J Grid Computing (2014) 12:321–345
DOI 10.1007/s10723-013-9290-3

Consuming Resources and Services from Multiple Clouds

From Terminology to Cloudware Support

Dana Petcu

Received: 15 December 2012 / Accepted: 15 December 2013 / Published online: 16 January 2014

Abstract The consumption of resources and services from multiple Clouds for reasons like high availability, cost reductions or special features is a natural evolution from in-silo Clouds. Several middleware are already available for multiple Clouds. However, due to the complexity of the technical solutions, their approaches are quite different and a classification is needed to guide the potential users. This paper looks to the reports on multiple Cloud topics and proposes a specific taxonomy. It identifies the ready-to-use software and services and classifies them according the taxonomy. It also underlines the driving needs and requirements from consumers' and providers' point of views. A particular Cloudware is provided as an example for the degree of requirements fulfillment.

Keywords Cloud computing · Multi-Cloud · Inter-Cloud

This work was partially supported by the grant of Romanian National Authority for Scientific Research, CNCS UEFISCDI, PN-II-ID-PCE-2011-3-0260 (AMICAS). It refers as case study to the platform developed in the frame of the grant of European Commission FP7-ICT-2009-5-256910 (mOSAIC), and provides a preliminary study on multiple Clouds for FP7-ICT 2011-8-318484 (MODAClouds).

D. Petcu (✉)
West University of Timişoara and Institute e-Austria, Timişoara, Romania
e-mail: petcu@info.uvt.ro

1 Introduction

The current Cloud technologies are designed according to the needs of service providers. The main expectation of the providers is that one size fits all needs. Therefore the service consumers are forced to adapt their applications to the available stack of software.

The usage of services and resources from multiple Clouds is push forward by the needs of their consumers or their providers. The sequentially usage of services from multiple Clouds is related to the migration from one Cloud to another driven by economic reasons (cost reductions, back-ups, emergencies, contract ending etc). The simultaneous usage of services from different Clouds can also have several benefits like high availability and fault tolerance, or cost reduction. In this context, a *first challenge of this paper is to identify the main forces that are driving the development of multiple Clouds technical solutions*. Various reasons for the use of services from multiple Clouds are discussed in the next section.

Currently, there are two basic delivery models in place for the multiple Clouds: Federations and Multi-Clouds. In the first case the Cloud providers are in agreement with each others to provide the Federation aiming to enhance the service offer to their service consumers, while in the second case, a third party is building unique entry point for mltiple Clouds, without a prior agreement with and between the Cloud providers. The delivery model of Cloud Federation is focusing mainly on infrastructure services (IaaS), and

is mostly implemented in the academic worlds where the agreements between the infrastructure providers are easier to be established compared with the commercial world. On the other hand, the Multi-Cloud model is more attractive for the commercial sector as not being intrusive for the Cloud providers, while is bringing an added value to the third party which is gathering the services. The interest in Multi-Cloud in the commercial world was raised no more than three years ago and the middleware that is now available is mainly focusing on providing unique entry points for various Clouds, targeting the sequential usage of Cloud services, rather that enabling the simultaneous usage of services from different Clouds. Strongly driven by the market forces, a new and advanced delivery model is started to be shaped: the Inter-Cloud, built on top of a Federation or a Multi-Cloud, and allowing Cloud service governance and blueprinting, as well as ad-hoc gathering of resources or self-adaptation.

With the growing interest in the subject of multiple Clouds, an artificial barrier is introduced: that of terminology. Beyond Federation, Multi-Cloud or Inter-Cloud, several other terms were coined to depict particular architectural solutions for multiple Clouds usage. We name here only few of them: Cloud-of-Clouds, Hybrid Cloud, Sky Computing, Aggregated Clouds, Hierarchical Clouds, Multi-tier Clouds, Cross-Cloud, Distributed Clouds, Cloud Merge, or SuperCloud. An immediate question which is raised looking to the long list is how these terms are related to each other and what needs are covered by each of them. The answer is unfortunately not straightforward as the youngness of the multiple Clouds field is going hand in hand with terms' fuzziness and lack of general agreement in what concerns these terms' definitions. Therefore, *a second challenge of this paper is to propose a schema for the terms relationships*.

Despite the fact that it practically emerged with the Cloud computing concept, the technical field of multiple Clouds is nowadays still in an infancy stage, mainly due to the diversity of the approaches of the concept implementation. The desired semi-automated guidance through the variety of the offers, based on monitoring tools for the quality of services, is not yet technically possible. The differences between the current APIs are hindering the easy composition or configuration of service to be consumed from multiple Clouds. In order to make the usage of services from multiple Clouds a reality, several other technical barriers should be overpassed too, like interoperability and portability, data and services mobility, middleware openness. Currently these barriers are focusing the interest of many researchers and the literature has plenty of innovative solutions for particular problems, while there is still a lack of middleware prototypes that can support a large number of scenarios of using services from multiple Clouds. In this context, *the third challenge of this paper is to identify the coverage of the various multiple Clouds requirements by existing software and services*.

Starting from the experience acquired in building Multi-Cloud middleware (see [66]), and looking to other current open-source or commercial middleware for multiple Clouds, we are going backward to identify the needs and requirements of the middleware customers (following the example of [68] for the single Clouds). Consequently, the *fourth challenge of this paper is to establish the main requirements of the Multi-Clouds in terms of basic principles, tools and services*. The most important feature that a middleware for multiple Cloud is supposed to cover, is the interoperability in the case of Federations and the portability in the case of Multi-Clouds. While these terms are well known in more general contexts of distributed systems, the main question that is raised in this paper *is how interoperability and portability are mapped into technical requirements in multiple Clouds*.

The paper is organized as follows. Section 2 is presenting the driving forces of the multiple Clouds. Section 3 is proposing a taxonomy for multiple Clouds. Section 4 is identifying the software products available for multiple Clouds. Two subsections are focusing on the particular subject of the Multi-Cloud requirements, respectively interoperability and portability requirements. Section 5 is looking to one concrete middleware. Finally, Section 6 draws the main conclusions.

The contributions of the paper can therefore be summarized in several categories:

1. a taxonomy for multiple Clouds;
2. match of the taxonomy with the support middleware, with a special accent on deployable services for Multi-Cloud;
3. a check list of the requirements for a middleware supporting a Multi-Cloud;
4. a check list for the interoperability and portability as expected to be supported by multiple Clouds.

The current paper is a follow up of the proceedings papers [64], in what concerns Multi-Cloud requirements, and [57, 65], in what concerns the interoperability and portability.

2 The Reasons for Using Multiple Clouds

The NIST report [53] has stated that the multiple Clouds can be used serially, when an application or service is moved from one Cloud to another, or simultaneously, when services from different Clouds are used.

The simplest scenarios are the migration from a Private Cloud to a Public Cloud (for the serial case), respectively the Hybrid Cloud, when some services are relying on the Private Cloud, while other services are lying on a Public Cloud (for the simultaneous use).

The reasons for which the services and resources from multiple Clouds are needed are various. Such reasons were reported in various research or business papers (like [26, 67, 71]). We considered useful to gather them here, in a non-exhaustive manner, by focusing on the ones that are appearing several times in different use cases. We name them 'Top 10 reasons'. Table 1 is exposing these reasons using also a mapping to the two cases mentioned by the NIST report (serial or simultaneous use).

The first need in the 'Top 10' is surely the cost optimization or improvement of service quality. It can be considered therefore as the main driving force for the use of the multiple Clouds.

The driving forces for multiple Clouds usage are depending on the actors' interests. The main actors in the usage scenarios are the following: the Cloud provider, the Cloud user (or Cloud consumer), the Cloud application developer, and the Cloud broker. Fig. 1 is suggesting a potential split of these interests.

3 Taxonomy of Multiple Clouds

While several taxonomies and ontologies are already available for the single Cloud (for example in [51]), the taxonomy of multiple Clouds is not well established and the border between different terms is still not well defined.

3.1 Basic Categories

We started from the list of the labels or names that are referring to multiple Clouds usage scenarios. However, we considered only those that are constantly repeated in the literature. Finally we arrived to the 'Top 25 categories of multiple Clouds'. By this frequency method, terms like 'Cloud Merge' or 'SuperCloud' (see [77]), which are only sporadically used, do not appear in this list. Table 2 is pointing to these categories and proposes for each of them a short definition, based on literature on that category (with the most significant reference being listed in the third column).

On the way of defining the relationships between these categories, we discuss in details each category in what follows.

3.2 General Criteria for Differentiation

Two main categories can be distinguished by the type of *delivery model* (named so in [24]): Federated Cloud and Multi-Cloud. The difference is made by the degree of collaborations between the Clouds involved and by the way by which the user interacts with the Clouds. The first model, called *Federated Cloud*, assumes a formal agreement between the Cloud providers. Service providers sub-contract capacity from other service providers and offer spare capacity to the federated group of providers. The consumer of the service is not aware of the fact that the Cloud provider he or she pays is using the services of another Cloud provider. The second model, called *Multi-Cloud*, assumes that there is no priori agreement between the Cloud providers and a third party (even the consumer) is responsible for the services. This third party contacts the service providers, negotiates the terms of service consumption, monitors the fulfillment of the service level agreements, and triggers the migration of codes, data and networking from one provider to another. The recent paper [28] underlines that this classification takes into account whether Cloud providers collaborate voluntarily (Federation) or not (Multi-Cloud).

Another way to distinguish the two categories is by looking to a simple example: application or service migration. The migration from one Cloud to another can be one-time or real-time. The Multi-Cloud

Table 1 Top 10 reasons for using multiple Clouds

Type of use	Reason
Serial usage	Optimize costs or improve quality of services
	React to changes of the offers by the providers
	Follow the constraints, like new locations or laws
	Avoid the dependence on only one external provider
	Ensure backup-ups to deal with disasters or scheduled inactivity
Simultaneous usage	Deal with the peaks in service and resource requests using external ones, on demand basis
	Replicate applications/services consuming services from different Clouds to ensure their high availability
	Act as intermediary
	Enhance own Cloud resource and service offers, based on agreements with other providers
	Consume different services for their particularities not provided elsewhere.

is usually supporting the one-time migration, by redeployment (with potentially synchronization of data) or even application rewriting, while the Federation is usually focusing on real-time migration. Therefore in Multi-Cloud one of the main issues is the portability of applications between Clouds, while in the Federation, the main issue is the interoperability between different Clouds, as earlier stated in [37, 57].

