Chapter 9
Federated IaaS Resource Brokerage

Bruno Veloso
Instituto Superior de Engenharia do Porto, Portugal

Benedita Malheiro
Instituto Superior de Engenharia do Porto, Portugal

Fernando Meireles
Instituto Superior de Engenharia do Porto, Portugal

Juan Carlos Burguillo
Universidade de Vigo, Spain

ABSTRACT

This paper presents the CloudAnchor brokerage platform for the transaction of single provider as well as federated Infrastructure as a Service (IaaS) resources. The platform, which is a layered Multi-Agent System (MAS), provides multiple services, including (consumer or provider) business registration and deregistration, provider coalition creation and termination, provider lookup and invitation and negotiation services regarding brokerage, coalitions and resources. Providers, consumers and virtual providers, representing provider coalitions, are modelled by dedicated agents within the platform. The main goal of the platform is to negotiate and establish Service Level Agreements (SLA). In particular, the platform contemplates the establishment of brokerage SLA – bSLA – between the platform and each provider or consumer, coalition SLA – cSLA – between the members of a coalition of providers and resource SLA – rSLA – between a consumer and a provider. Federated resources are detained and negotiated by virtual providers on behalf of the corresponding coalitions of providers.

INTRODUCTION

The emergence of Cloud Computing as a new trend computing paradigm is attractive to business owners as it eliminates the requirement to plan ahead for provisioning and gives enterprises the flexibility to manage the contracted resources according to the current workload. The market for the cloud services, specifically the IaaS cloud computing market, is still maturing and brokers are emerging as the preferential middleware to match demand and offer between stakeholders. In particular, Small and Medium sized Enterprises (SME), which are typically in the early stages of adopting the cloud paradigm or provid-

DOI: 10.4018/978-1-5225-0153-4.ch009

ing cloud services, require support services and platforms to increase their competitiveness. For SME providers, brokers offer additional business opportunities and simplify the management and integration of disparate cloud services – potentially across different providers – fostering the creation of provider coalitions; for SME consumers, brokers provide seamless provider lookup and invitation as well as SLA negotiation services, increasing the chances of meeting their resource requirements at the best price and in time. The ultimate goal of this research is to support the adoption and provision of IaaS by SME both as consumers and as providers.

This paper addresses the design and development of the CloudAnchor business-to-business (B2B) brokerage platform with provider discovery and invitation as well as negotiation, establishment and management of resource service level agreements regarding single provider and federated resources. This problem is by nature distributed, decentralised, dynamic and involves multiple stakeholders (consumer and provider businesses) continuously entering and leaving the system (open system). The stakeholders are, not only, loosely coupled, but, depending on the situation, can either compete (consumers and providers compete for getting and leasing resources) or cooperate (coalitions of providers). Furthermore, these businesses wish to retain autonomy, privacy and the control of their strategic knowledge, leading to the adoption of the agent-based paradigm.

The CloudAnchor brokerage platform implements an open event-driven multi-layered agent-based architecture. Businesses are represented by dedicated autonomous agents and are, thus, able to specify their self-models, by uploading their strategic knowledge (lookup, invitation, acceptance and negotiation strategies), resource offers (providers) or resource requests (consumers), as well as build peer models of their business partners based on the outcomes of their previous interactions (individual peer trust). The layered approach allows the distribution and delegation of the interface, agreement, enterprise (knowledge and processes) and negotiation related tasks to corresponding dedicated agents, representing each business by a set of task specialized agents rather than by a single agent to increase the overall responsiveness.

The platform provides multiple services, including (consumer or provider) business registration/ deregistration, provider lookup and invitation, provider coalition creation/termination and agreement negotiation/termination. To support these functionalities, the platform contemplates the negotiation, establishment and termination of brokerage, coalition and resource agreements: (*i*) a brokerage SLA (bSLA) is a contract between the platform and a business (provider or consumer); (*ii*) a coalition SLA (cSLA) is a contract between a coalition of providers and the resulting virtual provider; and (*iii*) a resource SLA (rSLA) is a contract between a consumer and a provider (single or a virtual provider) regarding a given resource.

In terms of external events, there are business registration/deregistration, resource request/offer and SLA fulfilled/violated events. These events drive the execution of the business registration service, business deregistration service, resource provider lookup and invitation service, provider resource availability publication service and SLA termination service. In particular, whenever a consumer requests a new resource via its interface agent (resource request event), it triggers the resource finding process. The consumer enterprise agent automatically looks up and invites providers for negotiation. If the invited provider enterprise agents accept the invitation, dedicated delegate market agents are created by both consumer and provider enterprise agents to negotiate and establish the rSLA. If the providers were not able to provide single-handedly the resource, the platform attempts to create a virtual provider. Virtual providers are temporary coalitions of single providers established on the fly to provide federated resources, *i.e.*, resources which were not offered by any single provider. When an rSLA terminates, the parties involved (consumer and provider) receive an agreement fulfilled or agreement violation event.

Each consumer and provider enterprise agent creates and maintains local trust models of their peers based on the outcomes of their direct interactions: partner invitation (accepted or rejected), negotiation (success or failure to establish a SLA) and SLA enforcement feedback (fulfilled or violated). These models support all future peer interaction, namely, provider invitation, consumer and coalition invitation acceptance and negotiation of SLA instances. From a platform's perspective, the local peer trust models constitute a decentralised trust system.

In terms of contributions, the proposed brokerage platform provides: (*i*) federated resources through the creation of virtual providers, representing provider coalitions; (*ii*) the negotiation and establishment of brokerage, coalition and resource SLA instances; (*iii*) a decentralised trust model regarding providers, consumers and the platform for the negotiation and renegotiation of SLA.

This paper is organized in five sections. Section 1 presents the context of the project, identifies the problems addressed and describes the approach adopted. Section 2 presents and compares the most relevant features of related agent-based cloud resource brokerage platforms, including our proposal, and discusses the issues associated with federated cloud resources. Section 3 describes our approach, including platform design, architecture and services, and details the different SLA types contemplated and their negotiation. Section 4 describes the future developments. Finally, Section 5 draws the main conclusions.

AGENT-BASED CLOUD RESOURCE BROKERAGE

Brokerage platforms frequently adopt the agent-based paradigm since resource, service or partner brokerage is an inherently distributed, decentralised complex problem involving autonomous entities. In this context, consumers compete for provider resources, providers compete or cooperate to fulfil consumer requests and, finally, consumers and providers play opposite roles (buyer and seller). Businesses are then represented within the brokerage platform as autonomous intelligent agents, regardless of their competitive or cooperative nature, maintaining autonomy and privacy. In this section we explore the state of the art regarding agent-based trust and reputation systems, agent-based SLA negotiation frameworks and cloud brokerage systems.

Trust and Reputation

Trust and reputation are central to effective interactions in open multi-agent systems (MAS) in which agents, owned by a variety of stakeholders, continuously enter and leave the system (Huynh, Jennings & Shadbolt, 2006). Trust can arise from individual and societal perspectives. While the individual perspective results from the direct interactions with peers, the societal perspective – usually referred as reputation – is based on the observations by a collection of agents. Trust is a one to one subjective perception between truster and trustee, which depends on their mutual interaction experiences. A trustee is considered to be trustworthy if it has, according to the truster, a high probability of performing a particular service, *i.e.*, has fulfilled its obligations (Teacy, Patel, Jennings & Luck, 2006).

The reputation of a Web service provider can be, according to He, Kowalczyk, Jin and Yang (2007), established using the past performance of the provider service and, then, applied by the consumers to assess and select service providers. Cascella, Blasi, Jegou, Coppola and Morin (2013) evaluate the reputation of cloud providers based on the number of SLA violations observed and the set of explicit user preferences and use it to filter provider offers. To select the best available provider, Alhamad, Dillon and Chang (2010) propose a trust model built from three different sources: (*i*) the local experience with cloud providers; (*ii*) opinions of external cloud services; and (*iii*) the reports provided by the SLA management agent.

In CloudAnchor we build a decentralised peer trust model from the past direct interactions between stakeholders (consumers, providers and platform) and apply this model when consumers invite potential providers, providers assess invitations to negotiate and when consumers choose the best provider proposal during a negotiation. The model is built from the outcomes of three stages: the partner invitation/ acceptance, SLA negotiation and SLA enforcement. In the future, we plan to reuse the model for the renegotiation of bSLA and cSLA.

SLA Negotiation

SLA negotiation is essential to cloud brokerage since it is the stage where provider and consumer negotiate the terms of the service provision with the intent to establish a binding contract. From the large number of agent-based SLA negotiation frameworks available in the literature, we selected the following illustrative and relevant subset:

1. The Web Services SLA negotiation framework presented by Al Falasi, Serhani and Elnaffar (2011) is an agent-based system composed of provider and consumer agents. WS SLA adopts the Foundation for Intelligent Physical Agents (FIPA) Iterated Contract Net Interaction Protocol (FICNIP) to negotiate SLA instances. The proposed framework follows the W3C Web Service Policy Specification (WS-Policy) to describe the negotiation policies and adopts a utility-based negotiation strategy.
2. The Framework for Automated Service Negotiation in Cloud Computing proposed by Pan (2011) is a multi-agent system for the negotiation of cloud computing resources. It is composed of autonomous provider and consumer agents and includes a marketplace where resource offers and requests are registered and matched. The negotiation between consumer and provider agents implements a bilateral multi-step monotonic concession negotiation protocol.
3. Cloudle is an agent-based search engine for cloud service discovery. It is composed of a service discovery agent, a cloud ontology, a database of cloud services and multiple cloud crawlers. It relies on a utility-oriented strategy for coordinating concurrent negotiations and establishing multiple SLA instances (Sim, 2012).
4. The Policy-based Web Services SLA Negotiation System by Zulkernine, Martin, Craddock and Wilson (2009) encompasses provider and consumer agents. The negotiation protocol is the FICNIP. The system provides a flexible framework for SLA negotiation by incorporating multiple strategy models based on high level policies.
5. The Framework for Negotiation and SLA Management described by He *et al.* (2007) is an agent-based platform that negotiates, interacts and cooperates to facilitate autonomous and flexible SLA management. It is composed of SLA initiator and responder agents and includes a store where resource offers and requests are registered and matched. The system relies on a Universal Description Discovery and Integration (UDDI) service registry to publish the resource announcements. The framework supports SLA formation, SLA recovery and, finally, SLA profiling. The negotiation between initiator and responder agents implements the OGF WS-Agreement negotiation specification.
6. CloudAnchor contemplates the negotiation, establishment and termination of brokerage, coalition and resource agreements regarding single and federated resources. In particular, the platform: (*i*)

handles three types of SLA (brokerage, resource and coalition SLA); (*ii*) implements two SLA specifications (WS-Agreement and WS-Agreement Negotiation); (*iii*) adopts two standard negotiation protocols (one shot and FICNIP); (*iv*) uses Business Process Modelling Notation (BPMN) to define the SME behaviour within the platform, including the SLA negotiation strategies; (*v*) maintains and uses a decentralised trust model of business counterparts to negotiate SLA instances; and (*vi*) negotiates and establishes coalitions between providers (virtual providers) to provide federated resources.

