Service Oriented Computing and Applications (2019) 13:109–125
https://doi.org/10.1007/s11761-019-00263-z

ORIGINAL RESEARCH PAPER

Service description languages in cloud computing: state-of-the-art and research issues

Falak Nawaz[1] · Ahmad Mohsin[2] · Naeem Khalid Janjua[2]

Received: 31 January 2019 / Revised: 30 May 2019 / Accepted: 3 June 2019 / Published online: 6 June 2019

Abstract
The continuous growth of cloud computing environment is supported by the automated provisioning of cloud services, which allows cloud users to dynamically procure and deploy their required services over the internet. However, to describe key characteristics of a cloud service, each cloud service provider uses its own service description language (SDL) utilizing its underlying syntax and semantics coupled with models and standards to fulfill its own objectives (such as deployment, provisioning, modeling, discovery and composition). This prevents cloud computing community to adopt and enforce a standard mechanism for SDLs that limits the ability of automation of cloud services and results in vendor lock-in problem for cloud users. In this paper, we investigate different techniques of cloud SDLs by focusing on their main purpose of use in different cloud service operations including deployment and provisioning, modeling and composition, discovery and selection, and service level agreement. We use a common comparison criteria to classify existing literature on cloud SDLs with the goal to identify their common features in different service operations. Based on identified gaps in the literature, research issues have been structured to develop the foundations of a standard cloud SDL in order to develop high-quality cloud applications of the future.

Keywords Cloud computing · Service description languages · SLA · Deployment and provisioning · Modeling and composition · Discovery and selection

1 Introduction

Cloud computing paradigm has enabled businesses to access on-demand resource provisioning and rapid deployment, thereby providing them with the required infrastructure for on-demand services. The real benefit of cloud computing comes with the provision of everything as a service (from hardware resources to high-level software applications). Cloud service providers classify these services in the form of three service delivery models: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). Many organizations are leveraging from these delivery models of cloud services depending upon their business needs for various types of services integration ranging from Internet of Things (IoT) to Cyber-Physical Systems (CPS), and Smart Systems to Industry 4.0.

Cloud service providers (e.g., Amazon, Microsoft etc.) offer different types of cloud services by targeting certain businesses with different pricing models. For example, Amazon alone offers more than 70 cloud services. Moreover, each cloud service provider uses its own SDL to describe its cloud services. These SDLs are also known as specification languages or modeling languages. The purpose of the SDLs vary depending upon their usage in different service operations (such as *deployment and provisioning* [15, 20, 23, 30, 55, 56, 61, 64, 70, 75, 78, 81, 85, 90, 95]), *modeling and composition* [16, 22, 26–28, 31, 35, 37, 51, 74, 83, 93, 97], *discovery and selection* [1, 4, 18, 26, 72, 82, 96, 99], *service level agreement* [6, 12, 46, 47, 54, 76, 88, 89, 92], and their scope (business, technical etc.) in different stages of cloud service life cycle. Furthermore, different

✉ Falak Nawaz
falak.nawaz@student.adfa.edu.au; falakniazi@gmail.com

Ahmad Mohsin
a.mohsin@ecu.edu.au

Naeem Khalid Janjua
n.janjua@ecu.edu.au

1 School of Business, University of New South Wales (UNSW), Canberra, Australia

2 School of Science, Edith Cowan University (ECU), Perth, Australia

service delivery models (such as IaaS, PaaS, and SaaS) have different service requirements in relation to the objective of cloud services. This compels each service provider to use its own tailor-made and customized SDL resulting in many heterogeneous SDLs. The main drawback of these heterogeneous SDLs is the lack of interoperability among cloud service providers, which causes vendor lock-in problem for the cloud users.

The current literature on cloud SDLs indicates that there exists neither a consensus among cloud service providers on a particular SDL nor any defined standard for SDL. Moreover, the focus of these SDLs is also fragmented. During the literature review, it was found that many such SDLs focus on service deployment and provisioning whereas there are very limited SDLs that cater solutions for service discovery and selection despite various research efforts in the related domains of service-oriented architecture (SOA) and Web services. In this paper, we provide a systematic literature review on cloud SDLs by classifying and comparing them using a common criteria. The comparison criteria includes domain (along with service delivery model), objective, scope, representation, and semantic support. The main purpose of this survey is to provide an objective-based analysis of the existing literature on cloud SDLs and to highlight the accomplished and the neglected areas of research in cloud SDLs. Finally, a discussion is included on the identified research issues to motivate further research in this field.

The rest of the paper is organized as follows. In Sect. 2, we discuss the related surveys on cloud SDLs. In Sect. 3, we present the research methodology and comparison criteria that is used to analyze and compare existing literature. The state-of-the-art in cloud SDLs using the objective-based classification is presented in Sect. 4. In Sect. 5, the identified research issues in cloud SDLs are discussed. Section 6 concludes the paper with a discussion on future research directions.

2 Related surveys

In this section, we present the related surveys on cloud SDLs. Although a significant amount of research exists in the literature on cloud SDLs, but there are very few recent surveys in this area. One of the first survey on cloud SDLs was conducted by Sun et al. [86]. In this work, seven different aspects of SDLs are considered that include domain, coverage, purpose, representation, intended user, semantics and features. In our survey, we cover all these aspects by analyzing service descriptions with a common criteria for comparison (that includes domain, coverage, purpose, representation and semantics), intended users and features with respect to cloud computing domain. Moreover, we further refine the domain with respect to objective (purpose) and service delivery model that has not been done in the above survey. Kritikos et al. [53] presented a survey of service quality descriptions that is independent of any particular domain covering all stages of service life cycle including advertisement, discovery, negotiation, monitoring, and adaptation. It covered different models and meta-models for service quality descriptions and SLAs. The scope of this survey is limited to quality aspects of service descriptions and SLAs; therefore, it did not discuss SDLs with respect to functional aspects and cloud computing perspective. Jula et al. [44] conducted a survey of cloud service composition approaches and categorized them into four main groups (i.e., class and graph-based, combinatorial, machine-based, structures and frameworks) based on problem-solving approaches. It also investigated these approaches with respect to quality of service, intended objective, and programming environment. It focused on the technical aspects of cloud service composition languages and approaches. Moreover, this work does not refine the approaches with respect to service delivery models in the cloud domain.

In a recent work, Ghazouani and Slimani [33] presented a systematic survey of cloud service description that considers domain, delivery model, technique, representation, and semantics as comparison criteria for the evaluation. By analyzing with our comparison criteria (domain, objective, representation, semantics, and scope) and categorizing services with delivery model in cloud computing, we cover all aspects of this work. Moreover, in contrast to this work, our work also classifies the services with respect to its objective in each domain and delivery model. The main advantage of objective-based classification is to provide holistic view of the cloud SDLs. In this way, we cover all aspects of cloud services (starting from service creation to the end of service consumption) with a goal to identify common features in different service operations. In another work, Mohsin and Janjua [63] presented state-of-the-art on modeling approaches for SOA especially targeting distributed large-scale complex systems. This work provides key features of languages and frameworks to model and reason services description, composition, and orchestration at structural and behavioral levels with a focus on QoS conformance. Authors have suggested to use formal modeling techniques to improve upon existing syntax and semantics in order to manage dynamic services composition to conform SLAs in cloud environments. In another recent work, Bergmayr et al. [11] conducted a systematic survey of cloud modeling languages that focussed on the modeling language scope, language characteristics (i.e., syntax, semantics, realization etc.), modeling capabilities, and tool support. This survey emphasized the cloud modeling languages at the IaaS and PaaS level, but does not consider languages at the SaaS level. Hence, it does not take a holistic view of cloud SDLs and neglected those aspects of description languages which also overlap with the description languages in SOA domain. Furthermore, in contrast to this work, we perform objective-based classification of cloud SDLs that groups conducted research with similar motivation and intentions together.

3 Review process

3.1 Research methodology

In this paper, the systematic literature review methodology is followed, which is recommended by [49]. According to this methodology, the literature is selected within a timeframe using well-defined search queries. The selected studies are then analyzed using a comprehensive and rigorous approach. Specifically, the key features and characteristics of the studies are taken into consideration to classify them accordingly and identify research issues that need to be addressed. For this purpose, relevant research questions (RQ) to be answered by this research have been devised [13].

- RQ1. What is the state-of-the-art of SDLs in cloud computing?
- RQ2. What are the key characteristics of existing SDLs for deployment and provisioning, modeling and composition, discovery and selection, and service level agreement (SLA)?
  - RQ2.1. What are the common characteristics of existing SDLs to support development of a generalized description language?
  - RQ 2.2. What are the key challenges of existing SDLs in cloud computing?
- RQ3. To how much extent existing vocabulary of SDLs is able to describe various aspects of service operations in cloud computing environments?

By following the guidelines of the systematic literature review methodology [49], research publications from 2008 to 2019 are selected, which include papers related to cloud SDLs from different well-known journals, conferences, and workshops. In addition, some important and relevant studies published earlier than 2008 from SOA and semantic web services domain are also included.

