Journal of Network and Computer Applications 91 (2017) 61–74

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

Review

A survey on cloud service description

Souad Ghazouani[a,b,c,*], Yahya Slimani[a,b,c]

[a] Univ. Manouba, ISAMM, Campus Universitaire Manouba, 2010 Tunisia
[b] Univ. Carthage, INSAT, LR11ES26, Centre Urbain Nord BP 676, 1080 Tunis, Tunisia
[c] LISI Laboratory of Computer Science for Industrial Systems, Carthage University, Tunis, Tunisia

ARTICLE INFO

Keywords:
Cloud computing
Cloud service
Cloud service description
SaaS service
PaaS service
IaaS service
USDL

ABSTRACT

Cloud service description (CSD) becomes an active area which attracts the attention of many research organizations, while no standard CSD exists. This lack of standardization is caused by the vendor lock-in problem, where cloud providers use various techniques (languages, standards, ontologies, models, etc.) to describe cloud services. Furthermore, there are few studies which dealt with the four aspects of service especially the technical, operational, business, and semantic aspects. Whereas, many other studies dealt only with some of these four aspects. This is the case of WSDL (*Web Service Description Language*) language, which focuses in the technical aspect, but it does not cover business and semantic ones. Our objective in this paper is to identify the foundations of a standardized CSD covering the technical, operational, business, and semantic aspects. We present a comparative study of the different approaches that address CSD issues. The result of this comparative study shows that USDL (*Unified Service Description Language*) is the appropriate language supporting a service description by covering three aspects (technical, operational, and business), despite its inability to cover the semantic aspect.

1. Introduction

Cloud computing (Mell and Grance, 2011) offers several cloud services, which are usually located in three layers: IaaS (*Infrastructure as a Service*), PaaS (*Platform as a Service*), and SaaS (*Service as a Service*). These cloud services are available as independent components providing IT solutions for consumers via Internet.

The number of cloud service providers (Google, Microsoft Azure, Amazon, etc.) augments constantly, which leads to the increase of the number of cloud services. However, each service provider uses different techniques (such as languages (Sun et al., 2015; Oberle et al., 2013; Liu and Zic, 2011; Hoberg et al., 2012; Kantola and Karwowski, 2012; Junker et al., 2012; Shetty et al., 2015; Ye et al., 2012; Charfi et al., 2010; Zhou et al., 2011; Goscinski and Brock, 2010; Goncalves et al., 2011; Ma et al., 2010; jClouds, 2017; LibCloud, 2017; Deltacloud, 2017; cloud, 2017), standards (Galán et al., 2009; Dastjerdi et al., 2010; Martino et al., 2015), ontologies (Nagireddi and Mishra, 2013; Martino et al., 2014; Zhang et al., 2012; Karim et al., 2014; Mi^ndruta and Fortis, 2013; Tahamtan et al., 2012; Afify et al., 2013; Alfazi et al., 2015; Modica et al., 2012; Joshi et al., 2014; Metwally et al., 2015; Kang and Sim, 2011; Youseff et al., 2008; Han and Sim, 2010; Ali et al., 2014; Martino et al., 2014; Modica and Tomarchio, 2014; Fang et al., 2016; Moscato et al., 2011; Deng et al., 2011; Kaushik and Chana, 2012; Weinhardt et al., 2009; Androcec et al., 2012; Rekik et al., 2015), models (Lee et al., 2011; Hamdaqa et al., 2011; Amato and Moscato, 2014; Gudenkauf et al., 2013; Sun et al., 2011; Mastelic et al., 2014; Quinton et al., 2014; Cayirci, 2013; Cai et al., 2009; Rochwerger et al., 2009; Carlini et al., 2011), templates (Nguyen et al., 2012, 2011; García-Gómez et al., 2012; Zhang et al., 2014; Fowley et al., 2014; El Houssaini et al., 2015; Papazoglou and van den Heuvel, 2011; Papazoglou and Vaquero, 2012), or tree structure (Hoefer and Karagiannis, 2010) to represent its cloud services. In other words, each service provider provides a distinct interface and uses various protocols to access its services. So, we conclude that there is not a unified service description which describes all cloud service types (IaaS, PaaS, and SaaS). This lack of a standardized cloud service description (CSD) is caused by the vendor lock-in problem (Monteiro et al., 2011), which prevents the interoperability between cloud services.

Furthermore, cloud services are delivered by a provider to consumers for a defined period with pricing structures (Junker et al., 2012; Charfi et al., 2010), SLA (*Service Level Agreement*) (Kouki and Ledoux, 2013), and well-defined legal obligations (Barros and Oberle, 2012) between the parties implied in the cloud service lifecycle (consumers, providers, etc.). So, all these business aspects should be specified in the CSD.

Besides, existing service description approaches (Nguyen et al., 2012, 2011; Lee et al., 2011; Hamdaqa et al., 2011; Galán et al., 2009; Liu and Zic, 2011; Kantola and Karwowski, 2012; Junker et al., 2012;

* Corresponding author at: Univ. Manouba, ISAMM, Campus Universitaire Manouba, 2010 Tunisie.

http://dx.doi.org/10.1016/j.jnca.2017.04.013
Received 21 November 2016; Received in revised form 22 April 2017; Accepted 24 April 2017
Available online 27 April 2017
1084-8045/ © 2017 Elsevier Ltd. All rights reserved.

Mastelic et al., 2014; Quinton et al., 2014; Charfi et al., 2010; Goscinski and Brock, 2010; Goncalves et al., 2011; García-Gómez et al., 2012; Cayirci, 2013; Cai et al., 2009; Ma et al., 2010; Papazoglou and van den Heuvel, 2011; Papazoglou and Vaquero, 2012; Youseff et al., 2008) support some aspects and neglect others. For example, researchers in Nguyen et al. (2012, 2011); Lee et al. (2011); Hamdaqa et al. (2011); Galán et al. (2009); Liu and Zic (2011); Kantola and Karwowski (2012); Mastelic et al. (2014); Quinton et al. (2014); Goscinski and Brock (2010); Goncalves et al. (2011); García-Gómez et al. (2012); Cayirci (2013); Ma et al. (2010); Papazoglou and van den Heuvel (2011); Papazoglou and Vaquero (2012); Youseff et al. (2008) are focalized on technical and operational aspects, but they neglect business and semantic ones. Also, other works (Junker et al., 2012; Charfi et al., 2010; Cai et al., 2009) have addressed only the business aspect. However, USDL (*Unified Service Description Language*) (Barros and Oberle, 2012; Oberle et al., 2013; Hoberg et al., 2012; Cardoso et al., 2010; Charfi et al., 2010) is considered as the language that supports service description from three perspectives (technical, operational, and business). Nevertheless, it cannot describe the semantic aspect (Sun et al., 2012), and it is not intended for cloud computing (Barros and Oberle, 2012; Sun et al., 2012; Cardoso et al., 2010, 2009; Terzidis et al., 2012). Hence, proposing a generic description that supports different cloud service types with all aspects (technical, business, operational, and semantic), becomes a crucial need.

This paper provides a systematic literature review on state-of-the art approaches in addressing CSD. This research concentrates on the classification and comparison of previous CSD approaches, making a rigorous analysis and discussion by taking into account different comparison criteria (e.g. type of delivery model, type of the technique adopted, type of the obtained service representation, and the covered domain), and from diverse perspectives (technical, operational, business, and semantic).

The main purpose is to identify the benefits, as well as, the limits of the existing CSDs, in order to propose a standardized CSD that addresses business, technical, operational, and semantic aspects. The discussed advancements and developments of this topic provide interesting perspectives to motivate further investigations in this area.

The remainder of this paper is organized as follows. Section 2 constitutes an overview of the selected research methodology. Section 3 presents an overview of USDL language. In Section 4, we present and compare the existing CSD approaches. Section 5 highlights the new requirements for CSD. Finally, in Section 6, we report some conclusive considerations and an outlook on possible future work.

2. Research methodology

We rely on a Systematic Literature Review (SLR) (Kitchenham and Charters, 2007) to collect and investigate all the studies published on a service description in cloud computing. More specifically, the extraction of salient features and methods of papers will be taken into consideration, and their characteristics will be described.

To achieve this goal, only papers in English published from 2008 to 2016, were selected from different prestigious journals and international conferences. Studies that are not related to the service description, discovery, and composition in cloud computing were excluded.

In the present work, we collect works by searching journals, conference proceedings, workshops, and books from Google Scholar and some electronic scientific databases (ScienceDirect, SpringerLink, IEEE Xplore, etc.). However, irrelevant studies were excluded according to the analysis of their titles, abstracts, and keywords.

After filtering the publications list by reading titles, abstracts, and keywords, the selected article was fully read, in order to ensure that the content is related to our research topic. Finally, 65 studies identified from 2008 to March 2016 were selected and are considered in Section 4.

Table 1
Distribution per publication source types.

Source	Count
Conference proceedings	44
Journals	13
Book chapters	5
Workshops	3
Total	**65**

Most of these studies were published in conference proceedings, workshops, book chapters, and journals. Table 1 presents an overview of the distribution of the studies and the number of studies from a particular source type.

3. USDL overview

USDL (Barros and Oberle, 2012) is a language describing Internet services from three different perspectives (technical, operational, and business). This language aims to offer a platform-independent description. It does not replace other existing languages (like WSDL, WS-Policy, WS-BPEL, etc.), but it complements them.

USDL is organized into several packages according to UML terminology (Unified Modeling Language, 2017). Each package represents one USDL module and it contains one class model. Fig. 1 represents USDL modules (*Service, Functional, Technical, Interaction, Participant, Pricing, Legal, Service Level, and Foundation*) and their dependencies:

- *Service module*: this module describes the general information of a service, such as title, description, etc.
- *Functional module*: this module contains the information regarding to service capabilities, input-output parameters, and constraints.
- *Technical module*: this module describes how the service capabilities are mapped to the technical realization of a service, such as WSDL operations, parameters, defaults, etc.
- *Participant module*: this module describes the participating organizations, the persons, and their roles in the service realization.
- *Interaction module*: this module concerns behavioral service aspect. It describes the actors and the interactions between them.
- *Pricing module*: this module contains the information about the pricing features of a service (pricing plans, pricing components, etc.).
- *Service Level module*: it is responsible of the service quality. It specifies the service agreement level such as availability, performance, service execution time, security, etc.
- *Legal module*: it contains the information about conditions, licenses, and user rights for the participating parties.
- *Foundation module*: this module provides a set of common properties such as time, location, organization, etc. These properties are

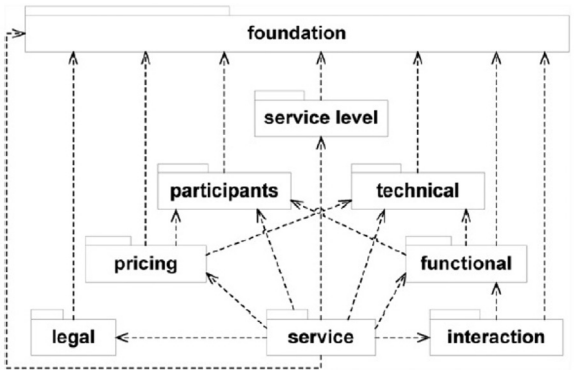

Fig. 1. UML package diagram of USDL (Barros and Oberle, 2012).