Table 1 summarises the most relevant features of these agent-based cloud resource brokerage platforms. Each platform is analysed in terms of service level agreement approach, negotiated resources, service interface and implemented peer model. In terms of specifications, Al Falasi *et al.* (2011) implement the WS-Policy specification to describe negotiation policies and CloudAnchor implements the WS-Agreement specification. In terms of negotiation protocols, there is a wider diversity, ranging from the implementation of specifications, adoption of standard negotiation protocols to custom-made approaches. Al Falasi *et al.* (2011), Zulkernine *et al.* (2009) and CloudAnchor use FICNIP; Cloudle by Sim (2012) uses a variant of FCNIP; and CloudAnchor adopt the one shot negotiation protocol included in the WS-Agreement. Al Falasi *et al*. (2011) adopt a utility-based negotiation strategy; He *et al.* (2007) adopt the one shot protocol; and Pan (2011) implements a bilateral multi-step monotonic concession negotiation protocol. Regarding the service interfaces, the most recent platforms, including CloudAnchor, expose Representational State transfer (REST) Web Service (WS) interfaces, while the others adopt Simple Object Access Protocol (SOAP) Remote Procedure Call (RPC) WS interfaces. Peer modelling is implemented only by He *et al*. (2007) and CloudAnchor.

CloudAnchor not only implements several specifications and adopts several standards, *e.g.*, the WS-Agreement and WS-Agreement negotiation specifications, the FICNIP one-to-many standard negotiation protocol and RESTful WS endpoints, but is the only platform that negotiates agreements regarding brokerage, coalition and single and federated resources with the support of a decentralised trust-based peer model.

Table 1. Comparison of agent-based SLA negotiation frameworks

Platform	**SLA Specification**	**SLA Negotiation Protocol**	**Service Interface**	**Resources**	**Peer Model**
Al Falasi *et al.,* 2011	WS-Policy	FICNIP	SOAP RPC	Single	
Pan, 2011		Bilateral multi-step monotonic concession	REST	Single	
Sim, 2012		FCNIP based	REST	Single	
Zulkernine *et al.,* 2009		FICNIP	SOAP RPC	Single	
He *et al.,* 2007	WS-Agreement	One shot	SOAP RPC	Single	Reputation
CloudAnchor	WS-Agreement WS-Agreement Negotiation	One shot FICNIP	REST	Single & Federated	Trust

Brokerage Platforms

Brokerage platforms manage the use, performance and delivery of cloud services and negotiate relationships between cloud providers and cloud consumers, including the aggregation, customisation and integration of services (Fowley, Pahl & Zhang, 2014). In particular, cloud brokerage platforms can be enriched with marketplaces to bring providers and customers together, supporting partner discovery, service negotiation, agreement establishment and enforcement. The following platforms provide a representative overview of the state of the art regarding brokerage platforms:

1. Stantchev and Schröpfer (2009) describe an approach to map business processes requirements to grid and cloud infrastructures, including SLA negotiation and corresponding quality of service (QoS) enforcement. The authors formalize a structure to describe the service level requirements of the business processes and the technical services of the providers. This approach uses on-demand service level objective (SLO) structures, specifying response time, throughput and transaction rate and contemplates only single resources. The proposed scheme supports the specification of trust relations.
2. The S-Cube project described by Mahbub and Spanoudakis (2010) proposes a proactive SLA Negotiation for Service Based Systems composed of monitor, service listener, negotiation broker and runtime service discovery tool. The services are described using the Organization for the Advancement of Structured Information Standards (OASIS) Business Process Execution Language (BPEL). This system is event-driven, *i.e.*, reacts to events created by providers or consumers. The broker module uses system-defined negotiation rules regarding both single and composite provider resources.
3. The Cross-cloud Federation Model (CCFM), proposed by Celesti, Tusa, Villari and Puliafito (2010), contemplates four distinct stages: discovery, *i.e.*, the search for available resources, matchmaking, *i.e.*, the selection of the resource which best fits the requirements; and authentication, *i.e.*, the establishment of a trusted context with the selected cloud. The overall system is structured in virtual machine manager layer, virtual infrastructure layer and cloud manager layer. The CCFM is implemented in the cloud manager layer by three different agents, corresponding to the three main stages, and provides single and federated resources.
4. The Cloud Agency architecture described by Venticinque, Aversa, Di Martino, Rak and Petcu (2011) includes service discovery, SLA negotiation and enforcement as well as peer trust models. It generates and monitors SLA as well as negotiates single and federated cloud resources. SLA violation is checked by monitoring the QoS and, thereafter, applying penalties to the failing providers. The development of Cloud Agency was supported by a multi-agent toolset which provides different protocols together with authentication, agent migration, workload balance and dynamic resource allocation services.
5. The OPTIMIS toolkit, described by Ferrer *et al*. (2012), offers a set of components to enable the use of multiple cloud architectures. This toolkit supports four types of implementation scenarios: federated architecture, multi-cloud architecture, aggregation of resources by a third party broker and hybrid cloud architecture. OPTIMIS provides monitoring and automated management of services and infrastructures as well as compares different configurations to enhance business efficiency. The toolkit, which includes components for resource discovery, SLA negotiation and enforcement, supports the provision of federated resources and implements trust-based peer models.

6. The STRATOS cloud broker service described by Pawluk, Simmons, Smit, Litoiu and Mankovski (2012) has three main constituents: cloud manager, broker and monitoring modules. The broker requires information about the desired configuration and the set of objectives to submit to a multi-criteria optimization process. In order to minimise costs and avoid provider lock-in, the broker uses two specific objective functions. To support different cloud providers, the authors chose the Apache Deltacloud abstraction layer. STRATOS provides resource discovery and provides federated resources.
7. The cloud brokering architecture described by Tordsson, Montero, Moreno-Vozmediano & Llorente (2012) implements federated resource discovery. It has two main components: the cloud scheduler and the virtual infrastructure manager. The cloud scheduler adopts templates to store the provider VM configurations which will be used by the cloud scheduling algorithms. The virtual infrastructure manager, which is based on the OpenNebula virtual infrastructure manager, provides a generic user interface to deploy, monitor and control the VM instances.
8. The mOSAIC project proposed by Amato, Di Martino and Venticinque (2012) includes a multi-agent system composed of Broker, Vendor, Meeter, Tier, Mediator and Archiver agents to discover and negotiate cloud infrastructure single provider resources. The SLA negotiation protocol is FICNIP. The brokering policy sets constraints and objectives on multiple parameters. The broker analyses a number of SLA proposals and chooses the SLA proposal that satisfies the applicable constraints and objective rules, including provider reputation.
9. The Cloud@Home project reported by Cuomo *et al.* (2013) adopts an SLA-based broker for the discovery cloud infrastructure resources, including federated resources. It has a modular architecture composed of resource management, SLA management and resource abstraction modules. The SLA management module supports one shot negotiations and implements the Open Grid Forum (OGF) Web Service Agreement (WS-Agreement) specification. The negotiation policies are setup only by the administrator.
10. The Contrail project described by Cascella *et al.* (2013) is a MAS dedicated to the discovery and negotiation of single and federated cloud computing resources. The SLA instances are described by the SLA@SOI template as well as the Open Virtualization Format (OVF) descriptors and the SLAs include the reputations of each cloud provider calculated by the Contrail federation. The negotiation between agents adopts the Contrail protocol.
11. The Sky model proposed by Al Falasi *et al.* (2013) allows the creation of a marketplace where cloud providers can exchange services and infrastructure resources using the premise of social networking. This marketplace supports federated service lookup, substitution, recommendation and supervision as well as provider collaboration and provider competition. In terms of architecture, Sky is structured in a socialization modules and a federation module. This model follows the concept of community with several relationships, reputations, penalties and rewards.
12. The Cloud broker described by Pawar, Rajarajan, Dimitrakos and Zisman (2014) works as a mediation layer and integrates a trust and reputation model for the cloud environment. This broker supports multi-cloud and federated cloud deployments. The trust model is based on the SLA monitoring and user feedback ratings. The cloud broker has four different components: the cloud service recommendation (discovery), the cloud service intermediation, the cloud service aggregation and the cloud service arbitrage modules.
13. The SLA-based service virtualization architecture described by Kertész, Kecskemeti and Brandic (2014) includes a meta-negotiator, a meta-broker, a broker and an automatic service deployment

module. The users use high level meta-negotiation constructs to describe the resource requirements and the meta-broker selects a broker capable of deploying the required service. The meta-negotiation documents are published in a searchable registry service. The service provides individual resources.

14. The CompatibleOne project proposes a cloud broker, based on open standards like Cloud Data Management Interface (CDMI) and Open Cloud Computing Interface (OCCI), for the discovery and transaction of single resources. This broker uses a model called CompatibleOne Resource Description System (CORDS) to describe applications, services and cloud resources. There is an external module named service level agreement manager which oversees all negotiated SLA, The SLA instances are described according to the WS Agreement standard (Yangui, Marshall, Laisne & Tata, 2014).
15. The Multi-Cloud Provisioning and Load Distribution for Three-Tier Applications approach described by Grozev and Buyya (2014) is intended for adaptive, dynamic and reactive resource provisioning. It provides load distribution algorithms to optimize cost and response delays without violating legislative and regulatory requirements. On top of the standard three-tier architectural pattern, *i.e.*, Data, Domain and Data Centre layers, it includes two additional layers – Entry Point Layer and Data Centre Control Layer – that enable legislation-compliant user routing to eligible clouds as well as manage the incoming workload and the dynamic provision of resources, without relying on advances in SLA specifications, *i.e.*, novel SLA formalisms. It takes into account the application regulatory requirements when selecting a data centre site and considers the cost as well as the degree of utilization of the employed resources within the data centre.
16. CloudAnchor offers trust-based discovery, negotiation and coalition establishment services regarding single and federated resources. It implements a business model where brokerage (bSLA), coalition (cSLA) and resource (rSLA) agreements and their dependencies are contemplated and builds trust based peer models to support provider invitation/acceptance, coalition establishment and resource negotiation. The adopted multi-agent paradigm allows businesses to enter/leave freely the platform, retain their autonomy and privacy as well as cooperate or compete to get/place resources.

