IJCC Editorial Board

Editors-in-Chief
Hemant Jain, University of Wisconsin–Milwaukee, USA
Rong Chang, IBM T.J. Watson Research Center, USA

Associate Editor-in-Chief
Bing Li, Wuhan University, China

Editorial Board
Danilo Ardagna, Politecnico di Milano, Italy
Janaka Balasooriya, Arizona State University, USA
Roger Barga, Microsoft Research, USA
Viraj Bhat, Yahoo, USA
Rajdeep Bhowmik, Cisco Systems, Inc., USA
Jiannong Cao, Hong Kong Polytechnic University, Hong Kong
Buqing Cao, Hunan University of Science and Technology, China
Keke Chen, Wright State University, USA
Haopeng Chen, Shanghai Jiao Tong University, China
Malolan Chetlur, IBM India, India
Alfredo Cuzzocrea, ICAR–CNR & University of Calabria, Italy
Ernesto Damiani, University of Milan, Italy
De Palma, University Joseph Fourier, France
Claude Godart, Nancy University and INRIA, France
Nils Gruschka, University of Applied Sciences, Germany
Paul Hofmann, Saffron Technology, USA
Ching-Hsien Hsu, Chung Hua University, Taiwan
Patrick Hung, University of Ontario Institute of Technology, Canada
Hai Jin, HUST, China
Li Kuang, Central South University, China
Grace Lin, Institute for Information Industry, Taiwan
Xumin Liu, Rochester Institute of Technology, USA
Shiyong Lu, Wayne State University, USA
J.P. Martin-Flatin, EPFL, Switzerland
Vijay Naik, IBM T.J. Watson Research Center, USA
Surya Nepal, Commonwealth Scientific and Industrial Research Organisation, Australia
Norbert Ritter, University of Hamburg, Germany
Josef Schiefer, Vienna University of Technology, Austria
Jun Shen, University of Wollongong, Australia
Weidong Shi, University of Houston, USA
Liuba Shrira, Brandeis University, USA
Kwang Mong Sim, University of Kent, UK
Wei Tan, IBM T.J. Watson Research Center, USA
Tao Tao, IBM T. J. Watson Research Center, USA
Kunal Verma, Accenture Technology Labs, USA
Raymond Wong, University of New South Wales & NICTA, Australia
Qi Yu, Rochester Institute of Technology, USA
Jia Zhang, Carnegie Mellon University – Silicon Valley, USA
Gong Zhang, Oracle Corporation, USA
Call for Articles
International Journal of Cloud Computing

Mission
Cloud Computing has become the de facto computing paradigm for Internet-scale service development, delivery, brokerage, and consumption in the era of Services Computing, fueling innovative business transformation and connected human society. 15 billion smart devices would be communicating dynamically over inter-connected clouds by 2015 as integral components of various industrial service ecosystems. The technical foundations of this trend include Service-Oriented Architecture (SOA), business & IT process automation, software-defined computing resources, elastic programming model & framework, and big data management and analytics. In terms of the delivered service capabilities, a cloud service could be, among other as-a-service types, an infrastructure service (managing compute, storage, and network resources), a platform service (provisioning generic or industry-specific programming API & runtime), a software application service (offering email-like ready-to-use application capabilities), a business process service (providing a managed process for, e.g., card payment), a mobile backend service (facilitating the integration between mobile apps and backend cloud storage and capabilities) or an Internet-of-things service (connecting smart machines with enablement capabilities for industrial clouds).

The International Journal of Cloud Computing (IJCC) aims to be a reputable resource providing leading technologies, development, ideas, and trends to an international readership of researchers and engineers in the field of Cloud Computing. IJCC only considers extended versions of conference papers published at reputable conferences such as IEEE International Conference of Cloud Computing.

Topics
The International Journal of Cloud Computing (IJCC) covers state-of-the-art technologies and best practices of Cloud Computing, as well as emerging standards and research topics which would define the future of Cloud Computing. Topics of interest include, but are not limited to, the following:

- ROI Model for Infrastructure, Platform, Application, Business, Social, Mobile, and IoT Clouds
- Cloud Computing Architectures and Cloud Solution Design Patterns
- Self-service Cloud Portal, Business Dashboard, and Operations Management Dashboard
- Autonomic Process and Workflow Management in Clouds
- Cloud Service Registration, Composition, Federation, Bridging, and Bursting
- Cloud Orchestration, Scheduling, Autoprovisioning, and Autoscaling
- Cloud Enablement in Storage, Data, Messaging, Streaming, Search, Analytics, and Visualization
- Software-Defined Resource Virtualization, Composition, and Management for Cloud
- Security, Privacy, Compliance, SLA, and Risk Management for Public, Private, and Hybrid Clouds
- Cloud Quality Monitoring, Service Level Management, and Business Service Management
- Cloud Reliability, Availability, Serviceability, Performance, and Disaster Recovery Management
- Cloud Asset, Configuration, Software Patch, License, and Capacity Management
- Cloud DevOps, Image Lifecycle Management, and Migration
- Cloud Solution Benchmarking, Modeling, and Analytics
- High Performance Computing and Scientific Computing in Cloud
- Cloudlet, Cloud Edge Server, Cloud Gateway, and IoT Cloud Devices
- Cloud Programming Model, Paradigm, and Framework
- Cloud Metering, Rating, and Accounting
- Innovative Cloud Applications and Experiences
- Green Cloud Computing and Cloud Data Center Modularization
- Economic Model and Business Consulting for Cloud Computing
Table of Contents

EDITOR-IN-CHIEF PREFACE
Hemant Jain, University of Wisconsin–Milwaukee, USA
Rong Chang, IBM T.J. Watson Research Center, USA

RESEARCH ARTICLES
1 METACONF: Learning Based Configuration File Discovery with File Metadata
Fanjing Meng, IBM Research China, Beijing, P.R. China
Han Wang, IBM Research China, Beijing, P.R. China; Rensselaer Polytechnic Institute, Troy, NY 12180, U.S.A.
Xuejun Zhuo, IBM Research China, Beijing, P.R. China
Jingmin Xu, IBM Research China, Beijing, P.R. China
Lin Yang, IBM Research China, Beijing, P.R. China
Changsheng Li, IBM Research China, Beijing, P.R. China

13 An Investigation of Anticipated Benefits of Cloud Computing Adoption in Australian Regional Municipal Governments
Omar Ali, University of Southern Queensland
Jeffrey Soar, University of Southern Queensland
Jianming Yongzhi, University of Southern Queensland

25 Towards Semantic-based Cloud Application Management
Kyriakos Kritikos, ICS FORTH
Dimitris Plexousakis, ICS FORTH

41 Call for Papers: IEEE CLOUD/ICWS/SCC/MS/BigData/SERVICES 2015
Call for Articles: International Journal of Services Computing (IJSC)
Call for Articles: International Journal of Big Data (IJBD)
Editor-in-Chief Preface:  
Cloud Management and Assessment  

Hemant Jain  
University of Wisconsin–Milwaukee, USA  
Rong Chang  
IBM T.J. Watson Research, USA  

Welcome to the inaugural issue of International Journal of Cloud Computing (IJCC), the first open access on-line journal on cloud computing. The increasing importance of cloud computing is evidenced from the rapid adoption of this technology in businesses around the globe. The cloud computing is redefining the business model of various industries from video rental (Netflix is enabled by cloud) to small start-up companies (companies can be started with very little investment using cloud infrastructure). The potential of cloud computing is even more promising. The cloud computing combined with developments like internet of things can significantly change the life as we know today. However, to deliver on these promises and to prevent cloud computing from becoming a passing fad significant technical, economic, and business issues need to be addressed. IJCC is designed to be an important platform for disseminating high quality research on above issues in a timely manner and provide an ongoing platform for continuous discussion on research published in this journal. We aim to publish high quality research that addresses important technical challenges, economics of sustaining this environment, and business issues related to use of this technology including privacy and security concerns, legal protection, etc. We seek to publish original research articles, expanded version of papers presented at high quality conferences, key survey articles that summarizes the research done so far and identify important research issues, and some visionary articles. We will make every effort to publish articles in a timely manner.

This inaugural issue collects the extended version of five IEEE CLOUD 2013 articles in the general area of managing Cloud computing environment.

The first article titled, *QOS-Based Resource Allocation Framework for Multi-Domain SLA Management in Clouds* by Lu, Yahyapour, Wieder, Kotsokalis, Yaqub, and Jehangiri tackles the issue of downtime and service unavailability due to live migration. They present an OpenStack based implementation of a cloud resource allocation framework, named Generic SLA Manager, that supports downtime-aware VM selection and allocation during live migration of VMs. A simulation based evaluation of the proposed framework is reported as well.

The second article titled, “*Rapidly Alternating Bottlenecks: A Study of Two Cases in N-Tier Applications*” by Wang, Kanemasa, Li, Shimizu, Matsubara, Kawaba, and Pu reveals the importance of identifying the location of performance bottlenecks when scaling n-tier applications in computing clouds. They propose a bottleneck detection method that could be used to rapidly detect alternating bottlenecks. Experimental evaluation results for the proposed method are reported via two use cases.

The third article titled, “*Cross Cloud MapReduce: A Result Integrity Check Framework on Hybrid Clouds*” by Wang, Wei, and Srivatsa tackles the trust issue in adopting large-scale MapReduce on public clouds. They present a framework, named Cross Cloud MapReduce (CCMR), which overlays the MapReduce computation on a hybrid cloud where a master ensures correct result. A result integrity check scheme is also presented for accuracy and performance. Both theoretical and experimental analyses are reported.

The fourth article titled, “*Implementation and Empirical Assessment of A Web Application Cloud Deployment Tool*” by Sampaio, Costa, Mendonça, and Filho tackles the time-consuming issue in migrating applications to an IaaS cloud via application-specific VM images. They present an automated application deployment approach that requires less cataloged VM images. The approach can be supported via a tool, called TREXCLOUD, and an empirical evaluation of the tool is reported.
The fifth article titled, “A Queuing Model to Achieve Proper Elasticity for Cloud Cluster Jobs” by Salah tackles the issue of achieving proper elasticity for parallelized jobs running on cloud clusters. Based on finite queuing systems, the article presents an analytical model that can be used to determine the minimal number of cloud resources needed to satisfy the SLO requirements with constraints. Discrete Event Simulation is reported to verify the correctness of the proposed model.

We would like to thank the authors for their effort in delivering those five quality articles. We would also like to thank the reviewers as well as the Program Committee of IEEE CLOUD 2013 for their help with the review process. Finally, we are grateful for the effort Jia Zhang and Liang-Jie Zhang made in giving birth to this inaugural issue of International Journal of Cloud Computing (IJCC).

About the Editors-in-Chief

Dr. Hemant Jain is the Interim Director of Biomedical and Health Informatics Research Institute, Roger L. Fitzsimonds Distinguished Scholar and Professor of Information Technology Management at University of Wisconsin–Milwaukee. Dr. Jain specializes in information system agility through web services, service oriented architecture and component based development. His current interests include development of systems to support real time enterprises which have situational awareness, can quickly sense-and-respond to opportunities and threats, and can track-and-trace important items. He is also working on issues related to providing quick access to relevant knowledge for cancer treatment and to providing medical services through a virtual world. Dr. Jain is an expert in architecture design, database management and data warehousing. He teaches courses in database management, IT infrastructure design and management, and process management using SAP. Dr. Jain was the Associate Editor-in-Chief of IEEE Transactions on Services Computing and is Associate Editor of Journal of AIS, the flagship journal of the Association of Information Systems.

Dr. Rong N. Chang is Manager & Research Staff Member at the IBM T.J. Watson Research Center. He received his Ph.D. degree in computer science & engineering from the University of Michigan at Ann Arbor in 1990 and his B.S. degree in computer engineering with honors from the National Chiao Tung University in Taiwan in 1982. Before joining IBM in 1993, he was with Bellcore researching on B-ISDN realization. He is a holder of the ITIL Foundation Certificate in IT Services Management. His accomplishments at IBM include the completion of a Micro MBA Program, one IEEE Best Paper Award, and many IBM awards, including four corporate-level Outstanding Technical Achievement Awards and six division-level accomplishments. He is an Associate Editor of the IEEE Transactions on Services Computing and the International Journal of Services Computing. He has chaired many conferences & workshops in cloud computing and Internet-enabled distributed services and applications. He is an ACM Distinguished Member/Engineer, a Senior Member of IEEE, and a member of Eta Kappa Nu and Tau Beta Pi honor societies.
METACONF: LEARNING BASED CONFIGURATION FILE DISCOVERY WITH FILE METADATA

Fan Jing Meng¹, Han Wang¹,², Xuejun Zhuo¹, Jing Min Xu¹, Lin Yang¹, Changsheng Li¹
¹IBM Research – China, Beijing, P.R. China
{ mengfj, hanw, zhuoxuej, xijingm, lycyang, lcsheng }@cn.ibm.com
²Rensselaer Polytechnic Institute, Troy, NY 12180, U.S.A.
wangh17@rpi.edu

Abstract
Discovery of configuration files is one of the prerequisite activities for a successful workload migration to the cloud. The complicated and super-sized file systems, the considerable variance of configuration files, and the multiple-presence of configuration items make configuration file discovery very difficult. Traditional approaches usually highly rely on experts to compose software specific scripts or rules to discover configuration files, which is very expensive and labor-intensive. In this paper, we propose a novel learning based approach named MetaConf to convert configuration file discovery to a supervised file classification task. We use the file metadata as learning features and learn a classifier based on these features such that it can be conducted automatically, efficiently, and independently of domain expertise. We report our evaluation with extensive and real-world case studies, and the experimental results validate that our approach is effective and outperforms the baseline method.

Keywords: cloud migration; configuration file discovery; file metadata; fuzzy string matching; data imbalance

1. INTRODUCTION
Cloud computing is dramatically changing the modern enterprise IT. More and more workloads have been, are being, and are going to be moved to the cloud. According to Cisco Global Cloud Index, global cloud traffic will account for nearly two-thirds of total data center traffic by 2016 [1]. In 2012, International Data Corporation (IDC) reported that more than 50% of larger European companies stated that they have a strategy for refactoring IT infrastructure in order to benefit from cloud economics, of which the cloud migration investment is forecast to grow at 30.6% compound annual growth rate (CAGR) in the next 5 years[2].

Migrating applications to the cloud is considered as a risky, costly, labor-intensive, and error-prone activity. A typical workload migration process include the source environment discovery, migration analysis and design, source-target environment mapping, target environment provisioning, migration execution, and remediation and testing[3]. Discovery of source environment is the foundation for a successful workload migration[4], which aims to discovery the configurations of the existing applications and understand the characteristics of the application.

However, configuration file discovery is fraught with many challenges and complexities, as it is not a trivial job to distinguish configuration files from other files in the target file systems. The difficulty is due to the following issues: 1) The huge amount of configuration files. Enterprise applications are usually deployed in a distributed and multiple-layer software environment. Each software usually requires a large number of configuration files. For example, a Linux operating system individually has over 100 configuration files [5]. Complex enterprise software may expose even more configuration files. For instance, WebSphere® Application Server (WAS) Network Deployment Version 8.5 has more than 900 configuration files [6]. It is impossible to manually recognize all these configuration files from the whole stack of an enterprise application. 2) The variance of configuration files. The locations of configuration files usually vary among different deployment environments. For example, WAS defines the WebSphere® cell configurations into the configuration file “cell.xml”, which can have different locations such as “/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/config/cells/vm-tpm-s169Node01Cell/cell.xml” and “/home/application/profiles/SFAApp/config/cells/SFACell/cell.xml”, depending on the locations of profiles and other dependent configurations items (e.g. WebSphere® profile name and cell name). Besides the locations, the various formats (e.g. binary files and text files) of configuration files can also complicate matters if one attempts to recognize the configurations by parsing the file contents. 3) The multiple-presence of configurations. A configuration is usually defined in one configuration file but referred to by multiple other files including other configuration files, temporary files, and log files. This will cause simple configuration keyword search to fail if used for configuration file discovery. For example, the “hostname” of a Red Hat® Enterprise Linux server (RHEL) is defined in “/etc/sysconfig/network”, but was referred to by another 27 files in one of our experiment servers. Furthermore,
simple file name search in the given environment is not a reliable way to identify configuration files. For instance, we found 14 files named “cell.xml” in one of our experiment servers. Human intervention is necessary to distinguish the real WebSphere® cell configuration file from others (e.g. WebSphere® profile template files and profile backup files).

Existing configuration file discovery approaches can be approximately categorized into three types: 1) Script-based approaches, which run pre-developed domain specific scripts to discover configuration files on the target server. 2) Rule-based approaches, which apply pre-defined rules written by application experts to annotate configuration files on the target server. 3) Human interactive approaches, which engage end users to interact with the system to define patterns to match configuration files or specify them directly. All of these approaches require rich domain expertise and high familiarity with the software and system configurations. The main advantage of these approaches is that they are capable of achieving relatively high accuracy of identifying configuration files once the expertise requirement is met. The disadvantage is also obvious: the involvement of human expertise and labor makes these approaches very expensive and not scalable. Moreover, the pre-conceived and fixed knowledge base does not tolerant the fast evolution of software products and custom applications.

In contrast to these conventional methods, we propose a novel learning based approach named MetaConf to address the three major challenges discussed above. MetaConf is a machine learning approach that does not rely on domain expertise, and thus tackles the data volume and scalability issue. MetaConf does not require the parsing of the file contents but takes the file metadata as input and make discovery of configuration files automatic. This solves the variance and multi-presence issues. The main contributions of MetaConf are fourfold:

1) We propose an efficient learning-based configuration file discovery method by identifying the repetitive patterns of the metadata instead of parsing the contents of configuration files.

2) We design a keyword search, variance replacement and clustering-based approach to identify and label the configuration files with Configuration Items (CIs) from super-sized file systems to automate training data collection.

3) We propose an effective feature extraction scheme to retrieve essential learning features from the file metadata by transforming non-numerical features such as file paths to numerical values with a fuzzy token matching mechanism.

4) We address the learning dataset imbalance issue by adopting techniques such as resampling the datasets and applying unequal classification error penalties.

The rest of the paper is organized as follows. Section II discusses the motivation of our research work. In Section III, we discuss the main technical challenges and provide an overview of MetaConf. Section IV describes the detailed design of MetaConf. In Section V we report the evaluations through extensive case studies. In Section VI, we briefly review the existing related work. Section VII concludes the paper and discusses future work.

2. MOTIVATION

Migrating applications to the cloud is considered as a risky, costly, labor-intensive, and error-prone activity due to the complexity of the applications and the constraints of the clouds. In general, application migration to cloud techniques can be categorized into three major types: image-based migration, application-centric migration, and re-architecture-based migration.

1) The image-based migration technique deploys a process that converts the source server into a virtual image, imports a virtualized server image into a target cloud, adjusts the image such that it is compliant with the target cloud requirements, and then brings the modified image on-board the cloud. This type of migration technique only support the migration of applications to the IaaS cloud. Most IaaS clouds support this type of migration. Besides, some migration services providers adopt this technique e.g. Racemi®[23], CohesiveFT [24]. The image-based migration technique is relatively simple but cannot be applicable to those VMs whose OS are not supported by the target cloud or adapted to the OS supported by the target cloud. Besides, the reconfiguration or remediation of the migrated application is required to ensure it function after the image migration. Therefore, discovery of the configuration files is necessary task for this type of migration technique.

