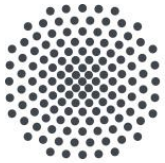
Deployment of Hybrid Quantum Applications



University of Stuttgart

Martin Beisel, Benjamin Weder

{beisel,weder}@iaas.uni-stuttgart.de

Institute of Architecture of Application Systems

PlanQK

SeQuenC

EniQmA

Tutorial Structure

- Session 1 (09:00 - 10:30): An Introduction to Quantum Computing
- Session 2 (11:00 - 12:30): Quantum Software Engineering
- Session 3 (14:00 - 15:30): Quantum Workflows
- **Session 4 (16:00 - 17:30): Operation of Hybrid Quantum Applications**
  - Hands-On Session Part 2
  - Deployment of Hybrid Quantum Applications
  - Wrap-Up

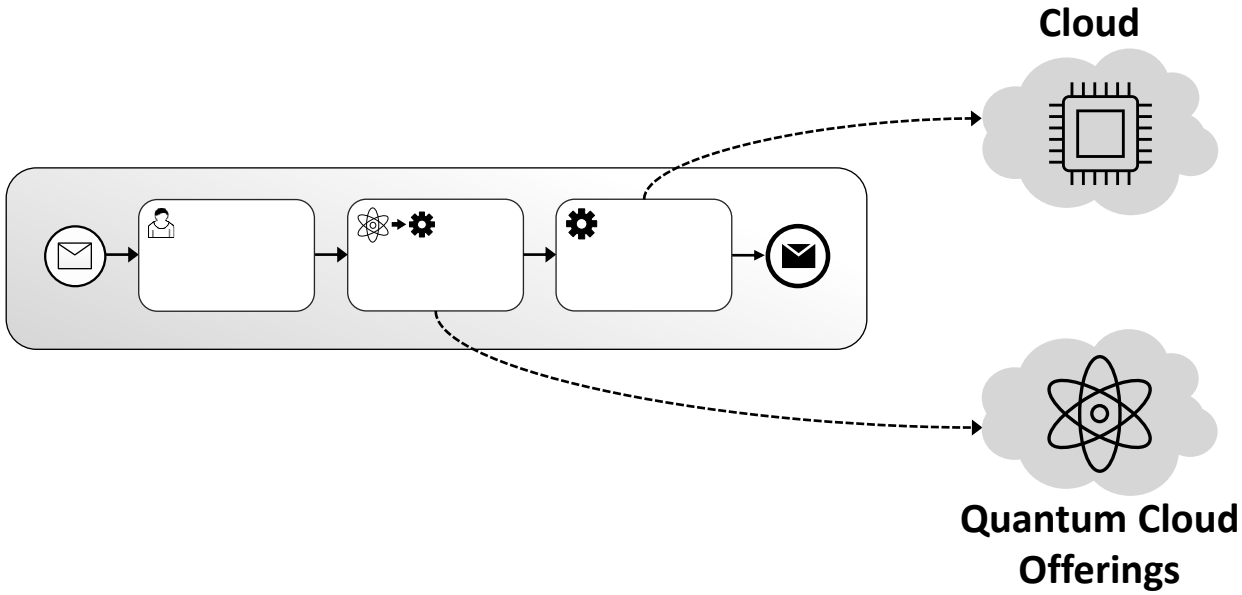
Tutorial Structure

- Session 1 (09:00 - 10:30): An Introduction to Quantum Computing
- Session 2 (11:00 - 12:30): Quantum Software Engineering
- Session 3 (14:00 - 15:30): Quantum Workflows
- **Session 4 (16:00 - 17:30): Operation of Hybrid Quantum Applications**
  - Hands-On Session Part 2
  - **Deployment of Hybrid Quantum Applications**
  - Wrap-Up

Motivation

Recap: Invocation of Quantum and Classical Programs

- Workflow invokes quantum and classical programs during runtime:



- Required programs and service are often not “always on”:
  - Must be deployed before using them in a workflow
  - Error-prone, time-consuming deployment

➔ Automation using deployment technologies

The Challenges

- How to deploy such applications?
- How to manage such applications?
- How to communicate the structure of such applications?
- How to achieve reliable operation?
- How to avoid vendor lock-in?
- How to achieve portability and interoperability?
- …

Currently used Technologies and APIs

- Currently in use are, for example, all these technologies:


Problems

- Each technology employs its own…
  - … API(s)
  - … domain-specific language(s) (DSLs)
  - … invocation mechanisms
  - … data model
  - … wording
  - … fault handling
  - … security mechanisms

How to describe the deployment of cloud applications in a portable manner?