Table 2 displays the comparison of the selected brokerage platforms in terms of services offered, namely service discovery, SLA negotiation and enforcement, type of resources transacted and adopted peer modelling approach. All platforms include service discovery, 94 % provide SLA negotiation and enforcement services regarding single and federated resources and 50% build and use peer models.

CloudAnchor, when compared with other platforms, implements two distinctive features: (*i*) a business model contemplating the negotiation, establishment and termination of brokerage, coalition and resource agreements; and (*ii*) a decentralised trust model of the peers, taking into account all past interactions, including provider invitation, resource negotiation and resource provision, and not just those regarding resource provision. The latter corresponds to the feedback from SLA monitoring and enforcement, which in our case is implemented by an external module. This novel approach allows consumers to invite providers based on the outcomes from past provider invitations and resource provisions as well as helps providers to decide whether or not to accept a consumer's invitation for negotiation based on the outcome of their common resource negotiation and resource provision history.

Table 2. Comparison of cloud brokerage platforms

Platform	Service Discovery	Service Negotiation	Service Enforcement	Peer Model	Resources
Stantchev *et al.*, 2009	Yes	Yes	Yes	Trust	Single
Mahbub *et al.*, 2010	Yes	Yes	No		Single & Federated
Celesti *et al.*, 2010	Yes	No	Yes		Single & Federated
Venticinque *et al.*, 2011	Yes	Yes	No	Trust	Single & Federated
Ferrer *et al.*, 2012	Yes	Yes	Yes	Trust	Single & Federated
Pawluk *et al.*, 2012	Yes	Yes	No		Single & Federated
Tordsson *et al.*, 2012	Yes	No	Yes		Single & Federated
Amato *et al.*, 2012	Yes	Yes	Yes	Reputation	Single
Cuomo *et al.*, 2013	Yes	Yes	Yes		Single & Federated
Cascella *et al.*, 2013	Yes	Yes	Yes	Reputation	Single & Federated
Al Falasi *et al.*, 2013	No	Yes	Yes	Trust	Single & Federated
Pawar *et al.*, 2014	Yes	Yes	No	Trust	Single & Federated
Kertész *et al.*, 2014	Yes	Yes	No		Single
Yangui *et al.*, 2014	Yes	Yes	Yes		Single
Grozev *et al.*, 2014	Yes	Yes	Yes		Single & Federated
CloudAnchor	Yes	Yes	No	Trust	Single & Federated

Federated Cloud Resources

Federated resources, in the context of this work, are sets of decomposable resources, *i.e.*, which can be broken into a collection of standard packages, serviced by multiple providers. The provision of such services suffers from the lack of interoperability and standardised interfaces between provider solutions and requires a versatile middleware capable of addressing the dynamic, distributed, decentralised nature of the resource brokerage problem. The adoption of a common abstraction layer minimises the lack of interoperability and standardisation in the application domain at the cost of customisation. On the one hand, abstraction layers offer a unique Application Programming Interface (API) and return single resource endpoints and, on the other hand, do not support the level of customisation offered by standalone provider solutions. In terms of the middleware and given the nature of the cloud resource brokerage problem at hand, the adoption of a multi-agent brokerage platform is the most appropriate solution.

Cloud service brokers play an essential role in this context, since they facilitate partner lookup, resource discovery and abstract the management and integration of disparate cloud services – potentially across different providers – allowing SME to migrate seamlessly to the cloud while meeting their unique requirements. However, at the cloud broker level, there are also several issues to be tackled: (*i*) the adoption of specifications and standards to represent SLA instances, *e.g.*, WS-Agreement (OGF) or WSLA (IBM); (*ii*) the definition of a standard model to represent cloud resources; (*iii*) the definition of a security mechanism to protect the negotiation strategies of each cloud provider as well as business information; and (*iv*) the provision of the federated resources. In the latter case, while existing approaches undertake one-to-one negotiations and merge the results into a single resource package in the end, our approach creates coalitions of providers – virtual cloud providers – which negotiate directly with the consumers.

CLOUDANCHOR

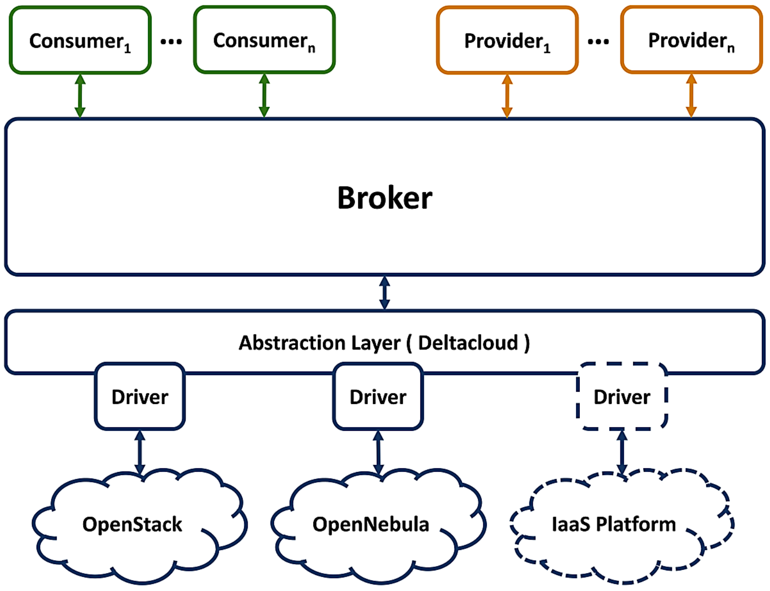
CloudAnchor was a 24 month research, development and innovation (RD&I) project funded by the Portuguese Agency for Innovation. The main focus of the project was the development of a modular platform for connecting and managing federations of IaaS cloud service providers. It integrates, among other components, a brokerage platform, including an automated partner lookup and resource negotiation mechanism on behalf of the stakeholders, and an abstraction framework, interfacing with different IaaS platforms. The CloudAnchor architecture presented in Figure 1 is composed of the broker layer (the CloudAnchor brokerage platform), the abstraction layer (Deltacloud framework) and the different IaaS provider platforms.

This approach offers a single API for the broker to manage and interact with the different underlying cloud computing and storage resource platforms' interfaces, acting as an abstraction layer. The IaaS resources are virtualized computing elements with standardised packages, *e.g.*, a standard Virtual Machine (VM) comprising four virtual Central Processing Units (CPU), 16 GiB of Random Access Memory (RAM) and an Operating System (OS) image.

Abstraction Layer

The goal of the abstraction layer, which is composed by the Deltacloud framework, is to overcome the lack of interoperability, vendor lock-in, terminology issues and distinct authentication methods, while exposing a common API to manage resources across different IaaS clouds. There are other cloud abstrac-

Figure 1. CloudAnchor approach



tion solutions *e.g.*, jClouds and Libcloud that are standard programming libraries and, unlike Deltacloud, do not integrate additional development tools. Thus, Deltacloud exposes broadly used interface libraries, Cloud Infrastructure Management Interface (CIMI) API and AWS EC2 API, making the overall development project more complete and also provide documentation for the development of new drivers to integrate unsupported IaaS platforms. Meireles and Malheiro (2014) provide further details on the selection, adoption and integration of several IaaS platforms in the Deltacloud framework.

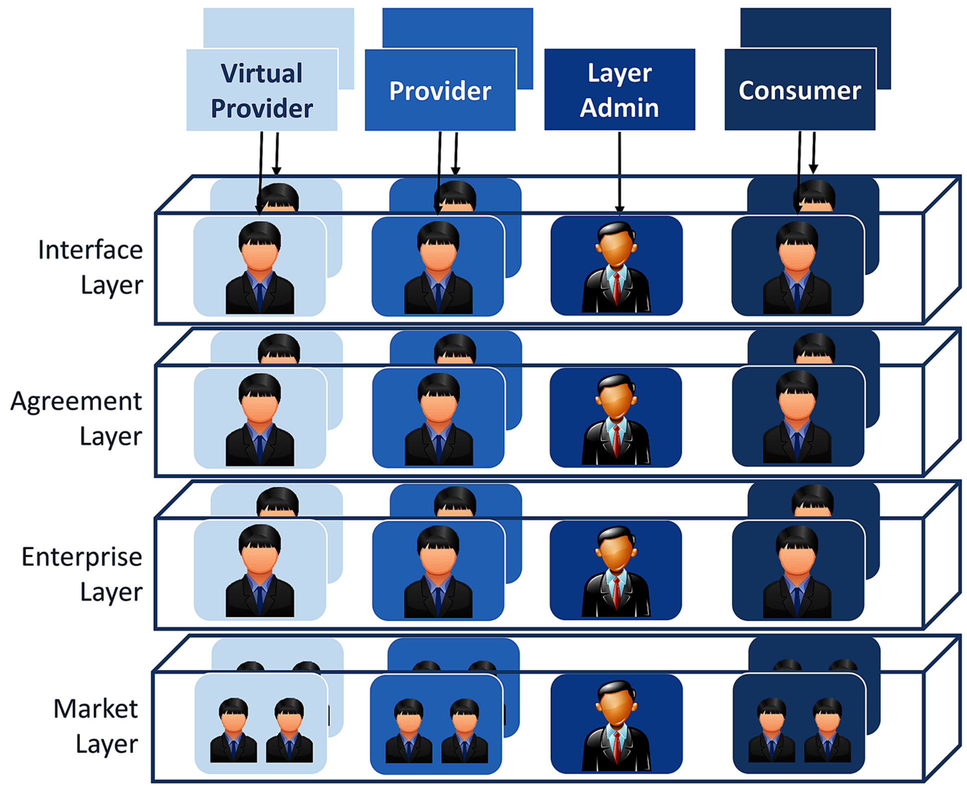
Deltacloud is composed of three main layers: (*i*) Driver's Layer; (*ii*) Core Layer; and (*iii*) User Interface Layer. The Driver's Layer contains the individual drivers to interact with the specific back-end IaaS platforms and process the HTTP requests/responses. It uses external cloud clients to interact with the corresponding IaaS platform API and libraries with methods to perform API calls that comply with the IaaS API. The Core Layer is composed of modules and classes that support the remaining two layers. It defines the group of operations (also called collections) and the features that each API currently exposes and that each driver supports, and maps them to the syntax that is exposed by the User Interface Layer. The User Interface Layer is responsible for the exposure of the Deltacloud Web services. Deltacloud integrates a REST API and a graphical user interface Web dashboard as well as a Distributed Management Task Force (DMTF) open standard CIMI REST API and the Amazon Web Services (AWS) EC2 API. By default, Deltacloud provides drivers for multiple IaaS cloud providers and platforms including the open-source OpenNebula and OpenStack. The list of supported cloud platforms is available at the official project page (Apache DeltaCloud, 2013).