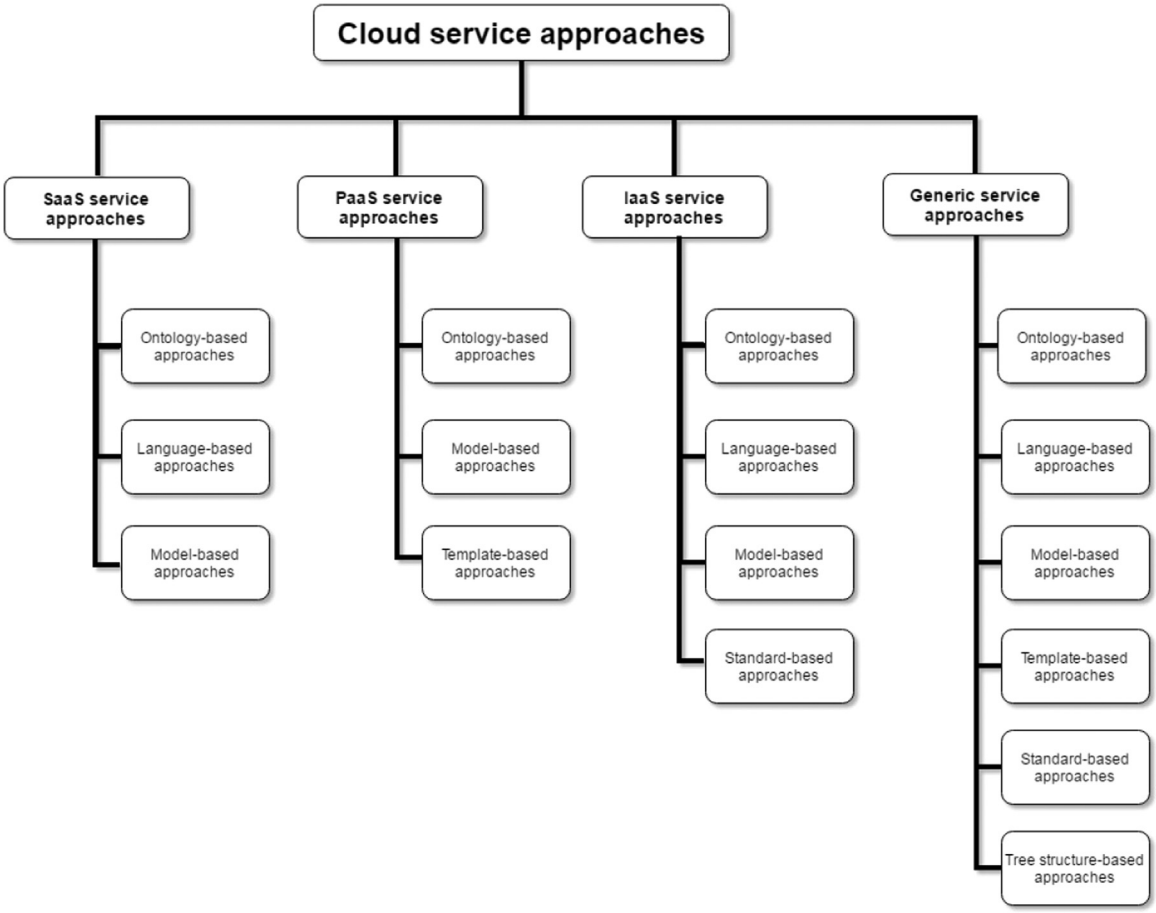

Fig. 2. Tree of cloud service approaches.

used by all modules.

Technical, operational, and business aspects are expressed by USDL modules:

- **Operational aspect** is covered by *Service*, *Functional*, and *Foundation* modules.
- **Technical aspect** is supported by *Technical* and *Foundation* modules.
- **Business aspect** is described by *Participant, Interaction, Pricing, Legal, Service Level,* and *Foundation* modules.

4. Cloud service approaches

We present, in this section, a broader vision on the studies of CSD in recent years. In cloud computing, services are delivered according to three delivery models (SaaS, PaaS, and IaaS). For this reason, we will classify these approaches according to the type of delivery models as shown in Fig. 2: (i) the first category represents the approaches that have addressed IaaS delivery model (Zhang et al., 2012; Lee et al., 2011; Galán et al., 2009; Liu and Zic, 2011; Dastjerdi et al., 2010; Sun et al., 2011; Goncalves et al., 2011; Metwally et al., 2015; Rochwerger et al., 2009; Carlini et al., 2011; jClouds, 2017; LibCloud, 2017; Deltacloud, 2017; cloud, 2017); (ii) the second category concerns approaches that have dealt with PaaS delivery model (Martino et al., 2014; Quinton et al., 2014; García-Gómez et al., 2012); (iii) the third category represents works that focus only on SaaS delivery model (Hamdaqa et al., 2011; Zhou et al., 2011; Afify et al., 2013; Goscinski and Brock, 2010; Modica et al., 2012; Modica and Tomarchio, 2014); (iv) however, the fourth category represents the approaches that are focalized on the generic description of all cloud services for SaaS, PaaS, and IaaS models (Nagireddi and Mishra, 2013; Sun et al., 2015; Oberle et al., 2013; Nguyen et al., 2012, 2011; Amato and Moscato, 2014; Karim et al., 2014; Hoberg et al., 2012; Gudenkauf et al., 2013; Mi^ndruta and Fortis, 2013; Kantola and Karwowski, 2012; Junker et al., 2012; Shetty et al., 2015; Hoefer and Karagiannis, 2010; Martino et al., 2015, 2014; Mastelic et al., 2014; Ye et al., 2012; Charfi et al., 2010; Tahamtan et al., 2012; Alfazi et al., 2015; Joshi et al., 2014; Cayirci, 2013; Cai et al., 2009; Ma et al., 2010; Zhang et al., 2014; Fowley et al., 2014; El Houssaini et al., 2015; Papazoglou and van den Heuvel, 2011; Papazoglou and Vaquero, 2012; Kang and Sim, 2011; Youseff et al., 2008; Han and Sim, 2010; Ali et al., 2014; Fang et al., 2016; Moscato et al., 2011; Deng et al., 2011; Kaushik and Chana, 2012; Weinhardt et al., 2009; Androcec et al., 2012; Rekik et al., 2015). Therefore, our first classification criterion is the type of service delivery model. For this reason, we dedicate four sections (i.e. 4.1. IaaS service approches, 4.2. PaaS service approches, 4.3. SaaS service approches, 4.4. Service delivery models-independent approaches). Then, we expand each section with subsections depending on the techniques used: ontologies, models, templates, languages, standards, or tree structure (see Fig. 2).

In Table 2, we summarize for each cloud service category the corresponding approaches.

4.1. IaaS service approaches

We classify, in this section, IaaS approaches according to the technique adopted (ontologies, models, languages, etc.). Therefore, we present approaches that use ontologies (Zhang et al., 2012; Metwally et al., 2015), models (Lee et al., 2011; Sun et al., 2011;

Table 2
Categorization of cloud service approaches.

Approaches	SaaS delivery model	PaaS delivery model	IaaS delivery model	All delivery models
Ontologies-based approaches	✓	✓	✓	✓
Models-based approaches	✓	✓	✓	✓
Templates-based approaches		✓		✓
Languages-based approaches	✓		✓	✓
Standards-based approaches			✓	✓
Tree structure-based approaches				✓

Rochwerger et al., 2009; Carlini et al., 2011), languages (Liu and Zic, 2011; Goncalves et al., 2011; jClouds, 2017; LibCloud, 2017; Deltacloud, 2017; cloud, 2017), or standards (Galán et al., 2009; Dastjerdi et al., 2010) to describe IaaS services.

4.1.1. Ontology-based approaches

In Zhang et al. (2012), Zhang et al. used ontologies to describe IaaS services. They proposed an OWL ontology called CoCoOn (*Cloud Computing Ontology*) which describes configuration information (functional and non-functional) related to IaaS services (computing, storage, and network). The proposed ontology is able to capture static and dynamic configuration of IaaS layer. Then, they implemented the proposed domain ontology as a relational model in the *CloudRecommender* system. This system uses SQL and regular expressions for matching user requests to service descriptions in order to discover and to select IaaS services.

Metwally et al. (2015) proposed a unified semantic ontology model with reasoning capability to represent, discover, and affect diverse cloud resources (compute, storage, and network) to IaaS requests.

4.1.2. Model-based approaches

Lee et al. (2011) proposed a generic abstraction model to describe IaaS services from multiple providers. Then, they provided a common interface to IaaS providers (AWS and GoGrid) for managing IaaS environments between public and private clouds. The integrated IaaS services are presented to users via a portal that allows them to register, monitor, and manage the lifecycle of IaaS services from multiple providers. The used environment in this work is Web2Exchange, which is a model-based service integration environment to facilitate service deployment, discovery, and integration. The service models used by Web2Exchange are described by MOF (*Managed Object Format*).

In Sun et al. (2011), authors proposed a constraints-based programming model to discover cloud resources and to deploy applications. The cloud programming model allows to map application requirements to cloud resources constraints. These resources are used to execute applications. The constraints may concern hardware, software, storage, data, security, performance, and compliance.

Rochwerger et al. (2009) proposed a federated cloud computing model, architecture, and functionality which are developed in the Reservoir project. The Reservoir model tackles the lack of interoperability among cloud providers by creating a federation of multiple cloud providers. However, a federation suffers from the lack of a standard language to describe providers' offers. As a consequence, the discovery and selection of an appropriate provider within a federation becomes difficult.

Carlini et al. (2011) proposed the Contrail project which provides a platform to implement an IaaS infrastructure and a PaaS platform. It allows cloud providers to integrate, in a transparent manner, the resources of other clouds with their own infrastructure in order to federate multiple clouds. Thus, Contrail enhances the portability of applications from one cloud to another. Contrail extends a SLA by integrating the SLA management approach of SLA@SOI project in the federation architecture. Besides, Contrail federation manages the user authentication and coordinates the applications deployment.

4.1.3. Language-based approaches

Liu and Zic (2011) proposed a specification language Cloud# in order to model the internal organization of cloud. Cloud# language models the cloud of infrastructure as a service on which customers launch their virtual machines. Authors proposed four cloud models: *basic cloud model*, *cloud model with fair scheduler*, *cloud model with storage allocation*, and *cloud model with security and monitoring mechanisms*. These models allow clients to understand how cloud services are delivered. By this way, service users have more confidence to migrate their business applications to cloud. Could# is used by providers to formally model their services, while the resulting models are used by customers in order to understand the behavior of cloud services.

Goncalves et al. (2011) proposed CloudML (*Cloud Modeling Language*) as an XML-based language extension, which allows to model service profiles and developer requests, and also, to represent virtual and physical resources status in D-Clouds (*Distributed Clouds*). CloudML is a description language able to represent different levels of abstraction, such as provider services, computational and network resources, and developer requirements. CloudML does not allow to describe scaling rules. It is expected that *RequestType* in CloudML would be extended to support rules which are described when new virtual nodes and virtual links can be added to the virtual network.

The multi-cloud libraries (jClouds (2017); LibCloud (2017); Deltacloud (2017); cloud (2017), etc.) provide access to multiple cloud platforms through a common interface (API). These libraries enable an abstract description of the services. The API helps to access and manage services provided by different cloud vendors in a uniform manner. These libraries enhance both portability and interoperability. Each library is implemented in a different programming language (for example, LibCloud uses Python, jClouds uses Java, DeltaCloud uses Ruby, and simpleAPI uses PHP). Among existing solutions, there is no high-level abstraction layer that hides the complexity of resources provision on several clouds.

4.1.4. Standard-based approaches

We start with the research work Galán et al. (2009) that adopted OVF standard (*Open Virtualization Format*) (DMTF, 2017) to define a service definition language for deploying Internet applications in IaaS clouds. These applications consist of a collection of virtual machines (VM) with several configuration settings (host names, IP, specific settings to application, etc.) for software components included in virtual machines (such as Web server, application server, database, operating system). Specifically, OVF standard enables to package applications as a collection of virtual machines in a single file called *OVF Package* in a portable, secure and technology-independent manner.

Also, Dastjerdi et al. (2010) adopted OVF standard to package the disk images with the user and hardware requirements into a standard format. In this work Dastjerdi et al. (2010), they proposed an automated approach to deploy virtual appliances on cloud service providers. The proposed approach allows to: (i) convert user requirements to OVF to be a standard package format for cloud deployment; (ii) propose an advertisement approach to IaaS providers; (iii) apply an

ontology-based discovery to find the best suitable providers.

