A Systematic Mapping Study of Software Architectures for Cloud Based Systems

Muhammad Aufeef Chauhan
PhD Student
Software and Systems Section
muac@itu.dk

Muhammad Ali Babar
Professor
Software and Systems Section
maba@itu.dk

IT University Technical Report Series TR-[2014-175]

ISSN 1600-6100 [March 2014]

ISSN 1600-6100

ISBN 978-87-7949-313-1

Copies may be obtained by contacting:

IT University of Copenhagen
Rued Langgaards Vej 7
DK – 2300 Copenhagen S
Denmark

Telephone: +45 72 18 50 00
Telefax: +45 72 18 50 01
Web: www.itu.dk

A Systematic Mapping Study of Software Architectures for Cloud Based Systems

Muhammad Aufeef Chauhan, Muhammad Ali Babar
Software and System Section
IT University of Copenhagen
muac@itu.dk, malibaba@itu.dk

Abstract

Context: Cloud computing has gained significant attention of researchers and practitioners. This emerging paradigm is being used to provide solutions in multiple domains without huge upfront investment because of its on demand recourse-provisioning model. However, the information about how software systems are constructed for cloud based systems and what architecture approaches are used to build these systems in not available in synthesized form, which makes it hard to find common architecture solutions for building applications for cloud and identify research gaps.

Object: The main objective of this study is to systematically identify and analyze the currently published research on the topics related to software architectures for cloud-based systems in order to identify architecture solutions for achieving quality requirements.

Method: We decided to carry out a systematic mapping study to find as much peer-reviewed literature on the topics related to software architectures for cloud-based systems as possible. This study has been carried out by following the guidelines for conducting systematic literature reviews and systematic mapping studies as reported in the literature. Based on our paper selection criteria defined by the requirements of the study's objectives and research questions, we have found 86 papers to be included in this study out of initial set of 1491 papers.

Results: We have grouped selected papers into different categories of themes including: quality attributes (15%), multi-tenancy (3%), frameworks (3%), workflow based systems (4%), support for hybrid devices (7%), middleware infrastructure for managing services and resources (10%), architecture refactoring considerations for migrating applications to cloud (4%), generic architecture solution (12%), emerging research areas (3%) and cloud application domains (39%). We have described problems being addressed in each category, which architecture solutions have been proposed to solve these problems and technologies that have been used.

Conclusions:

The selected studies reports challenges and potential solutions related to maintaining underlying infrastructure for supporting large number of users, utilizing cloud computing for making applications available for devices with limited resource, performing sensitive data processing on cloud and achieving service level agreements at different levels or abstractions. Energy optimization, service adaptability, reliability, resource provisioning and service integration are most commonly addressed quality attributes in cloud bases system. Work reported in this study focuses on architectural constructs of cloud-enabled systems reported in literature. We have synthesized architecture approaches proposed in paper into architecture styles

that can be used for building cloud-based applications. After analyzing and synthesizing solutions proposed in independent studies we have provided architecture patterns for management, execution and realization of services. We have also proposed synthesized architecture patterns for managing multi-tenancy at database, service monitoring, service interoperability and management of distributed workflow execution.

Keywords

Cloud computing, Software architectures, Infrastructure as a Service (IaaS), Platform as a Service (PaaS), Software as a Service (SaaS), Service Level Agreement (SLA), Quality of Service (QoS) and systematic mapping study.

Table of Contents

1. Introduction

Cloud computing has become an active area of research and practice over the last few years. It is based on computing utility and service provisioning approach. It offers organizations an opportunity to have on demand scalability and flexibility of computing as well as storage resources [1-3]. This utility model enables organizations

to save upfront investment costs, needed for setting up and running large scale computing infrastructure. It frees organizations from low level infrastructure related tasks and offers them opportunity to concentrate on their core business operations. A report by IDC (International Data Corporation) suggests the growth in cloud service spending to $42 billion by the end of 2012 [4]. This growth trend is supported by big players of IT including Amazon[1], Google, Microsoft[2] and SalesForce[3]; that are providing cloud based infrastructure and services to consumers. Applications of heterogeneous domains ranging from social networking sites and gaming portals to scientific workflow systems and business applications are utilizing the power of cloud computing platform [4].

Different people have different interpretations of the word cloud computing and there are many definitions [5-7]. US national institute of standards and technology (NIST) has more comprehensive definition of cloud computing and defines it as "*A model for enabling convenient, on-demand network access to shared pool of configurable computing resources (e.g. storage, application services, servers and network) that can be rapidly provisioned and released with minimal management effort or service provider interaction*" [7]. Key feature of this paradigm is the ability to deliver services and infrastructure as pay per use basis [4]. Service level agreements (SLAs) are used for specification of QoS requirements between cloud service providers and consumers.

For achieving flexible hardware and software resource provisioning, cloud computing infrastructure should have the ability of on demand resource acquisition, billing schemes to charge users and resource publication through a single provider [8]. Cloud computing solutions offered by public cloud providers are broadly classified into three services and five deployment models [1, 2, 9-11]. Three categories of service models are: Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS). Five deployment models include: public, private, hybrid, community and virtual private clouds.

IaaS cloud provides abstraction to underlying computing, storage and network resources using virtualization technologies. It also provides basic software resources such as operating system to utilize the virtualized hardware resources. It poses additional overhead to applications and technical staff for monitoring and optimizing resources to meet QoS requirements specified in SLAs. This infrastructure has advantage of support for customization. Additional tools and software can be installed as per requirements of the applications and end users. Amazon Elastic Cloud[4], Amazon Simple Storage Services[5], Eucalyptus[6] and OpenNebula[7] are example of IaaS cloud platforms. ***PaaS*** cloud provides application programmable interfaces (APIs) to develop applications. Application build using PaaS APIs do not need to handle resource provisioning of underlying infrastructure. Google App Engine[8], Microsoft Azure platform[9] and SalesForce[3] are examples of the PaaS. Although PaaS provide

---

1 http://aws.amazon.com/

2 http://www.microsoft.com/

3 http://www.salesforce.com/

4 http://aws.amazon.com/ec2/

5 http://aws.amazon.com/s3/

6 http://www.eucalyptus.com/

7 http://opennebula.org/

8 http://code.google.com/appengine/

9 http://www.windowsazure.com/

supports for seamless scalability and easy way to develop applications for cloud, it also has some disadvantages [8, 12]. One of major disadvantage is that applications developed on PaaS are tightly bound to the platform. Porting these applications on other platforms may requires major refactoring and has negative impact on long-term evolution of the system. Enhancements in applications developed using this platform are also tightly coupled with new features supported by platform provider. Moreover, as PaaS does not provide support for customization; it may not be straightforward to deploy application using multiple frameworks on PaaS because of unavailability of required frameworks. ***SaaS*** represents applications that are build on top of either IaaS or PaaS clouds and offer business solutions to end users. One of the key features of these applications is multi-tenancy. It enables single instance of the application to service large number of organizations and end users. SaaS provide limited support for customization.

Public cloud represent cloud infrastructure and software resources maintained by an organization and is offered to end users for lease on basis of some pricing model. End users can access infrastructure by using Internet. Amazon Elastic Compute Cloud (EC2) and Simple Storage Service (S3), Google App Engine and Microsoft Azure are examples of public clouds. ***Private cloud*** represents infrastructure and software resources maintained by organization for its internal use. In some cases, organizations adopt a hybrid strategy and combine private infrastructure with public clouds. It is called ***hybrid cloud***. ***Virtual private cloud*** (VPC) [11] and ***community cloud*** [9] are build on top of the public and private clouds [8]. A VPC utilizes resources of public cloud with additional features of virtual private network. It provides support for customizable network topology an security settings [8]. In some cases, organizations with shared business objectives decide to collaborate with each other and form a common cloud by combining their private clouds. It is referred as community cloud.

Different deployment models have their advantages and disadvantages. Public cloud offers advantage to organizations of leasing resources from third party only when they need and do not require investing on infrastructure. However, in public clouds, application and data is hosted at third party's premises; so the organization has less control over applications and data. In private cloud, an organization has control over resources but it requires investment and maintenance of infrastructure and requires additional training of staff. Private cloud is more suitable with high data sensitivity requirements. The organizations having their private infrastructure offload some processing on public clouds during peak hours. The hybrid approach enables the organizations to have their data on secure premises and utilize processing capabilities of cloud. This approach also has a drawback as it introduces latency delays as a result of network speed limits that may become a significant problem when public cloud infrastructure is at a distant geographic location. Community cloud provides more control over data and resources but have less flexibility of resource acquisition because of limited resource availability.

Cloud computing paradigm provides enormous opportunities for having cost effective solutions, but it also requires some special considerations for architecting applications. Many solutions have been devised to address application architecting challenges and many research areas are still under infancy. The work being presented in this report aims at providing a systematic map and review of published literature related to software architecting for cloud. This work provides a brief overview of the research areas that are under infancy and needs attention of the researcher in order to effectively utilize feature of this emerging paradigm. One of the important challenges is to ensure privacy of users data when all the tenants are sharing same instance of

application running on cloud. Processing on sensitive data and constrains on data movements are important factors to consider. Moreover, satisfying service level agreements (SLAs) and quality of service (QoS) requirements of each individual application tenant when all are sharing same instance of application is a feature unique to cloud computing. In order to satisfy the challenges of cloud computing; applications build for cloud requires specialized architecting techniques. We have also evaluated how practitioners are utilizing existing architecting techniques to resolve cloud-computing problems.

The report is organized as follows. Section 2 provides a comprehensive view of our research methodology. Section 3 provides overview of included studies, different categories of themes representing classification of included studies and results. Section 4 addresses threats to research validity. Section 5 provides conclusions and directions for future work.

2. Research Method

We have conducted this research following the formal guidelines of systematic literature review/mapping study [13]. It is a repeatable process for extracting and interpreting available material related to the research objectives. This approach begins with identifications of research objectives and developing research protocol. In next stage, search strategy is defined along with inclusion and exclusion criteria. It follows with identification of publication venues and defining strategy for quality assessment. After establishing foundation for review, searches are performed on target data sources to select relevant studies as defined in inclusion/exclusion criteria. Selected studies are accessed for quality according to research objectives as well as inclusion and exclusion criteria. In final stages of the review data is extracted from selected list of sources and results are synthesized for presentation. Following subsection describes our research process.

2.1. Review Protocol

The review protocol was formulated based on the guidelines of systematic review/mapping study presented in [13]. The review protocol described background of research, research objectives and detailed research questions, criteria for inclusion and exclusion of target studies, search strategies, selection of target electronic data sources along with customized search string for each data source, detail of search and selection process for relevant publications, and data extraction and synthesis. The protocol also specified measured to access quality.

2.2. Objective and Research Questions

The objective of our research was to find the published literature related to application architecting for cloud computing. Following were the main objectives of the review study.

- RQ1: What are publication trend and publication venues of the studies related to software architecture of cloud-based systems?
- RQ2: What are different dimensions of software architectural solutions for cloud being focused by researchers?
    - RQ2.1: What are different problems addressed in each dimension?
- RQ3: What are different architecture styles that can be used to address the problems?

- RQ3.1: What architecture solutions have been reported in literature with respect to each dimension for architecting application?
- RQ4: What are different platform tools and technologies that can be used for building applications?
- RQ5: What are different application domains utilizing cloud-computing paradigm for providing solutions of business problems?

2.3. Search Strategy

We had performed our searches on scientific electronic databases that were accessible online. We did not specifically look for information in books and printed sources. Table 1 presents a list of our selected electronic database. We selected these databases because these include all high impact factor conferences and journal publications. Hence, there was no need to look into other electronic sources of information.

Table 1: Selected Electronic Databases

Electronic Database	URI
IEEE	http://ieeexplore.ieee.org/Xplore/
ACM	http://dl.acm.org/
Springer	http://www.springerlink.com/
ScienceDirect	http://www.sciencedirect.com/

Cloud computing and its services are referred by many different ways in literature. Literature related to cloud computing is often referred with its different service models such as infrastructure as a service, software as a service and platform as a service. Therefore, we had included names of different service models while preparing our search string. Moreover, for making sure that we did not miss any relevant paper we had also included different names used for cloud computing including cloud and cloud technologies in our search string. Following search string represent our generic search query utilizing terms related to cloud computing and combining AND and OR operators.

(“cloud computing" OR "cloud" OR "cloud technologies”) AND ("architecture" OR "architectures" OR "software as a service" OR "SaaS" OR "platform as a service" OR "PaaS" OR "infrastructure as a service" OR "IaaS")

We had customized this generic search query according to standard of each of the target electronic database, so that we can get more accurate and refined search results. Details of the customized search queries can be found in Appendix A. We had performed searches using customized search strings on document metadata including both title and abstract in order to make sure that none of the relevant studies were missed. EndNote library was used to maintain studies references. Figure 1 shows searching and study selection process, which was a multi-staged process and was performed as described in following lines.

- Searches were performed in target electronic database using customized search queries and we got initial set of 1491 studies.
- After excluding studies from primary set of studies according to inclusion and exclusion criteria, and after reading title and abstract of the studies; we shortlisted a secondary set of 225 studies.

- In third stage separate EndNote reference libraries for electronic databases were combined and three duplicate entries were removed. It step gave us 222 studies.
- In the fourth stage papers were filtered reading introduction and conclusion. It produced a list of 111 studies. We used these studies for data extraction process.
- The studies that were discovered irrelevant during data extraction process were also excluded. Finally, we included 86 papers in our study.

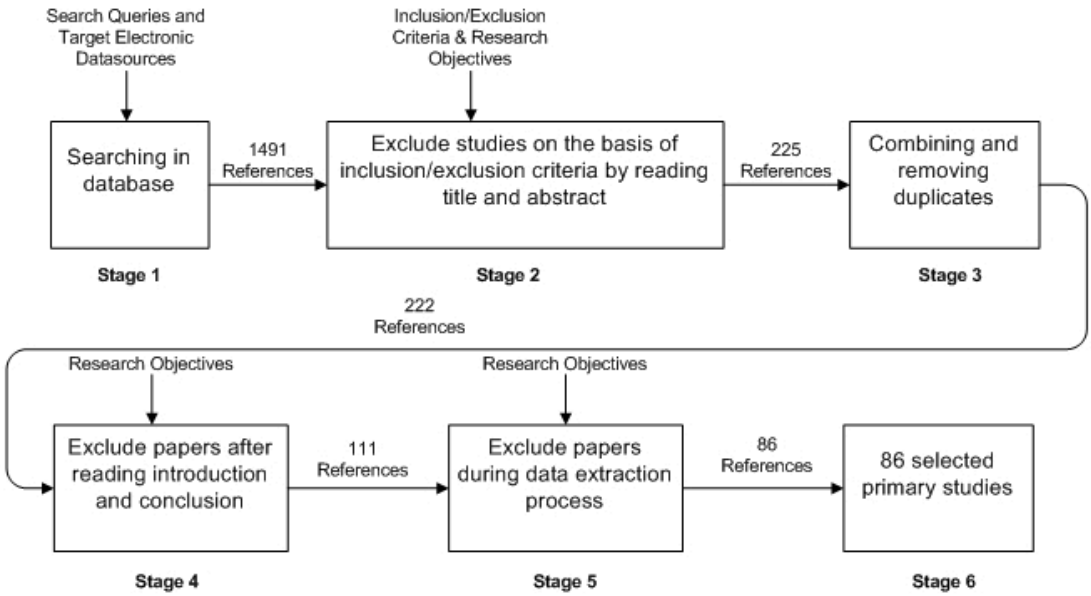

Figure 1: Studies Selection Process

2.4. Inclusion and Exclusion Criteria

The goal of defining selection criteria was to find and include all relevant published literature in our review. We had selected paper published in peer reviews journals, conferences in workshops up to July 2011. Only studies published in English language were selected. We did not specify initial year (lower boundary) so that all relevant studies could be included in our review. We excluded studies that were not related to software architecture on cloud or were not addressing any aspect of software architecture. We also discarded papers on panel discussions, tutorial summaries, presented slides, prefaces and editorials. In case, the different versions of the same paper were published, we had included only most comprehensive version and had discarded papers discussing parts of the comprehensive version. Table 2 shows our inclusion and exclusion criteria.

Table 2: Inclusion and Exclusion Criteria

Inclusion Criteria	Exclusion Criteria
Material related to software architecture of cloud-based systems publishes till July 2011, focusing on architecture related problems and their solutions, architecture styles being used, and describing tools as well as technologies being used in cloud-base systems.	Material not related to software architecture of cloud-based systems or published after July 2011.
Published in peer-reviewed venues.	Published in non-peer reviewed venues.
Research papers published in journals,	Panel discussions, presented slides, prefaces,

conferences, workshops and books.	tutorials and book reviews.
Material published in English language.	Papers published in languages other than English.

3. Overview of the Included Studies and Results

We carried our data extraction and synthesis process by reading 86 selected papers. In order to have a consistent data extraction process we used data extraction form presented in Table 3. Bibliographies and extracted data were maintained in EndNote, spreadsheets and word documents. The extracted data is organized into groups based on main focus of paper we refer as categories of theme. The categories of theme present summary of papers describing architecting challenges and their solutions, architectural changes required during migration of existing system to cloud and a list of papers presenting solution to domain specific cloud computing applications.

Table 3: Data Extraction Form

Attributes	Description
Study Identity	Unique identity for the study
Reviewer	Reviewer's name
Review Date	The date of data extraction
Bibliographic References	Author, year of publication, title and source of publication
Type of Study	Book, journal paper, conference paper, workshop paper
Focus of the Study or Study Category	Main topic area, concepts, motivation and objective of the study
Extracted Data	Description of the problems, architecture solutions, architecture styles being used to solve problems, tools and technologies used and application domains which are taking advantage of cloud computing.
Constraints and Limitations	Where applicable Identified constraints and limitations in the application of an approach as well as the identified areas for future research
Technologies Used in Implementation	Which tools and technologies are used for implementing the architecture?

Table 4 provides mapping between research questions and fields of data extraction form. Table 5 provides mapping of research questions to the relevant sections of the study and provides a quick reference to the sections containing answers of the research questions.

Table 4: Mapping of Research Questions to Fields of Data Extraction Form

Research Questions	Data Extraction Form Attribute
RQ1	Bibliographic References, Types of Study, Focus of Study or Study Category
RQ2	Type of Study, Focus of Study, Extracted Data
RQ2.1	Focus of Study, Extracted Data - Problems
RQ2.2	Extracted Data – Architecture Solutions
RQ3	Extracted Data – Architecture Styles
RQ4	Extracted Data – Tools and Technologies
RQ5	Extracted Data – Application Domain

Table 5: Mapping of Research Questions to Relevant Sections

Research Questions	Relevant Sections
RQ1	Section 3.1, Section 3.2, Figure 2
RQ2	Section 3.3, Table 9

RQ2.1	Table 11, Table 12, Table 13, Table 14, Table 15, Table 16
RQ2.2	Section 3.3, Table 11, Table 12, Table 13, Table 14, Table 15, Table 16
RQ3	Section 3.4
RQ4	Section 3.5
RQ5	Section 3.3.10

3.1. Publication Status of the Data Sources

It has been mentioned that we had performed our searches on IEEE, ACM, Springer and ScienceDirect. Table 6 shows number of papers published by each electronic data source. IEEE is leading by publishing 55 paper addressing software architectures related aspects of cloud computing system.

Table 6: Number of Papers Published in Data Sources

Data Source	Number of Papers Published
IEEE	55
ACM	15
Springer	12
ScienceDirect	4

Table 7 shows publication sources with more than one study publication. Other than IEEE International Conference on Cloud Computing there is no publication source with significant number of publications targeting software architecture. The comprehensive list of publication sources along with number of publication in each source is shown in Appendix B.

