Institute of Architecture of Application Systems

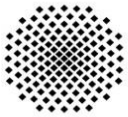
Winery – A Modeling Tool for TOSCA-based Cloud Applications

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

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

BibTeX:

```
@InProceedings{Winery,
  Title     = {{Winery} -- Modeling Tool for {TOSCA}-based Cloud Applications},
  Author    = {Oliver Kopp and Tobias Binz and Uwe Breitenb\"{u}cher and
               Frank Leymann},
  Booktitle = {11\textsuperscript{th} International Conference on
               Service-Oriented Computing},
  Year      = {2013},
  Publisher = {Springer},
  Series    = {LNCS}
}
```

University of Stuttgart
Germany

Winery – A Modeling Tool for TOSCA-based Cloud Applications

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

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

Abstract TOSCA is a new OASIS standard to describe composite applications and their management. The structure of an application is described by a topology, whereas management plans describe the application's management functionalities, e. g., provisioning or migration. Winery is a tool offering an HTML5-based environment for graph-based modeling of application topologies and defining reusable component and relationship types. Thereby, it uses TOSCA as internal storage, import, and export format. This demonstration shows how Winery supports modeling of TOSCA-based applications. We use the school management software Moodle as running example throughout the paper.

Keywords: Cloud Applications; Modeling; TOSCA; Management; Portability

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 software and hardware components involved and relationships between them. It is a graph consisting of nodes and relationships, where each of them has a type: a node type or a relationship type. These types offer management functionality, which is collected in node type and relationship type implementations. Concrete implementations, such as shell scrips or WAR files, are bundled through artifact templates, which can be referenced by multiple implementations making them reusable. Management plans capture knowledge to deploy and manage an application and are typically modeled as BPMN or BPEL workflows. The topology, management plans, and all required software artifacts such as installables, business logic, and management logic are condensed in an application package called TOSCA Cloud Service ARchive (CSAR for short). As TOSCA is standardized, CSARs are portable across different TOSCA-compliant runtime environments of different vendors.

To enable modeling of TOSCA-based applications in a tailored environment, we have developed Winery, which supports Web-based creation of CSARs using standard Chrome and Firefox browsers. Therefore, no additional software

installation is required to use the tool on client side. Winery's main features are type management and graphical topology modeling where the defined types are instantiated and interlinked. To facilitate collaboration, Winery not only supports sharing of TOSCA topologies, but also supports sharing of all related elements such as types or templates, which all are uniquely identified and accessible by URLs. This allows sharing information through passing simple references rather than exchanging entire documents.

Winery itself does not include a TOSCA-compliant runtime environment. One possible runtime environment is the OpenTOSCA system presented by Binz et al. [1].

2 Winery System Overview and Use Case

The TOSCA meta model defines 45 elements in total which can be used to model applications (cf. [4]). We subdivided this set into two classes: The first one contains seven elements that are directly related to visual topology modeling—namely relationship template, relationship constraint, node template, deployment artifact, requirement, capability, and policy. These elements are used in the Topology Modeler. The second class contains all remaining elements that are used to define semantics and configurations such as types, implementations, and policy templates. These elements can be created, modified, and deleted exclusively by using the Element Manager. This way, Winery separates concerns: The Topology Modeler eases modeling of application topologies by depicting elements and combinations thereof visually. On the one hand, this helps architects, application developers, and operators to understand and model applications without the need for technical insight into the type implementations and configurations. On the other hand, technical experts are able to provide and configure node types and relationship types by using the Element Manager. Thus, Winery enables collaborative development of TOSCA-based applications. As a consequence, Winery conceptually consists of three parts: (1) the Topology Modeler, (2) the Element Manager, and (3) the Repository, where all data is stored (see Fig. 1).

