

# Cloud Service Discovery and Analysis: A Unified Framework



THE UNIVERSITY  
*of* ADELAIDE

**Abdullah Alfazi**

School of Computer Science

The University of Adelaide

This dissertation is submitted for the degree of

*Doctor of Philosophy*

Supervisors: Prof. Ali Babar, Prof. Michael Sheng

and Dr. Lina Yao

November 2017

© Copyright by  
Abdullah Alfazi  
November 2017

All rights reserved.

No part of the publication may be reproduced in any form by print, photoprint,  
microfilm or any other means without written permission from the author.

*To my mother,  
my sisters,  
my brothers,  
who made all of this possible,  
for their endless encouragement and patience.*



## **Declaration**

I certify that this work contains no material which has been accepted for the award of any other degree or diploma in my name, in any university or other tertiary institution and, to the best of my knowledge and belief, contains no material previously published or written by another person, except where due reference has been made in the text. In addition, I certify that no part of this work will, in the future, be used in a submission in my name, for any other degree or diploma in any university or other tertiary institution without the prior approval of the University of Adelaide and where applicable, any partner institution responsible for the joint-award of this degree. I give permission for the digital version of my thesis to be made available on the web, via the University's digital research repository, the Library Search and also through web search engines, unless permission has been granted by the University to restrict access for a period of time.

Abdullah Alfazi

November 2017



## Acknowledgements

During the course of my PhD journey, I was privileged to work with the school, staff and students at the University of Adelaide. I would like to take this opportunity to thank many people who made this journey such a pleasant and memorable experience. To begin with, I would like to express my sincerest gratitude to Prof. Ali Babar and Prof. Micheal Sheng. I cannot thank them enough for their tremendous support, help and guidance. They are truly outstanding researchers and great leaders with a penchant to drive results. They triggered the ambition of success within me and I am continuing to learn from them. They taught me invaluable lessons on undertaking high-quality research and becoming a contributing/successful member of the research community, while imparting critical insights on how to be an able teacher and supervisor, and improve my academic career in general. Words cannot be enough to explain my appreciation for their motivation, encouragement and support. Secondly, I would also like to thank my co-supervisors Dr. Lina Yao for her constant suggestions, feedback and encouragement. She was always willing to undertake a meaningful and productive discussion about current research problems.

I would like to acknowledge my co-authors: Quan Z. Sheng, Ali Babar, Yongrui Qin, Talal H. Noor, Wei Emma Zhang, Lina Yao, Wenjie Ruan and Yong Xu for their support and joyful collaborations. Also, I would like also to thank anonymous reviewers for their useful feedback on previous drafts.

I also express my humble gratitude towards my mother, my brothers and my sisters for their patience, support and encouragement. I would not have been able to

attain any kind of success without their presence and timely interventions - in the past, present and future.

Finally, I express my sincere appreciation to the Ministry of Higher Education, Kingdom of Saudi Arabia, who awarded me King's Salman selective Postgraduate Scholarship, Kingdom of Saudi Arabia, to financially support me during my PhD journey.



## **Abstract**

Over the past few years, cloud computing has been more and more attractive as a new computing paradigm due to high flexibility for provisioning on-demand computing resources that are used as services through the Internet. The issues around cloud service discovery have been considered by many researchers in the recent years. However, in cloud computing, with the highly dynamic, distributed, the lack of standardized description languages, diverse services offered at different levels and non-transparent nature of cloud services, this research area has gained a significant attention. Robust cloud service discovery approaches will assist the promotion and growth of cloud service customers and providers, but will also provide a meaningful contribution to the acceptance and development of cloud computing. In this dissertation, we have proposed an automated cloud service discovery approach of cloud services. We have also conducted extensive experiments to validate our proposed approach. The results demonstrate the applicability of our approach and its capability of effectively identifying and categorizing cloud services on the Internet. Firstly, we develop a novel approach to build cloud service ontology. Cloud service ontology initially is built based on the National Institute of Standards and Technology (NIST) cloud computing standard. Then, we add new concepts to ontology by automatically analyzing real cloud services based on cloud service ontology Algorithm. We also propose cloud service categorization that use Term Frequency to weigh cloud service ontology concepts and calculate cosine similarity to measure the similarity between cloud services. The cloud service categorization algorithm is able to categorize cloud services to

clusters for effective categorization of cloud services. In addition, we use Machine Learning techniques to identify cloud service in real environment. Our cloud service identifier is built by utilizing cloud service features extracted from the real cloud service providers. We determine several features such as similarity function, semantic ontology, cloud service description and cloud services components, to be used effectively in identifying cloud service on the Web. Also, we build a unified model to expose the cloud service's features to a cloud service search user to ease the process of searching and comparison among a large amount of cloud services by building cloud service's profile. Furthermore, we particularly develop a cloud service discovery Engine that has capability to crawl the Web automatically and collect cloud services. The collected datasets include meta-data of nearly 7,500 real-world cloud services providers and nearly 15,000 services (2.45GB). The experimental results show that our approach i) is able to effectively build automatic cloud service ontology, ii) is robust in identifying cloud service in real environment and iii) is more scalable in providing more details about cloud services.

# Table of contents

<b>List of figures</b>	<b>xv</b>
<b>List of tables</b>	<b>xvii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Introduction . . . . .	1
1.2 Motivating Scenario . . . . .	2
1.3 Research Issues . . . . .	5
1.4 Contributions Overview . . . . .	6
1.4.1 Cloud Service Ontology . . . . .	7
1.4.2 Cloud Service Categorization Model . . . . .	7
1.4.3 Cloud Service Identifier Model . . . . .	7
1.4.4 Dataset of Cloud Services Collection . . . . .	8
1.4.5 Implementation and Performs Study . . . . .	9
1.5 Dissertation Publications . . . . .	9
1.6 Dissertation Organization . . . . .	11
<b>2 Background</b>	<b>13</b>
2.1 Overview of Cloud Computing . . . . .	13
2.1.1 Cloud Service Models . . . . .	15
2.1.2 Deployment Models . . . . .	16
2.2 Overview of Service Discovery . . . . .	18

2.3	Cloud Service Discovery . . . . .	19
2.3.1	Cloud Service Ontology . . . . .	19
2.3.2	Cloud Service Language . . . . .	25
2.3.3	Cloud Service Model . . . . .	29
2.4	Summary . . . . .	32
<b>3</b>	<b>Cloud Service Ontology and Categorization</b>	<b>35</b>
3.1	Cloud Service Categorization Framework . . . . .	37
3.1.1	CSCF Architecture . . . . .	37
3.2	Cloud Service Ontology Generation . . . . .	40
3.3	Cloud Service Categorization . . . . .	47
3.3.1	Cloud Service Concepts Investigator . . . . .	47
3.3.2	Cloud Service Concepts Weight . . . . .	47
3.3.3	Cloud Service Concepts Similarity . . . . .	49
3.4	Related Work . . . . .	52
3.5	Summary . . . . .	53
<b>4</b>	<b>A Robust and Adaptive Identifier Model for Cloud Service in Real Environment</b>	<b>55</b>
4.1	Overview of Cloud Service Identifier Framework . . . . .	57
4.1.1	CSIF Architecture . . . . .	57
4.2	Cloud Service Identifier . . . . .	61
4.2.1	Cloud Service Identification Learning Features . . . . .	61
4.2.2	Cloud Service Identification Text Features . . . . .	67
4.2.3	Cloud Service Clustering . . . . .	68
4.3	Cloud Service Profile . . . . .	69
4.4	Related Work . . . . .	71
4.5	Summary . . . . .	73

<b>5 Automated Discovery of Cloud Service Using Crawling and Data Collector</b>	<b>75</b>
5.1 Cloud Service Discovery Engine . . . . .	76
5.2 Cloud Service Crawling Procedures . . . . .	79
5.3 Statistical Analysis and Crawling Results . . . . .	80
5.3.1 Cloud Services Identification . . . . .	81
5.3.2 Cloud Service Extracting Data Error . . . . .	82
5.3.3 Cloud Service Locations . . . . .	84
5.3.4 Cloud Service Provider Categorization . . . . .	88
5.3.5 Cloud Computing and Service-Oriented Computing (SOC) . .	88
5.3.6 Cloud Service Advertisement . . . . .	89
5.3.7 Cloud Service IP . . . . .	90
5.3.8 Discussion . . . . .	90
5.4 Challenges of Discovery Cloud Services . . . . .	91
5.5 Related Work . . . . .	93
5.6 Summary . . . . .	95
<b>6 Implementation and Performance Study</b>	<b>97</b>
6.1 Cloud Service Dataset . . . . .	98
6.2 Cloud Service Ontology . . . . .	98
6.2.1 Threshold Identification . . . . .	99
6.2.2 Identifying Cloud Services Using Ontology . . . . .	101
6.2.3 Discussion . . . . .	102
6.3 Cloud Service Categorization . . . . .	104
6.3.1 Cloud Service Categorization . . . . .	104
6.3.2 Robustness Cloud Service Categorization . . . . .	106
6.4 Cloud Service Identifier . . . . .	108
6.4.1 Cloud Service Identifier Learning Features . . . . .	108

---

6.4.2	Identifying Cloud Services Based Text Features . . . . .	110
6.4.3	Identification Robustness of Cloud Service Features . . . . .	111
6.4.4	Cloud Service Profile . . . . .	118
6.5	Summary . . . . .	119
<b>7</b>	<b>Conclusion</b>	<b>121</b>
7.1	Summary . . . . .	121
7.1.1	Cloud Service Ontology . . . . .	123
7.1.2	Cloud Service Categorization Model . . . . .	123
7.1.3	Cloud Service Identifier Model . . . . .	123
7.1.4	Dataset of Cloud Service Collection . . . . .	124
7.1.5	Implementation and Performance Study . . . . .	125
7.2	Future Work . . . . .	125
7.2.1	Cloud Service Standards . . . . .	125
7.2.2	Cloud Service Recommendation . . . . .	125
7.2.3	Cloud Service Personalization . . . . .	126
	<b>References</b>	<b>127</b>
	<b>Appendix A CV</b>	<b>139</b>

# List of figures

1.1	Using Search Engine . . . . .	4
2.1	Cloud Service Model . . . . .	16
3.1	Architecture of the Cloud Service categorization framework . . . . .	38
3.2	Cloud Service Main Concepts . . . . .	42
3.3	Software as a Service (SaaS) . . . . .	42
3.4	Platform as a Service (PaaS) . . . . .	43
3.5	Infrastructure as a Service (IaaS) . . . . .	44
3.6	Illustration on the Cloud Service Categorization Procedure . . . . .	50
4.1	The first 100 searching results from two most popular search engines (Google and Bing) using different keywords, such as cloud service, cloud storage, cloud service provider, cloud hosting, cloud software, cloud platform and/or cloud infrastructure to search cloud services . . .	56
4.2	Cloud Service Identifier Framework . . . . .	58
4.3	lof . . . . .	66
4.4	Cloud Service Modelling . . . . .	70
5.1	Architecture of the Cloud Service Discovery Engine . . . . .	77
5.2	Cloud Service Advertisement . . . . .	82
5.3	Cloud Service Locations 2013 . . . . .	86
5.4	Cloud Service Locations 2016 . . . . .	87

---

5.5	Cloud Service Categorization . . . . .	88
5.6	Service-Oriented Computing . . . . .	89
5.7	Cloud Service Advertisement . . . . .	90
5.8	Cloud Service IP . . . . .	91
6.1	Identifying E Threshold . . . . .	100
6.2	Upper Bound Threshold . . . . .	102
6.3	Lower Bound Threshold . . . . .	102
6.4	Noisy Data of Cloud Service Search Engine . . . . .	103
6.5	Cloud Service Categorization . . . . .	105
6.6	Cloud Service Categorization Type . . . . .	106
6.7	Identifying Cloud service . . . . .	107
6.8	Robustness Cloud Service Categorization . . . . .	107
6.9	Precision and Recall under Jaccard and Semantic features . . . . .	109
6.10	Term features classification accuracy (%) using k-nearest neighbor classification, with LSA dimension reduction cloud service. . . . .	111
6.11	Precision and recall using k-nearest neighbor classification, with LSA dimension reduction cloud service . . . . .	112
6.12	Bing Normal Search . . . . .	117
6.13	Bing Advanced Search . . . . .	117
6.14	Google Normal Search . . . . .	117
6.15	Google Advanced Search . . . . .	117
6.16	Cloud Service Profile . . . . .	118
6.17	Cloud Service Profile Features . . . . .	119



# List of tables

2.1	Cloud Service Ontology . . . . .	21
2.2	Cloud Service Language . . . . .	26
2.3	Cloud Service Model . . . . .	31
4.1	Cloud Service features . . . . .	69
5.1	Cloud service Collection Dataset . . . . .	80
5.2	Breakdown of Cloud Services Crawling Results . . . . .	83
5.3	Error Codes for Inactive Cloud Services . . . . .	83
5.4	Cloud Service Location . . . . .	85
6.1	Top 10 High Frequency Concepts in CSO . . . . .	99
6.2	Learning cloud service identification: SVM Accuracy, Precision and Recall . . . . .	109
6.3	Comparison results between Google and Bing . . . . .	115
6.4	Comparison results between search engine and our identifier . . . . .	115



# Chapter 1

## Introduction

### 1.1 Introduction

Over the past few years, cloud computing has become more attractive as a computing paradigm due to its high flexibility in provisioning on-demand infrastructures, platforms and software as services through the Internet. Historically, cloud computing has combined concepts, such as grid computing [26], service-oriented computing [115] and virtualization techniques [17] to both exploit and deliver computer resources (data centres). Regarding grid computing, a computer's resources are collected from several locations to obtain a goal, such as high CPU performance. As to service-oriented computing, computer resources are both managed and delivered as services. However, cloud computing uses virtualisation techniques to manage, deliver and share computer resources as services via highly dynamic resource provisions. This adoption has created numerous benefits, including high performance, reduced costs, automation and self-provisioning, scalability, accessibility and availability. For instance, the New York Times paid \$240 to archive 11 million articles (1851–1980) in 24 hours by using Amazon Web Services [78].

In cloud computing, many research efforts have focused on data processing and migration [40, 65, 110], security [98, 104], privacy [125, 113], and trust management

[80, 79]; however, cloud service discovery is still encountering challenges regarding finding appropriate services for cloud users on the World Wide Web (WWW). Indeed, according to researchers [103, 34, 49, 15], the tasks of discovering and selecting a suitable cloud service represent only a few obstacles among many in adopting cloud computing. In addition, the unique characteristics of cloud services, such as that they are highly dynamic, distributed and non-transparent as well as that diverse services are offered at different levels and that they lack a standardized description, make cloud service discovery and selection even more challenging.

An effective service discovery system will help cloud service consumers and providers gain the benefits that are available via cloud computing technologies. However, to discover a cloud service on the WWW, potential customers normally rely on a search engine, which is a tedious task because the search results provide a large quantity of irrelevant results, including news, blogs, journal papers, wikis, articles etc. This is because the term “Cloud” is a general and widely-used term.

This chapter is organized as follows. In Section 1.1, we illustrate a motivating scenario, which will be used as an example throughout this dissertation. In Section 1.2, we outline the research issues tackled in this dissertation. In Section 1.3, we summarize our contributions in addressing the research issues and in Section 1.4, we enumerate the publications by the author that are related to this work. Finally, in Section 1.5, we describe the structure of this dissertation.

## **1.2 Motivating Scenario**

In this dissertation, we explore many research issues regarding both discovering and selecting a cloud service, with our focus on detecting cloud services on the WWW. Despite the fact that the proposed approaches are generic enough to be applicable on a wide range of applications, this motivating scenario can be used as a practical example. Cloud service providers who offer either one or more cloud services, including

either Infrastructure as a Service (IaaS), Platform as a Service (PaaS), Software as a Service (SaaS) or a combination thereof, are publicly available on the Internet, and they advertise their cloud services via search engines [82]. Cloud service consumers often use popular search engines to find a suitable cloud service provider that meets their requirements.

However, search engines can be tedious (as shown in Fig1.1). According to Deursen et al. [111] 91% of internet searchers who use search engines do not go past the first page of search results. Also, the study shows that more than 50% do not get past the first three results on page 1. Furthermore, the term ‘cloud‘ is quite general and therefore widely used. For example, “Cloud” is one of the most important and popular terminologies used on websites that are related to meteorology. Moreover, many websites that discuss cloud services are not providers of them. Finally, certain businesses that have nothing to do with cloud computing use the term “Cloud” in their names and/or service descriptions (e.g., cloud9carwash<sup>1</sup>).

This motivating scenario poses several major concerns including the following: i) how to distinguish cloud services from other services available on the Internet, ii) how to develop the most effective unified discovery approach of cloud services by efficiently identifying and detecting cloud services and their attributes to enhance cloud service selection and iii) development of an automatic cloud services crawler to maintain an up-to-date cloud services repository that allows consumers to search for new cloud services.