Since cloud computing is very diverse area and there is no standard naming convention for description languages, determining the relevant work solely on the basis of search queries is a challenging task. Therefore, inclusion and exclusion criteria to determine relevant work is supported by manual selection based on title, abstract, and content, in addition to the search queries. The key terms for search queries are also defined according to the guidelines as mentioned in [49]. We considered "description," "specification," "language," "modeling," and "profile" as key terms for search queries in cloud computing domain. These search queries are executed on different electronic scientific databases recommended by Kitchenham et al. [50], which include Google Scholar, IEEE Xplore, ScienceDirect, SpringerLink, etc. Finally, 65 studies are selected (listed in Table 2) and analyzed using the comparison criteria described in the next section.

Table 1 A summary of comparison criteria

Criterion	Brief description
Domain	It covers broader fields of research (such as cloud, SOA and Semantic web) as well as the specific areas including (IaaS, PaaS, SaaS, XaaS)
Objective	It classifies cloud SDLs into four categories with respect to their objectives. These four categories include: (1) deployment and provisioning, (2) modeling and composition, (3) discovery and selection, and (4) service level agreement
Scope	It defines the coverage of the cloud SDLs that includes business, technical, and both (i.e., business and technical)
Representation	It specifies the representation language used to describe the description language. Some of the frequently used representation languages are XML, DTD, OWL, UML etc.
Semantics support	It determines the semantic support of the description language in order to identify its support for interoperability and wide-scale applicability

3.2 Criteria for comparison

As described previously, the focus of this survey is on the cloud SDLs. An SDL usually defines the functionality and quality of service (QoS) aspects of the service from technical perspective for the expert users (e.g., application developers). However, in some cases, business perspective is also included to attract the non-technical service users (e.g., business user). In particular, SLA templates are used to describe the service functionality from the non-technical (i.e., business) perspective. In this survey, we emphasize on all approaches, specifications, languages, and tools that are used to describe the functional or QoS aspects of the service in cloud computing environment. A summary of the comparison criteria to analyze and compare the cloud SDLs is shown in Table 1. This includes domain, objective, scope, representation and semantic support. The detailed evaluation of the existing literature according to the comparison criteria is shown in Table 2.

As shown in Fig. 1, in this survey, we broadly classify the cloud SDLs into the following four categories based on the objective criterion: (1) deployment and provisioning, (2)

Table 2 A comparison of SDLs in cloud computing and related domains

Description languages	Domain	Delivery model	Semantics	Representation	Scope	Objective
Cloud# [56]	Cloud	IaaS	L	Formal specification language	T	Deployment and provisioning
Data-aware resource provisioning [64]	Cloud	IaaS	L	Aneka APIs	T	Deployment and provisioning
soCloud [75]	Cloud	PaaS	M	OASIS component architecture	B&T	Deployment and provisioning
STRATOS [77]	Cloud	PaaS	L	XML	T	Deployment and provisioning
User-centric cloud registries [82],	Cloud	XaaS	M	Domain-specific lang.	B&T	Discovery and selection
Ontology-based categorization [4]	Cloud	XaaS	H	OWL/DRF	B&T	Discovery and selection
CSOnt [10]	Cloud	XaaS	H	OWL/RDF	B&T	Discovery and selection
CSDC [38]	Cloud	XaaS	H	OWL/RDF	B&T	Discovery and selection
Cloudle [45]	Cloud	XaaS	H	OWL/RDF	B&T	Discovery and selection
Blueprint model [70]	Cloud	XaaS	L	Template	T	Deployment and provisioning
CBDI-SAE-based model [58]	Cloud	SaaS	L	UML	B&T	Modeling and composition
Manifest language [20]	Cloud	SaaS	M	MOF-based model	T	Deployment and provisioning
Application meta-model [37]	Cloud	SaaS	L	Reference model	B&T	Modeling and composition
TOSCA [55]	Cloud	XaaS	L	XML	T	Deployment and provisioning
Customer-centric model [16]	Cloud	SaaS	L	Reference model	B	Discovery and selection
Cloud-agnostic middle ware [61]	Cloud	SaaS	L	Reference model	T	Deployment and provisioning
Ontology-based resource allocation (Sun et al.) [85]	Cloud	XaaS	H	OWL/RDF	T	Deployment and provisioning
Microsoft Azure [62]	Cloud	PaaS	L	XML	B	Deployment and provisioning
CompatibleOne [96]	Cloud	PaaS	M	XML	T	Discovery and selection
Amazon Beanstalk [7]	Cloud	PaaS	L	WSDL	B	Deployment and provisioning
SOCCA [91]	Cloud	PaaS	H	OWL/RDF	T	Deployment and provisioning
RESERVIOR [78]	Cloud	IaaS	L	XML	T	Deployment and provisioning
Virtual solution model [51]	Cloud	IaaS	L	IBM modeling platform	T	Modeling and composition
Virtual environment [23]	Cloud	IaaS	L	XML	T	Modeling and composition
WSDL [21]	SOA	–	L	XML	T	Discovery and selection
O'Sullivan [71]	SOA	–	L	Attribute taxonomy	B	Discovery and selection
WADL [36]	SOA	–	L	XML	T	Discovery and selection
WSDL-S [2]	SWS	–	H	WSDL/XML	T	Modeling and composition/Discovery and selection

Table 2 continued

Description languages	Domain	Delivery model	Semantics	Representation	Scope	Objective
OWL-S [60]	SWS	–	H	OWL/RDF	T	Modeling and composition/Discovery and selection
WSMO [29]	SWS	–	H	WSML	T	Modeling and composition
SAWSDL [52]	SWS	–	H	RDF	T	Modeling and composition/Discovery and selection
SML [83]	SOA	–	L	XML	T	Modeling and composition
WS-policy [8]	SOA	–	L	XML	B	Modeling and composition
USDL [18]	SOA	–	M	MOF-based model	B	Discovery and selection
WSLA [47]	SOA	–	L	XML	B	Service level agreement
WS-Agreement [6]	SOA/Cloud	XaaS	L	XML	B	Service level agreement
SLAng [54]	SOA	–	H	MOF	B	Service level agreement
WSOL [89]	SOA	–	L	XML	B	Service level agreement
RBSLA [76]	SOA	–	M	RuleML	B	Service level agreement
QoWL [12]	SOA/Grid/Cloud	XaaS	M	XML	B	Service level agreement
GXLA [88]	SOA/Grid/Cloud	XaaS	L	XML	B	Service level agreement
SLA* [46]	SOA/Grid/Cloud	XaaS	L	XML/BNF	B	Service level agreement
SLAC [92]	Cloud	XaaS	H	EBNF	B	Service level agreement
OASIS Reference Architecture for SOA [59]	SOA	–	L	UML	B&T	Modeling and composition
SoaML [74]	SOA	–	L	UML	T	Modeling and composition
ASD Meta-model [93]	SOA	–	L	UML	T	Modeling and composition
DEVAs [23]	Cloud	IaaS	L	UML	T	Deployment and provisioning
Caglar et al. [15]	Cloud	IaaS	L	UML	T	Deployment and provisioning
CloudMIG [30]	Cloud	PaaS	–	–	B&T	Deployment and provisioning
Resource provisioning for hosted SaaS [95]	Cloud	SaaS	–	–	B&T	Deployment and provisioning
Cloud DSL [81]	Cloud	XaaS	L	XML	T	Deployment and provisioning
GEMBus [27]	Cloud	PaaS	L	XML	T	Modeling and composition
Agent-based cloud service composition [35]	Cloud	XaaS	L	JADE	–	Modeling and composition
Multivariate cloud service composition [98]	Cloud	XaaS	–	–	–	Modeling and composition
WS-negotiation [94]	SOA/Cloud	XaaS	L	XML	B&T	Modeling and composition
SWRL [41]	SOA	–	M	XML/RuleML	T	Modeling and composition
CloudLaunch [1]	Cloud	IaaS	L	JSON	T	Discovery and selection
CloudRec [99]	Cloud	XaaS	–	–	B&T	Discovery and selection
SEMREG-Pro [65]	SWS	–	H	OWL/RDF	T	Discovery and selection

Table 2 continued

Description languages	Domain	Delivery model	Semantics	Representation	Scope	Objective
PORSCE II and VLEPPO [40]	SWS	–	H	PDDL	T	Modeling and composition
Generalized SWS Composition [9]	SWS	–	H	Constraint logic programming	T	Modeling and composition
ManuHub [17]	SWS	–	H	OWL/RDF	T	Discovery and selection
WSMO-Lite and hRESTS [79]	SWS	–	H	WSMO	T	Modeling and composition
SPARQL and intelligent agents [80]	SWS	–	M	Jena and SPARQL	T	Discovery and selection
Dynamic and attribute-based discovery [34]	Cloud	SaaS	L	XML/WSDL	T	Discovery and selection

Semantics (L—low, M—medium, H—high)
Scope (B—business, T—technical, B&T—business and technical)

