Porting an N-tier Application on Cloud using P-TOSCA: A Case Study

Magdalena Kostoska, Aleksandar Donevski, Marjan Gusev, and Sasko Ristov
Ss. Cyril and Methodius University, Faculty of Computer Science and Engineering,
Skopje, Macedonia
magdalena.kostoska@finki.ukim.mk, donevski.aleksandar@students.finki.ukim.mk,
{marjan.gushev, sashko.ristov}@finki.ukim.mk

Abstract—Although many companies and universities migrated their services, applications and data on the cloud, still significant part did not move yet, due to variety of reasons. Migration to other platform can cause the client many problems, and several directions have emerged to automate it as much as possible.

P-TOSCA is a recently proposed Platform as a Service (PaaS) extension for TOSCA for automated application portability. In this paper, we demonstrate the migration of an application with most common N-tier architecture from on-premise to cloud, as well as from one cloud to another, by using P-TOSCA portability model, the development of application topology and execution plan. Although the presented demo was tested to migrate and transfer an N-tier application to Eucalyptus and OpenStack open source clouds, this model and procedure can be used for automated migration and transfer among any open standard that supports PaaS and P-TOSCA application specification.

Index Terms—Automated management; Cloud; Portability.

I. INTRODUCTION

Several years after Amazon promoted the first cloud, IT managers had a dilemma whether to migrate their IT systems to a cloud. It was a new emerging concept with many challenges, such as security, trustworthiness, how to choose an appropriate cloud service provider, vendor lock-in or migration cost [1] and problems. Some challenges are mitigated. For example, almost all most common cloud service providers implemented information security management system [2], defined strict SLAs with several "nines" for their availability, free trial accounts, thus improving the trustworthiness with their customers.

Raising new cloud service providers with new and improved cloud services, transformed the dilemma whether to migrate into when to migrate to a cloud. Unfortunately, some challenges remained, such as automated portability. This means to migrate an application to a cloud neither with changing the application, nor server configuration, nor the new operating and cloud environment [3]. The automated portability will mitigate the risk of vendor lock-in [4] for the applications that are built on open standards. This will allow a customer to easily migrate to the cloud, transfer from one cloud to another, or even migrate back to its premise, without concerning the cloud platforms, infrastructures, APIs etc.

Recently, Kostoska et al. [5] identified several weaknesses and ambiguities of TOSCA, and promoted the P-TOSCA model as an extension of the TOSCA 1.0 standard [6], which can be used for portable cloud services [7]. P-TOSCA is used to describe a PaaS application specification and for automatic porting it to cloud environment. In this paper, we demonstrate how an N-tier application analyze the PaaS applications hosting portability regardless of their deployment model, whether they are public or private, as long as they integrate the open standards or use standardized (Application Programming Interfaces) APIs.

The rest of the paper is organized as follows. Related work in the area of cloud application portability is elaborated in Section II. Section III describes the P-TOSCA and definition and procedures of porting the N-tier application. Evaluation of results and discussion are given in Section IV, while the contribution of our work and plan for our future work in Section V.

II. RELATED WORK

In the area of cloud portability standards there are few active working areas. Two standard oriented researches represent Open Virtualization Format (OVF) and Topology and Orchestration Services for Applications (TOSCA). The Distributed Management Task Force (DMTF) suggests the OVF [8], which specifies transport mechanism for moving virtual machines from one hosted platform to another and between different hypervisors. TOSCA defines a portable and manageable specification of services and applications that can be deployed on any cloud provider.

The TOSCA standard is research interest in many directions. OpenTOSCA, approach defined by Binz et al. [9], represents execution environment that offers imperative processing of the TOSCA. Other directions include modeling tools for TOSCA like Vino4TOSCA and Winery [10], [11], as well as TOSCA topology specification [12], [13].

Other efforts in the area of standardization are delivered by IEEE Cloud Computing Initiative (CCI), where two new standards are in progress: Cloud Portability and Interoperability Profiles (CPIP) [14] and Standard for Intercloud Interoperability & Federation (SIIF) [15].