4.2. PaaS service approaches

We will focus, in this section, on PaaS service approaches and we will classify them based on the technique used to describe PaaS service.

4.2.1. Ontology-based approaches

We mention, for example, the work Martino et al. (2014), where researchers proposed an OWL ontology (W3C, 2017) to represent cloud services and virtual appliances based on a common structure. This ontology is focused on functional characteristics for service providers and virtual appliances by defining the relationship between them in order to help consumers to choose the appropriate offers for them, and also, to enable an automatic composition of services and appliances from different vendors.

4.2.2. Model-based approaches

In Quinton et al. (2014), Quinton et al. proposed an approach for selecting a cloud environment, setting the configuration of this environment, and deploying applications. This approach is based on a combination of SPLs (*Software Product Lines*) and a domain model, which enables developers to: (i) automatically select a cloud environment that meets a set of requirements; (ii) automatically obtain description files and executable scripts to configure the related cloud environment. Authors proposed an extension of feature models (FMs) with cardinalities and attributes as variability models to describe cloud environment.

4.2.3. Template-based approaches

The work García-Gómez et al. (2012) is about 4CaaSt project which aims to create a PaaS framework for defining, marketing, deploying, and managing cloud-based services and applications. 4CaaSt introduces the concept of blueprint as a technical description of an application or a service.

4.3. SaaS service approaches

We will classify SaaS service approaches into: approaches using ontologies (Afify et al., 2013; Goscinski and Brock, 2010; Modica et al., 2012), approaches using models (Hamdaqa et al., 2011), and approaches using languages (Zhou et al., 2011).

4.3.1. Language-based approaches

Multiple research studies have considered cloud services as Web services. Therefore, they used WSDL (W3C, 2017) language to describe them by using the same parameters such as: inputs, outputs, service name, service location, etc. WSDL is the most used representation that provides an XML-based standard interface definition for software components implemented in different languages. WSDL defines Web services from technical aspect such as interaction interface, protocols, exchanged messages, etc. Service location is described in WSDL document, while non-functional attributes are not defined. WSDL is a popular language, but it suffers from a lack of semantic support to precise the sense and the data constraints integrated in Web service. This limitation allows to generate ambiguities during service discovery process.

Zhou et al. (2011) used WSDL-S (W3C, 2016) to semantically describe services that will be discovered. They proposed a P2P-based unstructured method for SaaS discovery. They developed a localized search scheme by using semantic routing protocols and topology reconstruction techniques for service query routing.

Goscinski and Brock (2010) proposed a framework based on dynamic attributes and Web services to represent services and resources in cloud. This cloud service representation helps to facilitate service publication, discovery, and selection. Goscinski et al. proposed a WSDL file extension to take into account cloud resources characteristics. Also, they introduced *Cluster-as-a-Service* as a regrouping means of services and resources, in order to facilitate their publication and selection.

4.3.2. Ontology-based approaches

In Afify et al. (2013), authors proposed a publication, discovery, and selection system based on a cloud ontology that describes various aspects of SaaS. The unified ontology integrates service domain knowledge, QoS metrics, SaaS characteristics, and real SaaS offers.

The proposals of Modica et al. (2012); Modica and Tomarchio (2014) aim to help providers to better characterize and advertise their offers according to the business strategy on the one hand, and customers to define resource requests according to their business needs on the other hand. To reach this goal, they developed a set of ontologies (*Application Ontology, Support Ontology, SLA Ontology, Market Ontology, Offer Ontology*, and *Request Ontology*) to express the business aspects of providers' offerings and customers' requests. The proposed cloud ontology is an OWL ontology and it is focused on the business aspect rather than the technical aspect.

4.3.3. Model-based approaches

Hamdaqa et al. (2011) proposed a meta-model that enables cloud users to design elastic applications as a service, independently from any platform to prevent the vendor lock-in problem. This meta-model is able to describe the syntax of cloud applications (capabilities and technical interfaces), but it is able to capture just some semantics.

4.4. Service delivery models-independent approaches

Several initiatives have emerged in recent years that aim at providing a uniform representation for all cloud services types (SaaS, PaaS, and IaaS). We classify the approaches of this category according to the technique used (models (Amato and Moscato, 2014; Gudenkauf et al., 2013; Mastelic et al., 2014; Cayirci, 2013; Cai et al., 2009), templates (Nguyen et al., 2012, 2011; Zhang et al., 2014; Fowley et al., 2014; El Houssaini et al., 2015; Papazoglou and van den Heuvel, 2011; Papazoglou and Vaquero, 2012), languages (Sun et al., 2015; Oberle et al., 2013; Hoberg et al., 2012; Kantola and Karwowski, 2012; Junker et al., 2012; Shetty et al., 2015; Ye et al., 2012; Charfi et al., 2010; Ma et al., 2010), ontologies (Nagireddi and Mishra, 2013; Karim et al., 2014; Mi^ndruta and Fortis, 2013; Tahamtan et al., 2012; Alfazi et al., 2015; Joshi et al., 2014; Kang and Sim, 2011; Youseff et al., 2008; Han and Sim, 2010; Ali et al., 2014; Martino et al., 2014; Fang et al., 2016; Moscato et al., 2011; Deng et al., 2011; Kaushik and Chana, 2012; Weinhardt et al., 2009; Androcec et al., 2012; Rekik et al., 2015), standards (Martino et al., 2015), and tree structure (Hoefer and Karagiannis, 2010) in the service definition.

4.4.1. Model-based approaches for the three layers

Gudenkauf et al. (2013) proposed a reference architecture as a feature model to describe service offers in a uniform manner. This model contains nine features; namely service type, deployment, pricing, role, service integration aspects, sourcing options, SLA, organization specific aspects, and cloud service characteristics. They also developed *Cloud Service Navigators* (CSNs) as a visualization technique to describe visually and to compare cloud service offers and requests. The main characteristic of this approach is to detect rapidly the service description.

Authors in Mastelic et al. (2014) proposed a model-driven approach to establish and manage arbitrary cloud services. They defined a meta-model of a cloud service called CoPS which describes cloud service as a composition of software and hardware elements by using three models: component, product, and service. This meta-model tends to obtain a uniform cloud service representation. The authors, also, proposed an architecture of *Cloud Management System* (CMS), which is able to

manage cloud services by automatically transforming service models from an abstract representation to a real deployment. The service components are represented by templates as black boxes. The transformation to the implementation model and the final deployment is achieved by plugins. The modularity of CoPS models allows CMS architecture to reuse templates and plugins for several cloud services. The modules and interfaces are described by UML.

Amato and Moscato (2014) described MeraMORP(h)OSY methodology based on multi-agent models which allows the description, the composition, and the verification of cloud service requirements. The methodology uses a MDE-based modeling profile able to describe services as agents in a multi-agent environment. This methodology includes a verification process for requirements that exploit formal methods during the service lifecycle.

In Cayirci (2013), Cayirci et al. proposed MSaaS (*Modeling and Simulation as a Service*), which is a model for provisioning modeling and simulation (M & S) services from a cloud service provider. MSaaS maintains the infrastructure, the platform, and the software details hidden from users. Cayirci (2013) considered MSaaS as a derivation of SaaS and PaaS. There are three types of MSaaS: (i) modeling as a service; (ii) model as a service; and, (iii) simulation as a service. Authors defined MSaaS architecture which is based on the deployment models (public, private, hybrid, and community). This architecture allows MSaaS service to be provided by any type of cloud. MSaaS is designed in one of the four following forms: *Standalone MSaaS applications*, *Federated standalone MSaaS applications*, *Composed MSaaS*, and *Automatically composed MSaaS*. Then, they presented the privacy and security. Besides, they introduced the composition of MSaaS from services provided by multiple clouds.

Cai et al. (2009) aimed at providing a customer-centric cloud service model to look into the relationship between cloud service providers and customers. The proposed model is focalized on the business side of cloud computing and particularly on customer business requirements, but it does not cover technical aspect.

4.4.2. Template-based approaches for the three layers

Nguyen et al. (2011) proposed a uniform representation called *Blueprint Template* to describe cloud services, in order to capture the knowledge of cloud service offers. The proposed representation supports SBA (*Service Based Application*) developers during different development phases. This representation allows: (i) to help the application developers to choose the offers of several software, platforms, and infrastructures, and to configure them dynamically in an optimal manner to meet their application requirements; (ii) to combine various independent services that require a uniform description format which facilitates their conception, personalization, and composition. This approach gives more importance to the technical aspect such as the deployment.

In Nguyen et al. (2012), researchers defined a blueprint as an abstract and uniform description of a cloud service offering that hides all complexities and technical details in order to facilitate the selection, the personalization, and the composition of cloud services coming from different layers. They proposed *Blueprint Template* as structural schema which is used to outline and personalize the blueprint. They achieved the first version of the template in the EC's 4caaSt FP7 project (Morfeo 4caaSt, 2017). In this work, they presented a universal *Blueprint lifecycle* that outlines: (i) how cloud services are described by blueprints; and, (ii) how are designed, composed, and deployed on cloud infrastructure. The lifecycle is composed of six steps (*target blueprint design, target blueprint resolution, target blueprint customization, target blueprint Checking & Testing, target blueprint deployment, and target blueprint monitoring*).

Authors, in Zhang et al. (2014) and Fowley et al. (2014), developed a service description template that describes the mediated and integrated cloud services. The structure of this template is composed of four parts: operation, quality, resources, and policies. These components are formalized in the form of an ontology and mapped onto a set of sublanguages (*Service Template Definition Language*, *Service Template Configuration/Instantiation Language*, and *Service Template Constraints Language*).

El Houssaini et al. (2015) proposed a cloud service template divided into three parts: (i) *Service Description*: it represents the functional requirements that allow to filter the offers and to realize a perfect need mapping; (ii) *Service Parameters*: it represents non-functional requirements that help to compare and classify the offerings; and, (iii) *Service Migration*: it represents the migration requirements that help to evaluate the migration feasibility before decision making. Researchers proposed a context-awareness mechanism to enable the automated service template adaptation.

Papazoglou and van den Heuvel (2011); Papazoglou and Vaquero (2012) presented a cloud blueprinting approach. This approach improves cloud service interoperability and prevents the vendor lock-in problem. It relies on the use of the cloud service definition, constraint specification, and manipulation languages.

4.4.3. Tree structure-based approaches for the three layers

The work Hoefer and Karagiannis (2010) presented cloud services and their features in the form of a taxonomy based on a tree-structure to classify them. This taxonomy is based on the current cloud services (Google App Engine, SalesFroce, and Amazon EC2) and the principal features such as the service category, the license type, the target user groups, the payment system, the formal agreement, the security measures, and the standardization efforts.

4.4.4. Ontology-based approaches for the three layers

Nagireddi and Mishra (2013) used an ontology to represent cloud services and their features in OWL, to facilitate cloud service discovery according to user needs. These services are classified based on the AHP (*Analytic Hierarchy Process*) mechanism by the taxonomy of cloud services attributes.

In Karim et al. (2014), authors used an OWL-S ontology (Martin et al., 2004) to semantically define cloud services and their associated QoS properties. This ontology provides the knowledge base for the mapping process. Also, they used SWRL (*Semantic Web Rule Language*) to define the mapping rules and the computation models to integrate them into the OWL-S ontology.

Mi^ndruta and Fortis (2013) used the WSMO ontology (Roman et al., 2006) to describe cloud services, in order to facilitate their semantic discovery and composition. They defined cloud service as a Web service using WSML.

Tahamtan et al. (2012) proposed a framework based on a cloud ontology to store and find cloud services according to functional and non-functional user requirements. They have proposed a unified ontology for both business functions and cloud providers. This ontology describes multiple cloud service types (SaaS, PaaS, and IaaS).

Alfazi et al. (2015) proposed a cloud service search engine which employs an ontology to discover and categorize cloud services. They developed a cloud service ontology based on NIST cloud computing standard. They used the cloud service ontology concepts to categorize the cloud services into several clusters. This ontology provides a limited number of concepts, because it is formalized with NIST standard.