2) The application-centric migration is a migration technique that discovers and extracts the application artifacts, application dependent configurations and resources from source environments, matches the right deployment strategy on the target cloud, provisions the new application deployment environments on the target cloud, migrates the discovered application resources to the provisioned environment, and then remediates the applications on the target cloud. This type of migration technique is relatively complicated but the migrated application can fully leverage the value-added services provided by the cloud e.g. auto-scaling, fail-over, and etc. It can support the migration of applications to both IaaS cloud and PaaS cloud. Usually, the application-centric migration need a detailed strategic migration planning and execution led by the application owners or professional migration providers e.g. AppZero[25], CliQ™ [26]. Discovering the configuration files is the first step of the application-centric migration.

3) The re-architecture-based migration applies to those applications whose architecture or runtime
environment are not supported by the target cloud. In order to migrate these applications to the cloud, the migration team will discover the existing application configurations and dependencies, define the requirements of the application migration, assess the gaps of the current application design and the requirements of the cloud, re-architect and implement the application for the target cloud and then migrate the re-architected applications to the target cloud. In the three types of migration techniques, the re-architecture-based migration is the most complicated and costly but gives the application the opportunity to fully leverage the cloud advantages through the re-architecture e.g. the multi-tenancy enabled after the migration. This type of migration usually performed by dedicated migration team or experienced migration services providers e.g. IBM[27], Accenture[28].

From the detailed analysis of the three types of migration techniques, the discovery of application configuration is the prerequisite of a successful migration to cloud. However, the existing techniques generally depend on the pre-defined discovery knowledge and cannot solve the problem effectively. This motivates us to study a learning based application configuration file discovery approach and system to overcome these shortages.

3. OVERVIEW

Overall, we consider the configuration file discovery as a supervised file classification task where we treat each group of files containing the same configuration item as a file class, and when a new file is classified as a member of these classes, we claim that this file is a configuration file and it includes the same configuration item as its file class does. In this section, we discuss the main technical challenges and present an overview to the key processes of our approach, MetaConf.

3.1 CHALLENGES

In order to provide an efficient solution to the automatic discovery of configuration files without requiring the expense of human expertise and tedious manual intervention, we seek to learn the reusable knowledge for configuration file discovery from given examples with a reasonable volume. Specifically, for a given configuration item, we select files containing it as the positive instances and files not containing it as the negative instances, and then extract learning features from the file metadata to train a classifier. There exist three main challenges in this procedure: data labelling, feature extraction and data imbalance.

First, labelled training data is the prerequisite of the supervised learning algorithms. However, the labelled configuration files are not available even in the existing configuration management tools e.g. IBM Tivoli® Application Dependency Discovery Manager (TADDM) [13], HP Discovery and Dependency Mapping (DDM) [12], EMC Application Discovery Manager (ADM) [17]. These tools can only provide the discovered CIs from managed environment. In this case, we need to identify these configuration files given the CIs. It is still not trivial to acquire these configuration files from super-sized the file systems given the fact of considerable variances of configuration files and multi-presence of configuration values. Although the machine learning technique has highly potential to reduce the dependencies to pre-defined discovery knowledge, lack of labelled data is a big barrier and prevents its application in the configuration file discovery.

Second, by considering the fact that most configuration files have repetitive patterns in their file metadata including relative paths, file names, access permissions, etc. among different instances, we extract learning features from the file metadata instead of the file contents to significantly improve the training efficiency. However, how to represent and transform the non-numerical file metadata such as paths and extensions into numerical features to maintain their essential characteristics remains a challenging issue. Traditional string similarity metrics, either character-based similarity metric or token-based similarity metric, both have limitations in our scenario. The character-based similarity metric, such as edit distance that calculates the minimal number of edit operations to change one string to another, cannot deal with the case where two paths have similarity relative path but different absolute paths. For example, the file path “/etc/httpd/conf/httpd.conf” and the file path “/usr/local/apache2.0.65/conf/httpd.conf” would be considered less similar since they have a relatively large edit distance, but they actually both are the configuration file that contains the “MaxClient” configuration item. The token-based similarity, on the other hand, does not consider the relative position of the tokens in a repetitive pattern or subsequence. For instance, the path pair {“/usr/local/apache2.0.65/conf/httpd.conf”, “/usr/local/apache2.0.65/conf/custom.conf”} should have a lower similarity between its two paths than the pair {“/usr/local/ssh/etc/ssh/sshd_config”, “/etc/ssh/sshd_config”}; does, because the former pair refers to two configuration files containing different sets of configuration items while the latter pair usually include the same set of configuration items. But a token-based simiarity would give a higher similarity value for the former pair.

Third, due to the specificity of the problem, our learning datasets suffer from critical data imbalance. For a single server, there are usually less than ten configuration files that can be labeled as the positive instances for a given configuration item. However, the remaining thousands of files are all negative instances for this configuration item. This data imbalance tends make the classifier biased towards the majority class and therefore compromises the classification performance.

3.2 SYSTEM WORKFLOW

The workflow of MetaConf is shown in Fig. 1. The main steps include:
**Data Labelling:** In order to acquire the labelled data, we take a set of servers and their configurations as the input. Based on the fact that the configuration files share similar file metadata but with variances, we perform the configuration keyword search in the files to identify all the file candidates which include the given configurations. Then we match the configurations on the file metadata to identify the variable points and replace them with the CI names. For example, the configuration files for the CI “WASCellName” from the two different WAS deployment environment are the

```
"/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/profiles/wasCellName/config/cells/vm-tpm-s169Node01Cell/cell.xml"
```

and

```
"/home/application/profiles/sfaApp/config/cells/SFACell/cell.xml"
```

They have the low similarity on their file path. If we replace the variances with the CIs of “WASProfileHome”, “WASProfileName”, and “WASCellName” in the file path, we get the two file paths as

```
[WASProfileHome]/profiles/[WASProfileName]/config/cells/[WASCellName]/cell.xml
```

and

```
[WASProfileHome]/profiles/[WASProfileName]/config/cells/[WASCellName]/cell.xml
```

After the variable point matching and replacement, the variances have been reduced. Furthermore, we perform the clustering of these file candidates based on their metadata across servers. The configuration files share the most similar features will fall into the same cluster. Finally, the system administrators review the results and give the right labels on the positive instances and negative instances. The labelled data has been acquired and is ready for supervised learning.

**Token Base Construction:** As mentioned earlier, the file path and the file extension are two string features that should be transformed into numerical values the training process. Our idea is to match a file instance to a pre-built knowledge base that represents the target file class, and quantify this matching as the learning features. Therefore, as first step we construct two knowledge bases called Path Token Base and Extension Token Base respectively by capturing the representative tokens from the positive instances of a file class which contains a certain configuration item.

**Feature Extraction:** We select nine items from the file metadata and convert them into the learning features. They file path, file extension, size, user access, group access, other access, change time, access time, and modify time. For the file path, we design a fuzzy matching algorithm to compare a path to the Path Token Base, and calculate the similarity score as the feature. For the file extension, we search the Extension Token Base for a match of an extension, and use the matched token’s numerical weight as the feature. For the three time items, we replace them with diff(change time, modify time), diff(change time, access time), and diff(modify time, access time), to remove the environment dependency. For the rest of the items, we directly fed them into the training process since they are already numerical values.

**Data Imbalance Handling:** As discussed above, we introduced a system component to balance the class distribution in the training data to improve the classification performance. We implemented three different techniques: oversampling the minority (positive) class, undersampling the majority (negative) class, and assigning unequal error penalties for the two classes.

**Classifier Training:** With balanced datasets and extracted features, we trained a Support Vector Machine (SVM) [6] based configuration file classifier with cross-validation for each file class.

4. **System Design**

In this section we discuss the key components of our system including the population of labelled datasets, the algorithms of numerically representing file paths and file extensions, as well as the handling of data imbalance.

**Dataset Population**

Given the super-sized file systems and variances of configuration files in different environment, it is hard to identify the configuration files without the domain specific knowledge. The variance increases along the software stack from the OS, to the middleware, and to the applications. In order to address these challenges, we design a keyword search, variance replacement, and clustering-based approach
to reduce the dependency on these domain knowledge and human involvement in the data labeling process.

1) Data Cleaning

Firstly, we apply the iterative comparison approach reported in [3] and the domain independent rules to remove the obviously irrelevant files and narrow down the scope to discover these configurations. The iterative comparison approach can remove the software release files e.g. RHEL release files, WAS release files, which are the majority of files in the given server. Then we apply the domain independent rules specified by the Regular Expression to remove the files like the temporary files, log files, and cache files. After the data cleaning with the two approaches, the file number dramatically reduced to a reasonable size. Table 1 gives the data cleaning results on our 6 servers. The reduction ratio is over 92%.

**Table 1: Data Cleaning Results Example**

<table>
<thead>
<tr>
<th>Server</th>
<th># of Total Files</th>
<th># of files after cleaning</th>
<th>File Reduction Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server 1</td>
<td>282,887</td>
<td>9,014</td>
<td>96.8%</td>
</tr>
<tr>
<td>Server 2</td>
<td>149,396</td>
<td>11,234</td>
<td>92.5%</td>
</tr>
<tr>
<td>Server 3</td>
<td>256,428</td>
<td>11,295</td>
<td>95.6%</td>
</tr>
<tr>
<td>Server 4</td>
<td>151,710</td>
<td>11,245</td>
<td>92.6%</td>
</tr>
<tr>
<td>Server 5</td>
<td>572,603</td>
<td>11,856</td>
<td>97.9%</td>
</tr>
<tr>
<td>Server 6</td>
<td>273,372</td>
<td>11,471</td>
<td>95.8%</td>
</tr>
</tbody>
</table>

2) Keyword search

In order to identify the configuration file candidate, we perform the CIs keyword search in the remaining files. Take the CI “WASCellName” as the example, we use the CI value “vm81Node01Cell” as the keyword to do the file content search and get the CI configuration file candidates in the server vm81. After the search, 5 files containing the CI value “vm81Node01Cell” as the configuration file candidates, which are:

- /opt/IBM/WebSphere/AppServer/profiles/AppSr01/bin/setupCmdLine.sh
- /opt/IBM/WebSphere/AppServer/profiles/AppSr01/firststeps/firststeps.sh
- /opt/IBM/WebSphere/AppServer/profiles/AppSr01/config/cells/vm81Node01Cell/security.xml
- /opt/IBM/WebSphere/AppServer/profiles/AppSr01/config/cells/vm81Node01Cell/applications/Trade.ear/deployments/Trade/deployment.xml
- /opt/IBM/WebSphere/AppServer/profiles/AppSr01/config/cells/vm81Node01Cell/firststeps/firststeps.sh
- /opt/IBM/WebSphere/AppServer/profiles/AppSr01/config/cells/vm81Node01Cell/applications/Trade.ear/deployments/Trade/deployment.xml

For each CI, we get a set of the configuration file candidates on each server.

3) Variance Replacement

Although the configuration file share similar patterns in their file metadata e.g. file path, filename, file extension, the variances exist in the different deployment environment and reduce the configuration file discovery accuracy. In order to reduce these variances, we perform the variable point discovery and replacement in these file metadata with the CIs. For example, the three CIs are matched in the file metadata of the 5 “WASCellName” configuration file candidates. They are the “WASProfileHome”, “WASProfileName”, and “WASCellName”. After the variance replacement, these configuration file candidates are

- [WASProfileHome]/profiles/[WASProfileName]/AppSrv01/bin/setupCmdLine.sh
- [WASProfileHome]/profiles/[WASProfileName]/firststeps/firststeps.sh
- [WASProfileHome]/profiles/[WASProfileName]/config/cells/[WASCellName]/security.xml
- [WASProfileHome]/profiles/[WASProfileName]/config/cells/[WASCellName]/applications/Trade.ear/deployments/Trade/deployment.xml
- [WASProfileHome]/profiles/[WASProfileName]/config/cells/[WASCellName]/cell.xml

After the variance discovery and replace, we can get a set of configuration file candidates for each CI on each server.

4) Clustering

Given the notion that the configuration file share the similar file metadata, we perform the clustering on the all the variance reduced configuration file candidates from all the servers. Firstly, we normalize the textual file attribute with the edit distance into numeric values. Then, we apply DBScan, the density-based clustering algorithm, to cluster these configuration file candidates. Further, we calculate the average similarity degree by average of the pair-wise similarities of configuration file candidates within each cluster and rank the cluster by their similarity degree. Finally, the users with a certain level of domain knowledge e.g. system administrators review the clustering result and label the positive instances. Table 2 shows the clustering result for the CI “WASCellName”.

**Table 2: Clustering results for CI “WASCellName”**

<table>
<thead>
<tr>
<th>Cluster</th>
<th># of Files</th>
<th>Average similarity degree</th>
<th># of positive instances</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cluster 1</td>
<td>5</td>
<td>100%</td>
<td>5</td>
</tr>
<tr>
<td>Cluster 2</td>
<td>5</td>
<td>93.2%</td>
<td>0</td>
</tr>
<tr>
<td>Cluster 3</td>
<td>8</td>
<td>80.6%</td>
<td>0</td>
</tr>
<tr>
<td>Cluster 4</td>
<td>10</td>
<td>72.5%</td>
<td>0</td>
</tr>
</tbody>
</table>

4.2 REPRESENTATION OF FILE PATHS

An essential part of file metadata, the file path carries a great amount of information about a configuration file. Consider the following three instances of the WAS configuration file resources.xml:

- /opt/IBM/WebSphere/AppServer/profiles/AppSr01/config/cells/vm81Node01Cell/nodes/vm81Node01Cell/resources.xml
- /opt/IBM/WebSphere/AppServer/profiles/AppSr01/config/cells/vm81Node01Cell/nodes/logoHostNode11/resources.xml
- /home/wasadmin/Profiles/DefaultDaemon01/config/cells/TradeCell/resources.xml

the resources.xml file defines the key configuration items such as “JDBCDataSourceName”. In order to extract the repetitive patterns out of the file paths while tolerating the variance, we construct a knowledge base called Path Token Base for each class of configuration files. Each base entry (i.e. a path token or token sequence) represents a part of the most frequent locations of this file class and is assigned with a numerical weight that measures its significance based on its frequency, length, and position. A new file can then be
compared with this Path Token Base by quantifying the matching between the path of the new file and the Path Token Base. If the matching score is high enough, we will conclude that this new file is similar to the file class represented by this Path Token Base in the path dimension. We will introduce in detail how to build such a Path Token Base below.

1) Path token
As mentioned earlier, a class of configuration files tends to have a number of common string patterns in their paths. Our first goal is to tokenize the paths and obtain tokens, the smallest units constituting these patterns. We establish a series of heuristic rules to perform the tokenization, some of which are listed as follows:

- A path is split at each directory level.
- Digits are separated from letters and are ignored.
- Non-digital and non-alphabetic characters are used as delimiters to separate tokens if they are in the middle of a path, and are ignored if they are at the two ends.
- All tokens are converted to lowercase.
- Any token that has only two characters or less is ignored.
- File extensions are saved separately.

With these rules, a path such as 

```
"/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/config/cells/vm81Node01Cell/nodes/vm81Node01/resources.xml"
```

will be converted to a set of path tokens: 

```
{"opt", "ibm", "websphere", "appserver", "profiles", "appsrv", "config", "cells", "node", "cell", "nodes", "node", "resources"}.
```

2) Path token similarity
Having tokenized paths of a class of configuration files, we then deposit the most common path tokens into the Path Token Base. The commonality is measured by the similarity between the path tokens. If a set of tokens has higher pairwise similarity than others do, these tokens should be playing a more important part in representing the locations of this class of files. When comparing two paths and their tokens, we adopt a fuzzy matching method which is formulated as follows.

Given two paths $P_1$ and $P_2$, each of which has been tokenized and converted to a set of tokens $T_1$ and $T_2$, respectively. By treating the tokens as vertices, we consider a weighted bipartite graph $G = (T_1, T_2, E)$, where $E$ is a set of weighted edges such that (1) $T_1$ and $T_2$ are disjoint; (2) each vertex in one of $T_1$ and $T_2$ is connected to every vertex in the other set, i.e. $G$ is complete; (3) the weight associated with the edge connecting $t_1$ and $t_2$ is a Levenshtein distance [8] based similarity score $\text{sim}_{pt}(t_1, t_2)$ between $t_1$ and $t_2$. The Levenshtein distance is given by

$$
\text{lev}_{pt}(t_1, t_2) = \begin{cases} 
\min(\text{lev}_{pt}(t_1, t_2)) & \text{if } \min(i, j) = 0 \\
\text{lev}_{pt}(t_1, t_2) + 1 & \text{otherwise}
\end{cases}
$$

where $l_1$ and $l_2$ are the string lengths of the path tokens $t_1$ and $t_2$, respectively, and $(t_1, t_2)$ is a constant function equal to 0 when the last characters of $t_1$ and $t_2$ are the same, and equal to 1 otherwise. The path token similarity is then given by

$$
\text{sim}_{pt}(t_1, t_2) = \frac{1}{\max(l_1, l_2)} \text{lev}_{pt}(t_1, t_2)
$$

such that it becomes greater when the Levenshtein distance between path tokens is smaller. We then find the maximum weighted matching $M$ of $G$. Note that a matching in a graph is a set of pairwise non-adjacent edges. A maximum matching of a graph is a matching containing the largest possible number of edges. A maximum weighted matching is a matching where the sum of the weights on the edges in the matching has a maximum value. In $M$, each token (i.e. vertex) $t$ is connected to its neighbor by the edges that have the largest weight among all the edges it is incident with in $G$. Therefore $t$ is more similar to its neighbor than to any other tokens in $G$. In other words, $M$ provides a set of most similar token pairs from the two paths $P_1$ and $P_2$. With a similarity threshold $\lambda$, we include every token pair whose similarity is greater than $\lambda$ into the Path Token Base. We repeat this process for every path pair for the positive instances of a file class in the training data, and the Path Token Base will contain the most common path tokens for this file class. Obviously the commonality is associated with the value of the threshold $\lambda$, and we will show an experiment in the experiment section studying the influence of $\lambda$. The purpose of this maximum weighted bipartite matching setup is to avoid exact matching of the tokens and thus to increase the tolerance of small inconsistencies such as naming variance and typos in file paths.

Fig. 2 illustrates an example of using our maximum weighted bipartite matching setup to compute the path token similarities between two file paths:

- /opt/IBM/WebSphere/AppServer/profiles/AppSrv01/config/cells/vm81Node01Cell/nodes/vm81Node01/resources.xml
- /opt/IBM/WebSphere/AppServer/profiles/Dmgr01/conf/cells/TradeCell/resources.xml

![Fig. 2: The maximum weighted matching for two path token sets.](image-url)
The maximum weighted matching of these two paths is shown in the figure. The numbers on the edges are the path token similarity scores. If we set \( d = 0.3 \) for the similarity threshold, every path token pair in the matching excerpt (“appsrv”, “dmqr”) will be included the Path Token Base. Note that when the sizes of the two token sets are not equal, the maximum matching guarantees only the most similar token pairs get selected. For example, the path tokens “node” and “nodes” in \( T_1 \) are not included in the matching as they have very low similarities with the path tokens in \( T_2 \).