---

<sup>1</sup><http://www.cloud9carwash.com/>

Google cloud service providers

All Images News Shopping Videos More Settings Tools

About 18,800,000 results (0.96 seconds) **1**

**Best 10+ Cloud Service Providers - 2017 Reviews | Clutch.co**  
<https://clutch.co/cloud>  
 Feb 11, 2016 - Find the most popular **cloud service providers**. Read 590 detailed customer reviews of 23 leading cloud computing solutions.  
 Top 10+ Cloud Servers · Cloud Storage · Top Microsoft Azure Partners · AWS

**What is cloud provider? - Definition from WhatIs.com** **2**  
[searchcloudprovider.techtarget.com](https://searchcloudprovider.techtarget.com) > Cloud Networks > ERP  
 A **cloud provider** is a company that offers some component of **cloud computing** – typically Infrastructure as a **Service** (IaaS), Software as a **Service** (SaaS) or Platform as a **Service** (PaaS) – to other businesses or individuals. **Cloud providers** are sometimes referred to as **cloud service providers** or CSPs.

**Cloud Service Providers | Equinix**  
[www.equinix.com.au/industries/cloud-providers/](http://www.equinix.com.au/industries/cloud-providers/)  
 That's why 500+ **cloud** companies around the world have made Equinix their home. **Cloud** Infrastructure (IaaS and PaaS) Equinix provides more than colocation for **cloud** infrastructure **providers** – we provide peace of mind. Software **Services** (SaaS and ISVs) Managed **Cloud** and IT **Services**.

**What are Cloud Service Providers? - Definition - - SDxCentral**  
<https://www.sdxcentral.com> > Cloud Infrastructure > Cloud Resources  
 See how **Cloud Service Providers** (CSPs) deliver software and network applications from data centers and are changing how enterprises deliver applications.

**Category:Cloud computing providers - Wikipedia** **3**  
[https://en.wikipedia.org/wiki/Category:Cloud\\_computing\\_providers](https://en.wikipedia.org/wiki/Category:Cloud_computing_providers)  
 Pages in category "**Cloud computing providers**". The following 200 pages are in this category, out of approximately 262 total. This list may not reflect recent ...

**Cloud Service Providers - Trend Micro**  
[www.trendmicro.com.au](http://www.trendmicro.com.au) > Home > For Business > Service Providers  
 Sell your customers the confidence of industry-leading **cloud** security ... the **cloud** and gives you a competitive advantage over other **service providers** with less ...

**Secure Australian Cloud Provider of Cloud Products and Services**  
<https://www.cloudcentral.com.au/>  
**Provider** of Australian cloud infrastructure and platforms that enable our partners to be successful. For flexible, scalable, reliable and secure **cloud computing**.

**Cloud computing - Wikipedia**  
[https://en.wikipedia.org/wiki/Cloud\\_computing](https://en.wikipedia.org/wiki/Cloud_computing)  
**Cloud computing** is a computing infrastructure and software model for enabling ubiquitous .... In the PaaS models, **cloud providers** deliver a computing platform, typically including operating system, programming-language execution ...  
 You've visited this page 2 times. Last visit: 24/07/17

**The 10 Most Important Companies In Cloud Computing | Business ...** **4**  
<https://www.businessinsider.com.au/10-most-important-in-cloud-computing-2013-4>  
 Apr 20, 2013 - Before we get to the list, let's define **cloud computing**. There are three types: Software-as-a-Service (SaaS), Platform-as-a-Service (PaaS) and ...

**Top 10 Cloud Computing Companies in Australia | Top 10 | Business ...**  
[www.businessreviewaustralia.com/.../Top-10-Cloud-Computing-Companies-in-Austra...](http://www.businessreviewaustralia.com/.../Top-10-Cloud-Computing-Companies-in-Austra...)  
 Apr 17, 2016 - With **cloud computing** and services becoming more widespread and complex, ... Although several new software **providers** continue to enter the ...

Gooooooooooogle > **5**  
 1 2 3 4 5 6 7 8 9 10 Next

Fig. 1.1 Using Search Engine

## 1.3 Research Issues

Based on observations of the aforementioned motivating scenario, the need for efficient discovery of cloud services is based on the four key issues that follow.

### **Cloud Service Variety**

Cloud services are offered at different levels, such as IaaS (e.g., storage, hosting, vps and dedicated server), PaaS (e.g., libraries and api) and SaaS (e.g., Virtual Desktop and CRM). In addition, cloud services are delivered via different deployment models. The availability of various services and deployments increases the difficulty of discovering an appropriate cloud service [62].

### **Lack of Standards**

Cloud service is different than Web service because the lack of standards for describing and publishing cloud services [88]. Unlike Web services which use standard languages such as the Web Services Description Language (WSDL) or Unified Service Description Language (USDL) to expose their interfaces and the Universal Description, Discovery and Integration (UDDI) to publish their services to services' registries for discovery, In real environment the majority of the publicly available cloud services are not based on description standards which makes the cloud service discovery a challenging problem. This lacking standards for describing and publishing cloud services is made by the vendor lock-in problem [86], which limits the interoperability between cloud services.

## **Dynamic Nature of Cloud Services**

In real environment, cloud services are essentially, dynamic which means that new cloud services can be provided on the Web, while old cloud services could end up disappearing from the online scene (e.g., Nirvanix is cloud service provider that was shut down in 2013 due to bankruptcy [60]). Furthermore, cloud services have the capability to change over time due to evolving business demands. Therefore, in order to keep the cloud service discovery updated, it must have the capability to periodically and proactively visit cloud services.

## **Limitation of Search Engines**

Traditional search engines are not designed to find cloud service providers due to the aforementioned motivation scenario. In addition, general search engines are not appropriate for the provision of details about cloud service features (e.g. cloud service type, process limitation, storage maximums, memory capacity etc.), which leads to ambiguity in selecting a cloud service.

## **1.4 Contributions Overview**

We focus on the design and implementation of a cloud service discovery engine, including unified discovery, which helps distinguish between cloud services and other services that are available on the Internet. Firstly, we propose a cloud service categorization framework that includes cloud service ontology to categorize cloud services. Secondly, we propose a cloud service identifier framework that can automatically identify and determine cloud service attributes in a real environment. Finally, we propose an automatic cloud service discovery engine that could exploit both frameworks to automatically identify and categorize cloud services. The contributions of our work are detailed in the following sections.



### **1.4.1 Cloud Service Ontology**

We build a comprehensive ontology for reasoning during cloud service discovery, which is semi-automatic based on mining new concepts, after analysing real-world cloud services. Our approach involves two steps: creating a base ontology by following the US National Institute of Standards and Technology's (NIST) cloud computing standards, which we also call a cloud service ontology roadmap because limited number of concepts in this ontology, and developing an algorithm to analyse these cloud services and identify new cloud service concepts that could generate our cloud service ontology. Our ontology automatically enhances crawling procedures and both the identification and the categorization models [3].

### **1.4.2 Cloud Service Categorization Model**

We build a cloud services categorization model that is based on cosine similarity approach. Our model contains a processor that automatically categorizes given cloud service providers based on the service model (IaaS, PaaS and SaaS). This model relies on the cluster method to build the categorization, which can be updated automatically after categorizing a new cloud service provider to increase categorization knowledge. Furthermore, the model uses term frequency to weigh cloud service ontology concepts and calculate cosine similarity to measure the similarities between cloud services. We then design and develop cloud service categorization algorithm that divide cloud services into clusters [4].

### **1.4.3 Cloud Service Identifier Model**

The cloud service identifier model identifies features to determine whether a given source is a cloud service. This model relies on the classification method to build the identifier, which can be updated automatically after identifying a new cloud service provider to enhance the identifier knowledge. To identify cloud services in a real en-

vironment, we focus on learning features that can be shared by large cloud service providers on the web to predict whether the given web source provides cloud service. We discover a wide range of features between cloud service providers inside their sources, such as similarity of function, cloud service semantic concepts and cloud service components, and use those as identification features to train on learning models. Learning models are fed into the classification algorithm to detect cloud service providers. In addition, our identifier model can build a cloud service profile, which includes details about the service and its attributes (e.g., price, memory, CPU, storage and OS).

#### **1.4.4 Dataset of Cloud Services Collection**

From what we have seen, we consider that in order to gain invaluable knowledge of the technologies used in the field of cloud services, there is an urgent need to identify the present state of cloud services. Therefore, we collect, identify and analyze all cloud services that are available on the Web. We have used the cloud service discovery engine to undertake this task twice: in 2013 and 2017. Our automatic discovery managed to fetch 619,474 possible links and collected 35,601 possible Web sources for cloud services in 2013; by contrast, in 2017, the cloud service discovery engine managed to fetch 844,696 possible links and collect 55,474 possible Web sources for cloud services. From the data collected, the analysis was carried out, and the results landscape provided a comprehensive overview of the status of cloud services. Also, we prepared a large dataset of real-world cloud services that are available on the Web. Also, we released the collected dataset to the research community. These dataset includes 7,461 cloud services and nearly 15,000 services attributes (2.45 GB) [84].

### 1.4.5 Implementation and Performs Study

We implemented our proposed frameworks for discovering cloud services via cloud service crawling. We developed a unified platform for automatic cloud service discovery to identify and categorize cloud services. We conducted extensive experiments and performance studies of the proposed approaches using real-world cloud services available on the Web in order to validate the feasibility and benefits of our approaches. The results demonstrated the applicability of our approach and its ability to effectively identify and categorize cloud services on the Web [5, 4, 3].

## 1.5 Dissertation Publications

In the following, we include the papers from my work related to this dissertation. The list of papers, including all accepted, revised and submitted manuscripts is as follows:

### Journals

- Automated Discovery of Cloud Service in Real Environment. Abdullah Alfazi, Quan Z. Sheng, Ali Babar and Talal. H. Noor. submitted Knowledge-Based Systems
- Generating Cloud Service Ontology to Automatically Identify Cloud Service Categorization for Enhanced Cloud Service Discovery. Abdullah Alfazi, Quan Z. Sheng, Ali Babar, Yongrui Qin and Talal. H. Noor. Under review by Grid Computing.

### Conferences

- Toward Unified Cloud Service Discovery for Enhanced Service Identification. Abdullah Alfazi, Quan Z. Sheng, Ali Babar, Wenjie Ruan and Yongrui Qin. The

Sixth Australasian Symposium on Service Research and Innovation (ASSRI 17)  
. Sydney, October 19-20 2017.

- Identification as a Service: Large-Scale Cloud Service Discovery Over the World Wide Web. Abdullah Alfazi, Quan Z. Sheng, Wei Emma Zhang, Lina Yao, and Talal H. Noor. The 5th IEEE International Congress on Big Data. San Francisco, USA, June 27-July 2, 2016. (Chapters 4,6)
- Ontology-based Automatic Cloud Service Categorization for Enhancing Cloud Service Discovery. Abdullah Alfazi, Quan Z. Sheng, Yongrui Qin, and Talal H. Noor. The 19th IEEE International EDOC Conference (EDOC 2015). Adelaide, Australia, September 21-25, 2015 (Chapters 3,6)
- Towards Ontology-Enhanced Cloud Services Discovery. Abdullah Alfazi, Talal. H. Noor and Quan Z. Sheng, and Yong Xu. the 10th International Conference on Advanced Data Mining and Applications (ADMA 2014). Guilin, China, December 19-21, 2014 (*Best Paper*) (Chapters 3,6)
- CSCE: A Crawler Engine for Cloud Services Discovery on the World Wide Web. Talal. H. Noor, Quan Z. Sheng, Abdullah Alfazi, Anne H.H. Ngu, and Jeriel Law. The IEEE 20th International Conference on Web Services (ICWS 2013). Santa Clara Marriott, CA, USA, June 27 to July 2, 2013. (chapters 5,6)
- Cloud Armor: A Platform for Credibility-based Trust Management of Cloud Services. Talal. H. Noor, Quan Z. Sheng, Anne H.H. Ngu, Abdullah Alfazi, and Jeriel Law. The 22nd ACM Conference on Information and Knowledge Management (CIKM 2013). San Francisco, CA, USA, October 27-November 1, 2013.
- Detecting Occasional Reputation Attacks on Cloud Services. Talal. H. Noor, Quan Z. Sheng, and Abdullah Alfazi. The 13th International Conference on Web Engineering (ICWE 2013). Aalborg, Denmark, July 8-12, 2013.

- Reputation Attacks Detection for Effective Trust Assessment of Cloud Services. Talal. H. Noor, Quan Z. Sheng, and Abdullah Alfazi. The 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom 2013). Melbourne, Australia, July 16-18, 2013.
- Identifying Fake Feedback for Effective Trust Management in Cloud Environments. Talal H. Noor, Quan Z. Sheng, Abdullah Alfazi, Jeriel Law and Anne H.H. Ngu. The 1st International Workshop on Analytics Services on the Cloud (ASC 2012). Shanghai, China, November 12-16, 2012.

## 1.6 Dissertation Organization

The remainder of this dissertation is organized as follows. In Chapter 2, we present an overview of the cloud service models and discovery techniques and survey the representative research prototypes that efficiently support the discovery of cloud services. In Chapter 3, we present the overall view of the proposed cloud service categorization framework. We firstly provide details of a novel approach to build the cloud service ontology. Secondly, We propose the use of term frequency to weigh cloud service ontology concepts and calculate cosine similarity to measure the similarity between cloud services. We then design and develop cloud service categorization algorithms that categorise cloud services into clusters. In Chapter 4, we present the overall view of the proposed cloud service identification framework. We firstly introduce the cloud service identifier, which was built by utilizing cloud service features that were extracted from real cloud service providers. Secondly, we propose a novel Service Detection and Tracking (SDT) methodology for the detection of cloud services. Finally, we build a unified model to expose the cloud service's features to ease the process of searching and comparing many cloud services. In Chapter 5, we introduce the architectural design of the Cloud Service Discovery Engine (CSDE).

We discuss the cloud service crawling procedures as well as the design challenges of the crawler because the automatic discovery of cloud services is not a straightforward task. Finally, we report the results of a set of statistical analyses regarding the collected datasets, which offer an overall view on the status of cloud services. In Chapter 6, we describe the implementation of our approach for the cloud service categorization and identification frameworks. We then report the results of several experimental evaluations and performance studies on our model. Finally, in Chapter 7, we summarize the key points of each chapter of this dissertation then we discuss directions for future research.

# Chapter 2

## Background

In this chapter, we present an introduction to research fields related to the service discovery in cloud environments to help readers gain a better understanding of the work described in this dissertation. In addition, we overview particular service discovery and cloud service discovery and selection techniques, present a general comparison study to recent research and recent research on cloud service discovery approaches.

This chapter is organized as follows. In Section 2.1 and Section 2.2, we present an overview cloud services and their service and deployment models and service discovery. In section 2.3, we present and compare the existing cloud service discovery approaches that include different techniques such as ontologies, languages and modeling. Finally, we summarize this chapter in Section 2.4.

### 2.1 Overview of Cloud Computing

Cloud computing is an approach to increasing capacity and capabilities of IT networks by centralizing how data is stored and processed. It allows consumers to access applications without first installing them, as well as, increases access to personal information access over the Internet. Furthermore, cloud computing has led to reduced costs of building IT infrastructure and acquiring new resources. Cloud computing

service computers benefit from the multitenant architecture by maintaining one application. Additionally, cloud services are defined by five essential characteristics [62]:

- **On-demand self-service:** A user of cloud service is able to possess independent computing capabilities such as server, network or storage, whenever necessary, without depending on human interaction.
- **Broad network access:** Capabilities can be accessed on-line and through standard mechanisms which encourage the use of various client platforms such as mobile phones, tablets, laptops and workstations.
- **Resource pooling:** The provider's computing resources are shared among multiple consumers under a multi-tenant model, where dissimilar physical and virtual resources are engaged or disengaged according to consumer preferences. In normal circumstances, consumers do not have the knowledge of the precise location apart from information of a higher level of abstraction (e.g., country, state, or datacenter), that provided such resources. Resources include storage, processing power, memory and network bandwidth.
- **Rapid elasticity:** Resources are supplied elastically or released automatically/manually, depending on the demand. The capabilities seen by the consumers often appear to be unlimited and can be utilized at any quantum and time.
- **Measured Service:** Cloud services provided may charge according to a pay-per-use or charge-per-use basis. Resource usage of the service are monitored, controlled and reported, allowing transparency between the provider and the consumer.

Based on the definition provided by the National Institute of Standards and Technology (NIST) [63], cloud computing can be defined as follows:



**Definition 1** (Cloud Computing). *Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model is composed of five essential characteristics, three service models, and four deployment models.*

### 2.1.1 Cloud Service Models

Cloud services consist of three different service models, which include Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). This service model is based on different Service Level Agreements (SLAs) between a cloud service provider and a consumer [13, 18, 81, 63]. Figure 2.1 depicts the structured layers of cloud services:

- **Software as a Service (SaaS):** The term is referred to a service provided by a cloud application supported on a cloud infrastructure. Consumers can interact the application through a user interface like a Web browser installed on different client devices. It is superfluous for the consumers to manage or manipulate any underlying cloud infrastructure such as the servers or operating systems. However, they may be given the control limited to particular application configuration settings.
- **Platform as a Service (PaaS):** PaaS is another type of service that is built upon the cloud infrastructure and involves consumer developed or acquired applications with the help of programming languages, tools, libraries and services powered by the provider. The consumer is not required to manage or configure the cloud infrastructure (i.e., network, server, storage) except possessing the power to control the deployed applications and configuration settings of the hosting environment for which the applications are in.

