control and supervision systems, such as those intended to be modernized within NewDimensions project, need guarantees in its execution performance 24/7.

Cloud-based technologies do, in fact, offer quite a lot of benefits. We are able to take advantage of some of them from the moment the new modules are deployed in the cloud. Among several advantages, the cloud allows us to have an easier and more automatic scaling process and turning infrastructure management effortless to the new modules. Since it does not rely on hardware on-premises, or installed software in company servers, the cloud is essentially unlimited in its storage potential, turning the lack of storage space a problem of the past.

Applications have mainly evolved from huge monolithic structures, with fully coupled functions that only work on large physical machines with large computational resources, to fully running decoupled microservices, in a minimal container with high horizontal growth capacity. Therefore, as we can see in Fig. 3, every module or communication interface implemented is packed into an individual container each.

Containerisation granted a lightweight virtualisation through the creation of containers, as application packages that need less machine resources and time of deployment. They additionally address a more interoperable application packaging needed for portable software applications in the cloud [6]. Containerised modules give us the capability to develop, test and deploy applications to a large number of servers and to interconnect those containers.

These were the first steps to a new generation of EFACEC Transportation products. Products that will make use of cutting-edge software technology applied to the transportation business area. Taking this into account, a lot more is yet to come from the NewDimensions project. This evolution will continue, not only in the scope of the INOSS PIS Rail product

development but also in other projects. The next steps include implementing automation features, using artificial intelligence, and improvements that need to be addressed related to cybersecurity.

5 CONCLUSION

A modernization process such as the one explained in this the paper, must be a well-defined one. Such process is of extremely importance for the impact of changes on the entire system. We have shown that with a progressive modernization, besides the positive aspects, frequently one may face some drawbacks. As we have argued, technological incompatibilities may occur. In order to overcome the problem, we concluded that system architecture design techniques are the answer.

Our research also concludes that shifting to the cloud should be the next step to large business applications like EFACEC's operational control centre solutions. As a result, as we explained, a cloud-ready and good system's design is the key. The research also indicates that a system implemented using microservices and using containers can take more advantage of the cloud infrastructure.

Having this architecture implemented, we can build around the existing system without changing the legacy modules. This architecture allows us to make new and old modules communicate as if they were developed using the same technologies. Also, we can modernize the entire system, step by step, replacing old modules by new ones.

Future research on artificial intelligence and cybersecurity might shine the light on how to increase the capabilities and quality of EFACEC's OCC solutions. Therefore, those are the steps already underway in the NewDimensions project.

ACKNOWLEDGEMENTS

The work presented is under the scope of the project NewDimensions, being developed by EFACEC in cooperation with INESC TEC (Institute for Systems and Computer Engineering, Technology and Science) for state-of-the-art studies.

NewDimensions is supported by the operational programme COMPETE 2020, financed by PORTUGAL 2020, a partnership agreement between the Portuguese government and the European Commission.

REFERENCES

[1] Massé, M.H., *REST API Design Rulebook*, 2011.
[2] Newman, S., *Building Microservices: Designing Fine-Grained Systems*, 25 Dec. 2014.
[3] Advances in service-oriented and cloud computing, Workshops of *ESOCC 2015*, Taormina, Italy, 15–17 Sep. 2015.
[4] Seacord, R.C., Comella-Dorda, S. Lewis, G., Place, P. & Plakosh, D., *Legacy System Modernization Strategies*, Jul. 2001.
[5] Jetha, H. & Tagliaferri, L., *Running Cloud Native Applications on Digital Ocean Kubernetes*, White Paper, Jul. 2018.
[6] Pahl, C., Brogi, A., Soldani, J. & Jamshidi, P., *Cloud Container Technologies: A State-of-the-Art Review*, May 2017.