Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 97 (2016) 104 – 113

Procedia Computer Science

CLOUD FORWARD: From Distributed to Complete Computing, CF2016, 18-20 October 2016, Madrid, Spain

# Semantic Techniques for Multi-Cloud Applications Portability and Interoperability

Beniamino Di Martino[a], Antonio Esposito[a,*]

[a]*Second University of Naples, Department of Industrial and Information Engineering, Via Roma 29, Aversa (CE), 81031, Italy*

**Abstract**

The composition of Cloud Services to satisfy customer requirements is a complex task, owing to the huge number of services that are currently available. The advent of Big Data and Internet of Things (IoT), which rely on Cloud resources for better performances and scalability, is pushing researchers to find new solutions to the Cloud Services composition problem. In this paper a semantic-based representation of Application Patterns and Cloud Services is presented, with an example of its use in a typical distributed application, which shows how the proposed approach can be successfully employed for the discovery and composition of Cloud Services..



Peer-review under responsibility of organizing committee of the international conference on cloud forward: From Distributed to Complete Computing

*Keywords:* Cloud Computing; Semantics; Cloud Patterns; Application Patterns; Big Data; Distributed Algorithms

## 1. Introduction

Cloud Computing is a fast evolving and spreading resource sharing paradigm, which is being adopted by a growing number of companies to provide competitive services and functionalities, and at the same time to reduce upfront investments and management costs. The rapid adoption of Cloud solutions has pushed Cloud providers to continuously enhance and update their offers, releasing into the market a huge set of services. While this variety positively affects the market, in terms of services' costs and quality, it can however represent a hindrance for customers who have to select the right services suiting their exact needs and then compose them to obtain the desired functionality. In many cases, owing to the lack of a shared standard for services' interfaces description and incompatibilities between the adopted data formats, it can be difficult to effectively compose Cloud Services and exploit their full functionality. To provide a guide to service composition and application deployment on Cloud platforms, Patterns have been described by private companies, such as Amazon [1] and Microsoft [2], or have been defined as a result of research efforts [3 4]. In this paper we show how, exploiting a Cloud Pattern or a composition thereof, it is possible to guide a customer in building

* Corresponding author. Tel.: +39-377-186-2406
*E-mail address:* antonio.esposito@unina2.it


Peer-review under responsibility of organizing committee of the international conference on cloud forward: From Distributed to Complete Computing
doi:10.1016/j.procs.2016.08.285

her own Cloud application, regardless of her actual knowledge of the Cloud domain. In particular, we will show that, through a semantic representation of Application and Cloud Patterns, it is possible to support the development and deployment of Big Data, IoT and generally distributed applications.
The remainder of this paper is organized as follows: section 2 reports some existing approaches to Pattern description and a comparison with the **Topology and Orchestration Specification for Cloud Applications** (TOSCA) approach; section 3 introduces the Case Study used to describe our semantic approach; section 4 describes the semantic approach and how it is applied to the case study; section 5 draws conclusions and provides directions for future works.

## 2. State of the Art

A Design Pattern is defined as a general and reusable solution to a common and recurrent problem, within a given context in software design. The objective of a Design Pattern is to support developers in building their applications, suggesting fully functional solutions that are less error and bug prone than completely new implementations. Design Patterns provide both a static view of the architecture of the software, including its components (or participants) and their relationships, and a dynamic view of the interactions among such participants. The most famous Design Pattern catalogue is represented by[5], in which a set of 23 patterns has been introduced and deeply discussed. As of today, a number of Design Patterns catalogues exists for several purposes, like ontology creation[6 7] and definition of SOA-oriented applications[8 9]. Cloud Patterns can be considered as a particular Pattern category, focusing on the description of problems and solutions related to Cloud Computing. Cloud Patterns describe common aspects of cloud computing environments and of application design for cloud computing. They can be useful in understanding the changes to apply to an application, in terms of source code and architecture, in order to successfully migrate it from an in-house environment to the Cloud[10]. The use of Cloud Patterns for the design, implementation and management of Cloud Applications has been widely discussed[11 12 13].
Several online catalogues of Cloud Patterns have been published, and they are continuously updated. Vendor specific patterns, such as those published by Microsoft for their platform **Azure**[14] and by Amazon for **Amazon Web Services** (AWS)[1], are tailored for a target environment and provide optimized solutions for it. They provide many useful details regarding the actual Cloud components and services to use in order to deploy an application on the target platform, thus actively supporting developers in their work. Patterns have also been defined as a result of independent research efforts, such as those reported in the online catalogues accessible at[3] and[4]. Such patterns provide generic solutions, which are not bound to a specific platform and are therefore more flexible and seamlessly applicable to different targets. In the remainder we will refer to such patterns as **Agnostic Patterns**, because they are not related to a specific Cloud Platform and can virtually be applied to any target environment.

### *2.1. Orchestration and Composition of Cloud Services*

In the past few years the orchestration and composition of cloud services has been the topic of several initiatives and research efforts. The mOSAic Fp7 project[15] explicitly addressed the issues related to Cloud Services discovery and composition, by exploiting ontologies and semantic-web technologies to describe and annotate Cloud resources and then compose them through adapters and connectors. A similar approach to services composition is proposed in[16], where Cloud services' interfaces are described in terms of their inputs and outputs and a similarity function is applied to identify corresponding parameters and determine possible concatenations of services. These approaches do not refer to a Pattern collection, and require users to develop and compose their applications manually. By using a Pattern based approach, as in ours, users can leverage pre-defined solutions which already offer info on the services' composition.
Non-academical research efforts have produced user oriented solutions, with immediate applications to real-world situations. For an instance, orchestration and composition of Cloud Services is also the focus of the OpensStack **Heat** project[17], which has developed an interesting template-based formalism going under the name of **HOT**. While being compliant with the AWS **CloudFormation**[18] template format, HOT is still limited as regards the supported services and general expressivity. TOSCA[19], is an OASIS standard language used to describe both a topology of Cloud based web services, consisting in their components, relationships, and the processes that manage them, and the orchestration of such services, that is their complex behaviour in relation to other described services. The combination of topology