Table 7: Publication Sources versus Publication Count

Publications Source	Publication Count
IEEE International Conference on Cloud Computing	11
IEEE International Conference on e-Business Engineering (ICEBE)	3
ACM SIGPLAN conference companion on Object oriented programming systems languages and applications	2
International conference on Management of data	2
Springer Journal for Cloud Computing	2

3.2. Publication History and Trends Over Years

Cloud computing is relatively new area and covers multiple dimensions of computing environments ranging from managing hardware and software infrastructure resources for providing virtualized abstraction of these resources for constructing applications using cloud computing infrastructure. Our search results show that first paper addressing software architecture was published in 2008. In that year, there was only one paper published addressing architecting cloud applications. However, as more and more organizations are adopting cloud, numbers of publications addressing application architecting is also increasing. Table 8 shows numbers of papers published in 2009 and 2010. However, there are only 16 papers in 2011. The possible reason for less number of papers can be that we performed searches in July 2011 and all the papers had not been available by that time.

Table 8: Publication Trend over Years

Year	Number of papers published
2008	1
2009	22
2010	47
2011	16

3.3. Classification of the Studies

During the data synthesis phase we have grouped studies into categories according to their contents and research focus. The papers are grouped together in different categories of themes considering if they are focusing on a common set of problems and are proposing architectural solutions for these problems. The categories were identified after analyzing data that we extracted from papers to address our research objectives. We have identified set of components for each category of solutions. Categories that contain large number of papers are further sub divided into sub categories. Papers that are focusing on more than one aspect are included into multiple corresponding categories or subcategories. Sub groups present narrow classification of papers included in each category. Table 9 lists main categories of papers included in our study along with number of papers.

Table 9: Categories and Number of Papers in each Category

Category	No of Papers
Quality attributes	14
Multi-tenancy	3
Frameworks	3
Workflow processing and security	4
Architecture support for hybrid devices	6
Middleware infrastructure and platforms	9
Architectural considerations for migrating applications to cloud	4
General architecture guidelines	11
Challenges and emerging research areas associated with architecting cloud applications	3
Cloud Application Domains (applications implemented using cloud infrastructure)	34

Contribution of each research category in overall research of cloud computing is presented in Figure 2. Number of papers addressing research problems constitutes 63% of the selected studies whereas 37% papers are presenting domain or application specific solutions. The distribution of the studies shows that cloud computing is not only field of research but it is also highly applicable in commercial applications for solving problems associated with their ever growing resource requirements.

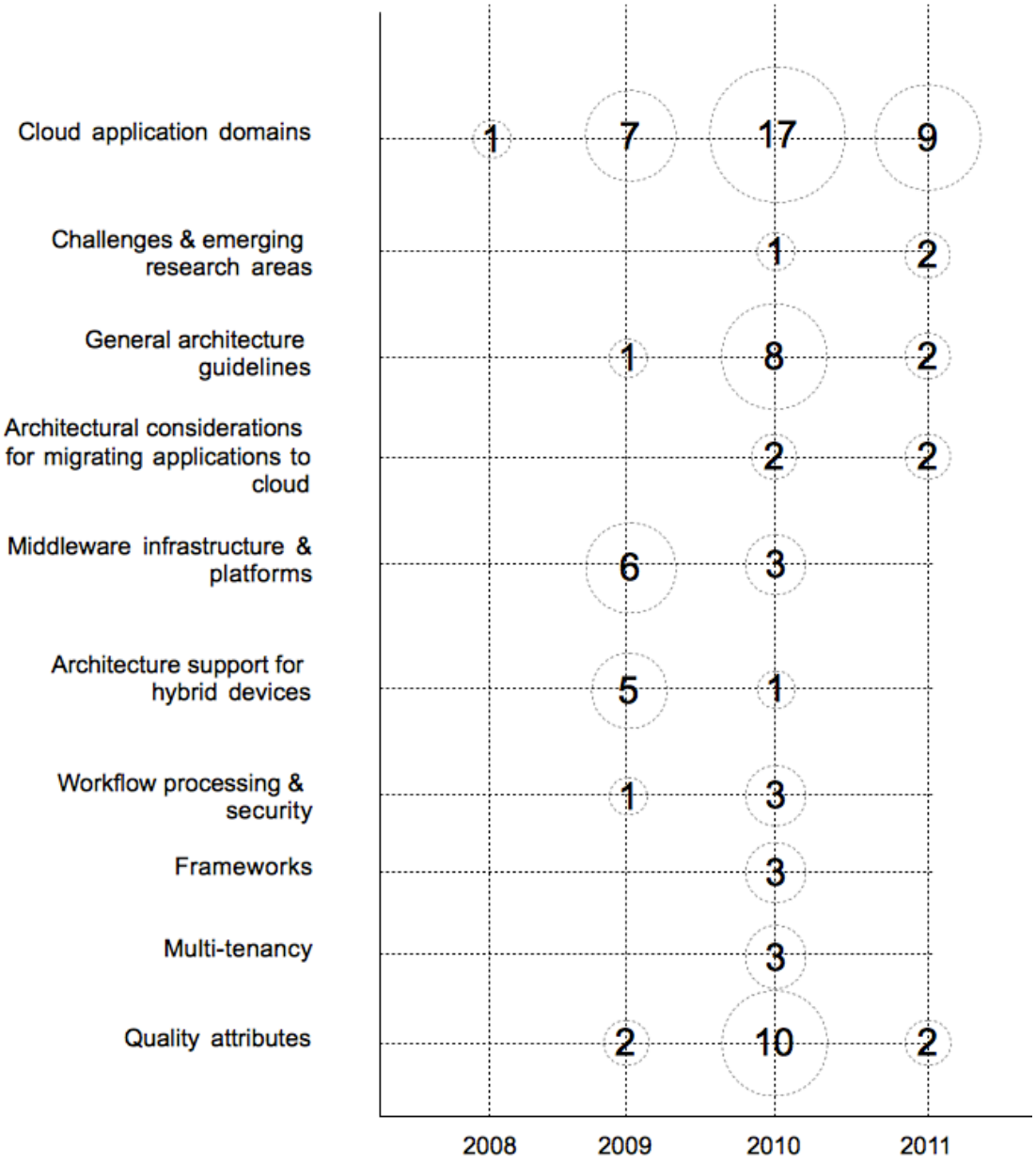

Figure 2: Bubble Graph Showing Distribution of Each Category over Years

3.3.1. Quality Attribute

Studies included in quality attributes category focus on cloud specific extra functional requirements. These requirements help exploiting cost effective resource provisioning and scalability features of cloud. Table 10 presents sub groups in this category.

Table 10: Subgroups of Quality Attributes

Quality Attribute	No of Papers	Study Reference
Energy Optimization and Efficiency	1	[14]
Service Adaptability	2	[15, 16]
Reliability	1	[17]
Resource Scalability and Provisioning	2	[18, 19]
Service Level Agreements	2	[20, 21]
Services Integration	5	[22-26]
High Performance Cloud Computing	1	[27]

Table 11: Problems Addressed in Quality Attribute Category and Solutions

Purpose	Solution Approach/Method	Study Reference
Energy Optimization and Efficiency	Optimal execution path of business processes and services to maximize QoS score.	[14]
Service Adaptability	Managed and unmanaged reactors for transparent adaptability of services. Mediation and negotiation services for handling environmental changes and system failures.	[15, 16]
Reliability	Insurance agents to support matching between service provider and service consumers according to SLA and QoS requirements, and charging insurance fee if services fail to comply to standards.	[17]
Resource Scalability and Provisioning	Services handling scalability with the help of elasticity rules for resource allocation, services replication, services migration and service composition.	[18, 19]
Service Level Agreements	A proxy based middleware service for monitoring service contract by observing interaction points between service providers and service consumers. An arbitrator service responsible for dispute resolution.	[20, 21]
Services Integration	Registry services and Enterprise Service Bus (ESB) for handling authentication, routing schemes and communication between services.	[22-26]
High Performance Cloud Computing	Service registry, service negotiation, service accountability and on demand licensing.	[27]

Table 11 presents purpose served in each category along with solutions. The architecture solutions presented in table are summary of the information presented in each sub category described in following subsections.

A. Energy Optimization and Efficiency

One of the main advantages of moving to cloud is having applications with minimum energy footprints. Approach presented in [14] considers services at business process level, service level and physical level for calculating execution path and determining optimum execution plan in terms of maximum quality of service (QoS) score. QoS score for each execution path is average of normalized QoS values.

B. Service Adaptability

It allows services to be deployed on heterogeneous cloud environments and with self-organizing deployment mechanisms services can manage resource provisioning without human administration. Services build using REST architecture style are easily adoptable because of their stateless nature. The term liquid services is used in [15] for adaptive and transparently deployable services. In the proposed model, every resource (reactor) can be a service provider as a well as service consumer. Web service application has one entry reactor to accept requests from the external clients and to provide the final results. It further categorize reactors into two types, managed (whose business logic is maintained within the application) and unmanaged reactors (proxies to external third party RESTful web services used by managed reactors to provide final services). The approach presented in [16] uses service mediation and negotiation techniques to handle environmental changes and system failures. However, both of these approaches neither discuss cross platform adaptability nor describe how services can be ported from one environment to other when they are live.

C. Reliability

Reliability deals with system functioning according to the desired behavior. It is measured in terms of systems availability and correctness of the results produced. It is more complicated to ensure reliability on cloud enables systems. These systems are often built on top of infrastructure and services provided by external providers. If any of the services fails to meet reliability standards, it will impact whole system. In order to cope with this challenge, the system should be able to dynamically find replacement services in case any of existing services fails to meet reliability conditions. In order to have risk free collaboration, services build using cloud infrastructure also requires assurance for reliability. An insurance based reliability scheme is introduced in [17] for achieving this quality attribute. Insurance agent supports matching between service provider and service consumer. It also guarantees that service providers offer services according to QoS requirements by charging insurance fee from each of the service provider in case they fail to meet SLA. A mathematical model to predict bankruptcy risk is also presented. According to this scheme, aggregated service provider is responsible for reliability of all the composed services and it does not take into account reliability of composed services.

D. Resource Scalability and Provisioning

Scalability and resource provisioning can be achieved by defining elasticity rules and load balancing algorithms [18, 19]. It requires explicit declaration of rules for resource allocation, replication, migration and decomposition of composite services as well as each individual service inside composite ones. The language for describing constrains on architecture for cloud computing infrastructure that can deal with resource allocation and implementation of service life cycle management process is also discussed in [18]. A dynamic scaling algorithm monitors the performance of application instances and launches new instances on pre-configured virtual machine images. It also handles shutting down of running instances if load is reduced. It considers number of concurrent users, number of active connections, number of requests per second and average response time per second are important scalability indicators for monitoring the scalability requirements of the services and application on cloud [19]. However, these studies do not describe applicability of this technique on PaaS environment where users don't have specify scalability rules for aggregated or composed services.

E. Service Level Agreements

Service level agreements facilitate services and infrastructure offerings. The studies [20, 21] emphasize introduction of a new layer for monitoring performance of services and [21] introduces concept of service accountability in REST architecture. In centralized accountability approach a proxy middleware service or service contract monitor is responsible for monitoring interaction point between all service providers and consumers using contract metadata and SLA requirements. In peer-to-peer approach, each service itself is responsible for SLA compliance and an arbitrator service is assigned the responsibility of dispute resolution.

F. Services Integration

Integration methodology presented in [22] describes a proxy-based firewall/NAT traversal solution for SaaS (PASS) integration. It allows SaaS applications to integrate with on-premise applications with reconfiguring firewalls. SaaS requires the integration with the existing frameworks in three different ways: as a component of

the whole business process, as a business process engine or as a combination of both. It also covers service registration methods, authentication and routing schemes, and communication mechanisms between services. However, this technique does not take into consideration sensitive data flow stored on organization's private infrastructure to SaaS components.
Services integration techniques presented in [23-26] suggest the use of service provider's module for providing service interface to service controller. Service controller (binding component) is responsible for establishing seamless communication between services and integrating services with enterprise service bus (ESB). These studies also describe negotiation methodologies for quality of service (QoS) requirements and role of collaboration agents in the negotiation process.

G. High Performance Cloud Computing

The tremendous computing and storage resources offered by cloud can be utilized in high performance computing systems. The study [27] presents an high performance computing application using SOA and cloud principles. The concepts of service encapsulation, service registration and discovery, service negotiation, on demand licensing and accounting are implemented in order to achieve high performance.

3.3.2. Multi-tenancy

Multi-tenancy allows SaaS applications to serve multiple clients through same instance of application. Multi-tenancy in cloud enabled system can be achieved at different levels of abstractions, i.e. at service level as well as at data level. Papers classified into this category described how multi-tenancy is supported by middleware infrastructure on cloud by application after separating persistence and persistence related logic and by routing service requests to tenant specific instances of components.
Multi-tenancy at middleware is supported by assigning separate service instance from repositories to each tenant [28, 29]. Information of services corresponding to tenants is maintained in service registry and a message handler at server routes client requests to services with the help of message dispatcher. A security component is responsible for handling user management by considering roles assigned to each user belonging to a specific tenant along with resources assigned to tenants and actions that users can perform on the specific resource. Services of a specific tenant have their own infrastructure, persistence units and management console for managing infrastructure.
Multi-tenancy at database layer is achieved by implementing a database driver capable of maintaining database index that can provide a mapping between tenants and their databases [30]. Moreover, inside each database, every table has an additional column that specify owner of the data. Having this type of multitenant database index allows applications to query databases in normal way without explicitly selecting separate databases for each tenant.
Table 12 lists the problems addressed by multi-tenancy and lists architecture solutions.

Table 12: Problems Addressed in Multi-tenancy Category and Solutions

Purpose	Solution Approach/Method	Study Reference
Isolation between services	Assigning separate service instances to each tenant. Service registry to define mapping between tenants and service instances. Security services for managing user roles of	[28, 29]

	services belonging to specific tenants.	
Isolation between data stored in databases	Multi-tenant database drivers for maintaining mapping between tenants and their databases as well as maintaining tenant specific indexes on data.	[30]

3.3.3. Frameworks

This category includes papers describing frameworks that can be used in application development and deployment on cloud. For facilitating application development for cloud computing environments the frameworks support requirements management, system design, application discovery, workflow, scheduling of services, and services instantiation on cloud infrastructure [4, 31, 32]. It will facilitate applications development in accordance with features provided by target cloud environment.

Framework presented in [31] comprises of tools for requirements management, cloud system and architecture design, architecture assessment, model based scalable programming, prototype simulation, application deployment and configuration for cloud operating system stack and service repository.

Aneka framework presented in [4] consists of middleware broker, responsible for suitable data sources, workflow manager for handling representation and execution of applications as workflows, market maker or meta-broker for matching user requirements with capabilities of the service providers, components for inter grid resource sharing and energy optimized service scheduling.

The framework described in [32] presents a systematic engineering process for facilitating design and development of SOA based applications on cloud. This framework consists of a gateway application installed at customers' datacenter and automatically manage cloud deployments; virtual machines that runs on behalf of gateway application on each target cloud; and an orchestration layer protecting customers' services, storage and networks on the cloud.

Table 13 lists the problems addressed in this category and lists architecture solutions.

Table 13: Problems Addressed in Frameworks Category and Solutions

Purpose	Solution Approach/Method	Study Reference
Tooling support	Tools for supporting cloud application development supporting requirements management, system and architecture design, architecture assessment, model based scalable programming and simulation.	[31]
Finding services for application development and deployment	Broker services for locating appropriate services and data sources, workflow managers for managing application workflows, and components for handling resource sharing and optimized service scheduling.	[4]
Design and development of SOA based application	Autonomous management of cloud deployments and virtual machines, and an orchestration layer for protecting customers' services, storage and networks on cloud.	[32]

3.3.4. Workflow Processing and Security

Papers included in this category provide architectures for trust worthy and continuous data processing in open distributed systems, workflow management for processing

huge volumes of data and distributed processing on sensitive data [33-37]. This section focuses on architecture constructs for taking advantage of combining private cloud with public infrastructure in workflow bases system.
The data processing approach presented in [33] propose that every component should know about minimum part of the workflow rather than knowing all of it. Components keep track of each data unit they receive and produce and know only about its upstream and downstream components. A randomized consistency check is performed for verification and integration of scalable dataflow processing. For distributed processing of data, use of cloud side engine and client side engine is presented in [35]. These engines are responsible for managing components' configurations according to data sensitivity requirements as well as with respect to processing capabilities of infrastructure at client and cloud side. The studies [34, 36, 37] describe components of workflow systems and multiple stages involved in data processing. The components are distributed among user layer, infrastructure layer and middleware. Cloud side engine and user end engine is responsible for components distribution on user end and cloud end. Data collection, re-projection, analysis, task scheduling and processing correspond to multiple data processing stages.
Table 14 describes the problems addressed in this category and lists architecture solutions.

Table 14: Problems Addressed in Workflow Processing Category and Solutions

Purpose	Solution Approach/Method	Study Reference
Security of processing nodes in workflow and data that is produced	Components keep track of data that they receive and produce, and components know only about their immediate upstream and downstream components.	[33]
Processing of sensitive data	Cloud side engine and client side engine managing components' configurations according to data sensitivity requirements.	[35]
Complex computing on data	Pipeline based data processing strategy consists of multiple stages including data collection, re-projection, analysis, task scheduling and processing.	[34, 36, 37]

3.3.5. Architecture Support for Hybrid Devices

Papers grouped in this category shows prospects and usage of mobile devices in combination with cloud computing. The ability of being part of network enables them for utilizing features of cloud computing for building smart and reliable applications [38, 39]. Limitation of processing and storage resources on mobile devices can be overcome by taking advantage of automated, dynamic and reliable runtime configuration of components on mobile devices and cloud end [39, 40]. Spatial and context awareness features allow building location aware services on mobile devices [38, 40, 41].
The studies [42, 43] discuss distribution schemes of components on mobile devices to perform complex tasks. The technique described in [43] divides components into two categories; i) components used for interaction with end users and ii) components used to perform business logic. Components used for business logic can be deployed on available mobile devices even though the devices are not being used by end users for interacting with system. The study [42] presents algorithms for selecting optimal configuration of components on mobile and on cloud to take maximum advantage of

mobile platforms and network bandwidth while keeping cloud resources and cost and bare minimum level.
Table 15 describes the problems addressed in this category and lists architecture solutions.

Table 15: Problems Addressed in Architecture Support for Hybrid Devices and Solutions

Purpose	Solution Approach/Method	Study Reference
Limited processing and storage resources	Automated, dynamic and reliable runtime configuration of components on mobile devices and cloud end.	[39, 40]
Location aware services	Utilize special and context aware features of mobile devices to build location aware services.	[38, 40, 41]
Component's distribution schemes	Classification of components into two classes: user interface component and business logic components. Using appropriate selection algorithm for components configurations in order to maximize utilization of resources on mobile platforms and network resources while minimizing resources cost on cloud side.	[42, 43]

3.3.6. Middleware Infrastructure and Platforms

This category contains papers addressing middleware software platforms supporting hosting of services and applications on cloud infrastructure and their interaction with each other. The studies [20, 39, 44] describe middleware support for reliable automated deployment of services and components on cloud as well as on front-end machines according to QoS requirements. The communication between services satisfying QoS agreements is also the focus of these studies. Representational State Transfer (REST) based approach for managing infrastructure and REST based services on cloud is described in [45, 46]. The techniques separately maintain configuration of distributed services and services associated with domain aware processes. This configuration is managed at three different levels of isolation: information common for all services containing contract information and device configuration, standard management information of services defining standard specifications, and private information of services like QoS requirements and SLAs. Altocumulus framework by IBM is described in [47] which address intra operability issues between clouds. It provides Dashboard supporting interaction point between end users, a core encapsulating rules and cloud adapters for supporting execution on different cloud platforms. An extended YML framework for scientific computing is presented in [48] explaining different components of the framework used in scientific computing.
SaaS applications on cloud can server a large number of tenants that can use different versions of the same software. Architecture for maintaining multiple versions is presented in [49] and suggested need of version management system module capable of maintaining partitioning between different versions of the system and capable to bundle specific version of the system with respect to a particular tenant's subscription. The platform presented in [50] allows home services to be selectively opened to remote users and semi-trusted external services. The motivations to host data and services at home are large and cheap storage availability at home as compared to the online, data privacy with full administrative and legal rights, real-time data access if the data is available directly from home and common repository with all contents at

the single location. Two steps request filtration mechanism is suggested by using with trusted proxy server on public Internet as first level of filtration and tunnel server between Internet and home network as second level of filtration.
Table 16 describes summary of the problems addressed in this category and the architecture solutions.