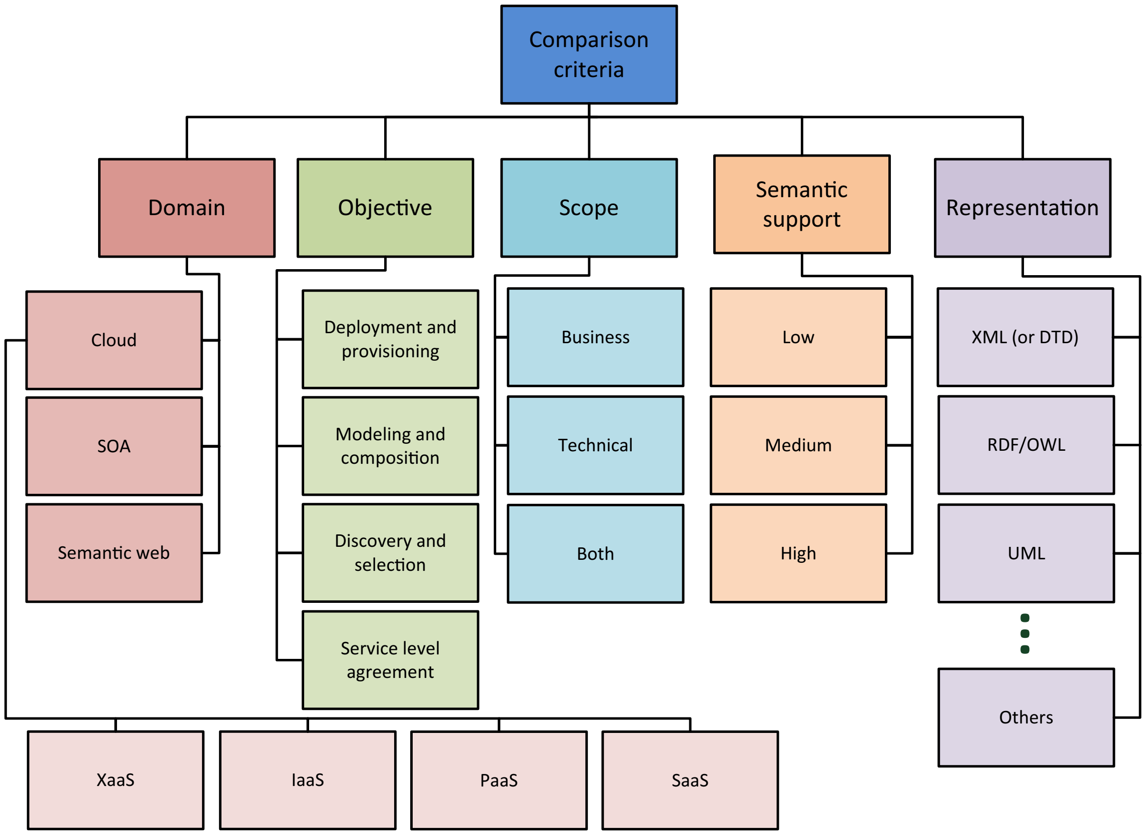

Fig. 1 The comparison criteria of cloud SDLs

modeling and composition, (3) discovery and selection, and (4) service level agreement. We further divide cloud SDLs into service delivery models (Infrastructure as a Service (IaaS), Platform as a Service (PaaS), Software as a Service (SaaS), and Everything as a Service (XaaS)) and their scope. Moreover, the related SDLs in service-oriented architecture (SOA) and semantic web services (SWS) domains are also included in this survey. The objective-based classification of the literature is conducted in the next section.

4 Objective-based classification of cloud SDLs

In this section, we present the classification of cloud SDLs based on the objective and then further divide them into domain, service delivery models, and scope, respectively. This draws interesting conclusions from the state-of-the-art of the cloud SDLs. In addition, it also demonstrates the features as well as limitations of the cloud SDLs.

Table 3 Deployment and provisioning languages

Objective	Domain	Delivery model	Scope	Description languages
Deployment and provisioning languages	Cloud	IaaS	Technical	Cloud# [56], RESERVOIR project [78], DEVAs [23], Amazon EC2 [5], Caglar et al. [15], Data-aware resource provisioning [64]
		PaaS	Business and technical	AWS Elastic Beanstalk [7], Microsoft Azure [62], STRATOS [77], SOCCA [91], CloudMIG [30], soCloud [75]
		SaaS	Business and technical	Resource provisioning for hosted SaaS [95], cloud-agnostic middleware [61], manifest language [20]
		XaaS	Technical	TOSCA [55], Blueprint Model [70], Ontology-based Resource Allocation [85], Cloud DSL [81]

4.1 Deployment and provisioning languages

The cloud SDLs that focus on service deployment and provisioning in the cloud environment are presented in this section. Service deployment in cloud computing refers to setting up a new cloud application (or service) with some defined configuration settings usually defined in a description file (i.e., configuration, manifest, and specification) while provisioning refers to getting virtual machine and installing the needed libraries on it. In this section, we combine deployment and provisioning (as shown in Table 3) into one category as many description or specification languages provide support for both of these operations.

4.1.1 IaaS description languages

Most of the existing cloud SDLs aim to support service deployment and provisioning by providing interoperability and flexibility. The interoperability issues occur due to heterogeneous cloud service descriptions used by different cloud service providers to offer their cloud services. While flexibility of specifying detailed service description is hindered by a variety of cost model offerings, variabilities in the granularity and specification of different resources are provided by the cloud service providers. In RESERVOIR project [78], the interoperability among cloud providers in a cloud federation is enhanced by developing a language to describe service offers. Adopting a standard language for service offers also solves the service discovery problem. To improve transparency and trust between cloud service users and providers, cloud# language was proposed in [56]. The main idea behind it was to model internal organization of the cloud and also to make cloud users understand how cloud services are delivered inside the cloud platform. It improves confidence and enhances trust of the cloud users on providers. However, it only supports limited characteristics of cloud's internal organization. Cloud services in fact can have complex technical as well as business specifications. This work did not discuss how it can support these specifications of cloud services.

In [23], a simplified model for configuring, deploying and provisioning of IaaS resources is presented. The model is called distributed ensembles of virtual appliances (DEVAs), which allows XML-based specification of deployment configuration for independent virtual appliances. Precisely, the model allows specifying the service endpoints for independent virtual appliances as connection between two appliances. Amazon Elastic Cloud Compute (Amazon EC2) [5] provides IaaS virtual computing instances for scalable computing capacity using Amazon Web Services (AWS) cloud. AWS cloud infrastructure allows users to configure and deploy IaaS offerings, based on their needs through easy to use web interfaces. Caglar et al. [15] presented a model-driven engineering (MDE)-based solution to tackle the issue of heterogeneity of service offerings as well to provide flexibility by estimating performance and cost to host services. It also provided a mechanism for automated deployment and resource management at IaaS level. A data-aware and deadline-driven cloud resource provisioning approach is presented in [64]. It provides dynamic resources from public cloud resources to the cloud applications that have deadline constraints and certain limit on data requirements such as bandwidth constraints. This work is implemented in Aneka cloud platform. However, due to lack of a standard language to describe service configuration, automated service deployment and provisioning in cloud federation cannot be achieved. As a consequence, cloud users suffer from vendor lock-in problem.

4.1.2 PaaS description languages

At PaaS level, AWS Elastic Beanstalk [7] is a set of APIs developed using WSDL-based web services, which provides an easy way to create, configure, deploy, and manage Amazon Elastic Cloud Compute (Amazon EC2) IaaS virtual machines. Similar to AWS Elastic Beanstalk, Microsoft Azure [62] also provides an interface for its customers to create, configure, deploy, and manage virtual machines. In particular, it allows its customers to configure virtual machines to scale up or down by adjusting the CPU, RAM, storage, and network bandwidth according to the requirements. In CloudMIG [30], a model-based approach is presented which aims at migrating deployment configurations from a source environment to deploy them at target environment. It supports service descriptions for both IaaS and PaaS level. It captures technical constraints of the platform and provides a semi-automatic migration of existing enterprise software systems at the PaaS level. Therefore, it offers flexibility and portability by describing higher level of specification of cloud services. In [77], a brokering service called STRATOS is proposed to enable composing and allocating cloud resources from multiple cloud providers. The proposed mechanism performs decision making to select and allocate the required resources based on the requirements described in the SLA. In [75], an soCloud platform for managing portability, elasticity, and provisioning of resources across multiple clouds is presented. This platform is built on top of component-based SOA using the OASIS Component Architecture standard. However, it has limited capability to manage large number of resources.

In short, the existing work in this category emphases on model-driven and tool-based approaches to provide portability among cloud providers. Nevertheless, a generic approach is needed for service deployment and provisioning that supports flexibility as well as portability.

4.1.3 SaaS description languages

In [95], an SLA-based resource provisioning approach for hosted SaaS applications is presented. This approach minimizes the resource and penalty cost for the users through handling their dynamic request by taking into account the customer profiles and providers' quality parameters. In [61], a cloud-agnostic middleware is presented. The middleware is described as meta-model that sits at the SaaS level and makes it independent of the lower levels (i.e., PaaS and IaaS). The key components of the middleware meta-model are users, cloud providers, cloud resources, and deployment information. In [20], architectural constraints for on-demand cloud provisioning such as co-location and service elasticity are defined. It also defines relationship by establishing constraints between architecture definitions and cloud management infrastructure. The goal of this approach is to meet QoS for users and optimizing resources of the service provider. This approach introduced a manifest language which is modeled using EMOF (Essential Meta-Object Facility), while constraints are defined using OCL (Object Constraint Language). The focus of the above-mentioned resource provisioning approaches is to describe efficient elasticity rule. As a result, cloud resources can be scaled up or down at application runtime.