Other researches propose introduction of abstraction-driven approach. In this area Ranabahu et al. [16] use an abstraction-driven approach to specify solutions. This approach is used in two research projects using Domain Specific Language (DSL). Hill and Humphrey [17] develop a storage abstraction layer
Usage of semantic technology is also popular approach. Loutas et al. [18] focus on this technology to enable deployment, migration and interoperability of PaaS hosted applications. Cretella and Di Martino [19] propose an approach that aims at automatic analysis of cloud vendor APIs. Petcu et al. [20] exploits this technology to create a new set of APIs and to enable functional portability through the vendor-agnostic API and data portability through the use of drivers for the same type of data.

In our previous research we demonstrated porting of a simple SOA application by using the P-TOSCA model and P-TOSCA engine [21]. In this paper, we present how to describe an N-tier application and migrate from on-premise to two different cloud environments, as well as to transfer the application from one cloud to another.

III. CLOUD APPLICATION PORTABILITY BASED ON P-TOSCA

This section describes the N-tier application and the P-TOSCA specification for its porting.

A. About P-TOSCA

The P-TOSCA model [5] enables:

- better description of application topology and
- definition of specific cloud properties,

in addition to the original TOSCA specification.

The application topology, which contains the types, templates and artifacts, are described by XML file. Additional files (artifacts) represent files needed for the information about the application deployment and configurations (scripts, libraries, wars or zips) are stored in other files called artifacts. The specification and the required artifacts are packed in a Cloud Service Archive (CSAR).

1) Application migration: The detailed procedure of migration is presented in Fig. 1. The first step is the client authentication, and then upload the CSAR archive in the step 2. The P-TOSCA PaaS platform processes the model and contacts the cloud controller to create instances (it is an N-tier application and need at least two instances). After creation of the instances, the user selects the plan for execution, followed by artifact deployment.

2) Application transfer from one cloud to another: This procedure differs from the migration at the first steps. After the authentication to the target P-TOSCA cloud, the user does not upload the CSAR, but he/she selects the hosted application and specifies the source and destination clouds. The CSAR is requested by the source and sent by the destination automatically between P-TOSCA. After unpacking the CSAR, the destination P-TOSCA creates instances, deploys the artifacts and acknowledges the user for finished transfer of the application.

B. N-Tier application

The application is a Java MVC student records management application. The selected application is a 2-Tier application (database and application server), but the model can be generalized for 3- or 4-Tier application. The first tier is a database tier and it is installed with MySQL DBMS to host a database. The second tier consists of the presentation and logic tiers. It is installed with Tomcat web application server, to host the application.

C. P-TOSCA topology description for the N-Tier application

We have generated the topology description for the N-Tier application.

The generated Node Types (base, specific and custom) for this application are shown in Fig. 2. It is specified by three main categories:

- Base type - requires basic components (operating system, web server, web application, database and DBMS);
Fig. 2. *Node Types* and their hierarchy for Student Records (*N*-tier) application

<table>
<thead>
<tr>
<th>Server property</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of CPUs</td>
<td>2</td>
</tr>
<tr>
<td>Memory size</td>
<td>2 GB</td>
</tr>
<tr>
<td>Disk size</td>
<td>10 GB</td>
</tr>
<tr>
<td>Initial instances</td>
<td>1</td>
</tr>
<tr>
<td>Security rule protocol</td>
<td>TCP</td>
</tr>
<tr>
<td>Security rule ports</td>
<td>80, 443</td>
</tr>
</tbody>
</table>

**Table I**

<table>
<thead>
<tr>
<th>Server property</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP port</td>
<td>80</td>
</tr>
<tr>
<td>HTTPS port</td>
<td>443</td>
</tr>
</tbody>
</table>

**Table II**

- **Specific type(s)** - requires application topology (Linux operating system), Tomcat web server, Java Web Application, MySQL database and MySQL DBMS; and
- **Custom node types** - defines specifically developed application components.

