Automatic Topology Completion of TOSCA-based Cloud Applications

Pascal Hirmer\textsuperscript{1}, Uwe Breitenbücher\textsuperscript{2}, Tobias Binz\textsuperscript{2}, Frank Leymann\textsuperscript{2}

\textsuperscript{1} Institute of Parallel and Distributed Systems, \textsuperscript{2} Institute of Architecture of Application Systems
Outline

- Motivation & TOSCA
- Method - Automated Topology Completion
- Prototype
- Conclusion & Future Work
Motivation – Automated Topology Completion (I)

- Mega-Trend: Provisioning of applications in the Cloud
  - Enable Cloud-advantages: availability, scalability, ...

- Challenges:
  - Manual application provisioning is complex and error-prone
  - Application developers are no Cloud experts (!)

- Improvement: automated provisioning of applications
  - But: requires means to define the application and infrastructure components to be provisioned
    - Enabled by cloud topologies (e.g., TOSCA)
Motivation – Automated Topology Completion (II)

- **Goal:** Creating complete cloud topologies automatically
  - Solely based on application-specific components

- **Solution:** Automated Topology Completion
  - Automated insertion of platform and infrastructure components
  - Ensuring guaranteed provisioning (!)
  - Allowing user control (only if desired)

→ Required technical expertise is minimized
→ Provisioning effort is minimized
TOSCA

- XML-based OASIS standard to enhance portability
  - Enabled by TOSCA Topology Templates
- Node Templates
  - Interoperable descriptions of application and infrastructure cloud services
- Relationship Templates
- Templates based on Types
  → Many details necessary to deploy a single application
TOSCA Requirements

- Requirement Type
  - Node Type
  - Requirement Definition
  - Requirement
    - Node Template
    - Node Template
  - Relationship Template

- Capability Type
  - Node Type
  - Capability Definition
  - Capability
    - Node Template
    - Node Template
  - requiredCapabilityType
Solution fits in the CloudCycle tool chain:
Agenda

- Motivation & TOSCA
- Method - Automated Topology Completion
- Prototype
- Conclusion & Future Work
Topology Completion - Method

1. Modeling of incomplete Topology Template
2. Selection of Target TOSCA Runtime
3. Automated Completion
5. Deploy Application
Modeling of Incomplete Topology Templates

Definition “Incomplete Topology”

- A TOSCA topology that can’t be provisioned in the respective run time environment due to missing components

- Application-specific components have to be modeled

- Modeling of infrastructure or platform components is optional
Incomplete Topology - Example

Web Archive (WAR) -> MySQL Database (MySQLDB)

Apache Tomcat Webserver (Apache_Tomcat) -> Ubuntu (Ubuntu) -> Amazon EC2 Instance (Amazon_EC2)

MySQL DBMS (MySQL_DBMS)

(SQLConnection)

(deployed_on)

missing components
CloudCycle

Topology Completion - Method

1. Modeling of incomplete Topology Template
2. Selection of Target TOSCA Runtime
3. Automated Completion
5. Deploy Application
Selection of Target TOSCA Runtime

Why this is important

- Features of Runtimes differ greatly
- Topologies have to be modeled for specific Runtime Environments

→ Completion has to consider Runtime features
Topology Completion - Method

1. Modeling of incomplete Topology Template
2. Selection of Target TOSCA Runtime
3. Automated Completion
5. Deploy Application
Black Box Approach

TOSCA Topology Completion

Incomplete Topology Template

Completed Topology Template
Glass Box Approach

Incomplete Topology Template  →  TOSCA Topology Completion  →  Completed Topology Template

Cloud Service Developer

observe  →  stepwise  ←  steer

CloudCycle
Topology Completion - Algorithm

Web Archive (WAR)
- Req. WebserverContainerRequirement
  - RequiredCapabilityType WebserverContainerCapabilityType

Apache Tomcat (Apache_Tomcat)
- Cap. WebserverContainerCapability

Runtime Environment
- ProvisioningAPI
  - No Provisionable?

Type Repository
- Node Types
  - Windows 7
  - Apache Tomcat Server
  - Amazon EC2 Instance
  - Glassfish Server
- Relationship Types
  - SQL_Connection
  - host_on
  - deploy_on

Cloud Service Developer
- Glass Box Approach
- ProvisioningAPI
  - Provisionable?
Completion Result

Web Archive (WAR)
  (hostedOn)
  (ApacheWebServer)
    (hostedOn)
    (UbuntuLinux)
      (hostedOn)
      (AmazonEC2VM)
CloudCycle

Topology Completion - Method

1. Modeling of incomplete Topology Template
2. Selection of Target TOSCA Runtime
3. Automated Completion
5. Deploy Application
Agenda

- Motivation & TOSCA
- Method - Automated Topology Completion
- Prototype
- Conclusion & Future Work
Prototype – Winery Topology Modeler Extension

[Diagram showing a user interface with options to drag and drop items into a model.]

Drag & Drop

*click*

CloudCycle
Prototype – Topology Completion Dialog

aka Glass Box Approach

*click*
Prototype – Glass Box Approach

![Diagram of a web application topology](image)

**Preview**

**Relationship Template Selection**

**Node Template Selection**

**Manual Changes**

*click*

There are several possible Relationship Templates to connect the Node Templates WAR and Apache_Tomcat_7. Please choose at least one connection:

- contained_in
- deploy_on

Use Template: Apache_Tomcat_7

Press **Cancel** to continue the completion manually.
Prototype – Completion Result

Diagram:

- WAR (WAR)
- Apache_Tomcat_7 (Apache_Tomcat_7)
- Windows_7 (Windows_7)
- Amazon_EC_2_Insta... (Amazon_EC_2_Insta...}

Diagram elements are connected with arrows indicating containment and deployment relationships.
Agenda

- Motivation & TOSCA
- Method - Automated Topology Completion
- Prototype
- Conclusion & Future Work
Conlusion

- We enabled Automated Topology Completion using TOSCA Requirements
- Solution fits in the CloudCycle Toolchain
Future Work

- Automated Completion of TOSCA Properties

- Policy-aware Topology Completion
  - E.g. costs, availability, locations
Thank you!

Pascal Hirmer\(^1\), Uwe Breitenbücher\(^2\), Tobias Binz\(^2\), Frank Leymann\(^2\)

\(^1\) Institute of Parallel and Distributed Systems, \(^2\) Institute of Architecture of Application Systems

www.opentosca.org