Topology and Orchestration Specification for Cloud Applications (TOSCA)

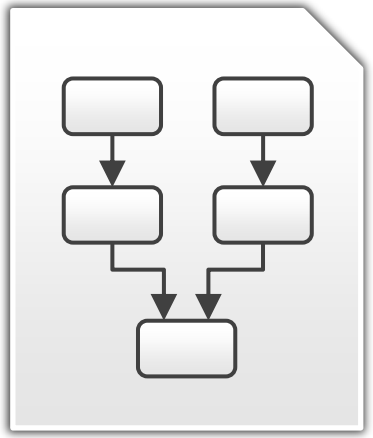
TOSCA Overview

Topology and **Orchestration** Specification for Cloud Applications

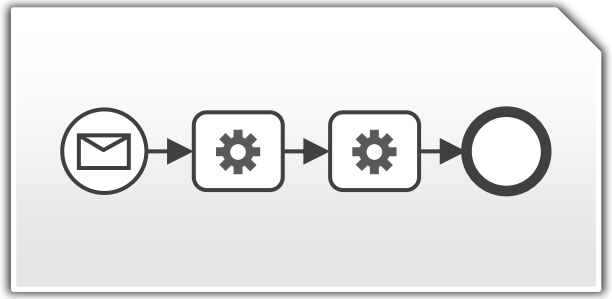
TOSCA Overview

and
Specification for Cloud
Applications

Topology
Application Structure

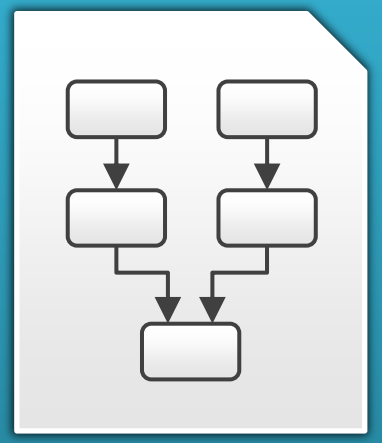


Orchestration
Deployment & Management

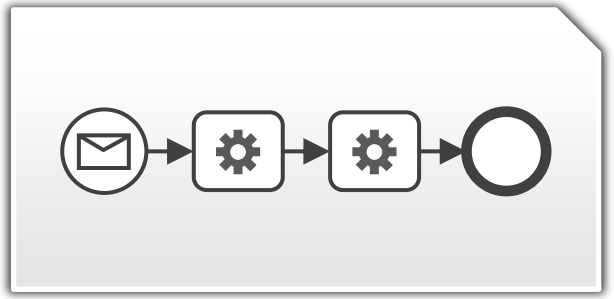


TOSCA Overview

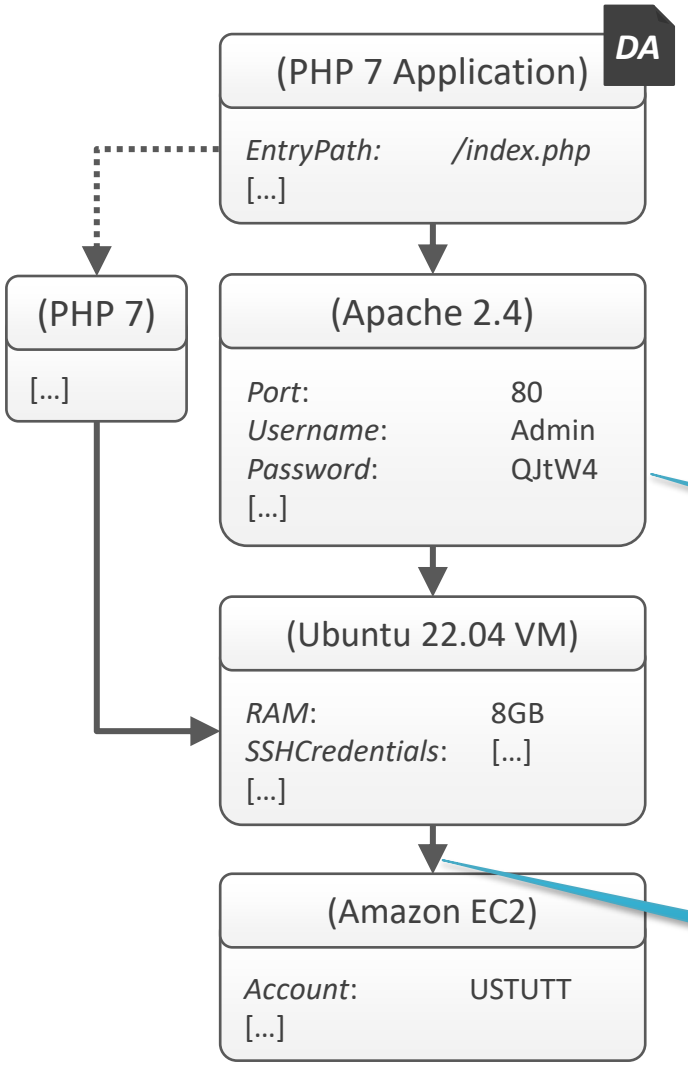
Topology
Application Structure



Orchestration
Deployment & Management