In Kang and Sim (2011), Kang et al. proposed a search engine for cloud computing system named *Cloudle*. They presented two cloud ontologies: old cloud ontology (CO-1) and new cloud ontology (CO-2). Moreover, they compared these two ontologies; noting that CO-1 contains only concepts while CO-2 contains a set of cloud concepts, individuals of these concepts, and the relationship among these individuals. CO-2 is used to determine the similarity among cloud services with three kinds of reasoning methods: (i) concept similarity reasoning; (ii) object property similarity reasoning; and, (iii) datatype property similarity reasoning.

Youseff et al. (2008) proposed the first attempt to establish a

detailed ontology for cloud. They suggest that clouds have 5 layers (*Cloud Application Layer, Cloud Software Environment, Cloud Software Infrastructure, Software Kernel*, and *Firmware/ Hardware*). They work at establishing the knowledge domain of cloud computing and its relevant components. Furthermore, they used the composability as a methodology to construct the cloud ontology, since this methodology allows the proposed ontology to capture relations between different cloud components.

Han and Sim (2010) built a cloud service discovery system (CSDS) to find cloud services over Internet. In addition, they constructed a cloud ontology for enhancing the performance of CSDS. The proposed cloud ontology consists of 424 concepts. It is the first attempt to build an agent-based discovery system that consults ontology in the time of capturing information about cloud services.

Ali et al. (2014) presents a domain ontological model for representing different cloud services. They followed a methodology called METHONTOLOGY to construct the ontology and they used Protégé tool to represent and to check the consistency of this ontology. The proposed ontology covers different aspects of cloud computing such as cloud services models, cloud computing deployment, service level agreements, QoS, etc. The cloud concepts of this ontology will be essential in the cloud service discovery and the cloud computing management.

Martino et al. (2014) proposed a semantic description of some cloud services by defining them in terms of functionalities, parameters, and collaboration with other services. Particularly, they presented a semantic description of Windows Azure APIs, which describes the functional and non-functional properties of the service using OWL-S.

In Fang et al. (2016), Fang et al. proposed a cloud service semantic model named AoFeCSO (*Agility-oriented and Fuzziness-embedded Cloud Service Ontology*), which adopts an agility-centric design along with OWL2 (*Web Ontology Language*) fuzzy extensions. The fuzziness accepts rating updates from users, which consequently allows the captured cloud service specifications to be maintained in a collaborative manner. Users can explore the model and also, contribute their own knowledge to it interactively, which enhances the presentation of cloud service information. This model allows to capture cloud concept details and their interactions, which offers a comprehensive service specification. The model is used as a knowledge base.

Moscato et al. (2011) proposed a detailed ontology for cloud systems which can be used to improve the interoperability among cloud Solutions, platforms, and services, both from end-users and developers side. The mOSAIC ontology is used to semantically retrieve and compose cloud services in the mOSAIC project. It has been developed in OWL.

In Deng et al. (2011), Deng et al. used an ontology to model the structure and relationships of service offerings and their operational processes. They introduced the notion of *safe sequences* to the ontological model to provide a transactional support for cloud services.

Kaushik and Chana (2012) proposed an ontology-based cloud framework. The use of an ontology facilitates the access and update of cloud by using semantic Web queries. It is possible to improve the cloud by importing several ontologies into this framework, for example resource management, service discovery, etc.

Weinhardt et al. (2009) proposed a *Cloud Business Model Ontology* which offers a framework to characterize and classify cloud offerings. It can be used by cloud users and providers to map products, to identify customers and providers, and to set pricing schemes.

Androcec et al. (2012) presented a review of cloud computing ontologies. They identify four categories of studies: (i) *cloud resources and services description studies* that use the ontologies to describe cloud resources and services or to define new types of cloud services; (ii) *cloud security studies* that describe the cloud security; (iii) *cloud interoperability studies* that use ontologies to establish the interoperability among different cloud providers and their services; and, (iv) *cloud services discovery and selection studies* that focus on the discovery and selection of the best cloud services.

In Joshi et al. (2014), Joshi et al. proposed a methodology for the lifecycle of IT services in cloud and they showed how it can be used to represent services and service requirements. They have divided the IT service lifecycle into five phases: requirements, discovery, negotiation, composition, and consumption. They detailed each phase and described the developed ontologies to define the concepts and the relationships for each phase.

Rekik et al. (2015) proposed a cloud description ontology that covers the three layers (IaaS, PaaS and SaaS). The proposed ontology treats the functional and non-functional properties, the attributes and relations of infrastructure, the platform and software services in order to facilitate the discovery and selection of suitable cloud services.

4.4.5. Language-based approaches for the three layers

In Kantola and Karwowski (2012), Kantola et al. divided cloud service domain languages in three languages: CSDL (*Cloud Service Declarative Definition Language*), CSCL (*Cloud Service Constraint Language*), and CSML (*Cloud Service Manipulation Language*). These cloud specification languages are focused on the technical and operational perspectives, such as the service configuration, the service manipulation, etc., but they don't support the business perspective such as the service discovery, the service selection, etc. Most of the public cloud services are delivered by Web service interfaces.

Ye et al. (2012) proposed CloudUDDI, which is an extended version of UDDI model. This proposed model allows to represent and store the quality and the low-level resources information of cloud services, in order to facilitate the storage and selection of the services. Ye et al. defined the quality model (QoS model), the *Resource of Service* model (RoS model), and the mapping between QoS parameters and lower-level resource metric. The QoS model includes availability, response time, processing time, reputation, cost, while the RoS model includes computing resources, network resources, storage resources, etc. The mapping rules between QoS parameters and resources metrics are stored in an XML or OWL document. The CloudUDDI architecture proposed by Ye et al. (2012) consists of physical devices, virtual resource layer, and middleware layer that includes multiple agents, such as search middleware and registering middleware.

Oberle et al. (2013) adopted USDL in cloud computing as a language addressing all services types. They presented the version 3.0 of USDL. They divided USDL into nine modules (*Service, Functional, Technical, Participant, Interaction, Service Level, Pricing, Legal, and Foundation*). USDL tools are developed by SAP Research and they include editors, repositories, and marketplaces.

Charfi et al. (2010) presented the USDL language meta-model and they focused particularly on the pricing side. In this work, USDL is based on a well-defined Ecore meta-model Ecore (2017). It is divided into eight modules (*Foundation, Service, Functional, Service Level, Pricing, Legal, Participant*, and *Interaction*). This meta-model does not contain a Technical module. The authors used USDL Editor as an Eclipse-based tool to create service descriptions and to serialize them into XML.

In Hoberg et al. (2012), Hoberg et al. extended USDL language in order: (i) to provide a service description covering all the required information to select services; and, (ii) to make it adaptable to cloud computing domain. The extended USDL is also a syntactic cloud service description language. Authors in Hoberg et al. (2012) divided USDL into nine modules (*Foundation, Service, Functional, Technical, Service Level, Pricing, Legal, Participant, and Interaction*). They redefined these modules by integrating cloud service features required by consumers. They aimed to help consumers to structure their service selections, and thus, to help cloud providers to create service descriptions according to consumer needs.

Junker et al. (2012) used USDL to define the price models of commercial services on the marketplaces. They introduced an aggregation approach of several price models into one price model to resolve

the pricing problem of composite service in cloud computing environment. They analyzed too the time complexity of the price aggregation algorithm.

Sun et al. (2015) extended the basic structure of USDL with specific attributes to cloud services, in order to obtain a semantic cloud service description model named CSDM (*Cloud Service Description Model*). This latter covers technical, operational, business, and semantic aspects related to cloud services. It splits the service information into ten modules (*Foundation, Service, Functional, Technical, Service Level, Pricing, Legal, Participant, Interaction, and Transaction*). These modules support different specification aspects. They added a new module called *transaction module* to the basic USDL. This new module captures concepts that measure and evaluate the relevant factors to the service transaction. These factors cover the rating systems (transactional risk assessment, trust analysis, and reputation evaluation).

Shetty et al. (2015) proposed a representation model to store the description of the infrastructure services using XML language. The proposed data representation model helps users to discover services through the syntactic and semantic information. This model may be used by providers in order to publish their services. It consists of service name, functional and non-functional properties, and server locations. These properties may be qualitative (operating system, security mechanism, etc.) or quantitative.

Ma et al. (2010) proposed a formal framework for service discovery and composition. They proposed an XML-based cloud service description in which the services are modelled by *Abstract State Services*. They used ABox and TBox statements related to XML Schema and XML to define services. Also, they used XQuery to discover services. But, this work did not differentiate between the different levels of cloud services and their functional and non-functional requirements.

Authors in Sun et al. (2012) presented a review of service description languages in three domains (general, SOA, and cloud computing). They compared the existing approaches on seven criteria: domain, coverage, objective, representation, semantic expressivity, intended users, and features. Authors in Sun et al. (2012) introduced various service description languages: (i) general Internet service description languages such as USDL; (ii) Web/SOA services description languages such as WSDL, SoaML, WSDL-S, OWL-S, and SAWSDL; and, (iii) cloud services description languages such as SMI (*Service Measurement Index*), Blueprint, Cloud#, and programming model. They concluded that USDL is a language that covers all three aspects (technical, operational, and business) but it lacks the semantic one.

4.4.6. Standard-based approaches for the three layers

In Martino et al. (2015), Martino et al. proposed two standards to describe and orchestrate cloud services: TOSCA (*Topology and Orchestration Specification for Cloud Applications*) and HOT (*Heat Orchestration Template*). These standards are used to represent cloud patterns. This latter is considered as a mean to describe the composition and the orchestration of cloud services in order to satisfy application requirements. TOSCA is used to describe services and their relations, but it is not intended to describe IaaS infrastructure. Thus, it describes cloud components as services at different layers. To manage the infrastructure, there are other standards such CIMI (*Cloud Infrastructure Management Interface*), which are more appropriate for providers to manage cloud infrastructure by a service described by TOSCA and CIMI (Davis and Pilz, 2012). TOSCA and HOT are almost similar but they differ in some ways. On the one hand, TOSCA is more flexible. On the other hand, it supports service composition and orchestration at different layers (SaaS, PaaS, and IaaS).

4.5. Comparison between cloud service approaches

We compare, in Table 3, cloud service approaches according to four comparison criteria: (i) the type of delivery models; (ii) the technique used; (iii) the proposed cloud service representation; and, (iv) the covered domain.

4.5.1. Type of delivery models

In cloud computing, services are delivered according to three delivery models (SaaS, PaaS, and IaaS). From this point, we classify cloud service approaches according to the type of the delivery models. So, we find approaches that have addressed only SaaS delivery model (Hamdaqa et al., 2011; Zhou et al., 2011; Afify et al., 2013; Goscinski and Brock, 2010; Modica et al., 2012; Modica and Tomarchio, 2014), approaches that have dealt with PaaS delivery model (Martino et al., 2014; Quinton et al., 2014; García-Gómez et al., 2012; Modica et al., 2012; Modica and Tomarchio, 2014), and approaches for IaaS delivery model (Zhang et al., 2012; Lee et al., 2011; Galán et al., 2009; Liu and Zic, 2011; Dastjerdi et al., 2010; Sun et al., 2011; Goncalves et al., 2011; Metwally et al., 2015; Rochwerger et al., 2009; Carlini et al., 2011; jClouds, 2017; LibCloud, 2017; Deltacloud, 2017; cloud, 2017). Also, there are other approaches focalized on a generic description of all cloud services for SaaS, PaaS, and IaaS delivery models (Nagireddi and Mishra, 2013; Sun et al., 2015; Oberle et al., 2013; Nguyen et al., 2012, 2011; Amato and Moscato, 2014; Karim et al., 2014; Hoberg et al., 2012; Gudenkauf et al., 2013; Miˆndruta and Fortis, 2013; Kantola and Karwowski, 2012; Junker et al., 2012; Shetty et al., 2015; Hoefer and Karagiannis, 2010; Martino et al., 2015, 2014; Mastelic et al., 2014; Ye et al., 2012; Charfi et al., 2010; Tahamtan et al., 2012; Alfazi et al., 2015; Joshi et al., 2014; Cayirci, 2013; Cai et al., 2009; Ma et al., 2010; Zhang et al., 2014; Fowley et al., 2014; El Houssaini et al., 2015; Papazoglou and van den Heuvel, 2011; Papazoglou and Vaquero, 2012; Kang and Sim, 2011; Youseff et al., 2008; Han and Sim, 2010; Ali et al., 2014; Fang et al., 2016; Moscato et al., 2011; Deng et al., 2011; Kaushik and Chana, 2012; Weinhardt et al., 2009; Androcec et al., 2012; Rekik et al., 2015). Therefore, our first classification criterion is the type of delivery models.

