A Domain-Specific Modeling Tool to Model Management Plans for Composite Applications

Oliver Kopp[1], Tobias Binz[2], Uwe Breitenbücher[2],
Frank Leymann[2], and Thomas Michelbach[2]

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

Abstract TOSCA is a standard to describe composite Cloud-applications and their management in a portable fashion. Thereby, BPMN4TOSCA is a proposed extension for BPMN to ease modeling of management plans. This demonstration presents a web-based modeling tool that supports an updated version of BPMN4TOSCA. The updated version supports direct wiring of data of tasks and events without the need of separate data objects.

1 Introduction

The *Topology and Orchestration Specification for Cloud Applications* (TOSCA [6]) is an OASIS standard for automating provisioning, management, and termination of applications in a portable and interoperable way. To enable this, TOSCA employs two concepts: (i) application topologies and (ii) management plans. An application topology describes the software and hardware components involved in an application and the relationships between them. Management plans capture knowledge to deploy and manage an application and are typically modeled as BPMN or BPEL workflows.

The current OpenTOSCA toolchain starts with Winery [4]. Here, a user specifies required types and models the application topology. Currently, management plans have to be created manually and uploaded as archive into Winery. The whole application package including the topology, required types and the management plans is exported as CSAR and imported in the OpenTOSCA runtime [1]. The runtime deploys the CSAR and adds the application to the Vinothek [2]. Therein, a user can instantiate an application with one click.

In this paper, we present tool support for modeling management workflows in Winery. It is based on BPMN4TOSCA, a domain-specific extension of BPMN to ease the modeling of management plans [3]. BPMN4TOSCA consists of four new elements: The TOSCA Topology Management Task, the TOSCA Node Management Task, the TOSCA Script Task, and the TOSCA Data Object. The TOSCA Topology Management Task is used to interact with the OpenTOSCA runtime. For instance, to retrieve the current service template. The TOSCA Node Management Task is used to invoke management operations on a node. The TOSCA Script Task is used to invoke a script on a node. The TOSCA Data

Object provides integration of properties of nodes and relationships as data. If data should be read from or written to a property of a node or relationship, the TOSCA Topology Management can be used or directly a TOSCA Data Object.

We developed a prototypical BPMN4TOSCA-based modeler [3]. In the meanwhile, we improved the OpenTOSCA container to natively support script execution [8] and, therefore, the TOSCA Script Task is obsolete. Furthermore, we discovered that BPMN4TOSCA is still too complex due to directly visible data dependencies; especially if complex application structures have to be handled in non-trivial management plans. Therefore, we reduced the complexity by replacing the TOSCA Data Object, which is the reason for the increased complexity, by a direct wiring of data dependencies between (i) tasks and (ii) properties of the components and relations in the TOSCA model.

First, in BPMN4TOSCA management tasks are modeled by specifying the component and the operation to be invoked on this entity. In the revised version of BPMN4TOSCA, input parameters can be directly wired with output parameters of former tasks. Thus, there is no need for an explicit data object. Although this is a common way to specify data flows, BPMN4TOSCA provides a new means to ease accessing properties of the components and relations described in the application topology model: To specify the input parameter value of an operation invocation, a property of an application component or relation in the topology can be referenced directly. As a consequence, if the component or relation is not instantiated when invoking this operation, the specified property is extracted out of the TOSCA model. If the component or relation is already running, the property is taken from the instance model of the application that holds current runtime information about each component and relation of the application. The same way, properties can be specified for output parameters: After the invocation, the output is written into the specified property of the instance model. Thus, the improved version of BPMN4TOSCA simplifies modeling data flow significantly by enabling a direct access to the instance model of the application, which changes the data flow from a task-centric perspective to a topology-plan perspective.

The extension itself does not follow the idea of data transfer in BPMN. We believe, however, that our extension is in line with the use of BPMN in executable workflows. For instance, camunda BPM also does not use data objects, but uses a global hashmap to store data [7].

Existing BPMN Modeling tools such as bpmn.io[1], Signavio[2], or the web-based modeler of Stardust[3] neither provide a tight integration with application topologies nor allow for a direct wiring of data dependencies. Therefore, we tightly integrated a plan modeler supporting the new version of BPMN4TOSCA in Winery [4], a web-based modeling tool for TOSCA-based application descriptions. In the next section, we describe the modeling of a TOSCA management plan using the presented concept and plan modeler tool. In Sect. 3 we outline next steps.

[1] `http://bpmn.io/`
[2] `http://www.signavio.com/`
[3] `https://www.eclipse.org/stardust/`

