SOFTWARE – PRACTICE AND EXPERIENCE
Softw. Pract. Exper. 2014; **44**:369–390
Published online 12 December 2012 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/spe.2168

Inter-Cloud architectures and application brokering: taxonomy and survey

Nikolay Grozev*,† and Rajkumar Buyya

Cloud Computing and Distributed Systems (CLOUDS) Laboratory, Department of Computing and Information Systems, The University of Melbourne, Victoria 3010, Australia

SUMMARY

Although Cloud computing itself has many open problems, researchers in the field have already made the leap to envision Inter-Cloud computing. Their goal is to achieve better overall Quality of Service (QoS), reliability and cost efficiency by utilizing multiple clouds. Inter-Cloud research is still in its infancy, and the body of knowledge in the area has not been well defined yet. In this work, we propose and motivate taxonomies for Inter-Cloud architectures and application brokering mechanisms. We present a detailed survey of the state of the art in terms of both academic and industry developments (20 projects), and we fit each project onto the discussed taxonomies. We discuss how the current Inter-Cloud environments facilitate brokering of distributed applications across clouds considering their nonfunctional requirements. Finally, we analyse the existing works and identify open challenges and trends in the area of Inter-Cloud application brokering. Copyright © 2012 John Wiley & Sons, Ltd.

Received 11 September 2012; Accepted 31 October 2012

KEY WORDS: Cloud computing; Inter-Cloud; Multi-Cloud; Federation; Application brokering

1. INTRODUCTION

Cloud computing has been attracting the interest of the Information and Communications Technology community since 2007 resulting in a massive amount of industry developments. It has been seen as the next logical step after Grid computing on the way to utility computing, where computing resources are available as subscription utility service just like water and electricity. Cloud computing has been defined as a type of parallel and distributed system providing virtualized resources in a pay-as-you-go fashion over the Internet [1–3]. From business perspective, it is widely viewed as an economical model for renting technical resources [4]. Being able to rent resources on demand and to avoid upfront investments in hardware and licenses proves to be very enticing for enterprises in a dynamic and unstable business environment.

However, the standard Cloud computing model, where a client utilizes a single cloud data centre, introduces several challenges. A cloud service unavailability can leave thousands of customers relying solely on it without access to essential and paid for resources. Also relying on a single cloud data centre makes it hard to implement adequate responsiveness and usability to clients distributed worldwide. These factors preclude the usage of multiple clouds (i.e. an Inter-Cloud) to achieve better QoS, reliability and flexibility.

The first academic publications about Inter-Clouds appeared a couple of years after the advent of Cloud computing and include the following seminal works – [5–9]. The term Inter-Cloud has been

*Correspondence to: Nikolay Grozev, Cloud Computing and Distributed Systems (CLOUDS) Laboratory, Department of Computing and Information Systems, The University of Melbourne, Victoria 3010, Australia.

†E-mail: ngrozev@student.unimelb.edu.au

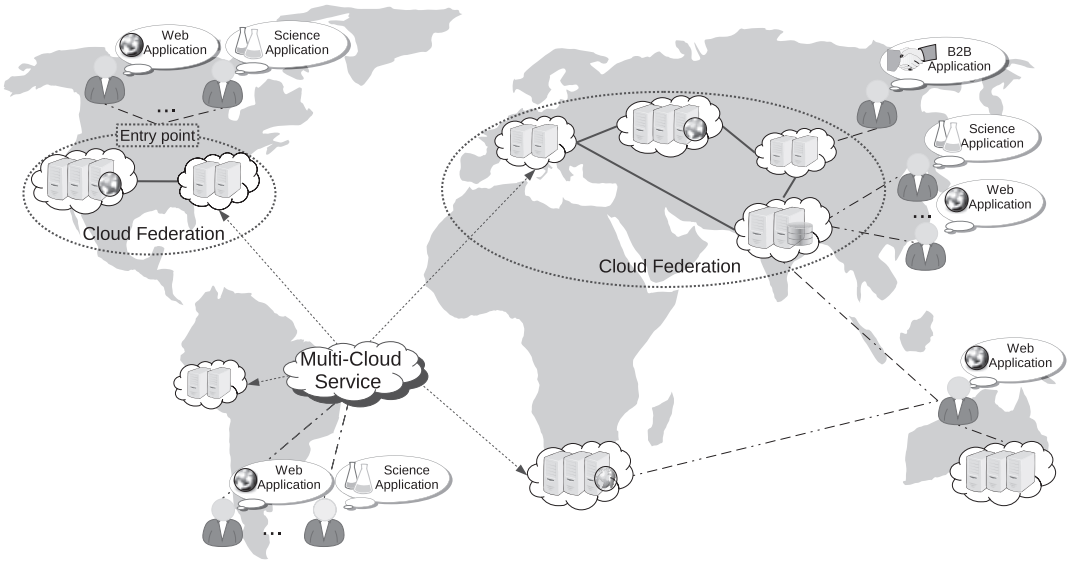

Figure 1. An overview of Inter-Cloud approaches and use cases.

described as a cloud of clouds [10], and a formal definition is provided in Section 2. Essentially, an Inter-Cloud allows for the dynamic coordination and distribution of load among a set of cloud data centres – see Figure 1.

The benefits of an Inter-Cloud environment for cloud clients are numerous and can be broadly summarized as follows:

- *Diverse geographical locations*. Leading cloud service providers have established data centres worldwide. However, it is unlikely that any provider will be able to establish data centres in every country and administrative region [5]. Many applications have legislative requirements as to where data are stored. Thus, a data centre within a region of countries may not be enough, and application developers will need fine-grained control (specific country or state) as to where resources are positioned. Only by utilizing multiple clouds can one gain access to so widely distributed resources and provide well-performing and legislation-compliant services to clients.
- *Better application resilience*. During the past several years, there have been several cases of cloud service outages, including ones of major vendors [11–14]. The implications from one of Amazon's data centres failure were very serious for customers who relied on that location only. In a post-mortem analysis, Amazon advised their clients to design their applications to use multiple data centres for fault tolerance [11]. Furthermore, in Berkeley's report on Cloud computing, Armbrust *et al.* emphasise that potential unavailability of service is the number one inhibitor to adopting Cloud computing [15]. Thus, they advise the use of multiple providers. Besides fault tolerance, using resources from different providers acts as an insurance against a provider being stopped because of regulatory or legal reasons as well.
- *Avoidance of vendor lock-in*. By using multiple clouds and being able to freely transit workload among them, a cloud client can easily avoid vendor lock-in. In case a provider changes a policy or pricing that impact negatively its clients, they could easily migrate elsewhere.

In essence, the main value of an Inter-Cloud environment for cloud customers is that they can diversify their infrastructure portfolio in terms of both vendors and location. Thus, they could make their businesses more adaptable to vendors' policy and availability changes and easily expandable in new legislative regions.

Cloud service providers may also have significant incentives from participating into an Inter-Cloud initiative. A paramount idea of Cloud computing is that a cloud service should deliver

DOI: 10.1002/spe

constant availability, elasticity and scalability to meet the agreed customers' requirements [4]. A cloud provider should ensure enough resources at all times. But how much is enough? Workload spikes can come unexpectedly, and thus, cloud providers need to overprovision resources to meet them. Another issue is the huge amount of data centre power consumption [1, 16]. Keeping an excess of resources in a ready to use state at all times for coping with unexpected load spikes leads to increased power consumption and cost of operation. Cloud providers' benefits can be summarized as follows:

- *Expand on demand.* Being able to offload to other clouds, a provider can scale in terms of resources like cloud-hosted applications do within a cloud. A cloud should maintain in a ready to use state enough resources to meet its expected load and a buffer for typical load deviations. If the workload increases beyond these limits, resources from other clouds can be leased [5].
- *Better service level agreement (SLA) to customers.* Knowing that even in a worst-case scenario of data centre outage or resource shortage the incoming workload can be moved to another cloud, a cloud provider can provide better SLAs to customers.

However, achieving all these benefits for both cloud providers and clients should be done without violating applications' requirements. Appropriate application brokering (consisting of provisioning and scheduling) should honour the requirements in terms of performance, responsiveness and legal considerations. Existing approaches to achieve this vary in terms of architecture, mechanisms and flexibility.

In this work, we investigate and classify Inter-Cloud environments and application brokering mechanisms. We focus on identifying and analysing the coarse-grained requirements and the state of the art in Inter-Cloud brokering of distributed applications. With this, we perform analysis of the status quo and identify trends and open issues in the area. To the best of our knowledge, this novel area of research has not been systematically surveyed before.

Because the area of Inter-Clouds is novel and there is ambiguity about some of the terms in Section 2, we discuss and define the main ones. In Section 3, we motivate and introduce an architectural taxonomy of existing Inter-Cloud developments. In Section 4, we classify existing approaches for brokering applications across clouds. In Section 5, we classify existing distributed applications, and for every major class of applications, we investigate what are the requirements for Inter-Cloud brokering. In Section 6, we review 20 major developments in the area fitting them into the previous taxonomies and discussing their capabilities to facilitate Inter-Cloud application brokering. Section 7 provides an analysis of the state of the art. The final Section 8 concludes the study and defines avenues for future work.

2. DEFINITIONS

Cloud computing is a novel area of research and still faces certain terminological ambiguity. The area of Inter-Clouds is even newer, and many works in the area use several terms interchangeably. In this section, we elaborate on their meaning.

Inter-Cloud computing has been formally defined as [17]

> A cloud model that, for the purpose of guaranteeing service quality, such as the performance and availability of each service, allows on-demand reassignment of resources and transfer of workload through a [*sic*] interworking of cloud systems of different cloud providers based on coordination of each consumers requirements for service quality with each providers SLA and use of standard interfaces.