The *Topology Template* defines the topology elements (nodes) that the application is consisted of. Each node is specified by node template and is defined (i.e. type of) given node type. The *Topology Template* presented in Fig. 3, which shows the topology of the application.

Each node of the topology is specified by a node template, and the relationships between the nodes are specified using relationship templates. Same as nodes, the relationship types are initially set in the specification and each relationship template specifies the type of relationship.

The information about the infrastructure (required instance for application and security protocols and port settings) is defined in Table I. Listing 1 presents the XML Tomcat Tier node template definition.

Table II defines http and https ports where the Tomcat server listens for request, while Listing 2 defines the TomcatWeb-Server Node template definition.

Values for artifact template for configuration of Student Records (*N*-tier) application are given in Listing 3. The configuration script requires input parameters for location (IP address) of the database server, as well as username and password required for the connection string to the database.

The CSAR is a zip archive that packs:
- all node and template definitions,
- Tomcat installation and deployment artifacts,
- MySQL installation and deployment artifacts,
- application deployment artifact, and
- database deployment artifacts.

**IV. Evaluation of the Results and Discussion**

This section presents the testing environment and testing data and evaluates and discusses the results of the migration from on-premise to both cloud environments and transfer from one to the other cloud environment.

**A. Testing environment**

The testing environment consists of two open source cloud frameworks:
• OpenStack cloud, which is installed with Ubuntu 12.04.3 LTS on one server with both cloud and node controller), and
• Eucalyptus cloud, which is installed on three servers, each with CentOS 6.5: one for the node tier - node controller, and two for cloud tiers - cloud and cluster controllers).

Fig. 4 depicts the cloud environments. More servers are used for Eucalyptus since it is more scalable than other open source clouds [22]. All physical servers of both clouds are the same, each containing 4 cores CPU and 4GB RAM. Internal networks are isolated from external. Public (float) networks of both clouds are isolated in separate VLANS [23], and since they are private IP addresses, they are NAT-ed to Internet through router.

Each experiment is conducted five times and average values are presented.

B. Evaluation and discussion

The evaluation of the migration and transfer using P-TOSCA model is presented in Table III. First of all, all four experiments were successful, that is, migration from on-premise to OpenStack and Eucalyptus.
clouds, as well as transfer of the N-Tier application from OpenStack to Eucalyptus and vise-versa were successful on each cloud environment.

Migration to OpenStack takes almost 6% more time than to Eucalyptus, due to better Eucalyptus’ architecture. Similar to migration, transfer of the application is also faster for 13.5% from OpenStack to Eucalyptus, rather than from Eucalyptus on OpenStack.

Finally, we observe that migration is much slower than the transfer between the clouds, which is normally since the former requires upload of CSAR from outside network.

V. CONCLUSION AND FUTURE WORK

This paper demonstrates the proof-of-concept and a procedure for porting a typical N-Tier application neither with changing the application nor source nor target environment. A P-TOSCA model of a typical N-tier application is defined.

Two demo scenarios are presented: migration from on-premise to two different clouds Eucalyptus and OpenStack, and transfer the application from one cloud to another. Not only that the porting is proved, but it is conducted in a reasonable time, even with definition of the P-TOSCA specification. That is, after preparing the P-TOSCA specification of an application, it can be used for efficient porting to a cloud. This portability demonstration can be generalized if an application is portable and uses open standards.

This is only a starting point for automated porting of an application of any type. Our further research will be to define and conduct the porting of other type of applications, such as software oriented architecture, and on other open source clouds. Special focus will be on improving the performance and definition of security rules.

ACKNOWLEDGMENT

This work was partially financed by the Faculty of Computer Science and Engineering at the "Ss. Cyril and Methodius University", Skopje, Macedonia through the project XaaSinCloud.

The iKnow project has been developed as a part of the Tempus JPGR 511342 Project.

REFERENCES