The SmartOrchestra Platform: A Configurable Smart Service Platform for IoT Systems

Andreas Liebing[1], Lutz Ashauer[1], Uwe Breitenbücher[2], Thomas Günther[3], Michael Hahn[2], Kálmán Képes[2], Oliver Kopp[2], Frank Leymann[2], Bernhard Mitschang[2], Ana C. Franco da Silva[2], Ronald Steinke[3]

[1]StoneOne AG, Berlin, Germany
andreas.liebing@stoneone.de

[2]University of Stuttgart, Stuttgart, Germany
[firstname.lastname]@informatik.uni-stuttgart.de

[3]Fraunhofer FOKUS, Berlin, Germany
[firstname.lastname]@fokus.fraunhofer.de

The SmartOrchestra Platform: A Configurable Smart Service Platform for IoT Systems

Andreas Liebing[1], Lutz Ashauer[1], Uwe Breitenbücher[2], Thomas Günther[3], Michael Hahn[2], Kálmán Képes[2], Oliver Kopp[2], Frank Leymann[2], Bernhard Mitschang[2], Ana C. Franco da Silva[2], and Ronald Steinke[3]

[1] StoneOne AG, Berlin, Germany
andreas.liebing@stoneone.de
[2] University of Stuttgart, Stuttgart, Germany
[firstname.lastname]@informatik.uni-stuttgart.de
[3] Fraunhofer FOKUS, Berlin, Germany
[firstname.lastname]@fokus.fraunhofer.de

Abstract. The Internet of Things is growing rapidly while still missing a universal operating and management platform for multiple diverse use cases. Such a platform should provide all necessary functionalities and the underlying infrastructure for the setup, execution and composition of Smart Services. The concept of Smart Services enables the connection and integration of cyber-physical systems (CPS) and technologies (i.e., sensors and actuators) with business-related applications and services. Therefore, the *SmartOrchestra Platform* provides an open and standards-based service platform for the utilization of public administrative and business-related Smart Services. It combines the features of an operating platform, a marketplace, a broker, and a notary for a cloud-based operation of Smart Services. Thus, users of cyber-physical systems are free to choose their control applications, no matter what device they are using (e.g., smartphone, tablet or personal computer) and they also become independent of the manufacturers' software. This will enable new business opportunities for different stakeholders in the market and allows flexibly composing Smart Services.

Keywords: SmartOrchestra Platform, Smart Services, Cyber-Physical Systems, Internet of Things

1 Introduction

The Internet of Things (IoT) paradigm has received great attention in the last years leading to a vast amount of heterogeneous IoT middleware, protocols and devices (e.g., sensors, actuators or gateways). Related IoT applications, for example, implementing the processing of sensor data in order to control an actuator, therefore need to be bound to certain concrete technologies, hardware devices, and protocols. This makes it a challenge to enable the interoperability and composition of different IoT applications, for example, to compose them to solve problems on another scale (e.g., from automating houses over streets to cities). Another challenge is the distributed nature of IoT environments and the large number of devices, which makes it infeasible to deploy and manage IoT applications together with their required software and middleware components manually. Therefore, a universal operating and management platform for multiple diverse IoT use cases is needed which enables interoperability and automated deployment for IoT applications through Smart Services. The concept of Smart Services enables the connection and integration of cyber-physical systems (CPS) and technologies (i.e., sensors and actuators) with business-related applications and services. Such a platform should provide all necessary functionalities and the underlying infrastructure for the setup, execution and composition of Smart Services. Within this work, we introduce the ***SmartOrchestra Platform***, which provides an open and standards-based service platform for the utilization of public administrative and business-related Smart Services. Therefore, it combines the features of an operating platform, a marketplace, a broker, and a notary for a cloud-based operation of Smart Services. Thus, users of cyber-physical systems are free to choose their control applications, no matter what device they are using (e.g., smartphone, tablet, or personal computer) and they also become independent of the manufacturers' software. This will enable and provide new business opportunities for different stakeholders in the market and allows flexibly utilizing and composing Smart Services.