In the rest of this work, we will adhere to this definition. The seminal works on Inter-Clouds by Buyya *et al.* [5] and Bernstein *et al.* [6] also implicitly express similar definitions. Buyya *et al.* emphasise the just-in-time, opportunistic nature of the provisioning within an Inter-Cloud that allows for achieving QoS and quality of experience targets in a dynamic environment [5]. The term *Cloud Fusion* has been used by Fujitsu Laboratories to denote a similar notion [18].

DOI: 10.1002/spe

Note that this definition is generic and does not specify who is initiating the Inter-Cloud endeavour – the cloud providers or the clients. Also, it does not specify whether cloud providers collaborate voluntarily to form an Inter-Cloud or not. Two other terms are used throughout the related literature to differentiate between these – *Federation* and *Multi-Cloud*. A Federation is achieved when a set of cloud providers voluntarily interconnect their infrastructures to allow sharing of resources among each other [8, 19, 20]. The term Multi-Cloud denotes the usage of multiple, independent clouds by a client or a service. Unlike a Federation, a Multi-Cloud environment does not imply volunteer interconnection and sharing of providers' infrastructures. Clients or their representatives are directly responsible for managing resource provisioning and scheduling [19]. The term *Sky Computing* has been used in several publications with similar meaning [7, 21]. Both Federations and Multi-Clouds are types of Inter-Clouds. We discuss them further in Section 3.

Another term used in the related literature is *Hybrid Cloud*. It has been defined as a composition of two or more different cloud infrastructures – for example, a private and a public cloud [3]. Thus, a Hybrid Cloud is a type of a Multi-Cloud that connects miscellaneous clouds in terms of their deployment models. Often, Hybrid Clouds are used for *cloud bursting* – the usage of external cloud resources when local ones are insufficient.

Throughout the literature, the term *Inter-Cloud broker* has been used with different meanings. In most cases, it means a service that acts on behalf of the client to provision resources and deploy application components [5, 19, 22]. We adhere to this general idea and define an *application broker* as an automated entity with the following responsibilities:

- Automatic resource provisioning and management across multiple clouds for a given application. Typically, this would include allocation and deallocation of resources (e.g. virtual machines (VMs) and storage).
- Automatic deployment of application components in the provisioned resources.
- Scheduling and load balancing of the incoming requests to the allocated resources.

3. ARCHITECTURAL TAXONOMY

By definition, Inter-Cloud computing is an endeavour that implies interconnecting multiple cloud providers' infrastructures. The extent to which providers would voluntarily lend their infrastructure within an Inter-Cloud depends on the political and financial incentives they have to do so. Figure 2 depicts how cloud infrastructures can be classified on the basis of their ownership.

On the first level, we differentiate between *Governmental* and *Private* cloud infrastructures. These can be described as:

- *Governmental* – owned and utilized by a government or nonprofit institution. Examples for this are science community clouds like Australia's National eResearch Collaboration Tools and Resources (NeCTAR) [23] and Canada's Cloud-Enabled Space Weather Modelling and Data Assimilation Platform (CESWP) [24].
- *Private* – owned by a private organisation.

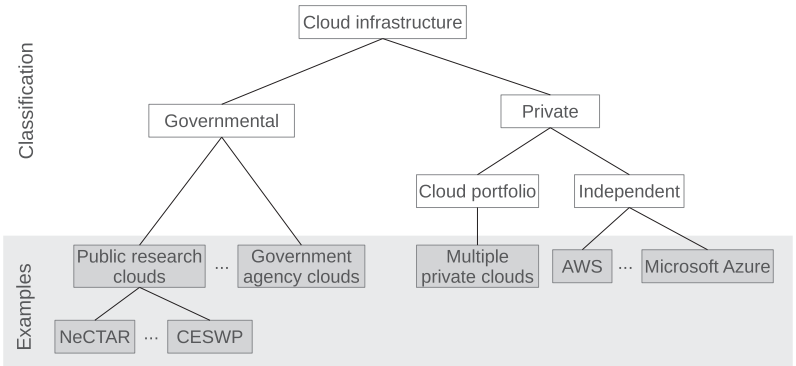

Figure 2. Classification of cloud infrastructures.

DOI: 10.1002/spe

Private clouds can be further classified as follows:

- *Cloud portfolio* – when the cloud is a part of a portfolio of clouds belonging to the same organisation. Examples for this are multiple private clouds belonging to a corporation.
- *Independent* – separate cloud infrastructure that is not a part of a portfolio of clouds.

Given that classification, we argue that *Independent* private clouds are less likely to participate voluntarily in an Inter-Cloud initiative. Such cloud providers will be reluctant to scale/transit their workload in the data centres of their competitors. Also, such providers may not be willing to provide access through unified APIs to their services in a federated market place because that would allow their customers to migrate easily and dynamically to competitors. This can be achieved by using specialized third party services and APIs, which will be discussed in details later.

For example, despite their data centre outages leaving customers without access to services for substantial periods of time, none of the leading commercial providers (like Amazon, Google and Microsoft) has announced plans to utilize external cloud resources to avoid future outages.

On the other hand, *Governmental* clouds are very likely to participate voluntarily within an Inter-Cloud among each other because that could benefit the quality of the overall public service. Similarly, *portfolio clouds* are likely to form an Inter-Cloud initiative among each other because they are not directly competing with each other and have clear incentives to collaborate.

With this observation, we can broadly classify Inter-Clouds as follows:

- *Volunteer Federation* – when a group of cloud providers voluntarily collaborate with each other to exchange resources. As identified, this type of Inter-Cloud is mostly viable for governmental clouds or private cloud portfolios.
- *Independent* – when multiple clouds are used in aggregation by an application or its broker. This approach is essentially independent of the cloud provider and can be used to utilize resources from both governmentally and private clouds. Another term used for this is *Multi-Cloud*.

From architectural perspective, *Volunteer Federations* can be further classified as follows:

- *Centralised* – in every instance of this group of architectures, there is a central entity that either performs or facilitates resource allocation. Usually, this central entity acts as a repository where available cloud resources are registered but may also have other responsibilities like acting as a market place for resources.
- *Peer-to-Peer* – in the architectures from this group, clouds communicate and negotiate directly with each other without mediators.

Independent Inter-Cloud developments can be further classified as follows:

- *Services* – application provisioning is carried out by a service that can be hosted either externally or in-house by the cloud clients. Most such services include broker components in themselves. Typically, application developers specify an SLA or a set of provisioning rules, and the service performs the deployment and execution in the background, in a way respecting these predefined attributes.

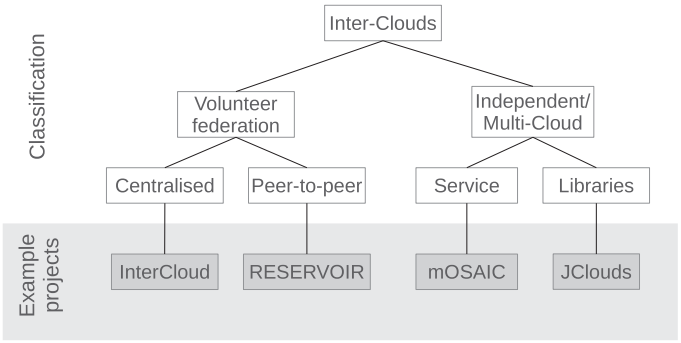

Figure 3. Architectural classification of Inter-Clouds.

DOI: 10.1002/spe

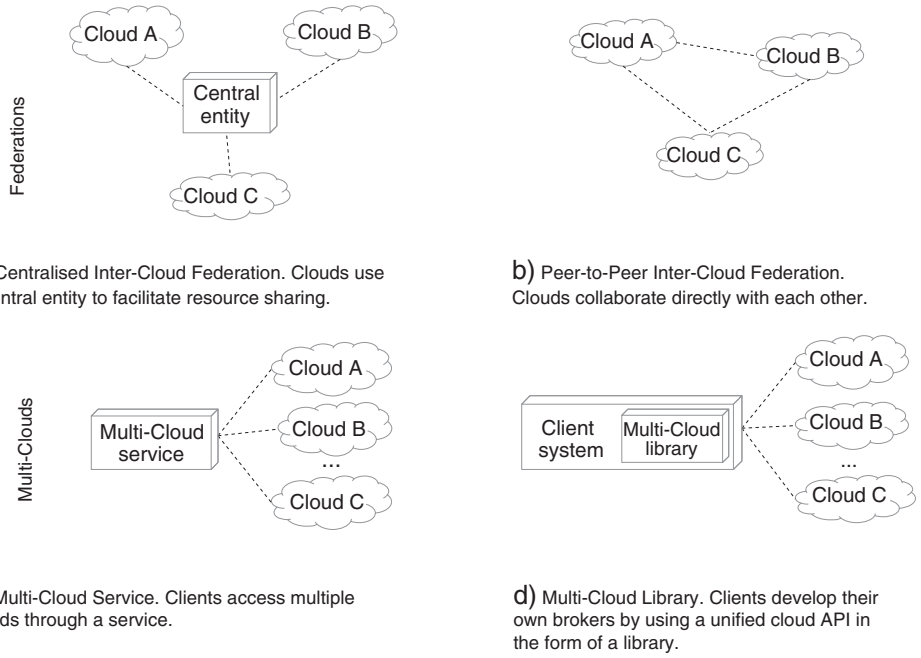

Figure 4. Inter-Cloud developments' architectures.

