OASIS TOSCA

... and how it could fit into OpenStack Heat

OpenStack Design Summit, April 15th 2013

Thomas Spatzier, IBM (thomas.spatzier@de.ibm.com)
Agenda

• TOSCA Overview and Examples
• TOSCA Concepts Explained
• About Encoding ...
• Discussion: TOSCA and Heat
What is TOSCA?

Topology and Orchestration Specification for Cloud Applications

A language for defining Service Templates ...

... including a Topology Template describing the structure of a service

... including the definition of plans for orchestrating the application

Packaging format (CSAR) for packaging models and all related artifacts.

Definition of building blocks for services

... along with the implementation artifacts for manageability operations

... and the definition of deployment artifacts for components

Definition of possible links between components
TOSCA TC Members

3M
ActiveState
Axway
CA Technologies
CenturyLink
Cisco
Citrix
Cloudsoft
EMC
Fujitsu
Google
HP
IBM
Huawei
Jericho Systems
NetApp
Nokia Siemens
Pricewaterhouse
Primeton
Red Hat SAP
Software AG
VCE
Vnomic
WSO2
Zenoss
1st Interop Example: SugarCRM

- SugarCRMApp [WebApplication] depends on PHPModule [ApacheModule]
  - Apache [ApacheWebServer] hosted on LinuxOS4Web [OperatingSystem]
    - Server4WebTier [Server]
- SugarCRMDB [MySQLDatabase] connects to MySQL [MySQL]
  - MySQL [MySQL] hosted on LinuxOS4DB [OperatingSystem]
    - Server4DBTier [Server]
Using TOSCA to model Applications

Language
TOSCA v1.0 spec

Types
MySQL Database
MySQL

Templates
SugarCRMApp [WebApplication]
Apache [ApacheWebServer]
MySQL [MySQL]
SugarCRMDB [MySQLDatabase]

Properties:
Context root: /mycrm
HTTP Port: 8080

Properties:
DB Name: “mysqldb”
Admin User: myadmin

Primary focus for users

“Orchestrator”
 Deployed Instances
Role of Relationships in Model Processing

- Use defined lifecycle operations to deploy and manage each component
- If a component is related to another component, see if relationship injects additional processing logic (e.g. pre-configure endpoint)
- Use base relationship types to derive processing order
  - Process a host before a hosted component (HostedOn)
  - Process a provider before a client (DependsOn, ConnectsTo)

... a pretty mechanical process
How is a Topology processed?

• Use base relationship types to derive component processing order
  – First process a host, then process hosted component
  – First process a component that another component depends on, then process the dependent component
  – First process a component that another component connects to, then process the connecting component

• For each component
  – Deploy its Deployment Artifacts
  – Invoke lifecycle operations in right sequence (create, configure, start ...); their can be no-ops

• If a relationship contributes logic, inject it into component operation invocations
Requirements & Capabilities

Some Application

Compute

Storage

Database

Database Provider

Requirements can be fulfilled explicitly by other components in the model

Requirements can be fulfilled by the runtime

Database provider = MySQL
version >= 5.5

Provider A
OR
Provider B
OR
Provider C

Some Component

R

C

C

C

arch = x86_64
cpus = 2
mem >= 4GB
OS = Linux

size >= 10GB

Requirements/Capabilities are base for substitutability
Subsystems can be abstracted in some models.

Other models can define details of subsystems.
→ separation of concern, re-use
Usage Scenarios for Model Composition & Substitution

Varying deployment options

- WebApp
  - WebTier
    - Web Server
    - Load Balancer
      - VM
    - Web Server
      - VM
      - VM
      - 1..*

Single node web tier
Scalable web tier

Layering of models

- App
  - Tomcat
    - MySQL
  - DB
  - VM
    - Network
    - Storage

Custom workload
Multiple options of middleware deployments
Multiple options of infrastructure configurations

Application Layer
Platform Layer
Infrastructure Layer
About Encoding ...

The TOSCA TC decided to use XML and XML Schema as the normative way for defining the TOSCA language.

But: The important thing about TOSCA are the concepts, not the encoding.

Alternative encodings (e.g. JSON) can be defined as part of TOSCA vNext work.
Alternative Encoding Example

```
<xs:schema>
  <xs:element name="ServerProperties" type="tServerProperties"/>
  <xs:complexType name="tServerProperties">
    <xs:sequence>
      <xs:element name="NumCpus" default="1">
        <xs:simpleType>
          <xs:restriction base="xs:int">
            <xs:enumeration value="1"/>
            <xs:enumeration value="2"/>
            <xs:enumeration value="4"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:element>
      <xs:element name="Memory" type="xs:int"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

```
"NodeTypes" : {
  "name" : "Server",
  "properties" : [
    {
      "name" : "NumCpus",
      "type" : "integer",
      "options" : [1, 2, 4]
    },
    {
      "name" : "Memory",
      "type" : "integer",
    }
  ]
},
...
```

```
"TopologyTemplate" : {
  "NodeTemplates" : [
    {
      "id" : "MyServer",
      "type" : "Server",
      "properties" : [
        "NumCpus" : 2,
        "Memory" : 4096
      ]
    },
    ...
  ],
  ...
}
```
Now, how could it fit into Heat?

- It’s all about concepts: as longs as a pattern engine’s concepts and TOSCA are aligned, mapping to TOSCA as an external format is straight forward
  - No need to use TOSCA as Heat’s core format

- Define a sub-set profile of TOSCA that is appropriate for use cases targeted by Heat

- Define an alternative JSON rendering for TOSCA sub-set profile

- Define a set of base Node- and Relationship Types for core OpenStack resources: Compute (Nova), Network (Quantum), Block Storage (Cinder), Object Storage (Swift)
  - As natively supported types
  - Users do not have to care about defining TOSCA types, but can just start defining templates

- Use implementation to improve and refine the TOSCA standard; use standardized concepts to shape implementation
Learn more about TOSCA

- **TOSCA Specification, Version 1.0**  
  *Committee Specification 01, 18 March November 2013,*  
  http://docs.oasis-open.org/tosca/TOSCA/v1.0/cs01/TOSCA-v1.0-cs01.pdf

- **TOSCA Primer, Version 1.0**  
  *Committee Note Draft (CND) 01, Public Review Draft 01, 31 January 2013,*  
  http://docs.oasis-open.org/tosca/tosca-primer/v1.0/cnd01/tosca-primer-v1.0-cnd01.pdf

- **TOSCA Implementer's Recommendations for Interoperable TOSCA Implementations, Version 1.0**  
  *Interoperability Subcommittee, Working Draft 01, Rev. 02, 14 January 2013,*  

- **TOSCA Interoperability Subcommittee, SugarCRM Scenario Sample CSAR**  
  *Preliminary Draft CSAR for Interop. Testing against TOSCA v1.0 Specification,*  