Coming back to the need for multiple Clouds, as exposed in a previous section, we see that the Cloud provider needs are mainly served by the Federated Cloud. The main driving force is the need of enhancing own Cloud resource and service offer, especially to acquire new resources due to limitation of the own ones. However, optimizing costs or following the constraints can be also the reasons of its usage,

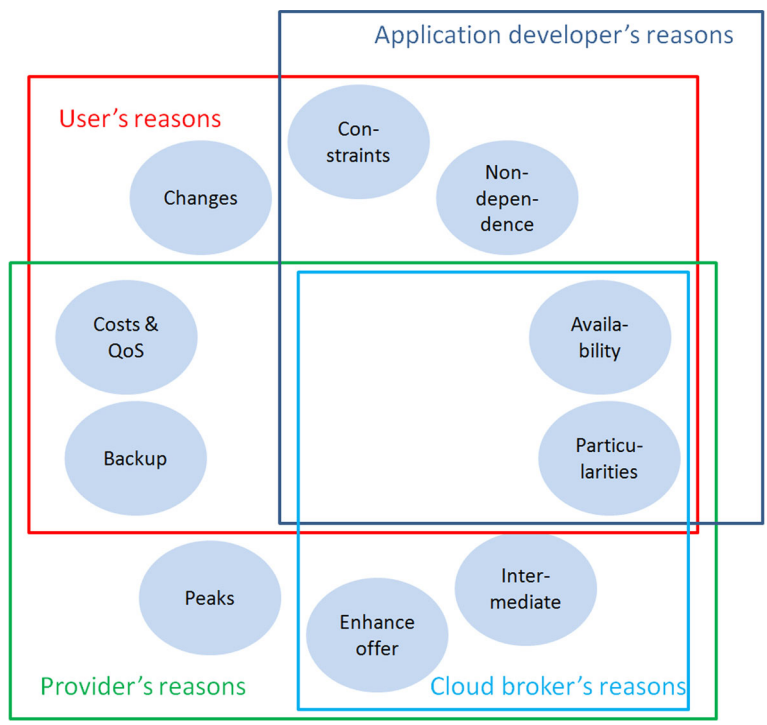

Fig. 1 Different actors' reasons for using multiple Clouds

Table 2 'Top 25' categories of multiple Clouds

Term in alphabetic order	Model	Ref
Aggregated clouds	Federation in which an agreement enables certain level of control over remote services, interchange of monitoring information, or advanced networking features.	[50]
Bursted cloud	A Private Cloud expanding its services on-demand bases using an external Cloud.	[23]
Centralized Federation	Federation in which a central entity performs or facilitates resource allocation, acting as a repository where available cloud resources are registered, or as marketplace for resources	[28]
Cloud blueprinting	Enhanced Cloud delivery model which use modular and easily combinable components, offerings an Integration-as-a-service functionality	[55]
Cloud broker	Entity that manages the use, performance and delivery of Cloud services and intermediates the relationships between Cloud providers and Cloud consumers	[53]
Cloud federation	Multiple Cloud delivery model assuming a formal agreement between the Cloud providers sub-contracting capacity from other providers or offering spare capacity to the federated group, in a non-transparent mode for their service consumers	[24]
Cloud governance	Set of decision making processes, criteria and policies involved in the planning, architecture, acquisition, deployment, operation and management of a Cloud computing capability	[52]
Cloud marketplace	A model, tool, interface or mechanism that can be use to buy, sell, configure, distribute and price Cloud oriented services	[34]
Clouds-of-clouds	Set of Cloud services that are composed of one or more services from other Clouds	[9]
Cross-clouds	Federation facilitating the management of computing resources from different Cloud providers in an homogeneous manner, including mechanisms for trust between Clouds and discovery of services, matching discovered services to customer requirements	[3]
Distributed cloud	Horizontal Federation based on a wide area network such as the Internet	[69]
Dynamic federation	Federation in which the providers collaborate dynamically to gain economies of scale and enlargements of their capabilities, and the inter-connections are dynamic in terms of joins and leaves, autonomous in terms of access control policy for each Cloud, and distributed in terms of trust establishment mechanisms like authorization delegation or role mapping across domains	[43]
Horizontal federation	Cloud Federation established at the same type of delivery model (IaaS, PaaS or SaaS)	[15]
Hybrid cloud	Set of services combined from a Private Cloud and one or more Public Clouds	[39]
Inter-cloud	Cloud Federation or Multi-Cloud oriented towards just-in-time, opportunistic and scalable application services provisioning, including at least one Cloud Broker and offering dynamic service provisioning	[7]
Library-based Multi-Cloud	Multi-Cloud based on a library that offers an uniform interface to the Clouds and a broker that takes care of provisioning of services across Clouds.	[27]
Multi-cloud	Multiple Cloud delivery model assuming no priori agreement between the Cloud providers and a third party responsible for provider contacting, consumption negotiation, SLA monitoring, inter-provider networking, code and data migration	[24]
Multi-tier cloud	Two or more Clouds tightly coupled with a third Cloud that has advanced control over remote services, full access to monitoring information and advanced networking features	[50]
Multiple cloud	Serial or simultaneous use of services and resources from geographically distinct Clouds	[53]
Peer-to-peer Federation	Federation in which the Clouds communicate with each other without mediators	[28]
Service-based Multi-Cloud	Multi-Cloud based on a service hosted externally or in-house by the customers and including a broker using service level agreement or a set of provisioning rules	[28]
SLA-based broker	Broker that acts according to the customer requirements specified in the form of a service level agreement	[28]

Table 2 (continued)

Term in alphabetic order	Model	Ref
Sky computing	Federation with dynamic provisioning of distributed domains, virtual image compatibility between providers, standard API for the services, and trusted networking environments	[36]
Trigger-based broker	Broker that follows rules triggering actions when predefined conditions considering the externally visible application performance indicators become true	[28]
Vertical federation	Federation of Cloud providers offering services to different layers, IaaS, PaaS and SaaS	[76]

especially in the case of own cost reduction politics or geographical location restrictions specified by the consumer.

The Cloud clients needs are mainly served by the Multi-Cloud model. The main drive is the need of cost optimization and improvement of service quality, as relying upon own or third party capacity to identify the appropriate service or resource for a concrete application or new service. However, other reasons like changes of the provider offers, constraints, availability, dependence avoidance, or backup need, can also motivate the usage of the Multi-Cloud.

In [2] a similar classification is considering, but on three levels, according to *types of collaborations*: federated, loosely coupled, and ad hoc. The metrics used to evaluate the type of collaborations are as follows:

(a) degree of inter-operation, which indicates the level of service and resource sharing among multiple Clouds;
(b) autonomy, which refers to a Cloud's ability to perform its local operations without any interference from other Cloud accesses;
(c) degree of privacy, which specifies the extent of information a service provider discloses about its internal policies and local constraints.

The *federated collaboration* is based on mutual dependence and trust among collaborating Clouds and supports a long-term inter-operation. The *loosely coupled collaboration* supposes that the local policies govern interactions among multiple Clouds (autonomous in terms of access policies and resource management). Since the last scenario involves interaction between Clouds, we consider in this paper that both two cases are instantiations of the Federated Clouds concept. The *ad hoc collaboration* supposes that the consumer or its software identifies at deployment phase the needs in terms of services and then contacts the Cloud providers. From our point of view this is a particular case of Multi-Cloud where the term 'collaboration' is not referred to the Cloud providers, but to their services inside a particular applications.

3.3 Categories of Federations

The Federation brings business advantages for the service providers. The main advantage is the overcome of their service limitations. According to [15], a Cloud provider chooses to establish a federation relationship with other providers for several reasons, as the followings:

(1) needs extra resources since the capabilities of its infrastructure are limited;
(2) needs to have some resources placed in a certain geographical location;
(3) wants to reduce costs allocating services in other Clouds.

According [75], usual examples of federation scenarios include: (a) capability enlargement and resource optimization; (b) provisioning of distributed anything as a service; (c) service consolidation and power saving. The authors of [27] evaluated the profitability of being involved in a Federation in such various scenarios.

In [50] federation scenarios are classified according the level of coupling or inter-operation among the Cloud services that are involved, on a range from loosely coupled, with no or little interoperability among Cloud services, to tightly coupled, with full interoperability among Cloud services. More precisely the degrees of coupling are regarding:

(1) level of cooperation among Cloud services;

(2) level of control and monitoring allowed over remote services;
(3) possibility of deploying cross-site networks;
(4) possibility of migrating virtual machines, data and networking between Clouds.

According these criteria the Federation can loosely, partially or tightly coupled. In a *loosely coupled Federation* a Cloud service has no or little control over remote Cloud services (e.g. no control on resource placement), monitoring information is very limited (e.g. only CPU, memory or disk consumption are reported), and no advanced features are supported (e.g. no cross-site networks or migration). In a *partially coupled Federation* the agreement enables certain level of control over remote services, interchange of monitoring information, or advanced networking features. A *tightly coupled Federation* is composed by Clouds belonging to the same organization, with advanced control over remote services, full access to monitoring information and advanced networking features.

Following the above classification, the authors of [50] identify four main *federation architectures*: bursting, broker, aggregated and multi-tier. *Cloud Bursting* (or more general Hybrid Cloud, in this paper) combines Private Cloud services with services from one or more Public Clouds (loosely coupled). *Cloud Broker* (Cloud Federation Broker in this paper) has access to several Public Cloud services (loosely coupled). *Aggregated Cloud* consists of two or more partner Clouds that inter-operate and aggregate their resources to provide their users with a larger virtual infrastructure (partially coupled). *Multi-tier Cloud* consists of two or more Cloud sites, usually belonging to the same corporation, that are managed by a third Cloud following a hierarchical arrangement (tightly coupled). We consider that the Cloud Broker from this category can represent a Federated architecture only it has several agreements with the Clouds to which it ensure the intermediation, and either is part of the mechanism of a particular Cloud, or it has some special mechanisms, like a single sign-on mechanism, that make it looks like a (small and specialized) Cloud service provider (i.e. a hierarchy of Clouds is established with one-directional contracts). A PaaS that runs on top of two or more IaaS services of other Cloud service provider can included in such category.

The *Multi-tier Cloud* is a first example of a *Hierarchical Cloud Federation*, terms that express the control degree of one Cloud over another. Particular and extreme cases are also Vertical or Horizontal Federations.

In [76] a complex theoretical model of a *Vertical Federation* is presented in which two Cloud service providers offering services to different layers, IaaS, PaaS and SaaS are federated. A layer can increase capacity through delegation as working together with its underlying layers to provide the required computing needs: the SaaS layer can ask the PaaS layer in the local Cloud for additional resources, while this can delegate to the local IaaS layer a request for more resources; if sufficient resources are not available locally the PaaS layer can attempt to acquire them from another Cloud in the Federation through brokering at the PaaS layer. A challenge to achieve the delegation is to introduce a standardized form of expressing inter-layer mappings.

A particular case of a *Horizontal Federation*, between the Cloud providers offering services at the same deployment level, is the *Cross-Cloud* aiming at facilitating the management of computing resources from different Cloud providers in an *homogeneous* manner, the primary goal being to provide flexibility and adaptability. According [3] a Cross-Cloud Federation model includes also mechanisms for: discovery of services; matching discovered services to necessary requirements; authentication (establishing a trust context between Federated Clouds). These characteristics are specific for a Cloud Federation Broker.

A *Horizontal Dynamic Cloud Federation* [31] assumes that different Cloud providers collaborate dynamically to gain economies of scale and enlargements of their service capabilities in order to meet consumer requirements (a difference from other models is given by a dynamic pricing). In order to achieve such dynamicity, in [43] is proposed a Cloud Virtual Organization based on a loosely coupling. The inter-connection are expected to be:

(a) dynamic: joins in or leaves out a virtual organization dynamically;
(b) autonomous: the participating Clouds have requirements to enforce the access control policy for their services independently;
(c) distributed: some scalable trust establishment mechanisms like authorization delegation or role mapping across domains, are established.

The concept of *Sky Computing* introduced in [36] refers to the dynamic provisioning of distributed domains over several Clouds is a particular case of Horizontal Dynamic Federation; this approach assumes, among others, the virtual image compatibility between providers, standard API for the services, and trusted networking environments.