Broker Layer

The brokerage platform architecture is presented in Figure 2. It is organized in interface, agreement, enterprise and market layers and comprises of five types of specialised dedicated agents: (*i*) interface agents to interact with consumer and provider SME businesses; (*ii*) agreement agents to manage SLA instances; (*iii*) enterprise agents to model consumer and provider SME businesses; (*iv*) market delegate agents to negotiate specific resources on behalf of consumer and provider SME businesses; and (*v*) layer manager agents (interface, agreement, enterprise and market layer agents) responsible for the management of the corresponding layers. Each business (consumer or provider SME) is represented in the platform by the corresponding: (*i*) interface agent located in interface layer; (*ii*) agreement agent located in agreement layer; (*iii*) enterprise agent in the enterprise layer; and (*iv*) an undetermined number of delegate agents involved in specific resource negotiations in the market layer. These agents are identified by a trading code, preventing third parties from intruding in undergoing negotiations (Veloso, Malheiro & Burguillo, 2015).

In terms of standard technologies, the platform adopts: (*i*) standard REST WS interfaces for the interaction with provider and consumer SME businesses (via the dedicated interface agents); (*ii*) BPMN to define the provider and consumer behaviour within the platform; (*iii*) standard specifications for the representation of Service Level Agreements (SLA); and (*iv*) standard SLA negotiation protocols. The platform is open (*i.e.*, providers and consumers can register and deregister at will), modular, (*i.e.*, functionalities can be added and removed at will), and scalable (*i.e.*, the platform can be distributed over multiple computing nodes).

Figure 2. CloudAnchor approach



Brokerage Platform Workflow

The brokerage workflow encompasses six stages; (*i*) business registration, negotiation and establishment of the bSLA; (*ii*) consumer resource request; (*iii*) provider lookup and invitation; (*iv*) negotiation and establishment of the rSLA and provision of the resource endpoint to the consumer; (*v*) reputation update based on the SLA enforcement feedback data; and (*vi*) creation of virtual providers to supply federated resources.

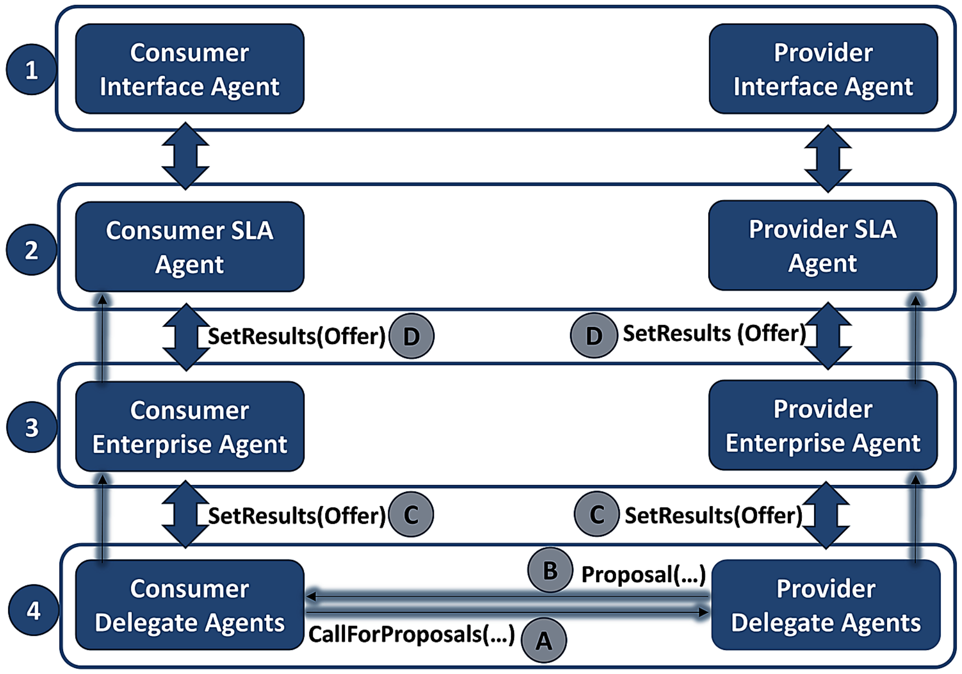
In order to join the platform, a consumer or provider SME first needs to register with the interface administrator agent. Once registered, the interface layer administrator creates a dedicated interface agent to interact with the business and the agreement administrator creates a dedicated agreement agent to maintain the business SLA templates and instances. The first task of the agreement agent is to establish a bSLA with the platform on behalf of the business using the provided negotiation strategy according to the WS-Agreement one shot protocol. In the case of a consumer, the bSLA specifies that the consumer must invite all providers for negotiation with resources matching their future needs. In the case of a provider, the bSLA states that it must accept all resource negotiation invitations issued by potential partner consumers. The business agreement agent launches the two-step bSLA negotiation process: (*i*) obtains the SLA template; and (*ii*) negotiates the platform services fruition conditions, using the single proposal WS-Agreement Negotiation protocol. Upon success, the business completes its registration in the platform and is ready to take advantage of the platform brokerage services.

Every time a provider uploads a new resource, the corresponding agreement agent creates an rSLA template, which is an instance of a WS-Agreement template. The negotiation strategy of any resource is stored in the corresponding enterprise agents. Whenever a provider accepts an invitation to negotiate one of its resources, the provider and consumer launch ephemeral delegate provider and consumer agents to negotiate the terms of the rSLA. The goal of a provider delegate is to successfully negotiate the resource for the highest price. The goal of a consumer delegate is to negotiate the provision of a resource matching its needs at the lowest price. A celebrated rSLA not only specifies the resource provision terms, but implies that the consumers must share the relevant resource features as well as feedback data with the provider. Once the rSLA is established, the brokerage platform obtains the resource endpoint via the abstraction layer (Deltacloud). Figure 3 illustrates the negotiation and establishment of an rSLA between a consumer and a provider.

This process is conducted in three steps; (*i*) the consumer and provider delegates negotiate using FICNIP and the consumer delegate chooses the proposal with highest utility (represented by A and B in Figure 3); (*ii*) the outcome is communicated to the enterprise and agreement agents (represented by C and D in Figure 3); and (*iii*) the chosen provider and consumer agreement agents establish the rSLA according to the negotiated terms.

The creation of virtual providers is event-driven. The platform attempts to create a virtual provider based on a consumer request, *i.e.*, whenever the consumer requires a decomposable resource that cannot be provided by single providers. The platform, in turn, invites all providers with compatible, but insufficient resources to establish a virtual provider coalition in order to fulfil the consumer request. All invited providers take part in a negotiation to define the set of providers and resources to be committed to the virtual provider. The creation of a virtual provider consists of the creation of the corresponding interface,

Figure 3. rSLA negotiation



agreement and enterprise agents. Although the operation of a virtual provider coalition is identical to a single provider, once a federated resource is successfully negotiated, the brokerage platform obtains the corresponding collection of resource endpoints via the abstraction layer (Deltacloud).

The SLA enforcement feedback data feeds the distributed reputation system. Each provider and consumer builds a local reputation model of its business counterparts. This reputation model will condition any future negotiations with these counterparts.

Service Level Agreements

The platform implements two OGF specifications: (*i*) the WS-Agreement, used to represent all SLA templates used and to support the one shot bSLA negotiation; and (*ii*) WS-Agreement Negotiation, used to support the multi-round rSLA negotiation. These specifications are, in our case, implemented on top of the FIPA Agent Communication Language (ACL) in order to minimise the communication latency between parties. Additionally, to include a multi-round negotiation mechanism within the WS-Agreement Negotiation Protocol, we added FICNIP. Since the structure of a WS-Agreement template is highly extensible, bSLA, cSLA and rSLA are instances of WS-Agreement templates as displayed in Figure 4.

The SLA template presented in Figure 5 refers to an rSLA and includes context (validity, parties, etc.), terms (service terms and guarantee terms) and constraint fields. The service terms specify the service functionalities and the guarantee terms stipulate the service obligations and penalties. The constraints define the acceptable parameter negotiation range. All SLA instances are negotiated according to the negotiation strategy defined by the corresponding business.

The lifespan (validity) of a bSLA is typically longer than that of an rSLA. Figure 6 illustrates these different SLA lifecycles. In fact, multiple rSLA, involving a given business, can be celebrated, applied and terminated within the scope of the same bSLA. The arrows between the rSLA and the bSLA life cycles represent the feedback provided by the rSLA instances to the consumer and provider bSLA monitoring modules. When an rSLA expires, the provider pays the platform for the brokerage service, and the distributor reimburses the platform and the producer for the brokerage and the resource provisioning services, respectively. The lifecycle of a cSLA – which corresponds to the lifespan of a virtual provider – is identical to the corresponding bSLA. The main difference between the lifespan of a virtual and an individual provider is that the former tends to have a shorter life than the latter.