The remainder of the paper starts with an introduction of the SmartOrchestra Platform and its major building blocks in Sect. 2. This is followed by a description how the introduced components of the platform work together to provide a universal operating and management platform for multiple diverse IoT use cases in Sect. 3. Finally, a summary of the paper is given in Sect. 4.

2 The SmartOrchestra Platform

The SmartOrchestra Platform enables a uniform service description as well as a secure and safe internet-based composition and integration of heterogeneous cyber-physical systems and services based on standardized cloud and orchestration technologies. The platform serves both, a transparent catalog to evaluate suitable services from a widespread ecosystem as well as an operational platform and interface between control devices and sensor units with their respective applications. In this way, the platform will be an open, secure, and standardized Smart Service Platform.

The conceptual design of the SmartOrchestra Platform and its major building blocks in combination with provisioning workflows and IoT devices is depicted in Fig. 1. The main entry point to the platform is the Marketplace, which allows users to evaluate, run and compose existing services from the Service Catalog as well as provide and market new services. To deploy and configure Smart Services OpenTOSCA [2] is used as Provisioning Engine. The Provisioning Engine is responsible for the automated deployment of a Smart Service and the configuration of its underlying infrastructure. For example, this can comprise the installation of related software services in the Cloud providing the business logic of a Smart Service (e.g., data filtering, processing, or aggregation [13]), the configuration of IoT devices and gateways, and installing required software on them.

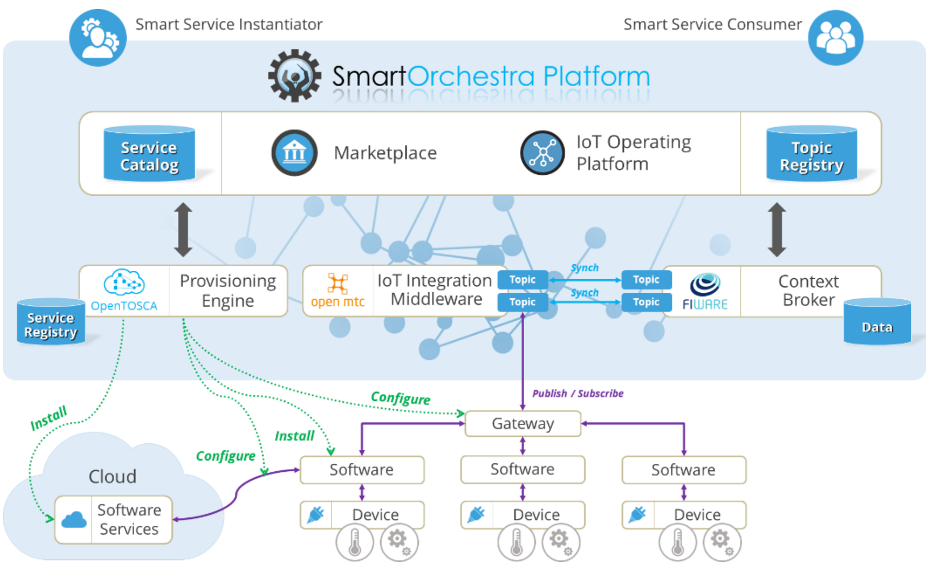

Fig. 1. SmartOrchstra Platform Architecture

For the integration of IoT devices into Smart Services, the SmartOrchestra Platform uses OpenMTC [6, 15] as IoT Integration Middleware [16, 17]. OpenMTC provides required protocols and adapters to integrate and mediate between the heterogeneous devices, sensors, and actuators within the SmartOrchestra Platform. Therefore, it comes with an embedded service layer that enables communication between devices through a Publish/Subscribe model. Furthermore, the platform provides the FIWARE Orion Context Broker [9] as Context Broker. While OpenMTC is responsible for enabling the communication between devices, sensors and actuators through corresponding generic interfaces, the FIWARE Orion Context Broker in combination with the FIWARE Short Time Historic [10] is used as a midterm data repository to enable later analysis of data provided by IoT devices. Therefore, OpenMTC synchronizes all published data to respective entities at the FIWARE broker as depicted by the synch arrows in Fig. 1.