3) Path token sequence

With the fuzzy matching method, we are able to select abundant representative path tokens to fill in the Path Token Base. However, a path pattern usually not only includes tokens but also sequences of tokens. Moreover, the order of the tokens in the sequences is often strongly indicative of the locations of configuration files. For example, “[config cells TradeCell resources]” is one of the token sequences appearing many times in the paths of the “JDBCDataSourceName” configuration files. If one file has this sequence in its path while another file has the four tokens in this sequence separately shown in its path, it is more likely that the first file is one of the configuration files containing the item “JDBCDataSourceName”. Therefore, instead of just taking single path tokens, we further keep token sequences in the Path Token Base. Note that a path token is nothing but a path token sequence which contains only one token element.

A path token sequence is formed only if the included tokens were neighbors in the original file path. For example, the path “/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/config/cells/vm81Node01Cell/nodes/vm81Node01/resources.xml” can be divided into three token sequences which contain more than one token: “[opt ibm websphere appserver profiles appsrv]”, “[config cells]”, and “[cell nodes]”.

4) Path token sequence properties

The fact that different path token sequences have different levels of significance when we differentiate the entries of a Path Token Base. Here are some observations from the positive training files:

- A path token sequence tends to be more important if it appears more frequently.
- A path token sequence tends to be more important if it contains more tokens in it. For example, a token sequence “[opt ibm websphere appserver profiles cells]” tells much more information about the training files’ location than a single path token “opt” does.
- A path token sequence tends to be more important if it locates at the end of a path, rather than the beginning of a path. For instance, the token sequence “[opt ibm websphere cells]” has relatively limited indication of the training file locations because it is usually close to the root directory and too many files can be under this directory. On the other hand, the token sequence “[config cells TradeCell resources]” is closer to the bottom subdirectory and only a small number of files will be found in this directory, which is more likely the location of the training files.

Based on these observations, three properties are defined for each path token sequence.

- Frequency: the number of a token sequence appears in the token base, normalized by the total number of token sequences.
- Length: the number of tokens a token sequence contains.
- Position: the position of the first token of a token sequence, normalized by the length of the token sequence. When a token sequence appears multiple times in the token base, the average value will be used.

The production of these three property values becomes the weight of a token sequence. It differentiates the contributions of the token sequences in the Path Token Base.

5) Matching to Path Token Base

With the extracted entries and their weights, the population of a Path Token Base is completed. Using this Path Token Base, it is possible to match the path of a new file to it and quantify the matching result as a measurement of how similar the new file is to the file class represented by this Path Token Base in terms of the file path.

Algorithm: findPathTokenSequenceMatching(path, T, c)

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>path</td>
<td>a Path Token Base</td>
</tr>
<tr>
<td>T</td>
<td>an ordered path token set</td>
</tr>
<tr>
<td>c</td>
<td>a matching result cache (a hash map)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>r</td>
<td>the matching result of ( T ) in ( \text{path} )</td>
</tr>
</tbody>
</table>

```plaintext
1 n, l = length(T)
2 while i > 0
3 for i = 0 to n-(l-1)
4 lTI = l
5 rTI = n-(i+1)
6 divide T into three ordered token sets: \( \text{lr}, \text{rr} \), \( \text{rt} \)
7 lr = findPathTokenSequenceMatching(path, IT, c)
8 rr = findPathTokenSequenceMatching(path, RT, c)
9 if mT in c
10 \( mr = mT \)’s value in c
11 else
12 for every path token sequence seq in path
13 \( s = \) computePathTokenSimilarity(seq, mT)
14 \( s = s * \text{path.getWeight(seq)} \)
15 \( mr = \) maximum value of \( s \)
16 put \( mr \) and \( \text{seq} \) into c
17 ss = lr + rr + mr
18 \( r = \) maximum value of \( ss \)
19 \( l-- \)
```
Fig. 3: The fuzzy matching algorithm for matching a path to a Path Token Base.

The matching process is as follows: the new file’s path will first be tokenized into a set of tokens. Then a fuzzy matching algorithm shown in Fig. 3 is used to explore every possible token sequence combination of this path in order to find the best match in the Path Token Base that yields the highest similarity score. This similarity score serves to represent this file in the path dimension and is used for classification later.

4.3 REPRESENTATION OF FILE EXTENSIONS

With a similar idea but much simpler, we construct an Extension Token Base to represent the information carried in the file extensions as the entries, with the frequencies of appearance as entries’ weights. When comparing an extension to the Extension Token Base, if a match is found, the weight of the matched entry will be returned as the similarity score to represent the file in the extension dimension, otherwise the matching result is zero.

4.4 DATA IMBALANCE HANDLING

As stated in the last section, the negative instances greatly outnumber the positive instances for each file class in our datasets, because for a specific configuration item there are only a very limited number of files containing it while the rest files on the same server are all irrelevant. Classifiers produced by standard machine learning algorithms usually have very poor performances when dealing with this kind of imbalanced datasets. These classifiers are designed to achieve maximum accuracy, and therefore are heavily biased towards the majority class since the minority class contributes relatively little to the overall accuracy.

We examined three types of techniques to re-establish the balance between the two classes in the training set: increasing the number of minority class instances, namely oversampling [9], decreasing the number of majority class instances, namely undersampling [17], and assigning unequal error penalties to the two classes. Note that the effect of adopting different penalties is in fact equivalent to resampling the dataset in a sense that they both aim at moving the decision boundaries of the classifier towards the majority class.

For oversampling, we implemented the well-known SMOTE algorithm [20] to enlarge the positive instance class by creating “synthetic” data points using the \( k \) (in our case \( k = 4 \) ) nearest neighbors. For undersampling, we developed a straightforward algorithm that shrinks the negative instance class by removing those data points that are far from the positive data points in terms of Euclidean distance. For the third approach, we applied a larger SVM class weight to the positive instance class such that when a decision error happens to a positive data point it would have a larger penalty. In the next section, we will show experiments studying the effects of these three different techniques applied on our imbalanced datasets.

5. EXPERIMENT

5.1 EXPERIMENT ENVIRONMENT

For experiments, we selected 6 web applications (e.g. DayTrader, Wordpress, etc.) and migrated them from vCenter-based cloud to OpenStack-based cloud. Table 3 describes the detail information of the experiment environment. These applications have been deployed on the 31 virtual machines and running on a popular software stacks including RHEL and SUSE® as the Operating System, WAS and Apache Tomcat as application servers, Apache HTTP Server as the web server, DB2® and MySQL® as the Database servers.

<table>
<thead>
<tr>
<th>Application</th>
<th># of servers</th>
<th># of Total Files</th>
<th>Software Stack</th>
</tr>
</thead>
<tbody>
<tr>
<td>DayTrader</td>
<td>5</td>
<td>2,634,212</td>
<td>RHEL IHS+WAS DB2®</td>
</tr>
<tr>
<td>Wordpress</td>
<td>3</td>
<td>1,023,831</td>
<td>SUSE® Tomcat MySQL®</td>
</tr>
<tr>
<td>App 1*</td>
<td>10</td>
<td>9,023,214</td>
<td>IHS+WAS DB2®</td>
</tr>
<tr>
<td>App 2*</td>
<td>2</td>
<td>502,324</td>
<td>RHEL Tomcat MySQL®</td>
</tr>
<tr>
<td>App 3*</td>
<td>5</td>
<td>3,376,981</td>
<td>RHEL IHS+WAS DB2®</td>
</tr>
<tr>
<td>App 4*</td>
<td>6</td>
<td>3,920,398</td>
<td>RHEL IHS+WAS DB2®</td>
</tr>
</tbody>
</table>

5.2 DATASET POPULATION

In this paper, we representatively selected configuration files containing 7 types of configuration items from both commercial enterprise software and open source software listed in Table 4. We applied the clustering-based configuration file labeling to identify the configuration files from the experiment environments and took them as the positive instances. All the other files were considered as the negative instances. We randomly divided the whole dataset into two sets of approximately the same size, and used them as the training set and the test set, respectively. As seen in Table 4, the imbalance ratios (ratio of the number of negative instances and the number of positive instances) are on the order of 10³.

<table>
<thead>
<tr>
<th>Configuration Item</th>
<th>Training Set Size</th>
<th>Test Set Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>WASCellName</td>
<td>Positive: 20,380,93</td>
<td>Negative: 1852812.5</td>
</tr>
<tr>
<td>HostName</td>
<td>Positive: 20,380,92</td>
<td>Negative: 1852812.5</td>
</tr>
<tr>
<td>JDBCDataSourceNa</td>
<td>Positive: 20,380,94</td>
<td>Negative: 20,380,94</td>
</tr>
<tr>
<td>ApacheMaxClients</td>
<td>Positive: 20,380,93</td>
<td>Negative: 20,380,93</td>
</tr>
<tr>
<td>PHPMemoryLimit</td>
<td>Positive: 20,380,94</td>
<td>Negative: 20,380,94</td>
</tr>
<tr>
<td>TomcatPort</td>
<td>Positive: 20,380,93</td>
<td>Negative: 20,380,93</td>
</tr>
</tbody>
</table>

5.3 EVALUATION METRIC

Due to the high imbalance in our datasets, using accuracy as the evaluation metric is simply inadequate. This is because, for example, for a test set with 99 negative instances and 1 positive instance, a classifier will be 99% accurate even if it classifies everything as negative. Obviously this classifier will not perform well on a more general test set. We followed many research works using sensitivity and specificity as the evaluation metrics, where

\[
\text{sensitivity} = \frac{\text{true positive}}{\text{true positive} + \text{false negative}}
\]
and

\[
\text{specificity} = \frac{\text{true negative}}{\text{true negative} + \text{false positive}}
\]

(4)

Note that sensitivity is nothing but the accuracy on the positive instances while specificity is nothing but the accuracy on the negative instances. We also used a single metric called Balanced Accuracy (BAC) that integrates these two numbers:

\[
\text{BAC} = \frac{\text{sensitivity} + \text{specificity}}{2}
\]

(5)

5.4 LEARNING ALGORITHM

As a widely used machine supervised learning algorithm, an SVM model can efficiently perform both a linear and a non-linear classification, which is very suitable for our task. We used LIBSVM [9], a popular open source SVM library in all of our experiments.

5.5 INFLUENCE OF THE PATH TOKEN SIMILARITY THRESHOLD

As mentioned above, a similarity threshold is used in the Path Token Base population step to control how many tokens should be taken into the base. The smaller is, the more tokens will be included from the maximum matching process. In order to see how influences the classifier’s performance, we trained a classifier on each of the 7 datasets with different \( \epsilon \) values, and showed the results in Fig. 4. Note that we did not deal yet with the data imbalance in this experiment.

![Fig. 4: Predicted BAC with respect to different \( \epsilon \) values on each dataset](image)

Fig. 4: Predicted BAC with respect to different \( \epsilon \) values on each dataset

From Fig.4 we can see that for most of the datasets the BAC value did not change much as \( \epsilon \) increases from 0.1 to 0.9. This was because during the Path Token Base population process, the two tokens in most token pairs were highly similar to each other such that the increase of did not filter out many pairs. For example, for the dataset “JDBCDataSourceName”, when \( \epsilon = \frac{1}{2} \), the Path Token Base received 28 unique path token sequences as entries from the training set. When \( \epsilon = 0.1 \), the number of unique entries in the Path Token Base just dropped to 24. The small changes of the number of entries caused a small change of the classification performances.

Although the influence of \( \epsilon \) was limited for some datasets, we could still see that 0.3 was a value that worked best for most of the datasets. Therefore we chose this value as \( \epsilon \) for the future experiments.

5.6 EFFECTIVENESS OF THE FILE METADATA FEATURE REPRESENTATION

In order to prove the effectiveness of our method of feature representation of the file metadata, especially how we transform the file path and file extension into numerical values, we formulated a baseline classification method which directly stored the entire file paths, performed exact string matching, and used the frequency of the matched path as the numerical value to represent the file path dimension. Note that the baseline method characterized the file paths instead of the file names because, as mentioned in the Introduction section, both configuration files and other types of files (e.g. backup files) can have the same file name and the file name match would therefore introduce errors. Fig. 5 presents the results of the comparison between this baseline method and our MetaConf. Note that we still had not touched on the imbalance issue in this experiment. Both methods directly used the SVM classifier.

![Fig. 5: BAC of the baseline method and MetaConf without handling data imbalance on each dataset](image)

Fig. 5: BAC of the baseline method and MetaConf without handling data imbalance on each dataset.

As seen in Fig. 5, among the 7 datasets MetaConf was able to significantly improve the classification performances for at least four of them. There were two datasets for which the baseline method and MetaConf achieved identical outcome. As we analyzed the experimental results, we discovered that the two methods tend to produce similar outputs if the positive file instances are highly consistent. In other words, in this case all the positive files have almost the same paths, extensions, sizes, etc. With this lack of variety in the training data, there will be no much difference between exact matching and fuzz matching, and that is why MetaConf did not outperform the baseline method. However, in other datasets where there are a greater variety of patterns in the file path, MetaConf was able to capture the patterns in a more flexible and accurate fashion, and thus produce better classification results.

5.7 EFFECTIVENESS OF THE DATA IMBALANCE HANDLING
HP DDM [12], EMC ADM [17], and Evolven [11] utilize the product specific scripts or sensors by sending them to the running systems to capture application configuration files and configurations. In addition, the academic efforts also apply these script-based approaches in discovering configuration files and configurations. Galapagos [15][16] used a model-based approach and product specific scripts to discover the configuration files and configurations from data centers. These approaches highly depend on the embedded domain specific discovery techniques developed by the domain experts. It limits the scalability of the products or tools based on these approaches. The rule-based approaches apply the rules to annotate configuration files and configurations from the target servers. PoVMiner [22] applied pre-defined rules to annotate the points of variability (PoVs) and uses a Bayesian network to estimate the confidence for annotated PoVs. The human interactive approaches allow end users to interact with the system to define patterns to match configuration files and configurations or specify them directly. All of these approaches require rich domain expertise and high familiarity with the software and system configuration, while MetaConf can discover these configuration files with much less human interaction based on the nature of self-similarity in the metadata of configuration files.

MetaConf solves the configuration file discovery problem by converting it to a learning based file classification problem. While there is a large body of work on file classification using supervised machine learning algorithms, most of these work focus on examining the binary file contents and then extracting the learning features based on the contents. Few research pays attention to the file metadata as MetaConf does. One example is [19], which developed an automatic learning method to classify files in self.* storage systems using file access pattern, lifetime, and size.

To represent the file path and the file extension, MetaConf utilizes a fuzzy token matching method to convert the path and extension strings to numerical values as the learning features. The key to this process is to recognize similar strings in a flexible manner. There is a great amount of research work on the topic of fuzzy string matching, and these work have been mostly concentrating on extending some simple similarity metrics such as Jaccard similarity and edit distance such that the matching can be less constrained and more flexible. [14] proposed a fuzzy token matching based similarity metric and used it to efficiently solve the string similarity join problem. [20] enhanced the TF-IDF based cosine similarity such that the matching could be performed efficiently along multiple strings.

7. CONCLUSIONS
In this paper we proposed a supervised learning based approach, MetaConf, to automatically perform configuration file discovery, which has been currently dominated by manual or semi-automatic methods that heavily depend on human expertise and labor. We discussed a novel learning-based approach to discover the configuration files with their metadata. We developed an effective keyword search, variable point replacement, and clustering-based approach to populate the labelled training data. We designed a fuzzy token matching based feature extraction schema to numerically represent the paths and the extensions of configuration files, and used them together with other file metadata as the learning features. We also implemented three different techniques to successfully solve the data imbalance issue that is inevitable in configuration file discovery. Experiments with real-world datasets were conducted to validate the effectiveness of each component of our system. It also showed that MetaConf outperformed our baseline method on our datasets.

In the future, we plan to conduct more analysis and experiments to distinguish the different contributions of each property of the path token sequences to the system overall performance, so that we can identify possible improvements. We are also working on extending the configuration file discovery to finer-grained configuration item mining using a similar methodology in order to meet application-centric cloud migration requirements.

8. REFERENCES


Authors

**Fan Jing Meng** is a Research Staff Member in IBM Research - China. She received her Ph.D. degree from BUAA(BeiHang University) and joined IBM in 2004. She has been working in various research areas including the model-driven business transformation, domain knowledge management, project portfolio optimization, workload migration to cloud. Her current research interests include the configuration discovery and analytics, IT operations analytics, and DevOps analytics.

**Han Wang** is currently a Ph.D. student at Tetherless World Constellation in Rensselaer Polytechnic Institute. His research focus is knowledge discovery and information extraction with natural language processing and machine learning techniques.

**Xuejun Zhuo** is a Senior Software Engineer in Linkedin. She received the BE degree in computer science from Huazhong University of Science and Technology in 2007 and the PhD degree in computer science from Tsinghua University in 2012. She also worked at IBM China Research Lab 2012-2013 as a Staff Researcher and Pattern Insight 2014-2015 as a Research Scientist. Her research interests include cloud computing, opportunistic mobile network, mobile social network, cloud computing, and network security.

**Jing Min Xu** is a senior technical staff member from IBM Research - China. He has been working in various research areas including XML/EDI commerce, model driven business transformation, Web service delivery platform, Cloud computing, etc. since he joined IBM in 1999. His recent research mainly focuses on Cloud application life-cycle management and leveraging analytics in both development and operational phases to drive application quality improvement, delivery pipeline optimization and operational efficiency. Some of his works have been incorporated into several IBM products and services. Jing Min has 13 granted patents and published about 20 papers in international journals and conferences.

**Lin Yang** is currently a senior research at IBM Research China. He received Ph.D. degree from Nankai University in 2010. In IBM, he worked on various projects in could computation, big data and IT operational analytics. He's current focuses are log analytics, application performance management and DevOps.

**Changsheng Li** is a Staff Researcher from IBM Research-China. He received his B.E. degree from the University of Electronic Science and Technology of China (UESTC) in 2008, and his Ph.D. degree in pattern recognition and intelligent system from the Institute of Automation, Chinese Academy of Sciences in 2013. He also studied as a research assistant in The Hong Kong Polytechnic University in 2009-2010. He joined IBM Research-China in 2013. His research interests include machine learning, pattern recognition, and data mining.
AN INVESTIGATION OF ANTICIPATED BENEFITS OF CLOUD COMPUTING ADOPTION IN AUSTRALIAN REGIONAL MUNICIPAL GOVERNMENTS

Omar Ali, Jeffrey Soar, and Jianming Yong
University of Southern Queensland
{Omar.Ali, Jeffrey.Soar, and Jianming.Yong}@usq.edu.au

Abstract
It is expected that computing services will increasingly be accessed as another utility a similar way to other services such as water, electricity, telephone or gas. Many organizations including government deliver their services through information and communication technology (ICT) tools including e-government. Cloud computing is a relatively new way of providing services over the internet. In this research in-depth interviews of Australian councils’ IT managers were conducted with the aim of providing insights into the perceived benefits of cloud computing adoption. Also, survey data from 480 IT staff across 47 local councils were collected to confirm the findings of the exploratory stage. The research indicated that cloud computing is seen to offer many benefits that are identified in the research literature; additional anticipated benefits emerged which related to reduced level of risk, remote access, reduce staff, and time efficiencies. The findings of this research may help IT managers and top management evaluate possible adoption by increasing their understanding about anticipated benefit which will encourage them when planning or decided to adopt cloud computing.

Keywords: Cloud computing, anticipated benefits, adoption, e-government, local councils.