Node Templates and Relationship Templates



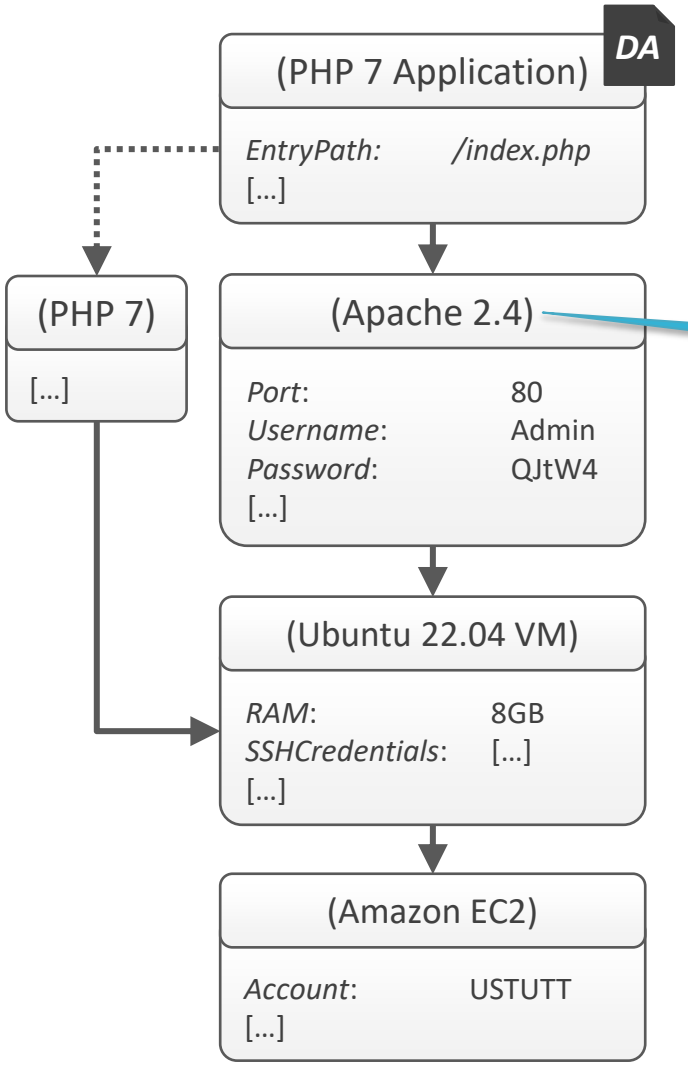
- TOSCA enables describing the structure of the application to be deployed in the form of a directed, acyclic graph
  - Nodes of the graph represent ***components***
    - e.g., an Apache Webserver, a VM, a PHP Application, or a MySQL database
    - These nodes are called ***Node Templates***

Node Template

  - Edges of the graph represent ***relationships***
    - e.g., that one componenti is *hosted on* another component or *connects* to another component
    - These edges are called ***Relationship Templates***