As indicated in Fig. 3, we notice that current works are moving towards the proposal of a generic CSD (64%). Another trend turns to IaaS (22%) and SaaS (9%) layers. While works on PaaS layer (5%) are still immature.

4.5.2. Techniques used

We note that each approach uses a specific technique to describe cloud services: standards (Galán et al., 2009; Dastjerdi et al., 2010; Martino et al., 2015), languages (Sun et al., 2015; Oberle et al., 2013; Liu and Zic, 2011; Hoberg et al., 2012; Kantola and Karwowski, 2012; Junker et al., 2012; Shetty et al., 2015; Ye et al., 2012; Charfi et al., 2010; Zhou et al., 2011; Goscinski and Brock, 2010; Goncalves et al., 2011; Ma et al., 2010; jClouds, 2017; LibCloud, 2017; Deltacloud, 2017; cloud, 2017), ontologies (Nagireddi and Mishra, 2013; Martino et al., 2014; Zhang et al., 2012; Karim et al., 2014; Miˆndruta and Fortis, 2013; Tahamtan et al., 2012; Afify et al., 2013; Alfazi et al., 2015; Modica et al., 2012; Joshi et al., 2014; Metwally et al., 2015; Kang and Sim, 2011; Youseff et al., 2008; Han and Sim, 2010; Ali et al., 2014; Martino et al., 2014; Modica and Tomarchio, 2014; Fang et al., 2016; Moscato et al., 2011; Deng et al., 2011; Kaushik and Chana, 2012; Weinhardt et al., 2009; Androcec et al., 2012; Rekik et al., 2015), models (Lee et al., 2011; Hamdaqa et al., 2011; Amato and Moscato, 2014; Gudenkauf et al., 2013; Sun et al., 2011; Mastelic et al., 2014; Quinton et al., 2014; Cayirci, 2013; Cai et al., 2009; Rochwerger et al., 2009; Carlini et al., 2011), templates (Nguyen et al., 2012, 2011; García-Gómez et al., 2012; Zhang et al., 2014; Fowley et al., 2014; El Houssaini et al., 2015; Papazoglou and van den Heuvel, 2011; Papazoglou and Vaquero, 2012), or tree structure (Hoefer and Karagiannis, 2010). This means that there is no standardized CSD.

The choice of a technique to describe cloud services is done according to the objective of the research work (description, discovery, or composition, etc).

As shown in Fig. 4, most of the works use to describe cloud services

Table 3
Comparison of cloud service approaches.

Research work	Technique used	Proposed representation	Delivery models	Covered domain
Nagireddi and Mishra (2013)	Ontology (OWL)	Cloud ontology	SaaS, PaaS, IaaS	Description, discovery
Martino et al. (2014)	Ontology (OWL)	Cloud ontology	SaaS, PaaS	Description
Zhang et al. (2012)	Ontology (OWL)	CoCoOn ontology	IaaS	Discovery, selection
Karim et al. (2014)	Ontology (OWL-S)	Cloud QoS ontology	SaaS, PaaS, IaaS	Selection
Miˆndruta and Fortis (2013)	Ontology (WSMO)	Cloud ontology	SaaS, PaaS, IaaS	Discovery, composition
Tahamtan et al. (2012)	Ontology (OWL)	Cloud service ontology	SaaS, PaaS, IaaS	Description, storage
Afify et al. (2013)	Ontology (OWL)	Unified ontology	SaaS	Discovery, selection
Alfazi et al. (2015)	Ontology	Cloud service ontology	SaaS, PaaS, IaaS	Description, Discovery
Modica et al. (2012)	Ontology (OWL)	Business Ontology	SaaS, PaaS	Description, discovery
Lee et al. (2011)	Model (MOF–based meta-model)	Generic abstraction model	IaaS	Composition
Hamdaqa et al. (2011)	Model	Reference model of cloud applications	SaaS	Description, discovery
Amato and Moscato (2014)	Model (UML model)	Extended UML profile	SaaS, PaaS, IaaS	Description, composition
Gudenkauf et al. (2013)	Model (Feature model)	Cloud service feature model	SaaS, PaaS, IaaS	Description
Sun et al. (2011)	Model (Programming model)	Programming model	IaaS	Discovery
Mastelic et al. (2014)	Model (UML model)	CoPS cloud service model	SaaS, PaaS, IaaS	Composition
Quinton et al. (2014)	Model (Feature model, Domain knowledge model)	Feature model, Cloud knowledge model, Mapping relationships	PaaS	Selection
Sun et al. (2015)	Language (USDL)	CSDM model	SaaS, PaaS, IaaS	Description
Liu and Zic (2011)	Language (Cloud#)	Cloud models	IaaS	Description
Hoberg et al. (2012)	Language (USDL)	Cloud models	SaaS, PaaS, IaaS	Description
Kantola and Karwowski (2012)	Language (CSDL, CSCL, and CSM)	CSDL, CSCL, and CSM languages	SaaS, PaaS, IaaS	Description
Junker et al. (2012)	Language (USDL)	Cloud models	SaaS, PaaS, IaaS	Composition
Shetty et al. (2015)	Language (XML)	Data representation model	SaaS, PaaS, IaaS	Discovery, selection
Ye et al. (2012)	XML, OWL	QoS model, RoS model	SaaS, PaaS, IaaS	Description, storage
Zhou et al. (2011)	Language (WSDL-S)	WSDL-S	SaaS	Discovery
Goscinski and Brock (2010)	Language (WSDL)	WSDL	SaaS	Publication, discovery, selection
Nguyen et al. (2012)	Blueprint Template	Blueprint Template	SaaS, PaaS, IaaS	Description
García-Gómez et al. (2012)	Blueprint Template	Blueprint Template	PaaS	Description
Galán et al. (2009)	Standard (OVF)	OVF package	IaaS	Description
Dastjerdi et al. (2010)	Standard (OVF)	OVF Package, Requirements ontology, Virtual units ontology	IaaS	Discovery
Martino et al. (2015)	Standard (TOSCA, OpenStack HOT)	Cloud patterns	SaaS, PaaS, IaaS	Composition
Hoefer and Karagiannis (2010)	Tree structure	Taxonomy based on a tree structure	SaaS, PaaS, IaaS	Description

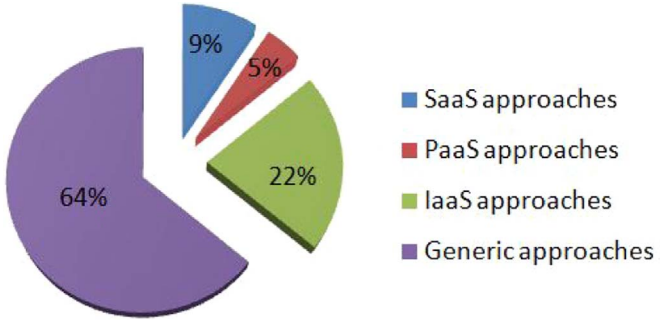

Fig. 3. Categories of cloud service approaches.

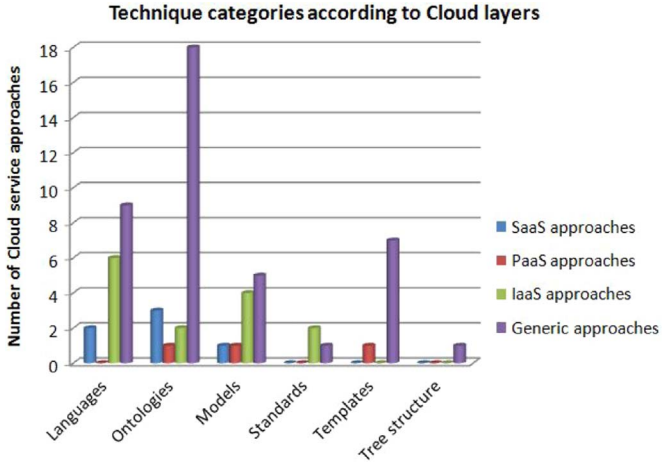

Fig. 4. Categories of the techniques used according to cloud layers.

respectively: ontologies (24 appraoches), languages (17 approaches), models (11 approaches), templates (8 approaches), standards (3 approaches), and tree structure (1 approach).

- IaaS approaches used ontologies (2 approaches), languages (6 approaches), models (4 approaches), and standards (2 approaches).
- PaaS approaches used ontologies (1 approach), models (1 approach), and templates (1 approach).
- SaaS approaches used ontologies (3 approaches), languages (2 approaches), and models (1 approach).
- Generic approaches used ontologies (18 approaches), languages (9 approaches), templates (7 approaches), models (5 approaches), standards (1 approach), and tree structure (1 approach).

We conclude that a great amount of works move to propose a semantic CSD.

Table 4
Comparison of cloud service approaches according to technical, operational, business, and semantic aspects.

Research work	Technical aspect	Operational aspect	Business aspect	Semantic aspect
Martino et al. (2014), Zhang et al. (2012), Amato and Moscato (2014), Miˆndruta and Fortis (2013), Martino et al. (2015), Zhou et al. (2011), Afify et al. (2013), Alfazi et al. (2015), Metwally et al. (2015), Kang and Sim (2011), Han and Sim (2010), Martino et al. (2014), Deng et al. (2011), Kaushik and Chana (2012),	✓	✓		✓
Nguyen et al. (2012), Nguyen et al. (2011), Lee et al. (2011), Hamdaqa et al. (2011), Galán et al. (2009), Liu and Zic (2011), Kantola and Karwowski (2012), Mastelic et al. (2014), Quinton et al. (2014), Goscinski and Brock (2010), Goncalves et al. (2011), García-Gómez et al. (2012), Cayirci (2013), Ma et al. (2010), Papazoglou and van den Heuvel (2011), Papazoglou and Vaquero (2012), Youseff et al. (2008)	✓	✓		
Karim et al. (2014), Tahamtan et al. (2012), Modica et al. (2012), Modica and Tomarchio (2014), Weinhardt et al. (2009)			✓	✓
Junker et al. (2012), Charfi et al. (2010), Cai et al. (2009)			✓	
Oberle et al. (2013), Hoberg et al. (2012), Gudenkauf et al. (2013), Hoefer and Karagiannis (2010), Sun et al. (2011), Ye et al. (2012), El Houssaini et al. (2015)	✓	✓	✓	
Nagireddi and Mishra (2013), Sun et al. (2015), Dastjerdi et al. (2010), Shetty et al. (2015), Joshi et al. (2014), Zhang et al. (2014), Fowley et al. (2014), Ali et al. (2014), Fang et al. (2016), Moscato et al. (2011), Rekik et al. (2015)	✓	✓	✓	✓

4.5.3. Technical, operational, business, and semantic aspects

Technical and *operational* aspects describe the interface and functionalities of services. Whereas, *business* aspect describes the actors interacting with a service, pricing, SLA, legal obligations, etc. Regarding *semantic* aspect, it is distinguished by the use of ontology (OWL, WSMO, etc).