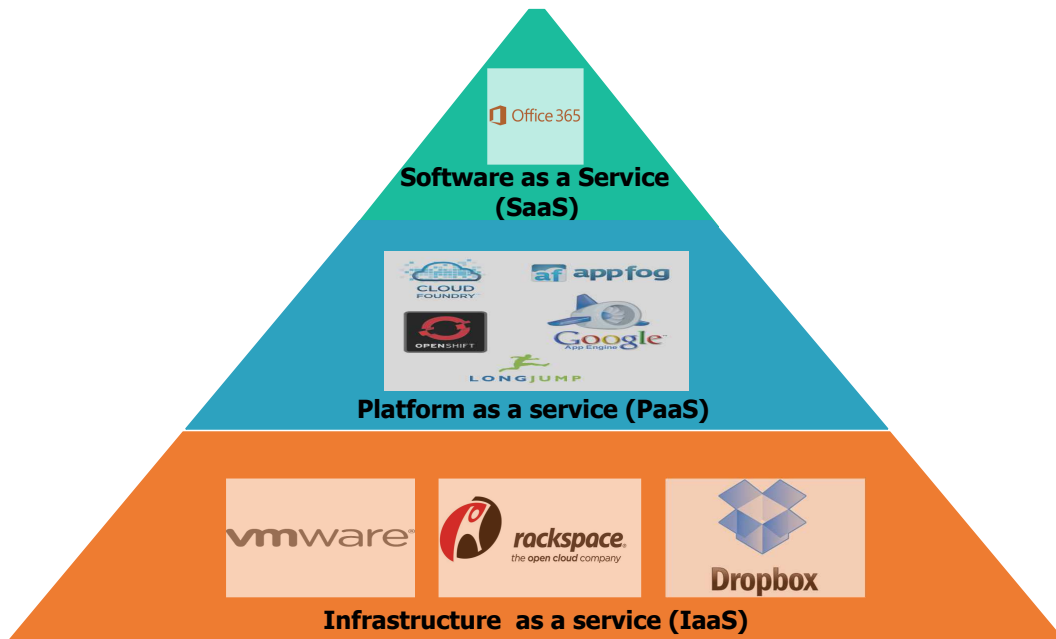


Fig. 2.1 Cloud Service Model

- **Infrastructure as a Service (IaaS):** It is defined as the ability to supply the consumer with resources from the ground up - processing, storage, network and other basic resource, in order to allow consumer to develop and run software as well as operating systems and applications. The consumer need not to manage or configure the cloud infrastructure but has authority over controlling the operating systems, storage and other mounted applications with the possibility of limited control over selected networking components like the host firewalls settings.

## 2.1.2 Deployment Models

All cloud service model (SaaS,PaaS,IaaS) can provide through four different cloud service deployment models, namely Private, Community, Public, and Hybrid [13, 18, 81, 63]. relying on the cloud service customer's requirements.

- Private cloud: Refers to the cloud infrastructure that is owned, managed and run by a single organization, a third party or a combination of the two. This cloud is for exclusive use and may be located on or off sites.
- Community cloud: Is defined as a cloud infrastructure shared among one or more organizations in the community, a third party or a combination of them. This cloud is also for exclusive use and may be located on or off sites.
- Public cloud: Public clouds can be used by the general public. They are either owned, managed and operated by a business, academic or government organization or a combination of them. This cloud is located in the sites of the cloud provider.
- Hybrid cloud: A hybrid cloud is a combination of two or more unique cloud infrastructures from private, community or public clouds and adhere to the standards or patented technology which allows portability of data and applications.

We argue that there is no single service discovery approach that can solve all the challenges facing cloud services - merely by looking at service and deployment models within cloud environments. A service discovery may be independent of cloud services, but the approach must be in congruence with cloud service models. We believe that it is important to define the possible service discovery approaches that identify the types of cloud services ideally suited to address a particular concern and decipher which of these techniques can impart a better understanding on the most appropriate service discovery in the context of cloud service. In the following section, we present the overview of service discovery, give examples of service discovery and its approaches and present several examples for cloud service discovery system.

## 2.2 Overview of Service Discovery

Service discovery has been very active research field attracting many researchers around the world as a fundamental problem in many research areas, such as ubiquitous computing [24, 27], mobile ad-hoc networks [68, 19], peer-to-peer (P2P) [66, 41], service oriented computing [115, 91] and Web service [36, 118]. The Web service discovery has been conducted many pieces of research in the last decade. In addition, The Web service can be defined as set of standards that aim to share data between different software applications and distribute services on the Internet [37]. Web services use standard languages such as the Web Services Description Language (WSDL) or Unified Service Description Language (USDL) to expose their interfaces and the Universal Description, Discovery and Integration (UDDI) to publish their services to services' registries for discovery[84]. However, there is a limitation to find a web service during service discovery process. This is because WSDL has data constraint and weakness of semantic support.

Several approaches have been proposed to discover Web service such as Semantic matching, decentralized distributed and information retriever. Firstly, semantic matching [112, 73] approach has been used to overcome the problems in traditional discovery process, which included a Web service customer who only used keywords to search in UDDI, which in turn defined a registry for service providers to publish their services; these keywords are often not able to find all the related services [76, 126, 87, 117]. Furthermore, decentralized distributed approaches are mainly based on Peer-to-peer (P2P) that are proposed in Web service discovery using a Chord P2P protocol as overlay network [41, 123, 53, 100] to. Finally, Information Retriever and Ontology have also applied a role in Web service discovery [87, 51]. Segev and Sheng [101] develop a bootstrapping approach for building Web services ontology. Their approach exploits Term Frequency/Inverse Document Frequency (TF/IDF) and

Web context generation to automatically build the ontology for describing the Web services functionality.

Service discovery can be defined as the ability to identify hardware or software service automatically in networks such as a scanner, printer, Web server or shared file. Service discovery systems can use registry system to register services in a centralize repository (e.g., UDDI and Web services). It can also provide a technique to query about every service in a network (e.g., UPnP and Zeroconf) [59].

## 2.3 Cloud Service Discovery

At the beginning of cloud computing, many researchers focused on study the classification of this new technology. They offer some solutions on the discovery and selection of cloud services. However, each cloud service provider uses a different approach to publish his service and offers a distinct interface in reality. Therefore, there is not any discovery approach that can be fully automated and effective enough to discover cloud service in real environment. In addition, the unified discovery approach should be able to discover all cloud service models (IaaS, PaaS, and SaaS). However, different approaches have been proposed to discover cloud services such as ontologies [122, 116, 46, 39, 124, 39, 50], languages [32, 56, 105] and models [90, 7, 54]. In this section, we discuss these approaches in details.

### 2.3.1 Cloud Service Ontology

Ontology-based approach has been used by many researchers for service discovery [67, 116, 22, 39, 124]. An ontology can be defined as “*a formal and explicit specification of a shared conceptualization*” [34]. Therefore, we provide the current status of research on cloud service ontology. In the cloud service domain, there have been several attempts to build an ontology for cloud services for both discovery and selection

purpose. Moreover, many of these studies aim to select the best cloud service based on cloud service user requirements. Furthermore, several of these studies have conducted their experiments on simulated cloud service data. Table 4.1 compares several recent studies of using cloud service ontology to discover, select or recommend cloud services. Also, it shows the domain coverage, the ability to be hierarchy taxonomy, how the ontology is built, which language is used, the evaluation and ability to work on a large scale of cloud services.

Youseff et al. [122] build an ontology that relies on classifying cloud computing based on its components, which consists of five layers: the *applications*, the *software environment*, the *software infrastructure*, the *software kernel*, and the *software hardware*. Each layer can contain one or more services depending on the level of abstraction. Also, each layer relies on computing concepts to measure limitations and strengths.

Weinhardt et al. [116] propose to build an ontology based on a cloud business ontology model. This ontology model consists of three layers: the *platform*, the *infrastructure* and the *application*, as well as a content pricing model to help clarify the relationship between cloud service providers and customers.

Kang and Sim [46] propose a cloud service discovery system that uses an ontology-based approach to discover cloud services close to users' requirements. However, cloud service providers still need to register in the discovery system in order to publish their cloud services. Furthermore, their work relies on software agents to perform reasoning tasks (e.g., similarity reasoning, equivalent reasoning and numerical reasoning).

Ontology	Aim	Domain	Taxonomy	Automation	Language	Evaluation	Large-Scale
[122]	C	Cloud Service	✓	Manual	<i>x</i>	No evaluation	<i>x</i>
[116]	C	Cloud Service	✓	Manual	<i>x</i>	No evaluation	<i>x</i>
[46]	D	Cloud service	✓	Manual	<i>x</i>	Simulation	<i>x</i>
[125]	R	IaaS	✓	Manual	Ontology (OWL)	Only some IaaS	<i>x</i>
[20]	D	IaaS	✓	Manual	WSMO	Case study	<i>x</i>
[58]	S	Cloud Service	✓	Manual	Ontology (OWL)	Simulation	<i>x</i>
[28]	D	ICT	✓	✓	Ontology (OWL)	Limited data	<i>x</i>
[43]	D	Cloud Service	✓	Manual	Tree structure	Only some CS	<i>x</i>
[1]	D	SaaS	✓	Manual	Ontology (OWL)	Simulation	<i>x</i>
[21]	S	Cloud Service	<i>x</i>	Manual	Ontology (OWL)	Case study	<i>x</i>
[25]	R	Cloud Service	✓	Manual	Ontology (OWL2)	Case study	<i>x</i>
[22]	D	PaaS	<i>x</i>	Manual	Ontology (OWL-S)	Windows Azure	<i>x</i>
[107]	S	Cloud Service	✓	Manual	Ontology (OWL)	Simulation	<i>x</i>
[93]	S	Cloud Service	<i>x</i>	Manual	Ontology (OWL)	Case study	<i>x</i>
[67]	S	IaaS	<i>x</i>	Manual	Ontology (OWL2)	Simulation	<i>x</i>
[74]	D	Cloud Service	✓	Manual	Ontology (OWL)	Case study	<i>x</i>

D:Discovery S:Selection R:Recommendation

Table 2.1 Cloud Service Ontology

Han and Sam [39] design A CSDS (Cloud Service Discovery System) to locate cloud services across the Internet. Further, they build a framework for cloud ontology which would enhance CSDS performance. The planned ontology consists of 424 models. In principle, this attempt at producing an agent-based cloud discovery system, one which accesses ontology in collecting information relating to cloud services.

Kang and Sim[50, 47, 48] propose Cloudle which is a search engine that uses a Cloud ontology for reasoning about the relations among Cloud services. they develop three types of reasoning methods, (1) concept similarity reasoning, (2) object property similarity reasoning, and (3) datatype property similarity reasoning. The Cloudle uses agent-based to search about the service in the search engine.

Yoo et al. [120] propose an approach to select cloud services that best meet a user's requirements by using a cloud ontology based on resource services. The authors use the similarity computing degree of virtual cloud service physical resources to determine the best cloud services for users.

Ma et al. [58] propose an ontology-based resource management approach for cloud providers. This cloud computing ontology defines the concepts for describing their relations. However, the ontology developed to meet cloud service requirements is conducted in a simulated environment.

Rodríguez-García et al. [96, 95] propose an automatic general ICT domain that can be used to discover the cloud services that best match user needs. The authors use semantic annotation in order to improve the cloud service discovery results. From cloud service descriptions, semantic content can be extracted by using the annotation platform. Then, the semantic content can be used by the semantic search engine to assist users in finding those services that meet with their requirements and expectations.

Deng et al. [21] create an ontological framework approach to demonstrate the structure and illustration of general cloud services and their methods of operation.



The concept of secure systems with the framework to deliver cloud service business support was introduced.

Fang et al. [25] propose the Agility-oriented & Fuzziness-embedded (AoFeCSO). It is a semantic cloud service model, and its design is agility-centered with Semantic Web Ontology Language (OWL2) fuzzy extensions. Each time there is user interaction, the fuzzy-ontology receives a rating update. Consequently, it enables the specified cloud services to be collectively sustained. Users can not only collaboratively contribute to it with their knowledge, but they can also examine the model. Thus, improving the overall operation of the cloud service data. This process permits cloud-oriented information and interactions to be gathered, which enables wide-ranging service provisions.

Rekik et al. [93] propose a description ontology that included IaaS, PaaS, and SaaS. The operational and non-operational properties of the ontology, the characteristics and infrastructure associations, the software services and platform are used to locate the appropriate cloud services.

Modica et al. [23] propose a group of ontologies focused from a business point of view as opposed to a technical one. This group assisted providers in better describing and advertising their services about business plans. Furthermore, consumers are only able to define the resources they need by the activities they undertake. With that in mind, the following ontology, Applications, Support, SLA, Market, Offer, and Request have been created in line with customer demand, and organizational viewpoints of the services providers offer.

Nagireddi et al. [74] propose an ontology epitomizing cloud services and its associated attributes. It is used to assist cloud service discovery apropos user requirements. AHP (Analytical Hierarchy Process) is utilized for ranking cloud services, by way of cloud service attribute categorization.

Joshi et al. [44] propose ontologies outlining the models and relationships in the lifespan of IT services. Further, the lifecycle of IT services was divided into

five stages: requisites, discovery, negotiation, configuration, and use. Each stage was described and defined as compliant with the relevant ontologies and utilized to epitomize services and requirements.

Moscato et al. [72] propose an ontology which could be utilized to expedite the operation ability across cloud service platforms and cloud service discovery. This mOSAIC ontology has been created through analysis of cloud principles and applications.

Tahamtan et al. [107] propose a cloud ontology framework. This framework locates and stores cloud services in accordance with functionality and non-functionality of user needs. It is capable of describing Saas, Paas and Iaas, cloud service models. Furthermore, the combined ontology is developed for both cloud service providers and business purposes.

### **Cloud Service Ontology based IaaS Model**

Zhang et al. [125] proposed a Cloud Recommender system to select cloud infrastructure based on their cloud computing ontology called CoCoO. This ontology defines functional and non-functional concepts of infrastructure services with their attributes and relations. However, the ontology does not provide any details about PaaS and SaaS. In addition, the validation of ontology concepts is only limited to some cloud infrastructure providers, such as Amazon, Microsoft Azure, and GoGrid.

Dastjerdi et al. [20] propose an approach that uses ontology-based discovery for QoS-aware deployment of appliances on IaaS providers. This approach supports end users to meet their needs from a range of IaaS providers based on QoS preferences. However, the ontology is designed only to find suitable IaaS providers for end users. Furthermore, the ontology does not support PaaS and SaaS providers.

Metwally et al. [67] propose a unified IaaS ontology system that is able to describe, and discover several of computer resources relying on IaaS requests. The

ontology model is used an efficient reasoning capability to prevent any repetition or conflict in the unified ontology model.

### **Cloud Service Ontology based SaaS**

Afify et al. [1] propose a semantic system focus on Software as a Service (SaaS). This system eases publishing, discovering and selecting SaaS by using cloud ontology that concentrates on SaaS aspects. The unified ontology is developed to integrate service domain knowledge, QoS metrics, SaaS characteristics, and real SaaS offers.

### **Cloud Service categorization**

In cloud service categorization, Hofer and Karagiannis [43] build a tree structure taxonomy that is used to classify cloud services. This hierarchical taxonomy has been built on top of cloud service characteristics. However, the classification is limited in enhancing cloud service discovery and categorization because it only considers a few cloud services such as Google apps.

Zeng et al. [124] propose an approach to determine the interoperability between two given Cloud services by building a service matching algorithm and a service composition algorithm.

## **2.3.2 Cloud Service Language**

Firstly, cloud services have been thoroughly considered in many research studies as Web services. Therefore, the WSDL language has been used to describe cloud services. Moreover, cloud services utilize the same Web service parameters, such as inputs, outputs, service name, service location, protocols. WSDL is a commonly applied representation that gives XML-based standard interface description for software components implemented in various languages. WSDL has the capability of describing Web services from functional standpoints such as interaction interface,

Study	Aim	Domain	languages	Automation	Evaluation
[127]	D	SaaS	WSDL-S	Manual	Simulation
[33]	D	SaaS	WSDL	Manual	Case study
[56]	M	IaaS	Cloud#	Manual	<i>x</i>
[32]	M	IaaS	XML	Manual	Case study
[119]	D	Cloud Service	XML and OWL	Manual	Simulation
[85]	M	Cloud Service	USDL	Manual	Case study
[42]	S	Cloud Service	USDL	Manual	<i>x</i>
[45]	M	Cloud Service	USDL	Manual	Case study
[105]	M	Cloud Service	USDL	Manual	Case study
[102]	D	IaaS	XML	Manual	Case study

D:Discovery S:Selection M:Modeling

Table 2.2 Cloud Service Language

protocols, exchanged messages and location. However, non-functional attributes are not described inside WSDL document[31].

In the cloud service domain, some attempts have been made to build a cloud service language that provides the description of several cloud aspects, such as computational and network resources, services profiles, and the developers' requests to enhance both cloud service discovery and selection. Table 2.2 compares several recent studies that leveraged cloud service language to describe, discover or select cloud services. It also highlights the domain, the purpose of using the language, the language as well as its evaluation.