In [28] Federations are classified as centralized or peer-to-peer. In a *Centralised Federation* a central entity performs or facilitates resource allocation (acts as a repository where available cloud resources are registered, or as market place for resources). In a *Peer-to-Peer Federation* the Clouds communicate and negotiate directly with each other without mediators.

3.4 Categories of Multi-Clouds

According to [28], the Multi-Cloud denotes the usage of multiple, independent Clouds by a client or a service. It does not imply interconnection and sharing. Clients or their software representatives are responsible for managing resource provisioning and scheduling. The authors of [28] are also classifying the Multi-Clouds as based on a *library* or a *service*. In the first case, a special service is offering brokerage between multiple Clouds based on clients' service level agreements or provisioning rules and performs deployment, execution and monitoring. In the second case, a library facilitates a uniform way to access multiple services and resources, as well as the provisioning of services and resources from multiple Clouds.

We consider that a *Hybrid Cloud* is a type of a Multi-Cloud that connects two or more Clouds in terms of their deployment models (Public, Private, Community). Often Hybrid Clouds are used for *Cloud Bursting*, i.e. the usage of external Cloud services when the Private ones are not sufficient. One of the current main concerns is the transfer of huge amount of data among Clouds. Note that in [23] the Bursted Private Cloud is considered to the same level with the Federation and Multi-Clouds.

3.5 Categories of Inter-Clouds

A Cloud Federation or a Multi-Cloud that includes at least one Cloud Broker and offers dynamic service provisioning is an *Inter-Cloud*. Several detailed definitions for the Inter-Cloud, pointing mainly towards a dynamic Federation and its aim, are currently available. According [12] the Inter-Cloud depicts several Cloud providers and Brokers that dynamically negotiate resources between themselves in order to seamlessly meet elastic applications service level agreements by scaling applications across various Clouds. According [28], the Inter-Cloud refers to the dynamic coordination and distribution of load among a set of Cloud data centres. According [11], the Inter-Cloud is a Cloud Federation oriented, just-in-time, opportunistic and scalable application services provisioning environment. It is expected to provide an opportunistic and scalable application service provisioning environment. Moreover, it goes beyond the management of resources and service from different Clouds that is encountered in Federations or Multi-Clouds, by including Cloud governance or marketplace, as well as a Cloud Broker.

The Inter-Cloud is focusing nowadays the developers and researchers attention. Recent architectures for particular Inter-Clouds were proposed for example in [20, 25]. The functional requirements for Inter-Cloud were identified in [3] as being the followings: provider selection based on SLA-defined quality requirements, monitoring, provisioning, resource discovery and protection, resource management, service setup, authentication, network interworking alternation and retrieval of data, resources releasing. According [12] the basic components of an Inter-Cloud are the Cloud Coordinator, which represents providers in the marketplace, the Cloud Exchange, which acts as a discovery system and offers a publication system (catalog), and the Market Engine that evaluates offers and resources.

NIST Cloud Computing Reference Architecture [53] considers as major parties involved in a Cloud ecosystem beyond the Cloud consumer, and Cloud provider, the Cloud broker. The Broker is an entity that manages the use, performance and delivery of Cloud services and intermediates the relationships between Cloud providers and Cloud consumers. Its roles are (extension of the ones from [32, 33, 47]): optimizer in finding the best match between requirement and offer; adapter by offering a unique management interface; extension of existing services; aggregation with indexing; splitter of user requests to multiple providers; or arbitrage between Clouds. According to [28], in the case of Multi-Cloud, the Broker is often part of the service or library. In the case of Federations, it is implemented either in a centralized entity or by the

Cloud providers. Same paper is classifying the brokering mechanisms as SLA based, when requirements are specified by clients in the form of a service level agreements (as the example from [17]), or trigger-action, when rules are becoming active, triggering an action, when a predefined condition considering the externally visible application performance indicators becomes true.

Building the Inter-Cloud is more than technical protocols: a blueprint for an Inter-Cloud economy must be architected, with a technically sound foundation, but also including governance and marketplace architecture enabling all the attributes of an economy [7].

In a *Cloud Marketplace* the service consumers can rate and recommend services and providers. For a Cloud provider, such marketplace is not only a means to expose service offerings, but also a means to enrich new offerings via custom adaptation [1].

According [55] a *Cloud Blueprinting* is an enhanced Cloud delivery model, a reference architecture that offers Integration-as-a-service functionality and in which the Cloud stack is build from modular and easily combinable components (the blueprints). A blueprint offers means to rapidly and easily deploy pre-built, pre-configured, pre-optimized application payloads on virtual resource pools on the Cloud and helps configure a multiple Cloud environment to meet application requirements and policies. It includes also a detailed deployment plan and a integration solution description, and abstracts the technical details of the interaction with the Cloud. A meta-data or model-driven approach is used to represent and manage Cloud services Meta-data constructs in form of templates are used for providing a common understanding of the service features; a model-driven approach is employed in order to automatize the application deployment.

3.6 Relationships between Categories

Figure 2 proposes a schema of the relationships between the various terms depicting usage of multiple Clouds.

The first level is splitting the multiple Clouds according to the delivery model. We consider that Inter-Cloud, as an emerging model, should be considered at the same level of Federations and Multi-Clouds. The second level is related to the organization of various interactions between the Clouds involved. The third and four level are referring to the architecture's organization.

3.7 Examples of Multiple Clouds Usage

Several applications that are using services from multiple Clouds were reported in the recent years. We point here only four different examples, relevant for the categories that were discussed in the previous section. The first is a real-time online interactive application using a business-oriented Federated Cloud that was built to provide scalable infrastructure and QoS-assured hosting services (with security features and on-demand resource provisioning across Clouds) – details can be found in [78]. A weather research and forecasting service is introduced in [76] as SaaS that illustrates the concepts of delegation between service layers (in Hierarchical Clouds), and the needs in terms of Broker functions. A P2P monitoring application for a service-based Multi-Cloud of thirteen Cloud environments is reported in [56]: the support middleware is able deploy services based on a open service model, a configurable kernel able to fit the characteristics of concrete Clouds, and infrastructure management services. Olaii is a commercial product that helps discovering events or finding friends to go out with, and uses RightScale for processing and Amazon AWS for the interfaces [66].

In [28] is introduced a classification of the distributed applications deployed in multiple Clouds: interactive or batch-processing. Interactive (online) applications have constant interaction and input from the user, most being data centric (like in online transaction processing, multilayer online games). In batch processing case, singular jobs are representing resource intensive applications (high performance/high throughput/many task computing, scientific workflows), or periodical jobs are repeatedly executed over a period of time (data centric, long running tasks over huge enterprise data bases, administrative periodical jobs). Online applications for example can benefit from the geographical diversity of multiple Clouds. In [31] is pointed that a Dynamic Horizontal Federation is needed for massively multiplayer online role-playing games like World of Warcraft or Second Life that require huge amount of Cloud services which cannot be provided by a single Cloud provider.

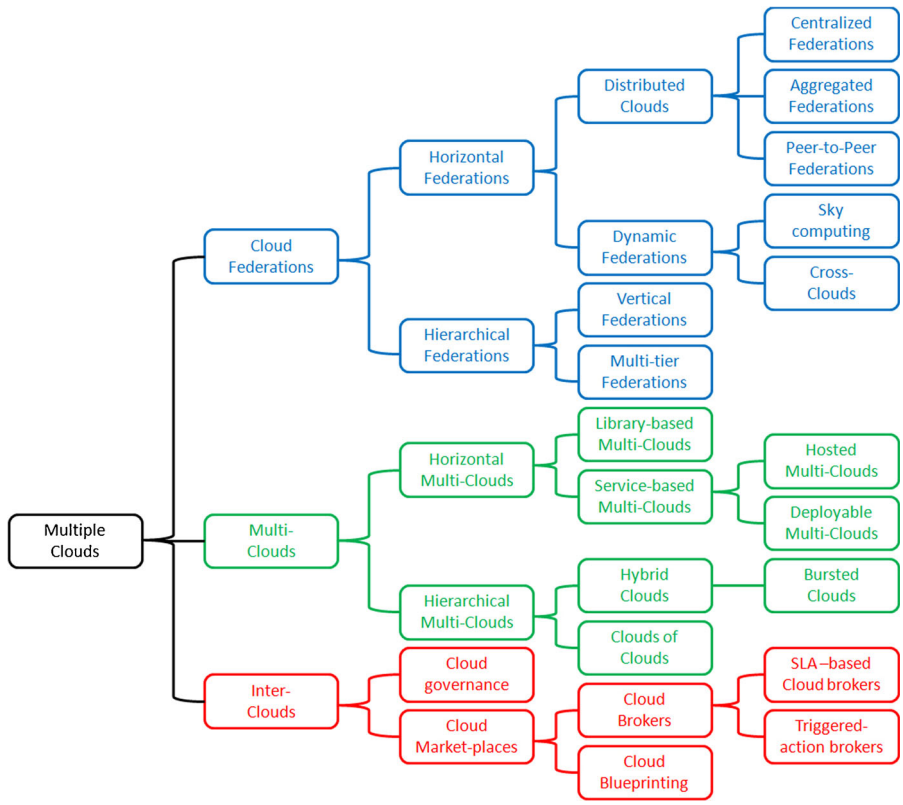

Fig. 2 Relationships between the terms depicting multiple Clouds

Same problems are encountered by the Social Networks that serve dynamic content to millions of users, whose access and interaction patterns are hard to predict. Cloud Bursting is needed at different locations, for instance, whenever new system features become popular or a new plug-in application is deployed.

4 Cloudware Support for Consuming Resources and Services from Multiple Clouds

In order to extract the main requirements for a Multi-Cloud middleware we go backward, from the results to the requirements: first we identify the current available middleware, and then we establish the least common multiple of their functionalities. The reason is that each tool is expected to have behind a list of user/consumer requirements. At least the common multiple of these requirements is desirable and possible to be satisfied by the current middleware.

4.1 Available Cloudware for Cloud Federation and Multi-Cloud

Cloudware is a general term referring to a software that enables building, deploying, running or managing applications in a Cloud computing environment. While in the case of a single Cloud, the term is sometime associated with a PaaS, as referring to the development tools for creating Cloud aware applications and services, we consider that in the wider context of multiple Clouds it can receive a new meaning, as it should act as a middleware that hides the heterogeneity of the resources and services.