As mentioned in Table 4, most of the works have described the services from some aspects. Thus, we find works that address the operational and the technical aspects (Nguyen et al., 2012, 2011; Lee et al., 2011; Hamdaqa et al., 2011; Galán et al., 2009; Liu and Zic, 2011; Kantola and Karwowski, 2012; Mastelic et al., 2014; Quinton et al., 2014; Goscinski and Brock, 2010; Goncalves et al., 2011; García-Gómez et al., 2012; Cayirci, 2013; Ma et al., 2010; Papazoglou and van den Heuvel, 2011; Papazoglou and Vaquero, 2012; Youseff et al., 2008), works that support the business and the semantic aspects (Karim et al., 2014; Tahamtan et al., 2012; Modica et al., 2012; Modica and Tomarchio, 2014; Weinhardt et al., 2009), works for only the business aspect (Junker et al., 2012; Charfi et al., 2010; Cai et al., 2009), works that cover the technical, operational, and business aspects (Oberle et al., 2013; Hoberg et al., 2012; Gudenkauf et al., 2013; Hoefer and Karagiannis, 2010; Sun et al., 2011; Ye et al., 2012; El Houssaini et al., 2015), works that describe the technical, operational, and semantic aspects (, 2015; Zhang et al., 2012; Amato and Moscato, 2014; Miˆndruta and Fortis, 2013; Zhou et al., 2011; Afify et al., 2013; Alfazi et al., 2015; Metwally et al., 2015; Kang and Sim, 2011; Han and Sim, 2010; , 2015; Deng et al., 2011; Kaushik and Chana, 2012). However, there are other works that cover the technical, operational, business, and semantic aspects (Nagireddi and Mishra, 2013; Sun et al., 2015; Dastjerdi et al., 2010; Shetty et al., 2015; Joshi et al., 2014; Zhang et al., 2014; Fowley et al., 2014; Ali et al., 2014; Fang et al., 2016; Moscato et al., 2011; Rekik et al., 2015).

As concluded, the approaches that have not supported all aspects are not useful to describe cloud services, because of the lack of its expressiveness.

In the following, we will focus on the studies that cover all the four aspects (Nagireddi and Mishra, 2013; Sun et al., 2015; Dastjerdi et al., 2010; Shetty et al., 2015; Joshi et al., 2014; Zhang et al., 2014; Fowley et al., 2014; Ali et al., 2014; Fang et al., 2016; Moscato et al., 2011; Rekik et al., 2015), in order to compare them (see Table 5) and to identify their limitations.

- Authors in Dastjerdi et al. (2010) proposed an approach considering only IaaS services, which means that it is not generic and it cannot support SaaS and PaaS delivery models. Thus, this approach is not able to describe all cloud services, due to the specificities of each cloud service model.
- As shown in Table 5, research works (Nagireddi and Mishra, 2013; Sun et al., 2015; Dastjerdi et al., 2010; Shetty et al., 2015; Joshi et al., 2014; Zhang et al., 2014; Fowley et al., 2014; Ali et al., 2014; Fang et al., 2016; Moscato et al., 2011; Rekik et al., 2015) are compared according to nine criteria: technical aspect, functional aspect, actors, SLA, pricing, legal, security, user request, and context changes. We note that there is no approach that supports all these criteria.
- Some works (Nagireddi and Mishra, 2013; Dastjerdi et al., 2010; Shetty et al., 2015; Joshi et al., 2014; Zhang et al., 2014; Fowley et al., 2014; Ali et al., 2014; Fang et al., 2016; Moscato et al., 2011; Rekik et al., 2015) provide a few number of concepts of cloud computing. They are unable to support all cloud computing criteria (all cloud service types, all actors and their interactions, SLA, legal obligations, and pricing, etc.).
- Most of the works (Nagireddi and Mishra, 2013; Shetty et al., 2015; Joshi et al., 2014; Zhang et al., 2014; Fowley et al., 2014; Ali et al., 2014; Fang et al., 2016; Moscato et al., 2011; Rekik et al., 2015) have treated the security, but in a simplistic manner. Only a few concepts have been identified.
- The work in Sun et al. (2015) describes cloud services in a detailed manner, thanks to the expressiveness of USDL (actors, pricing, SLA, legal, etc.). This study does not treat some problems like user request, context adaptation, and security.

So, we need a generic CSD that supports technical, operational, business (actors, SLA, legal, pricing, etc.), and semantic aspects. Also, this generic service description should well describe: (i) the user request to ensure a better result of services that meet user needs; and, (ii) the context changes to establish a self-adaptation to context changes.

4.5.4. Discussion

We can conclude, from Table 3, Figs. 3 and 4, the following remarks:

Table 5
Comparison of the generic service description approaches.

Research work	Functional aspect	Technical aspect	Actors	SLA	Pricing	Legal	Security	User request	Context changes
Nagireddi and Mishra (2013)	✓	✓		✓	✓	✓	✓		
Sun et al. (2015)	✓	✓	✓	✓	✓	✓			
Dastjerdi et al. (2010)	✓	✓		✓	✓				
Shetty et al. (2015)	✓	✓			✓		✓		
Joshi et al. (2014)	✓	✓	✓	✓	✓		✓	✓	
Zhang et al. (2014)	✓	✓		✓			✓		
Fowley et al. (2014)	✓	✓		✓			✓		
Ali et al. (2014)	✓	✓	✓	✓	✓	✓	✓		
Fang et al. (2016)	✓	✓		✓	✓		✓		
Moscato et al. (2011)	✓	✓	✓	✓			✓		
Rekik et al. (2015)	✓	✓			✓		✓		

- Current research works move towards a standardized CSD (see Table 3 and Fig. 3).
- Current research works move towards a semantic CSD (a massive use of the ontology)(see Table 3 and Fig. 4).

As research perspectives, we propose:

- To take into account the context-aware in CSD (Table 5 shows clearly that this aspect is not considered).
- To define a CSD that covers technical, operational, business, and semantic aspects (see Table 4).
- To use USDL as a suitable language that describes services from technical, operational, and business perspectives.
- To define a user-centric CSD (see Table 5).

5. Requirements for cloud service description

USDL is a platform-independent language. It has been designed, in the beginning, to describe Internet services. Therefore, it is unable to describe cloud services directly, due to the specific characteristics of cloud computing environment.

In work Sun et al. (2015), authors defined only six requirements a CSD should satisfy: (i) supporting multiple perspectives (technical, operational, and business); (ii) capturing different service deployment models; (iii) being a more granular specification model to enable user-centric; (iv) representing different actors and their relationships and interactions with the services; (v) supporting a measurement mechanism for service attributes; and, (vi) supporting the semantic aspect.

In our proposal, we add four new requirements: (i) supporting context changes; (ii) well defining environment constraints; (iii) well defining user request; and, (iv) well covering security concerns (see Table 6).

Table 6
New requirements for cloud service description.

Number	New requirements
R1	Need for a specification model to manage context changes (increasing number of consumers, adding a new service, unavailability of service).
R2	Need for a specification model to describe environment constraints.
R3	Need for a specification model to manage user requests.
R4	Necessity to well define security concerns.

These requirements are detailed in the following.

R1: Supporting context changes Services exist in a dynamic environment, where external factors can be triggered in an unforeseen manner (El Houssaini et al., 2015; Mezni et al., 2014; Mezni and Sellami, 2015; Ettazi et al., 2015; Wenzhi et al., 2014; Fan et al., 2015; Mostefaoui and Younas, 2007; Gupta and Agarwal, 2015; Moltchanov, 2011). The environment changes for several reasons, such as: new better service partners are detected, a service failure, a low QoS, a change in user needs, or a loss of services due to network problems, etc.

USDL cannot describe the dynamic and unforeseen changes in the environment. Besides, none of works that are based on USDL addressed this issue (see Table 5). Therefore, we aim to propose a context-aware CSD which allows to obtain self-adaptive cloud services.

Therefore, it is necessary to know how CSD can handle the contextual information as there are so many contextual factors impacting cloud services. To do this, CSD must contain all context factors that can trigger events. It should, also, contain multiple adaptation plans to resolve the occurred changes.

There are several attempts in the literature which proposed a contextual service description such as (El Houssaini et al., 2015; Mezni et al., 2014; Mezni and Sellami, 2015; Mostefaoui and Younas, 2007; Moltchanov, 2011): El Houssaini et al. (2015) proposed a context-aware service template, which includes the functional, non-functional, and migration requirements. Mezni et al. (2014) proposed an AWS-Policy which is an extension for the description of autonomic Web services. Also, Mezni and Sellami (2015) proposed a common self-management ontology used to describe self-adaptive Web services. This ontology includes much contextual information (event, QoS, context, adaptation). Researchers, in Mostefaoui and Younas (2007), presented how to take into account the context information in the service description based on CWSDL (*Context Based Web services Description Language*). Also, Moltchanov (2011) proposed a context-aware service called CaaS (*Context as a Service*) where the context information and data are embedded within service.

The advantage of integrating context information in CSD is to propose a detailed and rich context-aware CSD. This latter includes multiple aspects (technical, operational, business, semantic, and contextual). As a consequence, it becomes generic, easy to use and to be composed with other services descriptions, and thus, it will ensure a high interoperability between the services of different providers.

R2: Supporting environment constraints A cloud service runs in a

dynamic environment, in which some parameters can affect the performance or the quality of a service. These parameters are the environmental constraints (*temporal* or *spatial* constraints) (El Houssaini et al., 2015; Mezni and Sellami, 2015; Fan et al., 2015). They are considered as a set of conditions, which influence the service execution and must be met in order to use a specific service.

Spatial constraints represent restrictions over the service execution in specific locations. A user, who is in a particular region, should invoke a service specific to the specified location. For example, a developer requests services located at two different locations: "France, Germany, or United Kingdom" or "Singapore, Japan".

Temporal constraints represent time-related restrictions (e.g., a period of unavailability, a time of receiving responses to requests). For example, users want to receive responses to their requests for a limited period, without decreasing the service quality.

R3: Well defining the user request Customers must express their requests in terms of their preferences and requirements (Sivaraman et al., 2012, Espinoza and Mena, 2007; Algergawy et al., 2010; Amorim et al., 2011; Dastjerdi et al., 2010; Jung and Sim, 2011). After sending the request, it is matched to existing service descriptions to identify which suitable service is. To allow this, it is necessary to define them in a flexible way.

USDL does not treat the end-user request. None of the aforementioned USDL-based works addressed this problem. It is necessary that CSD describes the user request, in order to facilitate service discovery.

R4: Well covering security concerns Security is considered as a non-functional aspect (QoS). In some cases, a service may be chosen according to its security aspect. In cloud computing, the security is defined in the SLA contract and particularly in the Service Level module of USDL.

There are several research works that have addressed security problems in cloud computing. For example, Barros and Oberle (2012) have treated: (i) security goals (integrity, confidentiality, identification, authentication, authorization, privacy, and accountability); (ii) security requirements (the required level of security with respect to the security goal); and, (iii) security metrics (which define what the service provider requires in terms of technical artifacts). Al Almorsy et al. (2016) have analyzed cloud security. They illustrated that cloud security should cover: (i) IaaS issues (VM security, VM images repository security, virtual network security, VM boundaries security, hypervisor security); (ii) PaaS issues (API security, SOA related security issues); (iii) SaaS issues (Web application vulnerability scanning, Web application security miss-configuration and breaking); (iv) cloud management security issues; and, (v) cloud access method security issues. Nie and Suo (2012) have illustrated the issues of cloud computing security challenges (privacy and data protection, system reliability, data isolation, security management, operators' ability). Besides, they have proposed the corresponding solutions according to the security challenges (Data encryption, access control, etc.). Based on the existing research works, security in cloud computing is considered as a complex task, because it includes various characteristics. For that, it is recommended to describe cloud security in a detailed manner.

Furthermore, the description of security characteristics should be supported on technical and non-technical (abstract) levels. The abstract description of cloud service security characteristics (Barros and Oberle, 2012) enables non-technicians, on the consumer side, to express their security demands and to find cloud services that comply with these demands. In other words, on a business level, users usually do not want to specify properties such as *Confidentiality* or *Authorization*. Instead, they want to specify more abstract properties such as "*comply to the following regulations*" or "*it is only allowed to share this data between the following parties*". For this purpose, we propose that CSD should describe service security on technical and non-technical levels.