Figure 4. SLA template hierachy

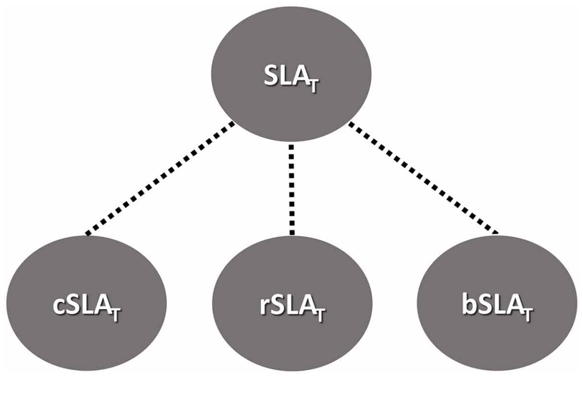


Figure 5. rSLA template

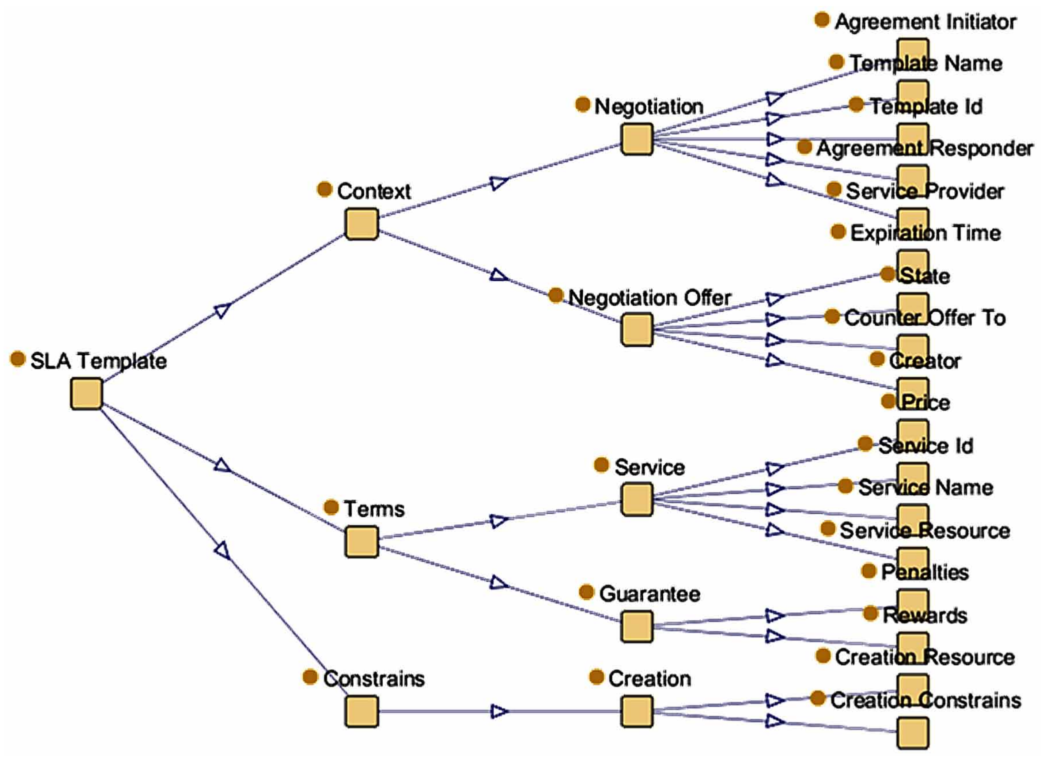
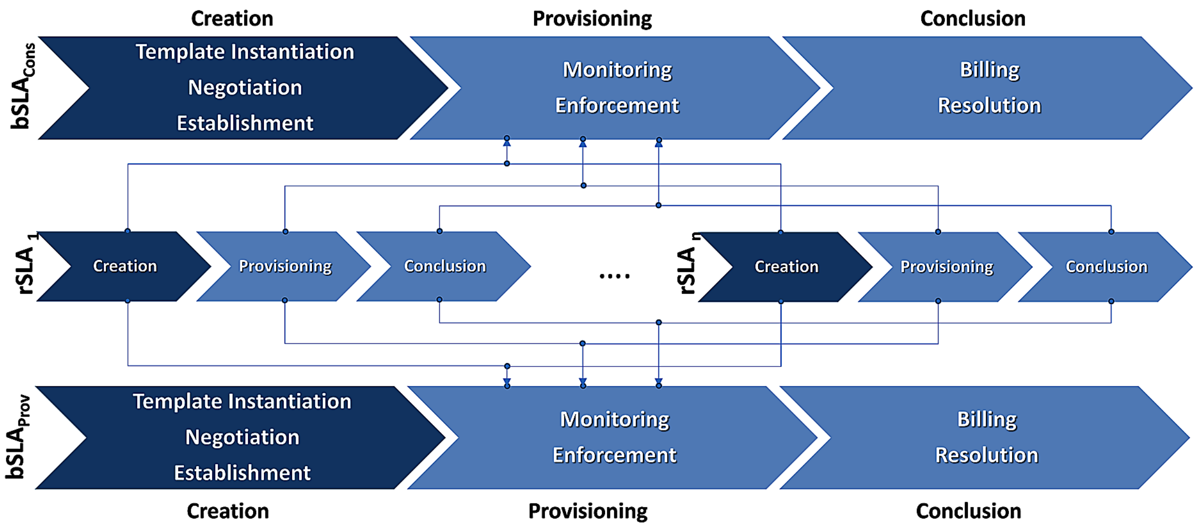


Figure 6. SLA lifecycle



At the business level, the platform adopts a pay-per-use model together with the hierarchical SLA model illustrated in Figure 7.

Under this hierarchical model, bSLA instances define the brokerage service provisioning terms under which all cSLA and rSLA are established. The cSLA instances specify the federated resource provisioning terms under which the individual federated resources (rSLA) are provided. The rSLA regarding a provider resource inherits the terms of the bSLA established between the platform and the provider, and the rSLA of a federated resource inherits the terms of the bSLA established between the platform and the federated providers as well as the terms of the cSLA celebrated between the federated providers. The celebrated rSLA remain latent until the resources are actually provisioned, *i.e.*, when consumers use the service/resources. Only then, the corresponding rSLA instances are activated. Figure 8 illustrates the provision of Resource_01 by Provider_01 to Consumer_01, which implies fulfilling the terms of two bSLA and one rSLA.

Figure 7. SLA instance hierarchy

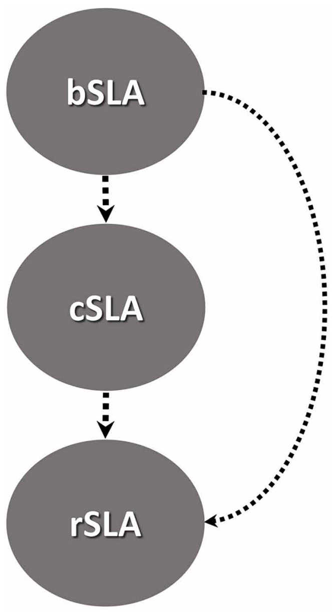


Figure 8. Single provider rSLA

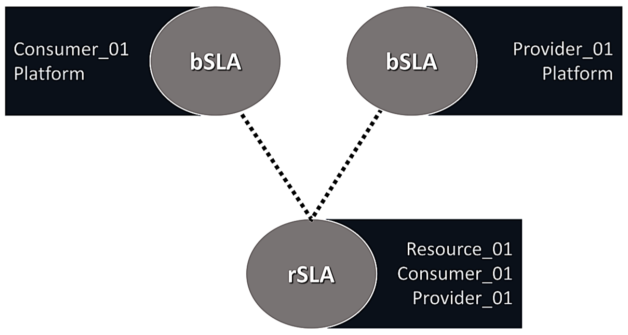


Figure 9. represents, in terms of SLA instances, the provision of Resource_01 by Virtual_Provider_01 to Consumer_01. On the one hand, it involves fulfilling the terms of one bSLA and one cSLA and, on the other hand, since the cSLA is composed by three rSLA from three providers, it, additionally, has to fulfil the terms of the corresponding bSLA.

The provision of a resource results in several payments: (*i*) the consumer pays the provider the established resource provisioning fee; (*ii*) the provider pays to the platform of the accorded brokerage fee; and (*iii*) the consumer pays the platform of the negotiated brokerage fee. In the case of a federated resource, there is an additional brokerage fee that the federated providers pay the platform. These fees are typically distinct. Table 3 and Table 4 illustrate the payment fees involved in the case of a single and a virtual resource provider, respectively.

SLA Negotiation

The platform negotiates the establishment of brokerage, resource and coalition SLA. A business wishing to join and benefit from the platform services needs first to establish a bSLA with the platform. The bSLA negotiation strategy is stored in the corresponding business agreement agent and the negotiation with the platform adopts the WS-Agreement one shot protocol. In the case of a consumer, the bSLA implies that it must invite all trusted providers for negotiation with resources matching its needs (IaaS resource profiles). The provider bSLA implies that it must accept all resource negotiation invitations issued by

Figure 9. Virtual provider rSLA

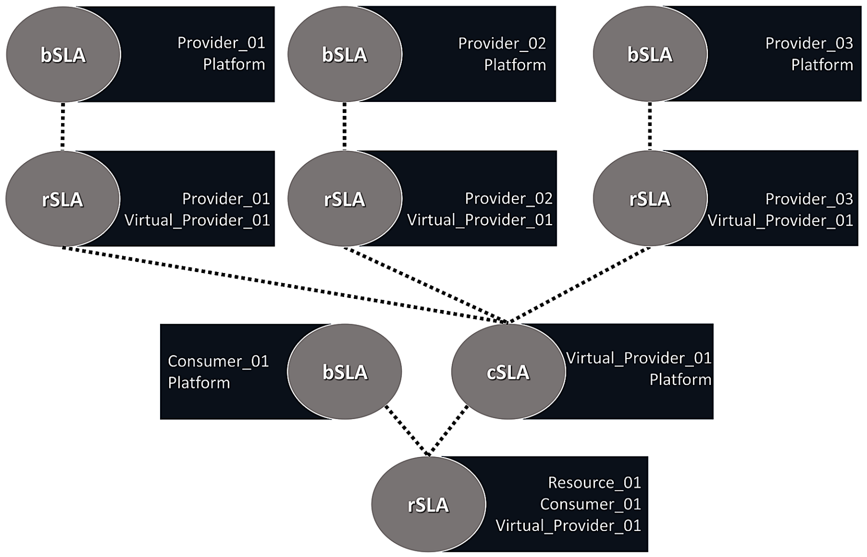


Table 3. Single provider resource service fees

		Platform
	Provider	bSLA Fee
Consumer	rSLA Fee	bSLA Fee

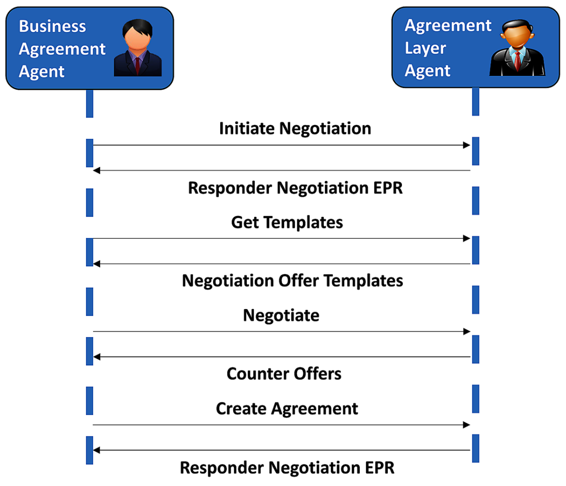
Table 4. Virtual provider resource service fees

					Platform
				Provider$_k$	bSLA Fee
			Provider$_j$		bSLA Fee
		Provider$_i$			bSLA Fee
	Virtual Provider	rSLA Fee	rSLA Fee	rSLA Fee	bSLA Fee
Consumer	rSLA Fee				bSLA Fee

trusted consumers. For the platform, the bSLA stipulates the level of the brokerage service. A provider may reject an invitation and a consumer may not invite a provider for negotiation if they are untrusted. For a given consumer or provider business, a peer is untrustworthy when its local trustworthiness is below the average local trust of the set of peers under consideration.