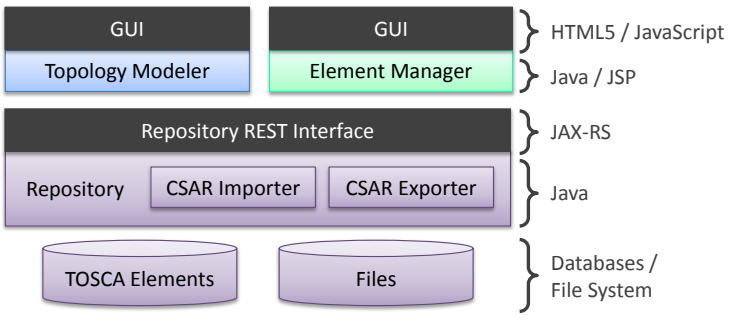


Figure 1. Components of Winery

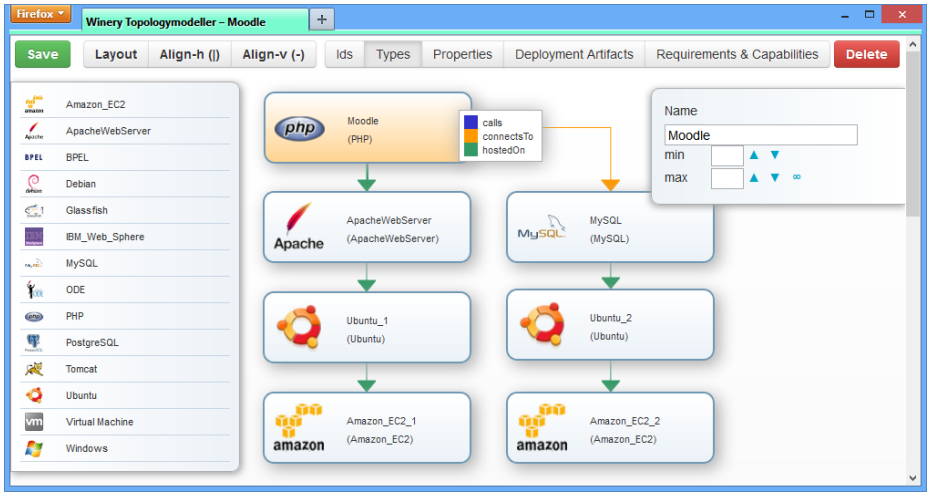


Figure 2. Moodle Application Topology. Adhering Vino4TOSCA [3], node templates are depicted as rounded rectangles and relationship templates as arrows between the rectangles. The possible relationship types starting from a PHP node template are depicted in the white box.

To create a TOSCA-based application, the first step is to create a new service template that contains an application topology by using the Topology Modeler. Therefore, Winery offers all available node types in a palette. From there, the user drags the desired node type and drops it into the editing area. There, the node type becomes a node template: a node in the topology graph. Node templates can be annotated with requirements and capabilities, property values, and policies. Most importantly, nodes may define deployment artifacts, which provide the actual implementation of the node template, e. g., a VM image, an operating system package for the Apache Web Server, or an archive containing a PHP application's files. Relations between node templates are called relationship templates. They can be created by clicking on a node template, which offers possible relationship types supporting this node template as valid source. Selecting one relationship type creates a new relationship template that has to be connected to the desired target. Figure 2 shows the TOSCA application topology of our use case—the Moodle[1] scenario. Amazon EC2 is used to host two virtual machines: One is used to host a MySQL database, the other one to host an Apache Web Server, which serves the Moodle PHP application. The PHP application connects to the MySQL database, which is depicted as orange arrow.

The Element Manager (Fig. 3) may, for instance, be used to define new types if required types are not provided by the community. For existing types, Winery's rendering information such as the border color and the icon can be configured. The Element Manager also handles the management of artifact templates and

[1] `http://www.moodle.org`

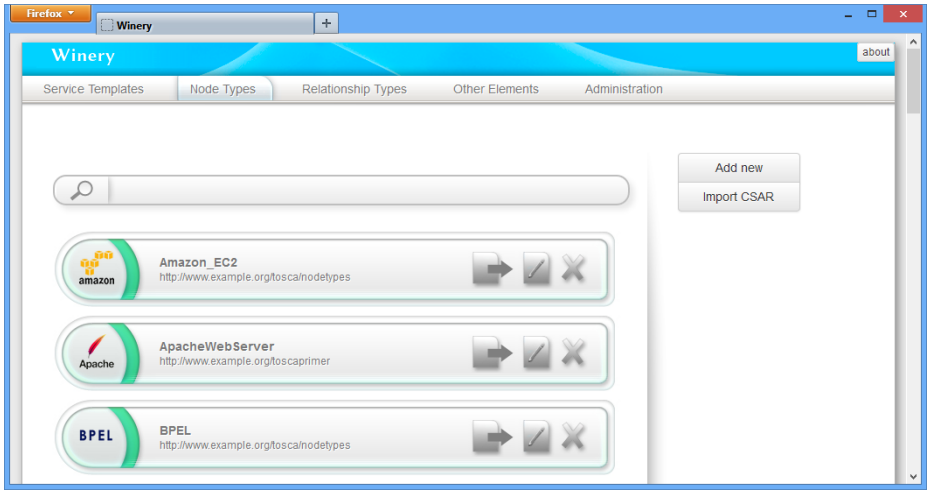


Figure 3. Element Manger Showing Available Node Types

related components: Files can be associated with an artifact template, which in turn are referenced from implementations as concrete implementation.