Table 16: Problems Addressed in Middleware Infrastructure Category and Solutions

Purpose	Solution Approach/Method	Study Reference
Reliable automated deployment of services and components	Middleware services to handle deployments and inter service communication according to quality of service requirements.	[20, 39, 44]
Infrastructure management	Separating business logic from services that is maintaining services distributions strategy.	[45, 46]
Intra operability issues between clouds	Providing interaction points between service consumers and core services. Cloud adapters for supporting inter operability between clouds.	[47]
Service versioning	Version management modules for maintaining partitioning between different versions of systems and services. Services bundling according to clients' subscription.	[49]
Access to remote users and semi-trusted external services	Trusted proxy servers on public internet along with tunnel servers between Internet and network.	[50]

3.3.7. Architectural Considerations for Migrating Applications to Cloud

Migrating existing applications on cloud requires changes in architecture to utilize cloud utility model and offering solutions as services. The papers grouped in this category layout guidelines addressing architecture modifications required for porting applications on cloud. Architectural modification process is driven by objectives behind moving applications to cloud, cloud attributes of the new system and architecture evaluations for target cloud platforms [51]. Use of key-value store is suggested in [52] along with functional decomposition for avoiding bottlenecks of key-value stores persistence approach. Scalability, orchestration, throughput, asynchronous communication and multiplatform support are key quality attributes suggested in [51, 52]. A technique for supporting migration of live databases [53] from one cloud to another cloud proposes migrating data to the destination node before making modification in it when a modification request is received during data migration. A case study presented in [10] shows 36% cost reduction and 21% reduction in support calls after migrating IT system of an oil and gas firm.
There are few aspects related to architecture modification for cloud-based systems that are not covered in published literature so far and required attention of researchers. Most of the existing systems being used by organizations now days are network based or web based system not built following SOA. There is also a need to investigate whether these systems can be migrated to cloud keeping their centralized architecture intact or it is required to transform them into SOA for making them eligible to take advantages of cloud-computing paradigm. Moreover, in many cases; moving these systems to cloud may require significant transformation and many additional components to control and adapt themselves according to changes in environments.

Tools and technologies support by cloud hosting platforms in general and PaaS in particular may not fulfill centralized applications.

3.3.8. General Architecture Guidelines

Studies presented in [54, 55] layouts factors to consider for considerations while building applications for cloud. The main considerations are integration with existing ecosystem of application, exposing reusable business logic as services, subscription based resource provisioning of cloud resources through application and service offerings according to QoS requirements. The cloud aware approach is presented in [55] that suggests building applications after considering underlying cloud virtual infrastructure of the target platform in order to achieve application's non functional quality attribute requirements. The cost of CPU, storage and networking resource should be carefully analyzed for estimating overall application cost.

Multiple alternatives for implementing databases in cloud applications are presented in [56-59]. Data partitioning and replication is commonly used technique for handling data in distributed environment while exposing database services as REST interfaces is convenient approach for using a shared database by multiple services. Memory-based column database is presented in [58] to make aggregations and joins at run time without having materialized views. It consists of three types of processing components: cluster leader, router and instance managers. Cluster leader assigns the data to the instance manager, which takes care of multiple tenants. Assignment information is maintained in a cluster map and managed by cluster leader. Cluster map is propagated to the routers and instance managers. Cluster leader also track changes to the cluster state. As a response to the valid request, a change is made in the node for which request is received and later it is asynchronously replicated to other nodes.

The studies [59-61] describe techniques for utilizing distributed cloud resources. Google has developed a Map Reduced programming framework for supporting parallel computing and access to the database [60]. Client side execution, cashing, pre-fetching of data, paging scheme for faster data retrieval, environmental aware evolution and adaptation, autonomous handling of scheduling, prioritization, safety and security issues, design of product services, design of business cases and considerations for service delivery platform are key features of architecture for real time and enterprise service oriented cloud computing [59].

Risks associated with cloud computing are grouped into four categories: operational risks, contingent risks, security risks and business risks [62]. Operational risks deal with availability, reliability, integrity of services and data, service provision according to users' requirements and maintainability. Contingent risks address service and data survival, impact of major service interruption, frequency of interruption and resilience, as well as compatibility and flexibility. Security risks include service and data security, authentication and authorization, and susceptibility to denial of service attacks. Business risks involve cost, unsatisfied customer service by cloud providers, privacy breach, and legal conflict of application requirements with how infrastructure is maintained by cloud providers.

An approach to achieve pre-specified scalability of SaaS application and mash-up services using resources pooling from multiple clouds is presented in [63]. System level schedulers and load dispatchers are used for sharing load between services deployed on cluster of clouds. Some other system services are suggested for providing enhanced availability of the system. These services include fault identification services to determine services failures and possible cause of failure, actuator services

for finding remedies of the failures, proxies for providing encapsulation and orchestration, and modules taking care of adaptability of partially mismatching services.
A model for defining boundaries of sensitive data and achieving data privacy in SOA is discussed in [64]. The model suggests that services for managing data privacy and protection at run-time should take care of boundaries, ownership and legal requirements regarding the sensitivity of data. The use of separate management services on cloud and data hosting platform is also suggested.

3.3.9. Challenges and Emerging Research Areas associated with Architecting Cloud Applications

A considerable attention is being paid on architecture aspects of cloud computing despite the fact that it is a new area. However, there are few dimensions still lagging considerable work for example cloud services modeling, locating and selecting alternative services when no exact match of existing services is available, and programming models targeting cloud based systems. The process and component of the service modeling are presented in [65]. The process begins by specifying functional requirements using linguistics symbols following by transformation into fuzzy terms. In the next stage weights are assigned to linguistic variables and linguistics variables along with weights are evaluated in terms of QoS criteria. Evaluation results are shown to users and their feedback is captured. Feedback is used for the next round of group consensus. This paper presents an impressive approach of mapping requirements to QoS criteria when there is no exact match available. This kind of techniques can be very helpful in finding alternative services in real world when there is no exact match available.
Directions for programming models targeting cloud are presented in [15, 66]. These models help achieving autonomous distribution of business logic on cloud resources without explicit management. Control flow algorithms described in [66] elaborates distribution of business logic and operations in a program based on platform resources. Operations and data units in a program are divided and combined with the help of program controllers.
According to model presented in [15], web services are composed of many entities that correspond to each other through messages. Every resource can be a service provider as a well as service consumer. Web service application should have one entry reactor to accept requests from the external clients and to provide the final results. Reactors are further categorized into managed and unmanaged. Managed reactors are reactors whose business logic is maintained within the application whereas unmanaged reactors are proxies to external third party RESTful web services used by managed reactors to provide final services. This model helps achieve transparent deployment, infinite scalability and long life of services.

3.3.10. Cloud Application Domains

There are a large number of papers reporting architecture and implementation of specific applications on clouds. These applications are utilizing cloud computing infrastructure and providing domain specific solutions. Table 17 lists different domains that are utilizing cloud-computing infrastructure to provide cost effective and efficient solutions. The table shows that cloud computing is being used in all domains ranging from processing intensive video processing and health care systems to simple warehouse management and e-commerce applications.

Table 17: Application Domains using Cloud Computing

Study Domains	Study Reference
Video processing	[67-70]
Health care system	[39, 71, 72]
Traffic control system	[73]
Collaborative writing support tool	[74]
E-government	[75-77]
Simulation system	[78]
Warehouse management system	[79]
Framework for SOA test-beds (for testing SOA applications on cloud)	[80]
Cyber physical systems (integration of computation and physical processes)	[81]
E-commerce	[61, 82]
Program recommendation system for digital TV platforms	[83]
Travel reservation system	[84]
Application to monitor user experiences	[85]
Customer relationship management system	[86, 87]
Business intelligence	[88]
GIS systems	[89, 90]
Project management for IC design	[91]
Real-time monitoring system on mobile	[92]
Ubiquitous information agent	[93]
Freight system for railway	[94]
Online trading platform	[95]
E-learning system	[96]
Aircraft design	[97]
Cloud architecture for smart homes	[98]

3.4. Problems Addressed in Main Categories and Architecture Styles Used to Addressed these Problems

There are multiple architecture styles being used in the selected studies in order to solve problems addressed in main categories of themes. Following tables specifies the patters along with number of studies that are using architecture styles and their variants. These styles can be applied to solve problems of corresponding categories of themes as specified under "can be used for" attribute. Table 18 till Table 24 shows architecture styles extracted from selected studies.

Table 18: Service Management

Name: Service Management
Problem: When a large number of services are deployed on cloud, it is difficult to host services in a way that they are discoverable by the clients, and manage and compose them in a way to satisfy SLA of each client. It is also a challenge to enable cloud end services to collaborate with services hosted on customers' local environment.
Solution: Separate management engines are used at cloud side and client side to manage services. On cloud side, requests from different clients are handled through an orchestration layer. Once request is identified, a controller is used to send request to a respective service repository. Service composition module aggregates services in order to satisfy tenants' requests. Each client has its own local service registry to facilitate service discovery and composition for aggregation of services. Cloud side management engine and client side management engine hold responsibility for services integration hosted on client

and cloud. Activity scheduler at cloud side schedules and executes services according to SLAs. In some case additional services may required for managing virtualized hardware and software resource. When services are required to be deployed both on client side and cloud side, special management services for handling service distribution and composition is required to perform successful business operations. For having optimal execution strategies, task scheduling and intelligent resource provisioning services are required.

Impact:	**Study References:**
Ensures seamless integration and execution of services hosted on cloud and client environments.	[14, 19, 22, 31, 35, 36, 40, 43-47]

Is used for:
Supporting hybrid devices, middleware infrastructure, frameworks, achieving quality attributes, workflow processing and processing on sensitive data.

Reference Diagram:
Figure 3 outlines reference architecture and components for the pattern. Following is description of the components shown in the figure.

- REST Service Interfaces (RESTful Web Service)

This component provides interface of the system services in REST format to its clients. After receiving request, it forwards request to REST Event Handler.

- Generic Service Interfaces

It contains three different types of sub-services. Dashboard, provides interaction point to the system for end users. Service Portal provides unified interface to system services. Discovery service makes services discoverable to external users and systems.

- Service Modeling and Enactor

This module is used for service modeling and their realization. It consists of Application Modeling Tool that provides flexibility of composing services together and make them capable of performing some functionality. Workflow Manager supports linking services to each other in a workflow for processing data. Workflow Enactor is responsible for enactment of workflow.

- REST Event Handler

It received input via service interfaces and consists of two sub components, Asynchronous Event Sender and Asynchronous Event Handler. Asynchronous Event Sender receives event from interface and send notification to Asynchronous Event Handler. Asynchronous Event Handler stores notification into database and sends back acknowledgement (meaningful notification) if necessary. It also passes on this information to REST based Service Management Component.

- Service Management (REST)

It manages configuration parameters, handles logical and physical topologies, obtain monitoring information from database or monitoring components if there are any, handled necessary data logging, manages notification data from Asynchronous Event Handler, and interacts with Service Initialization and Realization Module for managing and invoking services.

- Service Management

SOA service management module consists of four sub-modules. SLA Management module keeps record of SLAs of different services and ensures that services performs according to agreed SLAs. Distributed Configuration Module takes care of distributed processing of components on cloud end and at the user end. All configuration items are discovered locally and exposed by service interfaces. This distribution is managed by considering SLAs, sensitivity of data and available computing resources at client end. Distributed Service Management module ensures seamless execution of distributed components, task completion or rollback in case services on client end fail to respond on time, security of the data transmission. Scheduler and Data Transfer Manager schedules tasks on cloud and user end components. It also handles data transformation and transmission between service on cloud and remote services and data security during transformation.

- Service Initialization and Realization

This layer handles initialization and maintenance of services on cloud virtual infrastructure. It also takes care of allocation of virtual resources to application components from cloud resource pool.

- Performance Estimation Service

This service calculates cloud resources required for satisfying SLAs. This module is used by Service Initialization and Realization module when services are invoked first time and deployed on cloud resources. This service is also used by Service Monitoring Module during pre-emptive measures for reallocation of cloud resources.

- Service Monitoring Module

This module monitor services' performance during their execution for their compliance with SLAs. If

services are finding it hard to achieve SLAs because of virtual cloud resources, then component interacts with Service Initialization and Realization module to adjust cloud resources to satisfy SLAs.

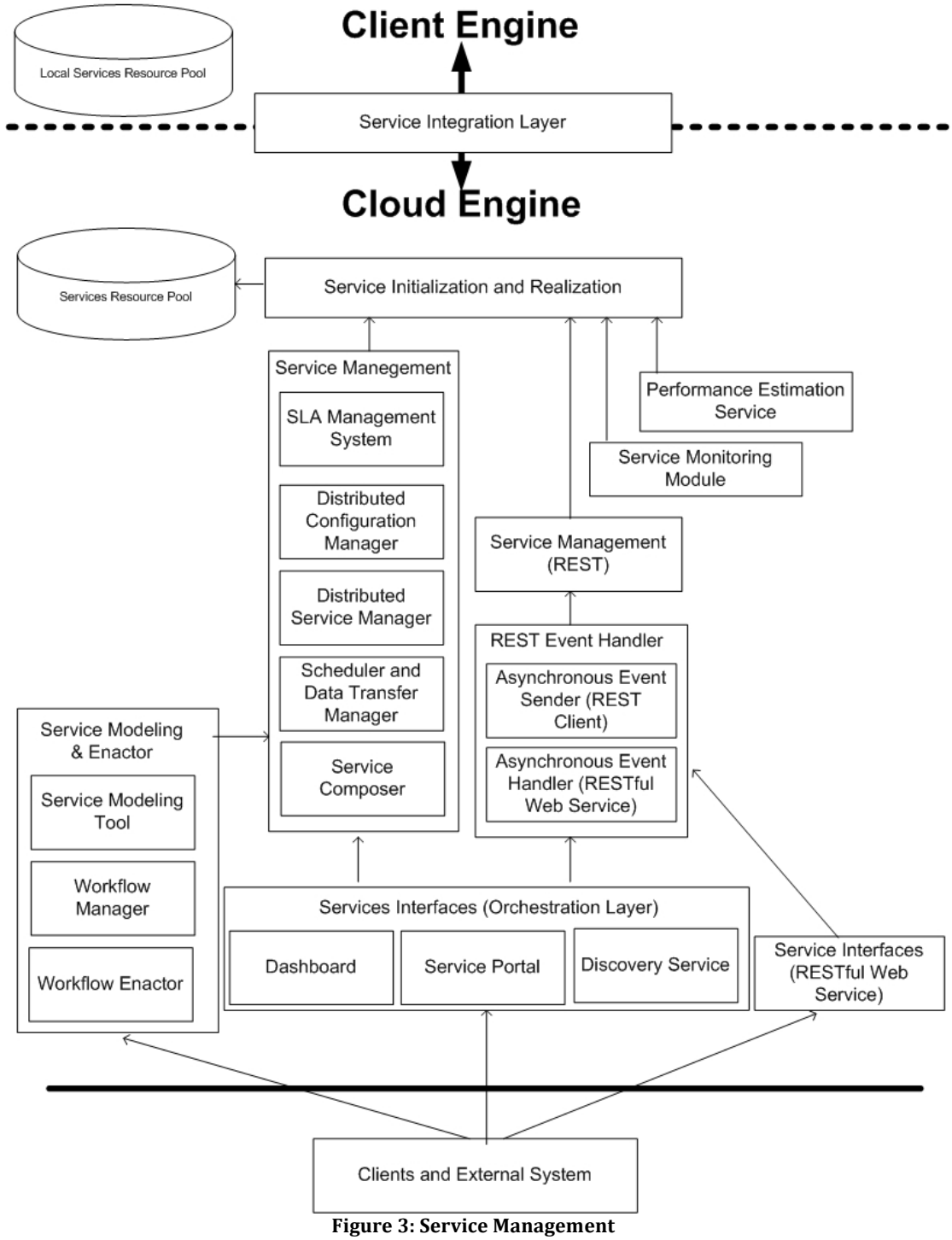

Figure 3: Service Management

Table 19: Service Request Handling

Name: Service Request Handling
Problem: When same system is serving multiple tenants, it is important to analyze request from clients in order to identify the tenant and then route request to tenant specific services. It is also important to identify client device so that system responses can be generated in a format recognizable by the client device.

Solution:
The request handling services are needed for providing interfaces to the clients. Routing management services take responsibility for routing request to appropriate services. When different types of clients request for services, services hosted at cloud need to adjust themselves according to capabilities of the clients' devices so that load can be optimally distributed between clients' devices and servers. Specialized services for request dispatching are required that can compose response in format understandable by client devices.

Impact:	**Study References:**
Ensuring proper handling of tenants' requests and composing response in format recognizable by client devices.	[28, 29, 41]

Is used for:
Multi-tenancy and Supporting hybrid devices.

Reference Diagram:
Figure 4 outlines reference architecture and components for the pattern. Following is description of the components shown in the figure.

- Request Handler

It receives requests from multiple types of devices and pass these requests on to routing engine. This service acts as firewall and filters requests from being passed to internal system services.

- Routing Services

Next set of services responsible for evaluating request is routing services. It consists of Request Parser and Validation Rules Engine. As name suggests, Validation Rules Engine stores rules for validating requests. Request parser parse requests using rules stored in rules engine. After parsing request, it extracts information of client device, target services, security information and authentication credentials. This service also utilizes two other types of services, logging services and security services.

- Security Services

This set of services consists of Access Controller and Authentication Rules using information passed by Routing services. Goal of Access Controller service is to verify if the client is authenticated to access the requested services and Authentication Roles determine to which version of the software client is subscribed with.

- Service Execution Manager

It receives information about service request, clients and for which version of the service clients are subscribed to. This information is used to pass to corresponding services if the services are already invoked and otherwise this module is also responsible for initiating those services with help of Service Initialization and Realization module. Other than initiation and realization of services, this module is also responsible for establishing communication bridges between services and service clients with the help of service dispatcher.

- Service Initialization and Realization

It keeps track of all available and active services as well as different version of those services. When it receives a request from Service Execution Manager it first looks up into running services and their heartbeat to evaluate if they can service request. If they can, then request is redirected to respective services, otherwise a new instance of service is invoked using underlying Cloud infrastructure for Virtual Computing Resources to handle request.

- Multitenant Database Drivers and Indexes

These components handle multi tenancy at persistence level. Introduction of this layer takes off responsibility of data isolation requirements of multi tenancy from services and can store data in separate physical/logical persistence units. This layer also introduces an additional security point for avoiding accidental access of data by other tenants. In case isolation for multiple tenants is handled by services themselves it poses a higher risk of undesired access. Moreover, faster access to data can be provided if database indexes are capable of multi tenancy because size of index tables will be reduced and search can be performed efficiently.

- Service Dispatcher

It dispatched request to client and transform this request to interface into interface that client is expecting. Requests can be made from different types of client machines and components using different types of protocols (SOA, REST) so it is required to return data in format that those devices are expecting.