During registration, the business agreement agent launches the two-step bSLA negotiation process: (*i*) obtains the SLA template; and (*ii*) negotiates the conditions of the fruition of the platform services, using the single proposal WS-Agreement Negotiation protocol. Upon success, the business completes its registration in the platform and is ready to take advantage of the platform brokerage services. Figure 10 illustrates the negotiation of a bSLA, using the one shot negotiation protocol, where the business agent represents a consumer or a provider and the endpoint reference (EPR) corresponds to the business endpoint in the platform, *i.e.*, the corresponding business interface agent. This negotiation involves the business agreement agent, acting as the initiator and representative of the business, and the agreement layer agent, playing the role of respondent and representative of the platform. The business agreement agent sends gets a negotiation template, fills it with its proposal and sends the negotiation proposal to the platform agreement agent. The platform agent accepts the proposal, establishing the bSLA and registering the business in the platform, or sends a counter-offer. If the business agent rejects the counter-offer, the negotiation of the bSLA fails and the business registration aborts.

Figure 10. bSLA one-shot negotiation



On the one hand, every time a provider business generates a new resource offer event, the corresponding agreement agent creates an rSLA template instance. On the other hand, when a consumer business uploads a new resource request, it triggers the provider lookup and invitation, resulting in the invitation of the set of providers with compatible resources. In the case of the trust-based invitation, the consumer invites the sub-set of providers with trustworthiness higher or equal to the mean trustworthiness of the set of providers with compatible resources. A provider, by default, accepts all invitations as long as it has resources. In the case of trust-based acceptance, the provider accepts invitations from consumers with trustworthiness higher or equal to the mean trustworthiness of the set of known consumers. Alternatively, the provider business can define a threshold. Whenever the provider accepts a consumer invitation and selects this resource for negotiation, the provider and consumer launch ephemeral delegate provider and consumer agents to negotiate the terms of the provision of the resource (rSLA). The goal of a provider delegate is to negotiate the resource successfully. The goal of a consumer delegate is to negotiate the terms of usage and ensure that the resource under negotiation matches the profile of the required IaaS resource. The provisioning of the resource implies the payment off the fees described in the previous section. Virtual providers, which are coalitions of single providers, offer federated resources. The negotiation of a federated resource is identical to the negotiation of a single provider resource. Renegotiation of long lasting bSLA and cSLA can be performed periodically or on request from one of the parties involved.

Each consumer or provider may participate in multiple trading rooms by launching multiple delegate agents, *i.e.*, a provider business can negotiate multiple resources simultaneously. Each delegate agent is identified by a universally unique identifier (UUID) code generated when the provider is invited to negotiate. The negotiation protocol is the FICNIP, where the consumer delegate and provider delegates play the roles of the initiator and participant agents, respectively. At the end of each iterative multi-round, the consumer delegate assesses the set of received provider delegate proposals, accepts the one with highest utility and rejects the remaining others or rejects all proposals. When the consumer accepts a provider's proposal, it corresponds to the establishment of the corresponding rSLA, which additionally implies that the consumer must share feedback data with the provider and that the provider must make the resource available for use in the accorded terms. The consumer delegate agent assesses provider proposals through the resource utility function presented in Equation 1:

$$Utility(c,p,r) = Trust_{ENF}(c,p)\left(\alpha_c\left(1 - price(p,r)\right) + \left(1-\alpha_c\right)time(p,r)\right) \tag{1}$$

where $Utility(c,p,r)$ represents the utility that the consumer c attributes to the resource r proposed by provider p, $Trust_{ENF}(c,p)$ is the trustworthiness that consumer c attributes to provider p based on the ratio of fulfilled SLA, $price(p,r)$ is the normalise resource price proposed by provider p, $time(p,r)$ is the proposed resource uptime and, finally, α_c is the weight attributed to the price by consumer c.

Whenever a consumer c requires a decomposable resource which cannot be supplied single-handedly by any provider, it requests to the platform the creation of a virtual provider (vp). The platform attempts to create a virtual provider on behalf of the consumer capable of providing the requested resource. First, it creates a potential virtual provider agent, which looks up and invites trustworthy provider agents holding resources of the requested type to negotiate a cSLA, *i.e.*, the terms of a joint commitment to provide federated resources. The cSLA negotiation, which occurs in the market layer between delegate agents of the virtual provider and of the invited providers, follows the FICNIP protocol. During the negotiation, the providers offer to commit a given number of standard resource packages to the coalition under the specified terms. In the end, the virtual provider agent assesses the utility of the different provider offers through Equation 1. $Trust_{ENF}\left(vp, p\right)$, which is the SLA-based trustworthiness the virtual provider *vp* attributes to provider *p*, corresponds to $Trust_{ENF}\left(c, p\right)$, *i.e.*, the SLA-based trustworthiness that consumer *c* attributes to provider *p*. The negotiation succeeds if the resulting federated resource package fulfils the initial consumer request and fails otherwise.

Finally, the brokerage platform interacts with the abstraction layer to provide the negotiated IaaS resources. The interaction with the brokerage platform is conducted via the Deltacloud API. The brokerage platform is limited to the IaaS platforms supported by Deltacloud.

Decentralised Trust Model

The platform creates and maintains a distributed decentralised trust model of all business counterparts (platform, providers and consumers) based on their past behaviour, *i.e.*, the number of invitations, acceptances as well as the number of negotiated, established and fulfilled SLA. Each entity (platform, consumer, provider and virtual provider) builds and maintains a model of its counterparts. While consumers, providers and virtual providers hold a partial incomplete trust model of their counterparts, the platform is in a unique position as it keeps a global and complete trust model.

The trust model supports the invitation, acceptance and negotiation stages and is based on the past interaction data, including the number of invitations *versus* acceptances to negotiate, the number of SLA negotiated *versus* SLA established and the number of SLA established *versus* SLA fulfilled. In the future, it will also support bSLA and cSLA renegotiation stage. Whereas the number of invitations *versus* acceptances and negotiated *versus* established SLA is known by each partner, the SLA enforcement, which is based on the QoS and the payment of service (PoS), is communicated by an external SLA monitoring and enforcement module. This feedback is received when a bSLA, cSLA or rSLA terminates and is maintained by the involved entities. In the case of a bSLA, it is maintained by the platform and the consumer, the platform and the provider or the platform and the virtual provider; in the case of a cSLA, is kept by the virtual provider and the set of providers; and, in the case of an rSLA, by the consumer and the provider or the consumer, the virtual provider and the set of providers. Internally, the local trust model is kept by the corresponding agreement agents. Each entity determines the trustworthiness of a counterpart, *i.e.*, a business partner, based on this data. The trustworthiness varies between 0% (untrustworthy) and 100% (trustworthy) and, by default, a new business partner is 100% trusted. The trust model supports the invitation (*INV*), SLA negotiation (*NEG*) and SLA enforcement (*ENF*) stages. Equation 2 represents the dynamic trustworthiness attributed by a trustor *a* to a trustee *b* in stage *S* after concluding *n* interactions:

$$Trust_S\left(a,b\right)_n = \frac{n-1}{n} Trust_S\left(a,b\right)_{n-1} + \frac{1}{n} Out_{S,n} \tag{2}$$

where $Out_{S,n}$ is the Boolean outcome of the last interaction within stage *S*: success (1) or failure (0).

The trustworthiness of the peers is updated every time new data is available and has three components: the ratio of accepted invitations between a consumer and a provider $\left(Trust_{INV}\left(c, p\right)_n\right)$, the ratio of established SLA between a provider and a consumer $\left(Trust_{NEG}\left(p,c\right)_n\right)$ and the ratio of fulfilled SLA $\left(Trust_{ENF}\left(c,p\right)_n\right)$ and $\left(Trust_{ENF}\left(p,c\right)_n\right)$.

In the invitation stage, a consumer invites providers to negotiate a resource based on their published resource availability and local trustworthinesses. The providers are filtered by the ratio of fulfilled SLA $\left(Trust_{ENF}\left(c,p\right)_n\right)$ and the ratio of accepted invitations $\left(Trust_{INV}\left(c,p\right)_n\right)$. Identically, a provider accepts a consumer invitation to negotiate a resource based on its resource availability and on the consumer's local trustworthiness. In this case, consumers are selected according to the ratio of fulfilled SLA $\left(Trust_{ENF}\left(p,c\right)_n\right)$ followed by the ratio of established SLA $\left(Trust_{NEG}\left(p,c\right)_n\right)$. A consumer chooses a provider based on its proposal and local trustworthiness by applying the resource utility function presented in Equation 1, which takes into account the ratio of fulfilled SLA $\left(Trust_{ENF}\left(c,p\right)_n\right)$.

TESTS AND RESULTS

We performed several tests to verify the correct operation of the platform in terms of: (*i*) single resource provision; (*ii*) federated resource provision; and (*iii*) mixed resource provision. In these experiments, according to the default SLA terms, if a business fails to fulfil an established SLA, it reimburses the partner. At start up consumers and providers are equally and fully trusted (100% trustworthiness). During these tests, consumers remain fully trusted, while providers experience changes in trustworthiness, *i.e.*, do not fulfil all established SLA. These experiments were performed with and without the application of the decentralised trust model, and using equal and diverse provider price adaptation strategies for the generation of negotiation proposals.