- *Libraries* – often, custom application brokers that directly take care of provisioning and scheduling application components across clouds are needed. Typically, such approaches make use of Inter-Cloud libraries that facilitate the usage of multiple clouds in a uniform way.

The whole taxonomy of developments is depicted in Figure 3. Figure 4 depicts the architectures from the taxonomy. In Section 6, we discuss in details many examples of developments in all these areas. Note that the discussed herein classifications are orthogonal to the general classification of Cloud computing services as Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS) as defined by NIST [3]. For example, cloud providers within a federation can exchange resources at the infrastructure level (e.g. VMs) but at the same time provide PaaS to clients. Also in theory, a Multi-Cloud service may access IaaS, PaaS or SaaS cloud services on behalf of its clients.

4. TAXONOMY OF INTER-CLOUD APPLICATION BROKERING MECHANISMS

There are several mechanisms for implementing application-specific brokering as depicted in Figure 5. At the first level, we differentiate as to who is responsible to provide implementation of the application broker. Brokers using *externally managed* mechanisms usually are not hosted in-house and are a part of the system entities that facilitate the Inter-Cloud. In the case of *Multi-Cloud services*, application brokers are often part of the service providing access to the set of clouds. In the case of *Volunteer Federations*, application brokering is implemented either in a centralized entity or by the cloud providers. In essence, *externally managed* brokers are transparent to the application developers and provide some 'hooks' for application-specific logic.

On the basis of the way application-specific logic is specified, we can further classify *externally managed* brokering mechanisms as follows:

- *SLA based* – application developers specify the brokering requirements in an SLA in the form of constraints and objectives. The cloud provider or the Inter-Cloud service acting on behalf of the client decides on brokering approach honouring the specified SLA.
- *Trigger-Action* – the application developers specify a set of triggers and associate one or more actions to each of them. A trigger becomes active when a predefined condition considering

DOI: 10.1002/spe

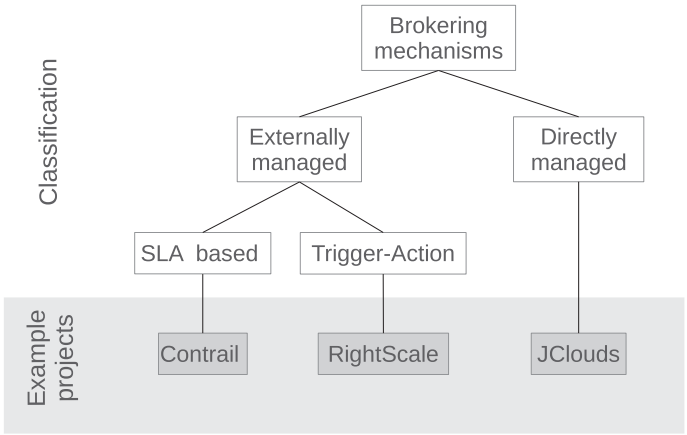

Figure 5. Application brokering mechanisms.

the externally visible application performance indicators becomes true. An action is executed when a correspondent trigger becomes active. Actions usually include scale up/down or other provisioning activities. For example, a trigger may become active if the number of active TCP/IP sessions becomes more than 50 or if the memory consumption of an allocated VM exceeds 512 MB. An associated action may be the allocation of a new VM.

The main benefit of the *SLA-based* brokering approaches is that they are completely transparent to the application developers. The provider or the service takes care of the provisioning behind the scenes.

However, in *SLA-based* brokering, clients do not have direct control on how their applications are provisioned across clouds. Thus, there needs to be a certain level of trust between the two sides. Also, to specify the provisioning requirements of an application, one needs mature SLA specification formalisms and services. In a report from The Cloud Standards Customer Council, it was emphasised that SLA contracts offered by current cloud providers are immature [25]. Often, clients are offered only nonnegotiable standard SLA contracts, thus limiting their ability to define application-specific clauses. Most such contracts only specify performance related clauses and do not allow for provisioning restrictions, for example, which location to use for data storage. Thus, the adoption of *SLA-based* brokering approaches depends on the advancements of providers' and mediators' SLA offerings.

The *Trigger-Action* approach is less transparent than the *SLA-based* one, because application developers need to specify the exact scalability and provisioning rules. This gives a finer grained control about how the application behaves.

The *directly managed* brokering mechanisms are mostly used when there is no mediator between the application and the set of utilized clouds. *Directly managed* brokers are hosted separately and need to keep track of the performance characteristics of the application themselves. It is the responsibility of the application developers to develop such brokers in a way that meets the availability and dependability requirements.

5. APPLICATION CENTRIC PERSPECTIVE TO INTER-CLOUDS

Clouds as a deployment environment are general purpose and cover the whole spectrum of distributed applications. Cloud providers have always delivered computing resources to enterprises. Thus, many database centric enterprise applications are deployed within clouds. Being an inherently distributed environment, clouds are also suitable for standard Grid and Cluster job-based applications. Research community clouds like Australia's NeCTAR [23] and Canada's CESWP [24] have made the use of clouds for resource intensive jobs apparent, although there are still concerns about the worse performance of cloud clusters compared with physical ones as reported by

DOI: 10.1002/spe

Jackson *et al.* [26]. Developments like EC2 Cluster Compute Instance [27] are supposed to mitigate such issues. It is logical to expect that the same spectrum of applications will be targeted for the Inter-Cloud.

In this section, we perform a taxonomic analysis to identify classes of distributed applications with similar characteristics. Then we continue to identify the common nonfunctional requirements of these application classes with respect to the Inter-Cloud environment in which they are deployed.

5.1. Taxonomy of Inter-Cloud applications

Figure 6 depicts the taxonomy of typical distributed applications deployable in Inter-Cloud environment that will be discussed from now on. At the first level, we differentiate between *batch processing* and *interactive* applications. *Batch processing* applications allow the users to submit and execute jobs, which then run to completion without further user input. Thus, they can also be thought of as job-based applications. *Batch processing* applications can be further classified as allowing *singular* or *periodical* jobs.

Singular jobs are executed only once, unless they are rerun by the users or automatically rescheduled upon failure. This branch of the taxonomy represents the typical Grid and Cluster resource intensive applications. Such applications are used by research, industry and military organisations to solve optimization models, build complex simulations and so on. Most of the applications in this branch fall into the High Performance Computing (HPC), High Throughput Computing (HTC) or Many-task Computing (MTC) categories. Raicu *et al.* defines these three types of applications and discusses their differences and similarities [28]. *Scientific workflows* (further discussed by Barker and Van Hemert [29]) are also a kind of *singular* jobs, although they are composed of multiple subjobs/tasks, because they appear to the user as a single unified job.

Singular jobs typically require fast connection between the processing nodes and are less likely to be deployed across clouds. Exceptions to this are high-throughput computing applications, where independent long running tasks may be scheduled across clouds. Another reason for scheduling a job in multiple clouds is fault tolerance – if a job fails in one cloud environment, it may succeed in another.

Periodical jobs are repeatedly executed over a period of time. Most of them are *data centric* and perform long running tasks over huge enterprise databases. Periodic Extract Transform Load (ETL) jobs in data warehouses, reporting jobs and Big Data analytical jobs are examples of this. *Data centric* periodical jobs are of great significance for enterprises with large-scale information systems. They are often scheduled to run at nights or weekends to ensure smooth business operation during business hours. *Administrative* periodical jobs are relatively simple and take care of maintaining the system in a healthy and running state. Examples are scripts that periodically consolidate log files from all nodes or release unneeded resources across nodes.

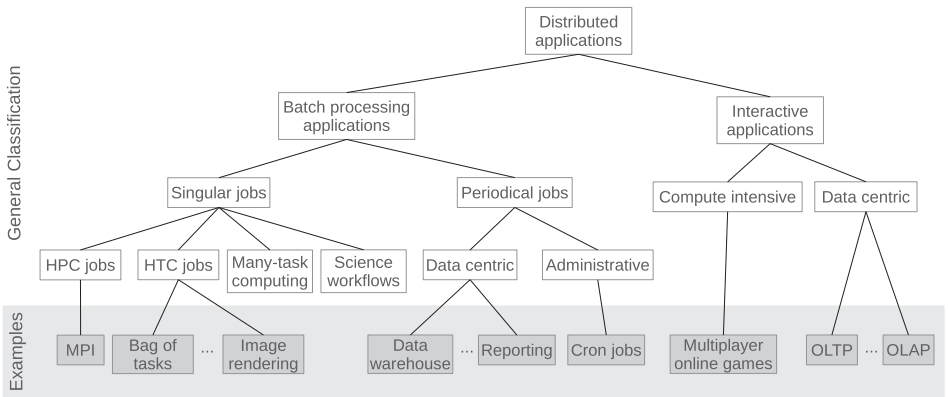

Figure 6. Classification of distributed applications.

DOI: 10.1002/spe

Data centric periodical jobs typically need to be executed close to where the corresponding data are stored. Thus, Inter-Cloud scheduling and provisioning are sensible when the persistent data are dispersed in multiple clouds.

Interactive applications are also known as *Online* applications. Unlike the *batch processing* ones they imply constant interaction and input from the user. Most interactive distributed systems are *data centric*. That is, their main goal is to facilitate user access to a persistent storage (most often a relational database). Typical examples are Online Transaction Processing applications. Another group of *interactive* applications are the *compute intensive* ones. Examples for such are multilayer online games.

Unlike *batch processing* applications, *interactive* ones need to be constantly available and thus would benefit the most from using multiple clouds. Also, many online applications serve users from around the world and thus would benefit from the geographical diversity of an Inter-Cloud's data centres.