INTRODUCTION
The implementation of information and communication technology (ICTs) strategies, such as electronic services, by organizations has allowed them to benefit both economically and competitively (DTI 2004; Pan & Jang 2008; Sultan 2010). Governments in particular are increasingly delivering their services electronically through electronic government systems (e-government) (Pyke 2009; Tuncay 2010). This simplifies the interaction between people and government organizations (Jansen 2005); and offers reliability, affordability and ease of maintenance (Jansen 2005; Smitha et al. 2012).

An efficient application of e-government is possible through cloud computing (Gosinski & Brock 2010; Tuncay 2010). It can save on the overall costs and is a novel method of providing services over the internet (Armbrush et al. 2010; Dillon et al. 2010; Leavitt 2009; Leimeister et al. 2010; Lyer & Henderson 2010; Salleh et al. 2012; Zhang et al. 2010). Several benefits are provided by cloud based e-government in the form of cost reductions, enhanced storage, security management, scalability and accountability (Hayes 2008; Mell & Grance 2009). Cloud computing may be the main way organization will manage information processing in the future (Grossman 2009; Marston et al. 2011; Melvin & Greer 2009; West 2011).

Research about the use of cloud computing in the public sector in general is limited (Janssen & John 2011). Some previous researchers have studied the benefits of cloud computing in a relation to the cost (Buyya et al. 2008; Buyya et al. 2009; Kondo et al. 2009), green technology (Baliga 2010; Marston et al. 2011; Tripathi & Parihar 2011; Zhang et al. 2010), infrastructure (Das et al. 2011; Lenart 2011; Marston et al. 2011). There is a lack of exploratory studies that provide an in-depth and holistic investigation of all actual anticipated benefits of adopting cloud computing (Low et al. 2011; Misra & Mondal 2011). That is, we could not find any studies that listed all benefits and explained why and how they are benefits.

Despite its potential benefits the adoption rate of cloud computing in regional municipal government sectors in Australia has been lower compared to urban areas (IT Industry Innovation Council 2011). The paucity of empirical studies about anticipated benefits of cloud computing adoption in Australian regional municipal governments has hindered understanding and thus strategy development to improve its adoption (IT Industry Innovation Council 2011). This situation has prompted regional municipal governments to request further research to guide their implementation decisions (Department of Innovation Industry Science and Research 2011). The current gap in the literature has led us to the following research problem: What are the actual benefits of adopting cloud computing to Australian regional municipal government.

The paper was provides an overview about the research topic and the literature. Then, explain the methodology used to collect data for this research involved in-depth interviews with IT managers in Australian regional councils and survey data from 480 IT staff across 47 regional councils. Finally, the paper discusses findings limitations and suggestions for future research.

CLOUD AND GOVERNMENT
A broad and beneficial perspective into the computing world is provided by the various computing paradigms such as, cluster computing, grid computing and cloud computing.
Cloud computing poses certain technical, and social challenges for e-government systems. Technical challenges include interoperability issues between existing IT infrastructure and cloud computing. Security issues relate to the protection of data from unauthorized access, and privacy. Trust issues related to the lack of customer trust and weak trust relationships. Social challenges including a lack of technical knowledge by the government organizations and the client.

By contrast, cloud computing offers several benefits to e-government, some of these benefits are as:

Protection, care and technical support: The service providers of cloud computing provide access to applications and data services. The uniqueness of cloud system pertaining to e-government services is that the system is efficient enough to sort out problems particularly for government departments outside of urban areas where recruitment of IT staff is more difficult. Cloud technology, makes it easier to upgrade software applications, located in a single system. Cloud service providers are accountable for upgrading software and providing technical assistance. Cloud technology can undertake computations, obtain software applications, and provide data access and storage to end-users without the need to know the physical location and configuration of the system that delivers the services. Many other benefits of cloud computing have been outlined in different studies such as; simplified cost and consumption model, faster provisioning of systems and applications, right size to address business changes, ease of integration, highly secure infrastructure, and compliant facilities and processes.

Disaster recovery: A disaster recovery system is essential, a government can maintain a backup of the server using a cloud system for the disaster recovery on a day-to-day basis and can store it off-site through implementing a third party storage service provider that holds the ability to store in a different location. Disaster recuperations schemes in cloud systems is a better choice compared to traditional disaster recuperation programmes because it can restore data in a more prompt and swift manner and because this swift recovery reduces the cost of the operation.

Old technologies and migrating to new technologies: Some of the functions of data centres for e-government include the ability to implement diverse versions of the software, programs and security packages. But changing an out-dated technology to a highly sophisticated one has traditionally been a complicated task. By contrast, cloud computing does not require upgrading from one version to another because multiple versions of the software can be operated simultaneously. This system can therefore offer greater flexibility and efficiency for e-government.

Policies management: E-government applications need to be in compliance with governmental policies. In order to increase efficiency in daily performance, these policies need to be implemented in unison with the infrastructure and data centres. Cloud architectures can assist with compliance with policy in data centres.

Promoting business development: Benefits can be obtained from cloud computing in order to improve businesses by lowering the overall cost of investment in ICT infrastructure. Cloud computing allows users to undertake computations, obtain software applications, and provide data access and storage to end-users without the need to know the physical location and configuration of the system that delivers the services. Many other benefits of cloud computing have been outlined in different studies such as; simplified cost and consumption model, faster provisioning of systems and applications, right size to address business changes, ease of integration, highly secure infrastructure, and compliant facilities and processes.

Reduced IT infrastructure cost: One of the major benefits that businesses are expecting from using cloud services is cost saving. Cloud computing provides almost direct access to shared computing resources and small and start-up businesses can launch new operations quickly with little to no upfront capital investment; this will assist with a faster time to market in many businesses. Using software from the cloud will lead to a reasonable reduction in systems maintenance and updating requirements. Clients will be able to reduce software updating and maintenance costs, by having most of the IT software, operations and functions undertaken by a third party.

Ease of use and flexibility: The interfaces of cloud applications use browser web based applications or windows based applications. Both interfaces tend to be intuitive and easy to use. Many cloud computing suppliers offer more flexible contract terms, which encourages firms to implement cloud services as needed to expand their businesses.
computing, there is the portability and accessibility feature, as the Internet is considered the backbone of the utilization idea, through which computing services are provided for clients through an active Internet connection. On-demand access to any application can be at any time from any location, provided the client has network access (Lanman et al. 2011). This can assist small businesses, which have a wide market and broad horizontal company operations, such as regional or international, to decrease external costs and make them less location dependent. Unfortunately, perceived complexity hinders adoption and realisation of benefits.

**Methodology**

The data collection of this research was conducted in two major phases. Phase 1 involved a qualitative investigation to understand the significant anticipated benefit of cloud computing adoption in Australian regional municipal governments. Phase 2 involved the use of quantitative questionnaire data to confirm the findings from the phase 1. The next two subsections describe the methods and results from two phases in detail.

1.1 Qualitative Investigation (Phase 1)

The research reached the saturation level within the research topic (Carson et al., 2001). A total of 24 interviews were carried out with IT managers of the chosen councils. The research reached the saturation level within the interview number 18, when the researcher noticed that, there is no more new information or patterns in the data emerging from the interview. Another six interviews were conducted to ensure inclusion of all segments and size classification of the councils to obtain a comprehensive overview of issues (refer to table 1). Only 21 interviews were used in this study. Three interviews were excluded from the analysis because it was discovered during the interview that these three IT managers did not come from an IT background and did not have any experience or knowledge related to cloud computing.

The first phase of this research is exploratory in nature seeking to investigate and provide a qualitative overview of the concepts relating to the anticipated benefits from the adoption of cloud computing in Australian regional councils.

**Data collection method.** A series of in-depth interviews were conducted between May 13, 2014 and August 12, 2014. These obtained inputs from 24 local government employees at senior management levels: IT Manager (10); IT Coordinator (4); Technical Director (2); Information Service Manager (2); IT Officer (1); IT Consultant (1); IT Network Manager (1); Chief Information Officer (CIO) (1); Enterprise Architecture Manager (1); and Team Leader ICT Operation (1). These occupational groups were selected based on the assumption that they represent key stakeholder groups likely to be responsible for planning and adoption of cloud computing for regional municipal governments.

The sample reflects the geographical spread and size classifications of regional municipal governments throughout Queensland (Coastal – 29%; Resource – 14%; Indigenous – 10%; Rural/Remote – 29%; South East Queensland – 18%) (See Table 1).

Table 1. Size classification

<table>
<thead>
<tr>
<th>Segments</th>
<th>Extra small</th>
<th>Small</th>
<th>Medium</th>
<th>Large</th>
<th>Very large</th>
<th>Total</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coastal</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>6</td>
<td>25%</td>
</tr>
<tr>
<td>Resource</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>3</td>
<td>12%</td>
</tr>
<tr>
<td>Indigenous</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>17%</td>
</tr>
<tr>
<td>Rural/Remote</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>7</td>
<td>29%</td>
</tr>
<tr>
<td>South East Qld</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>17%</td>
</tr>
<tr>
<td>Total</td>
<td>24</td>
<td>100%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Source: Research data

To improve the reliability of this research, the process explained by Kirsch (2004) was followed for collecting data. This process defines a set of procedures: firstly, identify and select the research issues, secondly, determine who to interview, and finally, determine how the interviews will be conducted.

The interviews lasted between 30 and 50 minutes. The interview questions were designed as open-ended questions to encourage the interviewees to provide answers that revealed their attitudes and perceptions relating to the research topic (Carson et al., 2001). A total of 24 interviews were carried out with IT managers of the chosen councils. The research reached the saturation level within the interview number 18, when the researcher noticed that, there is no more new information or patterns in the data emerging from the interview. Another six interviews were conducted to ensure inclusion of all segments and size classification of the councils to obtain a comprehensive overview of issues (refer to table 1). Only 21 interviews were used in this study. Three interviews were excluded from the analysis because it was discovered during the interview that these three IT managers did not come from an IT background and did not have any experience or knowledge related to cloud computing.

**Data analysis.** The interviews data was analysed using manual content analysis method (Miles & Huberman 1984), and using Leximancer. Manual content analysis was undertaken as a first step in the analysis which included three concurrent flows of activities: data reduction, data display and conclusion drawing/verification (Faust 1982; Hsieh & Shannon 2005; Miles & Huberman 1984). After the completion of each interview session, the recorded interviews were immediately transcribed. Interview transcripts were reviewed to create summary sheets for every interview (Rao & Perry 2007). This summary sheet included main themes, issues, problems and brief answers to each question, resulting in an overall summary of the main points in the contact (Patton 2002; Schilling 2006). Then the summary sheets were reviewed to develop a pattern code for the research data. The next step of the analysis was to develop data display, which organised assembly of information to permit the researcher to draw conclusions and taken actions (Miles & Huberman 1984). Once manual coding was completed, the data was then reanalysed using Leximancer to improve the reliability of the findings (Middleton et al. 2011; Smith & Humphreys 2006).

Leximancer is a data mining tool that can be used to analyse the content of collections of textual documents and
to visually display the extracted information (Smith 2003). It uses ontological relativity and dynamics to assemble bits of information to structure and evaluate concepts (Cummings & Daellenbach 2009). Words are combined to form concepts (thematic analysis) and identify relationships (semantic analysis) between concepts. A ‘concept map’ displays the main concepts in the text data, depicting the relationships through visual summaries of concepts and their co-occurrences – similar to a mind map (Cummings & Daellenbach 2009). Combined use of both manual and software analytical approaches provided a robust basis for clearly delineating concepts, themes and aggregate dimensions (Middleton et al. 2011; Smith & Humphreys 2006).

**Phase 1 Findings.** After the interview, the anticipated benefits of the adoption of cloud computing was given a rating of Important, Unimportant, Not sure by the respondents. The way the impact is rated will be in accordance to how the informants responded to them along with what was derived and collected from empirical sources. These rates were checked and accepted by the respondents after we sent our interviews’ findings to them. This allows us to categorize the cost impact according to their importance of impact for Australian regional municipal governments and present them accordingly. For more details see Table 2.

<table>
<thead>
<tr>
<th>Anticipated Benefit</th>
<th>Freq.</th>
<th>%</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>Provide better services</td>
<td>15</td>
<td>71%</td>
<td>“We get the benefit of regular maintenance and update so you do not have an aging solution, maintaining currency can be an issue for council with the upgrades. And we suffer the service solution that all happens in the background of the main equipment. So that might be a benefit with the intensive functionality connected them previously” (C52-UFM).</td>
</tr>
<tr>
<td>Cost reduction</td>
<td>13</td>
<td>62%</td>
<td>“The major benefits to move to a cloud is, obviously, is going to be cost savings because you do not have to invest in big data centres, in any infrastructure, in the people to maintain that infrastructure or support the infrastructure and maintain the infrastructure, well like upgrading and depreciating” (C34-UFV).</td>
</tr>
<tr>
<td>Reduce IT infrastructure</td>
<td>12</td>
<td>57%</td>
<td>“The best benefits of adopting cloud computing in the organisation is reduce IT infrastructure” (C72-URS).</td>
</tr>
<tr>
<td>Remote access</td>
<td>8</td>
<td>38%</td>
<td>“I guess one of the benefits is that to try and get the level of skilled IT workforce in remote and rural areas is significantly harder to get than in a capital city so I can see that would be a benefit of cloud computing to leverage those issues with getting those skilled workers to those regions” (C28-URS).</td>
</tr>
<tr>
<td>Disaster recovery and Backup</td>
<td>7</td>
<td>33%</td>
<td>“The ability of vendor to be able to look after the security side for things such as back-up from disaster recovery from rural and regional area. Back to the supplier, who would carry them out and as a part of the agreement, we need to make sure that they have proper disaster recovery mechanism in place as far as back-up and restore” (C61-URM).</td>
</tr>
<tr>
<td>Flexibility</td>
<td>5</td>
<td>24%</td>
<td>“One of the greatest benefits of cloud computing is the ability to access your data from anywhere and generally speaking in different devices as well” (C21-RTX).</td>
</tr>
<tr>
<td>Availability</td>
<td>5</td>
<td>24%</td>
<td>“Availability, it can be improved by having a probably architectured and redundancy solutions and possibly speed too” (C39-URM).</td>
</tr>
<tr>
<td>Reduce staff</td>
<td>5</td>
<td>24%</td>
<td>“The other benefits would be that you would not need certain specialized people internally as in, possibly, database administrators, possibly network administrators. That could possibly reduce some costs” (C11-RAV).</td>
</tr>
<tr>
<td>Time efficiencies</td>
<td>4</td>
<td>19%</td>
<td>“All the benefits come around being able to enter and use the software as a service and infrastructure as a service, that quick response bring the system quickly without spending much time or wait for service to arrive and installation of the processes” (C61-URM).</td>
</tr>
<tr>
<td>Reduce level of risk</td>
<td>2</td>
<td>10%</td>
<td>“Basically, by moving stuff in the cloud, the biggest benefit by far is that there is reduced…or the risk is moved away from council in that we do not have to deal with the risk as such” (C15-RAL).</td>
</tr>
<tr>
<td>Storage capacity</td>
<td>1</td>
<td>5%</td>
<td>“The amount of data that we generate now, storage is becoming a problem and cloud computing obviously can relieve that and relieve council of the financial burden to provide that” (C25-RTM).</td>
</tr>
</tbody>
</table>

Source: Research data

**Comparative analysis.** As stated in the methodology, the interview data was reanalysed using Leximancer to enhance the reliability of the findings from the manual content analysis (Middleton et al. 2011; Smith & Humphreys 2006). The first step it focused on the wide range of business-related words used by the respondents and identified from the exploratory Leximancer analysis. The second step for analysing the data was to examine the thematic groupings. Leximancer uses a natural language processing algorithm, so the theme is titled by the concept with highest prominence in the thematic aggregation. In this analysis, Leximancer clustered the concepts into six
themes namely: cloud; benefits; data; hardware; need; and moment), each theme aggregating two or more concepts and represented by labelled circles as they have been illustrated in Figure (1). Figure (1) illustrates the IT managers’ views of anticipated benefits of the adoption of cloud computing in regional municipal governments. This figure depicts the central theme within the map was ‘cloud’, and being strongly linked to the themes benefits, data, and hardware. The dominate theme cloud has strong associations with most other concepts on the map. Cloud is multifaceted in its use: relating to infrastructure, services, cost, systems, ability, rural regional, people, time, computing, and community. The concepts community, computing, benefits, cost, service, and infrastructure are shown to be frequently occurring and strongly connected to the theme cloud. Other themes illustrated but not connected to the theme ‘cloud’ include ‘need’ and ‘moment’. The centrality of this theme provides a starting point for the research analysis.

Because this research concentrated to find out the anticipated benefits of the adoption of cloud computing in the regional and municipal governments, the theme ‘benefits’ which contains the concept ‘benefit’ links strongly to the findings within the manual content analysis that suggested that IT managers saw cloud computing as having anticipated benefits on their organizations as shown in Figure (1).

The concept ‘benefit’ and its linkages on the concept map, through the analysis, have been illustrated through Figure (1). This concept is linked to all other concepts on the map. These linkages are to be expected with ‘benefit’ being the top ranking concept. The strongest linkages shown in Figure (1) are: (a) between benefit and service, (b) between benefit and cost, (c) between benefit and disaster recovery, (d) between benefit and time, (e) between benefit and infrastructure. These strengths are expected due to the focus of the research study and the qualitative questions asked, which were related to the anticipated benefits of cloud computing adoption.

Through the analysis of the data it is clear that in the discussion of benefits a number of aspects are addressed by IT managers. These aspects include benefits of cost, infrastructure, services, disaster recovery, ability, staff, time, rural, regional, hardware, systems, software, and councils. After having a comparison between the results from Leximancer and the manual analysis, it was found by the researchers that both the methods gave the same result in a relation to the anticipated benefits of the adoption of cloud computing in regional municipal governments.

1.2 Questionnaire Survey (Phase 2)

The second phase of this research is confirmatory in nature seeking to confirm the findings from the exploratory stage relating to the anticipated benefits from the adoption of cloud computing in Australian regional councils.

Data collection. A questionnaire was selected as the instrument for the second phase of data collection in this research. A questionnaire provides quick, affordable, efficient, and relatively accurate means to procure data to fulfill several goals (Zikmund 2003; Zikmund et al. 2013). The questionnaire was developed based on the previous literature on technological and organizations studies and the findings from qualitative study (exploratory stage), we developed the questionnaire to empirically test the research question. Feedback on the initial questionnaire was obtained from six IT managers of local councils. Minor modification were made based on the comments received.

An important step for the improvement of the effectiveness of the questionnaire is to do a pilot study (Shaughnessy et al. 2012). A pilot study includes actually running the questionnaire to a similar sample of respondents, under the same conditions to those anticipated in the final running of the survey (Shaughnessy et al. 2012). Running a pilot study before the final one is the best way to explore and identify issues and improve the design of the research survey (Waters 2011). The survey was pretested by 30 IT managers, nine curtailed surveys were discarded and entire of 21 surveys that were submitted with a 70% response rate.

An online survey method was chosen because of the accessibility of the internet for all intended participants and the belief that participants would prefer this approach. In order to make the survey available 24/7, an online survey service provider was found and the online survey link was offered for 3 months from March 1, 2015 to May 31, 2015.