Zhou et al. [127] propose a P2P-based technique to discover Software as a Service. WSDL-S is used to describe services that will be discovered. They employ semantic protocols and topology reconstruction techniques in order to develop a localized search scheme for service query.

Goscinski and Brock [33] design a framework to exemplify cloud resources and services established on Web services and attributes. This framework assists in facilitating service discovery, publication and choice. A Web Services Description Language (WSDL) file extension is proposed by Goscinski et al. One which takes into account the characteristics of cloud resources. Similarly, they presented CaaS

(Cluster as a Service) as a way to reorganize services and resources, to assist in their selection and publication.

The first language attempt which doesn't rely on Web service language is by Liu and Zic [56]. They propose a unique language Cloud# that works on Infrastructure As a Service. This language model design to model the internal organization of cloud. In addition, the proposed language model aims to cloud service customer to understand how cloud services are delivered and increase the confidence to migrate their business applications to the cloud. In this language model, cloud service customer can check whether their applications are allocated resources fairly and their data are isolated correctly from the data of other customers.

Goncalvs et al. [32] propose CloudML (Cloud Modeling Language) by way of an Extensive Markup Language (XML). The XML allows for service profile models and developer needs, ensuring the represented simulation and tangible resource statuses in D-Clouds (Distributed Clouds). Cloud Modeling Language (CloudML) is capable of characterizing the varying levels of extraction, for example, services, computing, system networking resources and developer needs. Nonetheless, the cloud modeling language specification is lacking in its description of scaling rules.

Ye et al. [119] propose CloudUDDI, an extension model for cloud services. It assists in the storing and selection of services. The model enables the description of data from low-level cloud service resources to be stored. The structure of CloudUDDI comprises physical devices, a simulated layer of resources and a middleware layer, which contains various agents, for example, search and registering middleware. Further, Quality of Service (QoS) and Resource of Service (RoS) models were defined, along with mapping which linked the QoS limitations and low-level resource metric. Included in the QoS model are accessibility, response and processing times, reputation, and cost. The Ros model encompasses network, computing, and storage resources, and so on. The mapping rubrics linking the parameters of QoS and low-level resource metrics are collated and stored in an OWL or XML document.

USDL within cloud computing is used as a language to define the service model, and the properties of functionality and non-functionality [85]. USDL was divided up into nine modules: i) Service, ii) Functionality, iii) Methodology, iv) Participation, v) Interface, vi) Level of Service, vii) Price, viii) Legal, ix) Basis.

Hoberg et al. [42] USDL language was used to: i) provide a service definition which covered all the information needed for service selection, ii) make it compliant with the cloud computing field. USDL is also a metric description language for cloud services. The nine modules were redefined, by way of consumer-required features which were incorporated into cloud services. They intended to support consumers in structuring their service selection, thereby helping cloud service providers to generate descriptions consistent with consumer requirements.

Junker et al. [45] use USDL for defining commercial services cost models within the marketplace. An accumulation method was introduced and used in several cost models into a single model to try and resolve the costing issues of the combined services within the cloud computing industry. The complication of time and price were analyzed through the accumulation method.

Defined attributes in cloud services were extended through the fundamental structure of USDL. This was done in order to attain the definition of the cloud services model referred to as the Cloud Service Description Model (CSDM) [105]. This model illustrates the following aspects with regard to cloud services: i) Methodology, ii) Operation, iii) Commercial, iv) Semantic. It divides the service data into ten modules (Basis, Service, Purpose, Methodology, Level of Service, Cost, Legal, Contributor, Communication, and Operation). The modules provide various aspect specifics. An upscale module was also added to the original USDL, known as the transaction module. It encapsulates hypotheses which measure and assess the pertinent factors of the service transaction. Those factors comprise the rating system (assignable risk assessment, reliance analysis and status evaluation).

Shetty et al. [102] propose an illustration model of storing organization service descriptions through XML. The data illustration model assists users by means of meaningful descriptive language. It can also be used by service providers to market their services. It comprises the name of the service, properties for functionality and non-functionality and locations of servers. The properties can be qualitative or quantitative (OS - operating system, safety mechanisms).

### 2.3.3 Cloud Service Model

In the cloud service domain, there have been some attempts to build a cloud service model that is able to provide the description of several cloud services in order to capture the knowledge of cloud service offers. Cloud service models have the capability to enhance both cloud service discovery and selection by describing the cloud service resources. Table 2.3 compares several recent studies of using cloud service models to model, discover or select cloud services. Also, it shows the domain, an approach and language is used and evaluation.

Lee et al. [54] propose a standard extraction model to define IaaS services of several providers. A typical interface, Amazon Web Service (AWS), and GoGrid were supplied to IaaS providers aimed at dealing with IaaS situations relating to private and public clouds. The combined IaaS services are offered to users through a gateway (portal) where they can sign up, monitor/observe and control the IaaS services lifecycle from several providers. Web2Exchange is the working environment for this portal, one that is an integrated model in order to make distribution, discovery, and incorporation possible. Those service models utilized by Web2Exchange are considered as Managed Object Format (MOF).

Sun et al. [106] propose a constraint-centered programme model able to determine cloud resources and install applications. The programming model allows for map requirement application toward constraints of cloud resources. Such resources are

utilized when executing applications. Constraints can include, software, hardware, statistics, storage, safety, functionality, and conformity.

Quinton et al. [90], propose a method for choosing cloud environments, establishing the environment configuration and installing applications. The method was centered around a group of Software Product Lines (SPL) and a domain model, one that allows developers to be able to: i) routinely choose a cloud environment which suits the relevant set of requirements, ii) Routinely gathers file descriptions and script executions to align with the cloud-related environment. The authors made a proposition for an extension of the Feature Models (FMs) through the cardinalities and characteristics as inconsistency models to define the cloud environment.

Hamdaqa et al. [38] propose a meta-model which allows cloud consumers to create flexible applications by way of a service, free from any other platform, preventing seller lock-in issues. The model is capable of describing cloud application syntax (abilities and methodology interfaces). However, it is only equipped to encapsulate a few semantics.

Gudenkauf et al. [35] propose a referencing architecture by way of a feature model for describing service offers in an even manner. The feature model comprises nine modules (type, distribution, cost, function, integration, sourcing choices, SLA, business specificity, and cloud service appearance). The authors also created CSNs (Cloud Service Navigators) as a technique for visual descriptions and comparisons to cloud service needs and offers. The key feature of this method is to distinguish the service description quickly.



Study	Aim	Domain	Model	Language	Automation	Evaluation
[54]	D	IaaS	Abstraction models	UML	Manual	Simulation
[106]	D	IaaS	Programming model	<i>x</i>	Manual	Case study
[90]	S	SaaS	Knowledge Model	<i>x</i>	✓	Case study
[38]	M	SaaS	Meta-model	UML	Manual	<i>x</i>
[61]	M	Cloud Service	Model-driven	<i>x</i>	Manual	Case study
[7]	M	Cloud Service	multi-agent	UML	Manual	Case study
[14]	S	Cloud Service	Customer-Centric	UML	Manual	Case study

D:Discovery S:Selection M:Modeling

Table 2.3 Cloud Service Model

Mastelic et al. [61] propose a model-focused method to verify and govern random cloud services. A meta-model referred to as CoPS was defined. It depicts cloud services as a fundamental configuration of hardware and software, through use of three modules (component, product, and service). The meta-model leans toward gaining a constant representation of cloud services. A Cloud Management System (CMS) design was also proposed by the authors, one which was capable of managing cloud services autonomously, converting service models with an intangible representation into actual deployment. The components of this service are represented by prototypes called 'black boxes'. The conversion into an application model and ultimate deployment is attained through plugins. The CoPS modular enables reuse of prototypes and plugins in other cloud services through the CMS architecture. These interface modules are referred to as UMLs.

The MeraMORP(h)OSY approach is described by Amato and Moscato [7] as appertaining to multiple agent models that enable, depiction, configuration, and authenticity of cloud service needs. The method employs a modeling profile known as MDE, which is capable of describing services as agents within a multiple agent setting. This approach comprises an authenticity procedure for needs which manipulate recognized methods in the course of the lifecycle of the service.

Cai et al. [14] propose a customer-oriented cloud services model which examined service providers and consumers relationships. The approach focuses on the commercial sector of cloud computing, more specifically, commercial consumer needs, but not the procedural characteristics.

## 2.4 Summary

In this chapter, we have introduced some basic concepts related to cloud service discovery in cloud environments and presented the state-of-the-art. In particular, we discussed cloud service essential characteristics and their models. We then presented the

---

service discovery in particular focus on Web service discovery. We present a generic study of cloud service discovery that includes three different approaches which are cloud service ontologies, languages, and models. In contrast, 33 alternative cloud service studies were compared for cloud computing as well as the related study areas. In the next chapter, we describe our cloud categorization framework and how this framework exploit a comprehensive ontology to discover and categorize cloud services.



# Chapter 3

## Cloud Service Ontology and Categorization

Discovering and categorizing cloud services is challenging due to a number of reasons. Firstly, cloud services are provisioned at various levels, including both the level of data and business logic, and the level of infrastructure capabilities. Secondly, cloud service providers may not follow a standard to describe their services and resources [29]. Interestingly, the majority of the publicly available cloud services are actually not based on service description standards [83]. In contrast, Web services use standard languages such as the Web Service Description Language (WSDL) or Unified Service Description Language (USDL) to expose their interfaces for effective discovery. Thirdly, the variety of Service Level Agreements (SLAs) between cloud service users and cloud service providers makes it challenging to identify cloud services [88]. Furthermore, to discover an appropriate service that suits a particular user's need from the vast number of services available in real environments is another challenge. This is because many irrelevant search results (e.g., blogs, news and research papers) will be returned if the current search engines are used to search cloud services. For instance, some businesses that have nothing to do with cloud computing

(e.g., cloud9carwash<sup>1</sup>) may use “cloud” in their names or service descriptions. On the other hand, some real cloud services (e.g., Dropbox<sup>2</sup>) may not be returned by the current search engines because they do not mention “cloud” at all.

In this chapter, we design and implement a cloud categorization framework. This framework employs a comprehensive ontology to discover and categorize cloud services in real environments. The main idea behind our framework is categorizing cloud services using an ontology-based technique for identifying and categorizing cloud services in the process of service discovery. This cloud service ontology is built semi-automatically by mining new concepts from real-world cloud services. In a nutshell, the main contributions of our work are summarized in the following:

- We design and develop a cloud service categorization framework that achieves more accurate categorizing results by consulting a comprehensive cloud service ontology that reasons on the relations of cloud services.
- We develop a novel approach to build the cloud service ontology. We first develop an initial cloud service ontology (i.e., as a roadmap for the ontology builder) based on the NIST (US National Institute of Standards and Technology) cloud computing standard. New cloud service related concepts are then discovered and added to the cloud service ontology by automatically analyzing real cloud services.
- We propose using Term Frequency to weigh cloud service ontology concepts and calculate on cosine similarity to measure the similarity between cloud services. We then design and develop cloud service categorization algorithms that categorize cloud services to clusters for effective identification of new cloud services.

---

<sup>1</sup><http://www.cloud9carwash.com/>

<sup>2</sup><http://www.dropbox.com/>

This chapter is organized as follows. Section 3.1 briefly presents the architecture of our cloud service categorization framework. In the Section 3.2 we present the details of the cloud service ontology including the ontology roadmap, cloud service concept discovery, the concepts' position determination and cloud service categorization. In Section 3.3 we describe the cloud service categorization procedures to categorize cloud service. Section 3.4 overviews the related work. Finally, Section 3.5 provides our summary.

## 3.1 Cloud Service Categorization Framework

In this section, we first briefly introduce our Cloud Service Categorization Framework (CSCF), then we focus on describing our approach to cloud service ontology generation.

### 3.1.1 CSCF Architecture

Figure 3.1 depicts the architecture of CSSE, which consists of five major layers, namely the *Cloud Services Collection Layer*, the *Cloud Services Extracting Data Layer*, the *Cloud Service Ontology Layer*, and the *Cloud Services Categorization Layer*.

#### Cloud Services Collection Layer

This layer is responsible for collecting possible cloud service seeds (or Web sources, i.e., the cloud services' URLs) in real environments. We initially collect cloud service seeds using two approaches. Firstly, we develop the cloud service source collector module that is able to collect cloud services automatically by crawling Web portals and indexes on search engines, such as Google, Yahoo, and Baidu. Secondly, we develop the cloud service seed inquiry based module that has the capability to inquire a

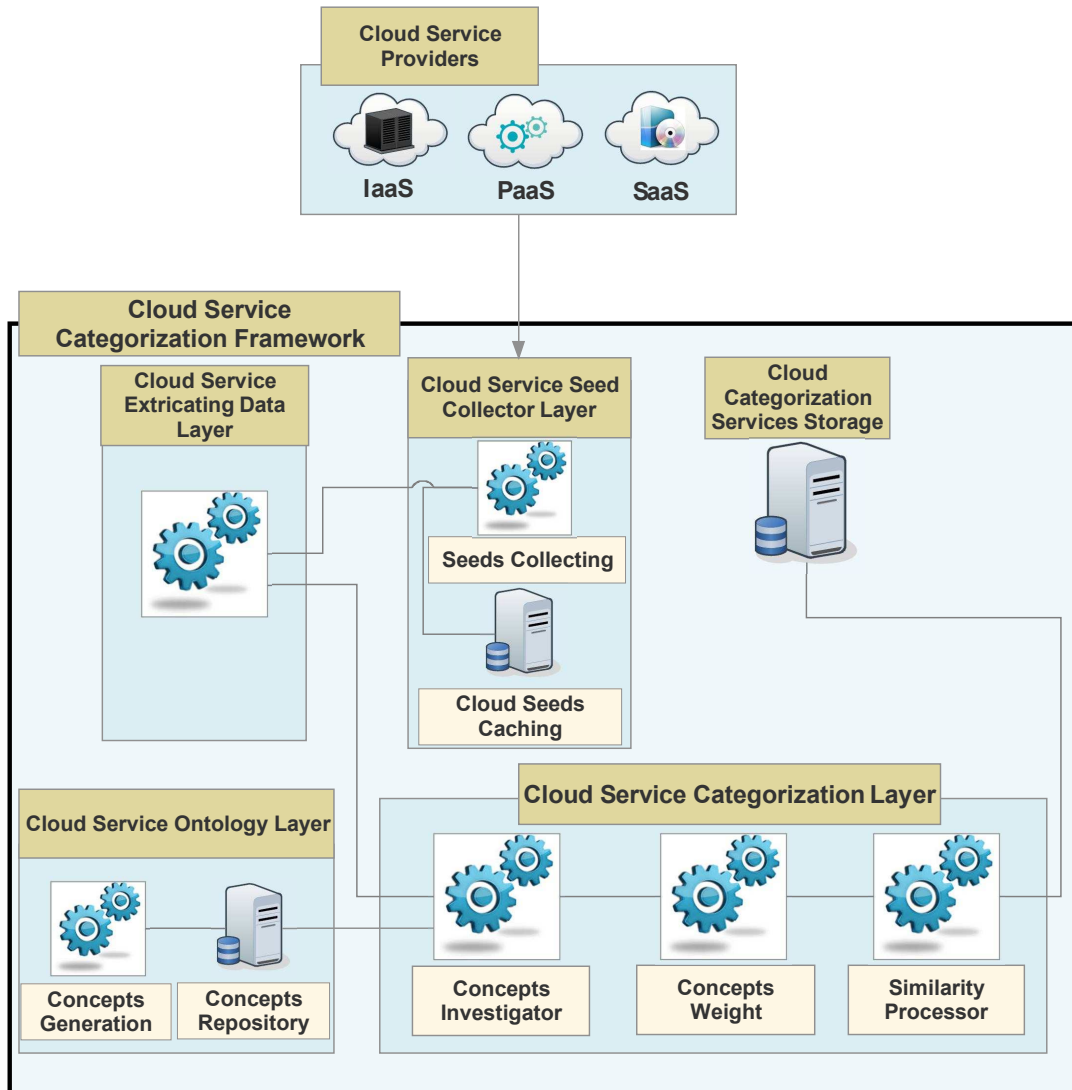


Fig. 3.1 Architecture of the Cloud Service categorization framework

cloud service and determine whether this cloud service has been cached in cloud service seeds repository. This inquiry can be done by both cloud service customers and cloud service providers. Furthermore, if the cloud service users inquire about a seed registered in the system, the system can return the inquiry result directly. Otherwise, the seed will be sent to the Cloud Services Extracting Data Layer for obtaining the essential details that can determine if the seed provides the cloud service.



### **Cloud Services Extracting Data Layer**

This layer is responsible for extracting essential content in the Web source (or seed), such as description, keywords, and text content. If the Web source is a cloud service, the cloud service content can be obtained automatically by filtering the Web source content (i.e., from cloud service source HTML page to text). The cloud service content provides support to the building of automation cloud service categorization. Then the cloud service source will be sent to Cloud Services Categorization Layer.

### **Cloud Service Ontology Layer**

The cloud services ontology layer is responsible for maintaining the cloud services ontology, which consists of two main parts: the *cloud service ontology repository* and the *ontology builder*. The cloud service ontology repository contains concepts that are generated by the ontology builder. The ontology builder generates concepts semi-automatically using an ontology roadmap and the cloud services' metadata. More information on the cloud service ontology generation approach can be found in Section 3.2.