Having the topology ready, the next step is to model management plans. Winery does not support plan modeling by itself, but relies on other modeling tools to create plans. We usually use the Eclipse BPEL Designer[2] to model plans and compress the workflow and related files into one archive. In the service template, for each management plan, a plan element is created and the corresponding archive is uploaded. For deployment, we attach a BPEL workflow that provisions the Moodle application on Amazon EC2 virtual machines. The workflow installs the applications as defined in the topology and establishes the "connectsTo" relation by assigning the IP address of the MySQL instance to the Moodle configuration on the Apache Web Server.

After finishing modeling, the backend allows for exporting a CSAR file containing all required definitions. The resulting CSAR file can be deployed on a TOSCA-compliant runtime, which in turn deploys the implementation artifacts and the management plans to appropriate runtime environments. Finally, the user can start a build plan to instantiate an application instance. For more details, we recommend the detailed overview by Binz et al. [2], the TOSCA specification [6], and the TOSCA primer [7].

The Repository itself stores TOSCA models and enables managing their content. It offers importing existing CSARs into the Repository, which, for instance, makes community-defined node types and relationship types available for topology modeling. Winery is built to be integrated into other tool chains and projects which can reuse Winery's type repository, graphical modeling capabilities, or export functionality.

[2] `http://www.eclipse.org/bpel/`

3 Conclusion and Outlook

We presented the open source TOSCA modeling tool "Winery". It offers support for the complete TOSCA standard: Most importantly, types can be defined in the Element Manager and composed in the Topology Modeler. Although the Moodle application topology consists of less than 10 nodes, it could be used to show the basic concepts of Winery and TOSCA. Describing complex applications and their management in existing infrastructures is not in this paper's scope, but part of our ongoing work.

The current prototype is under submission to the Eclipse Software Foundation[3] to ensure open development. The next step is to create a BPMN4TOSCA [5] modeling component, which offers integrated topology and plan modeling: Each BPMN Service Task may directly link to a node template, where it works on.

Acknowledgments This work was partially funded by the BMWi project CloudCycle (01MD11023). We thank Kálmán Képes, Yves Schubert, Timur Sungur, and Jerome Tagliaferri for their work on the implementation of Winery.

References

1. Binz, T., Breitenbücher, U., Haupt, F., Kopp, O., Leymann, F., Nowak, A., Wagner, S.: OpenTOSCA – A Runtime for TOSCA-based Cloud Applications. In: 11th International Conference on Service-Oriented Computing. LNCS, Springer (2013)
2. Binz, T., Breitenbücher, U., Kopp, O., Leymann, F.: Advanced Web Services, chap. TOSCA: Portable Automated Deployment and Management of Cloud Applications, pp. 527–549. Springer (2014)
3. Breitenbücher, U., Binz, T., Kopp, O., Leymann, F., Schumm, D.: Vino4TOSCA: A Visual Notation for Application Topologies based on TOSCA. In: CoopIS (2012)
4. Kopp, O.: TOSCA v1.0 as UML class diagram (2013), available at `http://www.opentosca.org/#tosca`
5. Kopp, O., Binz, T., Breitenbücher, U., Leymann, F.: BPMN4TOSCA: A Domain-Specific Language to Model Management Plans for Composite Applications. In: Business Process Model and Notation. LNBIP, Springer (2012)
6. OASIS: OASIS Topology and Orchestration Specification for Cloud Applications (TOSCA) Version 1.0 Committee Specification 01 (2013)
7. OASIS: Topology and Orchestration Specification for Cloud Applications (TOSCA) Primer Version 1.0 (January 2013)

All links were last followed on August 26, 2013.

[3] `http://www.eclipse.org/proposals/soa.winery/`