Data analysis. Queensland local councils, are key organizations that provide public services to the citizens, community organizations and businesses. The 77 local councils in Queensland have great dependence on IT innovation to provide their services (LGAQ 2013). This research focuses on these local councils and in particular the IT department of these local councils as the main part of the target population. The survey was distributed online to Queensland’s 77 councils through USQ’s Ad hoc Survey System. IT Managers from 47 local councils responded to the survey which represented a response rate of 61% as shown in Table 3.
were in IT as operations/systems administrator/user support systems development/analyst/programmer roles, and 21% a management role (49.6%), followed by 28.8% were in knowledgeable level, which is computing, and total years experience related to cloud computing. The highest number of respondents (49.6%), followed by the knowledgeable level, which is computing.

The participating 47 local councils had around 786 IT staff who may have been invited to participate and 480 responded.

Table 3. Survey details

<table>
<thead>
<tr>
<th>Survey participant</th>
<th>No. of councils</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Survey received</td>
<td>47 Councils</td>
<td>61%</td>
</tr>
<tr>
<td>Survey not replied</td>
<td>30 Councils</td>
<td>39%</td>
</tr>
<tr>
<td>Total</td>
<td>77 Councils</td>
<td>100%</td>
</tr>
</tbody>
</table>

Table 4. Respondents' demographics

<table>
<thead>
<tr>
<th>Roles in the field of IT</th>
<th>Frequency</th>
<th>Percent</th>
<th>Cumulative %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Management</td>
<td>238</td>
<td>50 %</td>
<td>49.6%</td>
</tr>
<tr>
<td>Systems development/Analyst/Programmer</td>
<td>138</td>
<td>28.8%</td>
<td>78.3%</td>
</tr>
<tr>
<td>Operations/Systems administrator/User support</td>
<td>101</td>
<td>21%</td>
<td>99.4%</td>
</tr>
<tr>
<td>Other</td>
<td>3</td>
<td>0.6 %</td>
<td>100%</td>
</tr>
<tr>
<td>Total</td>
<td>480</td>
<td>100 %</td>
<td></td>
</tr>
</tbody>
</table>

Knowledge related to cloud computing

<table>
<thead>
<tr>
<th>Knowledge related to cloud computing</th>
<th>Frequency</th>
<th>Percent</th>
<th>Cumulative %</th>
</tr>
</thead>
<tbody>
<tr>
<td>No knowledge</td>
<td>5</td>
<td>1%</td>
<td>1.0%</td>
</tr>
<tr>
<td>Little knowledge</td>
<td>106</td>
<td>22.1%</td>
<td>24.2%</td>
</tr>
<tr>
<td>Some knowledge</td>
<td>111</td>
<td>23.1%</td>
<td>73.8%</td>
</tr>
<tr>
<td>Good knowledge</td>
<td>238</td>
<td>49.6%</td>
<td>95.8%</td>
</tr>
<tr>
<td>Excellent knowledge</td>
<td>20</td>
<td>4.2%</td>
<td>100%</td>
</tr>
<tr>
<td>Total</td>
<td>480</td>
<td>100%</td>
<td></td>
</tr>
</tbody>
</table>

Total years' of experience with IT

<table>
<thead>
<tr>
<th>Total years' of experience with IT</th>
<th>Frequency</th>
<th>Percent</th>
<th>Cumulative %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Never</td>
<td>12</td>
<td>2.5%</td>
<td>2.5%</td>
</tr>
<tr>
<td>Less than 1 year</td>
<td>95</td>
<td>19.8%</td>
<td>22.3%</td>
</tr>
<tr>
<td>2-5 years</td>
<td>250</td>
<td>52.1%</td>
<td>74.4%</td>
</tr>
<tr>
<td>6-10 years</td>
<td>111</td>
<td>23.1%</td>
<td>97.5%</td>
</tr>
<tr>
<td>11-14 years</td>
<td>8</td>
<td>1.7%</td>
<td>99.0%</td>
</tr>
<tr>
<td>More than 14 years</td>
<td>4</td>
<td>0.8%</td>
<td>100%</td>
</tr>
<tr>
<td>Total</td>
<td>480</td>
<td>100%</td>
<td></td>
</tr>
</tbody>
</table>

Source: Research data

The demographic information consisted of three items: role in the field of IT, knowledge in relation to cloud computing, and total years’ experience with cloud computing. Most of the participants were in an IT management role (49.6%), followed by 28.8% were in systems development/analyst/programmer roles, and 21% were in IT as operations/systems administrator/user support roles. Regard to their knowledge related to cloud computing, the results indicated that the highest knowledgeable level, which is “good knowledge”, was 238 (49.6%), followed by the knowledgeable level, which is “some knowledge”, was 111 (23.1). These percentages suggest that more than half of the respondents have a good knowledge related to cloud computing. The highest number of participants belongs to the years’ of experience (2-5 years), including 250 respondents (52.1%), followed by the years’ of experience (6-10 years), including 111 respondents (23.1%). These results indicate that most of the respondents have considerable experience related to cloud computing.

Phase 1 of this research come out with 11 main important benefits that identified by the participants based on their knowledge and experience as IT managers. Some of these findings already have been highlighted in the literature, and some of these findings are new. A question (on a seven-point scale) was included to evaluate the use of each of these benefits.

Findings and discussion. The results of phase 2 confirm that the adoption of cloud computing is perceived to provide better services to the organizations where 65 percent of the research participants identifying “strongly agree”. The research also confirmed that the adoption of cloud computing is perceived to reduce IT infrastructure with 74 percent selecting “strongly agree”. Seventy percent of the research participants selected “strongly agree” with statement that cloud computing reduced the level of risk. Sixty five percent selected “strongly agree” with statement that the adoption of cloud computing provided disaster recovery and backup and storage capacity to the organizations.

Provide better services. This research confirmed the literature findings that providing better services to the
stakeholders of organizations was the most significant benefit shared by the majority of the research participants. Figure 2 illustrates that nearly 65 percent of the research participants selected “strongly agree” with the statement that the adoption of cloud computing provided better services to the organizations. The benefit of better services entails many different positive effects that cloud computing technology may bring. Some examples are reduced risks for the organizations, having access over the data anytime and anywhere, and better management of services.

Cloud computing and its benefits have attracted strong interest amongst various public sector organizations (Saeed et al. 2011). Through employing cloud computing, government organizations and various public authorities can focus more on their core objectives of their business, instead of contemplating on the IT resource provision and maintaining the IT tasks (Saeed et al. 2011). By enabling government organizations services online the organization can improve the quality of service in subject to be timelier, richer in content and with greater availability (Al-Khouri 2011; Goel et al. 2012; Hashemi et al. 2013; Rastogi 2010; Sharma et al. 2012).

Cost reduction. The other benefit of cloud computing confirmed by this research was the expected cost reduction of the organizations especially once the cloud model has been perfected and maximized. Figure 3 demonstrates that about 58 percent of the research participants selected “agree” with the statement that the adoption of cloud computing reduced the cost. Participants indicated that cost reductions on operations by using cloud computing is anticipated. They believe that cloud computing bring a reduced use of physical hardware system which save a large portion of the financial resources for the organizations. Overall, participants indicated on how organizations will benefit by saving costs and managing their finances, which ensures productivity and sustainability in the long run.

The interest in cloud computing derives from the anticipated benefits (LGAQ 2013), it offers low starting expenses (Saeed et al. 2011; Saini et al. 2011; Jain & Bhardwaj 2010; Miller 2008). This financial benefit is expected mainly because of the usage-based pricing model. Start-up organizations in particular can use to help them to decrease their capital expenses and any hurdles to entry (Grossman & Gu 2009). Cloud computing provides almost direct access to shared computing resources and small and start-up businesses can launch new operations quickly with little to no upfront capital investment; this will assist with a faster time to market in many businesses (Lanman et al. 2011; Marston et al. 2011). Using software from the cloud will lead to a reasonable reduction in systems maintenance and updating requirements (West 2011).

Reduce IT infrastructure. This research confirms that the reduced of IT infrastructures by the organizations is one of the significant benefit of the adoption of cloud computing. This particular benefit is greatly connected to the reduced costs as discussed in the previous point. Figure 4 shows that 74 percent of the research participants selected “strongly agree” with the statement that the adoption of cloud computing resulted in reduced utilization of IT infrastructure.

There are several benefits in cloud computing and one of the prime benefits of this advanced technology, particularly with regard to government organizations services in which cloud computing plays a significant role in minimizing the IT infrastructure (Das et al. 2011). In other word, there is no other IT financing is needed in case of infrastructure, programming and support resources advancement (Beaubouef 2011; Sperling 2010).

Remote access. This research found that remote access is new important benefits that emerged upon the adoption of cloud computing. Figure 5 illustrates that 52 percent of the research participants selected “strongly agree” with the statement that remote access is one of the significant anticipated benefits of the adoption of cloud computing. Thirty four percent of the research participants selected “agree” with that. The participants indicated that by utilizing cloud computing, organizations then have access to the remote and rural areas which can in turn into greater advantages especially having a larger audience reach. The participants notice around the ability to move the requirement for supporting the backend infrastructure that is required of some systems out of the regional area and having that been able to be supported data centre in the
major cities, where is the support maintenance around that staff is a part of agreement. So, there is less in local skills, where there is a shortage of those skills within regional and rural areas, for them to be able to support.

Disaster recovery and backup. This research confirms that disaster recovery and backup is emerged as an important anticipated benefits of the adoption of cloud computing. As shown in Figure 6, this research found that around 64 percent of the research participants selected “strongly agree” with the statement that the adoption of cloud computing provided disaster recovery and backup to the organizations data. Twenty two percent of the research participants selected “agree” with that. Based on participants’ responses, disaster recovery and backup entails to the ability of cloud computing to function despite the unexpected issues and problems that may arise along the way. Having proper data backup can provide a quick recovery in unexpected cases and times.

Disaster recovery provisions are essential for the endurance and long-term existence of many organizations to make sure whether they hold the capability to survive at the proceedings caused by their IT infrastructure. Disaster recuperations schemes in cloud systems provide more choices when compared to traditional disaster recuperation programme in order to restore the data in a prompt and swift manner (Rajkumar et al. 2011). With regard to this recovery type, the overall cost can be reduced and consumes less time (Staten 2011). Government can maintain a backup of the server through employing cloud system as an efficient backup for the disaster recovery on a day-to-day basis and can store it off-site through implementing a third party storage service provider that holds the ability to store in a different location (Hashemi et al. 2013).

Flexibility. This research confirm that the concept of flexibility was emerged as an important benefits for the adoption of cloud computing. Figure 7 illustrates that 56 percent of the research participants selected “agree” with the statement that the adoption of cloud computing provided flexibility of the organizations systems. Twenty five percent of the research participants selected “strongly agree” with that. Flexibility was touched in the first major benefit founded by the researcher where in the stakeholders are given the opportunity to access their data anytime and anywhere.

Perceived complexity of the technology seriously hinders the increase in adoption rates and user satisfaction. Cloud computing, the operating interfaces of cloud applications look like browser web based applications or windows based applications. Both interfaces tend to be intuitive and easy to use (Melvin & Greer 2009). Most cloud computing suppliers offer more flexible contract terms, which encourages organizations to implement cloud services as needed to expand their businesses (Leavitt 2009). In addition to these significant characteristics of cloud computing, there is the portability and accessibility feature, as the Internet is considered the backbone of the utilization idea, through which computing services are provided for clients through an active Internet connection. On-demand access to any application can be at any time from any location, provided the client has network access (Lanman et al. 2011). This can assist small businesses, which have a wide market and broad horizontal company operations, such as regional or international, to decrease external costs and make them less location dependent.

Availability of services. Another connected benefit to flexibility is the availability of services. So, this research confirmed that the concept of availability of services within the organization was emerged as an important benefits for the adoption of cloud computing. Figure 8 shows that 52 percent of the research participants selected “agree” with the statement that the adoption of cloud computing provided availability of services within organizations. Thirty two percent of the research participants selected “strongly agree” with that.
Availability refers to the uptime of a system, a network of systems, hardware and software that collectively provide a service during its usage (Ahuja & Mani 2012). Could computing is a third party service and consumers heavily rely on the service providers for their computing needs. These computing needs range from research to businesses to high performance computing (Ahuja & Mani 2012). Technically there are several levels within cloud computing where high availability of services can be achieved. These levels include application level, data centre level, infrastructure level and geographic location level (Rackspace 2010).

**Reduce staff.** This research strongly confirm additional benefit of cloud computing would be staff reduction. Figure 9 demonstrates that 37 percent of the research participants selected “strongly agree” with the statement that the adoption of cloud computing reduced IT staff within the organizations. Twenty six percent of the participants selected “agree” with that, and other 18 percent of the participants selected “slightly agree”. But, 12 percent of the participants selected not sure if the adoption of cloud computing reduced IT staff within the organizations. The employment of reduced staff which may bring other advantages such as an organized management, cost reductions, and more. The adoption of cloud computing indicates that a reduced use of specialized people to maintain the organization is needed since cloud computing will then do much of the work upon adoption and transfer of data and information.

![Figure 9. Reduce staff](Image 71x337 to 212x397)

According to study by West (2011) one of the most important benefits of the adoption of cloud computing is reduce IT infrastructure. As a result of reduce IT infrastructure most of the IT software, operations and functions done by a third party. There will be fewer in-house IT staff and lower costs. On the other hand, some studies found that cloud computing does not remove the necessity for IT branch staff, on the grounds that clients still oblige access to the Internet and application configuration. Cloud computing permits IT administrator to focus on core business functions. As with any ICT operation, potential cloud computing adopters must be vigilant in testing their IT foundation and operations (Castellina 2011; Lenart 2011).

**Time efficiencies.** This research found that saving time or time efficiencies is new important benefit that emerged upon the adoption of cloud computing. Figure 10 shows that 40 percent of the research participants selected “strongly agree” with the statement that the adoption of cloud computing saved time of the operational process within the organizations. Forty one percent of the participants selected “agree” with that, and other 12 percent of the participants selected “slightly agree” with that. Within the adoption of cloud computing all the organization data is then stored in one or organized sets of location therefore time market will be a lot quicker and more effective.

![Figure 10. Time efficiencies](Image 324x343 to 464x403)

**Reduce the level of risk.** This research found that reduce the level of risk is new important benefit that emerged upon the adoption of cloud computing. Figure 11 illustrates that 70 percent of the research participants selected “strongly agree” with the statement that the adoption of cloud computing reduced the level of the risk. This can take effect as heightened protection and security is also expected once the cloud model works effectively for the organization and the stakeholders.

![Figure 11. Reduce level of risk](Image 324x578 to 464x638)

**Storage capacity.** This research confirm that the concept of storage capacity was emerged as an important benefits for the adoption of cloud computing. As shown in Figure 12, this research found that around 64 percent of the research participants selected “strongly agree” with the statement that the adoption of cloud computing provided storage capacity to the organizations. Due the increase of the amount of the data, storage capacity become a problem of most of the organizations.

![Figure 12. Storage capacity](Image 324x109 to 466x168)
Capacity includes increased computing power, improved performance, unlimited storage capacity, increased data safety, and fewer maintenance issues (Miller 2008). Many organizations fully utilize less than half of their total ICT resource capacity (Leavitt 2009), and most computing suppliers try to focus on the idea of offering computing services to their clients where they can scale up their capacity on demand (Grossman & Gu 2009). Whenever the client needs additional computing resources such as storage space, the provider can simply increase the provision accordingly in order to handle the increased business needs.

**Contributions**

This research contributes to the ICT technology adoption literature, by investigating the anticipated benefit from the adoption of cloud computing in regional councils. Looking at regional council’s adoption of new IS innovations can help enrich knowledge and understanding of the innovation adoption process in this era of rapid development of new technologies. This research leads to important practical implications for technology consultants. Regional councils represent organizations that provide services to the local citizen and the businesses in most economies, and consequently represent an important market segment for cloud service providers. Cloud service providers may need to improve their interaction with regional councils who are involved in the cloud computing, in an effort to create a healthy environment for cloud computing adoption, and to remove any vagueness surrounding this technology. The findings of this research may help IT managers and top management evaluate possible adoption by increasing their understanding about anticipated benefit which will encourage them when planning or decided to adopt cloud computing. Taking all the above into account, this research presents some useful information for organizations, technology consultants. This research is viewed as being relevant to the current era of rapid developments of cloud computing technologies.

**Limitations and Future Research**

There has not been much research done on cloud computing in reference to Australia. Future research could build on this research by investigating the anticipated benefit of cloud computing adoption in different sectors of the economy and industries. On a geographical dimension, this research was primarily limited to the regional councils in Queensland.

**Conclusion**

Cloud computing is a latest technological paradigm in IT world, related to the delivery of computing as a service. It has been proven that the application of cloud computing carries important benefits such as scalability and cost reduction. It also holds the advantages of maintenance, installation cost saving, and pay-as-you-go framework. Cloud computing as an exciting development is a significant alternative today’s local government sector. Employees and external users have the opportunity to quickly and economically access various application platforms and resources through the web pages on-demand. This automatically reduces the cost of organizational expenses and offers more powerful functional capabilities.

This research focuses on the anticipated benefits of cloud computing adoption in Australian regional municipal government environment. Since there is limited literature related to Australian regional municipal governments and cloud computing, The empirical research was done at Queensland local councils by employing in-depth interviews with the IT managers to investigate the anticipated benefits that related to cloud computing adoption. Also, survey data from 480 IT staff across 47 local councils were collected to confirm the findings of the exploratory stage.

The findings derived from this research have shown that cloud computing adoption in government organizations resulted in significant cost reductions, improved service delivery and reduced IT infrastructure. Furthermore, time-effective and convenient services were delivered to the public. From the points of benefits provided by cloud computing, there is a great result for local councils IT staff to take them away the responsibility of the maintenance burden in the councils. Adopting cloud network redundancy eliminates disaster recovery risks and its high costs. There can always be new tools and applications to improve IT features. These results validate the normative literature. In addition to this, since the core of Australian local government council’s initiatives includes information exchange, processing and service delivery, cloud computing seems to be an effective means of the next generation systems integration.

**Acknowledgments**

This research was funded through an Australian Postgraduate awarded at the University of Southern Queensland. The research was approved by the USQ Ethics Committee and was performed in accordance with the ethical standards as laid down in the 1964 Declaration of Helsinki and its later amendments or comparable ethical standards.

**Reference**


**Authors**

Omar Ali is PhD candidate in school of management and enterprise, Faculty of Business, Education, Art and Law at University of Southern Queensland, Australia. His PhD research is about the adoption of cloud computing in local government councils. He had two master degree. The first one in information and communication technology, and the second one in information systems and technology by research. His research interests include RFID technology; cloud computing; VoIP; IT governance; information system security; research methodology; and systems design and analysis.

Jeffrey Soar is professor in school of management and enterprise, Faculty of Business, Education, Art and Law at university of Southern Queensland, Australia. Also, he is a Chair in Human-Centred Technology. His research interests include Information technology; information system management; cloud computing; e-business; IT governance; health information management.