4.1.4 XaaS description languages

There are considerable number of cloud SDLs which focus on service deployment and provisioning at all levels of cloud services. For example, TOSCA (Topology and Orchestration Specification for Cloud Applications) [55] is an OASIS standard for describing structure and invocation of management behavior of composite cloud applications. It provides a template for specifying nodes and their relationship in application topology as well as provides the deployment topology and service management plan. Another similar work is Blueprint model and language for engineering cloud applications [70]. This work also takes a holistic view of SDLs for cloud service deployment. It differs from TOSCA with the ability to provide support for service composition at different layers of cloud stack. Both of these descriptions capture the technical aspects of cloud applications. However, less emphasis is made on the business requirements of cloud applications. Moreover, these approaches use application-specific description languages to describe cloud platform entities. In [81], a language for supporting portability by describing cloud platform entities is proposed. It provides wide coverage of cloud services and also provides integration of cloud DSL and TOSCA. In [85], authors provide a resource allocation-based ontology mapping technique. They used ontology to specify the resource requirements of a high-level SaaS application with the low-level available resources of the cloud infrastructure.

4.2 Modeling and composition languages

In this section, we present cloud SDLs that describe features for service modeling and composition. Moreover, these description languages are further classified based on the service delivery model in the cloud environment as detailed in Table 4.

4.2.1 IaaS description languages

At IaaS level in cloud computing, a cloud service composition framework is presented in [26] for non-technical users which can specify vague and ambiguous requirements. The framework maps user requirements to select suitable cloud service composition by analyzing the cloud service compatibility. For this purpose, ontology is developed to represent the knowledge of available cloud service descriptions. Moreover, a combination of evolutionary algorithms and fuzzy logic is used to express user preferences in linguistics terms which

Table 4 Modeling and composition languages

Objective	Domain	Delivery model	Scope	Description languages
Modeling and composition languages	Cloud	IaaS	Technical	Virtual environment [23], Compatibility-aware cloud [26], Virtual Solution Model [51]
		PaaS	Technical	GEMBus [27]
		SaaS	Business and technical	Application meta-model [37], CBDI-SAE meta-model [58], ASD Meta-model [93]
		XaaS	–	Agent-based cloud service composition [35], Multivariate cloud service composition [98]
	SOA	–	Technical	SoaML [74], SML [83]
			Business and technical	WS-Agreement [6], WS-Negotiation [94], WS-policy [8], OASIS Reference model [59, 28]
	SWS	–	Technical	WSMO [29], WSDL-S [2], OWL-S [60], SAWSDL [52], SWRL [41], WSMO-Lite and hRESTS [79], Generalized SWS composition [9], PORSCE II, and VLEPPO [40]

reduces user effort. The virtual environment presented in [22] provides the basis for distributed ensembles of virtual appliances for IaaS clouds. It allows easy to compose and ready to use cloud application architecture that hides all the unnecessary details of specifying IaaS instances for web application developers. A virtual solution model for composition and deployment of virtual services in cloud environment is presented in [51]. In this model, virtual appliances or services consist of virtual images which are treated as basic building block for composite solutions. To create a composite cloud service, a virtual solution model is created first in which requirements for virtual appliances are defined independent of the cloud environment. Once the requirements for virtual appliances are negotiated in the composition, the virtual solution model is converted to a cloud-specific virtual solution deployment model. However, since every cloud provider uses proprietary description language to describe its services and resources, therefore, providers who are interested in composing services from different cloud providers are unable to do so because of interoperability issues of different description languages.

4.2.2 PaaS description languages

At PaaS level, very little work exists focusing on modeling and composition. A service composition platform based on GEMBus (GEANT Multi-domain Bus) is presented in [27]. In this framework, Enterprise Service Bus (ESB) platform, which is a SOA- and web service-based solution, is extended with automated service composition functionality and core services. Authors in this work argue that ESB can be considered as a logical choice for creating PaaS service composition platform due to its distributed resource integration capability.

4.2.3 SaaS description languages

A reference meta-model is presented in [37] that provides a high-level cloud vocabulary and service design elements with semantic interpretation. However, no mechanism is provided to describe QoS attributes of cloud services. There are some related research efforts in the service-oriented environment, but they are also applicable to SaaS application for designing and modeling in the cloud-based applications. For example, in [58], an approach for modeling service-oriented architecture-based applications using CBDI-SAE meta-model is proposed. In this work, authors discussed different modeling activities to build models and describe relations between these models. Another work for modeling applications is presented in [93]. Authors in this work proposed abstract service descriptions (ASD) for service-based applications. It adopted MDE approach for modeling the structural part of ASD and the reasoning mechanism.

4.2.4 XaaS description languages

At XaaS level, a cloud service composition framework is presented in [97]. It is based on the assumption that business organizations usually find services and resources in the cloud environment as a partial solution to their requirements. This framework selects the optimal composition of services using long-term QoS requirements for the user. In [35], an agent-based cloud service composition framework is presented for multi-cloud environment. The agents in the framework are able to compose services based on user requirements. Moreover, agents also have the capability of semi-recursive contract net protocol and service capability tables. The service capability tables are the catalogues about cloud participant and the available services in the system. The main advantage of such composition frameworks is to assist automatic composition and integration of cloud services to provide value-added services to cloud users. However, the existing cloud composition frameworks lack integration of services from emerging domains such as IoT, CPS, and industry 4.0.

4.2.5 SOA description languages

A large amount of work already exists in defining standardization for SDLs in SOA domain. The most notable standard bodies in SOA domain are W3C and OASIS. W3C is developing standards for interoperable technologies for Web services such as WSDL [21] and WS-* protocol stack including WS-Agreement [6, 8, 42] and WS-policy [8]. On the other hand, OASIS is developing standards for e-business in a much broader context. For instance, OASIS Reference model [28, 59] for SOA defines a framework for understanding entities and their relationships in any service-oriented environment. The organization and utilization of distributed entities and their capabilities is the responsibility of this reference model. Another initiative for defining reference model for SOA is SoaML [74] from the Object Management Group (OMG) that provides specification and design of services within SOA using a meta-model and UML profile. These standardization efforts in SOA domain can also benefit cloud computing community and drive research to define a generic SDL to support different cloud service operations.

4.2.6 SWS description languages

There are several efforts that focus on defining standard format for machine understandable service descriptions in order to be delivered and consumed over the Internet. SWS add an additional layer of semantics to the current web services. In this section, SWS approaches and descriptions that focus on composition and modeling are presented. WSMO-Lite, a lightweight ontology for web service semantics, is proposed in [79]. It forms basis for semantic automation by distinguishing four aspects of services, which include function, behavior, information model, and non-functional properties. Moreover, this work also proposed hRESTS and MicroWSMO formats that mirror WSDL and SAWSDL to enable RESTful services to combine with WSDL services in a single semantic framework. In [9], a generalized framework for semantic-based automatic web service composition is proposed. In this framework, composition solution is obtained by utilizing conditional directed graph, control flow, information, and preconditions and postconditions. The result of the composition is represented using OWL-S documents. Moreover, the filtering and ranking of suitable services for composition utilizes centrality measure of social networks along with functional and non-functional attributes of the service. In [40], an integrated approach for automated web service composition thorough planning is proposed. In this work, OWL-S web service descriptions are transformed into planning problem using Planning Domain Definition Language (PDDL) and then composition process is performed to achieve optimal composite service. For this purpose, two software systems PORSCE II and VLEPPO are used. PORSCE II (utilized for transformation procedures and semantic enhancement) and VLEPPO (a general-purpose system for designing and solving planning problems).

In conclusion, SWS approaches focus on two main issues for web services. One is the development of new techniques to describe semantic descriptions and the contents of web services such as WSDL-S [2], OWL-S [60], WSMO [29], SWRL [41], and SAWSDL [52], and the second is the development of new methodologies to share service descriptions in order to discover them efficiently, which is discussed in the next section.

4.3 Discovery and selection languages

Cloud service discovery and selection is a process of searching the internet and then exploring the published service offers to find the most suitable cloud service for the user. It performs matching of the user requirements with the descriptions of available services, consisting of service functionality and performance (e.g., QoS descriptions), and returns a ranked list of suitable cloud services [66]. Service providers prepare descriptions of the service offers for users. Recent literature review shows that very little work has been done to support service discovery in cloud computing as shown in Table 5.