The definition of security aspect is a complex task. It should include several characteristics and it should be defined on technical and non-technical levels. For these reasons, we propose to describe cloud service security in a detailed manner in a separate module, but, this security module remains related to SLA module.

6. Conclusion

In this paper, we have addressed the service description in cloud computing. Therefore, we have presented, classified, and compared the different approaches that have dealt with this problem.

Based on a comparative study, we conclude that USDL is the most appropriate language that supports service description from three aspects (technical, operational, and business). In this paper, we have identified new requirements (such as context changes, user request, environmental constraints, and security concerns) that should be taken into consideration. A generic CSD should (i) be able to describe all cloud services (SaaS, PaaS, and IaaS); (ii) cover technical, operational, business, and semantic aspects; (iii) well define security in a detailed manner; (iv) be context-aware; and, (v) well specify the user request.

In our future work, we intend to propose a generic CSD based on USDL, by taking into consideration the new requirements defined in this study. Then, we plan to use the proposed generic CSD to discover and to compose cloud services according to user needs.

References

Afify, Y.M., Moawad, I.F., Badr, N.L., 2013. A Semantic-based Software-as-a-Service (SaaS) Discovery and Selection System. In: Proceedings of the 8th International Conference on Computer Engineering & Systems (ICCES), November 26–28, Cairo, IEEE, pp. 57–63.

Alfazi, A., Sheng, Q.Z., Qin, Y., 2015. Ontology-Based Automatic Cloud Service Categorization for Enhancing Cloud Service Discovery. In: 2015 IEEE Proceedings of the 19th International on Enterprise Distributed Object Computing Conference (EDOC), September 21-25, Adelaide, Australie, IEEE, pp. 151–158.

Algergawy, A., Nayak, R., Siegmund, N., 2010. Combining Schema and Level-Based Matching for Web Service Discovery. In: Proceedings of the 10th International Conference on Web Engineering, Vienna, Austria, Springer, pp. 114–128.

Ali, A., Shamsuddin, S.M., Eassa, F. E., 2014. Ontology-based Cloud Services Representation. In: Research Journal of Applied Sciences, Engineering and Technology, July 5, 2014, Maxwell Science Publishing, vol. 8, no. 1, pp. 83–94.

Almorsy, M., Grundy, J., Müller, I., 2016. An analysis of the cloud computing security problem. In: arXiv:1609.01107, September 5.

Amato, F., Moscato, F., 2014. A Modeling Profile for Availability Analysis of Composite Cloud Services. In: Proceedings of the 9th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), November 8–10, IEEE, pp. 406–413.

Amorim, R., Claro, D. B., Lopes, D., 2011. Improving Web service discovery by a functional and structural approach. In: 2011 IEEE International Conference on Web Services Web Services (ICWS), July 4–9, Washington, DC, USA, pp. 411–418.

Androcec, D., Vrcek, N., Seva, J., 2012. Cloud Computing ontologies: A systematic review. In: Proceedings of the 3rd International Conference on Models and ontology-based design of protocols, architectures and services, April, pp. 9–14.

Apache Deltacloud. 2017. Apache Delta Cloud. Available at ⟨http://deltacloud.apache.org/⟩ (Accessed 13.02.17).

Apache LibCloud. 2017. One Interface To Rule Them All. Available at ⟨http://libCloud.apache.org/⟩ (Accessed 13.02.17).

Barros, A., Oberle, D., 2012. Handbook of Service Description: USDL and Its Methods. Springer, New York.

Cai, H., Zhang, K., Wang, M., 2009. Customer centric Cloud service model and a case study on commerce as a service. In: 2009 IEEE International Conference on Cloud Computing, September 21-25, IEEE, pp. 57–64.

Cardoso, J., Barros, A., May, N., 2010. Towards a unified service description language for the internet of services: Requirements and first developments. In: 2010 IEEE International Conference on Services Computing (SCC), July 5-10, Miami, USA, IEEE, pp. 602–609.

Cardoso, J., Winkler, M., Voigt, K., 2009. A service description language for the internet of services. In: Proceedings of the First International Symposium on Services Science (ISSS'09), Logos, Berlin, Germany.

Carlini, E., Coppola, M., Dazzi, P., 2011. Cloud federations in contrail. In: Euro-Par 2011: Parallel Processing Workshops, 2011, Berlin Heidelberg, Springer, vol. 7155, pp. 159–168.

Cayirci, E., 2013. Modeling and simulation as a Cloud service: a survey. In: 2013 Winter Simulation Conference, December 8, Washington, IEEE, pp. 389–400.

Charfi, A., Schmeling, B., Novelli, F., 2010. An overview of the unified service description language. In: 2010 IEEE Proceedings of the 8th European Conference on Web Services (ECOWS), December 1–3, Ayia Napa, IEEE, pp. 173–180.

Dastjerdi, A. V., Tabatabaei, S. G. H., Buyya, R., 2010. An Effective Architecture for Automated Appliance Management System Applying Ontology-Based Cloud Discovery. In: Proceedings of the 10th IEEE/ACM International Conference on

Cluster, Cloud and Grid Computing, May 17–20, Melbourne, Australia, IEEE, pp. 104–112.
Dastjerdi, A.V., Tabatabaei, S.G.H., Buyya, R., 2010. An effective architecture for automated appliance management system applying ontology-based Cloud discovery. In: Proceedings of the 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, May 17, 2010, Washington, USA, IEEE, pp. 104–112.
Davis, D., Pilz, G., 2012. Cloud Infrastruct. Manag. Interface (CIMI) Model REST Interface HTTP DSP0263 (May).
Deng, Y., Head, M.R., Kochut, A., 2011. Introducing semantics to Cloud services catalogs. In: 2011 IEEE International Conference on Services Computing (SCC), July 4–9, IEEE, pp. 24–31.
DMTF. 2017. Open Virtualization Format (OVF).Available at ⟨http://www.dmtf.org/standards/ovf⟩ (Accessed 13.01.17).
Ecore Tools. 2017. EcoreTools-User guide. Available at ⟨http://www.eclipse.org/ecoretools/doc/index.html⟩(Accessed 13.01.17).
El Houssaini, C., Nassar, M., Kriouile, A., 2015. A Cloud service template for enabling accurate Cloud adoption and migration. In: International Conference on Cloud Technologies and Applications (CloudTech), June 2–4, IEEE, pp. 1–6.
Espinoza, M., Mena, E., 2007. Discovering Web Services Using Semantic Keywords. In: 2007 Proceedings of the 5th IEEE International Conference on Industrial Informatics, June 23–27, Vienna, Austria, IEEE, , vol. 2, pp. 725–730.
Ettazi, W., Hafiddi, H., Nassar, M., 2015. A cloud-based architecture for transactional services adaptation. In: 2015 International Conference on Cloud Technologies and Applications (CloudTech), June 2–4, IEEE, pp. 1–6.
Fan, H., Hussain, F.K., Younas, M., 2015. An integrated personalization framework for SaaS-based cloud services. Future Gener. Comput. Syst. 53, 157–173.
Fang, D., Liu, X., Romdhani, I., 2016. An agility-oriented and fuzziness-embedded semantic model for collaborative Cloud service search, retrieval and recommendation. In: Future Generation Computer Systems, March 31, Elsevier, vol. 56, pp. 11–26.
Fowley, F., Pahl, C., Zhang, L., 2014. Cloud Service Brokerage: A Conceptual Ontology-Based Service Description Framework. In: Handbook of Research on Architectural Trends in Service-Driven Computing, September 26, IGI, pp. 613–619.
Galán, F., Sampaio, A., Rodero-Merino, L., 2009. Service specification in Cloud environments based on extensions to open standards. In: Proceedings of the 4th International ICST Conference on COMmunication System softWAre and middlewaRE (COMSWARE'09), New York, USA, ACM, no. 19, pp. 1–12.
García-Gómez, S., 2012. Escriche-Vicente, M., Jiménez-Gañán, M., 2012. 4CaaSt: Comprehensive management of Cloud services through a PaaS. In: Proceedings of the 10th IEEE International Symposium on Parallel and Distributed Processing with Applications, July 10–13, Leganés, Madrid, Spain, IEEE, pp. 494–499.
Gonçalves, G., Endo, P., Santos, M., 2011. CloudML: An Integrated Language for Resource, Service and Request Description for D-Clouds. In: Proceedings of the 3rd International Conference on Coud Computing Technology and Science(CloudCom) IEEE, November 29–December 1, IEEE, pp 399–406.
Goscinski, A., Brock, M., 2010. Toward dynamic and attribute based publication, discovery and selection for Cloud Computing. Future Gener. Comput. Syst. 26 (7), 947–970.
Gudenkauf, S., Josefiok, M., Göring, A., 2013. A reference architecture for Cloud service offers. In: Proceedings of the 17th IEEE International Enterprise Distributed Object Computing Conference (EDOC), September 9–13, Vancouver, BC, IEEE, pp. 227–236.
Gupta, N., Agarwal, A., 2015. Context aware mobile cloud computing: review. In: Proceedings of the 2nd International Conference on Computing for Sustainable Global Development (INDIACom), March 11–13, IEEE, pp. 1061–1065.
Hamdaqa, M., Livogiannis, T., Tahvildari, L., 2011. A reference model for developing Cloud applications. In: Proceedings of the 1st International Conference on Cloud Computing and Services Science, SciTePress, pp. 98–103.
Han, T., Sim, K.M., 2010. An ontology-enhanced Cloud service discovery system. In: International MultiConference of Engineers and Computer Scientists, March 17-19, Hong Kong, IMECS, vol. 1, pp. 17–19.
Hoberg, P., Wollersheim, J., Krcmar, H., 2012. Service Descriptions for Cloud Services - The Customers Perspective. In: ConLife Academic Conference.
Hoefer, C.N., Karagiannis, G., 2010. Taxonomy of Cloud Computing services. In: 2010 IEEE Globecom Workshops, December 6–10, IEEE, pp. 1345–1350.
jClouds. The Java Multi-Cloud Toolkit. 2017. Available at ⟨http://www.jClouds.org/⟩ (Accessed 13.03.17).
Joshi, K.P., Yesha, Y., Finin, T., 2014. Automating Cloud services life cycle through semantic technologies. In: IEEE Transactions on Services Computing, January–March, IEEE, vol. 7, no. 1, pp. 109–122.
Jung, G., Sim, K. M., 2011. Agent-based Adaptive Resource Allocation on the Cloud Computing Environment. In: Proceedings of the 40th International Conference on Parallel Processing Workshops (ICPPW), September 13–16, IEEE, pp. 345–351.
Junker, F., Vogel, J., Stanoevska, K., 2012. Aggregating price models for composite services in Cloud service marketplaces. In: 2012 IEEE Proceedings of the 10th International Symposium on Parallel and Distributed Processing with Applications, July 10–13, IEEE, pp. 479–486.
Kang, J., Sim, K. M., 2011. Ontology and search engine for Cloud Computing system. In: 2011 International Conference on System Science and Engineering, June 8–10, IEEE, pp. 276–281.
Kantola, J., Karwowski, W., 2012. Knowledge service engineering handbook, CRC Press, May 17, 2012, New York, pp. 599.
Karim, R., Ding, C., Miri, A., 2014. End-to-end QoS mapping and aggregation for selecting Cloud services. In: 2014 International Conference on Collaboration Technologies and Systems (CTS), May 19–23, Minneapolis, MN, IEEE, pp. 515–522.
Kaushik, A., Chana, I., 2012. Ontology based Cloud Framework. In: International Journal of Computer Applications (0975–8887) on Advanced Computing and Communication Technologies for HPC Applications, June.
Kitchenham, B., Charters, S., 2007. Guidelines for performing Systematic Literature Reviews in Software Engineering. Keele Univ. Durh. Univ. 2 (3), 1–65.
Kouki, Y., Ledoux, T., 2013. RightCapacity: SLA-driven Cross-Layer Cloud Elasticity Management. Int. J. -Gener. Comput. (IJNGC) 4 (3), 250–262.
Lee, B. S., Yan, S., Ma, D., 2011. Aggregating IaaS Service. In: 2011 Annual SRII Global Conference, March 29-April 2, San Jose, CA, IEEE, pp. 335–338.
Liu, D., Zic, J., 2011. Cloud#: A specification language for modeling Cloud. In: 2011 IEEE International Conference on Cloud Computing (CLOUD), July 4–9, Washington, DC, IEEE, pp. 533–540.
Ma, H., Schewe, K. D., Xie, H., 2010. Using XML for Cloud specification and XQuery for service discovery. In: Proceedings of the 12th International Conference on Information Integration and Web-based Applications & Services, November 8, New York, NY, USA, ACM, pp. 126–133.
Martin, D., 2004. Paolucci, M., McIlraith, S., Bringing Semantics to Web Services: The OWL-S approach. In: International Workshop on Semantic Web Services and Web Process Composition, July 6, Berlin Heidelberg, Springer, pp. 26–42.
Martino, B.D., Cretella, G., Esposito, A., 2014. Towards a Unified OWL Ontology of Cloud Vendors' Appliances and Services at PaaS and SaaS Level. In: Proceedings of the 8th International Conference on Complex, Intelligent and Software Intensive Systems (CISIS), July 2–4, Birmingham, England, IEEE, pp. 570–575.
Martino, B.D., Cretella, G., Esposito, A., 2015. Defining Cloud Services Workflow: A Comparison between TOSCA and OpenStack Hot. In: Proceedings of the 9th International Conference on Complex, Intelligent, and Software Intensive Systems (CISIS), July 8–10, IEEE, pp. 541–546.
Martino, B.D., Cretella,G., Esposito, A., 2014. Semantic representation of Cloud services: a case study for Microsoft Windows Azure. In: 2014 International Conference on Intelligent Networking and Collaborative Systems (INCoS), September 10–12, IEEE, pp. 647–652.
Mastelic, T., Brandic, I., Garcia, A. G., 2014. Towards Uniform Management of Cloud Services by applying Model-Driven Development. In: 2014 IEEE Proceedings of the 38th Annual on Computer Software and Applications Conference (COMPSAC), July 21–25, Vasteras, Sweden, IEEE, pp. 129–138.
Mell, P., Grance, T., 2011. The NIST definition of Cloud Computing. NIST SP 800-145, Information Technology Laboratory, National Institute of Standards and Technology (NIST), September.
Metwally, K. M., Jarray, A., Karmouch, A., 2015. Two-phase ontology-based resource allocation approach for IaaS Cloud service. In: 2015 Proceedings of the 12th Annual IEEE Consumer Communications and Networking Conference (CCNC), January 9–12, IEEE, pp. 790–795.
Mezni, H., Chainbi, W., Ghedira, K., 2014. Extending Policy Languages for Expressing the Self-Adaptation of Web Services. In: J. UCS, 2014, vol. 20, no 8, p. 1130–1151.
Mezni, H., Sellami, M., 2015. AWS-Ont: an Ontology for the Self-management of Service-based Systems. In: 2015 IEEE Proceedings of the 8th International Conference on Service-Oriented Computing and Applications (SOCA), October 19–21, IEEE, pp. 85–92.
Miˆndruta, C., Fortis, T. F., 2013. A Semantic Registry for Cloud Services. In: Proceedings of the 27th International Conference on Advanced Information Networking and Applications Workshops (WAINA), March 25–28, Barcelona, IEEE, pp. 1247–1252.
Modica, G.D., Petralia, G., Tomarchio, O., 2012. A business ontology to enable semantic matchmaking in open Cloud markets. In: Proceedings of the 8th International Conference on Semantics, Knowledge and Grids (SKG), October 22–24, Beijing, China, IEEE, pp. 96–103.
Modica, G.D., Tomarchio, O., 2014. Matching the business perspectives of providers and customers in future Cloud markets. In: Cluster Computing, March 11, New York, Springer, vol. 18, no. 1, pp 457–475.
Moltchanov, B., 2011. Context representation formalism and its integration into context as a service in clouds. In: ITU on Kaleidoscope 2011: The Fully Networked Human? - Innovations for Future Networks and Services (K-2011), December 12–14, IEEE, pp. 1–8.
Monteiro, A., Pinto, J.S., Teixeira, C., 2011. Cloud Interchangeability: redefining expectations. CLOSER, 180–183.
Morfeo 4caaSt. 2017. Welcome to 4CaaSt. Available at ⟨http://www.4caast.eu/⟨. (Accessed 13.01.17).
Moscato, F., Aversa, R., Martino, B.D., 2011. An analysis of mosaic ontology for Cloud resources annotation. In: 2011 Federated Conference on Computer Science and Information Systems (FedCSIS), September 18–21, Szczecin, Poland, IEEE, pp. 973–980.
Mostefaoui, S.K., Younas, M., 2007. Context-oriented and transactionbased service provisioning. Int. J. Web Grid Serv. (IJWGS) 3, 194–218.
Nagireddi, V.S.K., Mishra, S., 2013. An ontology based Cloud service generic search engine. In: Proceedings of the 8th International Conference on Computer Science & Education (ICCSE), April 26–28, Colombo, IEEE, pp. 335–340.
Nguyen, D.K., Lelli, F., Papazoglou, M.P., 2012. Blueprinting approach in support of cloud cComputing. In Future Internet 2012, March 21, 2012. Mol. Divers. Preserv. Int. 4 (1), 322–346.
Nguyen, D.K., Lelli, F., Taher, Y., 2011. Blueprint template support for engineering Cloud-based services. In: Proceedings of the 4th European Conference Towards a Service-Based Internet (ServiceWave 2011), October 26–28, Poznan, Poland, Springer, pp. 26–37.
Nie, X., Suo, H., 2012. Security in the cloud computing: A review. In: Proceedings of the 2nd International Conference on Computer Science and Network Technology (ICCSNT), December 29-31, IEEE, pp. 2145–2149.
Oberle, D., Barros, A., Kylau, U., 2013. A unified description language for human to