Jianming Yong is an associate professor in information systems, school of management and enterprise, Faculty of Business, Education, Art and Law at university of Southern Queensland, Australia. His research interests include advanced networking, e-business, data integration, information system security, network management, web service for SMEs, digital identity management.
Abstract
Cloud computing promises to transform applications and services on the web into elastic and fault-tolerant software. To aid at this target, various research prototypes and products have been already proposed. However, especially with respect to the design phase of cloud-based applications, such prototypes do not enable the appropriate composition of cloud services at different levels to realise not only the functionality but also the underlying infrastructure support for such applications. Moreover, most existing prototypes and products lack the appropriate semantics to guarantee that the respective design product is the most suitable and accurate one according to the various types of user requirements posed. To this end, this article proposes a semantic cloud application management framework that addresses the aforementioned issues by relying on ontologies to semantically describe cloud service capabilities and application requirements, on semantic cloud service matchmakers considering both functional and non-functional aspects as well as on a novel cloud service composition approach which is able to perform concurrently service concretisation and deployment plan reasoning, thus catering for the different levels involved in a cloud environment and their respective dependencies by also satisfying all types of user requirements posed. The service composition approach is experimentally evaluated deriving quite promising results indicating that the state-of-the-art is advanced.

Keywords: [cloud, service, composition, semantic, ontology, QoS, constraint programming, requirements]

1. INTRODUCTION
Cloud computing has revolutionized the deployment and provisioning of applications by promising an infinite amount of underlying and cheap resources to enable applications to scale at any type of demand. To this end, many major software, application and business process vendors have migrated their business to the cloud. Moreover, various research prototypes and commercial products have been proposed which enable application designers to discover the most suitable cloud services and assist in the cloud-based application deployment.

The services offered in cloud computing lie in different levels. There exist software services (SaaS), platform services (PaaS) and infrastructure services (IaaS). Different levels can provide support to different phases in the application life-cycle. The design of an application can rely on SaaS in order to have the means to realise the application functionality, while the application deployment can rely on PaaS and IaaS services. Moreover, for an application which exploits cloud services at different levels, its quality of service (QoS) depends on the respective quality and characteristics at lower-levels of abstraction. Thus, there are actually dependencies between the different levels which should be taken into account in an integrated and non-isolated manner.

However, the existing prototypes and products, which focus on the design and/or deployment phases, not only fail to consider such dependencies but also do not produce a design and deployment solution which is accurate and optimised according to the application requirements. The latter problem is mainly due to the lack of semantics in the description of the cloud services and requirements which then maps to their non-accurate discovery before they are actually composed.

To remedy the above issues, this article presents a semantic framework for the management of cloud-based applications. This framework relies on novel ontology-based language to describe application requirements at different levels, a semantic matchmaker able to discover services which accurate fulfill both the functional and non-functional requirements of the application and a cloud service composition component which solves a combined design and deployment optimisation problem by considering all possible cloud levels. The rest of the lifecycle phases are covered via components which attempt to enable the adaptive deployment and provisioning of the cloud application by building on existing research work and open-source software.

The cloud service composition component advances the state-of-the-art as, apart from composing cloud services at different levels, it exhibits the following features: (a) it considers unary and binary component placement constraints indicating one component's location or the relative location between two components either at the same VM or cloud, respectively; (b) it considers high- and low-level security requirements in terms of security controls and Service Level Objectives (SLOs), respectively; (c) it exploits non-linear functions able to map high- to low-level quality capabilities; (d) it even takes decisions on whether to use in-house software components or external SaaS for a particular application task; (e) it is able to address multiple objectives which span quality, cost and security metrics.

The cloud service composition component has been experimentally evaluated against baseline cloud deployment approaches. The evaluation results show that this component produces in a faster way suitable cloud service
composition solutions which optimally satisfy the application requirements compared to the solution quality and performance of the baseline approaches.

The rest of the article is structured as follows. Section 2 introduces a use case to appropriately motivate the proposed work. Section 3 presents the semantic cloud application management framework. Section 4 introduces a novel application requirement language. Section 5 analyzes the cloud service composition approach. Section 6 discusses the main results of the experimental evaluation. Section 7 reviews the related work. Finally, Section 8 concludes the article and draws directions for further research.

2. USE CASE

This case concerns the design of a real-world traffic management application [Baryannis et al. 2013] that regulates traffic at particular areas of a city. This application comprises the following three main tasks:

- **Monitoring Task (MT).** It monitors traffic conditions in a particular city area as well as air pollution and noise levels.
- **Analysis Task (AT).** It analyzes all monitored information and produces traffic regulation plans that optimally address the current traffic situation.
- **Traffic Configuration Task (TCT).** It enforces the traffic regulation plans derived by AT. This can involve changing traffic lights frequency, informing drivers about congested places and emergency personnel about accident placement and the particular actions to follow.

### Table 1. The components mapping to application tasks

<table>
<thead>
<tr>
<th>Task</th>
<th>Component Name</th>
<th>Component Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MT, AT, TCT</td>
<td>Con</td>
<td>It hosts the three main servlets of the application</td>
</tr>
<tr>
<td>MT</td>
<td>MC (choice)</td>
<td>It realizes MT's functionality</td>
</tr>
<tr>
<td>AT</td>
<td>AC</td>
<td>It realizes AT's functionality</td>
</tr>
<tr>
<td>AT</td>
<td>DC</td>
<td>A DB storing the information used for the analysis</td>
</tr>
<tr>
<td>TCT</td>
<td>TCC</td>
<td>It realizes TCT's functionality</td>
</tr>
<tr>
<td>WO</td>
<td>WO-Internal</td>
<td>It orchestrates the application workflow</td>
</tr>
</tbody>
</table>

Various software components/services have been developed or are required to realize this application, where either one or more map to a particular task. Moreover, a service oriented architecture (SOA) is chosen to realize the application tasks, so some components have to be hosted on servlet containers. Table 1 clearly shows the respective task-to-component mapping.

For some components, there is a choice of either selecting an existing realization (developed in house by the end-user or purchased/downloaded) or exploiting an external service. This choice is indicated in parenthesis after the respective component's name.

Concerning service performance and cost, Table 2 indicates the respective offerings along with information on which cloud provider offers them if they are external. The performance of internal services was determined via benchmarking which also lead to the ir eventual VM requirements (given later in this section). In addition, these services' cost is zero as it maps to an already purchased hosting infrastructure. The symbols used in this table have the following meaning: RT maps to response time, Av to availability and Thr to throughput. Cost information is per month based on the providers' cost model.

### Table 2. The QoS and cost features of the services

<table>
<thead>
<tr>
<th>Comp. Name</th>
<th>Service Name</th>
<th>QoS/cost chars</th>
<th>Provider Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>MC</td>
<td>MonService</td>
<td>RT ≤ 4 sec, Av ≥ 99.99%, Thr ≥ 10 reqs/sec, cost = 10$</td>
<td>CP1</td>
</tr>
<tr>
<td>MC</td>
<td>TraffService</td>
<td>RT ≤ 8 sec, Av ≥ 99%, Thr ≥ 5 reqs/sec, cost = 5$</td>
<td>CP2</td>
</tr>
<tr>
<td>MC</td>
<td>MC-Internal</td>
<td>RT ≤ 8 sec, Av ≥ 99.99%, Thr ≥ 6 reqs/sec, cost = 0$</td>
<td></td>
</tr>
<tr>
<td>AC</td>
<td>AC-Internal</td>
<td>RT ≤ 1.5 min, Av ≥ 99.99%, Thr ≥ 6 reqs/sec, cost = 0$</td>
<td></td>
</tr>
<tr>
<td>TCC</td>
<td>TCC-Internal</td>
<td>RT ≤ 0.5 min, Av ≥ 99.999%, Thr ≥ 12 reqs/sec, cost = 0$</td>
<td></td>
</tr>
<tr>
<td>WO</td>
<td>Orchestrate</td>
<td>Av ≥ 99.99%, Thr ≥ 12 reqs/sec, cost = 19$</td>
<td>CP1</td>
</tr>
<tr>
<td>WO</td>
<td>WFEngine</td>
<td>Av ≥ 99%, Thr ≥ 8 reqs/sec, cost = 15$</td>
<td>CP2</td>
</tr>
<tr>
<td>WO</td>
<td>WO-Internal</td>
<td>Av ≥ 99.99%, Thr ≥ 8 reqs/sec, cost = 0$</td>
<td></td>
</tr>
</tbody>
</table>

The end-user also requires the satisfaction of the following types of application requirements:

- **Deployment Requirements:**
  - There is a communication requirement from WO to all application servlets, i.e., MC, AC, and TCC, and from AC to DC.
  - MC, AC and TCC should be deployed on Con. These components will be hosted at the same instance of Con only when it is decided that they will be collocated.
  - AC and DC require a "high" VM. WO and MC a "medium" VM while TCC a "small" VM.

---

26
• AC and DC should be co-located while AC should not be co-located with any other component (apart from Con that hosts it).

- Cost requirements: Application cost must be no more than 380 $ per month.
- Quality requirements:
  - Application duration should not be longer than 2.5 minutes.
  - MC, AC and TCC should have throughput greater than or equal to 10, 6 and 5 reqs/sec, respectively.
  - MC and AC should have availability of 99.99% while TCC of 99.999%.
- Security requirements:
  - The security controls (https://cloudsecurityalliance.org/research/ccm) to be supported for the application must be: AAC-02 (independent reviews and assessment of provider at least annually), DSI-01 (data & service classification), DSI-05 (data leakage prevention), TVM-02 (timely vulnerability detection) and SEF-05 (monitoring & quantification of security incident type, volume and cost).
  - Meantime between incidents [Pannetar 2013] should be 6 months (mti > 6)

Table 3. The offerings of the four cloud providers

<table>
<thead>
<tr>
<th>Provider</th>
<th>Offered VM</th>
<th>Security Control</th>
<th>Security SLO</th>
</tr>
</thead>
<tbody>
<tr>
<td>CP1</td>
<td>(A) 2 core, 7.5GB, 32GB →.140s (B) 4 core, 15GB, 80GB →.280s (C) 4 core, 7.5GB, 80GB →.210s</td>
<td>(A) AAC-02 (B) AAC-03 (C) DSI-01 (D) DSI-05 (E) EKM-03 (F) TVM-02 (G) SEF-05</td>
<td>(A) mti ≥ 8 (B) ir (99%) ≤ 3</td>
</tr>
<tr>
<td>CP2</td>
<td>(A) 2 core, 2GB, 10GB →.06s (B) 2 core, 4GB, 50GB →.12s (C) 4 core, 8GB, 130GB →.24s</td>
<td>(A) AAC-02 (B) AAC-03 (C) DSI-01 (D) DSI-05 (E) EKM-03 (F) TVM-02 (G) SEF-05</td>
<td>(A) mti ≥ 6 (B) ir (99%) ≤ 2</td>
</tr>
<tr>
<td>CP3</td>
<td>(A) 1 core, 2GB, 10GB →.02s (B) 2 core, 4GB, 20GB →.04s (C) 4 core, 4GB, 40GB →.18s</td>
<td>(A) AAC-02 (B) AAC-03 (C) DSI-01 (D) DSI-05 (E) EKM-03 (F) TVM-02</td>
<td>(A) mti ≥ 6 (B) ir (99%) ≤ 4</td>
</tr>
<tr>
<td>CP4</td>
<td>(A) 1 core, 2 GB, 20GB →.2s</td>
<td>(A) AAC-02 (B) DSI-01 (C) DSI-05 (D) TVM-02 (E) SEF-05</td>
<td>(A) mti ≥ 4 (B) ir (99%) ≤ 4</td>
</tr>
</tbody>
</table>

Let us now consider four cloud providers, namely CP1, CP2, CP3 and CP4 which offer particular cloud services/VMs and realize a certain set of security controls. Σφάλμα! Το αρχείο πρόλεψης της αναφοράς δεν βρέθηκε. shows the VMs satisfying the end-user requirements offered by these providers (along with cost information), a subset of security controls supported by these providers and the security SLOs promised.

Real values for VM characteristics and cost were considered by collecting them from cloud provider web pages. So, we are as realistic as possible, provided that cloud providers do not usually advertise SLO and security information. Thus, we have opted for an idealized use case matching the real world in the near future, when cloud providers decide to advertise the latter information due to the main benefits that this will provide to them.

By considering all above information, a common cloud service composition approach would not consider the alternative design choices and end-user’s security requirements. As such, we can assume that the end-user would not specify that his/her components can be realized via external services. Thus, in the end, the respective approach would solve a simple optimization problem to produce a concrete IaaS composition. The outcome of such an approach would be a solution mapping application components to the following VMs (i) AC + DC → CP3 (C), (ii) MC → CP3 (B) and (iii) HO + TCC → CP3 (B), where Cpi (X) means the X offering of Cloud provider i. Colocation of HO and TTC in CP3 VM of type (B), while not imposed by any direct constraint, is proposed as TCC does not demand strict VM requirements so that it can be supported via a VM with better characteristics that suit the HO’s resource requirements.

Table 4. Cost, QoS and security features of the solutions

<table>
<thead>
<tr>
<th>Solution</th>
<th>Cost</th>
<th>Availability</th>
<th>Duration</th>
<th>Security</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>130$</td>
<td>99.99%</td>
<td>128 sec</td>
<td>no</td>
</tr>
<tr>
<td>2</td>
<td>286$</td>
<td>99.99%</td>
<td>128 sec</td>
<td>yes</td>
</tr>
<tr>
<td>3</td>
<td>129.8$</td>
<td>99.99%</td>
<td>124 sec</td>
<td>no</td>
</tr>
</tbody>
</table>

Table 4 summarizes the QoS, cost and security features of three solutions. The first solution maps to the common cloud service composition approach. The second solution, produced by our approach, is the best one as it considers all possible information and user requirements, including the security ones. It maps to selecting the external services offered by CP1 for all choices, the CP1 (C) VM for AC + DC and the CP2 (B) offering for TCC. To enable a more fair comparison, we also consider the third solution produced
via our approach by not considering security requirements, which is identical to the solution of the common service composition approach with the sole exception that \( MC \) and \( HO \) are not mapped to SaaS services. Please note that we assume that the common composition approach (as in the case of our approach) will produce the best possible solution based on the input provided to it as it will use a solving technique guaranteeing optimality (see Section 7).

The third solution, while not considering security requirements, is still more optimal than the first. The comparison between the second and third solution indicates the trade-off between security and cost that must be considered to produce the best possible solution based on user requirements.

The last two solutions propose a multi-cloud application design product spanning over two cloud providers (CP1 and CP2 for the second and CP1 and CP3 for the third). The second solution has filtered the remaining cloud providers as they do not satisfy the user security constraints: CP4 violates the SLO for mean time between incidents while CP3 does not support DSI-01 and SEF-05 security controls.

3. SEMANTIC CLOUD APPLICATION MANAGEMENT FRAMEWORK

The architecture of the semantic cloud application management framework can be seen in Figure 1. This framework spans both the design as well as the adaptive deployment and provisioning of a cloud application. It comprises the following components:

- The **Requirements Editor** is a User Interface (UI) component which interacts with users to obtain their requirements. Users, through this component, are guided in providing different types of requirements at different levels. These requirements are then transformed into a *requirement model* described via the semantic requirement language analyzed in Section 4.

- The **Semantic Matchmaker** attempts to match the user requirements against semantic cloud service descriptions. It then enriches the user’s requirement model through indicating which requirements are met by which cloud service alternatives, thus producing an *enhanced requirement model*. The SaaS matchmaking exploits particular techniques for both functional and non-functional aspects. Functional SaaS matchmaking relies on semantic input/output (IO) matching [Klusch et al. 2006]. Non-functional SaaS matchmaking, performed after the functional one, exploits particular matching metrics and respective constraint solving techniques [Kritikos & Plexousakis 2014] (see Section 7.2.1). IaaS matchmaking uses the same techniques as in non-functional SaaS matchmaking as IaaS offerings can be regarded comprising sets of constraints on VM features.

- The **Cloud Service Composer** obtains the *enhanced user requirement model* and transforms it into a constraint optimisation problem. This problem is then solved based on particular constraint solving techniques. In the end, the solution is transformed into a *deployment plan* specified in CAMEL [Rossini et al. 2014]. Section 5 provides a detailed analysis about this component.

- The **Cloud Deployment Engine** retrieves the *deployment plan* produced by the Cloud Service Composer and executes it. The Cloudiator framework [Domaschka et al. 2015a] has been used to realise its functionality. This framework not only abstracts away from the technical peculiarities of different clouds but is also capable of deploying applications in multiple clouds. In fact, in our opinion and also shown by the use case in Section 2, multi-cloud application deployment should be the way to go forward due to the following reasons: (a) leads to a more optimal satisfaction of application requirements as the most suitable alternative cloud services are selected with the best functional and quality capabilities; (b) the hurdle of vendor lock-in is surpassed enabling applications to be deployed in multiple and across different clouds.

- The **Semantic Knowledgebase** (SKB) contains information that spans the lifecycle of a cloud application as well as the main capabilities offered by different cloud providers. The SKB also includes the description of semantic rules operating over its content which can drive the adaptation behavior of an application by reacting on incoming application measurement and contextual information.
existing triple store (Virtuoso\(^1\)) with a reasoner on top of it (Pellet\(^2\)) has been used to realise the SKB.

- The Monitoring Engine is responsible for monitoring the cloud-based application in each and across all clouds in which it has been deployed. The MetricsCollector component [Domaschka et al. 2015b] has been used to realise its functionality. This component has been enhanced in order to operate over the semantic description of metrics defined in OWL-Q [Kritikos & Plexousakis 2006].
- The Adaptation Engine is responsible for adapting the cloud-based application when critical situations occur. Such situations are detected based on the findings of the Monitoring Engine and the content of the adaptation rules. The latter indicate which pattern of events represent these situations and what are the adaptation actions required to resolve them. The description of adaptation rules relies on SRL [Kritikos et al. 2014] while the Adaptation Engine was realised based on the work in [Zeginis et al. 2015]. It should be noted that when a current critical situation cannot be any more coped by the existing adaptation rules, the Adaptation Engine informs the Cloud Service Composer to propose a new cloud solution for the application at hand which should be able to overcome this situation.

This framework, as can be easily understood, exhibits various features that enable it to be distinguished from other research prototypes and products. More importantly, it is able to deal with multiple cloud levels as well as address the whole lifecycle of a cloud-based application, going from its design and deployment until its adaptive provisioning. Moreover, the different components involved in this framework can well be exploited by other prototypes and products in order to enable the latter to function at different cloud levels. For instance, the Requirements Editor, the Semantic Matchmaker and the Cloud Service Composer could be exploited by a product for the derivation of concrete deployment plans based on the initial user requirements. Then, such a product could employ its own components for the actual deployment and adaptive provisioning of an application.

4. APPLICATION REQUIREMENTS

ONTOGY LANGUAGE

In order to express different types of application requirements at different levels, a particular semantic language has been developed based on OWL, the de-facto standard in ontology description. This language relies on OWL-S [Sycara et al. 2003] for the functional description of the SaaS services and OWL-Q [Kritikos & Plexousakis 2006] for the non-functional and cost description of any kind of cloud service. We should note here that we consider that an application, being a composition of cloud services in the end, is also considered as a SaaS service. The class diagram in Figure 2 shows the main classes and their respective relationships for this language.

The main class for describing an application or a SaaS service is Service in OWL-S for which the process model part (cf. ProcessModel class) can be used to indicate the domain classes to which its I/O maps for matchmaking purposes. Concerning the service composition description, we have semantically-enhanced the part of the component meta-model in [Zeginis et al. 2015] dedicated to the modelling of composite services. In that meta-model, a service can be distinguished into a CompositeService or a SingleService. A CompositeService is associated to a data flow and a control flow, where the former indicates the data bindings between the inputs and outputs of the composite services' components while the control flow explicates the order of execution between the component services according to particular composition patterns. As such, through the re-use of this meta-model part, the application structure at the highest level can be provided. Obviously, in case users do not require to provide such a flow as they are not modelling a pure service and workflow-based application, they can just indicate via exploiting the proposed ontology the components that their application consists of.