### **Cloud Services Categorization Layer**

This layer is responsible for identifying and categorizing cloud services. The Cloud Services Categorization Layer contains the categorization processor to determine whether a given seed (or a Web source) is a cloud service provider based on the acquired concepts in the cloud service ontology. This layer also relies on clustering methods to build the categorization. The categorization can be updated automatically after categorizing a new cloud service provider to increase the categorization knowledge. Furthermore, the categorization only focuses on cloud computing which does not include cloud mobile computing. However, our framework can avoid the lack of success in categorizing a cloud service source by allowing the cloud service provider

to register its service in our system and this cloud service source can be added to the categorization knowledge base as a new cloud service provider.

## 3.2 Cloud Service Ontology Generation

Our approach on cloud service ontology generation involves two main steps. Firstly, we create a base ontology by following the US National Institute of Standards and Technology (NIST) cloud computing standards. Since there are very limited concepts in this ontology, we also call it a *cloud service ontology roadmap*. Then, our ontology builder grows the cloud service ontology with new concepts by automatically analyzing the metadata of real-world cloud services. Furthermore, cloud service ontology concepts have two properties, which are the abbreviation and weight of cloud service ontology concepts. The cloud service ontology concepts abbreviation gathers for particular concepts in cloud services ontology roadmap such as SaaS for Software as a Service and PaaS for Platform as a Service, while the weight is computed from the Ontology Builder Algorithm. We will describe our approach in details in the following.

### Cloud Service Ontology roadmap(*CSOr*)

Our Cloud Service Ontology roadmap (*CSOr*) has been built by following the interpretation of the NIST standards for cloud computing [63] and other published ontologies for cloud computing, to obtain a set of concepts that can be used as a roadmap for our cloud services' ontology. These concepts have relationships that can be defined as *is\_a* and *is\_not\_a* cloud service ontology, to enhance the generation of a new concept of the ontology. According to NIST, the cloud model comprises of five *essential characteristics*, and three *service models*. The five essential characteristics are as follows:

- *On-demand self-service*: A consumer possesses independent provision of computing capabilities like server time and network storage, whenever necessary without seeking the attention of each service provider.
- *Broad network access*: Capabilities can be accessed on-line and through standard mechanisms which encourage the use of various client platforms such as mobile phones, tablets, laptops and workstations.
- *Resource pooling*: The provider's computing resources are shared among multiple consumers under a multi-tenant model, where dissimilar physical and virtual resources are engaged or disengaged according to consumer preferences. In normal circumstances, consumers do not have the knowledge of the precise locations apart from information of a higher level of abstraction (e.g., country, state, or data center) that provide such resources (e.g., storage, processing power, memory and network bandwidth).
- *Rapid elasticity*: Resources are elastically supplied or released automatically or manually on demand. Capabilities as seen by the consumers are often unlimited and are utilized at any amount of time.
- *Measured service*: Cloud services may charge according to a pay-per-use or charge-per-use basis. Resource usage of the service is monitored, controlled and reported, allowing transparency between the provider and the consumers.

The three service models specified by the NIST cloud computing standard are as follows:

- *Software as a Service (SaaS)*: The term is referred to a service provided by a cloud application supported on a cloud infrastructure. Consumers can interact with the application through a user interface like a Web browser installed on different client devices. It is in general straightforward for the consumers to

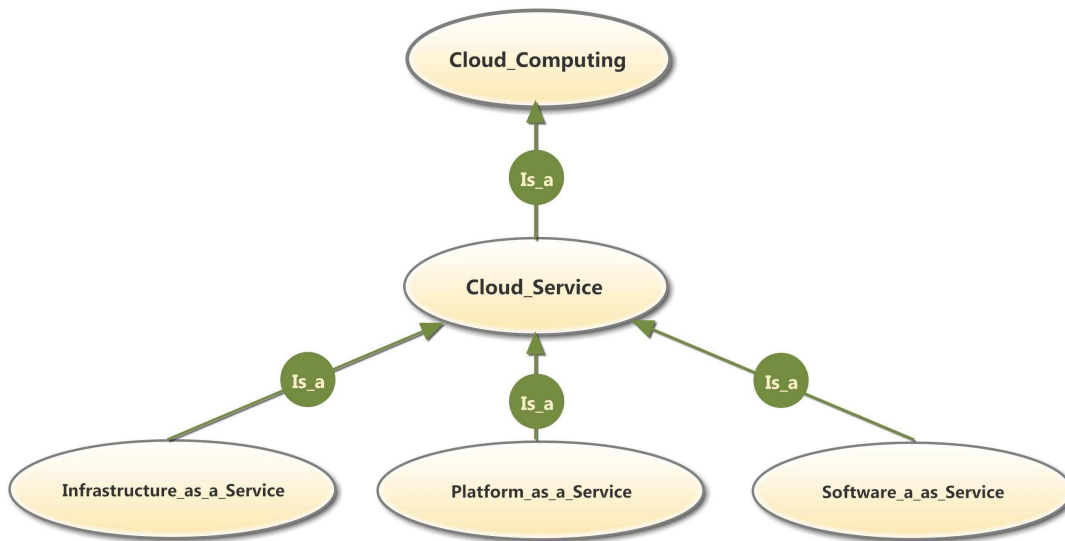


Fig. 3.2 Cloud Service Main Concepts

manage or manipulate any underlying cloud infrastructure such as the servers or operating systems. However, they may be given the control limited to some particular application configuration settings.

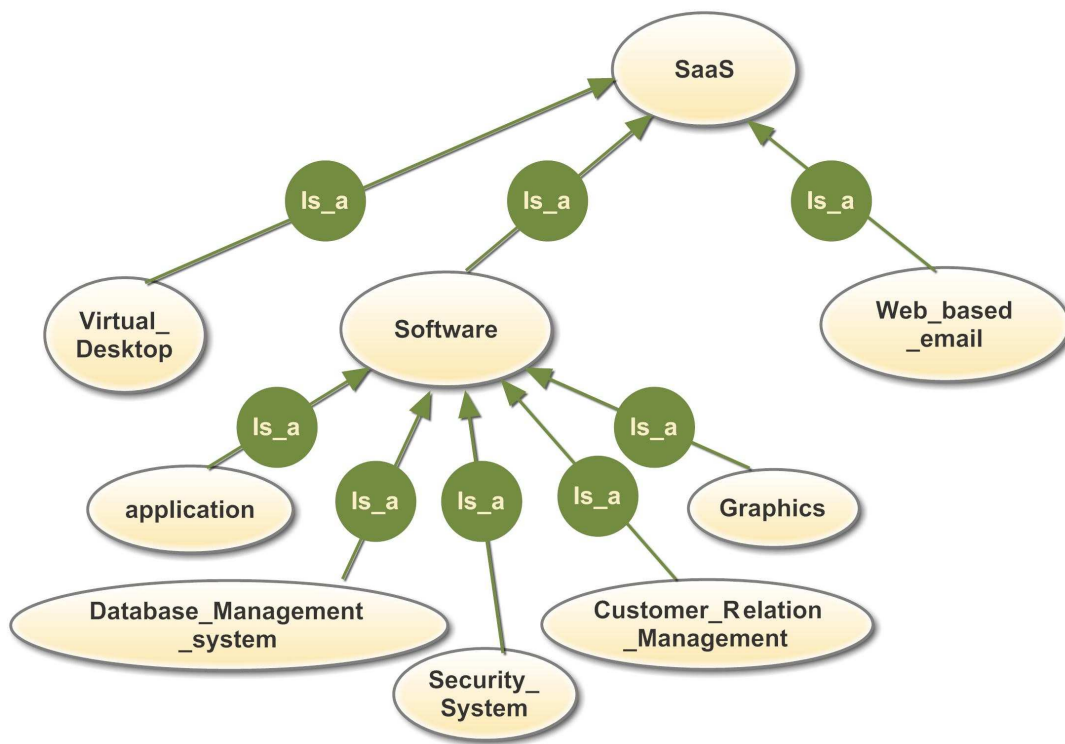


Fig. 3.3 Software as a Service (SaaS)

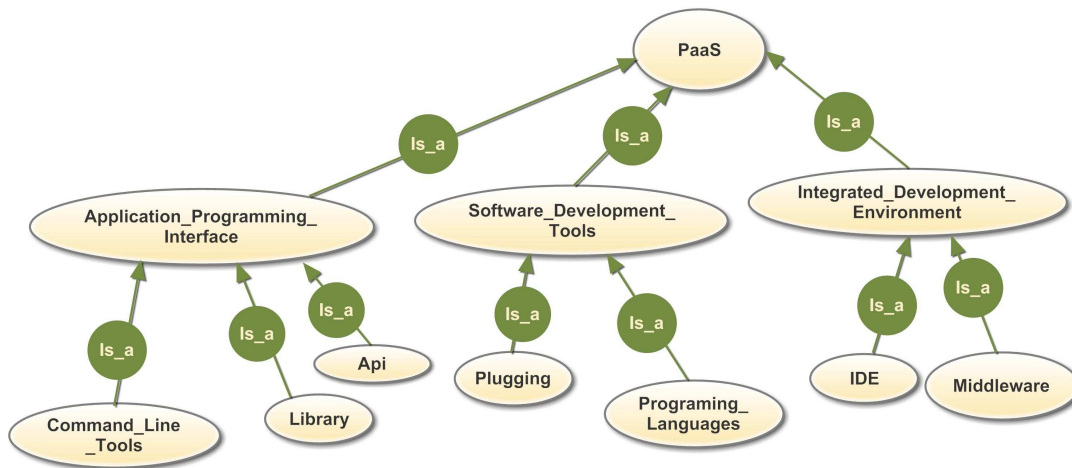


Fig. 3.4 Platform as a Service (PaaS)

- Platform as a Service (PaaS)*: PaaS is another type of service on which is built upon the cloud infrastructure and involves consumer developed or acquired applications with the help of programming languages, tools, libraries and services powered by the . The consumer is not required to manage or configure the cloud infrastructure. But provider possess the power to control the deployed applications and configuration settings of the hosting environment where the applications are deployed.
- Infrastructure as a Service (IaaS)*: It is defined as the ability to supply the consumer with resources from the ground up - processing, storage, network and other basic resources, in order to allow consumer to develop and run software as well as operating systems and applications. The consumer does not need to manage or configure the cloud infrastructure. But they may have the authority over controlling the operating systems, storage and other mounted applications with the possibility of limited control over selected networking components like the host firewalls settings. For simplicity, we only depict part of the IaaS concepts (see Figure 3.5).

While developing our CSOr based on the interpretation of NIST's cloud computing standards, we determine that cloud computing is the root node to the relationship

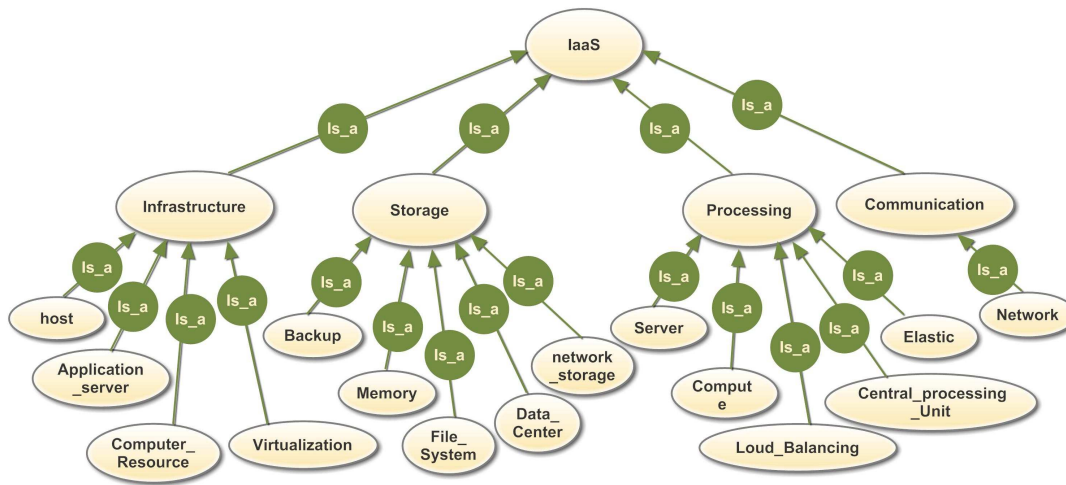


Fig. 3.5 Infrastructure as a Service (IaaS)

between *is\_a* and *is\_not\_a*. In addition, in CSOr, we consider *cloud\_service* a child of the root node *cloud\_computing* and the parent node for other cloud service concepts. We also define the concepts for Essential Characteristics and Service Models. For example, we treat Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS) as three main child nodes of cloud service and parent nodes for other cloud service concept levels. The Service Models builds the three main branches in CSOr and additional cloud service concepts are added into the appropriate branches depending on the type of services. For instance, we could add *storage* as a child node of Infrastructure as a Service (IaaS). In our cloud service ontology roadmap, we consider NIST-interpreted concepts having a higher priority than the available ontology concepts, because the NIST concepts are more valuable in describing cloud services. As a result, those NIST-interpreted concepts are added at a high level in CSOr.

We also consider *is\_not\_a* in our cloud service ontology roadmap, which represents a set of concepts unrelated to cloud services. One example is the concepts such as weather forecast, which may show word “cloud” but does nothing related to cloud services. Another example is the concepts related to research such as reports, articles, and publications. Clearly, social network items can also be considered as *is\_not\_a* re-

lations since the word “cloud” may appear in news items, blogs, Twitter or Facebook posts, but they are not actually a cloud service. Given the large-scale of the Internet, having `is_not_a` relations in the ontology is very useful for the cloud service crawler to validate and filter out non-cloud services.

### Generating Cloud Services Ontology

Since the cloud services ontology roadmap provides very limited number of concepts, it is necessary to generate a more comprehensive ontology with new concepts related to cloud services. One possible approach is to analyze known cloud services and mine new concepts from their metadata such as service descriptions. Based on our previous effort where 5,883 cloud services were identified [84], we develop an algorithm (see Algorithm 1) to analyze these cloud services and identify new cloud service concepts for our cloud service ontology. There are five steps in the algorithm, which are detailed in the following:

- *Detection*: In this very first step, we detect whether a term should be a candidate concept to be considered in our cloud service ontology. We use the term frequency, a well-known information retrieval technique [97], for this purpose. Term frequency represents the number of times that a term appears in the cloud services sources  $S$ . Only those terms with term frequencies that pass a threshold can be considered as candidate concepts, which are then passed to the next step for validation.
- *Validation*: The validation step compares the candidate concept with the existing concepts in the cloud services ontology. This eliminates possible repetition and ensures that this candidate concept is not a node or part of a node in the ontology. If the candidate concept has not appeared in CSO, it is considered to be a new cloud service concept, and will be passed to next step for further processing.

- *Balancing*: The purpose of the balancing step is to provide a weight for each new concept by using the popular *TF\*IDF* model:

$$TF(nc, C) = \frac{f(nc, s_j)}{f\{t, s_j : t \in s_j\}} \quad (3.1)$$

where  $nc$  is the new concept,  $C$  represents the set of all concepts in CSO, and  $f(nc, s_j)$  returns the frequency of  $nc$  in  $s_j$ .

$$TF*IDF(nc, C) = tf(nc, C) * \log\left(\frac{\sum S(s_j, S)}{\sum S\{s_j, s : nc \in s\}}\right) \quad (3.2)$$

The *TF\*IDF* equation provides weights for cloud service ontology concepts.

- *Addition*: This step adds a new concept to CSO by using the inverted index and *TF\*IDF*. More specifically, we first link the new concept to the concepts in CSO or using the inverted index, which allows us to determine where the new concept appears in the cloud services source. We then select the new concept and the documents where the new concept appears. We count the existing CSO concepts that appear in the same cloud services source. After that, we store the existing CSO concepts and pick up the CSO concept with the maximum appearance and create the link between them.

$$CFS = \langle c_1, c_2 \dots c_x \rangle_{|S| \times |C|}. \quad (3.3)$$

Furthermore, to determine the new concept's position in the cloud services ontology, we need to check the weight of the existing CSO concept that has the most appearances. If the weight of the new concept is more than one of the existing concept, the new concept is inserted to the ontology as the child of this concept. Otherwise, the new concept is added as a sibling.



- *Updating*: The updating step is responsible for upgrading the cloud service ontology after adding the new concept.

### 3.3 Cloud Service Categorization

In this section we introduce the process of cloud service categorization. The process identifies and categorizes cloud services. The process consists of three phases, including Cloud Service Concepts Investigator, Cloud Service Concepts Weight and Cloud Service Concepts Similarity.

#### 3.3.1 Cloud Service Concepts Investigator

Cloud Service Concepts Investigator is responsible for investigating cloud service ontology concepts in a given cloud service source. This investigator can detect and assure that the cloud service ontology concepts also appear in the cloud service source. Suppose  $S_j$  denotes the set of concepts in the  $j$ -th cloud service. For simplicity, we use  $S_j$  to represent the cloud service when the context is clear. Note that, we also have a set of concepts in  $CSO$ . The investigator procedure examines each cloud service ontology concept  $c_i \in CSO$  to check whether the concept  $c_i$  appears in  $S_j$ , i.e., to check whether we have  $c_i \in S_j \cap CSO$ . Furthermore, the investigator counts the frequencies of cloud service ontology concepts that appear in  $S_j$  and records them in  $f(c_i, S_j)$  for each concept  $c_i \in CSO$ .