In this section we point towards the available software and services for supporting the consumptions of resources from multiple Clouds. The list of the available libraries, tools, services and platforms for Multi-Clouds presented in this paper is not exhaustive. Instead we picked the most significant initiatives from the research and commercial sectors. Table 3 describes what we consider to be 'Top 25' Cloudware solutions for multiple Clouds. As the multiple Clouds

Table 3 'Top 25' software, services, interfaces for consuming resources and services from multiple Clouds (alphabetic order)

Product	Available at	Short description
Agility	arjuna.com/agility,	Commercial framework based on SLAs for establishing Federations and Cloud Bursting
Aoleus	aeolusproject.org	Open-source Cloud management software written in Ruby, allowing to choose between Private, Public or Hybrid Clouds, and using δ-Cloud library
BonFIRE	portal.integration.bonfire.grid5000.fr	Resource manager that operates a Multi-site Cloud-based facility on top of infrastructure testbeds
Cloud4SOA	cloud4soa.eu	Platform allowing portability of Java applications between PaaSs using semantic technologies and supporting OpenShift, CloudControl, CloudBees, CloudFoundry, Beanstalk, Heroku, AppEngine
CloudBroker	cloudbroker.com	Software-as-a-service application store that allows offering and using compute-intensive applications on different cloud infrastructures, accessible via a web browser, REST and Java APIs, as well as the command line
ConPaaS	contrail-project.eu/conpaas	Federation middleware that allows multi-tenancy and Cloud Bursting and upgrades the supported Cloud providers by adding SLAs features
δ-Cloud	deltacloud.apache.org	REST-based API written in Ruby necessary to connect to various Cloud providers: Amazon EC2, Eucalytus, SmartCloud, GoGrid, OpenNebula, RackSpace, OpenStack, etc
Enstratius	enstratius.com	Configuration management, monitoring, governance and automation system that supports AWS, CloudStack, CloudSigma, Eucalyptus, GoGrid, Joyent Cloud, OpenStack, Rackspace, vCloud, Azure etc
JClouds	www.jclouds.org	Open source Java library that introduces abstractions aiming the portability of applications and supports more than thirty Cloud providers and software stacks including AWS, GoGrid, vCloud, OpenStack, Azure
Kaavo	www.kaavo.com	Deployment and management system for distributed applications, workloads, and environments in various Clouds allowing resource management across Public, Private, and Hybrid Clouds based on AWS, Rackspace, Terremark, Eucalyptus
LibCloud	libcloud.apache.org	Python library that abstracts away differences among multiple Cloud provider APIs, like OpenNebula, GoGrid, Enomaly, SliceHost, Elastic Hosts, RackSpace, Eucalyptus, AWS, Joyent, vCloud etc
mOSAIC	bitbucket.org/mosaic	API and deployable PaaS that allows deployment, configuration management, control of the life-cycle of applications or services consuming IaaS; supports more than ten providers, including AWS, GoGrid, vCloud, Eucalytus, RighScale, CloudSigma, Flexiscale
MODAClouds	modaclouds.eu	Platform allowing model-driven engineering based on CloudML (www.cloudml.org), Modelio (forge.modelio.org) and Palladio (www.palladio-simulator.com) at design time and able to execute applications on top of mOSAIC and Cloud4SOA
Nimbus	nimbusproject.org	Virtual site layer for dynamically provisioned distributed resources of multiple data centers in a Federation model and allows the usage of OpenStack and AWS
OpenCirrus	opencirrus.org	Federation between universities and research centers to support research in design, provisioning, and management of services in scale of multi data centers.

Table 3 (continued)

Product	Available at	Short description
OpenNebula	opennebula.org	Open-source Cloud resource management system that allows Aggregated Clouds and Cloud Bursting
OPTIMIS	www.optimis-project .eu/Toolkit_v2	Platform for Cloud service provisioning that manages the lifecycle of the service and addresses risk and trust management; supports AWS, OpenNebula, Eucalyptus, Emotive, Flexiant
RightScale	www.rightscale.com	Private Cloud management platform for control, administration, and life-cycle support of deployments across multiple Clouds, with server templates to automatically install software, and support for AWS, Eucalyptus, GoGrid, VMware, FlexiScale
SAGA	saga-project.github.com	API for manaining e-infrastructures, from Grids to Clouds
Scalr	scalr.net	Broker that allows to deploy virtual machines across different Public and Private Clouds and uses automated triggers (trigger scale up/down actions).
SimpleCloud	www.simplecloud.org	PHP library providing common interfaces for file and document storage services, queue services and infrastructure services of AWS, RackSpace, Azure, Nirvanix
SpotCloud	spotcloud.com	Marketplace where service providers sell the extra capacity they have and the clients can select the 'best' service provider at a certain moment.
Stratos	wso2.com/cloud/stratos/	Core services and building blocks for federated identity and single sign-on, data-as-a-service and messaging-as-a-service, monitoring of SLAs, CPU, memory and bandwidth utilization and automatically scales up or down depending on the load.
StratusLab	stratuslab.eu	Open source IaaS distribution that can be used for Cloud Bursting
Xen-blanket	xcloud.cs.cornell.edu	Hypervisor running inside a virtual machine in a Public or Private Cloud, exposing a homogeneous interface to guest virtual machines, and enabling hypervisor-level techniques and management tools, like VM migration, page sharing, and over-subscription

are based on the values of the Cloud services of various providers, thin layers build on top to ensure the functionality of Multi-Cloud of Federated Cloud are often open-source. We tried to focus mostly, were possible, on these solutions, due to their potential for as wide adoption.

As identified in [28] and mentioned in the previous section, the Multi-Cloud middleware can be library-based or service-based. The most known library-based approaches are jclouds, libcloud, δ-cloud and SimpleCloud. Jclouds is an open source Java library designed to support the portability of Java applications, which allows the uniform access to the resources from various IaaS providers. Libcloud is a Python library that abstract the differences among the programming interfaces of Cloud services. δ-cloud is a REST-based API written in Ruby which allows also the connections to various Cloud resources. SimpleCloud is a PHP library offering uniform interfaces for file and document storage, queues and infrastructure services.

As stated earlier in this section and in Fig. 2, we consider that service-based approach for Multi-Cloud can also be classified in two categories: hosted or deployable. The most known hosted services are the commercial offers of RighScale and Kavoo. RightScale is offering a management platform for the control and administration of deployments in different Clouds. Its Multi-Cloud Engine is able to broker capabilities related to virtual machine placement in Public Clouds. Kaavo allows the management of distributed applications and workloads in various Clouds.

Several deployable services are results of open-source projects like Aoleus, mOSAIC, Cloud4SOA or OPTIMIS. Aeolus is an open-source cloud management software written in Ruby and provided for

Linux systems by RedHat and it is based on the δ-cloud library. mOSAIC offers an open-source API and a deployable Platform (-as-a-Service) allowing the deployment and the life-cycle control of applications consuming infrastructure services [66]. OPTIMIS offers a deployable Platform (-as-a-Service) that allows Cloud service provisioning and the management of the life-cycle of the services. Cloud4SOA is dealing with portability of applications between PaaSs by relying upon semantic technologies [18, 19].

Other current projects related to multiple Clouds are: 4CaaST that is building a Cloud blueprint (details at 4caast.morfeo-project. org), REMICS that is migrating applications from one Cloud to another using model-driven engineering, code introspection and rewriting (www.remics.eu), Vision Cloud that is dealing with vendor-agnostic data management services (www.visioncloud.eu), TClouds that is investigating the trust management in Clouds-of-Clouds (www.tclouds-project.eu). Broker@Cloud (www.broker-cloud.eu) develops methods and mechanisms for quality assurance and optimization of software-based services, while Cloudspaces (www.cloudspaces.eu) is looking into interoperability mechanisms between Personal Clouds.

MODAClouds is trying to follow the model-driven Blueprint approach and is relying upon the solutions of mOSAIC, Cloud4SOA and OPTIMIS projects for Multi-Clouds [5]. Beyond MODAClouds and REMICS, model-driven engineering is proposed to be used in multiple Clouds also by ARTIST (www.artist-project.eu) and PaaSage (www.paasage.eu).

The Cloud brokers are playing an important role in both Multi-Cloud and Inter-Cloud. The most known independent Cloud brokers (without offering a complete solution for a Multi-Cloud) are: SplotCloud, Scalr and Stratos. SpotCloud provides a marketplace for infrastructure service and a matching service with the client requirements. Scalr provides deployment of virtual machines in various Clouds and includes automated triggers to scale up and down. Stratos offers single sign-on and monitors resource consumption and the fulfillment of service level agreements and offers auto-scaling mechanisms.

Another commercial solution, Enstratius, allows the management, monitoring, automation and governance of resource consumption based on the services from various Cloud providers.

To complete the image of the existing management systems of the resources and services for multiple Clouds, we consider also the ones supporting the Federations of Clouds. Contrail project is promoting a management system (named ConPaaS) for web application deployments in Centralized Federations. Similar offer for Centralized Federations is provided by BonFire. OpenNebula is a widely used open source solution for Aggregated Clouds, developed partially in the frame of the RESERVOIR project (www.reservoir-fp7.eu).

The Nimbus platform implements the Sky computing concepts. The OpenCirrus is a research testbed for Peer-to-Peer Clouds. The Xen-Blanket exposes a single Cross-Cloud-controlled virtual machine interface and service suite to the users such that a guest virtual machine image can run on any provider infrastructure without modifications.

We observe that the middleware supporting the multiple Clouds usage scenarios needs to rely upon or even ensure a fair image or comparison of the costs and service quality. However, few implementations of the multiple Cloud concept are offering or are lying on scoreboards. This is a consequence also of the high technical challenge in measuring the services' quality. An overview of the on-line comparisons for the Cloud services was recently done in MODAClouds [29].

Several scientific prototypes are reported in the literature for multiple Clouds, but not yet available for a production stage. We mention here CCFM [13], CloudBus [11], CloudCmp [42], Cloudle [35], ORCA [49], PSIF [45], Scalia [54], SERA [22], Shrinker [69], Zeel/i [30].

Finally, Table 4 introduces a mapping of the existing middleware to the various categories of multiple Clouds.

4.2 Technical Requirements for a Multi-Cloud Cloudware

After an analysis of the main features and driving forces of the Cloudware presented in the previous section, we have identified the least common multiple of their features. We restricted our study only to Multi-Clouds, i.e. only almost half of the Cloudware from Table 4 as targeting all multiple Clouds categories is too complex at this moment. Figure 3 provides a schematic view, while details are given in Table 5.

Table 4 Examples of Cloudware for multiple Clouds

Delivery model	Organization	Type	Architecture	Middleware examples
Federation	Horizontal	Distributed	Centralized	BonFIRE, ConPaaS
			Aggregated	OpenNebula
			Peer-to-Peer	OpenCirrus
		Dynamic	Sky computing	Nimbus
			Cross-Clouds	Xen-Blanket
	Hierarchical	Vertical		
		Multi-tier		
Multi-Cloud	Horizontal	Library-based		jclouds, Libcloud, δ-Cloud, SimpleCloud, SAGA
		Service-based	Hosted	RightScale, Kaavo
			Deployable	mOSAIC, Cloud4SOA, Optimis, Aoleus, MODAClouds
	Hierarchical	Hybrid Cloud	Bursted Cloud	StratusLab, Agility
		Clouds of Clouds		TClouds
Inter-Cloud	Governance			Enstratius
	Marketplace	Cloud brokers	SLA-based	SpotCloud, Stratos, CloudBroker
			Trigerred-action	Scalr
		Blueprint		4CaaST

Two different points of views were used in the requirements classification: the middleware developer point of view – the requirements are presented in form of principles or tools constraints– and the Cloud application developer point of view interested in application phases – the requirements are associated to the development phase, deployment phase, or execution phase.