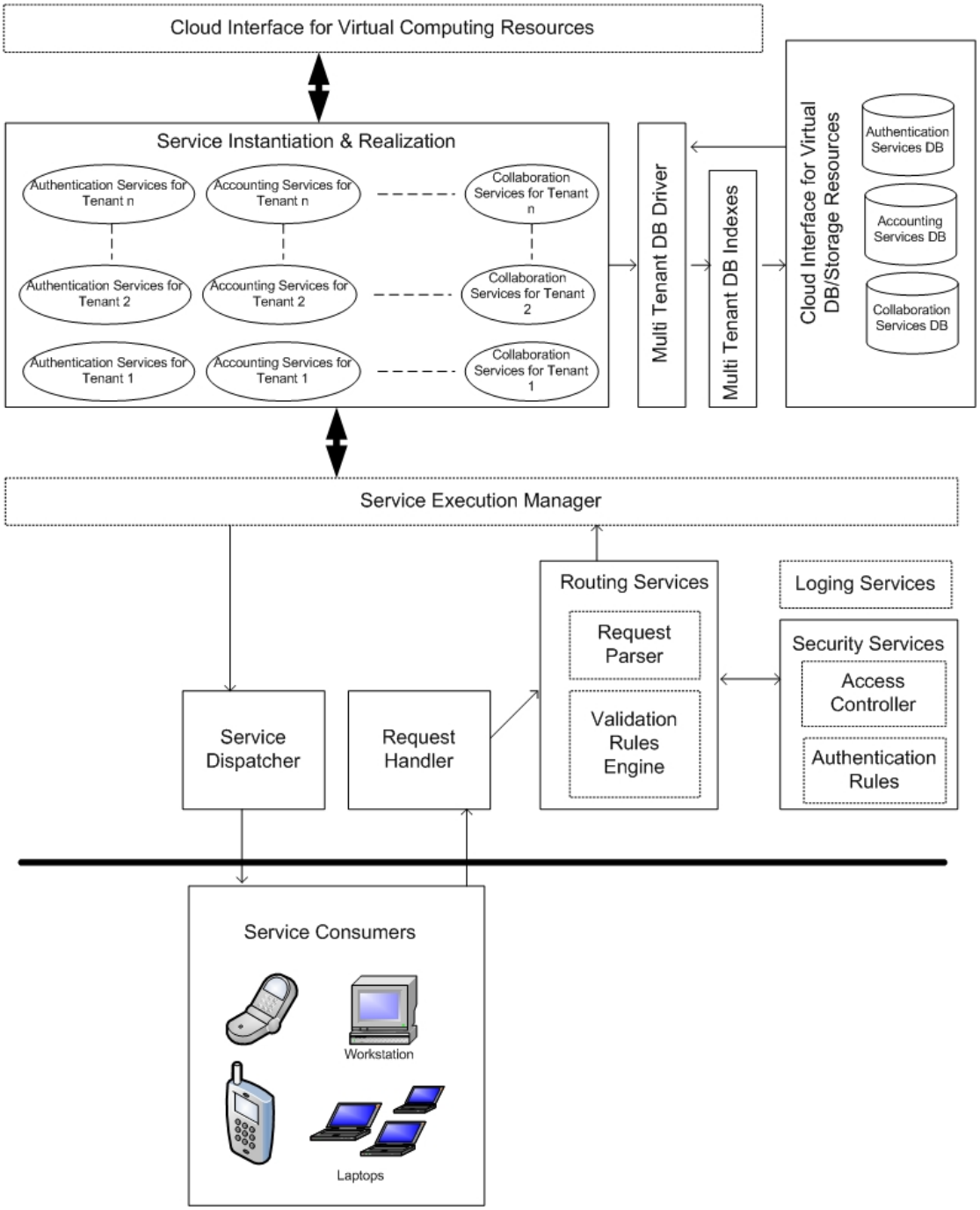

Figure 4: Service Request Handling

Table 20: Service Execution and Realization

Name: Services Execution and Realization
Problem: Services binding to the client according to SLA specification and execution following an optimal strategy for managing services life cycle are crucial. It is also not straightforward to present group of services involved in a business process for tenant to be represented as federated services. Moreover, run time services scalability and finding alternative of a service in case it fails to meet SLA is essential for keeping systems working as expected.
Solution: Intermediate services with the capability of automatically reading SLAs of service providers and service consumers can be used to make binding between qualifying services. Orchestration services need be used to act as a federation layer or façade for providing a unified view of system. Monitoring

services are used for observing services health and watch if services are operating according to service level agreements. If an inconsistency is detected, services can be scaled accordingly. In worst-case scenarios, if any of the services failed, then alternative services are to be located and rebinding of services has to be performed.	
Impact: Ensures that interlinked services are meeting SLAs and cooperating services act as a single unit to external clients.	**Study References:** [18, 19, 28, 29, 31, 34, 41, 43, 44]
Is used for: Multi-tenancy, supporting hybrid devices, middleware infrastructure, workflow processing, quality attributes and frameworks.	
Reference Diagram: Figure 5 outlines reference architecture and components for the pattern. Following is description of the components shown in the figure. • Service Management SOA service management module consists of four sub-modules. SLA Management module keeps record of SLAs of different services and ensures that services performs according to agreed SLAs. Distributed Configuration Module takes care of distributed processing of components on cloud end and at the user end. All configuration items are discovered locally and exposed by service interfaces. This distribution is managed by considering SLAs, sensitivity of data and available computing resources at client end. Distributed Service Management module ensures seamless execution of distributed components, task completion or rollback in case services on client end fail to respond on time, security of the data transmission. Scheduler and Data Transfer Manager schedules tasks on cloud and user end components. It also handles data transformation and transmission between service on cloud and remote services and data security during transformation. • Performance Estimation Service This service calculates cloud resources required for satisfying SLAs. This module is used by Service Initialization and Realization module when services are invoked first time and deployed on cloud resources. This service is also used by Service Monitoring Module during pre-emptive measures for reallocation of cloud resources. • Service Monitoring Module This module monitor services' performance during their execution for their compliance with SLAs. If services are finding it hard to achieve SLAs because of virtual cloud resources, then component interacts with Service Initialization and Realization module to adjust cloud resources to satisfy SLAs.	

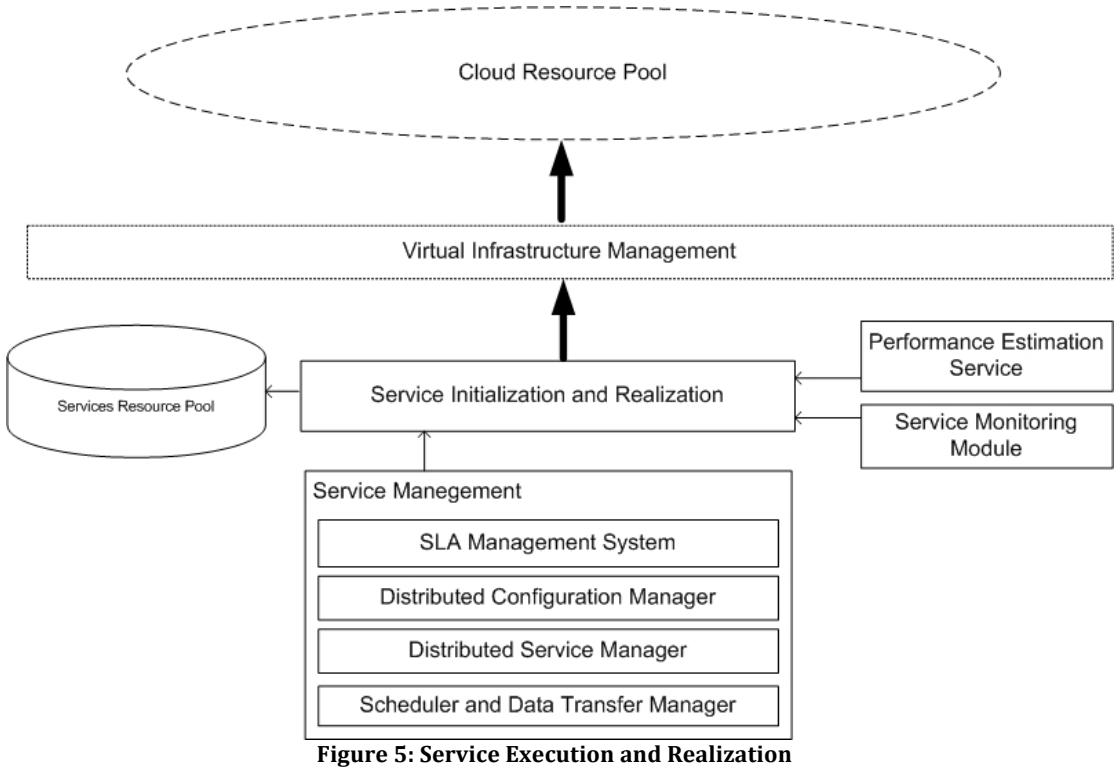

Figure 5: Service Execution and Realization

Table 21: Multi-tenancy Management in Databases

Name: Multi-tenancy Management in Databases	
Problem: For some applications it is critical to have logical isolation of the data to avoid accidental access to it.	
Solution: There are two ways to achieve multi-tenancy at database level. One approach is to have separate database instances for each tenant. This approach is feasible if each of the tenants has large data. For having abstraction of the underlying database selection process, a database driver service is required for routing request to the tenant specific database and for avoiding handling of database selection logic inside each business service. If the tenants do not have huge volumes of data, then having separate instances for each of them may not be a cost effective solution. In this case, database driver service also needs to implement a multi-tenant indexing mechanism for maintaining data of multiple tenants in the same database and allowing access to tenant's own data tuples only.	
Impact: Provides an abstraction to database selection process specific to each tenant.	**Study References:** [30]
Is used for: Multi-tenancy.	
Reference Diagram: Figure 6	

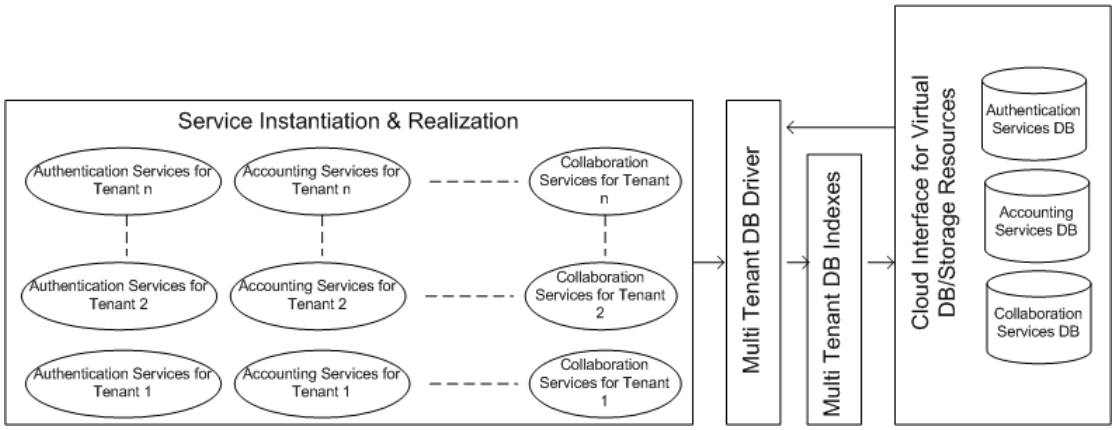

Figure 6: Multi-tenant Management in DB

Table 22: Service Monitoring

Name: Service Monitoring	
Problem: Continuous service monitoring is required when a large number of service providers are serving tenants to make sure that services are being executes according to SLAs and QoS requirements.	
Solution: Context monitoring services on clients' devices can be used to monitor the context of the devices for making adequate cloud side configuration of services. It is also required to monitor cloud services according to QoS requirements of tenants and client devices. QoS monitoring services can be used for monitoring the run time behavior of cloud services to check their compliance with SLA specification.	
Impact: Provides information about health of services.	**Study References:** [16, 18-20, 41]
Is used for: Supporting hybrid devices and achieving quality attributes.	
Reference Diagram: Figure 7 outlines reference architecture and components for the pattern. Following is description of the components shown in the figure. • Service Manager It handles resource provisioning to meet scalability requirements according to SLA agreement. Life	

Cycle Manager reads SLA with help of Manifest Parser and information about SLA is stored in Rule Engine. When there is a request for instantiation of services, these are invoked according to rules stored in Rule Engine and deployed on HTTP Internal Server. Life Cycle Manager also continuously monitors Services and Component Instances to see if they are running according to SLA requirements. In case a change is required in number of instances of services, they are provisioned accordingly. Manifest Parser also monitors SLAs and notifies Life Cycle Manager if there is a change required in Rule Engine as a result of change in SLA.

- Insurance Agent

It provides adaptability and reliability of services by reading SLAs and Service & Components Instances. In case, service are not running according to SLA, it try to reconfigure services to meet SLA and in case SLA cannot be satisfied, services users are compensated by agrees insurance mechanism.

- Service Monitor and Provisioning Sub-System

It monitors existing services as well as acquires or release existing resources depending upon the load on services. Instead of scaling application according to their SLA this service do resource provisioning by analyzing heartbeat of system. Scaling mechanism is dependent upon the scaling algorithm and can be customized by customizing it. In order to have more sophisticated scaling component and make it platform independent, architecture should have flexibility for defining scaling algorithm in some XML based language so that architecture remain independent of the underlying cloud infrastructure being used.

- Adaptable Services

One way to have adaptable services is to continuously monitor services after regular intervals and by analyzing data that is provided by service sensors and probes. If services are predicted to degrade their performance then a new execution plan is prepared and executed. It may involve acquisition of new virtual hardware and software resources from underlying cloud platforms and deployment of services on new resources so that new request can be handled by them.

- Service Reactor (REST)

Every web service has one entry reactor to receive request from external client and give back response. There are two types of Reactors, managed and unmanaged. Managed Reactors are where business logic to manage services is maintained inside reactors where as Unmanaged Reactors are proxies to external third party web services. These reactors handle execution and provisioning of services. These reactors receive client input and dispatch output through service orchestrator in order to provide consolidated view to service clients.

- Controller

For green computing, controller component looks at the services that are active but not serving any client. It can temporarily suspend such services in order to minimize energy foot print.

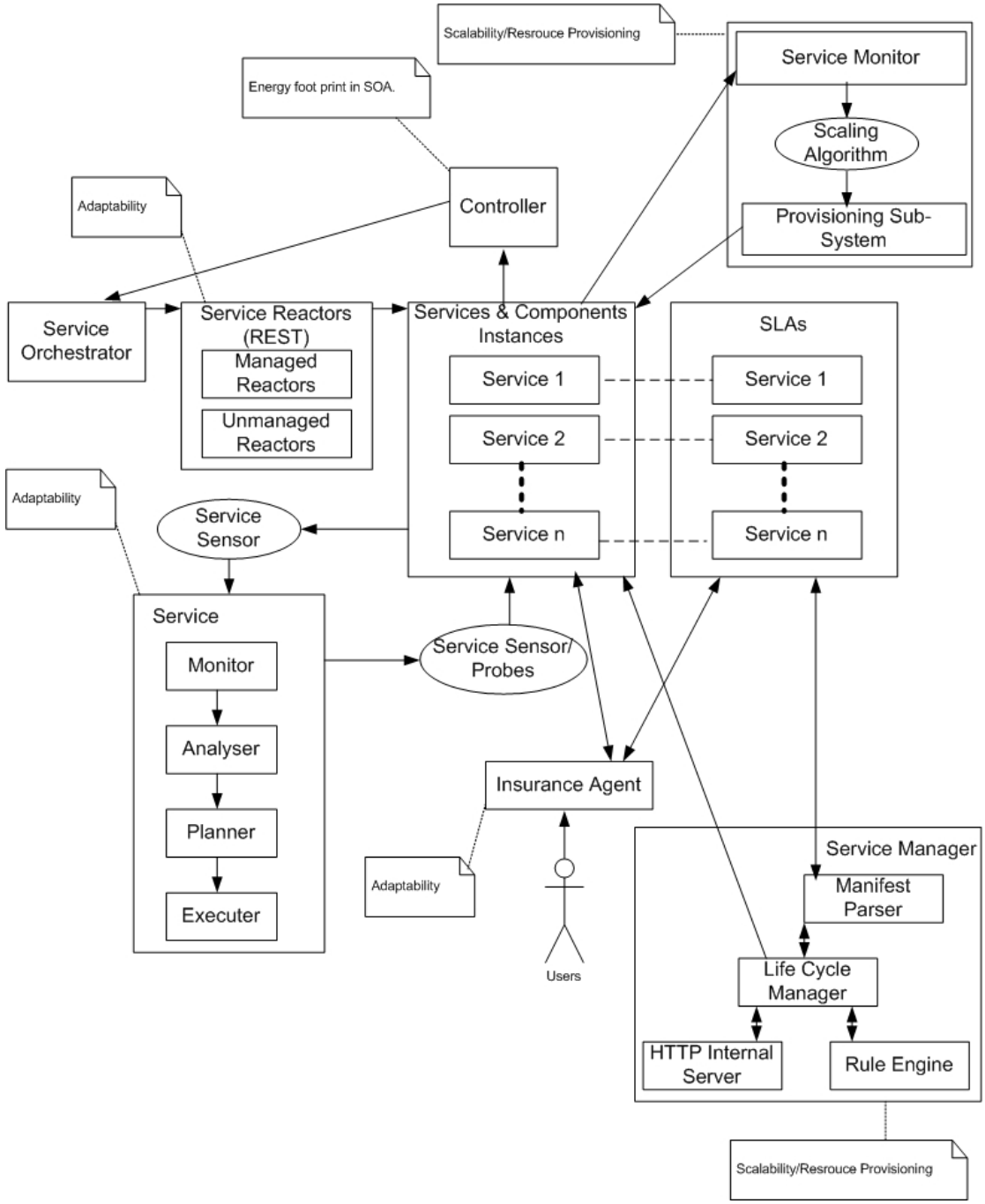

Figure 7: Service Monitoring

Table 23: Service Interoperability

Name: Service Interoperability
Problem: It is hard to have full compatibility between services with different interface styles when these are acquired at run time.
Solution: A solution for this problem is to have an Enterprise Service Bus (ESB) supporting inter operability between services with different styles of interfaces. In some cases it can be hard to have a fully compatible ESB. An extension to this approach is to provide hooks for registering interface compatibility services. Services that do not comply with the interfaces supported by ESB can register their compatibility services with the ESB in Adaptive Service Registry. ESB can use these for interface conversions.