5.2. Requirements for Inter-Cloud environments

As discussed, the main benefit from the clients' perspective of an Inter-Cloud environment is the diversification in terms of vendors and locations. It is however important that this is carried out without violating the nonfunctional application requirements. Not all Inter-Cloud environments allow for such control. Thus, in the rest of this section, we define several requirements for Inter-Cloud environments and discuss for which of the main classes of applications these are important. Table I summarizes the discussion.

Data location awareness. Ideally, the persistent data and the processing units of an application should be in the same data centre (even on the same rack) and should be connected with a high-speed network. If they reside on remote clouds, the performance penalty could be grave.

This consideration is especially important for interactive applications and periodical jobs using huge databases. Actually in this case, the database considerations are becoming a driving factor in Inter-Cloud brokering. The databaseoften becomes the performance bottleneck of a distributed enterprise application. Additional approaches like distributed caching and sharding are often used to mitigate to some extent the performance issues of traditional databases [30].

In recent years, there has been a trend to move away from standard relational databases to the so-called NoSQL and NewSQL ones. There has been significant amount of developments in these fields (Cattel reports on more than 20 projects [31]). The term NoSQL denotes a broad range of nonrelational database developments. In essence, the NoSQL approach aims at balancing between consistency and partition tolerance following the result of the famous CAP theorem [32, 33], whereas NewSQL approaches keep to standard relational approaches but discourage or ban performing certain inefficient operations [31].

Many enterprise systems will need to use several types of data sources simultaneously. For example, payment information may be stored in a transactional relational database to ensure consistency, but other data like catalogues of products may be stored in a NoSQL storage for better performance. This phenomenon has been termed *polyglot persistence* by several practitioners [34, 35].

Table I. Typical Inter-Cloud brokering requirements.

	Job based		Interactive applications	
Awareness	Singular jobs	Periodical jobs	Compute intensive	Data centric
Data location	✓	✓		✓
Geo-location			✓	✓
Pricing	✓	✓	✓	✓
Legislation/policy	✓	✓		✓
Local resources	✓	✓	✓	✓

DOI: 10.1002/spe

But why is this important in terms of Inter-Cloud brokering? An Inter-Cloud broker of a data-centric application is essentially responsible for allocating resources on multiple clouds and scheduling incoming requests to these clouds appropriately. Developers need the flexibility to implement brokering policies considering the structure and the location of distributed persistent data and cache servers. Thus, a successful broker should be well integrated with its application and should have feedback from it regarding the structure and location of data.

Most singular job-based applications are compute-intensive programs and thus avoid substantial input/output operations. Some of them however work on datasets that have been extracted in advance, for example, by conducting experiments. In such cases, the data are usually stored in standalone persistent storage, for example, relational database or a data file. Typically, either the data are replicated to the node running the job or the job is scheduled on a node near the data. The latter is preferred when the dataset has substantial size. There has been substantial work in the last decades on data aware scheduling of singular jobs in Grids, and it has been well summarized by Dong and Akl [36]. To apply these approaches in an Inter-Cloud environment, the location of the data (or its replicas) is needed.

Geo-location awareness. Besides database considerations, the location of the incoming requests is also important to consider when brokering interactive applications. Ideally, requests should be scheduled near the geographical location of their origin to achieve better performance. This is not usually considered when brokering jobs, because the users submit the jobs only once and do not interact with the system until they finish. In interactive systems however, network bandwidth latency caused by geographical distances may hurt the user experience significantly.

Pricing awareness. A common consideration for all types of applications deployed in multiple clouds is the pricing. Different providers have different pricing policies and mechanisms, and minimizing the overall hosting price is a nontrivial task. An application broker would need detailed and up to date information about providers' prices and policies to perform fiscally efficient provisioning.

Legislation/policy awareness. For some applications, the application broker should take into account legislative and political considerations upon provisioning and scheduling. For example, such a broker could avoid placing part of the data outside a given country or can avoid using cloud services of a specific vendor.

This is a typical requirement in both interactive and batch processing data centric applications. In rare cases, this can be a requirement for singular job applications as well, for example, when performing research on a dataset of medical records.

Local resources awareness. Another consideration that is common to all application types is the usage of local in-house resources with higher priority than that of external ones. That is, cloud bursting should be allowed only if the local resources are insufficient or unsuitable in terms of geographical location, proximity to data or operational costs.

6. STATE OF THE ART IN INTER-CLOUDS

In this section, we summarize several state-of-the-art Inter-Cloud developments. They were identified after an extensive analysis of the related literature and include both academic an industry projects. Each of the following subsections discusses the projects that fall into one of the architectural groups introduced in Section 3. Besides a technical summary for each project, we also discuss where it fits within the introduced taxonomies and how it addresses the requirements from Section 5.

6.1. Centralised federated Inter-Clouds

InterCloud. The *InterCloud* [5] project developed at the University of Melbourne is one of the first initiatives in the field. It extends previous efforts form the *InterGrid* project to allow the sharing of

DOI: 10.1002/spe

resources across cloud providers [37]. The proposed architecture is centralised and is built around a central entity called *Cloud Exchange* (CEx). In essence, CEx acts like a marketplace where clouds can sell resources. The buyers may be other clouds or application brokers. Thus, the architecture is clearly market oriented and facilitates pricing aware application brokering.

Each cloud data centre participating in an *InterCloud* Federation needs to have an agent called *Cloud Coordinator* (CC) installed. CCs manage the federation memberships of the clouds by communicating with CEx on their behalf [5]. An architecture of an extensible CC has been further discussed by Calheiros *et al.* [38].

To serve as a federation-wise marketplace, the CEx maintains an information registry of the clouds' resources. The CCs of the participating providers periodically update their details within this information registry [5]. Geographical location of resources can be maintained within CEx, and thus, location aware brokering is possible. Brokering in *InterCloud* can be either *SLA based*, when a cloud provider brokers on behalf of the client, or *directly managed*, when clients directly buy and use resources from CEx.

Contrail. The architecture of the European project *Contrail* [39] is built around a centralised composite entity that acts as a single entry point to a federation of cloud providers. It is responsible for periodically observing the states of the cloud providers and facilitating a federation-wise SLA. It also provides single sign on, so that users need to authenticate only once to work with the entire federation. A special architectural component called *Federation Runtime Manager* (FRM) is dedicated to mapping users' requests to cloud resources. It implements cost and performance optimization heuristics and ideally should have access to the geographical location and other meta information about every cloud provider [39]. Application brokering is achieved by specifying a detailed application SLA. It is the responsibility of the Federation Runtime Manager module to provision in accordance with the SLA and to minimize the costs. However, the documentation is not specific whether users are allowed to specify geographical, legislative and data location constraints in the SLA.

The *Adapters Layer* architectural component constitutes of adapters for every cloud provider in the federation. The adapters facilitate the communication between the federation management components and the clouds. The adapters can be classified as follows:

- Internal adapters – for clouds running the Contrail software. These are called Contrail clouds. Contrail clouds allow for the federation management components to reserve and configure their resources [39].
- External adapters – for clouds that do not run the Contrail software.

When using *external adapters*, the cloud providers themselves are agnostic of the federation – they do not voluntarily participate in it. Thus, in terms of the architectural taxonomy presented here, a Contrail federation using *external adapters* is considered a Multi-Cloud and will be discussed in a later section. On the other hand, a Contrail federation using *internal adapters* is considered a centralised federation. In other words, *Contrail* supports both types of architectures. It even supports a hybrid architecture that utilizes both clouds participating voluntarily within a federation and federation agnostic clouds.

Dynamic Cloud Collaboration (DCC). In the centralised federation architecture *Dynamic Cloud Collaboration* (DCC), the central entity that facilitates the federation is one of the clouds called *primary cloud provider* (pCP). This is the first cloud provider in the federation – the one that actually established it [40, 41]. The other cloud providers are called *collaborating clouds*. The primary cloud (pCP) maintains a registry of the services of the collaborating clouds. Application brokering is done via SLA contract with pCP. Cloud clients submit requests to the pCP, which on the basis of the requests' characteristics allocates resources within the federation. It has not been specified whether clients are allowed to declare requirements about the geographical location of the allocated resources. However, there has been work on the facilitation of a cloud market place where auctions for resources are held [41].

DOI: 10.1002/spe

Federated Cloud Management. The *Federated Cloud Management* (FCM) architecture relies on a generic repository called *FCM Repository* to store virtual appliances for all federated services [42]. It is replicated to the native repositories of the different IaaS providers. Clients interact only with the *Generic Meta-Broker Service* (GMBS) describing the requested service, and as far as they are concerned, further provisioning and scheduling are transparent [42].

For every IaaS provider in the architecture, there is a correspondent broker called *CloudBroker* managing the allocation and deallocation of VMs and dispatching incoming application calls to appropriate VMs. GMBS has access to the *FCM Repository* and can communicate with the *CloudBroker* components of the federated clouds. When GMBS receives a request from a user, it performs matchmaking between the request and an appropriate *CloudBroker*. The matchmaking is based on information from the *FCM Repository* and runtime metrics provided by the *CloudBrokers*.

Each *CloudBroker* maintains an internal queue of incoming application calls and a separate priority queue for every virtual appliance. VM queues represent the resources that can serve a virtual appliance-related service call. The priority of each VM queue is based on the currently available requests in the queue, their historical execution times, and the number of running VMs. On the basis of the VM queues, the *CloudBroker* needs to perform appropriate VM creation and destruction. A *CloudBroker* also handles the aforementioned queue of incoming calls by redirecting them to the VMs created as a result of the management of the VM queues.