In the following, each of the building blocks of the platform is described in more detail. Furthermore, the interplay of the components is outlined in order to give an idea how the overall SmartOrchestra Platform operates.

2.1 Marketplace and IoT Operating Platform

The open and secure SmartOrchestra marketplace brings together and combines intelligent private, industrial, and municipal Smart Services. This results in new innovative services that make data available for use. The marketplace allows for browsing, choosing, and configuring of Smart Services. Data of each running service can also be accumulated and visualized in the marketplace via customer specific dashboards including widgets and configurable taxonomies for structured presentation. Smart Services and/or data channels can be combined and orchestrated by rules and actions. The parameters of devices or services can be changed during runtime of the service. This turns the marketplace into an IoT Operating Platform.

2.2 Provisioning Engine: OpenTOSCA

The SmartOrchestra Platform enables the deployment of Smart Services by employing OpenTOSCA [2, 12] as Provisioning Engine. OpenTOSCA is an open-source ecosystem for the modeling, provisioning, and management of cloud applications based on the OASIS Topology and Orchestration Specification for Cloud Applications (TOSCA) standard [1, 3, 21] and supporting concepts for CPS and IoT [19, 22–24]. In TOSCA, the structure of an application, e.g., a Smart Service, is described using topology models. These models are represented as graphs containing typed nodes and directed typed edges. Nodes, called *node templates*, represent the software components of a Smart Service, while edges, called *relationship templates*, describe the relationships among the components, e.g., dependencies and connections.

TOSCA offers two approaches to application provisioning: (i) a declarative approach and (ii) an imperative approach [4, 5]. In the declarative approach, only the topology model has to be provided to a provisioning engine, which implicitly knows how to set up the application components. More precisely, it is sufficient to describe what needs to be provisioned, and not how this needs to be done. In contrast, the imperative approach relies on explicitly describing how an application has to be hosted. To realize this, a so called Build Plan has to be provided that describes, which steps have to be executed to set up the components of the topology.

The OpenTOSCA ecosystem is composed of following: (i) the graphical TOSCA modeling tool Eclipse Winery [20] and (ii) the TOSCA runtime environment OpenTOSCA container. Once the topology of a Smart Service is modeled using Eclipse Winery, it can be optionally checked against a collection of compliance rules [7] and exported as a Cloud Service Archive (CSAR), which can be deployed into the OpenTOSCA container to provision and instantiate the Smart Service. The provisioning of Smart Services can be secured by the specification of non-functional requirements through policies [18].

OpenTOSCA allows for an easy integration with other systems through its provided APIs, which offer the main functionalities to automatically provision applications using the OpenTOSCA container, and afterwards, retrieve information about the instantiated applications. The Marketplace, which is the user's entry point to the SmartOrchestra Platform, is integrated with OpenTOSCA through the provided OpenTOSCA API. In this way, the Marketplace can, for example, configure and automatically provision Smart Services, and retrieve information about all available Smart Services.

2.3 IoT Integration Middleware: OpenMTC

The OpenMTC platform [6, 15] is an open-source implementation of the oneM2M standard[1], which intends to support machine-to-machine (M2M) communication for applications in a simplified way. Furthermore, OpenMTC is as well available as a Generic Enabler in the FIWARE Catalogue [14].

OpenMTC consists of a gateway and backend component that provides a REST API and uses the CRUD principle for managing resources. Through protocol adapters, OpenMTC is able to interact with various devices of different technologies. Thus, information from heterogeneous data sources will be unified in a harmonized data model, so that applications can easily access the data without the need to know the device specific technologies. Furthermore, data can be already preprocessed close to the source before they are send to other endpoints.