Each Service is associated to a non-functional profile named as QoSDemand in OWL-Q. This profile maps to a logical combination of quality constraints indicating the set of conditions on particular quality metrics that are required for this service. Through using OWL-S and OWL-Q and by assuming that the SaaS services offered by cloud providers are described in an equivalent way, we enable the semantic matchmaker to combine the existing functional and non-function service matchmaking techniques in order to cover service discovery at the SaaS level.

The remaining types of requirements not covered by OWL-Q and OWL-Q are encapsulated as sub-classes of the Requirement class which is directly associated to a Service. This class comprises the DeploymentRequirement and SecurityControlRequirement sub-classes which represent deployment and security control requirements, respectively. A deployment requirement is further distinguished into service deployment, component deployment, communication and placement requirements.

---

1 http://virtuoso.openlinksw.com/
2 https://github.com/complexible/pellet
Figure 2. The class diagram of the requirements language

A service deployment requirement indicates whether a particular service component of the application will be realised via an external SaaS or an internal software component which has been either developed in house or has been purchased. Such a requirement can also leave both options open such that the proposed framework can decide the one that best satisfies the user requirements.

In case of external SaaS realisation, either the external SaaS service is fixed by pointing to its actual description or SaaS matchmaking will be performed to discover SaaS external services realising the respective service component’s functionality.

In case of a software component realisation (or of both options), a pointer to the description of the software component has to be provided. To this end, the SoftwareComponent class has been modelled which has a name and short description as attributes. It is related to Configuration information which maps to the handling of its lifecycle by indicating how the code of the component can be downloaded, configured, installed, executed and stopped. A software component may also have particular ports through which it can communicate with other components.

As it can be understood, this requirement specification is quite flexible. It can either rely on the framework to address the mapping of an application component to any kind of internal or external software or on the user to determine a concrete external SaaS and/or software component to address this mapping.
A component deployment requirement indicates the VM requirements for the IaaS service which will be used to host a software component which realises the functionality of an application component. The requirements for a VM span constraints over its main characteristics which are the amount of cores, the CPU frequency, the size of the main memory and the size of the hard disk, and are modelled via the VMRequirement class. The collection of a set of different constraints into a single class enables the re-use of VM requirements across different component deployment requirements.

A communication requirement actually connects two components indicating that they should communicate to each other. By also explicating the ports exposed by these components which will be used for their communication, the deployment of the application can be configured correctly such that there are no errors in the establishment of the communication links between these pair of components.

A placement constraint can indicate a particular physical (e.g., Europe) or virtual location (e.g., Amazon eu-west-1) in which a component can be placed or pair-wise placement constraints between components. As such, we can cater for cases where conformance to laws and regulations should be guaranteed as well as for cases where we desire to guarantee that the communication requirements between a pair of components are met. Pair-wise placement constraints can be of the following type: (a) SAME_VM: the same VM should be used to host the pair of components; (b) SAME_CLOUD: the same cloud should be used for the deployment of these two components; (c) DIFFERENT_VM: these two components should be placed in different VMs; (d) DIFFERENT_CLOUD: these two components should be placed in different clouds.

A SecurityControlRequirement actually maps an application component or the application itself to a set of security controls which have to be supported by the cloud provider(s) of the respective cloud service(s) selected to realise the application component(s) functionality. As can be easily understood, such a cloud service can be either an external SaaS or an IaaS used to host the internal software component realising the current application component's functionality. Each SecurityControl is described via a domain, a sub-domain, its id and a particular textual description. The set of security controls which is currently supported has been derived from the Cloud Control Matrix (CCM) of Cloud Security Alliance (SCA)\(^3\).

We should highlight here that VM requirements should be matched with VM capabilities. To this end, we have created another ontology-based semantic provider language, which is used to describe VM capabilities as well as the respective cloud providers that offer them. The VM capabilities are described in an equivalent, symmetric manner with respect to the respective requirements. The matching of them relies on a simple constraint problem derivation which is similar to the one used for non-functional SaaS matching. In this way, there is a uniform way in which non-functional and VM requirements are handled.

The semantic provider language also connects a cloud service provider to the SaaS services that it offers as well as to the set of security controls that it supports. In this way, we also enable the filtering of the cloud provider space according to the high-level security requirements posed.

The whole requirement model which can be expressed by our proposed ontology language can then be enhanced by the Semantic Matchmaker by indicating which application components can be realized by which cloud services. In this regard, the requirement model is updated with pointers to the descriptions of the respective cloud services in service deployment and component deployment requirements. The service deployment requirements will point to the external SaaS that can be used to realize an application component's functionality, while the component requirements will point to those IaaS that can be used to host the respective software component of a certain application component.

To summarize, the proposed requirement ontology language is quite flexible and expressive to specify any kind of cloud-based application requirement. It is also complemented with a cloud capability language to enable the appropriate matching of requirements to respective cloud service offerings. The final product derived through the modelling and matchmaking of requirements is an enhanced requirement model which drives the cloud service composition and the respective deployment plan production which will be dealt with in the next section.

5. CLOUD SERVICE COMPOSITION APPROACH

All possible design choice alternatives incarnated into the enhanced requirement model are transformed into a particular optimization problem by the Cloud Service Composer which, when solved, can discover the most optimal cloud service solution satisfying all user requirements posed irrespectively of their type. The approach followed was inspired by the service concretization work in [Ferreira et al. 2009]. In the following, we analyze the way the constraint problem is modelled in a step-wise manner, starting from optimization objectives and going down to the formulation of the high- and low-level constraints mapping to user requirements. Then, we check the complexity and possible constraint solving technologies for this problem.

5.1 CLOUD SERVICE COMPOSITION PROBLEM FORMULATION

To formulate the optimization objective of the problem, we rely on the Analytical Hierarchy Process (AHP) [Saaty 1980] to derive the relative importance of the quality

\(^3\) https://cloudsecurityalliance.org/group/cloud-controls-matrix/
parameters and cost to the user. The result of this process is an assignment of weights to all these parameters, indicating their relative importance, whose sum should equal to one. We also follow Simple Additive Weighting (SAW) technique [Hwang & Yoon 1981] which maps the optimization of all criteria considered to a single optimization objective which is equal to the weighted sum of the application of the global value derived for each parameter (QoS and cost) on its utility function posed. More formally, the objective is formulated as follows:

$$\text{maximize} \left( \sum_{q=1}^{Q} w_q \cdot uf_q (\text{val}_q) \right)$$

The utility function of each parameter is formulated based on the formulas in [Ferreira et al. 2009] which cater for slightly violating some problem constraints to address over-constrained user requirements. The following combined expression represents these formulas in which the first two cases depend on the monotonicity of the respective parameter (i.e., the first for negatively monotonic parameters like cost and the second for positively monotonic parameters like availability), where \( m \) is the max function, \( v^\text{max}_q \) and \( v^\text{min}_q \) are the maximum and minimum values requested by the end-user for the parameter \( q \) and \( a_q \) is a real number in \([0,0.1]\) used to regulate the percentage of values allowed outside the user-requested range.

$$uf_q (x) = \begin{cases} 
a_q + \frac{v^\text{max}_q - x}{v^\text{max}_q - v^\text{min}_q}( -a_q ), & v^\text{min}_q \leq x \leq v^\text{max}_q \land x \downarrow \\
a_q + x - \frac{v^\text{max}_q - v^\text{min}_q}{v^\text{max}_q - v^\text{min}_q}( -a_q ), & v^\text{min}_q \leq x \leq v^\text{max}_q \land x \uparrow \\
m \left( a_q - \frac{x - v^\text{min}_q}{v^\text{max}_q - v^\text{min}_q}( -a_q ) \right), & x < v^\text{min}_q \\
m \left( a_q - \frac{x - v^\text{max}_q}{v^\text{max}_q - v^\text{max}_q}( -a_q ) \right), & x > v^\text{max}_q 
\end{cases}$$

The value \( v_q \) that a parameter \( q \) can take depends on the type of this parameter and its derivation can be application-specific by depending on the application's structure [Ardagna & Pernici 2007]. To this end, in the general case, we consider that a particular application-specific function is provided taking as input the respective parameter values of the application components. In other terms: \( \text{val}_{q} = f_q (\text{val}_{q}^i) \) where \( \text{val}_{q}^i \) is the parameter value for application component \( i \). The use of a function covers all possible cases in parameter value derivation. In this way, by considering the running example, the application availability equals the product of availabilities of the three main components (i.e., \( MC, AC \) and \( TCC \)), while application cost is equal to the cost of all components (thus mapping to the respective cost of the infrastructure-as-a-service (IaaS) or SaaS exploited).

Before specifying the problem constraints, we introduce the main decision variables mapping to three variable arrays:

- \( y_i \) indicating whether the internal software component or the external SaaS services will be used to realize (application) component \( i \)
- \( x_{ijk} \) which indicates whether for the internal software of an application component \( i \), the IaaS offering \( k \) of the cloud provider \( j \) has been selected (internal service selection case) to host it.
- \( z_{il} \) indicating whether SaaS service \( l \) has been selected to realise the functionality of application component \( i \).

We differentiate between IaaS and SaaS services as they map to different formulas indicating how their parameter values can be mapped to the respective values at the (application) component level. While it could be argued that there is no need for explicating which IaaS offerings are provided by which cloud provider, we need to make this differentiation to be able to specify pair-wise placement constraints.

It is apparent that two exclusive cases exist for each application component: (a) there is no choice for realizing but just for deploying it and (b) there is indeed a realization choice. In the first case, it is enough to enforce that only one cloud provider and respective offering can be selected. In the second case, we need to indicate that only one external SaaS must be selected for realizing the component. Both cases lead to requiring the satisfaction of the following constraint:

$$\sum_j \sum_k x_{ijk} + \sum_l z_{il} = 1$$

Apart from the above constraints, we need to go down to the level of application components and indicate how their parameter values are derived from those of the offerings selected for them. We first assume that an application component's parameter value either maps to a one-to-one manner to the respective software component value which is computed as a function over the resources exploited (memory, CPU and storage), or is computed from the respective parameter value of the external SaaS realizing it. More formally:
val\textsuperscript{q} \textsubscript{i} = y_1 \cdot f_{i}^q(\text{core}_{i}, \text{mem}_{i}, \text{store}_{i}) + (1-y_1) \cdot \left( \sum_{l} z_{il} * val_{il}^q \right)

where $f_{i}^q$ is the function over the resources for parameter $q$ of (application) component $i$ while $val_{il}^q$ is the parameter value for the $l$ external SaaS of component $i$.

In case of internal software component deployment in the cloud the above (left part of the) computation is valid as the usual way of deriving high- from low-level requirements is either via benchmarking, simulation, or performance model learning [Xiong et al. 2013] such that we can map different service levels of application components to different resource levels.

Thus, we regard that the end-user has exploited one of the three possible approaches to produce the respective functions for those quality parameters of interest. We also envisage a step-wise approach to performance modelling. First, performance models for components are generated and then we go up to the level of the application. In this way, the component performance models will be more precise and will also lead to more accurate application performance models rather than attempting to map immediately the application performance to the underlying resources.

In this sense, we only need now to specify how the low-level resource values are produced for a particular component\textsuperscript{4}. This maps to the following three formulas:

\[
\text{core}_{i} = \sum_{jk} x_{ijk} * \text{core}_{jk}
\]

\[
\text{mem}_{i} = \sum_{jk} x_{ijk} * \text{mem}_{jk}
\]

\[
\text{store}_{i} = \sum_{jk} x_{ijk} * \text{store}_{jk}
\]

where $\text{core}_{i}$, $\text{mem}_{i}$, and $\text{store}_{i}$ are the variables mapping to the component's $i$ number of cores, main memory size and storage size, respectively, while $\text{core}_{jk}$, $\text{mem}_{jk}$, and $\text{store}_{jk}$ are the corresponding but fixed resource values for the concrete VM offering $k$ of provider $j$.

The cost of each component is calculated by considering the next formula:

\[
\text{cost}_{i} = y_1 \cdot \sum_{jk} x_{ijk} * \text{cost}_{jk} + (1-y_1) \cdot \sum_{il} z_{il} * \text{cost}_{il}
\]

where $\text{cost}_{jk}$ is the cost of IaaS offering $k$ of provider $j$ and $\text{cost}_{il}$ is the cost of SaaS $l$. Thus, a component’s cost equals the cost of the IaaS or SaaS it exploits.

The unary location constraints can regard either a physical or a virtual location. The latter maps to a specific cloud so this means that the enhanced requirement model will be already filtered according to such locations. There can be two types of physical locations, either countries or continents, where the former can obviously be included in the latter. By relying on the Food and Agriculture Organisation of United Nations (FAO) physical location ontology\textsuperscript{5}, we have covered all continents and countries by also modelling all their respective inclusion relationships. In this way, functions operating on the respective location elements can be employed in order to enforce the following two location constraints on IaaS and SaaS services, respectively.

\[
\text{equalOrIn\left(\text{loc}_{jk}, \text{loc}_{i} \right) == true}
\]

\[
\text{equalOrIn\left(\text{loc}_{il}, \text{loc}_{i} \right) == true}
\]

where equalOrIn is a function indicating whether the first location equals or is included in the second location, $\text{loc}_{jk}$ is the location of the $k$ VM offering of provider $j$, $\text{loc}_{il}$ is the location of service $l$ for component $i$ and $\text{loc}_{i}$ is the required location of component $i$. We should note here that we do not cover the case that the location of a cloud service includes the required location of the component as it is not certain that the cloud provider will be able to offer the respective service in the required country of the supported continent. Such a provider might support some but not all of the countries inside that particular continent.

We provide a specific formulation for pair-wise placement constraints depending on their type. A SAME_VM pair-wise placement constraint is formulated as follows:

\[
x_{ijk} \text{ is the cost of SaaS l}
\]

where $i$ and $i'$ are the two components for which the co- location constraint is posed. This constraint indicates that the decision for both components should coincide. Thus all values for respective array parts in which $i$ and $i'$ are fixed should be equal.

---

\textsuperscript{4} Please note that in the case of internal software deployment, we use the term component from now on to indicate both the application component and the software used to realise it as there is a one-to-one mapping between their respective parameters.

\textsuperscript{5} Available at: http://www.fao.org/countryprofiles/geoinfo/geopolitical/resource/
A DIFFERENT_VM placement constraint is expressed as follows:

\[
if (x_{ijk} = 1) \Rightarrow x_{i'jk} = 0
\]

indicating that if a particular offering \( k \) of a cloud provider \( j \) is selected for component \( i \), then this provider's offering cannot be selected for component \( i' \).

The SAME_CLOUD pair-wise placement constraint is formulated as follows:

\[
\sum_{k} x_{ijk} = \sum_{k} x_{i'jk}
\]

where \( i \) and \( i' \) are the two components for which the pair-wise placement constraint has been posed. This constraint indicates that for both components the same cloud has been selected which maps to requiring that the sum of values of the decision variables (mapping to the provider's offerings) for each cloud provider to be equal for these components.

The DIFFERENT_CLOUD placement constraint can be expressed as follows:

\[
if (\sum_{k} x_{ijk} = 1) \Rightarrow \sum_{k} x_{i'jk} = 0
\]

indicating that if any offering of cloud provider \( j \) is selected for component \( i \), then no offering from this provider can be selected for component \( i' \).

To conclude formulating the problem, we need to cater for the user security requirements which can be separated into high-level in terms of security controls and low-level in terms of SLOs. In the first case, we introduce set variables and enforce set operations to address the respective requirements. In particular, we enforce that if a particular cloud provider has been selected, then this provider should have realized all security controls required by the end-user. This is translated to the following complex constraint:

\[
\begin{align*}
if (y_i \land \sum_{k} x_{ijk} = 1) & \Rightarrow cc - ccp_j = \emptyset \\
\text{elseif} \quad (\neg y_i \land z_{il} = 1) & \Rightarrow cc - ccp_{z_{il}, \text{provider}} = \emptyset
\end{align*}
\]

where \( cc \) is a fixed set variable mapping to all required security controls, \( ccp_j \) is a fixed set variable mapping to the security controls supported by provider \( j \), and \( z_{il}, \text{provider} \) is the index of provider which offers SaaS \( l \) for component \( i \). We consider that the security control requirements should hold for any provider whose service is selected. In case such requirements are posed at the component level, the above formula can be remodelled by replacing \( cc \) with \( cc_i \) mapping to the fixed set variable for component \( i \) equal to the security controls to be realized by the provider whose service is used to realize or support this component.

In case of low-level security requirements, a similar constraint is posed:

\[
\begin{align*}
if \left( y_i \land \sum_{k} x_{ijk} = 1 \right) & \Rightarrow seq^p \geq seq^p \\
\text{elseif} \quad (\neg y_i \land z_{il} = 1) & \Rightarrow seq^p_{z_{il}, \text{provider}} \geq sec^p
\end{align*}
\]

where \( seq^p \) is the low required threshold for security property \( p \) while \( seq^p_j \) is the respective property value promised by provider \( j \). This formula is meaningful for positively monotonic security properties. The opposite case can be easily derived but due to space limitations is not shown. If the user provides both low and upper thresholds, the constraints introduced for both security property types must be enforced.

5.2 COMPLEXITY & SOLVING TECHNIQUES

The common cloud service composition problem is NP-Hard [Jula et al. 2014]. While we use additional sets of constraints, especially non-linear ones, and variables, the general problem formulation showed in previous subsection is still NP-Hard.

Due to the nature of this problem, Mixed-Integer Programming (MIP) techniques cannot be actually used. Thus, non-linear constraint solving techniques must be checked, from which we have selected the Constraint Solving Optimization Problem (CSOP) ones, as they seem the perfect candidate for our case. These techniques can address not only non-linear constraints but can also cater for the use of different variables, such as boolean, integer, and set variables. However, real variables are not natively supported. To this end, the current workaround that seems to work well in many circumstances is to combine the use of CSOP with either MIP or Constraint Programming techniques focusing on interval arithmetic. In fact, many hard and real-world problems are now solved through the combined use of these techniques [Timpe 2002; Milano 2003].

In our current implementation, we have used a well-known and free CSOP solver called Choco (choco-solver.org) which is also supported by a very active community, while performs well and even competes with proprietary solvers. Apart from supporting all types of variables required, Choco has implemented well-known
6. EXPERIMENTAL EVALUATION

We have conducted a preliminary experimental evaluation of our approach performance which aimed at assessing the effect of an increasing number of cloud provider offerings and placement constraints. To this end, two separate experiments were performed evaluating the effect that each different factor has. Three CSOP approaches were actually evaluated: (a) RESOURCE mapping to the common IaaS composition method used as a baseline where only resource constraints are considered and just one optimization parameter (cost), (b) RESOURCE_SEC which is same as previous method but enriched with security and placement constraints and (c) FULL which is the actual proposed approach.

The evaluation metric was the average solving time whose value was generated over 30 runs in order to minimize various interference types in the measurements, such as those attributed to the running OS. The computer on which the experiments were performed had the following characteristics: 1.7 GHz CPU, 2GB of main memory and 500 GB of disk.