The development of a Multi-Cloud middleware requires to offer solutions to multiple levels: business by establishing strategies, regulations, or mode of use; semantic by establishing a taxonomy for calls, responses, functionality; application and services by enabling automation or configuration; management by using rules, protocols, standards in deployment or relocation; image and data by using the specificities of each Cloud that is connected; network by allocation

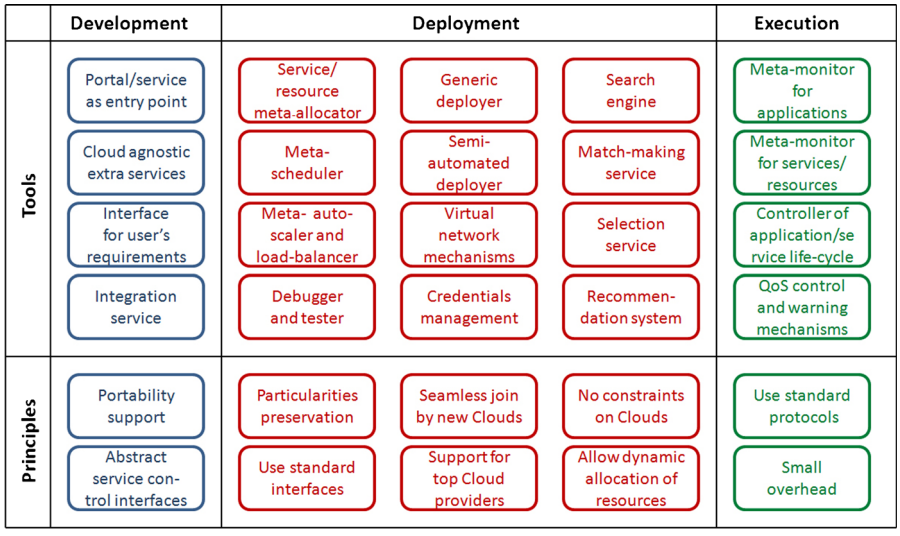

Fig. 3 Functional requirements for a Multi-Cloud Cloudware – schematic view

Table 5 Functional requirements for a Multi-Cloud Cloudware classified on stages, tools and principles

Group	Address	Requirement
Development	Tools	Resource/service (meta-)management (portal, service, interface)
		Services that are Cloud vendor agnostic
		An interface for describing functional and non-functional requirements
		An Integration-as-a-Service or service aggregators to combine services from different Clouds
	Principles	Abstract service control interfaces of multiple Clouds
		Support the application portability between the connected Clouds
Deployment	Tools	Selection service of consumable Cloud services and resources
		A service and resource meta-allocator
		A (meta-) scheduler
		A (meta-)load-balancing or auto-scaling mechanisms
		Deployer on Private Clouds to enable testing, debugging, or privacy
		Deployer of components of applications in multiple Clouds
		Automated procedures for deployments
		Network overlay mechanisms to overcome limited connectivity
		Authentication services for single sign-on or credentials repositories
		A search engine based on a taxonomy or using semantic processing
		A match-making or brokering service
		A service and resource selection interface
		A recommendation system, a trust management system or a reputation management system
	Principles	Preserve the particularities of various Clouds
		Comply with current standards/protocols for resource management
		Allow seamless join by new Cloud without changing local policies
		Support the connection with the top Cloud providers
		Do not impose any constrains to the connected Clouds
		Allow dynamic allocation of resources or mechanisms for self-adaptation
Execution	Tools	A (meta-)monitoring service for the deployed applications
		A (meta-)monitoring facility of the Cloud resource consumptions
		Control of the full life-cycle of the deployed applications
		Metering of the degree of fulfillment of the service level agreements
	Principles	Comply with current standards/protocols for Cloud resource usage
		Introduce only a small overhead in comparison with a direct connection to each supported Cloud

and admission procedures. A visual representation of these categories and issues is presented in Fig. 4.

In order to tackle with all the above enumerated levels, a large team with various expertises is needed, not necessarily available to one research team or one company.

As we have consider the least common multiple of the Cloudware features, none of the technical solutions for Multi-Clouds that were mentioned in the previous subsection are complying with all the requirements. Heterogeneity is encountered to both low and high levels, from virtualization technologies, to programming environments. It is expected that the Multi-Cloud is hiding this heterogeneity. If this is happening at the Cloud provider level to a certain level, the meta-level is still lacking a complete offer beyond the research prototypes (i.e. fulfillment of all requirements from the development group). An analysis of

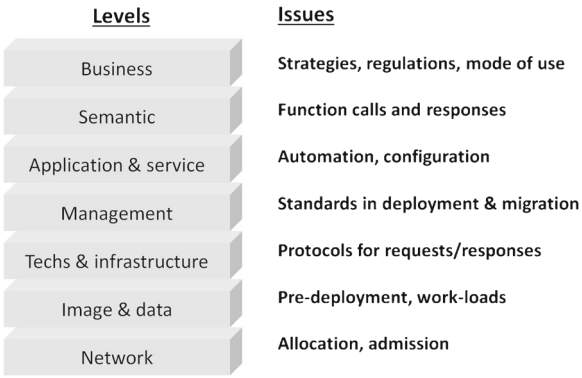

Fig. 4 Issues to tackle with when building a Multi-Cloud middleware

the degree of fulfillment of the requirements by each Cloudware is out of the scope of this paper (only one is considered in the next section). However we discuss in what follows which are the technological barriers in enabling a Multi-Cloud middleware.

A Multi-Cloud enabler is invited to follow the current *Cloud standards*. While the current Cloud standards are few (OCCI, CDMI, CIMI, TOSCA being the most relevant in the Multi-Cloud context), they are still not adopted on large scale. One reason is their limited scope (e.g. at IaaS level, not yet for PaaS level [41]). Another reason is the reluctance of the Cloud providers which do not see them as business needs or priority (or even the contrary, as a danger for innovation and market advantage [70]).

As mentioned earlier, there are several libraries abstracting the Cloud APIs that are more widely adopted than the standards. However these libraries are offering the *common denominator* of the underlying services, and are loosing their individuality, i.e. entering in contradiction with the requirement of preserving the particularities of each Cloud (see a study of the effect of the common denominator for the Federations in [8]). Moreover, they are compliant with portability requirement only for off-line case (stopping the application in the current Cloud, and restarting it entirely and from the beginning in another Cloud). A more complex case is that in which the relocated application is decomposed and relocated over a new set of Clouds.

The usage of services from multiple Clouds has been introduced first with the idea of Hybrid Cloud, where a Private Cloud and a Public Cloud are building a transitory Multi-Cloud. However the outages of Public Clouds and the security breaks have brought into discussions the trustfulness of the Hybrid Cloud. A solution to a *recommendation system* is therefore needed in Multi-Cloud, and very few prototypes are currently available as the trust management problem is more than a technical one.

The diversity of services is a challenge for the service selection. A methodology to compare Cloud service based on multiple criteria and for various user p.rofiles is needed. Comparison criteria can vary from cost, policies, performance and so on. Best matching in a certain context instead an optimal matching is expected in most cases due to the complexity of the problem. Moreover, for the performance measurements, independent observer services need to be built, in the context that the Cloud providers are reluctant in providing monitoring services or data about their services' performances. Furthermore, the few current *monitoring services* are heterogeneous and a meta-monitoring service is difficult to be build [38]. One of the highest challenge of the MODAClouds project is therefore to build a such meta-monitor (preliminary architecture in [6]).

The hosted services are the most common services and therefore the common understanding of Cloud services is referring to this group. Their interfaces conceived by the providers are unique entry points to complex processes and heterogeneous resources. In the condition of ignoring the Cloud standards and protocols, a common understanding on a certain action or feature is hard to be reached. The Multi-Cloud developers needs to understand each particular interface in order to connect each service, fact that is not compliant with the requirement of a seamless join procedure of a new Cloud. In particular, the achievement of *portability* is considered a moving target, difficult to reach, if the set of APIs to comply with is constantly increased.

4.3 Interoperability and Portability Requirements

Earlier in this paper we underlined that the fulfill of the interoperability requirement is crucial in Federations of Clouds, while the fulfill of the portability requirement is crucial in multiple Clouds. Interoperability and portability are needed for at least for the following three reasons: protection of the end user investments in developments; development of a Cloud

eco-system and market; exploit at full the advantage of elasticity and pay-as-you concept.

In previous papers [57, 65] we have elaborate on the dimensions, levels and technological requirements of interoperability and portability. We remind them here and we elaborate them further from the perspective of the requirements that should be imposed to the libraries, tools or platforms that are supporting the multiple Clouds.

Three dimensions should be escalated in interoperability and portability problem. Two are technical and encountered at run-time in Federated Clouds and at design time in Multi-Cloud. The third dimension is non-technical, as being related to policy, i.e. encompassing the agreements and contracts establishment between providers, as well as standards' elaborations and promotion.

Figure 5 is proposing a split of the requirements in different categories according to the three dimensions. The second level of classification refers to the level of expertise to which is addressing: public administration, business, application/service abstractions, etc (as mentioned in the previous subsection). Table 6 is detailing the requirements exposed in Fig. 5, by associating them with a certain level or dimension.

We have recently reviewed in [57] the technical solutions available to support interoperability and portability. The closest and recent snapshot of the interoperability solution was provided in [46]. We insist in this section more on the *gaps* that should be filled, using few examples.

At the business level, a unified policy of the contractual terms was not yet established at national or international levels, while several proposals are on the way. One of the most disputed topic is the privacy and data protection compliance, for which no general accepted proposal is currently available.

The diversity of the APIs is natural, as each providers intends to offer something new or unique compared with other offers, in order to attract customers. The interoperability issue is therefore an issue for the *management and governance levels where automation should be achieved* as much as possible. The mix-in of services from different providers can