Relationship Template

Node Types and Relationship Types

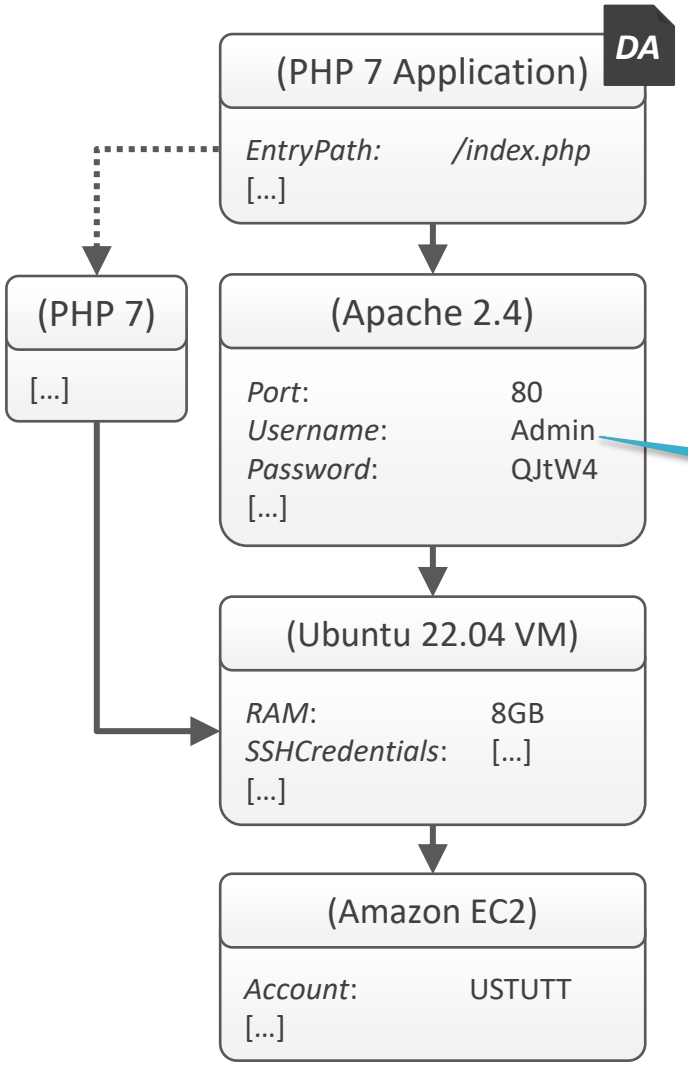


- Both Node Templates and Relationship Templates are typed to define the semantics of templates
  - ***Node Types*** define the semantics of Node Templates
    - e.g., a Node Template may be of Node Type “Apache2.4“
  - ***Relationship Types*** define the semantics of Relationship Templates
    - e.g., a Relationship Template may be of Relationship Type „hostedOn“ or „SQLConnection“

      → = *hostedOn*

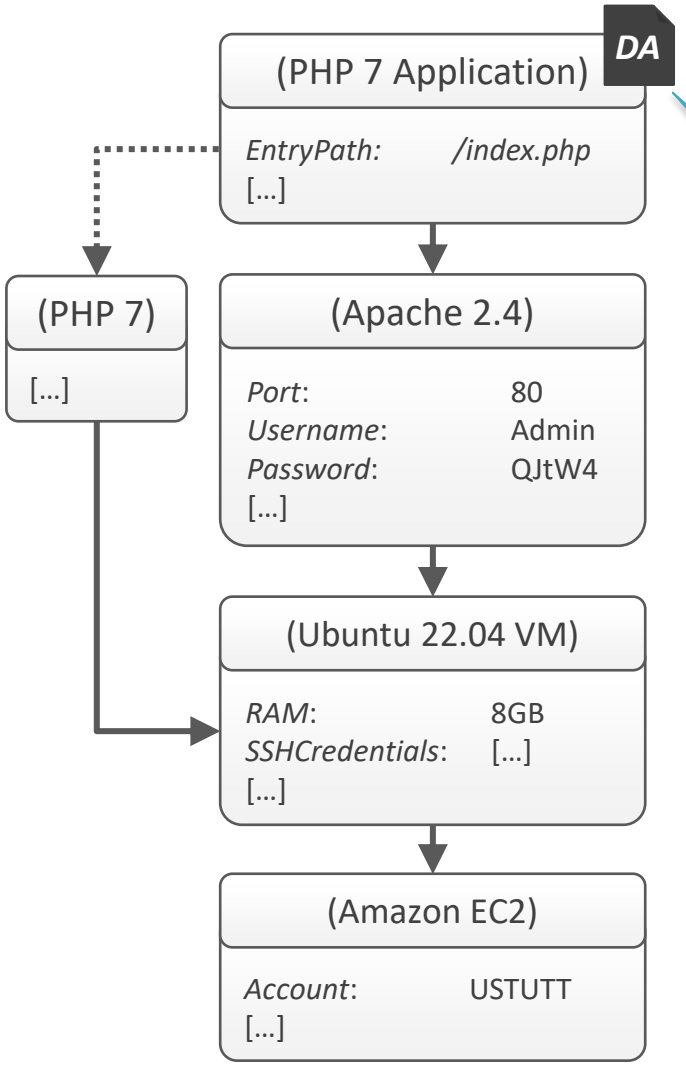
      ⇢ = *dependsOn*
  - The type system is extensible: New Node and Relationship Types can be defined arbitrarily
    - Also inheritance is supported

Properties



- To configure the deployment, Node and Relationship Templates may specify ***properties***
  - For example, to specify that the Apache Webserver shall serve HTTP requests at port 80
  - Or to specify the desired RAM of a virtual machine to be provisioned
  - Properties may also contain ***instance information at runtime*** about a node or relationship
    - For example, the IP-address of a provisioned virtual machine, which is not known at modelling time
  - The properties a Node or Relationship Template provides and their schemas are defined by the respective Node or Relationship Type