The input given to the three approaches was randomly generated but only realistic values were considered. For instance, the core number was given values from 1 to 8 while main memory from 512 to 8192 for a particular cloud provider IaaS. Security capabilities were formed by randomly assigning a specific percentage of all possible security controls for each cloud provider, while a respective smaller percentage was used as the application requirement. Placement constraints were formed by randomly picking up their type and component pair on which they should hold. Then, each approach exploited this input, created the respective CSOP problem and solved it. In the CSOP formulation, a linear function from resources to QoS attributes was utilized for each component. It was also assumed that the composition of values for execution time & cost, throughput and availability at the global application level exploited additive, minimum or multiplicative functions, respectively.

The initial values for the experiment configuration parameters were: application component number → 5, cloud provider number → 10, IaaS/SaaS number per provider → 5 and placement constraint number → 5. In the first experiment, we increased the value of the IaaS/SaaS offerings per provider in units of 5 until the value of 25. In this way, we simulate the case where either an increased number of offerings is supplied by each provider or an increased number of providers occurs. The evaluation results are shown in Figure 3. As it can be seen, due to the nature of the problem, all approaches exhibited an exponential behavior. However, our approach had a better performance than the others. This can be certainly justified by the fact that while slightly increasing the variable number, the constraint number is also increased. As such, the constraint solving algorithm more deeply cuts the search space to find the most optimal solution. The same holds when comparing RESOURCE_SEC and RESOURCE where again the increased number of constraints leads to a better performance. We have posed a limit of 3 minutes to the solving time so as to be acceptable by an application designer which justifies the first approach behavior.

The second experiment focused on examining the effect on increasing the placement constraint number from 1 to 5 (but not greater due to the small number of application components). Figure 4 shows the respective evaluation results only for the last two approaches that are indeed capable of considering such constraints. The same linear decreasing behavior is observed for both approaches. This is expected as placement constraints reduce the offering space to be explored. Again, FULL had a better performance than RESOURCE_SEC as it considers also high-level constraints.
7. RELATED WORK

7.1 SERVICE MODELLING

7.1.1 SOFTWARE SERVICE MODELLING

WSDL [Christensen et al. 2001] is the de-facto standard for the description of the interface for web-services. However, it is a structural language and does not cover other information aspects, such as the service functionality and its non-functional capabilities.

USDL was a semi-formal language used for the description of business and software services. Recently, it has been transformed to a Linked-Data counterpart [Pedrinaci et al. 2014] in order to become more formal. USDL is capable of covering SLA, quality, security, cost and legal aspects. There was also an approach [Cardoso et al. 2013] focusing on the integration of USDL with TOSCA [Palma and Spatzier 2013] to link service selection with deployment such that the cloud application lifecycle is better supported.

OWL-S is a W3C recommendation for the semantic description of web services. It mainly focus on functional aspects, covering the semantic description of the service I/O and its abstract interface. It also proposes a particular grounding mechanism in WSDL. However, OWL-S does not cover the non-functional aspects and is not able to describe service orchestrations.

WS-BPEL [Alves et al. 2007] is a service orchestration language which has been widely adopted. It is relies on WSDL for the description of the component services of the orchestration but there have been research-based extensions which have relied on WSMO [de Bruijn et al. 2005], another semantic service description language with equivalent capabilities to OWL-S. WS-BPEL also comes with additional extensions towards covering service choreographies and human tasks. However, this language does not cover non-functional aspects.

SoaML [Amsden et al. 2012] is a UML-based language for specifying Service-Oriented Architectures (SOAs) able to define components and their inter-relationships at the business and service levels. However, SoaML cannot actually describe service orchestrations and to this end, it needs to be complemented by a service orchestration language, like WS-BPEL. Moreover, it does not cover non-functional aspects.

Concerning the non-functional description of services, various approaches have been proposed which can be distinguished according to their performance on various comparison criteria, such as their expressiveness, complexity, formality and extensiveness [Kritikos et al. 2013]. Among these approaches, OWL-Q can be considered as the one which has the best performance across all these criteria. OWL-Q is semantic, extensible and very expressive, covering all major aspects in service quality description, including quality attributes, metrics, units, and measurement formulas. It is able to specify both service quality models that can be used for populating SLAs as well as quality-based service descriptions covering the non-functional profiles of services. Finally, OWL-Q can be connected to any functional service description language to enable the complete description of web services.

7.1.2 CLOUD SERVICE/APPLICATION MODELLING

TOSCA is considered as a de-facto standard for the deployment description of applications and seems to be widely used in research prototypes. However, it has particular shortcomings related to the non-coverage of the instance level which is required for dealing with runtime aspects, the lack of domain/cloud-specific constructs and the incomplete coverage of the non-functional aspects.

CAMEL is a multi-purpose DSL developed in the context of the PaaSage European project. It is able to cover many aspects related to the lifecycle management of cloud applications, such as the deployment, monitoring, scalability, organisation, security and cloud service ones. In terms of deployment, it relies on CloudML [Ferry et al. 2013] which follows the template-instance pattern and caters for runtime aspects through a models@runtime approach [Assmann et al. 2011]. However, the latter DSL does not cover all types of placement constraints and is
mainly oriented towards IaaS services. The description of cloud services relies on the Saloon framework's [Quinton et al. 2013] generic DSL able to cover any possible cloud service offering. The main drawback of CAMEL as a whole is that it is semi-formal as it is Ecore-based. Thus, it does not have the formality level and reasoning capabilities of an ontology-based modelling approach.

The language in [Nguyen et al. 2011] is able to support the semi-formal description of Blueprint Templates which cover cloud-offerings at multiple abstraction levels and capture service capability, virtual topology and QoS & policy aspects. Apart from being a semi-formal language, it does not capture all lifecycle aspects as covered, e.g., in CAMEL. In addition, it is not capable of defining the quality terms required for specifying the quality capabilities of the respective service offerings.

A cloud meta-model is proposed in [Galán et al. 2009] which extends OVF6 towards covering self-configuration, elasticity and performance monitoring. This meta-model is not capable of specifying placement constraints, component dependencies and quality capabilities and requirements.

Another OVF extension called service manifest has been proposed in [Rumpl et al. 2010] which covers placement and allocation constraints, security requirements and performance profiles according to the properties of trust, reputation, eco-efficiency and cost. This service manifest, however, stays mainly at the IaaS level and is not capable of describing component dependencies while does not cover additional quality attributes related, e.g., to performance.

The mOSAIC ontology [Moscato et al. 2011] has been developed in OWL and can be used for the semantic annotation of semi-formal cloud service descriptions. It is rich enough to cover various aspects, including cloud service requirements and resources, metrics, SLAs, components and policies.

7.2 SERVICE MATCHMAKING

7.2.1 SOFTWARE SERVICE MATCHMAKING

The functional matchmaking of software services has mainly relied on the service I/O. The approaches proposed rely on employing either Information Retrieval [Dong et al. 2004] or ontology-based [Paolucci et al. 2002] or both types of techniques [Plebani & Pernici 2009]. While the latter two types of approaches seem to cater for better accuracy, they do not consider the service behavior. As such, the results produced will never be as accurate as possible. To remedy for this, few approaches [Sycara et al. 2002] have employed behavior-based service matching by relying on full input-output-precondition-effects (IOPE) service profiles. While these approaches reach even higher matchmaking accuracy levels have the main drawback that full IOPE service profiles do not exist in reality and require additional modelling effort by the cloud service provider.

Non-functional software service matching approaches can be separated into constraint-based, ontology-based or mixed. Constraint-based approaches [Cortés et al. 2005] express the quality description of service requirements and capabilities as a constraint model and then employ constraint solving techniques and particular matching metrics (e.g., subsumption [Cortés et al. 2005]) in order to perform the service matchmaking. These approaches assume that the quality-based service specification comprise terms which have been defined in a common quality term repository. Ontology-based approaches [Zhou et al. 2004] provide ontology languages for enabling the service quality description and exploit subsumption reasoning to infer whether service quality capabilities match the respective requirements posed. Such approaches have the drawback that can only involve the processing of unary quality-based service specifications which comprise one quality term per constraint. Finally, mixed approaches [Kritikos & Plexousakis 2014] combine the best from both worlds with respect to the previous approach types. They rely on ontologies to enable the service quality description, they then align the descriptions according to the quality terms and finally transform the aligned descriptions into constraint problems in order to use the techniques in the first approach type to perform the actual service matchmaking. The latter types of approaches exhibit better accuracy than the other types and are also able to cope with n-ary quality service specifications.

7.2.2 CLOUD SERVICE MATCHMAKING

Ruiz-Alvarez & Humphrey [2011] have proposed an approach which is able to discover cloud storage services that match the respective application requirements posed. Cloud storage requirements and capabilities are expressed via a semi-formal language. The framework in [Garg et al. ] ranks cloud services according to their quality performance and weights given to each quality term according to the AHP process. Zeng et al. [2009] follow a Wordnet-based approach to measure the similarity of concepts in the I/O of cloud service specifications. The federated cloud environment in [Buyya et al. 2010] is able to match user quality requirements to cloud services. D' Andria et al. [2012] propose a PaaS matchmaking and selection framework which filters PaaS according to user requirements and ranks the remaining PaaS based on the number of user preferences satisfied. The approach in [Garcia-Gómez et al. 2012] addresses blueprint (see Subsection 7.1.2) matchmaking and is able to produce a composite blueprint document which comprises the cloud services that can be used for realising or supporting a cloud application. It does not propose though a concrete cloud

---

6 https://www.dmtf.org/standards/ovf
service composition but just set of alternatives for different application elements at different levels of abstraction.

7.3 Service Composition

7.3.1 Software Service Composition

The successful SOA paradigm has led to a proliferation of available services. Such services can then be optimally combined to produce added-value functionality incarnated into respective applications. To this end, various service composition approaches have been proposed which usually focus either on the functional or QoS aspect. Most of the QoS-based work follows either a statistical [Canfora et al. 2005] or path-based approach [Ardagna & Pernici 2007] leading to an over-simplification or a pessimistic view of the problem. Some approaches employ a heuristic [Yu et al. 2007] or a QoS decomposition [Alrifai et al. 2009] approach to cater for better performance but sacrificing optimality. In addition, all these approaches regard QoS service offerings as simple QoS parameter values which is quite unrealistic if we also regard that many services run in quite dynamic environments. Moreover, these approaches fail to produce any result for over-constrained end-user requirements. One promising approach resolving most of the above issues was proposed in [Ferreira et al. 2009]. Some key aspects of this approach were exploited in our cloud service composition work.

7.3.2 Cloud Service Composition

The cloud service composition problem is harder than that of service selection as it involves composing different types of services with different characteristics and the synthesis is performed in different but inter-dependent levels such that the solution at one level impacts the solution at other levels. However, the cloud service composition approaches proposed usually focus on just one cloud service type. Even when they consider additional types, they either solve a limited case of the actual problem or a slightly different problem by also neglecting all possible user requirement types.

Concerning SaaS composition, the respective approaches can be separated into those which: (a) consider semantics [Zeng et al. 2009], (b) use heuristics to solve the respective optimization problem [Kofler et al. 2010], (c) address multi-tenant SaaS [He et al. 2012], (d) exploit feature models and multi-criteria decision making [Wittern et al. 2012] to find the most optimal SaaS compositions and (e) consider some other aspects, such as the network latency and the multiple instances that a particular SaaS service can have [Klein et al. 2012]. Although not clearly addressing IaaS services, the latter approach seems interesting and could be used for further extending our proposed work towards selecting only the appropriate instances for each SaaS selected.

The self-organizing agent-based cloud service composition method in [Gutierrez-Garcia et al. 2013] exploits distributed problem solving techniques, by also relying on the contract-net protocol, and is able to produce vertical, horizontal, one-time and persistent service compositions. Both SaaS and IaaS type of services are handled. However, this approach seems to cater only for functional and cost requirements.

In [Karim et al. 2013], an hierarchical quality model is proposed going from user requirements down to the QoS capabilities of IaaS services. This quality model is then used for ranking the service candidates across the different cloud levels. However, the ranking algorithm proposed seems to work on a different problem type where the end-user requires one or more SaaS services and then the providers of these services have to find suitable IaaS offerings for hosting their services. In addition, this algorithm does not consider placement constraints, while only low-level security requirements are taken into account. Finally, the algorithm seems to work only for sequential application workflow specifications.

8. Conclusions & Future Work

This article has presented a semantic cloud service composition framework which is able to address the whole lifecycle of an application in a multi-cloud environment. The genuine features of this framework is that it relies on semantics and constraint optimisation techniques which guarantee the quality of the cloud service composition derived. It is able to consider different types of requirements and different types of design choices across different levels for which their dependencies are accounted for. This framework is also able to dynamically adapt the application at runtime via exploiting adaptation rules as well as re-configuration opportunities provided by the Cloud Service Composer when quite critical situations occurs which cannot be addressed by such rules.

The framework’s Cloud Service Composer advances the state-of-the-art as apart from considering a variety of different types of quite meaningful and critical application requirements, it is able to concurrently consider different cloud levels in order to produce the final cloud service composition product. Moreover, it does not sacrifice accuracy through the use of any kind of heuristics. This guarantees the optimality of the composition product and reduces the probability that such product has to be adapted at runtime. In addition, as shown from the experimental evaluation, it leads to a reduction in the composition execution time compared to common IaaS composition approaches, thus making the composition algorithm suitable for use even at runtime.

The following research directions are planned. First, a complete evaluation of the whole framework has to take
place. Second, the exploitation of PaaS services needs to be considered in order to make the solution even more complete with respect to the types of cloud services that can be exploited. Third, a complete UI spanning all the application lifecycle phases will be produced enabling application developers not only to pose requirements but also see the deployment progress and runtime performance of their applications and possibly interfere by e.g., enforcing adaptation actions or changing requirement models or plans. Fourth, it is planned to extend the functionality of the framework in order to deal with an additional level on the top mapping to the handling of business processes in order to realise the vision of BPaaS [Woitsch & Utz 2015].

9. ACKNOWLEDGMENT

The research leading to these results has received funding from the European Community’s Seventh Framework Programme (FP7/2007-2013) under grant agreement number 317715 (PaaSage) and from the European Community’s Framework Programme for Research and Innovation HORIZON 2020 (ICT-07-2014) under grant agreement number 644690 (CloudSocket).

10. REFERENCES


### Authors

Kyriakos Kritikos received his B.Sc., M.Sc., and Ph.D degrees in Computer Science from the University of Crete. He was a Post-Doc Researcher at Politecnico di Milano and CNR in Italy as well as FNR in Luxembourg. He is currently a Researcher at the Information Systems Laboratory of the Institute of Computer Science, FORTH in Greece. His research interests span the following areas: Quality-aware service management; Cross-layer service monitoring and adaptation; Cloud-based application modelling and deployment; Ontology modeling and reasoning; Constraint and Mathematical Programming; Distributed (Information) Systems.

Dimitrios Plexousakis is a Professor at the Computer Science Department, University of Crete and a Researcher as well as the head of the Information Systems Laboratory of the Institute of Computer Science, FORTH in Greece. He received his B.Sc. degree in Computer Science from University of Crete and M.Sc. and Ph.D degrees in Computer Science from the University of Toronto. His research interests span the following areas: Knowledge Representation and Knowledge Base Design; Formal knowledge representation models and query languages for the Semantic Web; Formal reasoning systems with focus on dynamic action theories and belief revision; Business process and e-service modeling, discovery and composition. He is a member of ACM and IEEE.
Call for Articles
International Journal of Services Computing

Mission
The International Journal of Services Computing (IJSC) aims to be a reputable resource providing leading technologies, development, ideas, and trends to an international readership of researchers and engineers in the field of Services Computing. To ensure quality, IJSC only considers extended versions of papers published at reputable international conferences such as IEEE ICWS.

From the technology foundation perspective, Services Computing covers the science and technology needed for bridging the gap between Business Services and IT Services, theory and development and deployment. All topics regarding Web-based services lifecycle study and management align with the theme of IJSC. Specially, we focus on: 1) Web-based services, featuring Web services modeling, development, publishing, discovery, composition, testing, adaptation, and delivery, and Web services technologies as well as standards; 2) services innovation lifecycle that includes enterprise modeling, business consulting, solution creation, services orchestration, services optimization, services management, services marketing, business process integration and management; 3) cloud services featuring modeling, developing, publishing, monitoring, managing, delivering XaaS (everything as a service) in the context of various types of cloud environments; and 4) mobile services featuring development, publication, discovery, orchestration, invocation, testing, delivery, and certification of mobile applications and services.

Topics
The International Journal of Services Computing (IJSC) covers state-of-the-art technologies and best practices of Services Computing, as well as emerging standards and research topics which would define the future of Services Computing. Topics of interest include, but are not limited to, the following:

- Services Engineering
- XaaS (everything as a service)
- Cloud Computing for Internet-based services
- Big Data services
- Internet of Things (IoT) services
- Pervasive and Mobile services
- Social Networks and Services
- Wearable services
- Web 2.0 and Web X.0 in Web services
- Service-Oriented Architecture (SOA)
- RESTful Web Services
- Service modeling and publishing
- Service discovery, composition, and recommendation
- Service operations, management, and governance
- Services validation and testing
- Service privacy, security, and trust
- Service deployment and evolution
- Semantic Web services
- Scientific workflows
- Business Process Integration and management
- Service applications and implementations
- Business intelligence, analytics and economics for Services
Call for Articles
International Journal of Big Data

Mission
Big Data has become a valuable resource and mechanism for the practitioners and researchers to explore the value of data sets in all kinds of business scenarios and scientific investigations. New computing platforms such as cloud computing, mobile Internet, social network are driving the innovations of big data. From government initiative perspective, Obama Administration in United States launched "Big Data" initiative that announces $200 Million in new R&D investments on March 29, 2012. European Union also announced "Big Data at your service" on July 25, 2012. From industry perspective, IBM, SAP, Oracle, Google, Microsoft, Yahoo, and other leading software and internet service companies have also launched their own innovation initiatives around big data.

The International Journal of Big Data (IJBD) aims to provide the first Open Access publication channel for all authors working in the field of all aspects of Big Data. Big Data is a dynamic discipline. One of the objectives of IJBD is to promote research accomplishments and new directions. Therefore, IJBD welcomes special issues in any emerging areas of big data.

Topics
IJBD includes topics related to the advancements in the state of the art standards and practices of Big Data, as well as emerging research topics which are going to define the future of Big Data. Topics of interest to include, but are not limited to, the following:

**Big Data Models and Algorithms** (Foundational Models for Big Data, Algorithms and Programming Techniques for Big Data Processing, Big Data Analytics and Metrics, Representation Formats for Multimedia Big Data)

**Big Data Architectures** (Cloud Computing Techniques for Big Data, Big Data as a Service, Big Data Open Platforms, Big Data in Mobile and Pervasive Computing)

**Big Data Management** (Big Data Persistence and Preservation, Big Data Quality and Provenance Control, Management Issues of Social Network enabled Big Data)

**Big Data Protection, Integrity and Privacy** (Models and Languages for Big Data Protection, Privacy Preserving Big Data Analytics Big Data Encryption)

**Security Applications of Big Data** (Anomaly Detection in Very Large Scale Systems, Collaborative Threat Detection using Big Data Analytics)

**Big Data Search and Mining** (Algorithms and Systems for Big Data Search, Distributed, and Peer-to-peer Search, Machine learning based on Big Data, Visualization Analytics for Big Data)