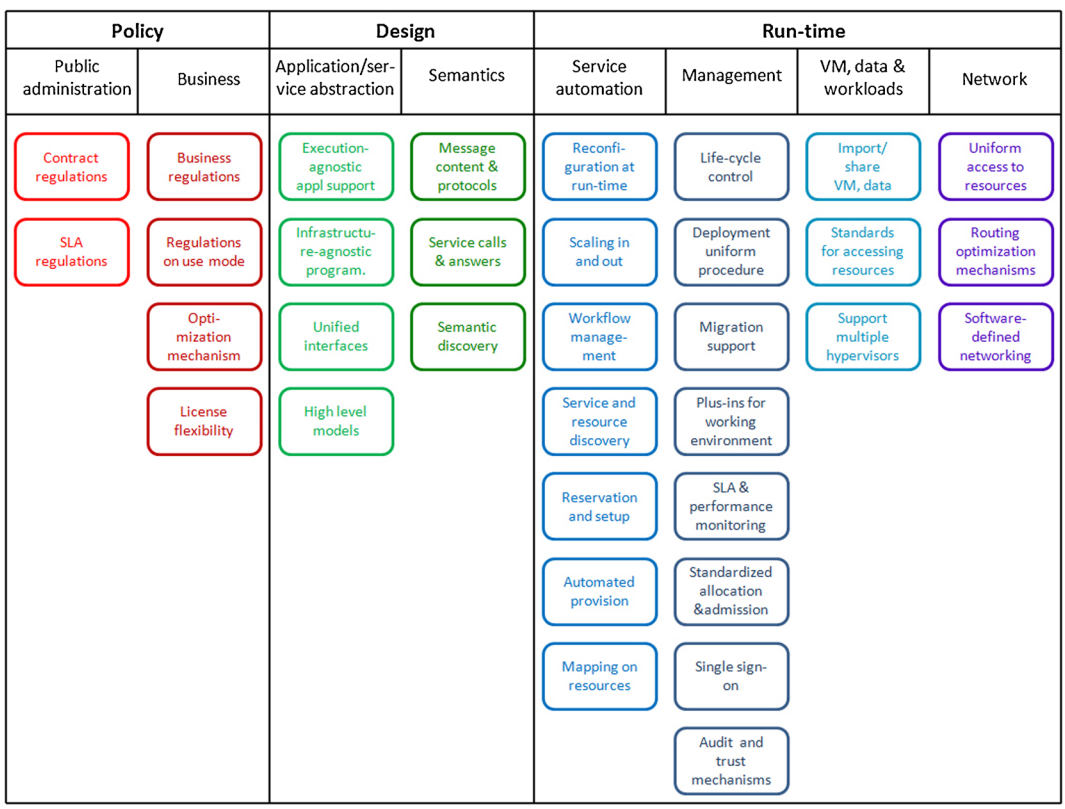

Fig. 5 Interoperability and portability requirements

Table 6 Minimal requirements of interoperability at various levels

Dimension	Level	Requirements
Policy	Public administration	Regulations on contracts
		Regulations on services level agreements
	Business	Business strategies
		Regulations on mode of use
		Economic model driven optimization mechanisms
		License flexibility
Design	Application/service abstractions	Execution-unaware application support
		Infrastructure independent programming
		Common set of interfaces, standard APIs
		High level modeling and programming models
	Semantic	Message content and communication protocols
		Service calls and answers
		Cloud ontology and semantic-based discovery of services
Runtime	Service automation	Re-configuration at run-time
		Scaling in and out on multiple sites
		Workflow management
		Service and resource discovery
		Reservation and setup
		Automatically provision resources in multiple Clouds
		Optimal mapping of services/applications on resources
	Management	Control the life-cycle of the application and its components
		Standardized functionality for deployment in multiple Clouds
		Migration support for application components
		Plug-ins for the working environment
		SLA and performance monitoring
		Standards for allocation and admission control
		Single sign-on for users accessing multiple Clouds
		Auditing and trust mechanisms
	VM images, data and workloads	Import and share VMs, data and workloads
		VM, data and workloads migration support
		Use standards for accessing compute and storage capacities
		Support for multiple hypervisor technologies
	Network	Uniform access to individual resources
		Routing optimization based on monitoring
		Software-defined networking

be a strong argument in using such entry level instead a direct connection to only one provider, despite the overhead that is associated with the new layer. Therefore the management and service automation levels in multiple Clouds are the hot-spots of the development activities in the last three years.

For the design time dimension, several *libraries* are available, like jclouds, libcloud, and others (basic

list in [40]), or *domain-specific languages* for modelling provisioning and deployment, like CloudML (cloudml.org) in MODAClouds, but there is no wide acceptance of one or another proposal.

Once the application is deployed and adapted to a certain Cloud, in order to move it in another Cloud, an *inspection of the source code* is needed to identify the specific API calls or to build a model or representation of the code. Tools that can do that are only in early stage of prototyping (in projects like Cloud4SOA, REMICS, MODAClouds or ARTIST) and not yet available for large scale usage. Commercial solutions like RightScale or Kaavo are able to deploy applications in various Clouds, but not yet migrate the running ones.

For the run-time dimension, at the infrastructure level (last two in the table and image), several partial solutions exists to migrate virtual machines, virtual storage or services. Despite the presence of standard OVF format or Xen-blanket, the *virtual machines and their associated data are not yet ready for real-time migration*. For example, Amazon is one of the few Cloud providers who are allowing to export virtual machines; however, their related resources (e.g., network, storage) cannot be exported too. In general, virtual machine images cannot be transferred from one hypervisor to another. The virtual machine images can be converted today with tools like qemu-image, but this requires to stop the virtual machine and to apply the adaptation off-line.

A classical way to bust the interoperability and portability is the *adoption of standards and open source*. Surely they are important mostly for the Cloud providers and service developers, and not in a similar degree, for the consumers. A classification of the standards at IaaS level was done in [72] and refers to access mechanism, virtual appliance, storage, network, security and SLA; the paper also analysis the three oldest standard proposals, OVF, OCCI and CDMI from these point of views, identifying their gaps. Note that the three nominated standards have partially failed to be adopted on large scale by the providers [48], but are implemented in few Cloud management middlewares. New ones are emerging nowadays, like CIMI or CloudML, and several collaborative groups are working to elaborate other proposals. An early list standard initiatives that are relevant for interoperability is provided in [44]. Comparing it with the current list, the tremendous changes from the last two years are evident. However there several gaps in the collections of available standards, like proposals for Cloud metrics and real-time monitoring, interfaces for security(-as-a-)services, accountability associated with transparency and responsibility.

5 Case study of a Cloudware Support for Multi-Cloud and Inter-Cloud

In this section we are looking in more details to a certain Cloudware, mOSAIC. It reflects the joint efforts of a multi-national team, of the FP7-ICT project with the same name. The acronym stands from *O*pen *S*ource *API* and Platform for *m*ultiple *C*louds. Several scientific papers that are enumerated on the main project site (www.mosaic-cloud.eu) are providing valuable hints about the concepts and prototype implementations of different building blocks. The latest overview paper is [66]. In this paper we will refer to the particular features that are supporting the consumption of resources and services from multiple Clouds. The identification of the missing features of mOSAIC compared with other Multi-Cloud middleware are necessary to evolve it in the second stage of evolution, as Execution Platform for MODAClouds (details are available in [16]).

mOSAIC's Cloudware includes five building blocks:

(1) application development tools: language-specific libraries in Java, Python, Erlang and Node.js, Eclipse-plug-ins, configurators, a Cloud desktop simulator (PTC) or Semantic Engine;
(2) platform: Packager, Deployer, Scheduler to the deployable re-usable codes and mOS, a the minimal kernel of the Platform runnable in Linux environments;
(3) vendor modules, for different Cloud providers and their technologies;
(4) infrastructure support – Cloud Agency components;
(5) proof-of-concept applications, in five different fields.

The repository of the open-source codes is located at bitbucket.org/mosaic.

In what follows we point towards few basic concepts of the mOSAIC Cloudware and how they are implemented in practice. The underlined keywords are

the ones indicated in Tables 5 and 6. We considered worth to mention them as being applicable to any Cloudware that supports multiple Clouds.

Support for Component-Based Applications (towards a Cloud Blueprinting) The most successful applications for Cloud computing are the Web-based ones, as the unexpected peaks in their usage can be compensated by fast provisioning a new resources to be consumed. Naturally, such applications are build from components, some of them being subject for scalability mechanisms. Therefore the main target of mOSAIC's PaaS was to support such scalable-component-based applications. The basic notion is that of a Cloudlet – an application component that is scalable. At run-time special Containers (not necessary virtual machines, can be a smaller unit) are managing the instances of Cloudlets, watching to their health, number and communication flows. When the communication queues are not consumed in a certain rate the Container is responsible for vertical scaling (potentially, also horizontal, between Containers). Details can be found in [58, 59, 62].

Abstract the Control Interfaces of Multiple Clouds The developer of new applications or services that are consuming resources or services from a Cloud provider prefers to build a Cloud agnostic code. Abstractions are therefore needed. JClouds, SimpleCloud, δ-Cloud, LibCloud and others are performing well in providing such abstraction to a certain point. They are still dependent on the programming style of the particular Cloud resource or service. Moreover the programmer is tight to a certain programming language. mOSAIC is introducing another abstraction layer, by which components written in various languages can be part of the same application. This is possible due an inter-operability layer that is acting as a proxy between so-called Connectors – that are language dependent (but Cloud independent) and allow the description of generic operation on Cloud resource – and the so-called Drivers – that are specific for the Cloud resources (can be JClouds, LibCloud or similar, or a generic key-value store Drive). Proof-of-the-concept libraries (including Connectors) have been build for Java, Python, Erlang and Node.js. Note that event-driven style of programming was adopted mainly in order to reduce the message traffic (details in [63]). The decision were to deploy the application is postponed from the design time to the deployment time. A code that is deployed on a certain Cloud environment using mOSAIC is running in its platform environment that is taking care of the translation of the resource requests into the specific APIs of the hosting Cloud. The same application can be redeployed in another Cloud environment, but the platform components can be different in terms of Drivers that are activated.

Reuse of Existing Cloud-Aware Open-Source Software Virtual appliances are often used to quickly deploy the complex environments for certain applications. However a set of such appliances needed for a certain application or service are usually not sharing the same virtual machine. In the latest year, the deployment of certain technologies in the acquired virtual machines has been possible if the selected technologies are able or enabled to work in the virtualized environment (including communication over virtual networks between instances). Open-source technologies are re-used and adapted to communicate with its platform, like RabbitMQ for message queuing, Riak for key-value store, CouchDB for NoSQL databases, HDFS for distributed file systems, or Jetty as HTTP server. An usage example can be found in [60].

Portability mOSAIC was designed to be a deployable Cloudware that supports the portability of the codes that are intending to consume services from different Clouds. Details about portability can be found in [61]. In order to allow the component-based codes to be portable, two basic rules should be followed: (1) the operations with Cloud resources should be described in a Cloud-agnostic way; (2) communication between the components should be done using messages, preferable asynchronously. For each type of resource (queuing system, key-value store, distributed file systems, and so on), language-specific Connectors are available in the available libraries of mOSAIC, as proof-of-the-concept examples. The open standard for passing messages between components that is adopted is AMQP.

Support for Easy Development and Deployment Well known development environment should be augmented with plug-ins and features that are allowing the easy development, debugging and even deploying

of new applications or the ones for which the code is available. The mOSAIC library for Java can be used from inside Eclipse. Debugging on the developers' computer is possible if the so-called Portable Testbed Cluster (PTC) is used (simulate the Cloud on desktop; based on VirtualBox). PTC includes a simple resource allocator that support the seamless deployment of the (finally-debugged and packed) codes to a Public or Private Cloud (if the credentials for that provider are given to PTC). Apache Maven is used for Java projects and packaging. An Application Descriptor is including the list of the application components and their dependences. The Deployer Descriptor includes also specifications of concrete Cloud resources and services to be consumed. A Semantic Engine is able to assist the new incomers to find the right APIs (details in [21]).

Selection of Consumable Cloud Services or Resources A comprehensive analysis by the developers of the available resources and their interfaces exposed by Cloud providers is currently no more possible manually due to the variety and large number. Specialized assistance can come in forms of brokers, negotiators, or expert systems. mOSAIC's Cloud Agency (details in [74]) is the central point for a brokerage system; clients and providers are represented by agents. The application descriptor is analyzed and application deployers are proposed to the Clients. The SLA is a key element in this phase as well as in the running phase. New services are detected using a Dynamic Service Discoverer. Vendor modules that are connected with corresponding agents in the Agency were developed for more than ten providers.

Control of the Life-Cycle of the Applications or Services Long-life running applications are expected to benefit from Cloud computing. Their life-cycle control is needed to be performed at component level. mOSAIC comes with a Web-based interface for monitoring the application components as well as the Drivers. Any of them can be started, stopped, paused or cloned using the interface. This interface is used also by the application developers or the application user as checkpoint for the healthy of the mOSAIC PaaS that is deployed on various Cloud environments since it has features for showing the running process and the logs on different virtual machines.