Deployment Artifacts

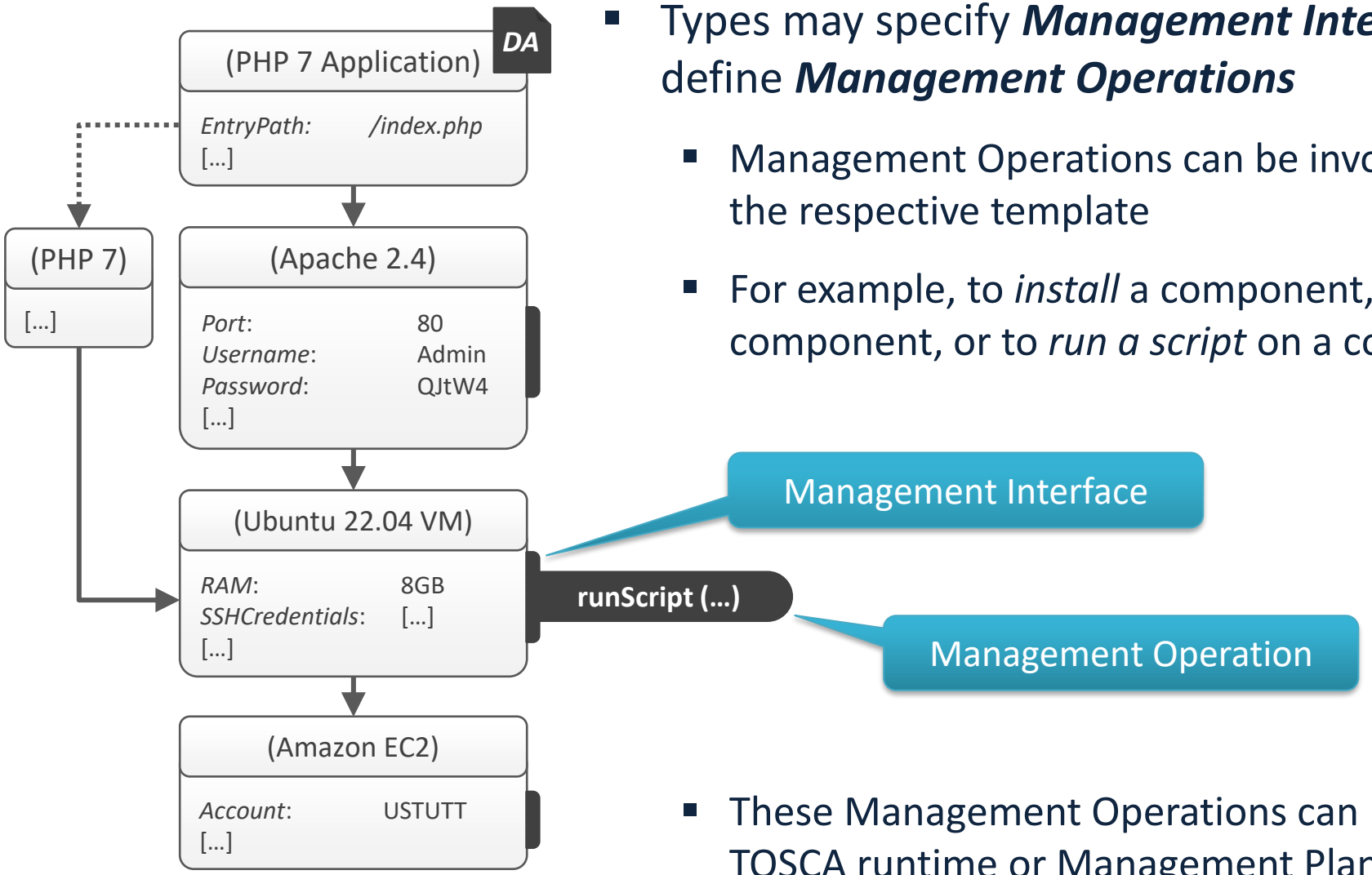


- To specify the implementations of components ***Deployment Artifacts (DA)*** are used
  - For example, a Deployment Artifact can be the PHP files of a Web application

Deployment Artifact

  - A Deployment Artifact typically specifies one or more files and some properties about the artifact
    - For example, the type of the files