#### 3.3.2 Cloud Service Concepts Weight

This process is responsible for weighing all concepts in  $c_i \in CSO$  by using a well-known information retrieval method called Term Frequency [97]. We apply Term Frequency instead of Term Frequency-Inverse Document Frequency (TF-IDF) because we focus on cloud service ontology concepts only. Meanwhile, we assume that each

---

**Algorithm 1: Ontology Builder Algorithm**


---

```

Step 1. Detection:
/* Determine candidate concepts from cloud services' sources S.*/
for each term  $\in S$  do
  count its frequency f
  if  $f < \text{Threshold}$  then
    term is not a candidate concept cc
  else
    term is a candidate concept cc and pass cc to validation
  end if
end for

Step 2. Validation:
/*Check if the candidate concept  $cc \in CSO$ .*/
if  $cc \in CSO$  then
  cc is already at CSO
  Back to Step 1.
else
  cc becomes a new concept nc of CSO
  Pass nc to balancing
end if

Step 3. Balancing:
/*Provide a weight for new concept nc */
The balancing step uses TF*IDF technique to give each new concept nc a weight.

Step 4. Addition:
/* Link the new concept with old concepts and determine the new concept position in CSO */
Select new concept nc
for each  $s \in S$  do
  if  $nc \in s$  then
    for each  $CSOc \in CSO$  do
      /* CSOc: cloud service concepts */
      if CSOc in s then
        store CSOc in nc Set
        /* The new concept set (nc Set) contains all cloud service ontology concepts that appear with the new concept
        */
      end if
    end for
  end if
end for
call maximum frequency concept max nc-CSOc in nc Set
/* this method provide the maximum repeating concept that appear with the new concept nc in nc Set */
call maximum frequency concept max nc-CSOc weight
call nc weight
if nc weight > max nc-CSOc weight then
  nc add to max nc-CSOc in CSO as child
else
  nc add to max nc-CSOc in CSO as siblibng
end if

Step 5. Updating:
The CSO ontology updates weights and relations for each concept.

```

---

cloud service is an isolated source from other cloud service sources when applying similarity measure.

$$TF(c_i, S_j) = \frac{f(c_i, S_j)}{TNT_{S_j}} \quad (3.4)$$

where  $c_i$  denotes the concept in  $CSO$ ,  $f(c_i, S_j)$  denotes the frequency of concept  $c_i$  in the given cloud service source  $S_j$ , and  $TNT_{S_j}$  denotes the total number of terms in  $S_j$ .

We construct a concept Term Frequency  $TF$  matrix  $M$ , built from the results of Equation (3.4) for all cloud services, i.e., each element  $m_{ij}$  in  $M$  can be calculated by  $m_{ij} = TF(c_i, S_j)$ .

### 3.3.3 Cloud Service Concepts Similarity

This process is responsible for determining the similarity between the cloud service sources. We apply a popular similarity model, cosine similarity, to measure the similarity of two cloud service sources in a vector space.

Based on the definition of  $TF$  matrix  $M$  and assuming that we have  $n$  concepts in total in  $CSO$ , for each cloud service  $S_j$ , we have a  $TF$  vector  $\bar{s}_j = \langle m_{1j}, m_{2j}, m_{3j}, \dots, m_{nj} \rangle$ . Then we can define the similarity between two cloud service sources  $S_i$  and  $S_j$  as follows:

$$cosine(S_i, S_j) = \frac{\bar{s}_i \cdot \bar{s}_j}{\|\bar{s}_i\| \times \|\bar{s}_j\|} \quad (3.5)$$

### Cloud Service Categorization Procedures

Cloud service categorization procedures are responsible for generating cloud service categorizations. The procedures aim to achieve a smaller number of clusters. These clusters will have the capability to identify and categorize new cloud sources. Figure 3.6 depicts the main steps of generating cloud service categorization. These steps (see Algorithm 3) can be described as follows:

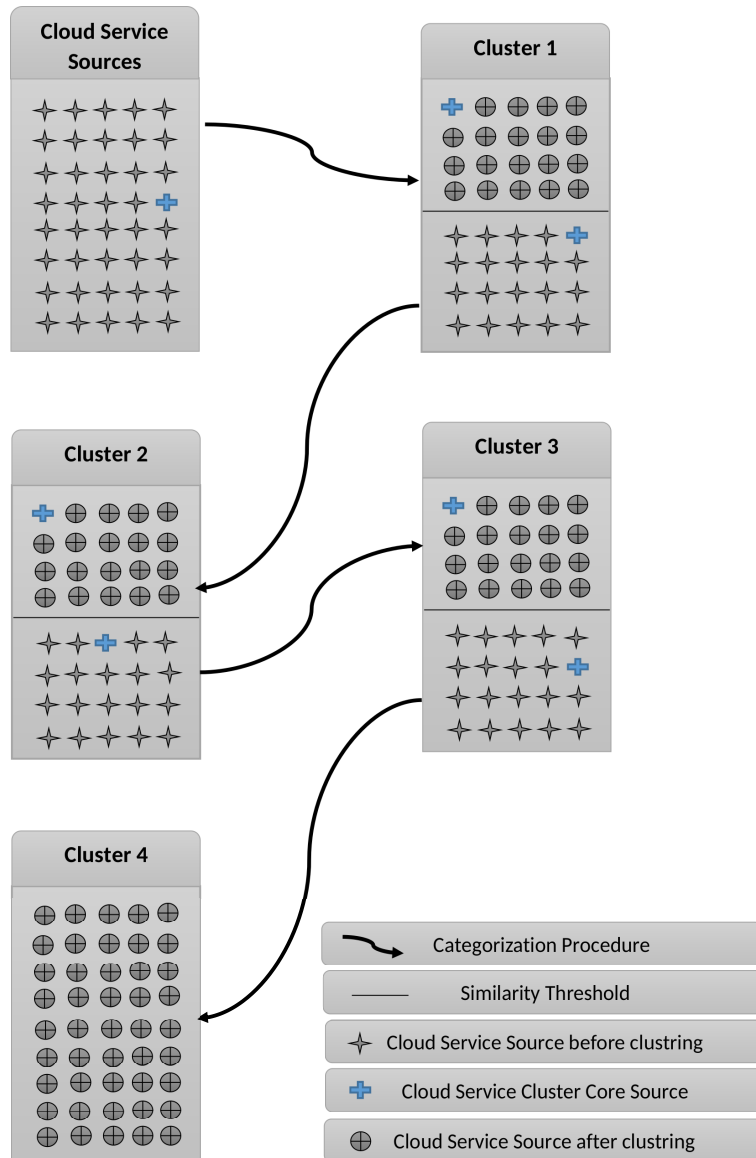


Fig. 3.6 Illustration on the Cloud Service Categorization Procedure

- Because it is hard to determine the minimum diversity for each cloud cluster, we randomly select a cloud service source as the core source of the first cluster. Then we calculate the similarities between this randomly selected cloud service source and other cloud service sources. All cloud service sources that have a larger similarity score than a predefined threshold will be added to the first cluster. After being added to the cluster, these sources will be removed from the candidate set of cloud service sources.

---

**Algorithm 2:** Cloud Service Categorization Algorithm
 

---

**Require:** A set of cloud service sources  $\mathbb{S} = \{S_1, S_2, \dots\}$ ,  $TF$  vectors  $\overline{s_1}, \overline{s_2}, \dots, \overline{s_{|\mathbb{S}|}}$  for each cloud service source in  $\mathbb{S}$ , similarity threshold  $\theta$ .

**Ensure:** Categorization result  $\mathbb{C}$

```

 $\mathbb{C} \leftarrow \emptyset$ 
count  $\leftarrow 1$ 
while  $\mathbb{S}$  is not empty do
  Select a new cloud service source  $core \in \mathbb{S}$  randomly
   $C_{count} \leftarrow \emptyset$ 
  Add  $core$  to cluster  $C_{count}$ 
  Remove  $core$  from  $\mathbb{S}$ 
  for each  $S_i \in \mathbb{S}$  do
    Compute  $cosine(core, S_i)$  based on  $TF$  vectors  $\overline{s_{core}}$  and  $\overline{s_j}$  for cloud service sources  $core$  and  $S_j$ 
    if  $cosine(core, S_i) \geq \theta$  then
      Add  $S_i$  to cluster  $C_{count}$ 
      Remove  $S_i$  from  $\mathbb{S}$ 
    end if
   $\mathbb{C} \leftarrow \mathbb{C} \cup \{C_{count}\}$ 
  count  $\leftarrow count + 1$ ;
end for
end while

```

---

- We continue to randomly select another core source from the remaining cloud service sources to generate a second cloud cluster. Again, we employ a similar selection process in the previous step based on Cloud Service Concepts Similarity to add similar cloud service sources to this cluster.
- We repeat the above procedure until all cloud service sources are categorized into a certain cluster. We count the number of clusters that have been generated.
- Because we select cloud service sources randomly and we target to obtain less number of clusters, we run our procedures for several times and choose the result with the minimum number of clusters as our final result.

### 3.4 Related Work

Cloud service discovery faces several new challenges that need to be addressed by exploiting previous research outcome in the areas like Web services, which has been an active area in the past decade [52, 92, 128].

However, the major difference between cloud services and Web services is the use of standard language for the description of services. Cloud services use limited standard languages to describe service details [43, 83, 30, 89] and the variety of service levels offered by service providers is also limited. Furthermore, categorizing cloud services in real environments has introduced new challenges in cloud service discovery due to its integration with other services available on the Web.

However, most of cloud service studies have been focused on ontology-based approach to discover or categorize cloud services. For example, Youseff et al. [122] build an ontology that relies on classifying cloud computing based on its components, which consists of five layers: the *applications*, the *software environment*, the *software infrastructure*, the *software kernel*, and the *software hardware*. Each layer can contain one or more services depending on the level of abstraction. Also, each layer relies on computing concepts to measure limitations and strengths. Weinhardt et al. [116] propose to build the ontology based on a cloud business ontology model. This ontology model consists of three layers: the *platform*, the *infrastructure* and the *application*, as well as a content pricing model to help clarify the relationship between cloud service providers and customers. Kang and Sim [46] propose a cloud service discovery system that uses an ontology-based approach to discover cloud services close to users' requirements. However, cloud service providers still need to register in the discovery system in order to publish their cloud services. Furthermore, their work relies on software agents to perform reasoning tasks (e.g., similarity reasoning, equivalent reasoning and numerical reasoning).

Zeng et al. [124] propose an approach to determine the interoperability between two given Cloud services by building a service matching algorithm and a service composition algorithm. Yoo et al. [120] propose an approach to select cloud services that best meet the user's requirements by using a cloud ontology based on resource services. The authors use the similarity computing degree of virtual cloud service physical resources to determine the best cloud services for users.

Rodríguez-García et al. [28] propose an automatic general ICT domain that can be used to discover the cloud services that best match user needs. The authors use semantic annotation in order to improve the cloud service discovery results. From cloud service descriptions, semantic content can be extracted by using the annotation platform. Then, the semantic content can be used by the semantic search engine to assist users in finding those services that meet with their requirements and expectations.

In cloud service categorization, Hofer and Karagiannis [43] build a tree structure taxonomy that is used to classify the cloud services. This hierarchical taxonomy has been built on top of cloud service characteristics. However, the classification is limited in enhancing cloud service discovery and categorization because it only considers a few cloud services such as Google apps. Unlike previous works which have failed to develop specific cloud service ontology automatically and perform cloud service discovery using real cloud service information and categorize cloud services automatically in large-scale, our ontology-based cloud service categorization framework distinguishes cloud services from other services available over the Internet using an automatically-built cloud service ontology and categorize cloud service providers based on cloud service model.

### **3.5 Summary**

In this chapter, we design and develop a cloud service categorization framework that achieves more accurate categorizing results by consulting a comprehensive cloud ser-

vice ontology that reasons on the relations of cloud services. In addition, we develop a novel approach to build cloud service ontology. We first develop an initial cloud service ontology (i.e., as a roadmap for the ontology builder) based on the NIST cloud computing standard. New cloud service related concepts are then discovered and added to the cloud service ontology by automatically analyzing real cloud services. Finally, we propose using Term Frequency to weigh cloud service ontology concepts and calculating cosine similarity to measure the similarity between cloud services. We then design and develop cloud service categorization algorithms that categorize cloud services to clusters for effective identification of new cloud services.

Cloud service Identification Framework for identifying and detecting cloud service automatically is described in Chapter 4. Moreover, the framework provides a Cloud Service Discovery Engine to discover cloud service automatically, explained in Chapter 5. In Chapter 6, we conduct extensive experiments to validate our proposed approach. The results demonstrate the applicability of our approach and its capability of effectively identifying and categorizing cloud services from the Web.



## **Chapter 4**

# **A Robust and Adaptive Identifier Model for Cloud Service in Real Environment**

In reality, identifying and detecting cloud service is a difficult problem due to the unique characteristics of cloud services such as highly dynamic nature and diverse services offered at different levels, together with the lack of standardized description languages pose significant challenges for effective cloud service identifier. Moreover, since the term “Cloud” is a general terminology, it increases the difficulty of discovering cloud services in real environments such as the World Wide Web. For example, we can find cloud one of the most important terminologies in any websites about meteorology. Moreover, Fig 4.1 shows the first 100 search results from a current general-purpose search engine (Google or Bing) using different keywords, such as cloud service, cloud storage, cloud service provider, cloud hosting, cloud software, cloud platform or cloud infrastructure to search for cloud services. Furthermore, general search engines are very weak in providing details about cloud service features (e.g., cloud service type, process limitation, storage maximums and memory capac-

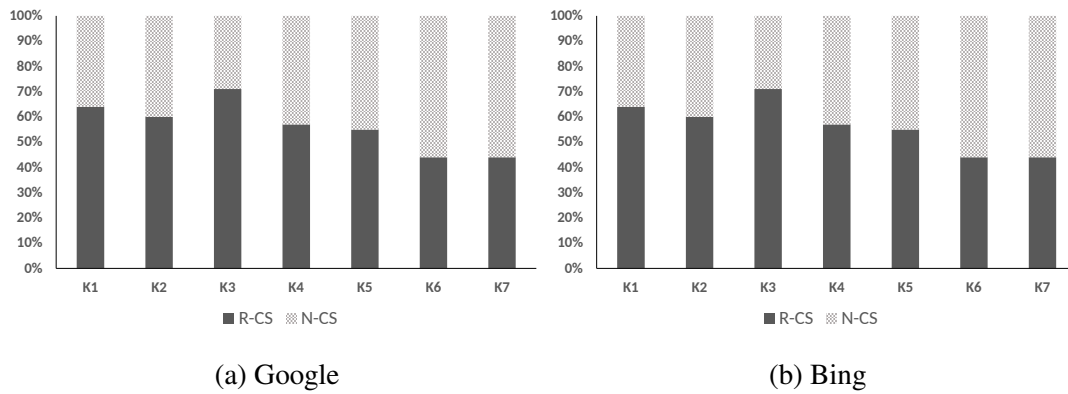


Fig. 4.1 The first 100 searching results from two most popular search engines (Google and Bing) using different keywords, such as cloud service, cloud storage, cloud service provider, cloud hosting, cloud software, cloud platform and/or cloud infrastructure to search cloud services

ity). Considering all these limitations of general-purpose search engines, in this work, we aim to design a cloud service identifier to address the aforementioned issues.

Several research questions centered around an effective cloud service identifier are as follows:

Q1: Is it possible to determine whether a given website on the Web is a cloud service?

Q2: How to identify whether a given website on the Web is a cloud service?

Q3: What kind of features can be used to identify cloud services on the Web?

With these questions, we believe that there is a need to build effective identification model for cloud services that are currently available on the Web. The identification model will enhance the cloud service identifier framework by providing highly accurate results in discovering cloud service. In this chapter, we describe our design of a cloud services identifier framework. This framework helps distinguish between cloud services from other services available on the Internet. Furthermore, the framework provides more details of a service and its features which can support cloud service search users on how to identify an appropriate cloud service towards their needs. The

two main components of the framework include i) a cloud service identifier and ii) a cloud service feature extractor. This identifier helps identify cloud services during the process of cloud service discovery and the process of determining a cloud service features. Moreover, the cloud service identifier can automatically identify cloud service by utilizing a classification method. Then, the feature extractor determines/extracts a cloud service's features using a cluster method and a novel approach, called Service Tracking and Detection (SDT), to detect and track other services. Finally, we can extract cloud service's features that can be used to facilitate the discovering process method.

This chapter is organized as follows: Section 4.1 presents Overview of Cloud Service Identifier Framework. Section 4.2 presents the details of cloud service identification and cloud service features. Section 4.3 provides details of cloud service profile and its algorithm. Section 4.4 presents related work. Finally Section 4.5 we present our summary.

## **4.1 Overview of Cloud Service Identifier Framework**

In this section, we first introduce our Cloud Service Identifier Framework (CSIF), then spotlight on describing our approach on cloud service identification and building cloud service profile.