Application brokering in FCM is done transparently to the client. In the definition of the architecture, nothing has been said about user-level control over the location of the used resources and how cost optimization can be achieved [42]. We could speculate that location-specific requirements can become a part of the SLA between the clients and the GMBS. In a separate work, Kecskemeti *et al.* extend the FCM approach to facilitate self-adaptable and autonomous management of the federation [43]. One of their goals is to achieve resource optimization (and consequently costs optimization) without violating the SLAs. However, this optimization concerns the resource utilization of the whole federation and does not necessarily lead to cost optimizations for all federation's clients.

6.2. Peer-to-peer federated Inter-Clouds

RESERVOIR. The *Resources and Services Virtualization without Barriers* project (RESERVOIR) is a European project that extends previous research on interconnecting Grids. Its architecture does not feature a central entity and is peer to peer – clouds communicate directly with each other to negotiate resource leasing. The usage of multiple clouds is facilitated through *Claudia* – an abstract layer for execution of services on top of a cloud federation [44].

Custom application brokering in RESERVOIR can be achieved through elasticity rules of the type *Trigger-Action* [45]. Developers can also specify how their applications scale in a declarative way. This is carried out by specifying the SLA of the application in terms of performance objectives (e.g. response time) and a control strategy characteristic (e.g. number of VMs). RESERVOIR then creates an approximate elasticity behavioural model that optimizes the control strategy characteristic without violating the SLA constraints [45].

RESERVOIR requires that all deployed applications should be agnostic of the hosting data centre location, thus allowing for transparent deployment anywhere within the federation. The clients and their application brokers have no control over the selected hosting cloud [45]. RESERVOIR can be considered to be pricing aware, because the price of the used resources could be optimized if set as a control strategy characteristic, as explained earlier.

Open Cirrus. Open Cirrus is a test bed for distributed systems research and federates several research data centres across the globe. Open Cirrus is not an Inter-Cloud environment per se, because it allows users to work directly with the hosts besides utilizing virtualized resources [46,47]. Publications about the project do not give much details about the middleware involved in it. It is known that the test bed runs the experimental cluster management software Tashi [48], the Hadoop framework [49] and a custom physical resources management system called Zoni [46].

The Open Cirrus environment has been designed to be data location aware. Tashi and the location metadata provided by the Hadoop Distributed File System are used to schedule computing tasks

DOI: 10.1002/spe

on the nodes where the correspondent persistent data resides [46]. However, in the publications on the topic, nothing has been said about location awareness when using datasets other than Hadoop Distributed File System files (e.g. relational database). Also, it has not been stated if users can control the location of the used resources and if they have access to pricing information if pricing is at all present in this research test bed.

OPTIMIS. The OPTIMIS toolkit requires that OPTIMIS software agents are deployed within cloud providers and application brokers. These agents communicate with each other to implement Inter-Cloud provisioning.

One of the agent components is the *Deployment Engine* (DE). It is responsible for the discovery and negotiation with suitable clouds for hosting a particular service. First, a set of suitable clouds meeting some predefined conditions are identified, and the *service manifest* (an SLA template) is sent to each of them. The cloud providers answer either with a rejection to accept the service or a deployment offer. DE selects the best offer on the basis of the quantitative (e.g. cost) and qualitative (e.g. some measurement of trust) aspects of the offer. DE is also responsible for initiating the deployment of the service within the selected cloud by providing the appropriate VM images that constitute the needed service. After the deployment of the service, the *Service Optimizer* module is responsible for continuously monitoring the service parameters for SLA violations [19].

On the side of the contacted by the DE cloud, the *Admission Controller* is responsible for either making an offer or rejecting the request. The decision is based on the current workload of the cloud, the requested resources and an evaluation of the potential profit. Allocation of resources for the provided VMs is done by the *Cloud Optimizer* [19].

Application brokering can be implemented by configuring the DE components of the application broker or the hosting cloud provider. During the selection process, DE considers the prices of the deployment offers together with other metrics including trust, risk and energy impact numerical assessments. The exact algorithm for this selection and how configurable it is have not been disclosed. Also, it has not been stated whether DE considers the geographical location of the clouds and the persistent data the provisioned service uses. It has only been mentioned that juridical and political restrictions can be specified and then honoured by the respective DE [19].

The OPTIMIS toolkit supports peer-to-peer federation, when cloud providers communicate directly with each other transparently to their clients. It also supports independent Inter-Clouds (Multi-Clouds), which will be discussed in a successive section.

Arjuna Agility. Arjuna Agility [50] is a commercial framework for establishing Inter-Cloud Federations. It is mostly tailored for federations of in-house data centres but can also facilitate cloud bursting (usage of public clouds) if the demand increases beyond in-house resource capabilities. Each federation site needs to install an Agility software agent that governs the interaction with the other sites [51]. Each site has its own policy regarding resource sharing and can define what resources in terms of hardware and software are shared. Being targeted mostly at in-house cloud federations, Arjuna Agility addresses provider specific problems like reducing the power consumption [52, 53]. To this point, it does not feature cost optimization. Priority usage of local resources is addressed as a data centre borrows resources only if it can not meet its own demands. Provisioning decisions are governed by provisioning policies set by an administrator. They are federation specific, not application specific [51].

Global Inter-Cloud by Bernstein et al. Bernstein *et al.* envision a worldwide federation of cloud providers, rather than separate small-scale federations [54]. They draw analogies from previous experience in integrating separate systems into large-scale utility services, for example, electricity and phone systems and the Internet. They propose a new Inter-Cloud architecture following some of the main design principles of the public Internet.

In a global Inter-Cloud environment, it is important for cloud providers to discover each other and to match their needs for resources. Bernstein *et al.* propose the usage of a global resource catalogue called *Cloud Computing Resource Catalogue*. It contains information about the shared resources

DOI: 10.1002/spe

within the federation and should be hosted by several community governed *Intercloud Root* servers. Root servers replicate the catalogue among each other to provide better performance and availability.

Negotiation between providers is facilitated by *Intercloud Exchanges* – distributed servers that allow cloud providers to discover suitable resources. *Intercloud Exchanges* perform match making between cloud providers on the basis of specified preferences and constraints. *Intercloud Exchanges* use the resource catalogue stored on the *Intercloud Roots* to provide a Distributed Hash Table suitable for fast querying. After the negotiation is done, cloud providers continue to communicate directly with each other.

The proposed architecture is not centralized because the *Intercloud Roots* and *Exchanges* are distributed and replicated. Thus, a failure of any of them cannot cause a failure of the federation as a whole.

To facilitate all phases of the described global Inter-Cloud system, each participant should have an entity called *Intercloud Gateway*. These entities allow cloud providers, *Intercloud Roots* and *Exchanges* to communicate with each other using a set of Inter-Cloud protocols and standards. To date, such interoperability protocols and standards have not been widely utilized, although there are already ongoing works in the area [6, 55–57].

From the clients' viewpoint, application brokering is achieved through specifying appropriate constraints and preferences in the SLA with the cloud provider. The cloud provider should then respect the SLA when submitting requests to an *Intercloud Exchange*. Both private and public clouds can participate within the envisioned global federation. Thus, private clouds can consider using local resources before utilizing external clouds. Bernstein *et al.* envision that cloud providers should be able to specify resource location preferences and constraints on behalf of their clients when negotiating to meet legislative and political requirements [54]. Cloud providers could also implement cost optimization policies on behalf their clients, although this has not been discussed in the related literature.

6.3. Multi-Cloud services

OPTIMIS. Besides federation, OPTIMIS also supports Independent Inter-Clouds (Multi-Clouds) [19]. In terms of the OPTIMIS architecture, a Multi-Cloud is achieved when clients use their DE and *Service Optimizer* directly to launch and monitor services within multiple cloud providers.

A major drawback here is the need to work with cloud providers that have OPTIMIS agents deployed in their data centres. As discussed often, cloud providers would not wish to voluntarily participate in an Inter-Cloud and install such agents. The OPTIMIS vision is to allow for externally developed adapters for such clouds that should facilitate the communication between OPTIMIS agents and OPTIMIS agnostic cloud providers. Such adapters should be developed for every used cloud service provider.

Contrail. Like OPTIMIS, Contrail [39] also supports both federation and Independent Inter-Clouds. Similar to OPTIMIS, a major drawback of Contrail in terms of supporting independent Inter-Clouds is the need to develop and maintain multiple vendor specific Contrail adapters.

mOSAIC. The *mOSAIC* open source API and platform allow for the development and deployment of applications that use multiple clouds [21]. Unlike most other Inter-Cloud technologies, mOSAIC has some assumptions about the architecture of the deployed application. It is assumed that the application is divided into components with explicit dependencies in terms of both communication and data between them. Also, the application architecture must be service oriented (SOA) and must use only the mOSAIC API for intercomponent communication. Other types of communication (e.g. direct sockets) are disallowed. For each component application, developers must specify the resource requirements in terms of computing, storage and communication. Some resource requirements are specified for the whole application as well, for example, overall budget [21].

The mOSAIC platform automatically provisions such applications across a set of predefined clouds, without direct involvement from the user. Thus, the only way to control the application

DOI: 10.1002/spe

brokering is through the predefined SLA on performance indicators at component and application level.

Application brokering in mOSAIC is done by a component called *Resource Broker*. It is responsible for the mediation between clients and cloud providers. Resource Broker is further composed of a *Cloud Agency*, which is responsible for discovery and negotiation with cloud providers and a *Client Interface* responsible for requesting additional resources from the *Application Executor* component. The *Application Executor* component manages the deployment, execution and monitoring of the application within the reserved resources [21]. In this architecture, pricing awareness is promoted through the *Cloud Agency* component, which takes into consideration and negotiates on the pricing policies of the cloud providers.