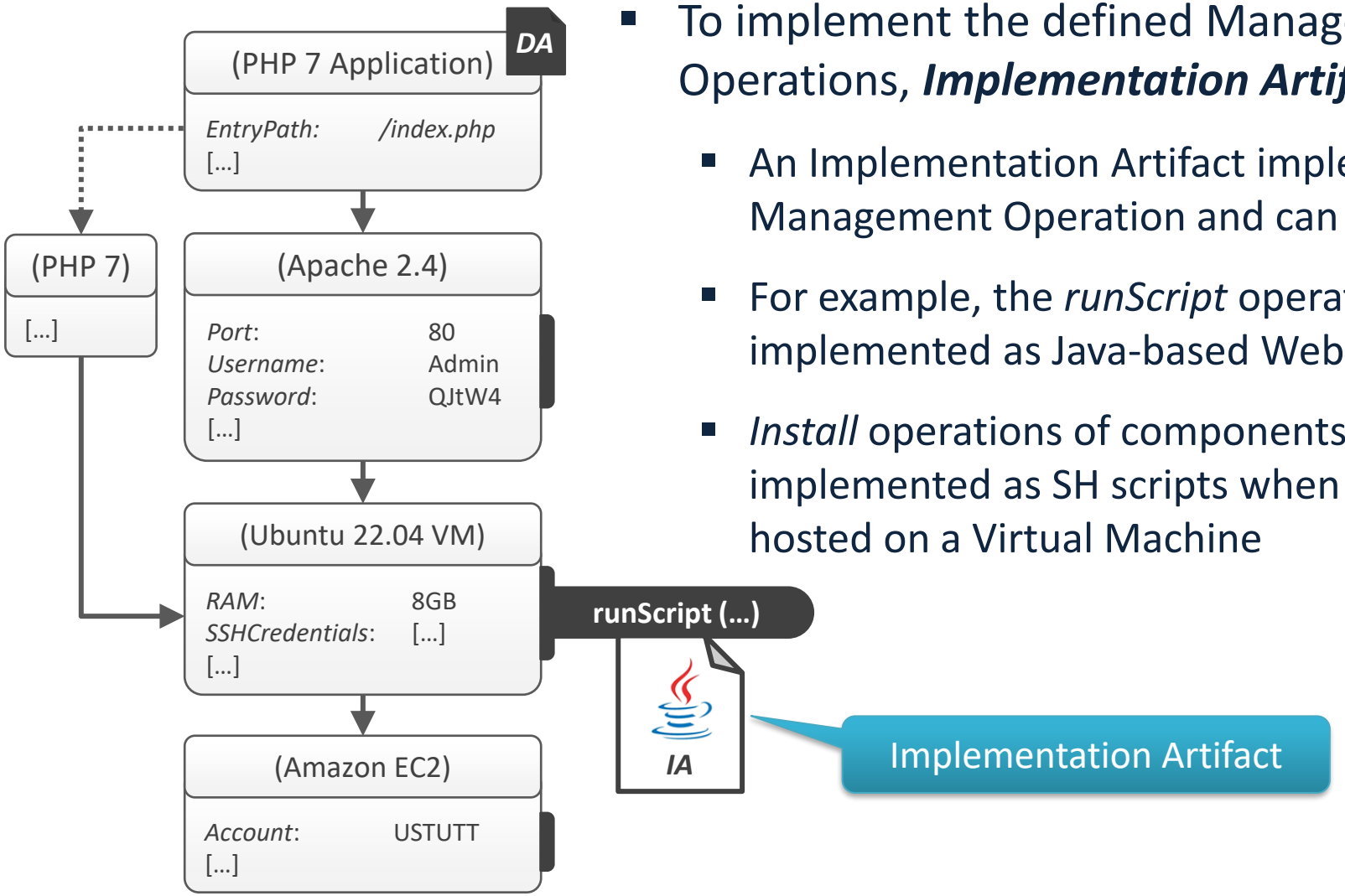
Management Interfaces and Management Operations



- Types may specify ***Management Interfaces*** that define ***Management Operations***
  - Management Operations can be invoked to manage the respective template
  - For example, to *install* a component, to *start* a component, or to *run a script* on a component