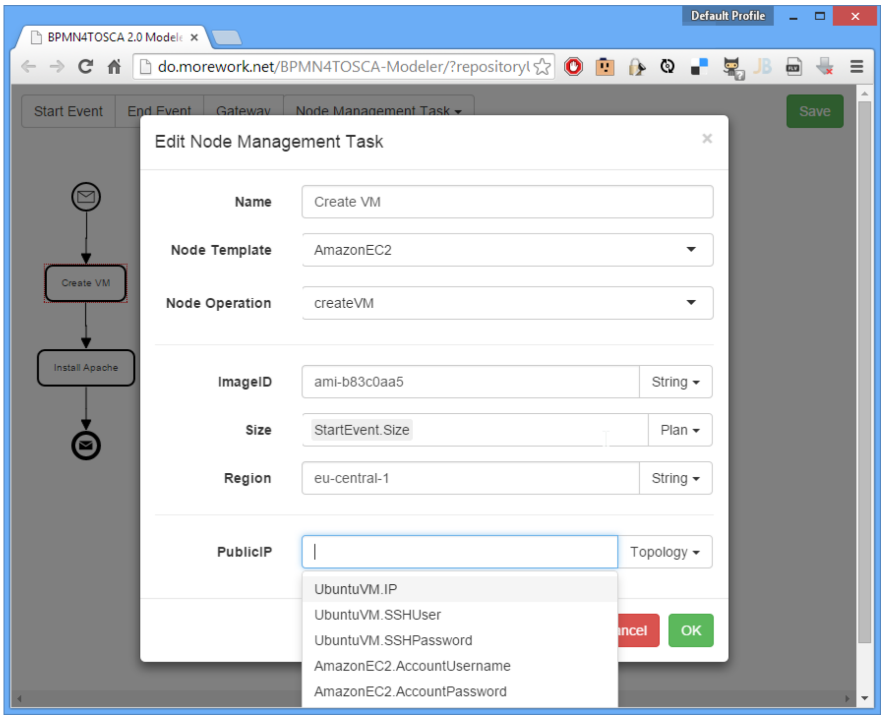
2 Modeling a Management Plan in BPMN4TOSCA 2.0

We call the improved version of our management-specific workflow language *BPMN4TOSCA 2.0*. The modeling of a management plan in TOSCA basically encompasses two tasks: (i) specifying the application topology model and (ii) modeling the management plans, for example, a so-called *build plan* that deploys the application. To model the topology, Winery provides a repository of available node and relationship types, for example, an *UbuntuVirtualMachine* node type and *hostedOn* relationship type. These types typically specify available operations, for example, the VM type provides an operation *runScript* to execute low level management commands. Afterwards, management plans can be modeled. To describe management logic, Node Management Tasks specify which operation shall be invoked on which node or relation. Based on that, the input and output parameters are shown. For the input parameter, there is the choice between String, Plan, Topology, Deployment Artifact, Implementation Artifact, and Concat. A string is taken as is as input parameter. When specifying "Plan," all elements preceding the current tasks are queried for their output parameters and offered as choice. In Fig. 1, "Size" of the StartEvent has been chosen. In case "Topology" is chosen, all properties of all node templates and relationship templates are shown for selection. In case of "Deployment Artifact," all deployment artifacts of all node templates are offered and one of them can be chosen. In case of "Implementation Artifact," all implementation artifacts of all node and relationship templates are offered and one of them can be chosen. One use case for the latter two is to specify files to be transferred to a node. For instance, a SQL script to be executed can be chosen here. In case of "Concat," a combination of the other options can be specified. This enables a simple transformation of output variables. For instance, a location path can be appended to the IP address returned by an operation. In the case of output parameters, only Plan and Topology are offered. When choosing "Plan," the parameter is offered as output for subsequent activities. When choosing "Topology," the parameter additionally is written to a property in the topology. Figure 1 shows available properties of an example topology. The current prototype of the BPMN4TOSCA 2.0 plan modeler is available via `http://dev.winery.opentosca.org`. A detailed description of the concepts, the architecture, and the implementation is provided by Michelbach [5].

3 Next Steps

The modeler provides a basis for modeling management plans. The prototype enables us to undertake a user evaluation to quantify the improvements of BPMN4TOSCA 2.0 in comparison to BPMN4TOSA 1.0. When using the modeler in our projects, we did not need complex transformations of the output parameters. An analysis of possible management plans is required to justify the absence of transformation capabilities.

Acknowledgments The research leading to these results has received funding from the German government through the BMWi projects CloudCycle



Edit Node Management Task

Name	Create VM	
Node Template	AmazonEC2	
Node Operation	createVM	
ImageID	ami-b83c0aa5	String
Size	StartEvent.Size	Plan
Region	eu-central-1	String
PublicIP		Topology

UbuntuVM.IP
UbuntuVM.SSHUser
UbuntuVM.SSHPassword
AmazonEC2.AccountUsername
AmazonEC2.AccountPassword

Figure 1. Editing the properties of a node management task

(01MD11023) and NEMAR (03ET401 8B), the BMBF project ECHO (01XZ13023G), and the DFG project SitOPT (610872).

References

1. Binz, T., et al.: OpenTOSCA – a runtime for TOSCA-based cloud applications. In: ICSOC. Springer (2013)
2. Breitenbücher, U., et al.: Vinothek – a self-service portal for TOSCA. In: ZEUS. CEUR (2014)
3. Kopp, O., et al.: BPMN4TOSCA: A domain-specific language to model management plans for composite applications. In: BPMN (2012)
4. Kopp, O., et al.: Winery – modeling tool for TOSCA-based cloud applications. In: ICSOC. Springer (2013)
5. Michelbach, T.: Ein Modellierungswerkzeug für BPMN4TOSCA. Diploma thesis, University of Stuttgart (2015)
6. OASIS: OASIS Topology and Orchestration Specification for Cloud Applications (TOSCA) Version 1.0 (2013)
7. Rücker, B.: Data in processes (2013), `https://app.camunda.com/confluence/display/BestPractices/Data+in+Processes`
8. Wettinger, J., et al.: Unified invocation of scripts and services for provisioning, deployment, and management of cloud applications based on TOSCA. In: CLOSER. SciTePress (2014)