STRATOS. STRATOS is a cloud broker service at an early stage of development [58]. The entry point of the system is the *CloudManager*. Clients describe the application topology and requirements in a *Topology Descriptor File* (TDF) and submit it to the *CloudManager*. It contacts the *Broker* component, which determines the optimal initial resource allocation across clouds. The *Broker* uses another architectural component called *Translation Layer* to access miscellaneous clouds through a uniform API. The *CloudManager* and the *Broker* continuously receive monitoring information based on which they take further provisioning decisions.

Application-specific brokering is achieved by specifying requirements and policies within the TDF. STRATOS is obliged to honour the terms from the TDF. Thus, the TDF can be considered as an SLA document between the client and the STRATOS service. The TDF format is XML based and allows for the specification of deployment topology configuration (in terms of VM images, middleware, storage, etc.), constraints, and objectives. The goal of the *Broker* is to optimize the objectives without violating the constraints or breaking the application topology. To build optimizational models, the *Broker* needs adequate measurements of the performance characteristics of cloud providers. For this purpose, STRATOS utilizes the Service Measurement Index [59] and the metrics defined by Garg [60] and Zachos *et al.* [61].

The cost is a usual candidate for an objective, and thus, STRATOS can be pricing aware [58]. We could speculate that geo-location and legislation/policy awareness can be promoted via TDF constraints, although this has not been discussed in the related literature. Also, the mapping of incoming requests to nodes near the appropriate data persistence storage has not been discussed.

Commercial Cloud Management Systems. Several companies provide paid independent Multi-Cloud services. In essence, their features are similar, and they compete against each other on the basis of the performance overhead they add and the variety of cloud providers they can integrate with. It is beyond the scope of this work to survey all players in this dynamic market niche, and here, we only outline the features provided by some of the major companies.

RightScale offers a service for deploying and managing applications across clouds. Users can manage VMs on multiple clouds through the RigthScale console [62]. Application brokering is achieved through the *alert-action* mechanism, similar to the *Trigger-Action* mechanism. All RightScale VMs have predefined 'hooks', which continuously send information back to the RigthScale console. This facilitates the check of the alert conditions and the execution of the predefined actions. Actions can define scale up/down policies but can also be administrative in nature (e.g. sending email to the system administrator). As an action, a user is allowed to specify in which cloud and location resources should be provisioned. Thus, scaling within RightScale can be location aware. Users can add private clouds within the RigthScale console to facilitate local resources utilization.

Generally speaking, the services of EnStratus [63], Scalr [64] and Kaavo [65] are comparable with those of RightScale. Naturally, there are difference in the pricing, the set of supported cloud vendors and technologies and some terminology. Similar to RightScale, they allow users to deploy VMs across different public and private clouds. Application brokering is achieved through automated triggers, similar to RigthScale's alerts, which can trigger scale up/down actions.

DOI: 10.1002/spe

6.4. Inter-Cloud libraries

Several Inter-Cloud libraries have been developed in recent years. Examples are the Java library JClouds [66], the Python library Apache LibCloud [67], the Ruby library Apache DeltaCloud [68] and the PHP library SimpleCloud [69]. Another library project at a very early stage of development is Apache Nuvem [70].

All these libraries are designed to abstract the programmers from the differences in the management APIs of clouds and provide control over the provisioning of resources across geographical locations. Such libraries can be used to ease the development of cloud adapters for technologies like OPTIMIS of mOSAIC. By using such libraries, application developers can program their own application-specific brokers. Such brokers directly manage the underlying cloud infrastructures and thus can meet all of the previously outlined requirements. A major issues with this approach is that developers need to define appropriate deployment and replication of the broker for availability reasons.

7. DISCUSSION

In the current work, we have discussed and classified 20 major Inter-Cloud developments, identified after detailed analysis of the related work. These include both academic and industry projects, which are summarized in Table II. Figure 7 outlines how many of them fall into each of the taxonomic

Table II. Summary of Inter-Cloud projects.

Project	Type, Organisation	Architecture	Brokering approach	Application type	Awareness
InterCloud	Research project, University of Melbourne	Centralised federation	SLA based and directly managed	Singular jobs	Geo-location, pricing
Contrail	Private and public European research organisations funded by EU	Centralised federation and independent service	SLA based	Singular jobs	Pricing
Dynamic Cloud Collaboration (DCC)	Academic research project supported by South Korean research funds	Centralised federation	SLA based	Singular jobs	Pricing
Federated Cloud Management	Academic research project supported by EU funds	Centralised federation	SLA based	Singular jobs	Pricing
RESERVOIR	Private and public European research organisations funded by EU	Peer-to-peer federation	SLA based and Trigger-Action	Singular jobs	Pricing
Open Cirrus	Research testbed by academic and industry partners. Partially funded by US NSF	Peer-to-peer federation	Directly managed	Singular jobs	Data location
OPTIMIS	Private and public European research organisations funded by EU	Peer-to-peer federation and Independent service	SLA based	Singular jobs, periodical jobs, compute-intensive and data-intensive interactive application	Pricing

DOI: 10.1002/spe

Table II. *Continued*.

Project	Type, Organisation	Architecture	Brokering approach	Application type	Awareness
Arjuna Agility	Commercially owned	Peer-to-peer federation	Trigger-Action	Singular jobs, periodical jobs, compute-intensive and data-intensive interactive application	Local resources
Global Inter-Cloud by Bernstein *et al.*	Publications are by people from miscellaneous companies – CISCO, Huawei Technologies, EMC Corporation	Peer-to-peer federation	SLA based	Singular jobs, periodical jobs, compute-intensive and data-intensive interactive application	Data location, Local resources
mOSAIC	Private and public European research organisations funded by EU	Independent service	SLA based	Singular jobs, periodical jobs, compute-intensive and data-intensive interactive application	Pricing
STRATOS	York University. Supported by Canada's NSERC funds, Amazon and CA Inc.	Independent service	SLA based	Singular jobs, periodical jobs, compute-intensive and data-intensive interactive application	Geo-location, pricing, legislation/policy and local resources
Commercial Cloud Management Systems (RightScale, EnStratus, Scalr, Kaavo)	Commercially owned	Independent service	Trigger-Action	Singular jobs	Geo-location, data location, pricing, legislation/policy and local resources
Libraries (JClouds, LibCloud, DeltaCloud, SimpleCloud, Apache Nuvem)	Open source projects	Multi-Cloud libraries	Directly managed	Singular jobs, periodical jobs, compute-intensive and data-intensive interactive application	Geo-location, data location, pricing, legislation/policy and local resources

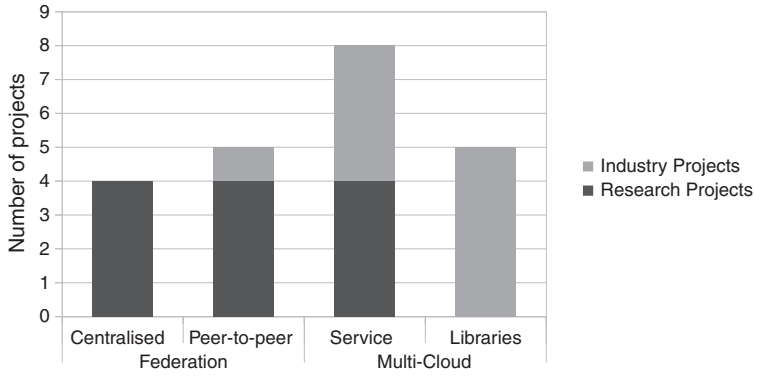

Figure 7. Inter-Cloud projects by architecture.

DOI: 10.1002/spe

groups discussed earlier. The sum of the counts is greater than 20, because some projects support multiple Inter-Cloud topologies.

Most *Multi-Cloud* projects are industry driven. That is, they are not research projects supported by academic grants or developed in an academic environment. Exceptions are OPTIMIS, Contrail and mOSAIC co-funded by the European Union and STRATOS, which is partially supported by Canada's Natural Sciences and Engineering Research Council (NSERC). Of these, only mOSAIC and STRATOS support exclusively a Multi-Cloud, and the other two support Inter-Cloud Federations as well. On the contrary, almost all Inter-Cloud Federation projects are visionary research projects. The only exceptions is Arjuna Agility, used to build in-house Inter-Cloud Federations.

This speaks for the fact that currently there is a market demand for Multi-Clouds and the technologies to facilitate them are available. As discussed, the predominant use of *Multi-Clouds* is to provide cloud agnostic access to multiple competing *independently owned* clouds without their collaboration and knowledge. Such mediating services and libraries are especially useful for small-sized and medium-sized businesses that want to deploy applications in multiple public cloud environments for better availability, flexibility and responsiveness. Thus, there is already competition between the providers in this market niche.

On the contrary, almost all federation initiatives are research projects at an early stage of development and adoption. This is because of the many open issues that need to be addressed with respect to provisioning resources and scheduling application components on behalf of the clients.

The majority of the discussed developments take an *SLA-based* approach to Inter-Cloud application brokering. These projects try to make the Inter-Cloud transparent to its clients, but as discussed, the SLA-based approach currently meets series of shortcomings. Thus, all of them are research projects at an early stage of development – see Figure 8. The direct approach is the most flexible but is also the hardest to work with from the clients' perspective, because clients would need to program their own application brokers. The *Trigger-Action* approach is used by both federations and independent Inter-Cloud services. It combines the declarative nature of the SLA approach and the provisioning flexibility of the direct approach. All discussed industry projects take either direct or Trigger-Action approach to application brokering.