Providers can implement diverse price adaptation strategies. In the single price (SP) policy all providers adopt the same strategy. It corresponds to a linear descending price adaptation strategy with a maximum value of 47 € per standard VM, a reference value of 37 € per standard VM and a minimum value of 27 € per standard VM package. The providers start by issuing a proposal with the reference price and will, subsequently, lower the price of the next proposal if their previous proposal is rejected. In the multiple price (MP) policy scenario, providers adopt different price policies (linear, quadratic and exponential descending as well as random and static). The descending price strategies start at the reference price, the random strategy varies between the maximum and minimum prices and the static strategy.

In terms of hardware, the tests were executed on a platform with one quad-core i7-2600 3.40 GHz Central Processing Unit (CPU) with 2 threads per core, 15 GiB Random Access Memory (RAM) and a 1.8 TiB of storage capacity.

Table 5 displays the six configuration settings used in the single, federated and mixed resource provision experiments.

The focus will be on the multiple price scenarios rather than on the single price scenarios. The single price scenarios have been included for completeness and comparison.

Single Resources

The first set of tests involves one consumer and fifty providers holding twenty standard virtual machine packages (VM) each, *i.e.*, the offer meets demand. The consumer fulfils all rSLA and each provider

Table 5. Test configuration

Test	Price Strategy	Trust-based invitation	Trust-based negotiation
SP	Single	No	No
MP	Multiple	No	No
SP+TN	Single	No	Yes
MP+TN	Multiple	No	Yes
SP+TIN	Single	Yes	Yes
MP+TIN	Multiple	Yes	Yes

will fail to provide 5 out of the 20 VM with the negotiated QoS, ending with a trustworthiness of 75% and obtaining revenues which depend on the price adaption strategy adopted. The goal is to assess the impact of the provider trustworthiness in the invitation and negotiation stages. Each consumer request is provided by a single provider. Table 6 presents the results.

This first set of test involves one thousand resource negotiations. The multiple price results show that trust-based invitation and negotiation decreases the runtime by 51% and increases the average VM price by 1. The slight difference in the average resource price in the multiple price strategy (1%) results from the fact that providers are being selected according to their trustworthiness, and not to the price of their proposal. This first test is the base test for all future comparisons.

In the following set of tests, the only change is that the offer exceeds the demand. It involves one consumer and hundred providers holding twenty standard virtual machine packages (VM) each, *i.e.*, the offer doubles demand. Table 7 holds the results of this second test.

When compared with the first set of tests, the average VM price decreases by 12% in the SP case and by 6% in the MP_TIN as expected. The test involved a thousand negotiations, but now, in this oversupply scenario, the number of provider agents per negotiation is considerably higher than the number of the first test, and the runtime increases 215%. Moreover, the results indicate that, in this oversupply scenario, the trust-based negotiation runtime increases by 5% and, in the trust-based invitation and negotiation case (MP+TIN), decreases by 48%. This increase in runtime in the case of trust-based negotiation results from the fact that, in the trust-based negotiation, not only the consumer needs the provider trustworthi-

Table 6. Single resources: offer meets demand

Test	SP	MP	SP+TN	MP+TN	SP+TIN	MP+TIN
Consumers	1	1	1	1	1	1
Providers	50	50	50	50	50	50
Requests per consumer	1000	1000	1000	1000	1000	1000
Resources per consumer request (VM)	1	1	1	1	1	1
Resources available per provider (VM)	20	20	20	20	20	20
Provider rSLA violation (%)	25	25	25	25	25	25
Total Runtime (s)	268	206	209	208	105	106
Average VM price (€)	36.39	31.56	36.39	31.88	36.39	31.88

Table 7. Single resources: offer exceeds demand

Test	SP	MP	SP+TN	MP+TN	SP+TIN	MP+TIN
Consumers	1	1	1	1	1	1
Providers	100	100	100	100	100	100
Requests per consumer	1000	1000	1000	1000	1000	1000
Resources per consumer request (VM)	1	1	1	1	1	1
Resources available per provider (VM)	20	20	20	20	20	20
Provider rSLA violation (%)	25	25	25	25	25	25
Total runtime (s)	562	456	483	477	236	228
Average VM price (€)	36.39	27.71	36.39	30.51	36.39	30.02

ness to assess each provider proposal, but he is negotiating with all potential providers, including the ones with low trustworthiness.

Federated Resources

The following set of tests focusses on federated resource provision. The setup involves one consumer and fifty providers, holding 20 VM each. The consumer requests 40 packages of 25 VM sequentially. Since no single provider can provide this number of resources, it forces the establishment of provider coalitions. The goal is to assess the impact of the provider trustworthiness in the invitation, coalition establishment and negotiation stages. Each consumer request is provided by a coalition of providers. Table 8 holds the results.

This set of tests encompasses 40 cSLA negotiations, 40 virtual provider bSLA negotiations and 40 federated resource rSLA negotiations. For the multiple price strategies, the average VM price is identical in the three cases; in the case of trust-based negotiation (MP+TN test), runtime increases 14% when compared to the unfiltered invitation and negotiation (MP test) and, in the case of trust-based invitation and negotiation (MP+TIN test), runtime remains unchanged when compared to the unfiltered invitation and negotiation (MP test). This increase in runtime in the case of trust-based negotiation results from

Table 8. Federated resources: offer meets demand

Test	SP	MP	SP+TN	MP+TN	SP+TIN	MP+TIN
Consumers	1	1	1	1	1	1
Providers	50	50	50	50	50	50
Requests per consumer	40	40	40	40	40	40
Resources per consumer request (VM)	25	25	25	25	25	25
Resources available per provider (VM)	20	20	20	20	20	20
Provider rSLA violation (%)	25	25	25	25	25	25
Total runtime (s)	38	29	38	33	36	29
Average VM price (€)	37.93	36.87	37.93	36.89	37.93	36.85

the fact that, in the trust-based negotiation, not only the consumer needs the provider trustworthiness to assess each provider proposal, but he is negotiating with all potential providers, including the ones with low trustworthiness. When compared with the corresponding single resource results, runtime decreases 73% and the average VM price increases 16%. Runtime improves because, although the total number of negotiated resources is the same, the number of negotiations drops from 1000 to 120. The average price increases because federated resources are more expensive.

In the second set of tests with federated resources the offer doubles the demand. These tests involve the same number of 120 negotiations. Table 9 summarises the results. When compared with the previous set of tests, runtime increases 234% in the MP test, 200% in the MP-TN and 230% in the MP+TIN test. The average VM price is equal in all cases and identical to the previous test results. These federated resource results show that oversupply affects significantly runtime opposed to the average VM price, which remains unchanged. The runtime increases because the number of agents involved in the establishment of the virtual providers is considerably large in the oversupply scenario than in the offer meets demand scenario.

Single and Federated Resources

The mixed resource provision tests contemplate the negotiation of single and federated resources. One consumer requests 40 sets of 25 VM sequentially, while the other requests 1000 VM sequentially. Table 10 presents the results.

This set of tests involves a total of 560 negotiations: 20 coalition negotiations, 20 brokerage negotiations, 20 federated resource negotiations and 500 single resource negotiations. Comparing the results of the MP+TIN test and of the MP test, there is a 32% decrease of the runtime and a 1% increase of the average VM price. The runtime decrease is due to the fact that the number of invited providers is smaller in the MP+TIN test.

The second test is identical to the previous one, but has twice the number of providers, reproducing an oversupply scenario. Table 11 holds the results.

Each test involves a total of 560 negotiations: 20 coalition negotiations, 20 brokerage negotiations, 20 federated resource negotiations and 500 single resource negotiations. When compared with the previous set of tests, these results show a significant increase in runtime and a slight decrease in average

Table 9. Federated resources: offer exceeds demand

Test	SP	MP	SP+TN	MP+TN	SP+TIN	MP+TIN
Consumers	1	1	1	1	1	1
Providers	100	100	100	100	100	100
Requests per consumer	40	40	40	40	40	40
Resources per consumer request (VM)	25	25	25	25	25	25
Resources available per provider (VM)	20	20	20	20	20	20
Provider rSLA violation (%)	25	25	25	25	25	25
Total runtime (s)	73	68	75	66	71	67
Average VM price (€)	37.93	36.39	37.93	36.39	37.93	36.39

Table 10. Mixed resources: offer meets demand

Test	SP	MP	SP+TN	MP+TN	SP+TIN	MP+TIN
Consumers	1	1	1	1	1	1
Providers	50	50	50	50	50	50
Requests per consumer	20	20	20	20	20	20
	500	500	500	500	500	500
Resources per consumer request (VM)	25	25	25	25	25	25
	1	1	1	1	1	1
Resources available per provider (VM)	20	20	20	20	20	20
Provider rSLA violation (%)	25	25	25	25	25	25
Total runtime (s)	131	104	121	104	101	71
Average VM price (€)	37.16	34.87	37.16	35.40	37.16	35.23

Table 11. Mixed resources: offer exceeds demand

Test	SP	MP	SP+TN	MP+TN	SP+TIN	MP+TIN
Consumers	1	1	1	1	1	1
Providers	100	100	100	100	100	100
Requests per provider	20	20	20	20	20	20
	500	500	500	500	500	500
Resources per consumer request (VM)	25	25	25	25	25	25
	1	1	1	1	1	1
Resources available per provider (VM)	20	20	20	20	20	20
Provider rSLA violation (%)	25	25	25	25	25	25
Total runtime (s)	298	271	287	255	232	145
Average VM price (€)	37.16	32.04	37.16	34.20	37.16	34.09

VM price, *e.g.,* in the case of MP+TIN, a runtime increase of 204% and an average VM price decrease of 3%. From the comparison of the MP+TIN and the MP results in Table 10, runtime decreases 46% and the average VM price increases 6%. These results are compatible with the selection of providers by trustworthiness and not by the price of their resources.

FUTURE RESEARCH DIRECTIONS

At the broker level and in order to improve the overall quality of the services provided, we plan: (*i*) to implement the renegotiation of bSLA and cSLA based on the trust model and on the success of the business in the platform and in the coalition, respectively; (*ii*) to refine the trust model using a sliding window rather than using the complete SLA enforcement history; (*iii*) to enrich SLA templates with new parameters to meet increasing business demands, *e.g.*, the time of provision in the case of rSLA;

and (*iv*) create virtual providers not only at the request of consumers, but also at the request of providers, *i.e.*, providers may, in order to try to increase their profits, decide to join efforts and provide federated resources.