4.3.1 IaaS description languages

At IaaS level, a semantic-based approach for cloud service discovery is proposed by Dastjerdi et al. in [25]. In this approach, the deployment of virtual appliances on the cloud is automated. For this purpose, it first used Open Virtualization Format (OVF) to package the disk images with the user and hardware requirements together in a standard format. Then these IaaS services were advertised using the proposed service descriptions. Finally, an ontology-based service discovery mechanism was proposed to find the best services for the users. In [1], a CloudLaunch application is developed as a uniform platform for discovering and the deploying applications for different cloud providers. It acts as a middleware between complex cloud applications having their customized interfaces, launch process, and cloud platform. The cloud service providers do not offer standard or uniform interfaces for service descriptions, which makes it difficult for such a middleware to discover cloud applications in the first place. Moreover, above-mentioned approaches are not able to autonomously manage and learn from historic service.

Table 5 Discovery and selection languages

Objective	Domain	Delivery model	Scope	Description languages
Discovery and selection languages	Cloud	IaaS	Technical	Compatibility-aware cloud [26], CloudLaunch [1]
		PaaS	Business and technical	CompatibleOne [24, 96]
		SaaS	Business	Customer-centric model [16], dynamic and attribute-based discovery [34]
		XaaS	Business and technical	CSDC [38], Cloudle [45], User-centric cloud registries [82], CloudRec [99], Ontology-based Categorization [4], CSOnt [10]
	SOA	N/A	Technical	WSDL [21], WADL [36]
		N/A	Business	USDL [19, 18], O'Sullivan [71]
	SWS	N/A	Technical	WSDL-S [2], OWL-S [60], SAWSDL [52], SPARQL and intelligent agents [80], SEMREG-Pro [65], ManuHub [17]

Therefore, an integrated approach is required to solve these problems by taking the perspective of both cloud provider and cloud users.

4.3.2 PaaS description languages

An open-source cloud service broker platform CompatibleOne is proposed for cloud service discovery in [24, 96]. It provides simple interface to discover and deploy cloud services and resources. It is based on open standards such as Cloud Data Management Interfaces (CDMI), Open Cloud Computing Interfaces (OCCI), and the proposed description language which is known as CompatibleOne Resource Description System (CORDS). As a service broker, it takes the service requests from the users in XML file called MANIFEST that captures the business and technical specifications. This document is then used to discover IaaS and PaaS resources. However, main drawback of using XML-based method for service discovery is that it offers less semantic support. Ontology-based methods such as OWL/RDF/WSML can be used to enhance matchmaking for efficient service discovery.

4.3.3 SaaS description languages

An approach that focuses business requirements of cloud applications is presented in customer-centric cloud service model [16]. This model finds the relationship among customers' subscriptions versus service providers offering and cloud infrastructure service versus cloud application service. This relationship helps the model to be used for service design and modeling by the service provider as well as service discovery and selection by the customer. However, this work is presented as a reference model for a case study only and does not provide any implementation detail for service discovery.

In addition, despite the increasing use in cloud computing infrastructures, there is still no easy way to dynamically offer IoT resources as services [84]. The major hurdle is the incapability of current approaches for on-demand service discovery, integration, and composition of IoT devices [3, 32]. Therefore, primitive nature of web service discovery approaches makes it difficult for cloud service to fully utilize existing methods. Authors in [14] argue that in order to resolve this issue and make cloud services to be discovered effectively, service providers need to publish as much dynamic attributes and up-to-date information about services as possible.

4.3.4 XaaS description languages

A cloud service discovery system (CSDC) is proposed in [38]. This system is based on cloud ontology that is used to identify similarities and dissimilarities among cloud services. It performs a similarity reasoning which is based on the earlier grid computing resource matching algorithms. The cloud ontology developed in this work was used to build a semantic-based cloud service search engine known as Cloudle [45]. It maintains a record for available cloud services in a database. The search query of the user is processed through a semantic reasoning engine that performs similarity matching between the user requirements and the service concepts using cloud ontology. The result of the Cloudle search engine is a list of matching cloud services based on the concept similarity, price, and cost utility. A similar work is also proposed in [4] which uses an ontology to discover and categorize cloud services. The ontology developed in this work is based on NIST cloud computing standard, but it provides support for limited number of concepts for ontology matching. This ontology is also used to classify cloud services into different categories based on their characteristics. In a recent work, cloud ontology known as CSOnt [10] was developed that is used to build a cloud service discovery crawler. This crawler employs a semantic similarity measures using two-level matching, which is based on term frequency–inverse document frequency (TF-IDF) and Latent Dirichlet Allocation (LDA) model. The requirements for establishing user-centric cloud service registries are highlighted in [82]. Authors analyzed six use cases and created business vocabulary for common service selection criteria and domain-specific language (DSL) to describe service. A brokering and matchmaking component was also implemented to support the users in their selection process. Another user-centric framework for cloud service recommendation is presented in [99]. This model supports to achieve personalized QoS assessment of cloud services by integrating community-based QoS assessment with iterative algorithm to discover a set of users and service communities from large-scale QoS data. Author in this work argues that discovered communities can serve as a bridge to relate users and services.

4.3.5 SOA description languages

The most notable and used standard in SOA domain is Web Service Description Language (WSDL) [43]. WSDL [21] is an XML-based specification that allows users to specify functional and operational characteristics of a Web service such as interfaces, operations, binding protocols, and endpoint addresses. WADL (Web Application Description Language) [36] is an XML description of HTTP-based Web applications (e.g., REST web services). Typically, these services are described using textual description. WADL provides a formal XML-based specification that allows these services to be machine processable. However, for the wide-scale application of these services in businesses, there are concerns that needs to be addressed such as the end-to-end delivery of a service, its usage, data sharing, quality-related metrics, provisioning to a consumer possibly over a specified period of time with a payment structure, service level agreement, and related legal obligations. The existing SDLs do not provide solutions for these problems. USDL (Unified Service Description Language) [19, 18] is developed by taking holistic and comprehensive view of the SDLs. It has been formed to define a general language for specifying both business and technical services. Technical services are considered as the specifications of WSDL, REST, or any other technical specifications, whereas the business services are defined as business activities provided by the service provider to create value for consumer. Consequently, it enables service providers to publish a service by describing its broad characteristics through USDL which also helps in better discovery and selection of the service. Nevertheless, it is not developed for cloud computing and, therefore, do not support cloud service delivery models. Moreover, it also lacks semantic support that is crucial for interoperability of cloud services provided by different service providers.

4.3.6 SWS description languages

In this section, few important studies in semantic web services domain are discussed that focus on service discovery and selection. SPARQL and intelligent agents are used in [80] for discovering semantic web services. Authors in [80] show that preconditions and postconditions of services can be expressed in SPARQL. Moreover, authors also discuss that goal of agent can also be expressed in SPARQL. This asserts, as a consequence, that a suitable service can be discovered by identifying the satisfiability of the agent's goal and the truth values of precondition and postcondition of the service. SEMREG-Pro, a semantic service registry, is proposed in [65]. This registry provides a user interface for the service user to discover semantic web service profiles from the registry using domain ontology. Moreover, it also allows service users to subscribe to the registry and get notified whenever a match to their requests is found. In another work [17], ManuHub is proposed that is semantic web system to administer distributed manufacturing service provided by ubiquitous virtual enterprises. It provides automatic retrieval of manufacturing services via ontological matching of the services' capabilities. Although it allows collaboration among manufacturing enterprises by providing interoperability, it still lacks to incorporate existing manufacturing resource ontology and domain-specific ontologies, which limits its deployment for real-world applications.

4.4 Service level agreement languages

The SLA languages have the capability to describe the functionality and quality of service (QoS) attributes of a service that service provider is willing to offer to its customer. It can describe the obligations of the involved parties such as definition of service level objective (e.g., response time must be less than 2 s for every invocation of service) and action guarantees (e.g., if service level objective is met then customer will pay certain service fee to provider). Table 6 shows the SLA languages covered in the survey.

4.4.1 Cloud (XaaS) SLA languages

There are many SLA languages in the literature that can be used to define SLA template for the service offering. WS-Agreement [6] is one of the widely used SLA language both in SOA and in cloud computing domains. It was proposed as a protocol for establishing service level agreement between two parties. But it has a drawback of syntactical matching of customer requirements with the service capabilities. Extensions of WS-Agreement have been proposed to enable semantic-based matching of agreements for efficient discovery [73]. Another extension of WS-Agreement is the WS-Negotiation [94] with aim of adding negotia-

Table 6 Service level agreement languages

Objective	Domain	Delivery model	Scope	SLA languages
Service level agreement languages	Cloud	XaaS	Business and technical	WS-Agreement [6], QoWL [12], GXLA [88], SLAC [92], SLA* [46]
	SOA	–	Business and technical	WSLA [47], WS-Agreement [6], SLAng [54], WSOL [89], RBSLA [76], SLA* [46]

tion capabilities to agreements. WS-Negotiation provides an additional layer when creating WS-Agreement by specifying the process of valid agreement offers. WS-Negotiation consists of three parts: Negotiation Message, Negotiation Protocol, and Negotiation Decision Marking. Negotiation Message describes format of the exchanged message among involved parties, Negotiation Protocol describes the rules that the involved parties should follow for negotiation, and Negotiation Decision Making contains private information about the decision process, cost–benefit model, and other strategies. QoWL [12] is an SLA language that extends BPEL to define QoS specification for workflows in Grid environment. It is an XML-based specification with language constructs for specification of QoS constraints. GXLA [88] is an implementation of generalized SLA model that defines the relationship of role-based multi-party service. The intention of GXLA is to capture complex nature of service interactivity of all sorts of IT business relationships by modeling SLAs in broader range. SLAC [92] is developed specifically for cloud computing environment. It is open source and formally defined in order to avoid ambiguity. It enables multi-party agreement that is a requirement in most cloud SLAs. It supports business aspects of the domain, such as pricing schemes, business actions, and metrics. However, it does not provide support for matchmaking and, hence, makes it difficult to discover SLA offerings.