Management of the Cloud Resources Virtual machines can be started automatically based on deployment descriptors. Storage and networking services can be deployed in these machines re-using the open-source deployable software, like RabbitMQ, Riak, CouchDB, MemcacheDB, REDIS (instances destroyed when virtual machines are destroyed). The control that is provided is related more to the application components that can be started (automatically using the deployment descriptor or manually from the Web-based interface of the platform), stopped, replaced or cloned. No run-time migration is supported. The developer should come back with the packed codes to the Deployer component of the platform, should start the application as for the first time, and, eventually, moves manually the data or states from one Cloud provider to another. The simultaneous usage of services from multiple Clouds is not excluded, at the level of the application, but is not completely supported by the Cloudware. If components of the same application are distributed on two Cloud sites, the Cloudware is able to offer an image of a single application: a virtual cluster is formed from the virtual machines that are acquired. Each virtual machine received a particular identifier that is related with the application name. However the key issue is the communication system. For example if RabbitMQ is used, it is based on AMQP, a protocol that uses TCP for reliable delivery. Between a Private and some Public Cloud the particular channels can be open. However not any Public Cloud allows TCP connections; in such cases RabbitMQ should be manually configured to use additional plugins, like the one for Http.

Marketplace Support mOSAIC's Cloud Agency that play the role of the market from the Inter-Cloud architecture. Clients and Providers are represented by agents in this agency (details in [74]). Multi-agent system technologies and standards are used to ensure the proper treatment of the brokerage or negotiation activities. At the deployment phase, the application developer is able to present to the Cloud Agency, using the SLA configurator and mechanism of the platform, a call-for-proposals for Cloud resources and services. Using it as well as the Cloud provider offers, and based on an multi-criterial optimization system, the Broker is able to make proposals for resource or service provisioning. After the approval of a certain offer by the application developer, the Cloud Agency is able

to reserve resources or services in the case of Private Clouds. The Public Clouds require the intervention of the developer to specify the credentials.

Cloudware that is Re-Used in Another Open-Source Multi-Cloud Middleware mOSAIC's Cloudware is one of the building blocks of the Execution platform of MODAClouds. Its improvements in terms of deployment capabilities are reported in [16]. MODAClouds provides methods, a decision support system, an open source IDE and run-time environment for the high-level design, early prototyping, semi-automatic code generation, and automatic deployment of applications on Multi-Cloud with guaranteed QoS [4]. Additionally, it should provide techniques for data mapping and synchronization among multiple Clouds. Using a model-driven approaches as well as a selection mechanism of the Cloud offers, MODAClouds is expected to offer a solution for Cloud Blueprint and Inter-Cloud.

6 Conclusions and Future Work

The young research and development field which deals with multiple Clouds' usage scenarios has already several strong actors. Commercial or research results, they are offering various options to the application and service developers and deployers. A drawback of the diversity is the fuzzy terminology that is currently used. In this context, the present paper intended to contribute to the classification of the terms and to provide a snapshot of the available products that are ready to be used. The classification that is provided is not absolute. Considerable improvements are expected to be done in the near future as the field rapidly evolves, and the taxonomy can be used in a wider context, as in building specific ontologies for multiple Clouds.

We have also identified the latest solutions enabling Multi-Clouds as well as the gaps that are needed to be filled in the near future. We applied a kind of reverse engineering method by starting from the existing libraries, tools and platforms supporting Multi-Clouds, as well the reports on the driving forces of the applications running in multiple Clouds, in order to identify the least common multiple of requirements, principles, or features to be satisfied, and which are followed by a Multi-Cloud middleware. Such wishing list is useful in the development of new tools for multiple Clouds, as well as in the identification of the gaps that must be filled by the current middleware. The exposure of the details related to the support from a particular Cloudware had the purpose to provide hints on what is currently technologically possible.

A first future step can consists in the identification of the degree of fulfillment of the proposed wishing list by the current theoretical approaches and technical solutions, and a second one, in the study and experimentation of how can they be composed in order to build a more complete enabler for a Multi-Cloud.

References

1. Akolkar, R., Chefalas, T., Laredo, J., Chang-Shing, P., Sailer, A., Schaffa, F., Silva-Lepe, I., Tao, T.T.: The future of service marketplaces in the cloud. In: IEEE Eighth World Congress on Services (SERVICES), pp. 262–269 (2012)
2. Almutairi, A.A., Sarfraz, M.I., Basalamah, S., Aref, W.G., Ghafoor, A.: A distributed access control architecture for cloud computing. IEEE Softw. **29**(2), 36–44 (2012)
3. Aoyama, T., Sakai, H.: Inter-Cloud computing. Bus. & Inf. Syst. Eng. **3**, 173–177 (2011)
4. Ardagna, D., Di Nitto, E., Casale, G., Petcu, D., Mohagheghi, P., Mosser, S., Matthews, P., Gericke, A., Ballagny, C., D'Andria, F., Nechifor, C.S., Sheridan, C.: Modaclouds: a model-driven approach for the design and execution of applications on multiple clouds. In: 2012 ICSE Workshop on Modeling in Software Engineering (MISE), pp. 50–56 (2012)
5. Ardagna, D., Balduini, M., Crăciun, C., Calcavecchia, N., Casale, G., Della Valle, E., Di Nitto, E., Perez, J.F., Sheridan, C., Wang, W.: Monitoring platform specification. MODAClouds deliverable D2.1.1, available at www.modaclouds.eu/publications/public-deliverables/
6. Ardagna, D., Casale, G., Crăciun, C., Ciavotta, M., Della Valle, E., Di Nitto, E., Gholibeigi, M., Matthews, P., Miglierina, M., Pérez, J.F., Nechifor, C.S., Sheridan, C., Wang, W.: Analysis of the state of the art and defining the scope. MODAClouds deliverable D6.1, available at www.modaclouds.eu/publications/public-deliverables/
7. Bernstein, D., Vij, D., Diamond, S.: An intercloud cloud computing economy technology, governance, and market blueprints. In: 2011 Annual SRII Global Conference, pp. 293–299 (2011)
8. Bermbach, D., Kurze, T., Tai, S.: Cloud federation: effects of federated compute resources on quality of service and cost. In: 2013 IEEE International Conference on Cloud Engineering (IC2E), pp. 31–37 (2013)
9. Bessani, A., Correia, M., Quaresma, B., André, F., Sousa, P.: DepSky: dependable and secure storage in a cloud-of-clouds. In: 2011 Sixth Conference on Computer Systems (EuroSys), pp. 31–46 (2011)
10. Bunch, C., Drawert, B., Chohan, N., Krintz, C., Petzold, L., Shams, K.: Language and runtime support for automatic

configuration and deployment of scientific computing software over cloud fabrics. J. Grid Comput. **10**(1), 23–46 (2012). doi:10.1007/s10723-012-9213-8
11. Buyya, R., Ranjan, R., Calheiros, R.N.: InterCloud: utility-oriented federation of cloud computing environments for scaling of application services. In: 2010 10th International Conference on Algorithms and Architectures for Parallel Processing (ICA3PP), vol. 6081, pp. 13–31. LNCS (2010)
12. Calheiros, R.N., Toosi, A.N., Vecchiola, C., Buyya, R.: A coordinator for scaling elastic applications across multiple clouds. Futur. Gener. Comput. Syst. **28**(8), 1350–1362 (2012)
13. Celesti, A., Tusa, F., Villari, M., Puliafito, A.: How to enhance cloud architectures to enable cross-federation. In: 2010 IEEE 3rd International Conference on Cloud Computing (CLOUD), pp. 337–345 (2010)
14. Celesti, A., Tusa, F., Villari, M., Puliafito, A.: Improving virtual machine migration in federated cloud environments. In: Second International Conference on Evolving Internet (INTERNET), pp. 61–67 (2010)
15. Celesti, A., Fazio, M., Villari, M., Puliafito, A.: Virtual machine provisioning through satellite communications in federated cloud environments. Futur. Gener. Comput. Syst. **28**(1), 85–93 (2012)
16. Crăciun, C., Panica, S., Neagul, M., Cassale, G., Perez-Bernal, J., Wang, W.: Run-time environment – proof of concept, MODAClouds deliverable D6.5.1, available at www.modaclouds.eu/publications/public-deliverables/
17. Cuomo, A., Modica, G., Distefano, S., Puliafito, A., Rak, M., Tomarchio, O., Venticinque, S., Villano, U.: An SLA-based broker for cloud infrastructures. J. Grid Comput. **11**(1), 1–25 (2013)
18. Zeginis, D., D'Andria, F., Bocconi, S., Cruz, S.G., Martin, O.C., Gouvas, P., Ledakis, G., Tarabanis, K.A.: A user-centric multi-PaaS application management solution for hybrid multi-cloud scenarios scalable computing: practice and experience **14**(1), 17–23 (2013)
19. D'Andria, F., Bocconi, S., Cruz, J.G., Ahtes, J., Zeginis, D.: Cloud4soa: multi-cloud application management across paas offerings. In: 2012 14th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), pp. 407–414 (2012)
20. Demchenko, Y., Ngo, C., De Laat, C., Garcia-Espin, J.A., Figuerola, S., Rodriguez, J., Contreras, L.M., Landi, G., Ciull, N.: Intercloud architecture framework for heterogeneous cloud based infrastructure services provisioning on-demand. In: 2013 27th International Conference on Advanced Information Networking and Applications Workshops (WAINA), pp. 777–784 (2013)
21. Di Martino, B., Cretella, G.: Towards a semantic engine for cloud applications development support. In: 2012 Sixth International Conference on Complex, Intelligent, and Software Intensive Systems (CISIS), pp. 198–203 (2012)
22. Ejarque, J., Sirvent, R., Badia, R.: A multi-agent approach for semantic resource allocation. In: 2010 IEEE Second International Conference on Cloud Computing Technology and Science (CloudCom), pp. 335–342 (2010)
23. Elmroth, E., Tordsson, J., Hernández, H., Ali-Eldin, A., Svard, S., Sedaghat, M., Li, W.: Self-management Challenges for Multi-cloud Architectures, in ServiceWave, vol. 6994, pp. 38–49. LNCS (2011)
24. Ferrer, A.J., Hernández, F., Tordsson, J., Elmroth, E., Ali-Eldin, A., Zsigri, C., Sirvent, R., Guitart, J., Badia, R.M., Djemame, K., Ziegler, W., Dimitrakos, T., Nair, S.K., Kousiouris, G., Konstanteli, K., Varvarigou, T., Hudzia, B., Kipp, A., Wesner, S., Corrales, M., Forg, N., Sharif, T., Sheridan, C.: Optimis: a holistic approach to cloud service provisioning. Futur. Gener. Comput. Syst. **28**(1), 66–77 (2012)
25. Gall, M., Schneider, A., Fallenbeck, N.: An architecture for community clouds using concepts of the intercloud. In: 2013 IEEE 27th International Conference on Advanced Information Networking and Applications (AINA), pp. 74–81 (2013)
26. Goiri, Í., Guitart, J., Torres, J.: Characterizing cloud federation for enhancing providers' profit. In: 2010 IEEE International Conference on Cloud Computing (CLOUD), pp. 123–130 (2010)
27. Goiri, Í., Guitart, J., Torres, J.: Economic model of a cloud provider operating in a federated Cloud. Inf. Syst. Front. **14**, 827–843 (2012)
28. Grozev, N., Buyya, R.: Inter-cloud architectures and application brokering: taxonomy and survey. Software Practice and Experience (2012). doi:10.1002/spe.2168
29. Gunka, A., Omerovic, A., Matthews, P., Muntés Mulero, V., DAndria, F.: Analysis of existing business models and cost/business methodology – initial version. MODAClouds deliverable D2.1.1, available at www.modaclouds.eu/publications/public-deliverables/
30. Harmer, T., Wright, P., Cunningham, C., Perrott, R.: Provider-independent use of the cloud. In: Euro-Par 2009 Parallel Processing, vol. 5704, pp. 454–465. LNCS (2009)
31. Hassan, M.M., Hossain, M.S., Sarkar, A.M.J., Huh, E.-N.: Cooperative game-based distributed resource allocation in horizontal dynamic cloud federation platform. Inf. Syst. Front. 1–20 (2012)
32. Houidi, I., Mechtri, M., Louati, W., Zeghlache, D.: Cloud service delivery across multiple cloud platforms. In: 2011 IEEE International Conference on Services Computing (SCC). pp. 741–742 (2011)
33. Jain, P., Rane, D., Patidar, S.: A novel cloud bursting brokerage and aggregation (CBBA) algorithm for multi cloud environment. In: 2012 Second International Conference on Advanced Computing and Communication Technologies (ACCT), pp. 383–387 (2012)
34. Junker, F., Vogel, J., Stanoevska, K.: Aggregating price models for composite services in cloud service marketplaces. In: 2012 IEEE 10th International Symposium on Parallel and Distributed Processing with Applications (ISPA), pp. 479–486 (2012)
35. Kang, J., Sim, K.M.: Cloudle: a multi-criteria cloud service search engine. In: 2010 IEEE Asia-Pacific Services Computing Conference (APSCC), pp. 339–346 (2010)
36. Keahey, K., Tsugawa, M., Matsunaga, A., Fortes, J.: Sky computing. IEEE Internet Comput. **13**(5), 43–51 (2009)
37. Kecskemeti, G., Kertesz, A., Marosi, A., Kacsuk, P.: Interoperable resource management for establishing federated clouds. In: Achieving Federated and Self-Manageable Cloud Infrastructures: Theory and Practice, IGI Global, pp. 18–35 (2012)