OpenMTC has a generic request/response model that can be mapped on different transport protocols, e.g., HTTP or MQTT. The provided functionality includes registration of applications, discovery of resources, subscription to new data, simplified addressing of resources, scheduled communication, and more [11].

2.4 Context Broker: FIWARE Orion Context Broker

The Orion Context Broker [9] is the main component of the FIWARE platform [8]. Through its REST API, the Broker allows the registration of *context elements*, which can be updated by *context producers*. Furthermore, *context consumers* can either query these context elements or subscribe to them to get notifications when the context elements are updated [11]. The Orion Context Broker can be configured through the marketplace to automatically work together with another FIWARE Generic Enabler, the FIWARE Short Time Historic [10], which is used for midterm storage of data.

3 Interplay of the SmartOrchestra Platform Components

In this section, we describe the interaction with the platform and the interplay of the platform components. Consequently, the interplay is described based on the different roles that interact with the SmartOrchestra Platform (cf. Fig. 1): Smart Service Instantiators and Smart Service Consumers.

The role of a *Smart Service Instantiator* is to provision different Smart Services from within the *Service Catalog* using the provided interfaces given by the IoT Operating Platform. These services can then be used by the Smart Service Consumers for their own applications. The Smart Service Instantiator is responsible for providing relevant data, such as credentials and endpoints, to enable access to the *IoT Devices* for the Provisioning Engine and therefore to enable the installation of different software components. To realize the integration of the IoT Devices into the platform, these software components, such as *adapters* and *gateways*, are responsible for binding the used hardware to the IoT Integration Middleware by sending the relevant data from the sensors

[1] http://www.onem2m.org/

and enabling the invocation of operations on actuators. As stated in the previous section, the IoT Integration Middleware is used as the first layer to integrate heterogeneous IoT hardware, while the Context Broker component is used as a second layer to store data from the sensors for midterm data analysis within the SmartOrchestra Platform.

The synchronization of the data across the IoT Integration Middleware and the Context Broker is based on a bi-directional exchange between respective entities managed by each of the components. These entities are automatically created when the provisioning of the software components is finished and therefore require no additional effort from the Smart Service Instantiator. After the provisioning and as soon as OpenMTC and the FIWARE broker receive data from sensors and actuators, a *Smart Service Consumer* is able to subscribe on the available data or issue commands through corresponding topics provided by the *Topic Registry* of the platform. Based on that, Smart Service Consumers are able to integrate Smart Services operated and managed through the SmartOrchestra Platform into their business applications.

4 Summary

This work presented the SmartOrchestra Platform as an enabler for interoperability, automated deployment and composition of IoT applications through Smart Services. Thereby, the different components of the platform build on well-established standards such as TOSCA for application provisioning and management or oneM2M for machine-to-machine communication and IoT. The result is an open and standards-based platform for the utilization of public administrative and business-related Smart Services by combining the features of an operating platform, a marketplace, a broker, and a notary for a cloud-based operation of Smart Services.

Acknowledgments This work is funded by the BMWi project SmartOrchestra (01MD16001F).

References

1. Bergmayr, A., Breitenbücher, U., Ferry, N., Rossini, A., Solberg, A., Wimmer, M., et al.: A Systematic Review of Cloud Modeling Languages. ACM Computing Surveys. 51, 1, (2018).
2. Binz, T., Breitenbücher, U., Haupt, F., Kopp, O., Leymann, F., Nowak, A., et al.: OpenTOSCA – A Runtime for TOSCA-based Cloud Applications. In: ICSOC. (2013).
3. Binz, T., Breiter, G., Leymann, F., Spatzier, T.: Portable Cloud Services Using TOSCA. IEEE Internet Computing. 16, 03, (2012).
4. Breitenbücher, U., Binz, T., Képes, K., Kopp, O., Leymann, F., Wettinger, J.: Combining Declarative and Imperative Cloud Application Provisioning based on TOSCA. In: IC2E. IEEE (2014).