4.4.2 SOA SLA languages

In SOA domain, one notable and widely used SLA language is Web Service Level Agreement (WSLA) [57] that defines an agreement among the involved parties. It helps the involved parties to know their capabilities and responsibilities in a single document. However, it cannot help involved parties to reach to an agreement. Therefore, WS-Agreement [6] and its negotiation protocol WS-Negotiation [94] were proposed. SLAng [54] is one of the earliest SLA language that focus on the management, performance metrics, and automation of systems, particularly in the IT domain. It provides a model-driven approach of SLA definition. It defines three main constructs for SLA: SLA metrics, SLA categories, and responsibilities. WSOL [89] is a formal specification language for functional constraints and QoS constraints particularly non-functional constraints, price offering, and relationship with other service offerings of the same service. A service offering can be defined as classification of service based on its QoS parameters such as usage privileges, service priorities, and response time guarantees to consumers. The objective of WSOL is the dynamic adaptation of web service composition using manipulation of service offerings. RBSLA [76] is another SLA language that is based on RuleML, a Semantic Web rule standard. The machine-readable syntax of this language allows SLAs to be specified in the form of rules that can then be fed into a rule engine in order to monitor the contract performance at runtime. Unlike other SLA specification languages, SLA* [46] is a domain-independent syntax for machine-readable SLA and SLA templates. It consists of SLAs and SLA templates—collectively called SLA(T)s. It was developed as part of SLA@SOI project that has been used in Cloud computing platforms such as CONTRAIL and Cloud-TM.

The focus of the above-mentioned SLA languages has been on providing efficient and automatic negotiation mechanism, while other important issues such as support for SLA offer/template matchmaking and discovery, technical description of the service, and runtime dynamic attributes have been neglected. Moreover, there is no framework for different SLA languages to interoperate in order to discover, select, and negotiate to reach to an agreement.

5 Research issues

In cloud computing environment, the description languages, models, and frameworks seem to be captivated by model-driven engineering (MDE) approach. There is neither a consensus to develop model-driven cloud applications, nor there is any defined standard for cloud service description. Moreover, the focus of these description languages is also fragmented. During the literature review, it was found that many SDLs focus on service deployment and provisioning and there were very limited description languages that cater solutions to service discovery and service selection. It can be concluded based on the above discussion that the existing cloud service description and specification languages are deficient in specifying the service from a broader perspective. Therefore, a comprehensive description language is required that can specify cloud services at different service delivery models, from different perspectives (e.g., technical or business) and for different users (e.g., consumer, provider, or developer). Although, USDL a language that is developed with holistic and wider vision to cover most of the issues discussed above, however, it still lacks the ability to specify service delivery and specification models for cloud services, without any well-defined mechanism to represent dynamic attributes of cloud services, and lacks the service user-related specifications (such as privacy preservation). In the following, we describe the identified research issues in detail.

- Interoperability issues due to heterogeneous cloud SDLs

There is no consensus among cloud service providers on a particular SDL. Different cloud providers (CPs) use different SDLs to describe their services. Moreover, every cloud provider has its own definition, interpretation, and value types for different service attributes. This poses some issues in cloud service operations (e.g., in service discovery, it makes difficult to match user requirements with service descriptions across heterogeneous multi-cloud environment). Another issue arises in cloud service compositions, when a service provider is interested in composing services from different cloud providers in order to give value-added services to its users. The service provider is unable to discover best services in the first place because of interoperability issues due to heterogeneous SDLs. This can be resolved by using a standard model, such as TOSCA, to develop new and align existing service descriptions for providing continuous and consistent cloud specifications. Moreover, ontologies provide great support to resolve interoperability issues by defining common vocabulary for different concepts for cloud computing. There are already a few research efforts to provide semantic support for the establishment of commoditized cloud service market infrastructure [87]. This can also benefit cloud SDLs to resolve interoperability issues. As a first step toward this direction, service ontologies [10, 25, 38, 45, 100] have been developed to provide domain knowledge and standard interfaces for service descriptions. However, they do not focus on the all the aspects of service descriptions and mostly support service discovery and coordination aspects.

- No support for dynamic attributes in cloud SDLs

Published SLA offers and service descriptions become frequently outdated due to change in service offers. This leads to the issue of dynamically updating the service offers in heterogeneous multi-cloud environment that will enable services to be formed autonomously. The existing cloud service descriptions do not support the update of the published service offers to change dynamically. This problem negatively effects the service matchmaking which ultimately makes cloud service discovery less efficient. In general, this issue arises due to the lack of support of service descriptions for on-demand service discovery, and composition of cloud services. There is no integrated framework that can provide on-demand service discovery, and composition of cloud services to suit the need of an application or service user autonomously. One of the work in this direction [14] argued that in order to resolve this issue and make cloud services to be discoverable effectively, cloud service providers need to publish as much dynamic attributes and up-to-date information about services as possible. In addition, due to the dynamic nature of such attributes, a mechanism to monitor service state and automatically learn from these changing attributes to provide effective service discovery technique is required.

- Lack of support for integration and provisioning of IoT resources through cloud platform

Due to the proliferation of Internet of Things (IoT), a lot of smart devices with sensor capabilities have emerged, which can produce huge amount of data. Cloud computing can be used as a platform for these IoT devices (such as sensors and smart devices) to be dynamically offered on-demand as utility-based (i.e., pay as you go) services. But existing cloud service descriptions are not designed for integration and provisioning of IoT resources. Different IoT devices have different types of capabilities and limitations. Moreover, IoT devices and services are dynamic in nature and change their states and QoS properties, frequently depending upon their usage environment. Authors in [39] outline some factors such as changing service requirements of consumers, unpredictable network performance, and frequent system maintenance as the reasons for such dynamic change in services. Some of the recent work has tried to address this issue through providing SLA guarantees for IoT resources in cloud environment [67, 68, 69]. However, despite the increasing use in cloud computing infrastructures, there is still no easy way to dynamically offer IoT resources as on-demand services [84]. The major hurdle is the incapability of current service description and specification approaches in providing IoT resources as services [3, 32, 48].

6 Conclusion

Several cloud SDLs have been proposed in the current literature. The purpose of these description languages varies considerably depending upon their main objective and usage in different service operations. Consequently, the investigation of diverse techniques of cloud SDLs and their features is of great importance due to lack of current surveys in this area. In this article, we proposed a common criteria for comparing and classifying existing cloud SDLs with the goals to (1) identify common features of cloud SDLs in different service operations by highlighting the main purpose of service description (such as deployment, provisioning, discovery, and composition) and (2) to identify research issues to develop the foundations of a standard cloud SDL. In this respect, this survey presented essential foundation for highlighting features and limitations of current SDLs.