Pricing awareness is the only brokering characteristic that can be facilitated by almost all Inter-Cloud projects – see Table II. Most federations do not support brokering considering any other of the predefined requirements. Few of them have limited awareness of data location, legislation/policy and local resources. On the contrary, all directly managed Inter-Clouds cover the whole range of discussed requirements and thus provide means for adequate brokering of miscellaneous applications.

As a consequence, most current federation projects are not tailored for *Interactive* applications, because they do not allow application brokering to consider such aspects. On the contrary, *Independent* Inter-Cloud projects can facilitate all of these and thus are more general purpose in nature.

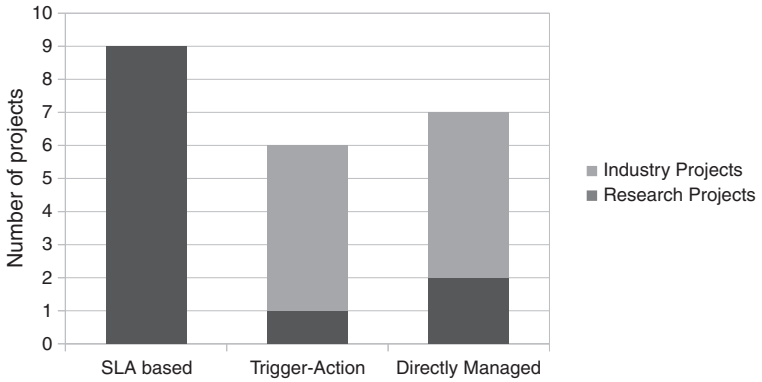

Figure 8. Inter-Cloud projects by brokering mechanisms.

DOI: 10.1002/spe

8. CONCLUSION AND FUTURE DIRECTIONS

In this work, we have classified and analysed the state of the art in Inter-Cloud developments. More specifically, we focus on how current Inter-Cloud projects facilitate the brokering of different applications across multiple clouds. This is a novel area of intensive research and development whose body of knowledge has not been well established yet.

We introduce classifications of Inter-Cloud architectures and brokering mechanisms. Then we classify Inter-Cloud applications and discuss their coarse-grained properties and brokering requirements. After an extensive analysis of the related literature, we identify, discuss and fit into the aforementioned classifications 20 major projects. After that, we analyse the state of the art and the trends in the area.

We have identified that there is a major difference in the directions that early research projects and industry projects are heading. Whereas most research projects focus on developing Inter-Cloud Federations, most industry projects provide services or libraries for direct provisioning and scheduling across clouds. Also, most research projects focus on SLA-based brokering, whereas industry driven ones focus on Trigger-Action based or directly managed brokering.

Both these two trends imply future work, which can be summarized as follows:

- *Advances in SLA specification techniques* – as discussed, current SLA approaches are not suitable for specifying appropriate provisioning across clouds. Further work is needed to develop them.
- *Combination of brokering approaches* – it is unlikely that any of the discussed approaches will be sufficiently flexible to facilitate all kinds of applications. Thus, it is worthy to investigate how they can be combined. For example, some coarse-grained provisioning rules may be specified in an SLA, whereas fine-grained provisioning and scheduling policies can be implemented by Trigger-Action rules.
- *Application-specific brokering* – further research is needed to define the best ways to broker specific types of applications across clouds. In this work, the introduced taxonomy of distributed applications (see Section 5) was used as a starting point to identify the coarse-grained requirements for the major types of applications. Further work is needed to identify more specific requirements and brokering policies for specific application types, for example, Online Transaction Processing systems.
- *Improved Inter-Cloud environments* – as discussed, many of the existing Inter-Cloud environments do not allow for adequate brokering of many types of applications. The improvement or development of new Inter-Cloud environments that allow for this is essential.

ACKNOWLEDGEMENTS

We thank Rodrigo Calheiros, Anton Beloglazov, Amir Vahid Dastjerdi and Adel Nadjaran Toosi for the insightful comments on improving this work.

REFERENCES

1. Buyya R, Yeo CS, Venugopal S, Broberg J, Brandic I. Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility. *Future Generation Computer Systems* 2009; **25**(6):599–616.
2. Foster I, Zhao Y, Raicu I, Lu S. Cloud computing and grid computing 360-degree compared. In *Proceedings of Grid Computing Environments Workshop (GCE 2008)*. IEEE: Austin, Texas, US, 2008; 1–10.
3. Mell P, Grance T. *The NIST Definition of Cloud Computing.* Special Publication 800-145. National Institute of Standards and Technology (NIST), U.S. Department of Commerce: Gaithersburg, Maryland, US, 2011.
4. Lewis G. Basics about cloud computing. *Technical Report September*, Software Engineering Institute – Carnegie Mellon, 2010.
5. Buyya R, Ranjan R, Calheiros RN. InterCloud: utility-oriented federation of cloud computing environments for scaling of application services. In *Proceedings of the 10th International Conference on Algorithms and Architectures for Parallel Processing*. Springer-Verlag: Busan, Korea, 2010; 13–31.
6. Bernstein D, Ludvigson E, Sankar K, Diamond S, Morrow M. Blueprint for the intercloud – protocols and formats for cloud computing interoperability. In *Proceedings of the 4th International Conference on Internet and Web Applications and Services (ICIW 2009)*. IEEE: Venice/Mestre, Italy, 2009; 328–336.

DOI: 10.1002/spe

7. Keahey K, Tsugawa M, Matsunaga A, Fortes J. Sky computing. *Internet Computing* 2009; **13**(5):43–51.
8. Rochwerger B, Breitgand D, Levy E, Galis A, Nagin K, Llorente IM, Montero RS, Wolfsthal Y, Elmroth E, Cáceres JA, Ben-Yehuda M, Emmerich W, Galán F. The RESERVOIR model and architecture for open federated cloud computing. *IBM Journal of Research and Development* 2009; **53**(4):1–11.
9. Celesti A, Tusa F, Villari M, Puliafito A. How to enhance cloud architectures to enable cross-federation. In *Proceedings of the 3rd International Conference on Cloud Computing (CLOUD 2010)*. IEEE: Miami, Florida, US, 2010; 337–345.
10. Kelly K. The Technium: A Cloudbook for the Cloud. Available from: http://www.kk.org/thetechnium/archives/2007/11/a_cloudbook_for.php [last accessed 1 June 2012].
11. Amazon. Summary of the Amazon EC2 and Amazon RDS Service Disruption. Available from: http://aws.amazon.com/message/65648/ [last accessed 1 June 2012].
12. Amazon. Summary of the AWS Service Event in the US East Region. Available from: http://aws.amazon.com/message/67457/ [last accessed 6 August 2012].
13. Google. Post-mortem for February 24th, 2010 outage. Available from: https://groups.google.com/group/google-appengine/browse_thread/thread/a7640a2743922dcf?pli=1 [last accessed 14 June 2012].
14. Microsoft. Windows Azure Service Disruption Update. Available from: http://blogs.msdn.com/b/windowsazure/archive/2012/03/01/windows-azure-service-disruption-update.aspx [last accessed 14 June 2012].
15. Armbrust M, Fox A, Griffith R, Joseph A, Katz R. Above the Clouds: A Berkeley View of Cloud Computing. *Technical Report*, Electrical Engineering and Computer Sciences, University of California, Berkeley, 2009.
16. Beloglazov A, Buyya R, Lee YC, Zomaya A. A taxonomy and survey of energy-efficient data centers and cloud computing systems. *Advances in Computers, M. Zelkowitz (ed.)* 2011; **82**:47–111.
17. Global Inter-Cloud Technology Forum. Use Cases and Functional Requirements for Inter-Cloud Computing. *Technical Report*, Global Inter-Cloud Technology Forum, 2010.
18. Sakashita Y, Takayama K, Matsuo A, Kurihara H. Cloud Fusion Concept. *FUJITSU Scientific & Technical Journal (FSTJ)* 2012; **48**(2):143–150.
19. Ferrer AJ, Hernández F, Tordsson J, Elmroth E, Ali-Eldin A, Zsigri C, Sirvent R, Guitart J, Badia RM, Djemame K, Ziegler W, Dimitrakos T, Nair SK, Kousiouris G, Konstanteli K, Varvarigou T, Hudzia B, Kipp A, Wesner S, Corrales M, Forgó N, Sharif T, Sheridan C. OPTIMIS: a holistic approach to cloud service provisioning. *Future Generation Computer Systems* 2012; **28**(1):66–77.
20. Arjuna Agility. What Is Federation. Available from: http://www.arjuna.com/what-is-federation [last accessed 14 June 2012].
21. Petcu D, Crciun C, Neagul M, Panica S, Di Martino B, Venticinque S, Rak M, Aversa R. Architecturing a sky computing platform. In *Proceedings of the International Conference Towards a Service-Based Internet ServiceWave'10*, Vol. 6569, Cezon M, Wolfsthal Y (eds). Springer-Verlag: Ghent, Belgium, 2011; 1–13.
22. Lucas Simarro J, Moreno-Vozmediano R, Montero R, Llorente I. Dynamic placement of virtual machines for cost optimization in multi-cloud environments. In *Proceedings of the International Conference on High Performance Computing and Simulation (HPCS 2011)*. IEEE: Istanbul, Turkey, 2011; 1–7.
23. NeCTAR. Home NeCTAR. Available from: http://nectar.org.au/ [last accessed 14 June 2012].
24. CESWP. Cloud-Enabled Space Weather Modelling and Data Assimilation Platform. Available from: http://www.cybera.ca/projects/ceswp [last accessed 14 June 2012].
25. CSCC Workgroup. Practical guide to cloud service level agreements version 1.0. *Technical Report*, Cloud Standards Customer Council (CSCC), 2012.
26. Jackson K, Ramakrishnan L, Muriki K, Canon S, Cholia S, Shalf J, Wasserman H, Wright N. Performance analysis of high performance computing applications on the amazon web services cloud. In *Proceedings of the IEEE Second International Conference on Cloud Computing Technology and Science (CloudCom 2010)*. IEEE: Indianapolis, Indiana, US, 2010; 159–168.
27. Amazon. High Performance Computing (HPC) on AWS. Available from: http://aws.amazon.com/hpc-applications/ [last accessed 20 August 2012].
28. Raicu I, Foster I, Zhao Y. Many-task computing for grids and supercomputers. *Proceedings of the Workshop on ManyTask Computing on Grids and Supercomputers (MTAGS 2008)*, Austin, Texas, US, 2008; 1–11.
29. Barker A, Van Hemert J. Scientific workflow: A survey and research directions. In *Proceedings of the 7th International Conference on Parallel Processing and Applied Mathematics*. Springer-Verlag: Gdansk, Poland, 2008; 746–753.
30. COUCHBASE. NoSQL Database Technology. *Technical Report*, COUCHBASE, 2012.
31. Cattell R. Scalable SQL and NoSQL data stores. *SIGMOD Record* 2010; **39**(4):12–27.
32. Brewer E. Towards robust distributed systems. In *Proceedings of the Annual ACM Symposium on Principles of Distributed Computing*, Vol. 19. ACM: Portland, Oregon, United States, 2000; 7–10.
33. Brewer E. CAP twelve years later: How the "Rules" have changed. *Computer* 2012; **45**(2):23.
34. Fowler M. Polyglot Persistence. Available from: http://martinfowler.com/bliki/PolyglotPersistence.html [last accessed 14 June 2012].
35. Leberknight S. Polyglot Persistence. Available from: http://www.nearinfinity.com/blogs/scott_leberknight/polyglot_persistence.html [last accessed 1 June 2012].
36. Dong F, Akl S. Scheduling Algorithms for Grid Computing: State of the Art and Open Problems. *Technical Report*, School of Computing, Queens University, Kingston, Ontario, 2006.

