SiDD: The Situation-Aware Distributed Deployment System

Kálmán Képes, Frank Leymann, Benjamin Weder, and Karoline Wild

Institute of Architecture of Application Systems, University of Stuttgart, Germany
{kepes,leymann,weder,wild}@iaas.uni-stuttgart.de

Abstract. Most of today's deployment automation technologies enable the deployment of distributed applications in distributed environments, whereby the deployment execution is centrally coordinated either by a central orchestrator or a master in a distributed master-workers architectures. However, it is becoming increasingly important to support use cases where several independent partners are involved. As a result, decentralized distributed deployment automation approaches are required, since organizations typically do not provide access to their internal infrastructure to the outside or leave control over application deployments to others. Moreover, the choice of partners can depend heavily on the current situation at deployment time, e.g. the costs or availability of resources. Thus, at deployment time it is decided which partner will provide a certain part of the application depending on the situation. To tackle these challenges, we demonstrate the situation-aware distributed deployment (SiDD) system as an extension of the OpenTOSCA ecosystem.

Keywords: Deployment, Choreography, Situation-aware System, TOSCA

1 Introduction and Motivation

Deployment technologies enable reusable and portable application deployments, making them key technologies for today's application management. A variety of technologies offer different capabilities and own domain-specific languages for modeling deployments. Many use declarative deployment models in which the desired state of an application can be specified by a structural description of the application with its components and their relationships among each other.

However, in recent years, deployment automation has focused primarily on centralized approaches, which allow the deployment of distributed applications, but the deployment execution is centrally coordinated either by a central orchestrator or a master in distributed master-workers architectures. Especially in industrial use cases, e. g., in a supply chain, or if specialized compute infrastructure is required, e. g., in quantum computing, several partners are involved each deploying a part of the overall application. Due to security concerns, organizations typically do not provide access to internal infrastructure to the outside or leave control over application deployments to others. Thus, centralized deployment approaches

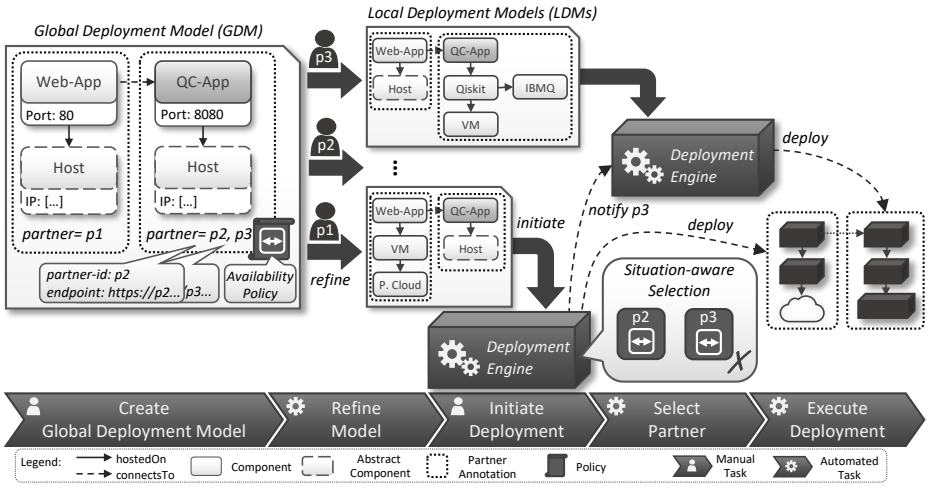

Fig. 1. Overview of the SiDD concept exemplary depicting the partner selection based on the available capacities of the partners (p2 and p3).

cannot be applied. Moreover, the involved partners can change depending on certain conditions, e. g., costs or availability. Thus, (i) the modeling of a distributed application involving multiple partners, (ii) the partner selection during deployment, and (iii) the decentralized execution of the overall deployment has to be enabled. To tackle these challenges, we present the *Situation-aware Distributed Deployment (SiDD) system* as an extension of the OpenTOSCA ecosystem [1], an open-source toolchain for modeling and executing application deployments using the Topology and Orchestration Specification for Cloud Applications (TOSCA).

2 Exemplary Application Scenario and SiDD Concept

In previous work, a concept for the *decentralized cross-organizational application deployment automation* [3] as well as the *situation-aware management* [2] has been introduced. In this work, we demonstrate how these concepts can be combined to enable the situation-aware partner selection for a distributed and decentralized deployment based on an exemplary scenario as shown in Fig. 1 on the left: Three partners *p1*, *p2*, and *p3* collaborate to run an application using a quantum algorithm and a visualization web application component. To reduce costs the partners share their infrastructure, p1 provides a classical private cloud, p2 a high-performance computing environment (HPCE) to run a quantum simulator, and p3 provides access to a quantum computer. At deployment time it has to be decided how the quantum algorithm shall be executed either using the quantum computer of p3 or running a quantum simulator in the HPCE of p2. This selection decision has to be made based on an availability policy, e. g., if there is no available time slot on the quantum computer, the quantum simulator in the HPCE is selected and vice versa.