References

1. Afgan E et al (2018) CloudLaunch: discover and deploy cloud applications. Future Gener Comput Syst. https://doi.org/10.1016/j.future.2018.04.037
2. Akkiraju R et al (2006) Web service semantics—WSDL-S. Available at: https://www.w3.org/Submission/WSDL-S/
3. Alam S, Chowdhury MMR, Noll J (2010) SenaaS: an event-driven sensor virtualization approach for internet of things cloud. In: 2010 IEEE international conference on networked embedded systems for enterprise applications, NESEA 2010. https://doi.org/10.1109/nesea.2010.5678060
4. Alfazi A et al (2015) Ontology-based automatic cloud service categorization for enhancing cloud service discovery. In: Proceedings—IEEE International Enterprise Distributed Object Computing Workshop, EDOCW, 2015-Novem, pp 151–158. https://doi.org/10.1109/edoc.2015.30
5. Amazon (2019) Amazon EC2. Available at: https://aws.amazon.com/ec2/
6. Andrieux A, Czajkowski K, Dan A, Keahey K, Ludwig H, Nakata T et al (2007) Web services agreement specification (WS-Agreement)', 2. https://doi.org/10.1007/s13398-014-0173-7.2
7. AWS (2019) AWS elastic beanstalk. Available at: https://aws.amazon.com/elasticbeanstalk/
8. Bajaj S, Box D (2006) Web services policy 1.2-framework (WS-policy). In: W3C Member …, pp 1–25
9. Bansal S et al (2016) Generalized semantic Web service composition. Serv Oriented Comput Appl 10(2):111–133. https://doi.org/10.1007/s11761-014-0167-5
10. Ben Djemaa R, Nabli H, Amous Ben AmorI (2019) Enhanced semantic similarity measure based on two-level retrieval model. In: Concurrency and computation: practice and experience (August 2017), p e5135. https://doi.org/10.1002/cpe.5135
11. Bergmayr A et al (2018) A systematic review of cloud modeling languages. ACM Comput Surv 51(1):1–38. https://doi.org/10.1145/3150227
12. Brandic I, Pllana S, Benkner S (2006) High-level composition of QoS-aware grid workflows: An approach that considers location affinity. In: 2006 workshop on workflows in support of large-scale science, WORKS'06. https://doi.org/10.1109/works.2006.5282347
13. Brereton P et al (2007) Lessons from applying the systematic literature review process within the software engineering domain. J Syst Softw 80(4):571–583. https://doi.org/10.1016/j.jss.2006.07.009
14. Brock M, Goscinski A (2009) Attributed publication and selection for Web service-based distributed systems. In: SERVICES 2009—5th 2009 world congress on services, (PART 1), pp 732–739. https://doi.org/10.1109/services-i.2009.82
15. Caglar F et al (2013) Model-driven performance estimation, deployment, and resource management for cloud-hosted services. In: Proceedings of the 2013 ACM workshop on Domain-specific modeling—DSM'13, pp 21–26. https://doi.org/10.1145/2541928.2541933
16. Cai H et al (2009) Customer centric cloud service model and a case study on commerce as a service. In: CLOUD 2009—2009 IEEE international conference on cloud computing, pp 57–64. https://doi.org/10.1109/cloud.2009.67
17. Cai M, Zhang WY, Zhang K (2011) 'ManuHub: a semantic web system for ontology-based service management in distributed manufacturing environments. IEEE Trans Syst Man Cybern Part A Syst Hum 41(3):574–582. https://doi.org/10.1109/TSMCA.2010.2076395
18. Cardoso J et al (2010) Towards a unified service description language for the internet of services: requirements and first developments. In: Proceedings—2010 IEEE 7th international conference on services computing, SCC 2010, pp 602–609. https://doi.org/10.1109/scc.2010.93
19. Cardoso J, Winkler M, Voigt K (2009) A service description language for the internet of services. Language 2009(1):229–240. https://doi.org/10.1109/SCC.2010.93
20. Chapman C et al (2012) Software architecture definition for on-demand cloud provisioning. Cluster Comput 15(2):79–100. https://doi.org/10.1007/s10586-011-0152-0
21. Chinnici R et al (2003) Web services description language (WSDL) Version 1.2. In: W3C, pp 1–78
22. Collazo-Mojica XJ et al (2010) Virtual environments : easy modeling of interdependent virtual appliances in the cloud. In: Proceedings of the SPLASH 2010 workshop on flexible modeling tools (SPLASH 2010). https://doi.org/10.1109/infocom.2006.139
23. Collazo-Mojica JX, Sadjadi SM (2011) A metamodel for distributed ensembles of virtual appliances. In: Proceedings of the 23rd international conference on software engineering and knowledge engineering (SEKE), pp 560–565. Available at: http://130.203.133.150/viewdoc/summary;jsessionid=EBAFEDF581E17AC0CCAB0005C2FDE092?doi=10.1.1.228.6989
24. CompatibleOne-Project (2012) CompatibleOne open source cloud broker architecture overview. In: CompatibleOne white paper, pp 1–10. Available at: http://www.compatibleone.org/bin/view/Discover/Overview, 10 Sept
25. Dastjerdi AV (2013) QoS-aware and semantic-based service coordination for multi-cloud environments, PhD thesis, University of Melbourne
26. Dastjerdi AV, Buyya R (2014) Compatibility-aware cloud service users. 7161(c): 1–14. https://doi.org/10.1109/tcc.2014.2300855
27. Demchenko Y et al (2012) GEMBus based services composition platform for cloud PaaS. In: Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics), 7592 LNCS, pp 32–47. https://doi.org/10.1007/978-3-642-33427-6-3
28. Estefan J et al (2012) Reference architecture foundation for service oriented architecture version 1.0. In: OASIS Committee …, pp 1–118
29. Fensel D et al (2011) Web service modeling ontology. In: Semantic web services. https://doi.org/10.1007/978-3-642-19193-0_7
30. Frey S, Hasselbring W (2010) Model-based migration of legacy software systems to scalable and resource-efficient cloud-based applications: the cloudmig approach. In: Proceedings of the first international conference on cloud computing, GRIDs and virtualization, (c), pp 155–158. Available at: http://www.thinkmind.org/index.php?view=article&articleid=cloud_computing_2010_6_40_50065
31. Gavvala SK et al (2019) QoS-aware cloud service composition using eagle strategy. Future Gener Comput Syst 90:273–290. https://doi.org/10.1016/j.future.2018.07.062
32. Georgakopoulos D et al (2016) Discovery-driven service oriented IoT architecture. In: Proceedings—2015 IEEE conference on col-

laboration and internet computing, CIC 2015, pp 142–149. https://doi.org/10.1109/cic.2015.34
33. Ghazouani S, Slimani Y (2017) A survey on cloud service description. J Netw Comput Appl 91:61–74. https://doi.org/10.1016/j.jnca.2017.04.013
34. 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. https://doi.org/10.1016/j.future.2010.03.009
35. Gutierrez-Garcia JO, Sim KM (2013) Agent-based cloud service composition. Appl Intell 38(3):436–464. https://doi.org/10.1007/s10489-012-0380-x
36. Hadley MJ (2006) Web application description language. Technical Report. Sun Microsystems, Inc., Mountain View, CA
37. 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, pp 98–103. https://doi.org/10.5220/0003393800980103
38. Han T, Sim KM (2010) An ontology-enhanced cloud service discovery system. In: Engineering and computer scientists, I, pp 644–649. Available at: http://www.iaeng.org/publication/IMECS2010/IMECS2010_pp644-649.pdf
39. Hani AFM, Paputungan IV, Hassan MF (2015) Renegotiation in service level agreement management for a cloud-based system. ACM Comput Surv 47(3):1–21. https://doi.org/10.1145/2716319
40. Hatzi O et al (2012) An integrated approach to automated semantic web service composition through planning. IEEE Trans Serv Comput 5(3):319–332. https://doi.org/10.1109/TSC.2011.20
41. Horrocks I, Patel-Schneider PF, Boley H, Tabet S, Grosof B, Dean M (2004) SWRL: a semantic web rule language combining OWL and RuleML. Available at: https://www.w3.org/Submission/SWRL/
42. Hung PCK (2004) WS-Negotiation: an overview of research issues. In: 37th annual Hawaii international conference on system sciences, 2004. Proceedings of the 37(C), pp 1–10. https://doi.org/10.1109/hicss.2004.1265100
43. January WCCR (2006) Web services description language (WSDL) Version 1.2, pp 1–86
44. Jula A, Sundararajan E, Othman Z (2014) Cloud computing service composition: a systematic literature review. Expert Syst Appl 41(8):3809–3824. https://doi.org/10.1016/j.eswa.2013.12.017
45. Kang J, Sim KM (2016) Ontology-enhanced agent-based cloud service discovery. Int J Cloud Comput 5(1/2):144. https://doi.org/10.1504/IJCC.2016.075125
46. Kearney KT, Torelli F, Kotsokalis C (2010) SLA*: an abstract syntax for service level agreements. In: Proceedings—IEEE/ACM international workshop on grid computing, pp 217–224. https://doi.org/10.1109/grid.2010.5697973
47. Keller A, Ludwig H (2003) The WSLA framework: specifying and monitoring service level agreements for web services. J Netw Syst Manag 11(1):57–81. https://doi.org/10.1023/A:1022445108617
48. Khaled AE et al (2018) IoT-DDL-device description language for the "T" in IoT. In: IEEE Access. IEEE, vol 6, pp 24048–24063. https://doi.org/10.1109/access.2018.2825295
49. Kitchenham B, Charters S (2007) Guidelines for performing systematic literature reviews in software engineering. Available at: https://community.dur.ac.uk/ebse/biblio.php?id=51
50. Kitchenham B et al (2010) Systematic literature reviews in software engineering-A tertiary study. Inf Softw Technol 52(8):792–805. https://doi.org/10.1016/j.infsof.2010.03.006
51. Konstantinou AV et al (2009) An architecture for virtual solution composition and deployment in infrastructure clouds. In: Proceedings of the 3rd international workshop on virtualization technologies in distributed computing—VTDC'09, p 9. https://doi.org/10.1145/1555336.1555339
52. Kopecký J et al (2007) SAWSDL: semantic annotations for WSDL and XML schema. IEEE Internet Comput 11(6):60–67. https://doi.org/10.1109/MIC.2007.134
53. Kritikos K et al (2013) A survey on service quality description. ACM Comput Surv 46(1):44
54. Lamanna DD, Skene J, Emmerich W (2003) SLAng: a language for defining service level agreements. In: Proceedings of the IEEE computer society workshop on future trends of distributed computing systems, pp 100–106. https://doi.org/10.1109/ftdcs.2003.1204317
55. Lipton P et al (2013) Topology and orchestration specification for cloud applications—PRIMER, pp 1–114
56. Liu D, Zic J (2011) Cloud#: a specification language for modeling cloud. In: Proceedings—2011 IEEE 4th international conference on cloud computing, CLOUD 2011, pp 533–540. https://doi.org/10.1109/cloud.2011.18
57. Ludwig H et al (2002) Web service level agreement (WSLA) language specification. In: IBM Corporation, pp 1–110. https://doi.org/10.1109/wecwis.2002.1021238
58. Ma Z, Kang L, Chen H (2010) An approach to modeling service-oriented solutions based on CBDI-SAE metamodel for SOA 2.0. In: Proceedings—5th IEEE international symposium on service-oriented system engineering, SOSE 2010. IEEE, pp 82–85. https://doi.org/10.1109/sose.2010.40
59. MacKenzie CM et al (2006) reference model for service oriented architecture. Public Rev Draft 2:1–31
60. Martin D et al (2007) Bringing semantics to web services with OWL-S. World Wide Web 10(3):243–277. https://doi.org/10.1007/s11280-007-0033-x
61. Maximilien EM et al. (2009) Toward cloud-agnostic middlewares. In: Proceeding of the 24th ACM SIGPLAN conference companion on object oriented programming systems languages and applications—OOPSLA'09, p 619. https://doi.org/10.1145/1639950.1639957
62. Microsoft (2019) Microsoft azure. Available at: https://azure.microsoft.com/en-au/
63. Mohsin A, Janjua NK (2018) A review and future directions of SOA-based software architecture modeling approaches for System of Systems. Serv Oriented Comput Appl 12(3):183–200. https://doi.org/10.1007/s11761-018-0245-1
64. Nadjaran Toosi A, Sinnott RO, Buyya R (2018) Resource provisioning for data-intensive applications with deadline constraints on hybrid clouds using Aneka. Future Gener Comput Syst 79:765–775. https://doi.org/10.1016/j.future.2017.05.042
65. Nawaz F, Qadir K, Ahmad HF (2008) SEMREG-Pro: a semantic based registry for proactive web service discovery using publish-subscribe model. In: Proceedings of the 4th international conference on semantics, knowledge, and grid, SKG 2008, pp 301–308. https://doi.org/10.1109/skg.2008.97
66. Nawaz F, Asadabadi MR et al (2018) An MCDM method for cloud service selection using a Markov chain and the best-worst method. Knowl Based Syst 159:120–131. https://doi.org/10.1016/j.knosys.2018.06.010
67. Nawaz F, Janjua NK et al (2018) Event-driven approach for predictive and proactive management of SLA violations in the cloud of things. Future Gener Comput Syst 84:78–97. https://doi.org/10.1016/j.future.2018.02.025
68. Nawaz F et al (2019) Proactive management of SLA violations by capturing relevant external events in a cloud of things environment. Future Gener Comput Syst. https://doi.org/10.1016/j.future.2018.12.034
69. Nawaz F, Janjua NK, Hussain OK (2019) PERCEPTUS: predictive complex event processing and reasoning in IoT-enabled supply chain. Knowl Based Syst. https://doi.org/10.1016/j.knosys.2019.05.024