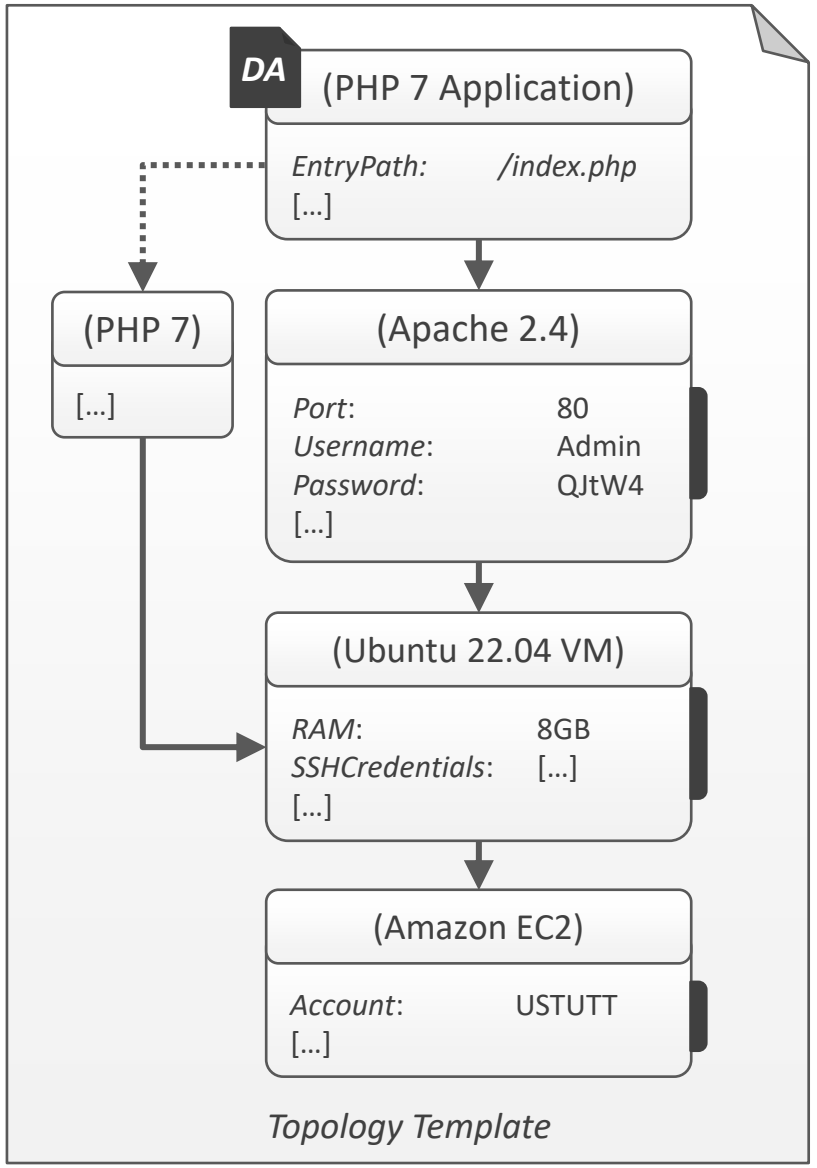
- These Management Operations can be called by the TOSCA runtime or Management Plans (see next)

Implementation Artifacts



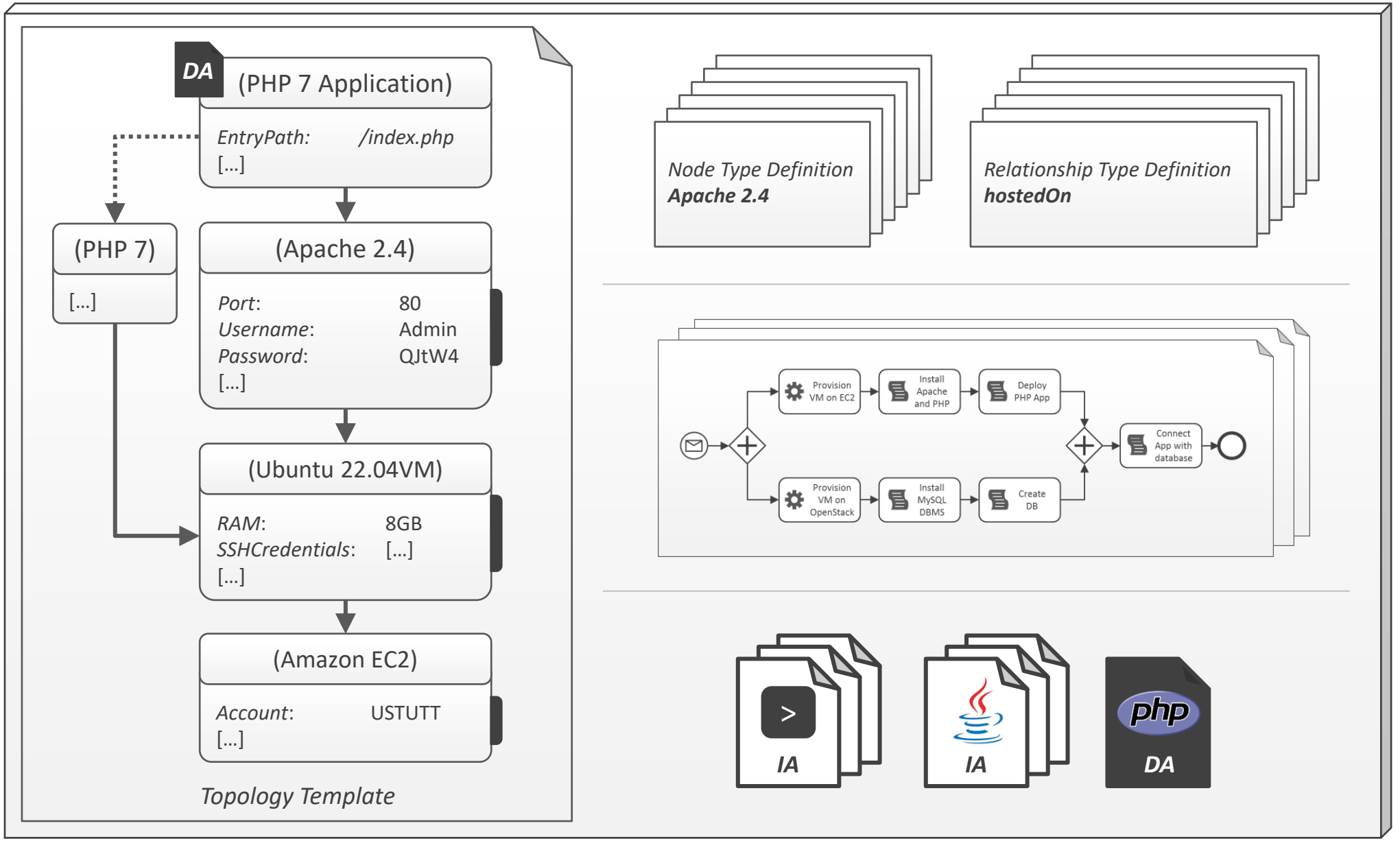
- To implement the defined Management Operations, ***Implementation Artifacts*** are used
  - An Implementation Artifact implements a certain Management Operation and can be executed
  - For example, the *runScript* operation could be implemented as Java-based Web Service
  - *Install* operations of components are often implemented as SH scripts when they shall be hosted on a Virtual Machine