Impact: Support compliance with services having non-standardized interfaces.	**Study References:** [16, 40]
Is used for: Supporting hybrid devices and achieving quality attributes.	
Reference Diagram: Figure 8	

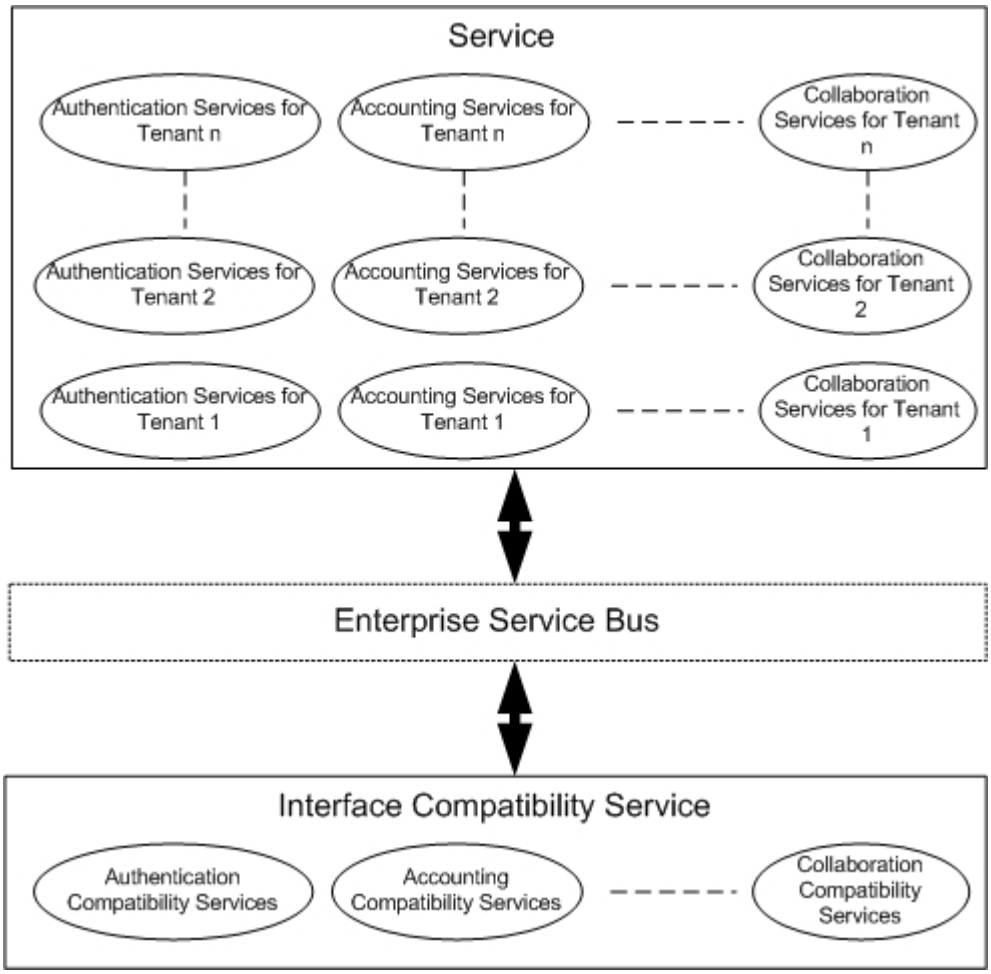

Figure 8: Service Interoperability

Table 24: Distributed Workflow Execution Management

Name: Distributed Workflow Execution Management	
Problem: Workflow processing of sensitive data on cloud is tricky because sensitive data cannot be processed on non-sensitive nodes.	
Solution: Using data collection services that can be deployed on data storage locations can facilitate workflow processing on sensitive data. These services can perform operations on sensitive data and send the higher abstraction of information to non-secure public cloud nodes. Components or services for further analysis, reduction and re-projection can be deployed on public cloud nodes. Services for workflow enactment and monitoring are required for configuration and keeping workflow according to SLA specification and QoS requirements.	
Impact: Allows workflow-based systems to be deployed on cloud having capability of processing sensitive data.	**Study References:** [20, 36, 44]

Is used for:
Middleware infrastructure, workflow processing and achieving quality attributes.

Reference Diagram:

Figure 9 outlines reference architecture and components for the pattern. Following is description of the components shown in the figure.

- Workflow Tools, Portals and Desktop Application

It represents components of systems that are used by system actors as an interface to system. This contains different types of modeling tools, web and desktop based applications interfaces.

- Programing and scripting language APIs

This provides the simplified APIs of system web services that can be used by programming and scripting languages to define system interfaces. It provides higher level of abstraction to detailed RESTful system APIs

- RESTful API layer

This layer of components provides RESTful APIs to internal system resources. It provides APIs for

  - Middleware and Workflow Utilities
    These APIs can be used to define workflows into the system.
  - Database Adapters
    It provides APIs to persistence layer and is used by external clients as well as internal system components to handle persistence.
  - Data and Resource Directories
    It manages data storage and other system resources used by system. For example, document management repositories used by workflow systems.

- Data collection components

These are external system components installed on external applications and act as data input feeds.

- Semantic Access Interface Layer

It validates workflows that are defined according semantic rules before activating and making them persistent.

- Core services

It contains services that are required for processing of a particular workflow. It contains Activity Manager used to handle activities of user on the workflow, Role Manager that handles access privileges to data in work flow according to user rights and data processing stage in workflow, Analysis and Reproduction Components that are responsible for deciding which system components are to be initialized on cloud side and which system components are to be initialized on client side depending on requirements of sensitivity of data and user requirements. Core services interact with Cloud-Side Engine and User-End Engine to have optimal set of service side and client side components for processing of a particular workflow.

- Cloud-Side Engine and User-End Engine

These components evaluate a workflow requirement of components and determine an optimal set of combination of cloud side and client side components depending on data sensitivity requirements. This optimal set is used to instantiate a workflow.

- Local service repository

It is repository of components stored on the user end. User-End engine fetches components from this repository when it has to instantiate it as part of distributed workflow on cloud and user end.

- Virtual Infrastructure Management

This layer handles management cloud virtual resources required to run cloud side part of workflow components.

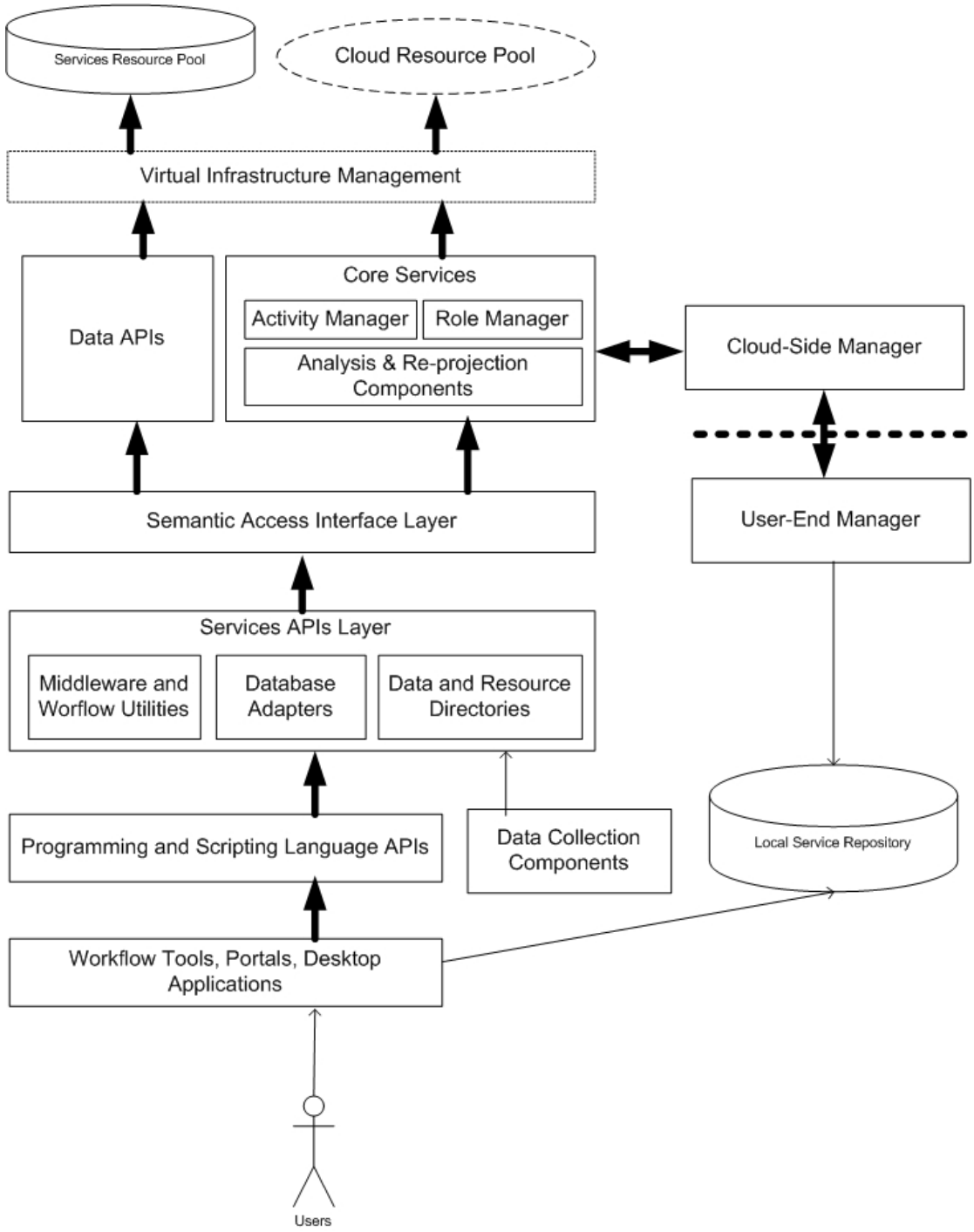

Figure 9: Distributed Workflow Execution Management

Table 25 shows mapping between architecture styles described above and purpose for which these architecture styles can be used.

Table 25: Mapping between Architecture Style and Purpose

Architecture Style Name	Purpose
Service Management	• Services for devices with limited processing and storage resources. • Location aware services. • Components distribution schemes. • Infrastructure management. • Service versioning.

Service Request Handling	• Isolation between services. • Isolation between data storage units. • Location aware services. • Components distribution schemes.
Services Execution and Realization	• Automated, dynamic and reliable runtime configuration and composition of services. • Components/services distribution schemes. • Infrastructure management. • Processing sensitive and computing intensive data. • Energy optimization and efficiency. • Adaptability. • Reliability. • Resource scalability and provisioning.
Multi-tenancy in Databases	• Isolation between data stored in databases.
Service Monitoring	• Service level agreement compliance. • Adaptability • Reliability • Components/services distribution schemes.
Services Interoperability	• Services integration with other services.
Distributed workflow execution	• Access to remote users and semi-trusted external services. • Intra operability issues between clouds. • Security of processing nodes and data in workflow. • Adaptability. • Services integration with other services.

3.5. Tools and Technologies used in Solutions

Table 26 lists frameworks and platforms being used in studies corresponding to main categories of themes. REST is most commonly used architecture style being used. Frameworks based on Java programming language are chosen by many researchers because of their platform neutral nature and strong support for server side development. Many of the studies classified into main categories are not explicitly specifying cloud platforms being used. A possible reason for it can be, they tend to provide generic solutions and tend to be platform neutral.

Table 26: Categories versus Technologies and Platforms being used

Category	Technologies Being Used
Multi-tenancy	WSO2 Carbon Platform [28]. JDBC [30].
Support for Hybrid Devices	eXtensible Messaging and Presense Protocol (XMPP), Amazon Simple Queue Service (SQS), OPEN ID, OPEN AUTH, User Request Token [38]. Nix, Disnix [39].
Middleware Infrastructure	REST [45] IBMAltocumulus [47]. YML [48]. Nix package manager [39]. Disnix [39].
Dataflow Security and Workflow Processing	JVM, Perl, MeDICi, Restlet [34]. Windows Azure Platform [36].
Quality Attributes	FraSCAti, WildCAT, Choco [14]. XSLT and XPath [16]. Apache HTTP load balancer, IBM Tivoli Provisioning Manager [19]. Open source JOnAS JavaEE application server [20].

Frameworks	Aneka development platform, InterGrid project, VMWare, Xen, KVM, OpenPEX [4]. KVM, Eucalyptus, Apache Hadoop MapReducer, memcached, Apache HBase, PostgreSQL/MySQL, Apache Web Server/Tomcat [31].

The use of ***WSO2*** carbon platform is described in [28] for implementing multi-tenancy features in cloud middleware. This is a componentized framework used to build servers. It consists of components that can fit into servers' runtime environment and offer web service APIs for service management. It supports launching of bundled components on tenant specific instances. The framework also supports components to manage themselves through an administration console and interact with other services being hosted by platform. It supports middleware to host self monitored services. Some of the other prominent features of the framework for facilitating services discovery and management are: authentication and authorization of services, offering of shared storage repositories for storing server states, shared registry to store and share system governance information and support of writing UI components for manual administration of services. Multi-tenancy in databases is proposed in [30] using ***JavaEE*** and ***JDBC*** technologies. It is achieved by allowing tenants to share tables in database with the help of ***JDBC Driver Manager*** and ***Multi-Tenant DB Index***. JDBC Driver Manager is implemented on top of Java JDBC driver to identify rows in tables that belong to tenant. A hash map with tenant identity as key and connection parameters of JDBC driver as value is used to identify which JDBC driver is to be used. This technique provides abstraction to the low level multi-tenant implementation and allows application to access database by using traditional programming styles.

The technologies for building RESTful applications for mobile devices are presented in [38]. ***Amazon Simple Queue Service (SQS)*** and ***eXtensible Messaging and Presence Protocol (XMPP)*** can be used for transaction management and advanced orchestration. XMPP can be used in transaction process monitoring for handling remote transactions and log managers. ***Multiparty dynamic signatures and keys*** can be used for building group auth methods for authentication of social information in the cloud. Web resource and service providers can be verified by using ***Open ID*** and ***Open Auth*** without compromising authentication credentials of the users. ***User Request Tokens*** is a simple authentication mechanism and send as a part of URI request. This mechanism is easier to be utilized in mobile devises because it does not involve any header manipulation and consequently requires less processing.

A package management tool for deployment of software packages using declarative specification is presented in [39] named ***Nix*** and its distributed version named ***Disnix***. It facilitates automation of the deployment of software packages on cloud and helps dynamic configurations of cloud side components and services in order to serve multiple types of devices.

The advantages of using REST based approach for mitigating difficulty of sharing information is presented in [45]. REST based infrastructure is easy to distribute because it is stateless and each resource in REST can be represented by a URI. Hence, a REST based middleware for distributed management of RESTful Web services can be effective for managing services on cloud. ***IBM Altocumulus*** middleware framework is presented in [47] for enabling inter operability across computing clouds. It is useful when organizations are using combination of private and public clouds to satisfy needs of applications. A middleware named ***YML*** is presented in [48] that can be used for managing dedicated as well as non dedicated computing resources,

hosting multiple types of operating systems; for writing customized algorithms, utilizing other middlewares by specifying interfaces for each of them and supporting user interaction with services through YML frontend. ***Nix Package Manager*** [39] supports building packages from resource using a functional language. This supports multiple versions of the packages in system with atomic upgrade and roll back features. ***Disnix*** [39] supports distributed deployments of packages using model based descriptions.

MeDICi framework [34] is used for the processing of data on distributed nodded. It facilitates creation of a pipeline based data processing system. It is used for performing analysis on huge volumes of data by deploying nodes of the processing framework o physical or virtual nodes. The nodes of the pipeline can be connected to each other by REST based interfaces. Use of the ***Windows Azure Platform*** is presented in [36] for processing satellite data. It is claimed by the authors that use of Azure virtual machine instances produces results in 90% less time. Azure virtual machine instances can be used by other data intensive applications to produce results efficiently.

FraSCAti [14] is used for runtime management and monitoring of components' or services' properties dealing with their association with other services. It can also be used for activating and deactivating components and services. ***WildCAT*** [14] is used to organize and ease the access of sensors. It is suitable for context aware applications that use to fetch information from sensors frequently. ***Choco*** [14], a Java based library, is used to define constraints as a model of problem variables and associated domain functions. It can be used in solutions for optimization problems on cloud. The use of ***XSLT*** and ***XPath*** to define SLA mappings is described in [16]. This technique can be used for defining service level agreement mappings between service providers and service consumers. ***IBM Tivoli Provisioning Manager (TPM)*** along with ***Apache HTTP load balancer*** is used to automate manual tasks of configuring and provisioning cloud resources [19]. The set of framework can be used for server, operation systems, middleware and network resource configurations. ***JOnAS JavaEE application server*** is used for monitoring core of middleware and services using parameters received from data collection units [20].

A set of tool is presented in [4] for building applications and managing resources in market-oriented cloud computing environment. The tools presented include ***Aneka development platform*** a PaaS SDK, ***InterGrid*** for federated integration of heterogeneous cloud computing environments, ***VMWare*** for managing virtual machines monitor, ***Xen*** for managing commodity hardware, ***KVM*** a kernel based virtual machine and ***OpenPEX*** a reservation based hardware resource management. Application development framework presented in [31] is using ***KVM*** as virtual machine monitor, ***Eucalyptus*** as a cloud controller, ***Apache Hadoop MapReducer*** as a distributed processing framework, memcached for cashing data, ***Apache HBase*** for large datastores, ***PostgreSQL/MySQL*** as relational database management systems and ***Apache Web Server/Tomcat*** as web and application servers.

4. Quality of Research and Validity Threats

The quality of research was ensured by defining the research protocol and conducting research following the guidelines of systematic review/mapping study [13]. Research protocol helped us to overcome bias in the study selection process. Research protocol contained research questions, inclusion and exclusion criteria, and research strategy. The research protocol was developed by first author and was reviewed by second

author. It was ensured in the review process that search strings were appropriately derived from research questions. Moreover, to ensue unbiased study selection, study selection was done in multiple stages. References of Included and excluded studies in each selection stage were maintained in EndNote libraries and in excel spreadsheets. Study selection process was verified by second author. In case where there was ambiguity about selection of study, full text screening was performed to decide about its inclusion or exclusion. Data extraction form was defined to perform consistent extraction of information to address the research questions.

Accuracy and consistency during the review process was highly based on a common understanding among the reviewers. Misunderstandings can result in biased effects. One of the main limitations of the review could be the possibility of bias in the selection of studies. To help ensure that the selection process was as much unbiased as possible, we developed detailed guidelines in the *review protocol* prior to the start of the review. During the papers' screening phase, we documented the reasons for its inclusion/exclusion. Finally, we rechecked the papers again based on the inclusion/exclusion criteria.

Our research process addressed threats to conclusion, internal and construct validity. We used data extraction form to address threats to ***conclusion validity*** [99]. The use of data extraction form reduced bias in data extraction process and ensures data extraction that is consistent and relevant to research questions. Biased in selection process could be a threat to ***internal validity*** [99]. This was addressed by our multi-stage search and study selection process as described in section 2.3. Initial set of included studies was selected by first author. References for included and excluded studies were maintained in Endnote library for traceability of included and excluded studies. Initial set of selected studies was reviewed by secondary author and disputed set of studies were included or excluded after discussion between first and second author. Validity of data selection and its representation to address research questions is referred as ***Construct Validity*** [99]. Searches were performed on multiple databases to get relevant journals and proceedings. Research protocol was developed in order to minimize threat to construct validity. Research question, inclusion and exclusion criteria, search strings used for searches and data extraction strategy ensured consistent data extraction process and valid results.

5. Conclusions

Cloud computing is being adopted by organizations to minimize their infrastructure cost and supporting large number of customers by utilizing scalability features of the paradigm. However, adopting this technology is not straightforward and requires special considerations during architecting application. The main motivation for the work reported in this paper was to investigate the state of the art software architectures for cloud based systems and systematically map the literature in order to determine challenges associated with it and how these issues are addressed by researchers.

The papers selected for this study address cloud computing challenges from various dimensions. The reported challenges are associated with maintaining underlying infrastructure resource to support large number of end users, allowing cloud based application to be accessed from multiple types of devices, processing of sensitive data on cloud which is stored on organizations' local platforms, satisfying service level agreements and applications quality of service requirements. After searching target

electronic databases and rigorous selection process, we selected 86 primary studies. The selected studies address challenges at varies level of application architectures. A significant number of software and tools using various technologies are also reported demonstrating implementation of the solutions. We have grouped studies into different categories of themes with respect to the main challenges/issues addressed in the papers. For each category of theme, we have described problems addressed by papers in that category, architecture solutions of the respective problems, and software, tools and technologies used to implement the architecture solutions. Based on the data, which was extracted from different categories of themes; we also have proposed common architecture styles. The identified architecture styles can be used in solving software architecture related problems on cloud.