38. Kertesz, A., Kecskemeti, G., Oriol, M., Kotcauer, P., Acs, S., Rodrguez, M., Merc, O., Marosi, A.C., Marco, J., Franch, X.: Enhancing federated cloud management with an integrated service monitoring approach. J. Grid Comput. **11**(4) (2013). in print
39. Leavitt, N.: Hybrid clouds move to the forefront. Computer **46**(5), 15–18 (2013)
40. Lee, C.A.: An open cloud computing interface status update (and roadmap dartboard) cloud interoperability roadmaps sessions. OMG Technical meeting (2009)
41. Lewis, G.A.: Role of standards in cloud-computing interoperability. In: 2013 46th Hawaii International Conference on System Sciences (HICSS), pp. 1652–1661 (2013)
42. Li, A., Yang, X., Kandula, S., Zhang, M.: Cloudcmp: comparing public cloud provider. In: 10th ACM SIGCOMM conference on Internet measurement (IMC), pp. 1–14 (2010)
43. Li, J., Li, B., Du, Z., Meng, L.: CloudVO: building a secure virtual organization for multiple clouds collaboration. In: 2010 11th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), pp. 181–186 (2011)
44. Loutas, N., Peristeras, V., Bouras, T., Kamateri, E., Zeginis, D., Tarabanis, K.: Towards a reference architecture for semantically interoperable clouds. In: 2010 IEEE Second International Conference on Cloud Computing Technology and Science (CloudCom), pp. 143–150 (2010)
45. Loutas, N., Kamateri, E., Tarabanis, K.: A semantic interoperability framework for cloud platform as a service. In: 2011 IEEE Third International Conference on Cloud Computing Technology and Science (CloudCom), pp. 380–387 (2011)
46. Loutas, N., Kamateri, E., Bosi, F., Tarabanis, K.: Cloud computing interoperability: the state of play. In: 2011 IEEE Third International Conference on Cloud Computing Technology and Science (CloudCom), pp. 752–757 (2011)
47. Lucas-Simarro, J.L., Moreno-Vozmediano, R., Montero, R.S., Llorente, I.M.: Scheduling strategies for optimal service deployment across multiple clouds. Futur. Gener. Comput. Syst. **29**(6), 1431–1441 (2013)
48. Machado, G.S., Hausheer, D., Stiller, B.: Considerations on the interoperability of and between cloud computing standards. In: OGF27: G2C-Net (2009)
49. Mandal, A., Xin, Y., Baldine, I., Ruth, P., Heerman, C., Chase, J., Orlikowski, V., Yumerefendi, A.: Provisioning and evaluating multi-domain networked clouds for hadoop-based applications. In: 2011 IEEE Third International Conference on Cloud Computing Technology and Science (CloudCom), pp. 690–697 (2011)
50. Moreno-Vozmediano, R., Montero, R.S., Llorente, I.M.: IaaS cloud architecture: from virtualized data centers to federated cloud infrastructures, prePrint. Computer **45**(12), 65–72 (2012)
51. Moscato, F., Aversa, R., Di Martino, B., Petcu, D., Rak, M., Venticinque, S.: An ontology for the cloud in mosaic. In: Cloud Computing: Methodology, Systems, and Application, pp. 467–486. CRC Press (2011)
52. Munteanu, V.I., Fortiş, T.-F., Copie, A.: Building a cloud governance bus. Int. J. Comput. Commun. and Control **7**(5), 900–906 (2012)
53. NIST: Cloud computing standards roadmap-version 1.0. Special 9 Publication, 500–291 (2011)
54. Papaioannou, T.G., Bonvin, N., Aberer, K.: Scalia: an adaptive scheme for efficient multi-cloud storage. In: International Conference on High Performance Computing, Networking, Storage and Analysis (SC), Art. 20 (2012)
55. Papazoglou, M.P.: Cloud blueprints for integrating and managing cloud federations. In: Software Service and Application Engineering, vol. 7365, pp. 102–119. LNCS (2012)
56. Paraiso, F., Haderer, N., Merle, P., Rouvoy, R., Seinturier, L.: A federated Multi-Cloud PaaS infrastructure. In: 2012 IEEE Fifth International Conference on Cloud Computing (CLOUD), pp. 392–399 (2012)
57. Petcu, D., Crăciun, C., Neagul, M., Panica, S., Di Martino, B., Venticinque, S., Rak, M., Aversa, R.: Architecturing a sky computing platform. In: Towards a Service-Based Internet. ServiceWave 2010 Workshops, vol. 6569, pp. 1–13. LNCS (2011)
58. Petcu, D., Crăciun C, Rak, M.: Towards a cross platform cloud api: components for cloud federation. In: 2011 1st International Conference on Cloud Computing and Services Science (CLOSER), pp. 166–169 (2011)
59. Petcu, D., Crăciun, C., Neagul, M., Lazcanotegui, I., Rak, M.: Building an interoperability api for sky computing. In: 2011 International Conference on High Performance Computing and Simulation (HPCS), pp. 405–411 (2011)
60. Petcu, D., Frîncu, M., Crăciun, C., Panica, S., Neagul, M., Macariu, G.: Towards open-source cloudware. In: 2011 Fourth IEEE International Conference on Utility and Cloud Computing (UCC), pp. 330–331 (2011)
61. Petcu, D., Macariu, G., Panica, S., Crăciun C.: Portable cloud applications – from theory to practice. Futur. Gener. Comput. Syst. **29**(6), 1417–1430 (2012)
62. Petcu, D., Şandru, C.: Towards component-based software engineering of cloud applications. In: Joint 10th Working IEEE/IFIP Conference on Software Architecture and 6th European Conference on Software Architecture, Companion Volume, pp. 80–81 (2012)
63. Petcu, D., Panica, S., Şandru, C., Crăciun, C., Neagul, M.: Experiences in building an event-driven and deployable platform as a service. In: Web Information Systems Engineering (WISE), vol. 7651, pp. 666–672. LNCS (2012)
64. Petcu, D.: Multi-cloud: expectations and current approaches. In: 2013 international workshop on Multi-cloud applications and federated clouds (MultiCloud), pp. 1–6 (2013)
65. Petcu, D.: On the interoperability in multiple clouds. In: 3rd International Conference on Cloud Computing and Services Science (CLOSER), pp. 581–590 (2013)
66. Petcu, D., Di Martino, B., Venticinque, S., Rak, M., Mahr, T., Lopez, G., Brito, F., Cossu, R., Stopar, M., Sperka, S., Stankovski, V.: Experiences in building a mosaic of clouds. J. Cloud Comput. Adv. Syst. Appl **2**(1), 12 (2013)
67. Prodan, R., Wieczorek, M., Fard, H.M.: Double auction-based scheduling of scientific applications in distributed

Grid and Cloud environments. J. Grid Comput. **9**(4), 531–548 (2011)
68. Rimal, B.P., Jukan, A., Katsaros, D., Goeleven, Y.: Architectural requirements for cloud computing systems: an enterprise cloud approach. J. Grid Comput. **9**(1), 3–26 (2011)
69. Riteau, P.: Building dynamic computing infrastructures over distributed clouds. In: 2011 IEEE International Symposium on Parallel and Distributed Processing Workshops and Phd Forum (IPDPSW), pp. 2097–2100 (2011)
70. Singhal, M., Chandrasekhar, S., Tingjian, G., Sandhu, R., Krishnan, R., Gail-Joon, A., Bertino, E.: Collaboration in multicloud computing environments: framework and security issues. Computer **46**(2), 76–84 (2013)
71. Sotiriadis, S., Bessis, N., Kuonen, P., Antonopoulos, N.: The inter-cloud meta-scheduling (ICMS) framework. In: 2013 IEEE 27th International Conference on Advanced Information Networking and Applications (AINA), pp. 64–73 (2013)
72. Teckelmann, R., Reich, C., Sulistio, A.: Mapping of cloud standards to the taxonomy of interoperability in iaas. In: 2011 IEEE Third International Conference on Cloud Computing Technology and Science (CloudCom), pp. 522–526 (2011)
73. Tordsson, J., Montero, R.S., Moreno-Vozmediano, R., Llorente, I.M.: Cloud brokering mechanisms for optimized placement of virtual machines across multiple providers. Futur. Gener. Comput. Syst. **28**(2), 358–367 (2012)
74. Venticinque, S., Tasquier, L., Di Martino, B.: Cloud computing interface and architecture for agents based resource provisioning and management. In: 2012 Sixth International Conference on Complex, Intelligent, and Software Intensive Systems (CISIS), pp. 249–256 (2012)
75. Villari, M., Tusa, F., Celesti, A., Puliafito, A.: How to federate VISION clouds through saml/shibboleth authentication. In: 2012 European Conference on Service-Oriented and Cloud Computing (ESOCC), vol. 7592, pp. 259–274. LNCS (2012)
76. Villegas, D., Bobroff, N., Rodero, I., Delgado, J., Liu, Y., Devarakonda, A., Fong, L., Sadjadi, S.M., Parashar, M.: Cloud federation in a layered service model. J. Comput. Syst. Sci. **78**, 1330–1344 (2012)
77. Williams, D., Jamjoom, H., Weatherspoon, H.: Plug into the supercloud. IEEE Internet Comput. **17**(2), 28–34 (2013)
78. Yang, X., Nasser, B., Surridge, M., Middleton, S.: A business-oriented cloud federation model for real-time applications. Futur. Gener. Comput. Syst. **28**(8), 1158–1167 (2012)