### **4.1.1 CSIF Architecture**

Fig 4.2 depicts the main components of the cloud service identifier framework (CSIF), which consists of five major layers: namely (1) *Cloud Services Seeds Collection Layer*, (2) *Cloud Services Extracting Data Layer*, (3) *Cloud Services Identification Layer Ontology Based*, (4) *Cloud Services Cluster Layer*, and (5) *Cloud Services profile Layer*.

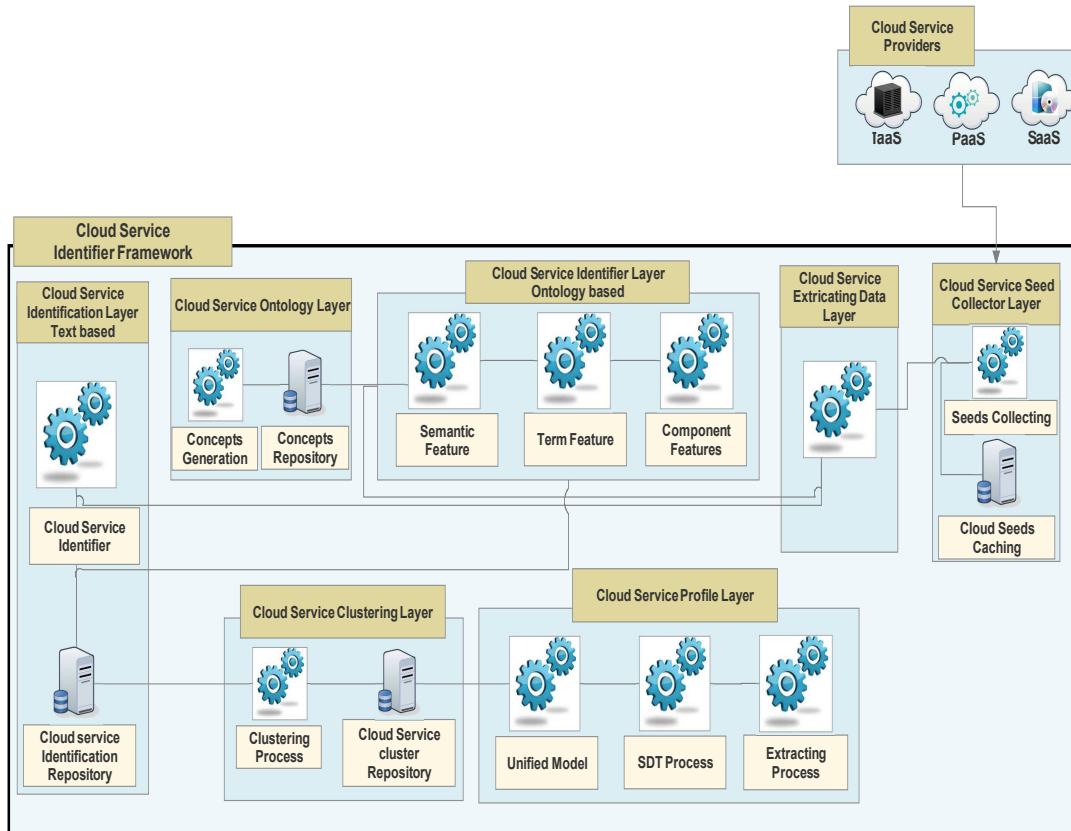


Fig. 4.2 Cloud Service Identifier Framework

### Cloud Services Collector Layer:

This layer is responsible for collecting possible cloud service seeds (i.e., the cloud services' URLs) in real environments. We initially collect cloud service seeds using two approaches. Firstly, we develop the cloud service source collector module that is able to collect cloud services automatically by crawling Web portals and indexes on search engines, such as Google, Bing, and Baidu. Secondly, we develop the cloud service seed inquiry based module that has the capability to inquire a cloud service and determine whether this cloud service has been cached in cloud service seeds repository. This inquiry can be done by both cloud service customers and cloud service providers. Furthermore, if the cloud service users inquire about a seed registered in the system, the system can return the inquiry result directly. Otherwise, the seed will

be sent to the Cloud Services Extracting Data Layer for obtaining the essential details that can determine if the seed provides the cloud service.

#### **Cloud Services Extracting Data Layer:**

This layer is responsible for extracting essential content in the cloud service source such as description, keywords, text content and hyper links. The cloud services' content lead to support of building automation cloud service identifier. The cloud services' content can be achieved automatically by filtering the cloud service source (i.e., cloud service source html homepage to text). Then the cloud services' content is sent to Cloud Services Identification Layer while cloud service hyper links sources can be sent to Cloud Service Cluster Layer if the source pass the Cloud Services Identification Layer.

#### **Cloud Services Identification Layer:**

This layer is responsible for identifying cloud service provider. The Cloud Services identifier contains the process of identifying features to determine whether a given source is cloud service or not. This processing relies on classification method to realize the identification. The identification can be updated automatically after identifying a new cloud service provider to enhance identifying knowledge. Furthermore, the identifier only focuses on cloud computing which does not include cloud mobile computing. However, our cloud service directory can avoid the lack of success in identifying a cloud service source. Because it allows cloud service provider to register their services in our system, these cloud service sources can be added and can be recognized by the identification as a new cloud service provider.

### **Cloud Services Clustering Layer:**

This layer is responsible for clustering cloud service providers. The cloud service clustering is able to collect to the most similar cloud service into clusters based on clustering method. The clusters are built depend on two features which are cloud services' text and cloud services' hyper-links. Firstly, using cloud services' text can lead to finding the most similar cloud services into one cluster. Secondly, cloud service hyper-links can lead to detecting the exact services the cluster provide. This clustering approach is able to decrease the distance during detecting the service and tracking.

### **Cloud Services Cloud Profile Layer:**

This layer is responsible for generating cloud service profile based on the following processes:

1. *Modelling*: this process is responsible for building cloud service model. the model is built based on the service features. In addition, we observe several cloud services to identify the service features such as type, price, capacity. Moreover, this model is used to find the service in a cluster by investigating about the features. The process begins by selecting a cloud service provider and determine the service features that provide. Then, we build JSON model for this service. The JSON model includes many details about the cloud service which are the cloud service features. More details can be shown in Section 4.3.
2. *Detecting and Tracking*: this process is responsible for detecting and tracking other cloud services based on service feature model. This processing can search for the service inside the cluster by taking the JSON model which is built for the service and track this model. The process of detecting and tracking can investigate the whole cloud service websites to find the service. After we find the service we directly build the cloud service model for the cloud service.

3. *Extracting and Storing*: this process is responsible for extracting and storing the cloud service JSON model. The details of cloud services can be received from Detecting and Tracking process. Then, we can store this details in JSON model to support in building a cloud service search engine. We update this process weakly to discover any difference in the cloud service features.

## 4.2 Cloud Service Identifier

In this section we demonstrate our approach to identify real cloud service providers. The proposed approach is a task that uses both information retrieval and machine learning techniques to identify cloud services. In addition, we propose two identification techniques to identify cloud services. The first technique relies on feature engineering which is the process of learning the features that can be shared by large cloud service providers. Second technique is considering cloud services as text documents to identify cloud services without human interaction. Those techniques can be integrated to provide more accurate result in identifying cloud services.

### 4.2.1 Cloud Service Identification Learning Features

To identify cloud services in real environment, we focus on learning features that can be shared by large cloud service providers in the Web to predict whether the given Web source provides cloud service or not. We discover a wide range of features between cloud service providers inside their sources such as similarity function, cloud service semantic concepts and cloud service components. We use them as identification features to train learning models. Learning models are fed into the classification algorithm in order to detect cloud service source (i.e., cloud providers). Following the methodology proposed in the content analysis which has been widely applied in detecting email spam [10, 121], and has also been used for identifying blog spam

[70], we model the cloud service identifier as a binary classification task that is able to both identify cloud service source (CS) and non-cloud service source (NCS). The cloud service identifier has to determine whether a given source is cloud service or not. Each source is represented as a set of features (terms similarity, cloud service semantic concepts, source description, Service Level Agreement, support, accessibility and availability) which are used to feed a classification model that learns an identifying process. Once we have learned to classify sources (S), we are able to determine if a given source belongs to a cloud service. More details about labeling the sources can be found in Chapter 6.

Our first goal is to find features that have the capability to distinguish between cloud service source and non-cloud service source. Once the classification model of cloud service identifier is built, it is confident to use cloud services identifier to determine if a given source belongs to a cloud service. Formally, let  $s$  be a source in a set  $S$ . We want to identify  $G(s)$  by using our identifying process that represents if a given source belongs to the cloud service or not in (Equation 4.1).

$$G(s) : CS \times NCS \rightarrow \{CloudService, NonCloudService\} \quad (4.1)$$

We define a list of features  $F(s) = (f1(s), f2(s) \dots fn(s))$  where each of the features represents a score to indicate CS or NCS according to different features. Then, we predict if  $G(s)$  is a cloud service or a non-cloud service. The CS represents cloud service sources data and NCS denotes non-cloud service sources data. We will discuss these features in the following.

### **Cloud Service Term Features**

The highest obvious feature can be taken into account is *term similarity*. Cloud service providers sharing a high percentage of terms are likely to provide the same service type. Therefore, we consider the term features because the number of services



are still limited. We experiment with Jaccard similarity [77] that provides similarity weight between two cloud services. We use Jaccard similarity because we think that the cloud service providers share many terms when they publish cloud services in the Web. However, to avoid Web sources such as Blog and News that can be intersected with cloud service source, we consider only the highest similarity weight that can be achieved by intersecting each Web source with other cloud service sources.

`Terms_jaccard` computes the Jaccard similarity between the set of terms in the cloud service

$$\Delta = \max(\text{terms\_jaccard}(S_{(s)}, CS_{(i,k)}) = \frac{|T_S(s) \cap T_{CS(i,k)}|}{|T_S(s) \cup T_{CS(i,k)}|} \quad (4.2)$$

where  $\Delta$  denotes the highest similarity weight obtained for Web source  $S_{(s)}$ .  $T_S(s)$  represents the terms that find on web source and  $T_{CS(i,k)}$  denotes the terms that appear in specific cloud service source.

### Cloud Service Semantic Feature

Perceptively, representing a cloud services provider with semantics extracted from a cloud service ontology can be useful to identify cloud service providers. Because in this way, we can ensure high accuracy that the source is a real cloud service. For example, Creative Cloud from Adobe<sup>1</sup> does not share high similarity terms with other cloud service providers because it provides a unique service. But it can achieve many cloud service ontology concepts. Therefore, we use Cloud Service Ontology CSO which is built relying on the interpretation of the National Institute of Standards and Technology (NIST) [63] definition and other publishing ontology that has been published in cloud computing field. Then we can achieve a set of concepts that have the ability to be used as the features for cloud services identification. These concepts have relationships which can be defined as `is_a` cloud services ontology [3].

<sup>1</sup><http://www.adobe.com/creativecloud.html>

Here is Equation 4.3 can be used to represent the cloud service semantic feature

$$Onology(c, cso) = \frac{(\sum c_{i,s_j} : c \in cso)}{(\sum t_{i,s_j} : t \in s_j)} \quad (4.3)$$

Where the numerator represents the total of cloud service ontology concepts that occur in source while the denominator represents all terms that appear in the source.

### Cloud Service Component Features

Cloud Service Component Features are extracted from HTML source for cloud service source. We use these features because we find many of cloud service providers share similar design and description to build their Webpages. Figure 4.3 shows real example of three components which are Accessibility, Support and Service Level Agreement features. We discuss the features as follows:

- **Cloud Service Description:** the cloud service description is the tag of Meta-data in <header> element of web pages. This description can be applied to validate cloud service sources by measuring concepts that are not related to the cloud services. For example, if the cloud service description contains concepts that are irrelevant for cloud service such as news, articles, paper, weather, etc., this means that the collected source could be any website that publishes articles or provides news.
- **Service Level Agreement:** the service level agreement can be occurred in HTML home page of web source as many forms such as terms and condition, privacy and condition, terms of service, service level agreement and SLA. Therefore, if we find Service Level Agreement, we will be represented as true for the Web source. Otherwise, it will be represented as false for the Web source.

- 
- **Cloud Service Support:** most of cloud service providers offer support for the services. However, the support may be in many ways such as 24\*7 services, help desk, support etc.
  - **Cloud Service Accessibility:** most of the cloud service providers allow customers to access the services by using user name and password. Therefore, we consider accessibility provided by cloud service providers as one feature to identify cloud services.

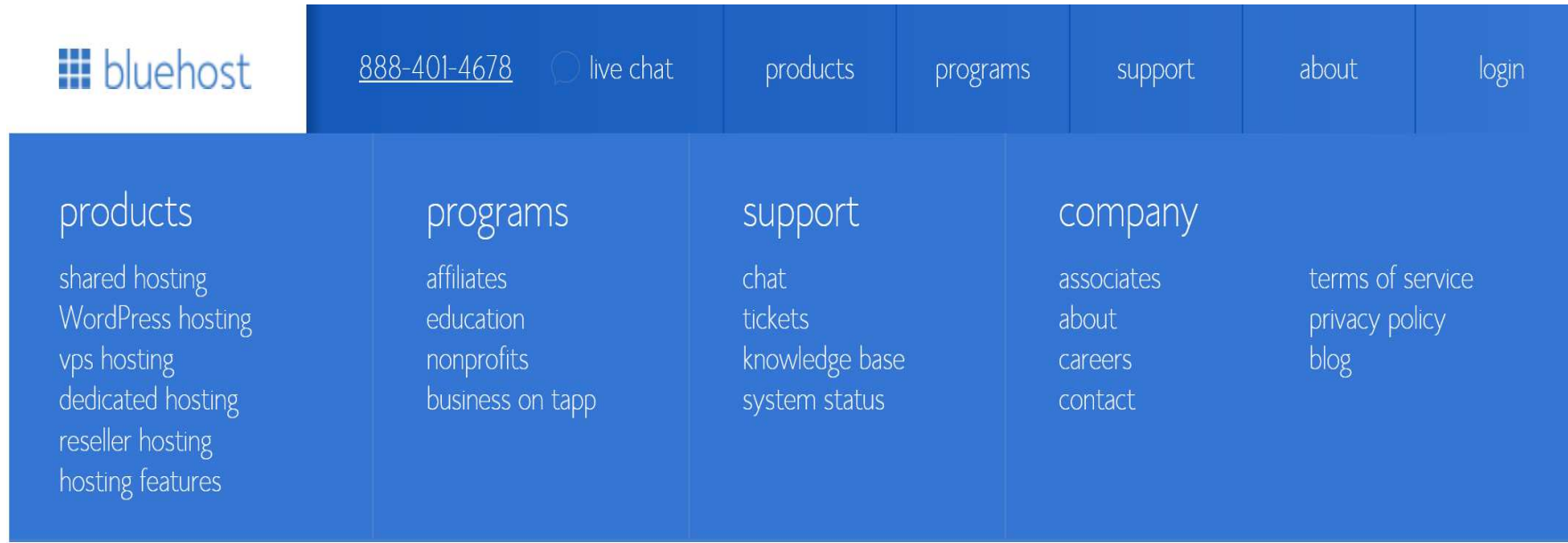


Fig. 4.3 Accessibility, Support and Service Level Agreement features<sup>2</sup>

<sup>2</sup><https://www.bluehost.com/>

- **Cloud Service Availability:** cloud service availability is represented using true and false. If a cloud service provider is available then it is true because we can extract the feature. If the cloud service is unavailable then it is false and is regarded as a non-cloud service source.

### 4.2.2 Cloud Service Identification Text Features

Instead of learning cloud services' features, we consider cloud services as documents then we implement Latent Semantic Analysis [87] to provide the features to classification method. The proposed uses both information retrieval and machine learning techniques to identify cloud services. The techniques of identifying task using text features consists of number of steps. Firstly, we aim to build documents corpus which is cloud service sources. Therefore we assemble a large collection of cloud services providers' homepages and other homepage are highly related to cloud service but they are not real cloud services. This documents corpus is generated from the cloud services' homepage  $S = \{s_1, \dots, s_n\}$ . Moreover, it exploits 5882 real cloud service and 5000 non cloud services. Then, a document matrix is built to include the documents and is vectorized each document using the following weighting function  $t = tf/(tf - td)$  where  $tf$  denotes the term frequency,  $tf - td$  the other total terms appear in the documents  $s = \{t_1, \dots, t_n\}$ . Since we achieve highly dimension of terms, we try to reduce the dimensionality of the terms by using Latent Semantic Analysis (LSA) which is implemented using randomized Singular Value Decomposition (SVD) to build our document matrix. After we have built the document matrix we consider each term that uses weighted function as features. Finally, we utilize the cloud service text features to apply the classification method. We use k-Nearest Neighbor Classification [114] which uses a Given data matrix of cloud services' terms  $T = \{t_1, \dots, t_n\}$  with K classifier and  $t$  vector  $s \in S$ . This classifier can find the nearest class to cloud

service vector  $s$ . The classification method can be helpful in distinguishes between cloud service and non cloud services.