70. Nguyen DK (2013) Blueprint model and language for engineering cloud applications, PhD thesis, Tilburg University, School of Economic and Management
71. O'Sullivan J (2006) Towards a precise understanding of service properties. In: Faculty of information technology, Ph.D., p 232. http://eprints.qut.edu.au/16503/
72. Oberle D et al (2013) A unified description language for human to automated services. Inf Syst 38(1):155–181. https://doi.org/10.1016/j.is.2012.06.004
73. Oldham N et al (2006) Semantic WS-agreement partner selection. In: Proceedings of the 15th international conference on World Wide Web, pp 697–706. https://doi.org/10.1145/1135777.1135879
74. OMG (2012) Service oriented architecture Modeling Language (SoaML) specification. In: Language, pp 1–144
75. Paraiso F, Merle P, Seinturier L (2016) soCloud: a service-oriented component-based PaaS for managing portability, provisioning, elasticity, and high availability across multiple clouds. Computing 98(5):539–565. https://doi.org/10.1007/s00607-014-0421-x
76. Paschke A, BichlerM, Dietrich J (2005) RBSLA—a declarative rule-based service level agreement language based on RuleML. In: Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics), 3791 LNCS, pp 209–217. https://doi.org/10.1007/11580072_19
77. Pawluk P et al (2012) Introducing STRATOS: a cloud broker service. In: Proceedings—2012 IEEE 5th international conference on cloud computing, CLOUD 2012. IEEE (ii), pp 891–898. https://doi.org/10.1109/cloud.2012.24
78. Rochwerger B et al (2011) RESERVOIR—when one cloud is not enough. In: Computer, vol 44. pp 1–7, IEEE. https://doi.org/10.1109/MC.2011.64
79. Roman D et al (2015) WSMO-lite and hRESTS: lightweight semantic annotations for web services and RESTful APIs. J Web Semant 31:39–58. https://doi.org/10.1016/j.websem.2014.11.006
80. Sbodio ML, Martin D, Moulin C (2010) Discovering semantic web services using SPARQL and intelligent agents. J Web Semant 8(4):310–328. https://doi.org/10.1016/j.websem.2010.05.002
81. Silva GC, Rose LM, Calinescu R (2014) Cloud DSL: a language for supporting cloud portability by describing cloud entities. In: CEUR workshop proceedings, vol 1242, pp 36–45
82. Slawik M, Zilci Bİ, Küpper A (2018) Establishing user-centric cloud service registries. Future Gener Comput Syst 87:846–867. https://doi.org/10.1016/j.future.2018.03.010
83. SML (2009) Service modeling language (SML), Version 1.1
84. Soldatos J, Serrano M, Hauswirth M (2012) Convergence of utility computing with the Internet-of-things. In: Proceedings—6th international conference on innovative mobile and internet services in ubiquitous computing, IMIS 2012, pp 874–879. https://doi.org/10.1109/imis.2012.135
85. Sun YL (2012) Mapping high-level application requirements onto low-level cloud resources. J Softw Eng Appl 05(11):894–902. https://doi.org/10.4236/jsea.2012.531104
86. Sun, L., Dong, H. and Ashraf, J. (2012) Survey of service description languages and their issues in cloud computing. In: Proceedings—2012 8th international conference on semantics, knowledge and grids, SKG 2012, pp 128–135. https://doi.org/10.1109/skg.2012.49
87. Sun L et al (2014) 'Cloud service selection: state-of-the-art and future research directions. J Netw Comput Appl 45:134–150. https://doi.org/10.1016/j.jnca.2014.07.019
88. Tebbani B, Aib I (2006) GXLA a language for the specification of service level agreements. Lect Notes Comput Sci 6:201–214. https://doi.org/10.1007/11880905_17
89. Tosic V, Patel K, Pagurek B (2002) Wsol—web service offerings language. In: Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics), vol 2512, pp 57–67. https://doi.org/10.1007/3-540-36189-8_5
90. Tsai WT et al (2011) An approach for service composition and testing for cloud computing. In: Proceedings—2011 10th International symposium on autonomous decentralized systems, ISADS 2011, pp 631–636. https://doi.org/10.1109/isads.2011.90
91. Tsai WT, Sun X, Balasooriya J (2010) Service-oriented cloud computing architecture. In: ITNG2010—7th international conference on information technology: new generations, pp 684–689. https://doi.org/10.1109/itng.2010.214
92. Uriarte RB, Tiezzi F, De Nicola R (2014) SLAC: a formal service-level-agreement language for cloud computing. In: Proceedings—2014 IEEE/ACM 7th international conference on utility and cloud computing, UCC 2014, pp 419–426. https://doi.org/10.1109/ucc.2014.53
93. Vara JM et al (2012) Towards model-driven engineering support for service evolution. J Univers Comput Sci 18(17):2364–2382
94. Waeldrich O et al (2011) WS-agreement negotiation version 1.0, p 64
95. Wu L et al (2014) SLA-based resource provisioning for hosted software-as-a-service applications in cloud computing environments. IEEE Trans Serv Comput 7(3):465–485. https://doi.org/10.1109/TSC.2013.49
96. Yangui S et al (2014) CompatibleOne: the open source cloud broker. J Grid Comput 12(1):93–109. https://doi.org/10.1007/s10723-013-9285-0
97. Ye Z et al (2014) Long-term QoS-aware cloud service composition using multivariate time series analysis. IEEE Trans Serv Comput. https://doi.org/10.1109/tsc.2014.2373366
98. Ye Z et al (2016) Long-term QoS-aware cloud service composition using multivariate time series analysis. IEEE Trans Serv Comput 9(3):382–393. https://doi.org/10.1109/TSC.2014.2373366
99. Yu Q (2015) CloudRec: a framework for personalized service recommendation in the cloud. Knowl Inf Syst 43(2):417–443. https://doi.org/10.1007/s10115-013-0723-x
100. Zhang M et al (2012) An ontology-based system for cloud infrastructure services' discovery. In: CollaborateCom, ICST/IEEE, pp 524–530. https://doi.org/10.1007/978-3-642-35194-5