This systematic review provides guidelines for both practitioners and researchers. Practitioners can use this review as a source for searching relevant architecture solutions for achieving cloud specific quality characteristics in application design. The problems described in each category of theme can provide foundation for identifying non/extra functional requirements and quality attributes of a specific cloud based application. The synthesized architecture styles can be used for building high-level architecture of the system. The set of identified technologies can be used as a quick reference to pick suitable frameworks for a desired application. This study provides a map of the software architecture research for cloud and provides a starting point to the researchers to find relevant literature, to have a brief overview of the research and unsolved issued which needs attention of the researchers. In future, we expect more research in this area for making the proposed architecture styles a part of the concrete architecture principles that cloud based applications and/or hosting platforms can follow.

Cross platform adaptability of services to support migrating them from one cloud to another, migrating live services between heterogeneous clouds to satisfy QoS requirements, investigation of accountability methods for transparent identification of failed/malfunctioning services when they are working in an aggregated fashion and visibility of scalability to application hosted on PaaS for addressing data sensitivity issues are some of the open research areas.

The tree in Figure 10 shows the summary of the results presented in section 3. Interpretation of tree nodes is presented in legend on top right of Figure 10. The tree shows the main categories and sub categories of study classification along with number of papers in each category. The problems that are being addressed in main categories along with tools and technologies used are also elaborated.

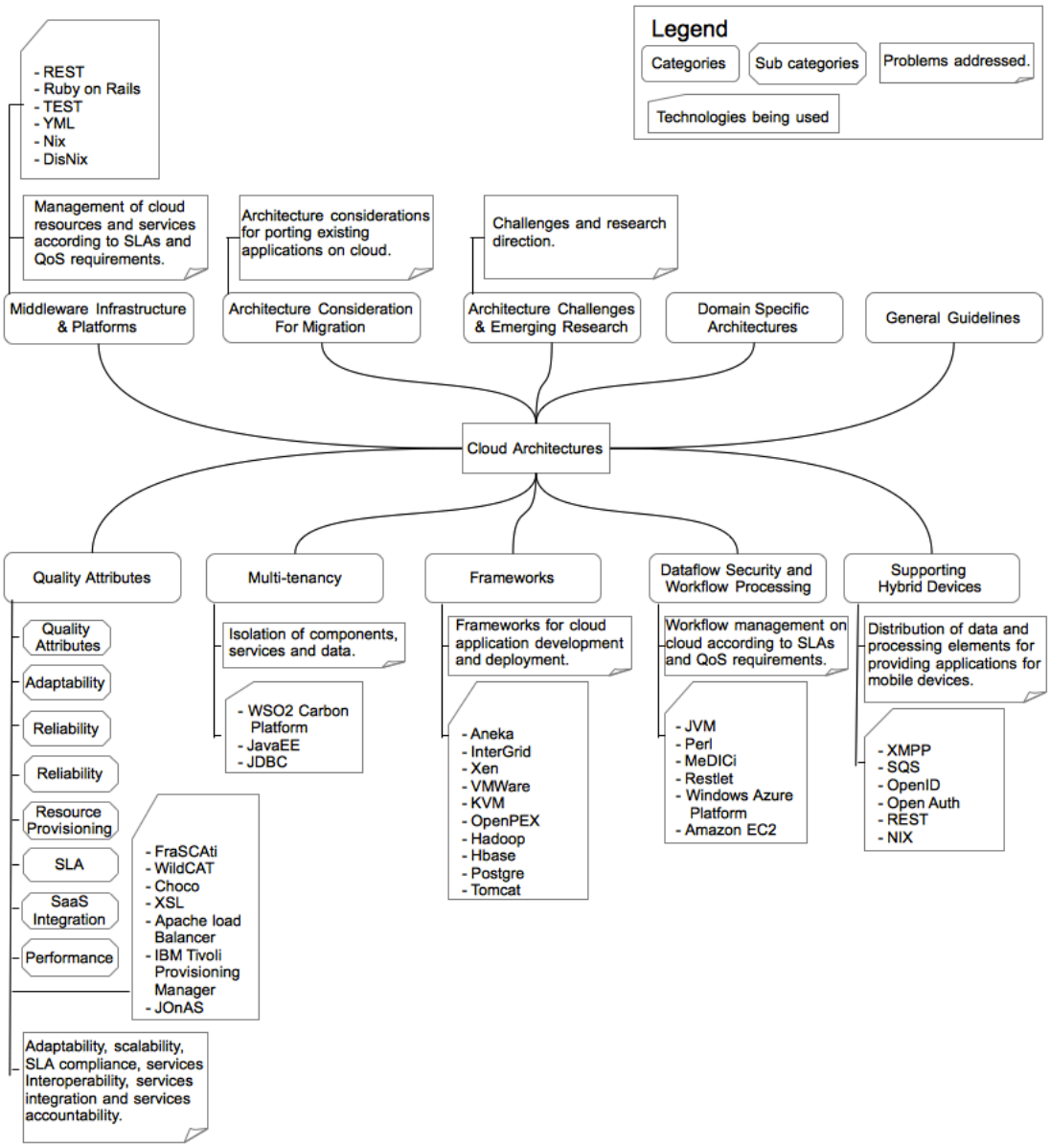

Figure 10: Summary of the Results

References

[1] M. Armbrust*, et al.*, "A view of cloud computing," *Commun. ACM,* vol. 53, pp. 50-58, 2010.

[2] A. Lenk*, et al.*, "What's inside the Cloud? An architectural map of the Cloud landscape," in *Software Engineering Challenges of Cloud Computing, 2009. CLOUD '09. ICSE Workshop on*, 2009, pp. 23-31.

[3] P. Louridas, "Up in the Air: Moving Your Applications to the Cloud," *Software, IEEE,* vol. 27, pp. 6-11, 2010.

[4] R. Buyya*, et al.*, "Cloudbus Toolkit for Market-Oriented Cloud Computing," in *Cloud Computing*. vol. 5931, M. Jaatun*, et al.*, Eds., ed: Springer Berlin / Heidelberg, 2009, pp. 24-44.

[5] P. Lago and T. Jansen, "Creating Environmental Awareness in Service Oriented Software Engineering

Service-Oriented Computing." vol. 6568, E. Maximilien*, et al.*, Eds., ed: Springer Berlin / Heidelberg, 2011, pp. 181-186.

[6] L. M. Vaquero*, et al.*, "A break in the clouds: towards a cloud definition," *SIGCOMM Comput. Commun. Rev.,* vol. 39, pp. 50-55, 2008.

[7] P. Mell and T. Grance, "The NIST Definition of Cloud Computing," *NIST,* vol. Special Publication 800-145, 2011.

[8] M. A. Babar and M. A. Chauhan, "A tale of migration to cloud computing for sharing experiences and observations," presented at the Proceedings of the 2nd International Workshop on Software Engineering for Cloud Computing, Waikiki, Honolulu, HI, USA, 2011.

[9] F. Baiardi and D. Sgandurra, "Securing a Community Cloud," in *Distributed Computing Systems Workshops (ICDCSW), 2010 IEEE 30th International Conference on*, 2010, pp. 32-41.

[10] A. Khajeh-Hosseini*, et al.*, "Cloud Migration: A Case Study of Migrating an Enterprise IT System to IaaS," in *Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on*, 2010, pp. 450-457.

[11] Q. Zhang*, et al.*, "Cloud computing: state-of-the-art and research challenges," *Journal of Internet Services and Applications,* vol. 1, pp. 7-18, 2010.

[12] M. A. Chauhan and M. A. Babar, "Migrating Service-Oriented System to Cloud Computing: An Experience Report," in *Cloud Computing (CLOUD), 2011 IEEE International Conference on*, 2011, pp. 404-411.

[13] B. Kitchenham and S. Charters, "Guidelines for performing Systematic Literature Reviews in Software Engineering," in *Version* vol. 2, ed, 2007, pp. 2007-01.

[14] J. Alvares de Oliveira and T. Ledoux, "Self-optimisation of the energy footprint in service-oriented architectures," presented at the Proceedings of the 1st Workshop on Green Computing, Bangalore, India, 2010.

[15] D. Bonetta and C. Pautasso, "Towards liquid service oriented architectures," presented at the Proceedings of the 20th international conference companion on World wide web, Hyderabad, India, 2011.

[16] I. Brandic*, et al.*, "Service mediation and negotiation bootstrapping as first achievements towards self-adaptable grid and cloud services," presented at the Proceedings of the 6th international conference industry session on Grids meets autonomic computing, Barcelona, Spain, 2009.

[17] Z. Changli and Y. Maode, "Insurance-Based Cloud Computing-Architecture, Risk Analysis and Experiment," in *2010 International Conference on Computational Intelligence and Software Engineering (CiSE)*, ed: IEEE, 2010, pp. 1-4 %@ 978-1-4244-5391-7.

[18] C. Chapman*, et al.*, "Software architecture definition for on-demand cloud provisioning," *Cluster Computing,* pp. 1-22, 2011.

[19] T. C. Chieu*, et al.*, "Dynamic Scaling of Web Applications in a Virtualized Cloud Computing Environment," in *IEEE International Conference on e-Business Engineering, 2009. ICEBE '09*, ed: IEEE, 2009, pp. 281-286 %@ 978-0-7695-3842-6.

[20] A. Chazalet, "Service Level Agreements Compliance Checking in the Cloud Computing: Architectural Pattern, Prototype, and Validation," in *2010 Fifth International Conference on Software Engineering Advances (ICSEA)*, ed: IEEE, 2010, pp. 184-189 %@ 978-1-4244-7788-3.

[21] J. Zou, *et al.*, "From Representational State Transfer to Accountable State Transfer Architecture," in *2010 IEEE International Conference on Web Services (ICWS)*, ed: IEEE, 2010, pp. 299-306 %@ 978-1-4244-8146-0.
[22] L. Feng, *et al.*, "SaaS Integration for Software Cloud," in *2010 IEEE 3rd International Conference on Cloud Computing (CLOUD)*, ed: IEEE, 2010, pp. 402-409 %@ 978-1-4244-8207-8.
[23] S. Basu, *et al.*, "Control plane integration for cloud services," presented at the Proceedings of the 11th International Middleware Conference Industrial track, Bangalore, India, 2010.
[24] F. o. Baude, *et al.*, "ESB federation for large-scale SOA," presented at the Proceedings of the 2010 ACM Symposium on Applied Computing, Sierre, Switzerland, 2010.
[25] M. Hassan, *et al.*, "A market-oriented dynamic collaborative cloud services platform," *Annals of Telecommunications,* vol. 65, pp. 669-688, 2010.
[26] M. Hiltunen and R. Schlichting, "Is Collaborative QoS the Solution to the SOA Dependability Dilemma?," in *Architecting Dependable Systems VII*. vol. 6420, A. Casimiro, *et al.*, Eds., ed: Springer Berlin / Heidelberg, 2010, pp. 227-248.
[27] H. Zhengxiong, *et al.*, "ASAAS: Application Software as a Service for High Performance Cloud Computing," in *2010 12th IEEE International Conference on High Performance Computing and Communications (HPCC)*, ed: IEEE, 2010, pp. 156-163 %@ 978-1-4244-8335-8.
[28] A. Azeez, *et al.*, "Multi-tenant SOA Middleware for Cloud Computing," in *2010 IEEE 3rd International Conference on Cloud Computing (CLOUD)*, ed: IEEE, 2010, pp. 458-465 %@ 978-1-4244-8207-8.
[29] Z. Pervez, *et al.*, "Multi-Tenant, Secure, Load Disseminated SaaS Architecture," in *2010 The 12th International Conference on Advanced Communication Technology (ICACT)* vol. 1, ed: IEEE, 2010, pp. 214-219 %@ 978-1-4244-5427-3.
[30] E. J. Domingo, *et al.*, "CLOUDIO: A Cloud Computing-Oriented Multi-tenant Architecture for Business Information Systems," in *2010 IEEE 3rd International Conference on Cloud Computing (CLOUD)*, ed: IEEE, 2010, pp. 532-533 %@ 978-1-4244-8207-8.
[31] S. Hosono, *et al.*, "A Lifetime Supporting Framework for Cloud Applications," in *2010 IEEE 3rd International Conference on Cloud Computing (CLOUD)*, ed: IEEE, 2010, pp. 362-369 %@ 978-1-4244-8207-8.
[32] G. Zhiyun, *et al.*, "A framework of enterprise cloud application," in *2010 IEEE 2nd Symposium on Web Society (SWS)*, ed: IEEE, 2010, pp. 729-732 %@ 978-1-4244-6356-5.
[33] J. Du, *et al.*, "Towards secure dataflow processing in open distributed systems," presented at the Proceedings of the 2009 ACM workshop on Scalable trusted computing, Chicago, Illinois, USA, 2009.
[34] I. Gorton, *et al.*, "Exploring Architecture Options for a Federated, Cloud-Based System Biology Knowledgebase," in *2010 IEEE Second International Conference on Cloud Computing Technology and Science (CloudCom)*, ed: IEEE, 2010, pp. 218-225 %@ 978-1-4244-9405-7.
[35] Y.-B. Han, *et al.*, "A Cloud-Based BPM Architecture with User-End Distribution of Non-Compute-Intensive Activities and Sensitive Data," *Journal of Computer Science and Technology,* vol. 25, pp. 1157-1167, 2010.
[36] L. Jie, *et al.*, "eScience in the cloud: A MODIS satellite data reprojection and reduction pipeline in the Windows Azure platform," in *2010 IEEE International Symposium on Parallel & Distributed Processing (IPDPS)*, ed: IEEE, 2010, pp. 1-10 %@ 978-1-4244-6442-5.
[37] K. Kim, "A model-driven workflow fragmentation framework for collaborative workflow architectures and systems," *Journal of Network and Computer Applications,* vol. In Press, Corrected Proof %U http://www.sciencedirect.com/science/article/pii/S108480451100083X.
[38] J. H. Christensen, "Using RESTful web-services and cloud computing to create next generation mobile applications," presented at the Proceeding of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications, Orlando, Florida, USA, 2009.

[39] S. van der Burg*, et al.*, "Software deployment in a dynamic cloud: From device to service orientation in a hospital environment," in *ICSE Workshop on Software Engineering Challenges of Cloud Computing, 2009. CLOUD '09*, ed: IEEE, 2009, pp. 61-66 %@ 978-1-4244-3713-9.

[40] W. Qian and R. Deters, "SOA's Last Mile-Connecting Smartphones to the Service Cloud," in *IEEE International Conference on Cloud Computing, 2009. CLOUD '09*, ed: IEEE, 2009, pp. 80-87 %@ 978-1-4244-5199-9.

[41] P. Papakos*, et al.*, "VOLARE: context-aware adaptive cloud service discovery for mobile systems," presented at the Proceedings of the 9th International Workshop on Adaptive and Reflective Middleware, Bangalore, India, 2010.

[42] I. Giurgiu*, et al.*, "Calling the Cloud: Enabling Mobile Phones as Interfaces to Cloud Applications," in *Middleware 2009*. vol. 5896, J. Bacon and B. Cooper, Eds., ed: Springer Berlin / Heidelberg, 2009, pp. 83-102.

[43] L. Qingfeng*, et al.*, "An Optimized Solution for Mobile Environment Using Mobile Cloud Computing," in *5th International Conference on Wireless Communications, Networking and Mobile Computing, 2009. WiCom '09*, ed: IEEE, 2009, pp. 1-5 %@ 978-1-4244-3692-7.

[44] S. V. Gogouvitis*, et al.*, "An Architectural Approach for Event-Based Execution Management in Service Oriented Infrastructures," in *2010 12th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)*, ed: IEEE, 2010, pp. 395-399 %@ 978-1-4244-9816-1.

[45] H. Hyuck*, et al.*, "A RESTful Approach to the Management of Cloud Infrastructure," in *IEEE International Conference on Cloud Computing, 2009. CLOUD '09*, ed: IEEE, 2009, pp. 139-142 %@ 978-1-4244-5199-9.

[46] H. Ludwig*, et al.*, "REST-based management of loosely coupled services," presented at the Proceedings of the 18th international conference on World wide web, Madrid, Spain, 2009.

[47] E. M. Maximilien*, et al.*, "IBM altocumulus: a cross-cloud middleware and platform," presented at the Proceeding of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications, Orlando, Florida, USA, 2009.

[48] L. Shang*, et al.*, "Extending YML to Be a Middleware for Scientific Cloud Computing," in *Cloud Computing*. vol. 5931, M. Jaatun*, et al.*, Eds., ed: Springer Berlin / Heidelberg, 2009, pp. 662-667.

[49] Y. Badr and G. Caplat, "Software-as-a-Service and Versionology: Towards Innovative Service Differentiation," in *2010 24th IEEE International Conference on Advanced Information Networking and Applications (AINA)*, ed: IEEE, 2010, pp. 237-243 %@ 978-1-4244-6695-5.

[50] P. Belimpasakis and S. Moloney, "A platform for proving family oriented RESTful services hosted at home," *IEEE Transactions on Consumer Electronics,* vol. 55, pp. 690-698, 2009.

[51] M. A. Babar and M. A. Chauhan, "A tale of migration to cloud computing for sharing experiences and observations," presented at the Proceeding of the 2nd international workshop on Software engineering for cloud computing, Waikiki, Honolulu, HI, USA, 2011.

[52] M. Hagiwara, "Development Procedure of the Cloud-Based Applications," in *Database Systems for Advanced Applications*. vol. 5982, H. Kitagawa*, et al.*, Eds., ed: Springer Berlin / Heidelberg, 2010, pp. 320-326.

[53] A. J. Elmore*, et al.*, "Zephyr: live migration in shared nothing databases for elastic cloud platforms," presented at the Proceedings of the 2011 international conference on Management of data, Athens, Greece, 2011.

[54] Z. Liang-Jie and Z. Qun, "CCOA: Cloud Computing Open Architecture," in *IEEE International Conference on Web Services, 2009. ICWS 2009*, ed: IEEE, 2009, pp. 607-616 %@ 978-0-7695-3709-2.

[55] B. Sodhi and T. V. Prabhakar, "Application architecture considerations for cloud platforms," in *2011 Third International Conference on Communication Systems and Networks (COMSNETS)*, ed: IEEE, 2011, pp. 1-4 %@ 978-1-4244-8952-7.

[56] D. Kossmann*, et al.*, "An evaluation of alternative architectures for transaction processing in the cloud," presented at the Proceedings of the 2010 international conference on Management of data, Indianapolis, Indiana, USA, 2010.

[57] Z. Wenjun, "2-Tier Cloud Architecture with maximized RIA and SimpleDB via minimized REST," in *2010 2nd International Conference on Computer Engineering and Technology (ICCET)* vol. 6, ed: IEEE, 2010, pp. V6-52-V6-56 %@ 978-1-4244-6347-3.

[58] J. Schaffner*, et al.*, "Towards enterprise software as a service in the cloud," in *2010 IEEE 26th International Conference on Data Engineering Workshops (ICDEW)*, ed: IEEE, 2010, pp. 52-59 %@ 978-1-4244-6522-4.

[59] T. Wei-Tek*, et al.*, "Real-Time Service-Oriented Cloud Computing," in *2010 6th World Congress on Services (SERVICES-1)*, ed: IEEE, 2010, pp. 473-478 %@ 978-1-4244-8199-6.

[60] X. Jia, "Google Cloud Computing Platform Technology Architecture and the Impact of Its Cost," in *2010 Second World Congress on Software Engineering (WCSE)* vol. 2, ed: IEEE, 2010, pp. 17-20 %@ 978-1-4244-9287-9.

[61] S. Wei*, et al.*, "Design Aspects of Software as a Service to Enable E-Business through Cloud Platform," in *2010 IEEE 7th International Conference on e-Business Engineering (ICEBE)*, ed: IEEE, 2010, pp. 456-461 %@ 978-1-4244-8386-0.