automated services. In: Information systems 38, March, Elsevier, vol. 38, no. 1, pp. 155–181.
Papazoglou, M.P., van den Heuvel, W.J., 2011. Blueprinting the Cloud. In: IEEE Internet Computing, November-December, vol. 15, no. 6, IEEE, pp. 74–79.
Papazoglou, M.P., Vaquero, L. M., 2012. Knowledge-intensive Cloud services: transforming the Cloud delivery stack. In: Knowledge Service Engineering Handbook, CRC Press, pp. 449–494.
Quinton, C., Romero, D., Duchien, L., 2014. Automated selection and configuration of Cloud environments using software product lines principles. In: Proceedings of the 7th International Conference on Cloud Computing 2014 IEEE. June 27–July 2, Anchorage, USA, IEEE, pp. 144–151.
Rekik, M., Boukadi, K., Abdallah, H.B., 2015. Cloud description ontology for service discovery and selection. In: Proceedings of the 10th International Joint Conference on Software Technologies (ICSOFT). July 20–22, IEEE, vol. 1, pp. 1–11.
Rochwerger, B., Breitgand, D., Levy, E., 2009. The reservoir model and architecture for open federated Cloud Computing. IBM J. Res. Dev. 53 (4), 1–4.
Roman, D., De Bruijn, J., Mocan, A., 2006. WWW: WSMO, WSML, and WSMX in a nutshell. In: Asian Semantic Web Conference (ASWC), September 3, Berlin Heidelberg, Springer, pp. 516–522.
Shetty, J., D'Mello, D. A., 2015. An XML based data representation model to discover infrastructure services. In: 2015 International Conference on Smart Technologies and Management for Computing, Communication, Controls, Energy and Materials (ICSTM), May 6–8, Chennai, India, IEEE, pp. 119–125.
Simple cloud. Simple cloud. 2017. Available at ⟨https://www.simple-cloud.fr/⟩Accessed on February, 13.
Sivaraman, S., Pabitha, P., Rajaram, M., 2012. An ontology model for mapping the user request to select web services. In: International Conference on Recent Trends In Information Technology (ICRTIT), April 19–21, IEEE, pp. 492–496.
Sun, L., Dong, H., Ashraf, J., 2012. Survey of service description languages and their issues in Cloud Computing. In: Proceedings of the 8th International Conference on Semantics, Knowledge and Grids (SKG), October 22–24, Beijing, China, IEEE, pp. 128–135.
Sun, L., Ma, J., Wang, H., 2015. Cloud Service Description Model: An Extension of USDL for Cloud Services. In: IEEE Transactions on Services Computing, August 28, 2015, IEEE.
Sun, Y. L., Harmer, T., Stewart, A., 2011. Mapping application requirements to Cloud resources. In: European Conference on Parallel Processing, August 29, Berlin Heidelberg, Springer, pp. 104–112.
Tahamtan, A., Beheshti, S.A., Anjomshoaa, A., 2012. A Cloud repository and discovery framework based on a unified business and Cloud service ontology. In: Proceedings of the 8th World Congress on Services IEEE. June 24–29, Honolulu, Hawaí, IEEE, pp. 203–210.
Terzidis, O., Oberle, D., Friesen, A., 2012. The Internet of services and usdl. In: Handbook of Service Description, February 11, New York, Springer, pp. 1–16.
Unified Modeling Language (UML). 2017. Welcome to UML Web Site. Available at ⟨http://www.uml.org/⟩ (Accessed 13.01.16).
W3C. 2016. Web Service Semantics-WSDL-S. Available at ⟨http://www.w3.org/Submission/WSDL-S/⟩. (Accessed 22.07.16).
W3C. 2017. OWL Web Ontology Language: Overview. Available at ⟨http://www.w3.org/TR/owl-features/⟩. (Accessed 13.01.17).
W3C. 2017. Web services description language (WSDL) 1.1. Available at ⟨https://www.w3.org/TR/wsdl⟩. (Accessed 13.01.17).
Weinhardt, C., Anandasivam, A., Blau, B., 2009. Business models in the service world. In: IT Professional Magazine, March/April 2009, IEEE, vol. 11, no. 2, p. 28–33.
Wenzhi, L., Xiusi, H., Zhaobin, L., 2014. The context-aware and collaboration of education Cloud resources. In: 2014 Proceedings of the 9th International Conference on Computer Science & Education (ICCSE), August 22-24, IEEE, pp. 189–192.
Ye, F., Wang, Z., Yue, Z., 2012. CloudUDDI: An extended UDDI model for Cloud services. In: Proceedings of the 2nd International Conference on Cloud Computing and Intelligent Systems 2012 IEEE , October 30–November 1, Hangzhou, China, IEEE, vol. 1, pp. 499–503.
Youseff, L., Butrico, M., Da Silva, D., 2008. Toward a unified ontology of Cloud Computing. In: 2008 Grid Computing Environments Workshop, November 12–16, 2008, Austin, Texas, USA, IEEE, pp. 1–10.
Zhang, L., Fowley, F., Pahl, C., 2014. A template description framework for services as a utility for Cloud brokerage. In: Proceedings of the 4th International Conference on Cloud Computing and Services Science, April 3–5, Barcelona, Spain, SCITEPRESS-Science and Technology Publications.
Zhang, M., Ranjan, R., Haller, A., 2012. An ontology-based system for Cloud infrastructure services' discovery. In: 2012 Proceedings of the 8th International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom), October 14–17, 2012, Pittsburgh, PA, IEEE, pp. 524–530.
Zhou, J., Abdullah, N.A., Shi, Z., 2011. A hybrid P2P approach to service discovery in the Cloud. In: International Journal of Information Technology and Computer Science (IJITCS), February 2011, MECS, vol. 3, no. 1, pp. 1–9.