DOI: 10.1002/spe

37. Di Costanzo A, De Assuncao M, Buyya R. Harnessing cloud technologies for a virtualized distributed computing infrastructure. *Internet Computing, IEEE* 2009; **13**(5):24–33.
38. Calheiros R, Toosi A, Vecchiola C, Buyya R. A coordinator for scaling elastic applications across multiple clouds. *Future Generation Computer Systems* 2012; **28**(8):1350–1362.
39. Carlini E, Coppola M, Dazzi P, Ricci L, Righetti G. Cloud federations in contrail. In *Proceedings of Euro-Par 2011: Parallel Processing Workshops*, Vol. 7155, Alexander Mea (ed.). Springer Berlin / Heidelberg: Bordeaux, France, 2012; 159–168.
40. Hassan M, Song B, Yoon C, Lee HW, Huh EN. A novel market oriented dynamic collaborative cloud service infrastructure. In *Proceedings of the World Conference on Services*. IEEE: Los Angeles, California, US, 2009; 9–16.
41. Hassan M, Song B, Huh EN. A market-oriented dynamic collaborative cloud services platform. *Annals of Telecommunications* 2010; **65**(11):669–688.
42. Marosi A, Kecskemeti G, Kertesz A, Kacsuk P. FCM: An architecture for integrating IaaS cloud systems. *Proceedings of the Second International Conference on Cloud Computing, GRIDs, and Virtualization*, Rome, Italy, 2011; 7–12.
43. Kecskemeti G, Maurer M, Brandic I, Kertesz A, Nemeth Z, Dustdar S. Facilitating self-adaptable Inter-Cloud management. In *Proceedings of the 20th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP 2012)*. IEEE: Munich, Germany, 2012; 575–582.
44. Rodero-Merino L, Vaquero LM, Gil V, Galn F, Fontn J, Montero RS, Llorente IM. From infrastructure delivery to service management in clouds. *Future Generation Computer Systems* 2010; **26**(8):1226–1240.
45. Rochwerger B, Breitgand D, Epstein A, Hadas D, Loy I, Nagin K, Tordsson J, Ragusa C, Villari M, Clayman S, Levy E, Maraschini A, Massonet P, Muñoz H, Toffetti G. Reservoir - When one cloud is not enough. *Computer* 2011; **44**(3):44–51.
46. Avetisyan A, Campbell RH, Gupta I, Heath MT, Ko SY, Ganger GR, Kozuch MA, O'Hallaron DR, Kunze M, Kwan TT, Lai K, Lyons M, Milojicic DS, Lee HY, Soh YC, Ming NK, Luke JY, Namgoong H. Open cirrus: A global cloud computing testbed. *Computer* 2010; **43**(4):35–43.
47. Campbell R, Gupta I, Heath M, Ko SY, Kozuch M, Kunze M, Kwan T, Lai K, Lee HY, Lyons M, *et al.* Open cirrus cloud computing testbed: Federated data centers for open source systems and services research. In *Proceedings of the Conference on Hot Topics in Cloud Computing*. USENIX Association: San Diego, California, 2009.
48. Kozuch MA, Ryan MP, Gass R, Schlosser SW, O'Hallaron D, Cipar J, Krevat E, López J, Stroucken M, Ganger GR. Tashi: Location-aware cluster management. In *Proceedings of the 1st Workshop on Automated Control for Datacenters and Clouds*. ACM: Barcelona, Spain, 2009; 43–48.
49. Apache Foundation. Apache Hadoop!. Available from: http://hadoop.apache.org/ [last accessed 14 June 2012].
50. Arjuna Agility. Arjuna. Available from: http://www.arjuna.com/agility [last accessed 14 June 2012].
51. Arjuna. Arjuna Agility: Removing the Barriers to Business Agility. *Technical Report*, University of Zurich, Department of Informatics, 2010.
52. McGough A, Gerrard C, Noble J, Robinson P, Wheater S. Analysis of power-saving techniques over a large multi-use cluster. In *Proceedings of the 9th International Conference on Dependable, Autonomic and Secure Computing (DASC 2011)*. IEEE: Sydney, Australia, 2011; 364–371.
53. McGough AS, Gerrard C, Haldane P, Sharples D, Swan D, Robinson P, Hamlander S, Wheater S. Intelligent power management over large clusters. In *Proceedings of the IEEE/ACM International Conference on Green Computing and Communications & International Conference on Cyber, Physical and Social Computing*. IEEE: Hangzhou, China, 2010; 88–95.
54. Bernstein D, Vij D, Diamond S. An intercloud cloud computing economy-technology, governance, and market blueprints. In *Annual SRII Global Conference (SRII), IEEE*. IEEE: San Jose, California, US, 2011; 293–299.
55. Bernstein D, Vij D. Intercloud directory exchange protocol detail using XMPP and RDF. In *Proceedings of the 6th World Congress on Services (SERVICES-1)*. IEEE, IEEE: Miami, Florida, US, 2010; 431–438.
56. Bernstein D, Vij D. Simple storage replication protocol (SSRP) for intercloud. In *Proceedings of the Second International Conference on Emerging Network Intelligence*. CSREA Press: Florence, Italy, 2010; 30–37.
57. Bernstein D, Vij D. Using semantic web ontology for intercloud directories and exchanges. In *Proceedings of the International Conference on Internet Computing*, 2010; 18–24.
58. Pawluk P, Simmons B, Smit M, Litoiu M, Mankovski S. Introducing STRATOS: a cloud broker service. In *Proceedings of the IEEE International Conference on Cloud Computing (CLOUD 2012)*. IEEE: Honolulu, Hawaii, US, 2012.
59. Cloud Service Measurement Index Consortium (CSMIC). Service Measurement Index (SMI), Jul 27 2012. Available from: http://www.cloudcommons.com/about-smi.
60. Garg S, Versteeg S, Buyya R. SMICloud: a framework for comparing and ranking cloud services. In *Proceedings of the 4th IEEE International Conference on Utility and Cloud Computing (UCC 2011)*. IEEE: Melbourne, Australia, 2011; 210–218.
61. Zachos K, Lockerbie J, Hughes B, Matthews P. Towards a framework For describing cloud service characteristics for use by chief information officers. In *Proceedings of the Workshop on Requirements Engineering for Systems, Services and Systems-of-Systems (RESS 2011)*. Trento: Italy, 2011; 16–23.
62. RightScale. RightScale. Available from: http://www.rightscale.com/ [last accessed 14 June 2012].
63. EnStratus. EnStratus. Available from: https://www.enstratus.com/ [last accessed 14 June 2012].
64. Scalr. Scalr. Available from: http://scalr.net/ [last accessed 14 June 2012].

DOI: 10.1002/spe

65. Kaavo. Kaavo. Available from: http://www.kaavo.com/ [last accessed 14 June 2012].
66. JClouds. JClouds. Available from: http://www.jclouds.org/ [last accessed 14 June 2012].
67. Apache Foundation. Apache Libcloud. Available from: http://libcloud.apache.org/ [last accessed 14 June 2012].
68. Apache Foundation. Apache Delta Cloud. Available from: http://deltacloud.apache.org/ [last accessed 14 June 2012].
69. Simple Cloud. Simple Cloud API. Available from: http://simplecloud.org/ [last accessed 20 August 2012].
70. Apache Foundation. Apache Nuvem. Available from: http://incubator.apache.org/nuvem/ [last accessed 20 August 2012].

DOI: 10.1002/spe