[62] R. Clarke, "User Requirements for Cloud Computing Architecture," in *2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing (CCGrid)*, ed: IEEE, 2010, pp. 625-630 %@ 978-1-4244-6987-1.

[63] L. Hyun Jung*, et al.*, "Technical Challenges and Solution Space for Developing SaaS and Mash-Up Cloud Services," in *IEEE International Conference on e-Business Engineering, 2009. ICEBE '09*, ed: IEEE, 2009, pp. 359-364 %@ 978-0-7695-3842-6.

[64] B. Liver*, et al.*, "Privacy in Service Oriented Architectures: SOA Boundary Identity Masking for Enterprises," in *2010 IEEE 12th Conference on Commerce and Enterprise Computing (CEC)*, ed: IEEE, 2010, pp. 204-211 %@ 978-1-4244-8433-1.

[65] D.-Y. Cheng*, et al.*, "A user centric service-oriented modeling approach," *World Wide Web,* vol. 14, pp. 431-459, 2011.

[66] G. Zhou and G. He, "One Program Model for Cloud Computing," in *Cloud Computing*. vol. 5931, M. Jaatun*, et al.*, Eds., ed: Springer Berlin / Heidelberg, 2009, pp. 589-594.

[67] M. Azambuja*, et al.*, "An Architecture for Public and Open Submission Systems in the Cloud," in *2010 IEEE 3rd International Conference on Cloud Computing (CLOUD)*, ed: IEEE, 2010, pp. 513-517 %@ 978-1-4244-8207-8.

[68] R. Pereira*, et al.*, "An Architecture for Distributed High Performance Video Processing in the Cloud," in *2010 IEEE 3rd International Conference on Cloud Computing (CLOUD)*, ed: IEEE, 2010, pp. 482-489 %@ 978-1-4244-8207-8.

[69] R. Pereira and K. Breitman, "A Cloud Based Architecture for Improving Video Compression Time Efficiency: The Split & Merge Approach," in *Data Compression Conference (DCC), 2011*, ed: IEEE, 2011, pp. 471-471 %@ 978-1-61284-279-0.

[70] P. Rodriguez*, et al.*, "VaaS: Videoconference as a service," in *5th International Conference on Collaborative Computing: Networking, Applications and Worksharing, 2009. CollaborateCom 2009*, ed: IEEE, 2009, pp. 1-11 %@ 978-963-9799-76-9.

[71] N. Botts*, et al.*, "Cloud Computing Architectures for the Underserved: Public Health Cyberinfrastructures through a Network of HealthATMs," in *2010 43rd Hawaii International Conference on System Sciences (HICSS)*, ed: IEEE, 2010, pp. 1-10 %@ 978-1-4244-5509-6.

[72] S. Chia-Ping*, et al.*, "Bio-signal analysis system design with support vector machines based on cloud computing service architecture," in *2010 Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, ed: IEEE, 2010, pp. 1421-1424 %@ 978-1-4244-4123-5.

[73] V. V. Brizgalov*, et al.*, "Architecture of traffic control systems using cloud computing," in *2010 International Conference and Seminar on Micro/Nanotechnologies and Electron Devices (EDM)*, ed: IEEE, 2010, pp. 215-216 %@ 978-1-4244-6626-9.

[74] R. A. Calvo*, et al.*, "Collaborative Writing Support Tools on the Cloud," *IEEE Transactions on Learning Technologies,* vol. 4, pp. 88-97, 2011.

[75] W. Cellary and S. Strykowski, "e-government based on cloud computing and service-oriented architecture," presented at the Proceedings of the 3rd international conference on Theory and practice of electronic governance, Bogota, Colombia, 2009.

[76] M. Pokharel, *et al.*, "Cloud Computing in System Architecture," in *International Symposium on Computer Network and Multimedia Technology, 2009. CNMT 2009*, ed: IEEE, 2009, pp. 1-5 %@ 978-1-4244-5272-9.
[77] D. Zissis and D. Lekkas, "Securing e-Government and e-Voting with an open cloud computing architecture," *Government Information Quarterly,* vol. 28, pp. 239-251 %U http://www.sciencedirect.com/science/article/pii/S0740624X10001383, 2011.
[78] S. Chungman, *et al.*, "Implementation of Cloud Computing Environment for Discrete Event System Simulation using Service Oriented Architecture," in *2010 IEEE/IFIP 8th International Conference on Embedded and Ubiquitous Computing (EUC)*, ed: IEEE, 2010, pp. 359-362 %@ 978-1-4244-9719-5.
[79] K. Durski, *et al.*, "Warehouse management system in Ruby on Rails framework on cloud computing architecture," in *2011 11th International Conference The Experience of Designing and Application of CAD Systems in Microelectronics (CADSM)*, ed: IEEE, 2011, pp. 366-369 %@ 978-1-4577-0042-2.
[80] L. Juszczyk, *et al.*, "CAGE: Customizable Large-Scale SOA Testbeds in the Cloud," in *Service-Oriented Computing*. vol. 6568, E. Maximilien, *et al.*, Eds., ed: Springer Berlin / Heidelberg, 2011, pp. 76-87.
[81] H. J. La and S. D. Kim, "A Service-Based Approach to Designing Cyber Physical Systems," presented at the Proceedings of the 2010 IEEE/ACIS 9th International Conference on Computer and Information Science, 2010.
[82] G. Lackermair, "Hybrid cloud architectures for the online commerce," *Procedia Computer Science,* vol. 3, pp. 550-555 %U http://www.sciencedirect.com/science/article/pii/S1877050910004667, 2011.
[83] C.-F. Lai, *et al.*, "CPRS: A cloud-based program recommendation system for digital TV platforms," *Future Generation Computer Systems,* vol. 27, pp. 823-835 %U http://www.sciencedirect.com/science/article/pii/S0167739X10001950, 2011.
[84] J. Namjoshi and A. Gupte, "Service Oriented Architecture for Cloud Based Travel Reservation Software as a Service," in *IEEE International Conference on Cloud Computing, 2009. CLOUD '09*, ed: IEEE, 2009, pp. 147-150 %@ 978-1-4244-5199-9.
[85] J. Sedayao, "Implementing and operating an internet scale distributed application using service oriented architecture principles and cloud computing infrastructure," presented at the Proceedings of the 10th International Conference on Information Integration and Web-based Applications & Services, Linz, Austria, 2008.
[86] F. Shaochong, *et al.*, "Remodeling traditional RTI software to be with PaaS architecture," in *2010 3rd IEEE International Conference on Computer Science and Information Technology (ICCSIT)* vol. 1, ed: IEEE, 2010, pp. 511-515 %@ 978-1-4244-5537-9.
[87] X. Siyu and C. Genguo, "Application Research of CRM Based on SaaS," in *2010 International Conference on E-Business and E-Government (ICEE)*, ed: IEEE, 2010, pp. 3090-3092 %@ 978-0-7695-3997-3.
[88] L. Tang, *et al.*, "A Conceptual Framework for Business Intelligence as a Service (SaaS BI)," in *2011 International Conference on Intelligent Computation Technology and Automation (ICICTA)* vol. 2, ed: IEEE, 2011, pp. 1025-1028 %@ 978-1-61284-289-9.
[89] L. Xiaolin, "An Approach to Service and Cloud Computing Oriented Web GIS Application," in *2010 International Conference on Internet Technology and Applications*, ed: IEEE, 2010, pp. 1-4 %@ 978-1-4244-5142-5.
[90] L. Xiaolin, "Service and cloud computing oriented web GIS for labor and social security applications," in *2010 2nd International Conference on Information Science and Engineering (ICISE)*, ed: IEEE, 2010, pp. 4014-4017 %@ 978-1-4244-7616-9.
[91] L. Xiaopeng, *et al.*, "Architecture of Web-EDA system based on Cloud computing and application for project management of IC design," in *2010 International Conference on Anti-Counterfeiting Security and Identification in Communication (ASID)*, ed: IEEE, 2010, pp. 150-153 %@ 978-1-4244-6731-0.
[92] L. Ya-Chun, *et al.*, "A cloud computing framework of free view point real-time monitor system working on mobile devices," in *2010 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS)*, ed: IEEE, 2010, pp. 1-4 %@ 978-1-4244-7369-4.
[93] S.-Y. Yang, *et al.*, "A new ubiquitous information agent system for cloud computing - Example on GPS and Bluetooth techniques in Google Android platform," in *2011*

International Conference on Electric Information and Control Engineering (ICEICE), ed: IEEE, 2011, pp. 1929-1932 %@ 978-1-4244-8036-4.

[94] B. Zhang*, et al.*, "An Efficient Cloud Computing-Based Architecture for Freight System Application in China Railway," in *Cloud Computing*. vol. 5931, M. Jaatun*, et al.*, Eds., ed: Springer Berlin / Heidelberg, 2009, pp. 359-368.

[95] X. Zhang and G. Dong, "A New Architecture of Online Trading Platform Based on Cloud Computing," in *2010 Asia-Pacific Conference on Wearable Computing Systems (APWCS)*, ed: IEEE, 2010, pp. 32-35 %@ 978-1-4244-6467-8.

[96] Y. Zhihe and Z. Zhiting, "Construction of OSS-based E-Learning cloud in China," in *2010 2nd International Conference on Education Technology and Computer (ICETC)* vol. 2, ed: IEEE, 2010, pp. V2-398-V2-401 %@ 978-1-4244-6367-1.

[97] T. Zhimin*, et al.*, "Integrating Cloud-Computing-Specific Model into Aircraft Design," in *Cloud Computing*. vol. 5931, M. Jaatun*, et al.*, Eds., ed: Springer Berlin / Heidelberg, 2009, pp. 632-637.

[98] W. Zhiqiang*, et al.*, "Research and design of Cloud architecture for smart home," in *2010 IEEE International Conference on Software Engineering and Service Sciences (ICSESS)*, ed: IEEE, 2010, pp. 86-89 %@ 978-1-4244-6054-0.

[99] C. Wohlin*, et al.*, *Experimentation in software engineering: an introduction*: Kluwer Academic Publishers, 2000.

Appendix A

Search Queries

Following are search queries that were used to search relevant electronic database.

IEEE

(("Abstract":"cloud computing") OR ("Abstract":"cloud") OR ("Abstract":"cloud technologies") OR ("Document Title:"cloud computing") OR ("Document Title":"cloud") OR ("Document Title":"cloud technologies")) AND (("Abstract":"architecture") OR ("Abstract":"architectures") OR ("Abstract":"software as a service") OR ("Abstract":"SaaS") OR ("Abstract":"platform as a service") OR ("Abstract":"PaaS") OR ("Abstract":"infrastructure as a service") OR ("Abstract":"IaaS") OR ("Document Title":"architecture") OR ("Document Title":"architectures") OR ("Document Title":"software as a service") OR ("Document Title":"SaaS") OR ("Document Title":"platform as a service") OR ("Document Title":"PaaS") OR ("Document Title":"infrastructure as a service") OR ("Document Title":"IaaS"))

ACM

(Owner:ACM) AND

(Abstract:"cloud computing" OR Abstract":"cloud" OR Abstract:"cloud technologies" OR Title:"cloud computing" OR Title:"cloud" OR Title:"cloud technologies") AND (Abstract:"architecture" OR Abstract:"architectures" OR Abstract:"software as a service" OR Abstract:"SaaS" OR Abstract:"platform as a service" OR Abstract:"PaaS" OR Abstract:"infrastructure as a service" OR Abstract:"IaaS" OR Title:"architecture" OR Title:"architectures" OR Title:"software as a service" OR Title:"SaaS" OR Title:"platform as a service" OR Title:"PaaS" OR Title:"infrastructure as a service" OR Title:"IaaS")

Springer

Its search interface is changes and new one does not allow search string greater than 100 characters that is why search string for springer is divided into multiple substrings

((“cloud computing” OR “cloud” OR “cloud technologies”) AND (“architecture” OR “architectures”))

((“cloud computing” OR “cloud” OR “cloud technologies”) AND (“software as a service” OR “SaaS”))

((“cloud computing” OR “cloud” OR “cloud technologies”) AND (“platform as a service” OR “PaaS”))

((“cloud computing” OR “cloud” OR “cloud technologies”) AND (“infrastructure as a service”))

((“cloud computing” OR “cloud” OR “cloud technologies”) AND (“IaaS”))

ScienceDirect

(abs("cloud computing”) OR abs("cloud") OR abs("cloud technologies") OR ttl("cloud computing") OR ttl("cloud") OR ttl("cloud technologies")) AND (abs("architecture") OR abs("architectures") OR abs("software as a service") OR abs("SaaS") OR abs("platform as a service") OR abs("PaaS") OR abs("infrastructure as a service") OR abs("IaaS") OR ttl("architecture") OR ttl("architectures") OR ttl("software as a service") OR ttl("SaaS") OR ttl("platform as a service") OR ttl("PaaS") OR ttl("infrastructure as a service") OR ttl("IaaS"))

Appendix B

Table 27: Publication Count of Study Sources

Study Source	Publication Count
ACM Workshop on Green Computing	1
IEEE International Conference on Cloud Computing	11
IEEE International workshop on Software engineering for cloud computing	1
IEEE International Conference on Advanced Information Networking and Applications	1
International Middleware Conference	1
ACM Symposium on Applied Computing	1
IEEE Transactions on Consumer Electronics	1
international conference companion on World wide web	1
Hawaii International Conference on System Sciences	1
international conference industry session on Grids meets autonomic computing	1
International Conference and Seminar on Micro/Nanotechnologies and Electron Devices	1
Springer Journal	2
IEEE Transactions on Learning Technologies	1
ACM International conference on Theory and practice of electronic governance	1
International Conference on Computational Intelligence and Software Engineering	1
Springer - Cluster Computing	1
International Conference on Software Engineering Advances	1
Springer - World Wide Web	1
International Conference of the IEEE Engineering in Medicine and Biology Society	1
IEEE International Conference on e-Business Engineering (ICEBE)	3
ACM SIGPLAN conference companion on Object oriented programming systems languages and applications	2
IEEE/IFIP International Conference on Embedded and Ubiquitous Computing	1
IEEE/ACM International Conference on Cluster, Cloud and Grid Computing	1
ACM workshop on Scalable trusted computing	1
International Conference The Experience of Designing and Application of CAD Systems in Microelectronics (CADSM)	1

International conference on Management of data	2
International Symposium on Symbolic and Numeric Algorithms for Scientific Computing	1
IEEE Conference on Cloud Computing Technology and Science (CloudCom)	1
Springer - Database Systems for Advanced Applications	1
Springer - Journal of Computer Science and Technology	1
Springer - Annals of Telecommunications	1
Springer - Architecting Dependable Systems	1
IEEE International Conference on e-Business Engineering	1
World Congress on Software Engineering (WCSE)	1
IEEE International Symposium on Parallel & Distributed Processing (IPDPS)	1
ScienceDirect - Journal of Network and Computer Applications	1
IEEE/ACIS International Conference on Computer and Information Science	1
ScienceDirect - Procedia Computer Science	1
ScienceDirect - Future Generation Computer Systems	1
IEEE International Conference on Web Services (ICWS)	2
IEEE Conference on Commerce and Enterprise Computing (CEC)	1
Study Source	**Publication Count**
International conference on World wide web	1
International Workshop on Adaptive and Reflective Middleware	1
Data Compression Conference (DCC)	1
International Conference on Advanced Communication Technology (ICACT)	1
International Symposium on Computer Network and Multimedia Technology	1
International Conference on Wireless Communications, Networking and Mobile Computing	1
International Conference on Collaborative Computing: Networking, Applications and Worksharing	1
IEEE International Conference on Data Engineering Workshops (ICDEW)	1
International Conference on Information Integration and Web-based Applications & Services	1
Springer - Cloud Computing	2
IEEE International Conference on Computer Science and Information Technology (ICCSIT)	1
International Conference on E-Business and E-Government (ICEE)	1
International Conference on Communication Systems and Networks (COMSNETS)	1
International Conference on Intelligent Computation Technology and Automation (ICICTA)	1
ICSE Workshop on Software Engineering Challenges of Cloud Computing	1
World Congress on Services (SERVICES-1)	1
International Conference on Computer Engineering and Technology (ICCET)	1
International Conference on Internet Technology and Applications	1
International Conference on Information Science and Engineering (ICISE)	1
International Conference on Anti-Counterfeiting Security and Identification in Communication (ASID)	1
International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS)	1
International Conference on Electric Information and Control Engineering (ICEICE)	1
Asia-Pacific Conference on Wearable Computing Systems (APWCS)	1
IEEE International Conference on High Performance Computing and Communications (HPCC)	1
International Conference on Education Technology and Computer (ICETC)	1

IEEE International Conference on Software Engineering and Service Sciences (ICSESS)	1
IEEE Symposium on Web Society (SWS)	1
ScienceDirect - Government Information Quarterly	1

List of Tables

List of Figures

List of Included Studies

[S1]. Alvares de Oliveira, J., & Ledoux, T. (2010). Self-optimisation of the energy footprint in service-oriented architectures. *Proceedings of the 1st Workshop on Green Computing*. Bangalore, India: ACM.

[S2]. Azambuja, M., Pereira, R., Breitman, K., & Endler, M. (2010). An Architecture for Public and Open Submission Systems in the Cloud. *2010 IEEE 3rd International Conference on Cloud Computing (CLOUD)*. IEEE.

[S3]. Azeez, A., Perera, S., Gamage, D., Linton, R., Siriwardana, P., Leelaratne, D., Weerawarana, S., et al. (2010). Multi-tenant SOA Middleware for Cloud Computing. *2010 IEEE 3rd International Conference on Cloud Computing (CLOUD)*. IEEE.

[S4]. Babar, M. A., & Chauhan, M. A. (2011). A tale of migration to cloud computing for sharing experiences and observations. *Proceeding of the 2nd international workshop on Software engineering for cloud computing*. Waikiki, Honolulu, HI, USA: ACM.

[S5]. Badr, Y., & Caplat, G. (2010). Software-as-a-Service and Versionology: Towards Innovative Service Differentiation. *2010 24th IEEE International Conference on Advanced Information Networking and Applications (AINA)*. IEEE.

[S6]. Basu, S., Graupner, S., Pruyne, J., & Singhal, S. (2010). Control plane integration for cloud services. *Proceedings of the 11th International Middleware Conference Industrial track*. Bangalore, India: ACM.

[S7]. Baude, F., Filali, I., Huet, F., Legrand, V., Mathias, E., Merle, P., Ruz, C., et al. (2010). ESB federation for large-scale SOA. *Proceedings of the 2010 ACM Symposium on Applied Computing*. Sierre, Switzerland: ACM.

[S8]. Belimpasakis, P., & Moloney, S. (2009). A platform for proving family oriented RESTful services hosted at home. *IEEE Transactions on Consumer Electronics*, *55*(2), 690-698.

[S9]. Bonetta, D., & Pautasso, C. (2011). Towards liquid service oriented architectures. *Proceedings of the 20th international conference companion on World wide web*. Hyderabad, India: ACM.

[S10]. Botts, N., Thoms, B., Noamani, A., & Horan, T. A. (2010). Cloud Computing Architectures for the Underserved: Public Health Cyberinfrastructures through a Network of HealthATMs. *2010 43rd Hawaii International Conference on System Sciences (HICSS)*. IEEE.

[S11]. Brandic, I., Music, D., & Dustdar, S. (2009). Service mediation and negotiation bootstrapping as first achievements towards self-adaptable grid and cloud services. *Proceedings of the 6th international conference industry session on Grids meets autonomic computing*. Barcelona, Spain: ACM.