5. Breitenbücher, U., Képes, K., Leymann, F., Wurster, M.: Declarative vs. Imperative: How to Model the Automated Deployment of IoT Applications? In: Summer-SOC. IBM Research Report (2017).
6. Corici, M., Coskun, H., Elmangoush, A., Kurniawan, A., Mao, T., Magedanz, T., et al.: OpenMTC: Prototyping Machine Type communication in carrier grade operator networks. In: IEEE Globecom Workshops. (2012).
7. Fischer, M.P., Breitenbücher, U., Képes, K., Leymann, F.: Towards an Approach for Automatically Checking Compliance Rules in Deployment Models. In: SECURWARE. Xpert Publishing Services (2017).
8. FIWARE: FIWARE Catalogue, https://www.fiware.org/developers/catalogue/.
9. FIWARE: Orion Context Broker, https://www.github.com/telefonicaid/fiware-orion.
10. FIWARE: Short Time Historic (STH) – Comet, https://github.com/telefonicaid/fiware-sth-comet.
11. Franco da Silva, A.C., Breitenbücher, U., Hirmer, P., Képes, K., Kopp, O., Frank Leymann, et al.: Internet of Things Out of the Box: Using TOSCA for Automating the Deployment of IoT Environments. In: CLOSER. (2017).
12. Franco da Silva, A.C., Breitenbücher, U., Képes, K., Kopp, O., Leymann, F.: OpenTOSCA for IoT: Automating the Deployment of IoT Applications based on the Mosquitto Message Broker. In: IoT. ACM (2016).
13. Franco da Silva, A.C., Hirmer, P., Breitenbücher, U., Kopp, O., Mitschang, B.: Customization and provisioning of complex event processing using TOSCA. Computer Science - Research and Development. (2017).
14. Fraunhofer FOKUS: OpenMTC Generic Enabler, https://catalogue.fiware.org/enablers/openmtc.
15. Fraunhofer FOKUS: OpenMTC Platform Architecture, http://www.openmtc.org/index.html#MainFeatures.
16. Guth, J., Breitenbücher, U., Falkenthal, M., Fremantle, P., Kopp, O., Leymann, F., et al.: A Detailed Analysis of IoT Platform Architectures: Concepts, Similarities, and Differences. Presented at the (2018).
17. Guth, J., Breitenbücher, U., Falkenthal, M., Leymann, F., Reinfurt, L.: Comparison of IoT Platform Architectures: A Field Study based on a Reference Architecture. In: CIoT. IEEE (2016).
18. Képes, K., Breitenbücher, U., Fischer, M.P., Leymann, F., Zimmermann, M.: Policy-Aware Provisioning Plan Generation for TOSCA-based Applications. In: SECURWARE. Xpert Publishing Services (2017).
19. Képes, K., Breitenbücher, U., Leymann, F.: Integrating IoT Devices Based on Automatically Generated Scale-Out Plans. In: SOCA. IEEE (2018).
20. Kopp, O., Binz, T., Breitenbücher, U., Leymann, F.: Winery – A Modeling Tool for TOSCA-based Cloud Applications. In: ICSOC. (2013).
21. OASIS: Topology and Orchestration Specification for Cloud Applications (TOSCA) Version 1.0. (2013).
22. Saatkamp, K., Breitenbücher, U., Leymann, F., Wurster, M.: Generic Driver Injection for Automated IoT Application Deployments. In: iiWAS. ACM (2017).

23. Franco da Silva, A.C., Hirmer, P., Breitenbücher, U., Kopp, O., Mitschang, B.: TDLIoT: A Topic Description Language for the Internet of Things. In: ICWE. Springer Berlin Heidelberg (2018).
24. Zimmermann, M., Breitenbücher, U., Leymann, F.: A TOSCA-based Programming Model for Interacting Components of Automatically Deployed Cloud and IoT Applications. In: ICEIS. SciTePress (2017).

All links were last followed on July 18, 2018.