At the abstraction layer, it is important to define conventions for the network advertisement of the provisioned resources, in particular the usage of a single domain name for resources that are created in a federated context and shared by multiple cloud providers. Currently, for the federated resources, there are as many endpoints as cloud providers involved. The goal would be to have an agreement between the parties to name the resources on behalf of the created virtual provider and, then, to change the Domain Name System (DNS) entries accordingly, providing a single endpoint for the federated resources.

Concerning the validation of the platform, we are performing experiments to assess the impact of the distributed trust model from the perspective of the providers, *i.e.*, when consumers violate agreements and, thus, exhibit different levels of trustworthiness.

CONCLUSION

This paper presents the CloudAnchor brokerage platform which offers brokerage services for SME cloud infrastructure vendors and SME cloud infrastructure consumers. It handles the negotiation and provision of infrastructure resources from single or colligated vendors (virtual providers). The brokerage services include the discovery of resource providers, the selection of providers via the negotiation of the resource provision terms, the establishment of the binding SLA with the brokerage platform, and the establishment of the resource SLA between cloud infrastructure consumer and vendor. When there are no single vendors capable of fulfilling a consumer request, the coalition mechanism is triggered with the goal to find providers able to deliver the required resources. These resources, which must be decomposable in standard VM packages, are then offered as federated resources.

In terms of contractual relationships, we identified and modelled three types: (*i*) bSLA – which defines the platform service provision terms for each business; (*ii*) rSLA – that specifies the resource provisioning terms between businesses; and (*iii*) cSLA – that details the platform coalition service provision terms between all businesses that form the federation. Any rSLA is celebrated under the scope of the involved business bSLA or cSLA, *i.e.*, they must fulfil the agreed brokerage and coalition service provision terms.

To improve the performance of the platform we designed a decentralised trust model. This model takes into account all past interactions, including provider invitation, resource negotiation and resource provision. The typical approach focusses on the resource provision outcome, *i.e.*, the feedback from SLA monitoring and enforcement, and disregards the outcomes from other interactions. Our approach allows consumers to rely on the outcomes from past provider invitations and resource provisions to invite providers as well as helps providers to decide whether or not to accept a consumer's invitation for negotiation based on the outcome of their common resource negotiation and resource provision history. In terms of contract negotiation, while bSLA adopts and implements the WS Agreement single round negotiation, the rSLA and cSLA implement the WS Agreement multi-round negotiation (FICNIP). In the case of rSLA and cSLA negotiations, the proposals are according to their utility, taking into account the resource price, the uptime and the trustworthiness of the provider. The subsumption of the individual rSLA terms by the terms of the involved bSLA or cSLA together with the three types of SLA are, as far as we know, a novel and relevant approach.

ACKNOWLEDGMENT

This work was partially financed by: (*i*) the ERDF – European Regional Development Fund through the COMPETE Programme (operational programme for competitiveness) and by National Funds through the FCT – Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) within project «23151 – CloudAnchor: Plataforma Modular de Interligação e Gestão de Federações de Fornecedores de Cloud Computing»; and (*ii*) the European Regional Development Fund (ERDF) and the Galician Regional Government under agreement for funding the Atlantic Research Center for Information and Communication Technologies (AtlantTIC).

REFERENCES

Al Falasi, A., Serhani, M. A., & Elnaffar, S. (2013). The sky: A social approach to clouds federation. *Procedia Computer Science*, *19*, 131–138. doi:10.1016/j.procs.2013.06.022

Alhamad, M., Dillon, T., & Chang, E. (2010). SLA-based trust model for cloud computing. *Proceedings of the 2010 13th International Conference on Network-Based Information Systems (NBiS)* (pp. 321-324). IEEE. doi:10.1109/NBiS.2010.67

Amato, A., Di Martino, B., & Venticinque, S. (2012). Evaluation and brokering of service level agreements for negotiation of cloud infrastructures. *Proceedings of the 2012 International Conference for Internet Technology and Secured Transactions* (pp. 144-149). IEEE.

Cascella, R. G., Blasi, L., Jegou, Y., Coppola, M., & Morin, C. (2013). Contrail: Distributed application deployment under SLA in federated heterogeneous clouds. In A. Galis & A. Gavras (Eds.), *The Future Internet (LNCS)* (Vol. 7858, pp. 91–103). Berlin, Germany: Springer. doi:10.1007/978-3-642-38082-2_8

Celesti, A., Tusa, F., Villari, M., & Puliafito, A. (2010). How to enhance cloud architectures to enable cross-federation. *Proceedings of the 2010 IEEE 3rd International Conference on Cloud Computing (CLOUD)* (pp. 337-345). IEEE. doi:10.1109/CLOUD.2010.46

Cuomo, A., Di Modica, G., Distefano, S., Puliafito, A., Rak, M., Tomarchio, O., & Villano, U. (2013). An SLA-based broker for cloud infrastructures. *Journal of Grid Computing*, *11*(1), 1–25. doi:10.1007/s10723-012-9241-4

Deltacloud Drivers. (2013, October). Apache Software Foundation. Retrieved from https://deltacloud.apache.org/drivers.html#drivers

Ferrer, A. J., Hernández, F., Tordsson, J., Elmroth, E., Ali-Eldin, A., Zsigri, C., & Sheridan, C. (2012). OPTIMIS: A holistic approach to cloud service provisioning. *Future Generation Computer Systems*, *28*(1), 66–77. doi:10.1016/j.future.2011.05.022

Fowley, F., Pahl, C., & Zhang, L. (2014). A comparison framework and review of service brokerage solutions for cloud architectures. In A. R. Lomuscio et al. (Eds), *Proceedings of the Service-Oriented Computing–ICSOC 2013 Workshop*s, LNCS (Vol. 8377, pp. 137-149). Cham, Switzerland: Springer International Publishing. doi:10.1007/978-3-319-06859-6_13

Grozev, N., & Buyya, R. (2014) Multi-Cloud Provisioning and Load Distribution for Three-Tier Applications. ACM Transactions on Autonomous and Adaptive Systems, 9(3), 13:1-13:21. New York, USA: ACM.

He, Q., Yan, J., Kowalczyk, R., Jin, H., & Yang, Y. (2007). An agent-based framework for service level agreement management. *Proceedings of the 11th International Conference on Computer Supported Cooperative Work in Design CSCWD '07* (pp. 412-417). IEEE. doi:10.1109/CSCWD.2007.4281471

Huynh, T. D., Jennings, N. R., & Shadbolt, N. R. (2006). An integrated trust and reputation model for open multi-agent systems. *Autonomous Agents and Multi-Agent Systems*, *13*(2), 119–154. doi:10.1007/s10458-005-6825-4

Kertész, A., Kecskemeti, G., & Brandic, I. (2014). An interoperable and self-adaptive approach for SLA-based service virtualization in heterogeneous Cloud environments. *Future Generation Computer Systems*, *32*, 54–68. doi:10.1016/j.future.2012.05.016

Mahbub, K., & Spanoudakis, G. (2010). Proactive SLA negotiation for service based systems. *Proceedings of the 2010 6th World Congress on Services (SERVICES-1)* (pp. 519-526). IEEE. doi:10.1109/SERVICES.2010.15

Meireles, F., & Malheiro, B. (2014). Integrated Management of IaaS Resources. In L. Lopes et al. (Eds.), Proceedings of the Euro-Par 2014: Parallel Processing Workshops (pp. 73-84). Cham, Switzerland: Springer International Publishing. doi:10.1007/978-3-319-14313-2_7

Pan, L. (2011). Towards a framework for automated service negotiation in cloud computing. *Proceedings of the 2011 IEEE International Conference on Cloud Computing and Intelligence Systems (CCIS)* (pp. 364-367). IEEE.

Pawar, P. S., Rajarajan, M., Dimitrakos, T., & Zisman, A. (2014). Trust Assessment Using Cloud Broker. In J. Zhou et al. (Eds.), *Trust Management VIII* (pp. 237–244). Berlin, Germany: Springer.

Pawluk, P., Simmons, B., Smit, M., Litoiu, M., & Mankovski, S. (2012). Introducing STRATOS: A cloud broker service. *Proceedings of the 2012 IEEE Fifth International Conference on Cloud Computing* (pp. 891-898). IEEE. doi:10.1109/CLOUD.2012.24

Sim, K. M. (2012). Agent-based cloud computing. *IEEE Transactions on* Services Computing, *5*(4), 564–577.

Stantchev, V., & Schröpfer, C. (2009). Negotiating and enforcing QoS and SLAs in grid and cloud computing. In N. Abdennadher & D. Petcu (Eds.), *Advances in grid and pervasive computing (LNCS)* (Vol. 5529, pp. 25–35). Berlin, Germany: Springer. doi:10.1007/978-3-642-01671-4_3

Teacy, W. L., Patel, J., Jennings, N. R., & Luck, M. (2006). Travos: Trust and reputation in the context of inaccurate information sources. *Autonomous Agents and Multi-Agent Systems*, *12*(2), 183–198. doi:10.1007/s10458-006-5952-x

Tordsson, J., Montero, R. S., Moreno-Vozmediano, R., & Llorente, I. M. (2012). Cloud brokering mechanisms for optimized placement of virtual machines across multiple providers. *Future Generation Computer Systems*, *28*(2), 358–367. doi:10.1016/j.future.2011.07.003

Veloso, B., Malheiro, B., & Burguillo, J. C. (2015). Media Brokerage: Agent-Based SLA Negotiation. In A. Rocha, A. M. Correia, S. Costanzo, & L. P. Reis (Eds.), *New Contributions in Information Systems and Technologies (Advances in Intelligent Systems and Computing)* (Vol. 353, pp. 575–584). Cham, Switzerland: Springer International Publishing.

Venticinque, S., Aversa, R., Di Martino, B., Rak, M., & Petcu, D. (2011). A cloud agency for SLA negotiation and management. In M. R, Guarracino et al. (Eds), Proceedings of the Euro-Par 2010 Parallel Processing Workshops, LNCS (Vol. 6585, pp. 587-594). Berlin, Germany: Springer. doi:10.1007/978-3-642-21878-1_72

Yangui, S., Marshall, I. J., Laisne, J. P., & Tata, S. (2014). CompatibleOne: The open source cloud broker. *Journal of Grid Computing*, *12*(1), 93–109. doi:10.1007/s10723-013-9285-0

Zulkernine, F., Martin, P., Craddock, C., & Wilson, K. (2009, July). A policy-based middleware for web services SLA negotiation. *Proceedings of the IEEE International Conference on Web Services ICWS '09.* (pp. 1043-1050). IEEE. doi:10.1109/ICWS.2009.157