Towards customizing multi-tenant Cloud applications using non-intrusive microservices*

Phu H. Nguyen[1], Hui Song[1], Franck Chauvel[1], and Erik Levin[2]
[1] SINTEF, Oslo, Norway
[2] Visma, Oslo, Norway
first.last@sintef.no, first.last@visma.com

Abstract

The customization requirements from different customers are often beyond what software vendors can offer in advance, at design-time. Enabling tenant-specific customizations in cloud-based multi-tenant Software-as-a-Service (SaaS) requires a novel approach. This paper proposes a Microservice-architecture-based non-intrusive Customization framework (MiSC-Cloud) to enable deep customization of SaaS.

1 Introduction

Cloud-based multi-tenant enterprise Software-as-a-Service (SaaS) cannot be customized directly for each customer because the same instance of code is shared by multiple customers at runtime. Using microservices is a promising way to customize multi-tenant cloud software because microservices architectures offer several benefits. First, microservices for customisation purposes can be packaged and deployed in isolation from the main product, which is an important requirement for multi-tenant context. Moreover, independent development and deployment of microservices ease the adoption of continuous integration and delivery, and reduce, in turn, the time to market for each service. Independence also allows engineers to choose the technology that best suits one and only one service, while other services may use different programming languages, database, etc. Each service can also be operated independently, including upgrades, scaling, etc.

We have experimented with the customization approach using "intrusive microservices" [1, 2]. Intrusive microservices solution means the main body of custom code runs in a separate microservice, isolated from the main service (of the main product), whilst specific parts of the custom code are sent back to the main product and dynamically compiled and executed within the execution context of the main service. While intrusive microservices are technically sound, its practical adoption by industry may be hindered by the intrusive way of custom code, which would be developed by “third-parties”

* This work is supported by the Cirrus project https://www.sintef.no/en/digital/software-and-service-innovation/secure-iot-software/cirrus/

that cannot be trusted by the software vendor (SV) to be dynamically compiled and executed within the execution context of the main service. We are working on a Microservice-architecture-based non-intrusive Customization framework for multi-tenant Cloud applications, called MiSC-Cloud.

2 Non-intrusive Customization using microservices

Figure 1 shows an overview of the MiSC-Cloud framework. In this architecture, the non-intrusive MiSC-Cloud solution avoids using call-back code for customization and rather orchestrates customizations using the API Gateway to which the API of the main product and the APIs of the microservices implementing the customizations are exposed.

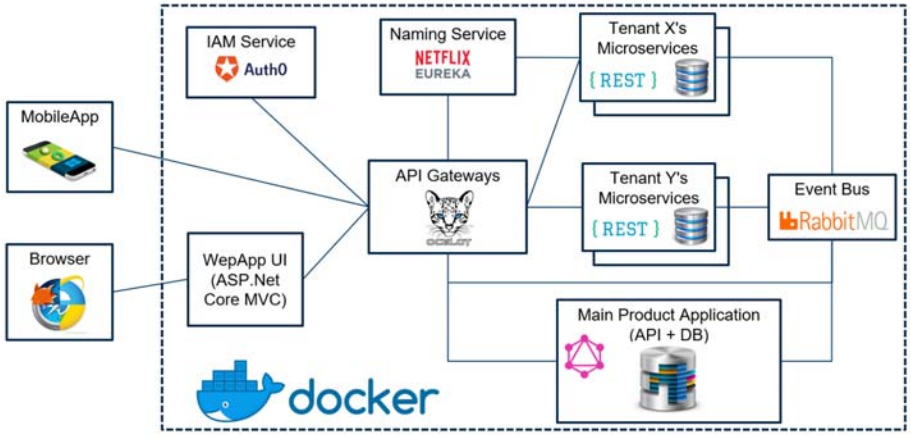

Figure 1: An overview of the MiSC-Cloud framework

After being tested and approved, the microservices of "tenants" are deployed and controlled by the SV (e.g., Visma) for customizing the main product. Figure 1 shows microservices specific for tenant X and tenant Y to customize the main product according to the tenants' needs. Only the end-users of tenant X after having logged in the system will have access to the customized features specific for tenant X. Via web browsers, end-users can interact with the main product and the corresponding microservices for customization via the API Gateway (e.g., Ocelot). If end-user has not logged in, the API Gateway redirects the end-user to the Identity and Access Management (IAM) service. After the end-user logged in successfully via the IAM service, its identity, tenant ID, and access tokens to the main product and the corresponding registered customizations will be used by the API Gateway to orchestrate the interactions of the end-user with the main product (empowered by GraphQL) and the specific microservices available for the end-user. The API Gateway leverages a Naming service (e.g., Netflix Eureka) to get the instances of the provided microservices. The API Gateway also leverages a message queue server such as RabbitMQ to support for its orchestration tasks among the main product and the microservices, which are deployed in separate Docker containers.

References

1. Song, H., F. Chauvel, and A. Solberg. *Deep customization of multi-tenant SaaS using intrusive microservices*. in *Proceedings of the 40th International Conference on Software Engineering: New Ideas and Emerging Results*. 2018. ACM.
2. Chauvel, F. and A. Solberg, *Using intrusive microservices to enable deep customization of multi-tenant SaaS*, in *QUATIC*. 2018, IEEE.