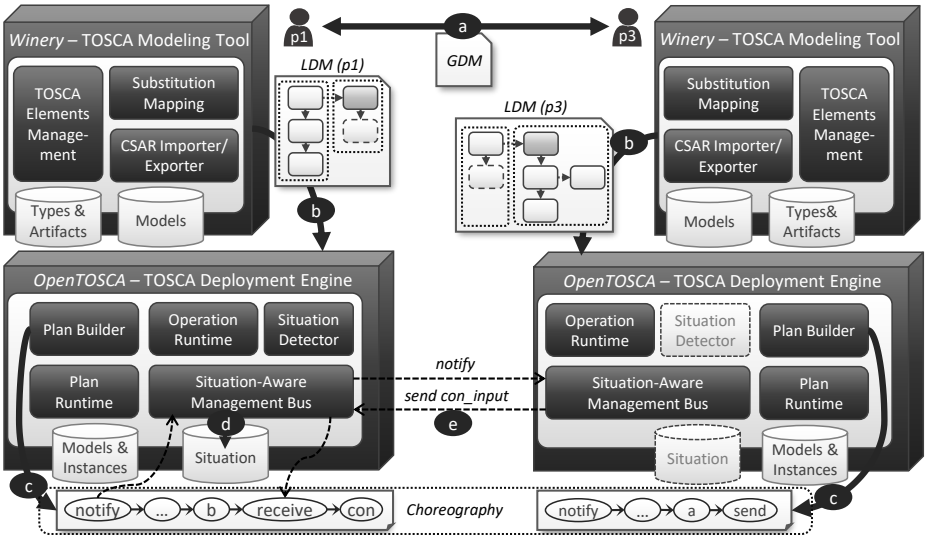

Fig. 2. Architecture of the SiDD system exemplary shown with two partners.

The described scenario can be partially automated with our SiDD concept depicted in Fig. 1. First, a so-called *global deployment model (GDM)* has to be specified as a declarative deployment model that contains all application components and their relationships (see left in Fig. 1). This model only contains the application components on which all partners have agreed upon and not necessarily contain any infrastructure components such as virtual machines or application servers. In addition, a GDM contains abstract components that are replaced by concrete infrastructure components by each partner. Thus, in the second step, each partner refines the GDM into a so-called *local deployment model (LDM)* which specifies the needed infrastructure components of their own infrastructure, as other partners usually do not have or must not have knowledge about these. The refinement can be automated, e. g., using available refinement fragments [4]. After all partners have defined their LDMs, in the third step, any partner can initiate deployment by requesting the deployment engine to start a deployment. In the fourth step, it is decided which partners are actually involved in the deployment, e. g., in our scenario the infrastructure will be used from p1 and additionally either from p2, which can run a quantum simulator, or from p3, which has access to a quantum computer. This is based on the annotated policies, such as the availability policy in the GDM in Fig. 1. For example, if the quantum computer of p3 is not available at deployment time, a quantum simulator can be deployed in the HPCE of p2 if there are enough resources available. Finally, the selected partners are notified to start the deployment of their components. For the deployment each partner generates a workflow containing tasks to install components as well as tasks to exchange data between the partners, e. g., endpoint information to establish a connection to components of other partners.

3 The SiDD System

The SiDD System is an extension of the OpenTOSCA ecosystem that consists of *Winery*, a graphical TOSCA modeling tool, and *OpenTOSCA container*, a TOSCA deployment engine [1] (see video at `https://youtu.be/AOJY9TW4ZFM`). The architecture of the system with the relevant components is shown in Fig. 2. *Winery* can be used to graphically model a declarative deployment model as a TOSCA *topology template* by using defined types and attaching the executable artifacts, e. g., a WAR for running a web application. Winery is used to model the GDM, which is then passed to the involved partners (a). The *CSAR Importer/ Exporter* enables the export of a standardized *Cloud Service Archive (CSAR)* that can be consumed by a TOSCA deployment engine. The *Substitution Mapping* Component can be used to refine abstract components, called node templates, by concrete ones using refinement fragments. This is used by each partner to refine the abstract node templates in the GDM and to obtain the LGM that can be processed by the deployment engine *OpenTOSCA container* (b). For deployment execution, a BPEL workflow is generated based on the declarative deployment model by the *Plan Builder* (c). The workflows of the partners form a choreography by sending and receiving messages to share deployment data. The *Plan Runtime* runs the plan when the deployment is instantiated. All operations that have to be executed and which are not provided as a service run in the *Operation Runtime*. For the situation-aware selection, the required information has to be provided by an external application which is then used by the *Situation Detector* to determine which situations are active. When an application is instantiated, the *Situation-Aware Management Bus* is responsible for the partner selection based on the current situation (d) and to exchange messages during deployment (e).

Acknowledgments This work was partially funded by the DFG project DiStOPT (252975529), the BMWi project *PlanQK* (01MK20005N), and the DFG's Excellence Initiative project *SimTech* (EXC 2075 - 390740016).

References

1. Breitenbücher, U., et al.: The OpenTOSCA Ecosystem - Concepts & Tools. European Space project on Smart Systems, Big Data, Future Internet - Towards Serving the Grand Societal Challenges - Volume 1: EPS Rome 2016 pp. 112–130 (Dec 2016)
2. Képes, K., et al.: Situation-Aware Management of Cyber-Physical Systems. In: Proceedings of the 9th International Conference on Cloud Computing and Services Science (CLOSER 2019). pp. 551–560. SciTePress (May 2019)
3. Wild, K., et al.: Decentralized Cross-Organizational Application Deployment Automation: An Approach for Generating Deployment Choreographies Based on Declarative Deployment Models. In: Proceedings of the 32nd Conference on Advanced Information Systems Engineering (CAiSE). pp. 20–35. Springer International Publishing (2020)
4. Wild, K., et al.: TOSCA4QC: Two Modeling Styles for TOSCA to Automate the Deployment and Orchestration of Quantum Applications. In: 2020 IEEE 24th International Enterprise Distributed Object Computing Conference (EDOC). IEEE Computer Society (2020)