Topology Template, Service Template, Cloud Service Archive

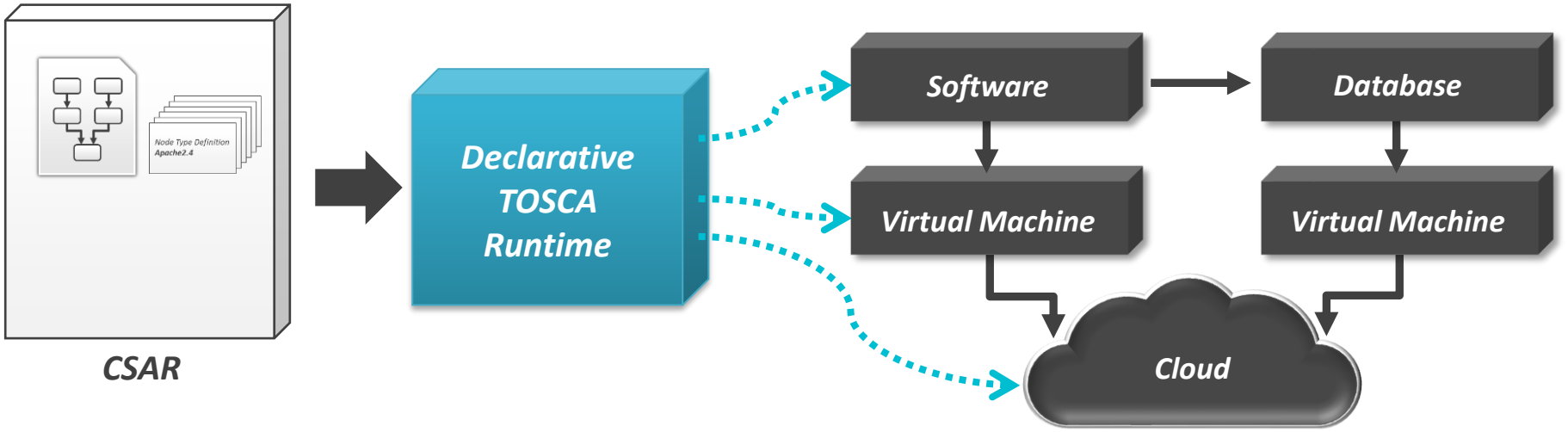


- A ***Topology Template*** represents the deployment model
  - Contains all Node and Relationship Templates of the application
- A ***Service Template*** contains one Topology Templates as well as all used type definitions and artifacts
  - A Service Template can be used also to package only type defintions or artifacts
- A ***Cloud Service Archive (CSAR)*** is a archive format standardized by TOSCA to package Service Templates as well as all required files, plans, etc. into a ZIP file

Cloud Service Archive (CSAR)



Declarative TOSCA Runtime



- A declarative TOSCA Runtime is able to provision the modelled application only based on the Topology Template and provided IAs
  - In this case, no Management Plans must be provided in the CSAR (see next)
  - Thus, TOSCA supports the Declarative Deployment Modelling Pattern
    - OpenTOSCA is a declarative runtime that supports the declarative approach

Conclusion & Outlook

Conclusion & Outlook

- Quantum workflows invoke quantum and classical programs during runtime
- Deployment of required programs complex and error-prone → Automation required
- TOSCA standard enables portability and interoperability
- OpenTOSCA ecosystem provides an end-to-end toolchain for the deployment of hybrid quantum applications

Thank you for your attention ☺