[S12]. Brizgalov, V. V., Chukhantsev, V., & Fedorkin, E. (2010). Architecture of traffic control systems using cloud computing. *2010 International Conference and Seminar on Micro/Nanotechnologies and Electron Devices (EDM)*. IEEE.

[S13]. Buyya, R., Pandey, S., & Vecchiola, C. (2009). Cloudbus Toolkit for Market-Oriented Cloud Computing. In M. Jaatun, G. Zhao, & C. Rong (Eds.), *Cloud Computing* (Vol. 5931, pp. 24-44). Springer Berlin / Heidelberg. doi:10.1007/978-3-642-10665-1_4

[S14]. Calvo, R. A., O'Rourke, S. T., Jones, J., Yacef, K., & Reimann, P. (2011). Collaborative Writing Support Tools on the Cloud. *IEEE Transactions on Learning Technologies*, *4*(1), 88-97.

[S15]. Cellary, W., & Strykowski, S. (2009). e-government based on cloud computing and service-oriented architecture. *Proceedings of the 3rd international conference on Theory and practice of electronic governance*. Bogota, Colombia: ACM.

[S16]. Changli, Z., & Maode, Y. (2010). Insurance-Based Cloud Computing-Architecture, Risk Analysis and Experiment. *2010 International Conference on Computational Intelligence and Software Engineering (CiSE)*. IEEE.

[S17]. Chapman, C., Emmerich, W., Márquez, F., Clayman, S., & Galis, A. (2011). Software architecture definition for on-demand cloud provisioning. *Cluster Computing*, 1-22. Springer Netherlands. doi:10.1007/s10586-011-0152-0

[S18]. Chazalet, A. (2010). Service Level Agreements Compliance Checking in the Cloud Computing: Architectural Pattern, Prototype, and Validation. *2010 Fifth International Conference on Software Engineering Advances (ICSEA)*. IEEE.

[S19]. Cheng, D.-Y., Chao, K.-M., Lo, C.-C., & Tsai, C.-F. (2011). A user centric service-oriented modeling approach. *World Wide Web*, *14*(4), 431-459. Springer Netherlands. doi:10.1007/s11280-011-0115-7

[S20]. Chia-Ping, S., Wei-Hsin, C., Jia-Ming, C., Kai-Ping, H., Jeng-Wei, L., Ming-Jang, C., Chi-Huang, C., et al. (2010). Bio-signal analysis system design with support vector machines based on cloud computing service architecture. *2010 Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE.

[S21]. Chieu, T. C., Mohindra, A., Karve, A. A., & Segal, A. (2009). Dynamic Scaling of Web Applications in a Virtualized Cloud Computing Environment. *IEEE International Conference on e-Business Engineering, 2009. ICEBE '09*. IEEE.

[S22]. Christensen, J. H. (2009). Using RESTful web-services and cloud computing to create next generation mobile applications. *Proceeding of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications*. Orlando, Florida, USA: ACM.

[S23]. Chungman, S., Youngshin, H., Haeyoung, L., Jung, J. J., & Chilgee, L. (2010). Implementation of Cloud Computing Environment for Discrete Event System Simulation using Service Oriented Architecture. *2010 IEEE/IFIP 8th International Conference on Embedded and Ubiquitous Computing (EUC)*. IEEE.

[S24]. Clarke, R. (2010). User Requirements for Cloud Computing Architecture. *2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing (CCGrid)*. IEEE.

[S25]. Domingo, E. J., Nino, J. T., Lemos, A. L., Lemos, M. L., Palacios, R. C., & BerbiÃÅs, J. M. G. (2010). CLOUDIO: A Cloud Computing-Oriented Multi-tenant Architecture for Business Information Systems. *2010 IEEE 3rd International Conference on Cloud Computing (CLOUD)*. IEEE.

[S26]. Du, J., Wei, W., Gu, X., & Yu, T. (2009). Towards secure dataflow processing in open distributed systems. *Proceedings of the 2009 ACM workshop on Scalable trusted computing*. Chicago, Illinois, USA: ACM.

[S27]. Durski, K., Murlewski, J., Makowski, D., & Sakowicz, B. (2011). Warehouse management system in Ruby on Rails framework on cloud computing architecture. *2011 11th International Conference The Experience of Designing and Application of CAD Systems in Microelectronics (CADSM)*. IEEE.

[S28]. Elmore, A. J., Das, S., Agrawal, D., & El Abbadi, A. (2011). Zephyr: live migration in shared nothing databases for elastic cloud platforms. *Proceedings of the 2011 international conference on Management of data*. Athens, Greece: ACM.

[S29]. Feng, L., Weiping, G., Zhi Qiang, Z., & Wu, C. (2010). SaaS Integration for Software Cloud. *2010 IEEE 3rd International Conference on Cloud Computing (CLOUD)*. IEEE.

[S30]. Giurgiu, I., Riva, O., Juric, D., Krivulev, I., & Alonso, G. (2009). Calling the Cloud: Enabling Mobile Phones as Interfaces to Cloud Applications. In J. Bacon & B. Cooper (Eds.), *Middleware 2009* (Vol. 5896, pp. 83-102). Springer Berlin / Heidelberg. doi:10.1007/978-3-642-10445-9_5

[S31]. Gogouvitis, S. V., Konstanteli, K., Kyriazis, D., & Varvarigou, T. (2010). An Architectural Approach for Event-Based Execution Management in Service Oriented Infrastructures. *2010 12th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)*. IEEE.

[S32]. Gorton, I., Yan, L., & Jian, Y. (2010). Exploring Architecture Options for a Federated, Cloud-Based System Biology Knowledgebase. *2010 IEEE Second International Conference on Cloud Computing Technology and Science (CloudCom)*. IEEE.

[S33]. Hagiwara, M. (2010). Development Procedure of the Cloud-Based Applications. In H. Kitagawa, Y. Ishikawa, Q. Li, & C. Watanabe (Eds.), *Database Systems for Advanced Applications* (Vol. 5982, pp. 320-326). Springer Berlin / Heidelberg. doi:10.1007/978-3-642-12098-5_26

[S34]. Han, Y.-B., Sun, J.-Y., Wang, G.-L., & Li, H.-F. (2010). A Cloud-Based BPM Architecture with User-End Distribution of Non-Compute-Intensive Activities and Sensitive Data. *Journal of Computer Science and Technology*, *25*(6), 1157-1167. Springer Boston. doi:10.1007/s11390-010-9396-z

[S35]. Hassan, M., Song, B., & Huh, E.-N. (2010). A market-oriented dynamic collaborative cloud services platform. *Annals of Telecommunications*, *65*(11), 669-688. Springer Paris. doi:10.1007/s12243-010-0184-0

[S36]. Hiltunen, M., & Schlichting, R. (2010). Is Collaborative QoS the Solution to the SOA Dependability Dilemma? In A. Casimiro, R. de Lemos, & C. Gacek (Eds.), *Architecting Dependable Systems VII* (Vol. 6420, pp. 227-248). Springer Berlin / Heidelberg. doi:10.1007/978-3-642-17245-8_10

[S37]. Hosono, S., He, H., Hara, T., Shimomura, Y., & Arai, T. (2010). A Lifetime Supporting Framework for Cloud Applications. *2010 IEEE 3rd International Conference on Cloud Computing (CLOUD)*. IEEE.

[S38]. Hyuck, H., Shingyu, K., Hyungsoo, J., Yeom, H. Y., Changho, Y., Jongwon, P., & Yongwoo, L. (2009). A RESTful Approach to the Management of Cloud Infrastructure. *IEEE International Conference on Cloud Computing, 2009. CLOUD '09*. IEEE.

[S39]. Hyun Jung, L., Si Won, C., & Soo Dong, K. (2009). Technical Challenges and Solution Space for Developing SaaS and Mash-Up Cloud Services. *IEEE International Conference on e-Business Engineering, 2009. ICEBE '09*. IEEE.

[S40]. Jia, X. (2010). Google Cloud Computing Platform Technology Architecture and the Impact of Its Cost. *2010 Second World Congress on Software Engineering (WCSE)*. IEEE.

[S41]. Jie, L., Humphrey, M., Agarwal, D., Jackson, K., van Ingen, C., & Youngryel, R. (2010). eScience in the cloud: A MODIS satellite data reprojection and reduction pipeline in the Windows Azure platform. *2010 IEEE International Symposium on Parallel & Distributed Processing (IPDPS)*. IEEE.

[S42]. Juszczyk, L., Schall, D., Mietzner, R., Dustdar, S., & Leymann, F. (2011). CAGE: Customizable Large-Scale SOA Testbeds in the Cloud. In E. Maximilien, G. Rossi, S.-T. Yuan, H. Ludwig, & M. Fantinato (Eds.), *Service-Oriented Computing* (Vol. 6568, pp. 76-87). Springer Berlin / Heidelberg. doi:10.1007/978-3-642-19394-1_9

[S43]. Khajeh-Hosseini, A., Greenwood, D., & Sommerville, I. (2010). Cloud Migration: A Case Study of Migrating an Enterprise IT System to IaaS. *2010 IEEE 3rd International Conference on Cloud Computing (CLOUD)*. IEEE.

[S44]. Kim, K. (n.d.). A model-driven workflow fragmentation framework for collaborative workflow architectures and systems. *Journal of Network and Computer Applications*, *In Press*, .

[S45]. Kossmann, D., Kraska, T., & Loesing, S. (2010). An evaluation of alternative architectures for transaction processing in the cloud. *Proceedings of the 2010 international conference on Management of data*. Indianapolis, Indiana, USA: ACM.

[S46]. La, H. J., & Kim, S. D. (2010). A Service-Based Approach to Designing Cyber Physical Systems. *Proceedings of the 2010 IEEE/ACIS 9th International Conference on Computer and Information Science*. IEEE Computer Society. Retrieved from http://dx.doi.org/10.1109/ICIS.2010.73

[S47]. Lackermair, G. (2011). Hybrid cloud architectures for the online commerce. *Procedia Computer Science*, *3*, 550-555 %U http://www.sciencedirect.com/science/ar.

[S48]. Lai, C.-F., Chang, J.-H., Hu, C.-C., Huang, Y.-M., & Chao, H.-C. (2011). CPRS: A cloud-based program recommendation system for digital TV platforms. *Future Generation Computer Systems*, *27*(6), 823-835 %U http://www.sciencedirect.com/science/ar.

[S49]. Liang-Jie, Z., & Qun, Z. (2009). CCOA: Cloud Computing Open Architecture. *IEEE International Conference on Web Services, 2009. ICWS 2009*. IEEE.

[S50]. Liver, B., Di Paolo, R., & Tice, K. (2010). Privacy in Service Oriented Architectures: SOA Boundary Identity Masking for Enterprises. *2010 IEEE 12th Conference on Commerce and Enterprise Computing (CEC)*. IEEE.

[S51]. Ludwig, H., Laredo, J., Bhattacharya, K., Pasquale, L., & Wassermann, B. (2009). REST-based management of loosely coupled services. *Proceedings of the 18th international conference on World wide web*. Madrid, Spain: ACM.

[S52]. Maximilien, E. M., Ranabahu, A., Engehausen, R., & Anderson, L. (2009). IBM altocumulus: a cross-cloud middleware and platform. *Proceeding of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications*. Orlando, Florida, USA: ACM.

[S53]. Namjoshi, J., & Gupte, A. (2009). Service Oriented Architecture for Cloud Based Travel Reservation Software as a Service. *IEEE International Conference on Cloud Computing, 2009. CLOUD '09*. IEEE.

[S54]. Papakos, P., Capra, L., & Rosenblum, D. S. (2010). VOLARE: context-aware adaptive cloud service discovery for mobile systems. *Proceedings of the 9th International Workshop on Adaptive and Reflective Middleware*. Bangalore, India: ACM.

[S55]. Pereira, R., Azambuja, M., Breitman, K., & Endler, M. (2010). An Architecture for Distributed High Performance Video Processing in the Cloud. *2010 IEEE 3rd International Conference on Cloud Computing (CLOUD)*. IEEE.

[S56]. Pereira, R., & Breitman, K. (2011). A Cloud Based Architecture for Improving Video Compression Time Efficiency: The Split & Merge Approach. *Data Compression Conference (DCC), 2011*. IEEE.

[S57]. Pervez, Z., Sungyoung, L., & Young-Koo, L. (2010). Multi-Tenant, Secure, Load Disseminated SaaS Architecture. *2010 The 12th International Conference on Advanced Communication Technology (ICACT)*. IEEE.

[S58]. Pokharel, M., YoungHyun, Y., & Jong Sou, P. (2009). Cloud Computing in System Architecture. *International Symposium on Computer Network and Multimedia Technology, 2009. CNMT 2009*. IEEE.

[S59]. Qian, W., & Deters, R. (2009). SOA's Last Mile-Connecting Smartphones to the Service Cloud. *IEEE International Conference on Cloud Computing, 2009. CLOUD '09*. IEEE.

[S60]. Qingfeng, L., Xie, J., Jicheng, H., Hongchen, Z., & Shanshan, Z. (2009). An Optimized Solution for Mobile Environment Using Mobile Cloud Computing. *5th International Conference on Wireless Communications, Networking and Mobile Computing, 2009. WiCom '09*. IEEE.

[S61]. Rodriguez, P., Gallego, D., Cervino, J., Escribano, F., Quemada, J., & Salvachua, J. (2009). VaaS: Videoconference as a service. *5th International Conference on Collaborative Computing: Networking, Applications and Worksharing, 2009. CollaborateCom 2009*. IEEE.

[S62]. Schaffner, J., Jacobs, D., Eckart, B., Brunnert, J., & Zeier, A. (2010). Towards enterprise software as a service in the cloud. *2010 IEEE 26th International Conference on Data Engineering Workshops (ICDEW)*. IEEE.

[S63]. Sedayao, J. (2008). Implementing and operating an internet scale distributed application using service oriented architecture principles and cloud computing infrastructure. *Proceedings of the 10th International Conference on Information Integration and Web-based Applications & Services*. Linz, Austria: ACM.

[S64]. Shang, L., Petiton, S., Emad, N., Yang, X., & Wang, Z. (2009). Extending YML to Be a Middleware for Scientific Cloud Computing. In M. Jaatun, G. Zhao, & C. Rong (Eds.), *Cloud Computing* (Vol. 5931, pp. 662-667). Springer Berlin / Heidelberg. doi:10.1007/978-3-642-10665-1_69

[S65]. Shaochong, F., Yanqiang, D., Yuanchang, & Zhu Xianguo, M. (2010). Remodeling traditional RTI software to be with PaaS architecture. *2010 3rd IEEE International Conference on Computer Science and Information Technology (ICCSIT)*. IEEE.

[S66]. Siyu, X., & Genguo, C. (2010). Application Research of CRM Based on SaaS. *2010 International Conference on E-Business and E-Government (ICEE)*. IEEE.

[S67]. Sodhi, B., & Prabhakar, T. V. (2011). Application architecture considerations for cloud platforms. *2011 Third International Conference on Communication Systems and Networks (COMSNETS)*. IEEE.

[S68]. Tang, L., Ni, Z., Wu, Z., & Wang, L. (2011). A Conceptual Framework for Business Intelligence as a Service (SaaS BI). *2011 International Conference on Intelligent Computation Technology and Automation (ICICTA)*. IEEE.

[S69]. Wei, S., ChangJie, G., Zhongbo, J., Xin, Z., Ning, D., Ying, H., & Yue Da, X. (2010). Design Aspects of Software as a Service to Enable E-Business through Cloud Platform. *2010 IEEE 7th International Conference on e-Business Engineering (ICEBE)*. IEEE.

[S70]. Wei-Tek, T., Qihong, S., Xin, S., & Elston, J. (2010). Real-Time Service-Oriented Cloud Computing. *2010 6th World Congress on Services (SERVICES-1)*. IEEE.

[S71]. Wenjun, Z. (2010). 2-Tier Cloud Architecture with maximized RIA and SimpleDB via minimized REST. *2010 2nd International Conference on Computer Engineering and Technology (ICCET)*. IEEE.

[S72]. Xiaolin, L. (2010a). Service and cloud computing oriented web GIS for labor and social security applications. *2010 2nd International Conference on Information Science and Engineering (ICISE)*. IEEE.

[S73]. Xiaolin, L. (2010b). An Approach to Service and Cloud Computing Oriented Web GIS Application. *2010 International Conference on Internet Technology and Applications*. IEEE.

[S74]. Xiaopeng, L., Yiyang, L., Huaiyu, D., & Donghui, G. (2010). Architecture of Web-EDA system based on Cloud computing and application for project management of IC design. *2010 International Conference on Anti-Counterfeiting Security and Identification in Communication (ASID)*. IEEE.

[S75]. Ya-Chun, L., Liao, I. J., Hua-Pu, C., & Wei-Tsong, L. (2010). A cloud computing framework of free view point real-time monitor system working on mobile devices. *2010 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS)*. IEEE.

[S76]. Yang, S.-Y., Dong-Liang, L., & Kune-Yao, C. (2011). A new ubiquitous information agent system for cloud computing - Example on GPS and Bluetooth techniques in Google Android platform. *2011 International Conference on Electric Information and Control Engineering (ICEICE)*. IEEE.

[S77]. Zhang, B., Zhang, N., Li, H., Liu, F., & Miao, K. (2009). An Efficient Cloud Computing-Based Architecture for Freight System Application in China Railway. In M. Jaatun, G. Zhao, & C. Rong (Eds.), *Cloud Computing* (Vol. 5931, pp. 359-368). Springer Berlin / Heidelberg. doi:10.1007/978-3-642-10665-1_33

[S78]. Zhang, X., & Dong, G. (2010). A New Architecture of Online Trading Platform Based on Cloud Computing. *2010 Asia-Pacific Conference on Wearable Computing Systems (APWCS)*. IEEE.

[S79]. Zhengxiong, H., Xingshe, Z., Jianhua, G., Yunlan, W., & Tianhai, Z. (2010). ASAAS: Application Software as a Service for High Performance Cloud Computing. *2010 12th IEEE International Conference on High Performance Computing and Communications (HPCC)*. IEEE.

[S80]. Zhihe, Y., & Zhiting, Z. (2010). Construction of OSS-based E-Learning cloud in China. *2010 2nd International Conference on Education Technology and Computer (ICETC)*. IEEE.

[S81]. Zhimin, T., Qi, L., & Guangwen, Y. (2009). Integrating Cloud-Computing-Specific Model into Aircraft Design. In M. Jaatun, G. Zhao, & C. Rong (Eds.), *Cloud Computing* (Vol. 5931, pp. 632-637). Springer Berlin / Heidelberg. doi:10.1007/978-3-642-10665-1_64

[S82]. Zhiqiang, W., Shuwei, Q., Dongning, J., & Yongquan, Y. (2010). Research and design of Cloud architecture for smart home. *2010 IEEE International Conference on Software Engineering and Service Sciences (ICSESS)*. IEEE.

[S83]. Zhiyun, G., Meina, S., & Qian, W. (2010). A framework of enterprise cloud application. *2010 IEEE 2nd Symposium on Web Society (SWS)*. IEEE.

[S84]. Zissis, D., & Lekkas, D. (2011). Securing e-Government and e-Voting with an open cloud computing architecture. *Government Information Quarterly*, *28*(2), 239-251 %U http://www.sciencedirect.com/science/ar.

[S85]. Zou, J., Jing, M., & Yan, W. (2010). From Representational State Transfer to Accountable State Transfer Architecture. *2010 IEEE International Conference on Web Services (ICWS)*. IEEE.

[S86]. van der Burg, S., de Jonge, M., Dolstra, E., & Visser, E. (2009). Software deployment in a dynamic cloud: From device to service orientation in a hospital environment. *ICSE Workshop on Software Engineering Challenges of Cloud Computing, 2009. CLOUD '09*. IEEE.