### 4.2.3 Cloud Service Clustering

Since using cloud service text features can be used to support identifying cloud service, identifying a portulaca service with its features is a totally different task and need to extract new features and using different techniques. Therefore, we use clustering approach to find the service features by adding the cloud services' web page hyper-links as features. Fig 4.3 shows the cloud service hyper-links features which can be used to support in building cloud service profile features. In addition, we consider each hyper-link occurring in cloud serviced home page as features to build the clusters. This clustering can assign each cloud service provider as nearest as possible to ease discovering the service type and its features. The process of clustering consists of two step. In the beginning, the clusters are built based on the term vectors that are weighted by term frequency. After we cluster the cloud services' terms. wWe use the cloud service hyper-link  $s = \{h_1, \dots, h_n\}$  as features weighted by *Term Frequency* that shows in cloud services home page. Then, we apply K-Means [69] which gives a number of clusters K that desirables and improved iteratively the Euclidean distance between each data point and the centroid nearest to it in our experiments. This processing is able to decrease the distance during applying Service Detection and Tracking process because it shows that the similar type of cloud service more frequently appear in the same cluster.

### 4.3 Cloud Service Profile

In this section we demonstrate the process of generating cloud service profile. The process detects a service then tracking this service over cloud services. The process consists of three phases, including Cloud Service Modelling, Service Detection and Tracking and Cloud Service Extracting and Storing. Algorithm 3 describes the cloud service profile process.

---

#### Algorithm 3: Cloud Service Profile Algorithm

---

**Require:** A set of cloud service sources belong to a cluster  $\mathbb{C} = \{S_1, S_2, \dots\}$ ,  $HF$  vectors  $\overline{s_1}, \overline{s_2}, \dots, \overline{s_{|\mathbb{C}|}}$  for each cloud service source in  $\mathbb{C}$ , the number of nearest cloud service is  $K$ , the target cloud service source  $S_t$

**Ensure:**  $\mathbb{NS}$ : a subset which contains  $K$  nearest cloud service sources of  $S_t$  from  $\mathbb{C}$

```

 $\mathbb{NS} \leftarrow \emptyset;$ 
while  $\mathbb{C}$  is not empty do
   $\mathbb{NS} \leftarrow \emptyset;$ 
  Add  $S_t$  to sub set cluster  $\mathbb{NS}$ ;
  Remove  $S_t$  from  $\mathbb{C}$ ;
  for each  $S_i \in \mathbb{C}$  do
    Compute  $\text{cosine}(S_t, S_i)$  based on  $HF$  vectors  $\overline{s_t}$  and  $\overline{s_j}$  for cloud service sources  $S_t$  and  $S_j$ ;
    Add  $S_i$  to subset  $\mathbb{NS}$ ;
    Sort  $\mathbb{NS}$  based on similarity score
    Pick up top  $K$  cloud service sources in  $\mathbb{NS}$ 
  end for
end while

```

---

Table 4.1 Cloud Service features

Constant Features	Variable Features
Cloud Providers	Cloud Service CPU Name
Cloud Service Name	Cloud Service CPU Capacity
Cloud Service URL	Cloud Service Memory Capacity
Cloud Service HTML Tag	Cloud Service Storage Capacity
Cloud Service HTML ID Tag	Cloud Service Storage type
Cloud Service HTML Class Tag	Cloud Service Price

```

CSPN : Godaddy
CSN : VPS
CSPURL : godaddy.com/pro/managed-vps
CSPHTML : DIV
CSPHTMLID : plans plan-container plan-
CSPHTMLclass : null
CSMemeory : 1GB RAM
CSStorage : 40GB
CSprice : 43.99/mo
CSBandwidth : unlimited
CSSystem : Linux

```

(a) Cloud Service VPS Godaddy

```

CSPN : Dropbox
CSN : Storage
CSPURL : www.dropbox.com/business/pricing
CSPHTML : DIV
CSPHTMLID : plans-table__pricing
CSPHTMLclass : null
CSStorage : 1TB
CSprice : 11.58/mo
CSUsers : 1

```

(b) Cloud Service Storage Dropbox

Fig. 4.4 Cloud Service Modelling

### Cloud Service Modelling:

Cloud service modelling is embedded in cloud service features. These cloud service features' have been achieved by observing several of cloud services in real environment. In addition, the cloud services features are different from service to service. For example, VPS features are totally different from storage features but they might share some common features see Fig 4.4. However, we provide some of constant features for all cloud service types that are provided by cloud service providers. Table 4.1 shows constant features and variable features.

### Service Detection and Tracking:

In the *Topic Detection and Tracking* (TDT) a topic is defined as a seminal event or activity, along with all directly related events and activities [6]. In order to replace the cloud service instead of topic, a cloud services is defined as a set of utilities can provide several services. These utilities can be shared several of providers. Therefore, the Service Detection and Tracking is a novel approach that has been built to discover a service then track the same type of service over bunch of cloud service providers. The service detection and tracking is able to compare the features of different services. For example, if we pick up a service from a cluster and we detect this service is VPS



then we will track this service inside the cluster based on cosine similarity score of hyper-links features because we assume this is VPS cluster. However, the service tracking is not easy task because each cloud service providers describe its service under different hyper-links. However, we observe many cloud services describe their service features under hyper-links such as plan, price, features or the name of service. Therefore, when we track a service we target to investigate those four hyper-links if they are available on cloud services' homepage, otherwise we can manually discover the cloud service.

### **Cloud Service Extracting and Storing:**

The aim of this processing is to extract cloud service features from cloud services inside the cluster. Therefore, we search for this cloud service features  $s = \{f_1, \dots, f_n\}$  inside the cluster then we start collect the cloud services' features. Our searching method uses initial point to search inside cloud service page which can be the plan, price, features or the name of service. This initial can be determined depend on the features that we already made. For example, if we search about VPS, we can investigate about VPS features such as memory capacity, storage capacity, CPU type, price and operating system. In addition, if we find these features we can add it to JSON model otherwise it becomes null.

## **4.4 Related Work**

Identifying and detecting the service is widely considered as an essential problem in several research areas such as ubiquitous computing, mobile networks, Peer-to-Peer (P2P) services, and service oriented computing [66, 75, 8, 115].

In cloud services, most studies have attempted to build cloud service ontology to identify cloud services. Deng et al. [21] create an ontological framework approach to demonstrate the structure and illustration of general cloud services and their methods

of operation . The concept of secure systems with the framework to deliver cloud service business support was introduced.

Fang et al. [25] propose the Agility-oriented & Fuzziness-embedded (AoFeCSO). It is a semantic cloud service model, and its design is agility-centered with Semantic Web Ontology Language (OWL2) fuzzy extensions. Each time there is user interaction, the fuzzy-ontology receives a rating update. Consequently, it enables the specified cloud services to be collectively sustained. Users can not only collaboratively contribute to it with their knowledge, but they can also examine the model. Thus, improving the overall operation of the cloud service data. This process permits cloud-oriented information and interactions to be gathered, which enables wide-ranging service provisions.

Rekik et al. [93] propose a description ontology that included IaaS, PaaS, and SaaS. The operational and non-operational properties of the ontology, the characteristics and infrastructure associations, the software services and platform are used to locate the appropriate cloud services.

Modica et al. [23] propose a group of ontologies focused from a business point of view as opposed to a technical one. This group assisted providers in better describing and advertising their services about business plans. Furthermore, consumers are only able to define the resources they need by the activities they undertake. With that in mind, the following ontology, Applications, Support, SLA, Market, Offer, and Request have been created in line with customer demand, and organizational viewpoints of the services providers offer.

Dastjerdi et al. [20] proposed an approach that uses ontology-based discovery for QoS-aware deployment of appliances on IaaS providers. This approach supported end user to meet their needs from range of IaaS providers based on QoS preferences. However, the ontology design only found the suited IaaS providers for end users. Furthermore, the ontology does not support PaaS and SaaS providers. Ma et al. [58] propose ontology-based resource management of cloud providers. This cloud com-

puting ontology defined the concepts that described their relations. However, the ontology has to meet cloud service requirement and has been conducted in simulated environment. Rodríguez-García et al. [96] exploit an automatic general ICT domain that can be used to discover the cloud services best matching user needs. The authors use semantic annotation in order to improve the cloud service discovery results. From cloud service descriptions, semantic content can be extracted by using the annotation platform. Then, the semantic content can be used by the semantic search engine to assist users in finding those services that meet with their requirements and expectations.

Unlike these works which do not consider the problem of how to use the cloud service web content to automatically identify cloud services on the Internet, we propose a cloud service identifier framework that helps distinguish between cloud services and other services available on the Internet by automatically generating the cloud service features based on identifying new cloud service with similar features.

## 4.5 Summary

In this chapter, we design and develop a Cloud Service Identifier Framework to provide highly accurate cloud service search results and provide useful details about cloud services' features, which can facilitate cloud service selection from search users. Moreover, cloud service identifier is built by utilizing cloud service features extracted from real cloud service providers. In addition, we propose a novel Service Detection and Tracking (SDT) approach for the detection of cloud services that inspired by the Topic Detection and Tracking model. Finally, we build a unified model to expose the cloud service's features to a cloud service search user to ease the process of discovering and comparison among a large amount of cloud services.

The framework provides a Cloud Service Discovery Engine for automatically detect and discover cloud services, explained in Chapter 5. We conduct extensive exper-

iments to validate our proposed approach. The results demonstrate the applicability of our approach and its capability of effectively identifying and categorizing cloud services from the World Wide Web in Chapter 6.

# Chapter 5

## Automated Discovery of Cloud Service Using Crawling and Data Collector

Nowadays cloud services are being increasingly used by professionals. A wide variety of cloud services are being introduced every day, and each of which is designed to serve a set of specific purposes. Currently, there is no cloud service specific search engine or a comprehensive directory that is available online which include all cloud services. Therefore, cloud service users might use the current search engine to find a cloud service that fit with their requirements. However, according to [111] it show that 91% of searchers do not go past page 1 of the search results and over 50% do not go past the first 3 results on page 1. Furthermore, cloud services are provisioned at various levels, not only data and business logic, but also infrastructure capabilities. Moreover, cloud service providers may not follow a standard to describe their services and resources when publishing them. By taking into account this issues, we assure discovering cloud service automticllay is complex task. In this study, we address some research questions centered around cloud services discovery are as the following:

- Does cloud service already available on the Web?

- Is it possible to automatically discover cloud service?
- How many cloud services are available on the Internet?
- What kind of cloud service providers are there on the Web?
- What technologies are used to publish their services?

Therefore, in this chapter we demonstrate our proposed Cloud Service Discovery Engine (CSDE) that has capability to automatically discover cloud service discovery. CSDE discovery search engines allows cloud service consumers to search for the cloud service that suits their need, as well as is able to collect cloud service information available on the Web, as well as provide details about cloud service. The collected data will exploit our cloud service identifier framework to automatically identify cloud service (Chapter 4). Also, it will employ cloud service categorization framework to categorize cloud service based on cloud service model Chapter 3). Therefore, our discovery engine has the capabilities to collect, identify, categorize and build cloud services profile automatically. This chapter is organized as follows. In Section 5.1, we give an overview of the proposed cloud service discovery engine. In Section 5.2 and Section 5.3, we present cloud service crawling procedures and describe the dataset collection, respectively. In Section 5.4, we discuss some main challenges for crawling cloud services. Finally, we discuss some related work in Section 5.5 and conclude this chapter in Section 5.6.

## 5.1 Cloud Service Discovery Engine

Figure 5.1 depicts the architecture of CSDE, which consists of seven major layers, namely the *Cloud Services providers Layer*, the *Cloud Services Crawler Layer*, the *Cloud Services Extracting Data Layer*, the *Cloud Service Ontology Layer*, the *Cloud Service Identification Layer*, *Cloud Services Categorization Layer* and the *Cloud Services Consumers Layer*.

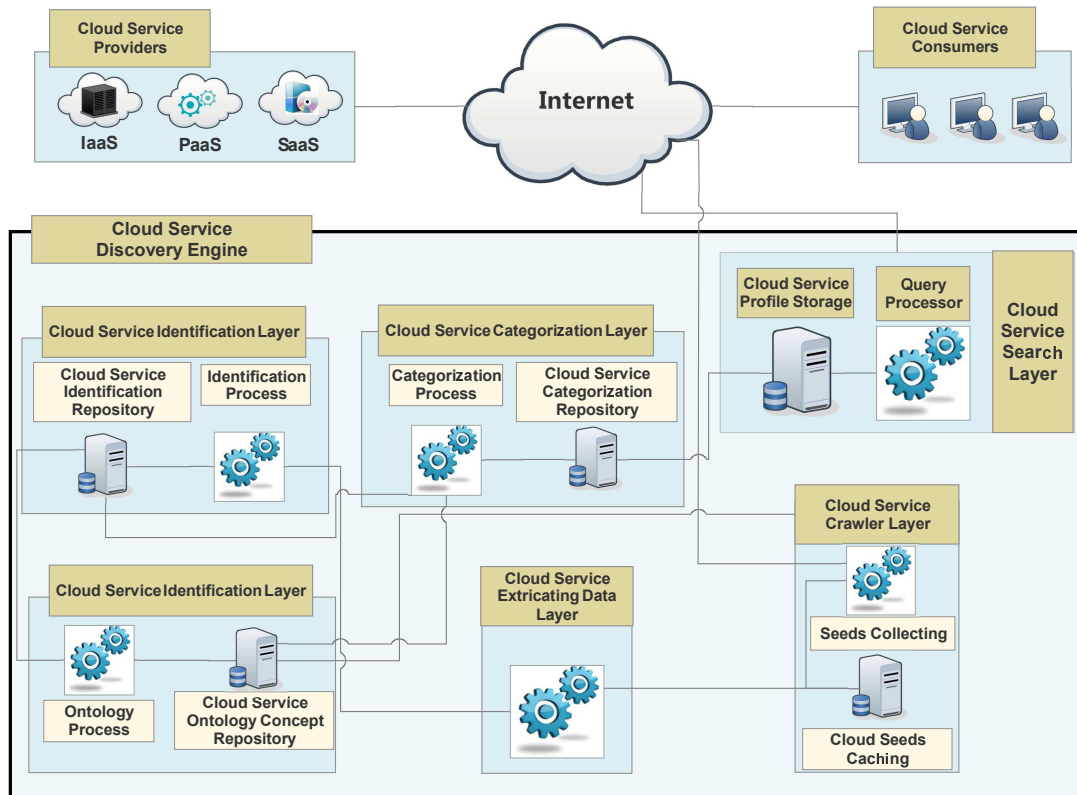


Fig. 5.1 Architecture of the Cloud Service Discovery Engine

*Cloud Service Providers Layer.* This layer consists of various cloud service providers. Those providers are publicly provision and advertise their cloud services on the Web (e.g., IaaS, PaaS, and SaaS). Our cloud service crawler can access these cloud services through Web-portals and indexed on search engines such as Google and Bing.

*Cloud Services Crawler Layer.* This layer is responsible for automatically crawling the Web and collecting cloud service seeds (e.g., cloud service URL). In addition, we develop the cloud service seed caching inquiry based module that has the capability to inquire about the seed to determine whether this seed has been cached in cloud service seeds repository. More information on the Cloud Services Crawler can be found in Section 5.2.

*Cloud Services Extracting Data Layer.* This layer is responsible for extracting essential content in the Web source, which is a seed after fetch it from the Web directly, such as description, keywords, text content and hyperlinks. Firstly, we automatically

by extract the Web source content and features (i.e., from cloud service source HTML page to text). Secondly, this sources will be sent to identification layer to determine whether this source is cloud service or not. In addition, this Web sources will be cached to be used for automation categorization and build cloud service profile.

*Cloud Services Ontology Layer.* This layer is responsible for maintaining the cloud service discovery. The cloud service ontology is used to provide a set of keyword to allow the crawler to automatically discover the Web. In addition, it is used to enhance our cloud service identification and categorization. More information on the cloud services ontology can be found in Chapter 3,4.

*Cloud Service Identification Layer.* This layer is responsible for identifying cloud service provider. The Cloud Services identifier contains the process of identifying features to determine whether a given source is a cloud service or not. This processing relies on classification method to realize the identification. The identification can be updated automatically after identifying a new cloud service provider to enhance identifying knowledge. In addition, we allow cloud service provider to register their services in our system, these cloud service that registered can be added and can be recognized by the identification as a new cloud service provider. More information on the cloud services Identification can be found in Chapter 4.

*Cloud Service Categorization layer.* This layer is responsible for categorizing cloud services and contains the categorization processor to assign a given cloud service source to one type of cloud service model based on the acquired concepts in the cloud service ontology. This layer also relies on clustering methods to build the categorization. The categorization can be updated automatically after categorizing a new cloud service provider to increase categorization knowledge. More information on the cloud services Identification can be found in Chapter 3.

*Cloud Service Search Layer.* This layer provides a Web interface for users to search cloud services. A user can simply specify a searching keyword for finding cloud services. She can also specify other constraints (e.g., categories like IaaS) to



narrow down the searching scope. Our system will contact the cloud service service repository. If it is found in the repository, the detailed information (e.g., access link, features, description) of satisfied cloud services will be returned to the user.

## 5.2 Cloud Service Crawling Procedures

In this section, we describe our cloud service crawling procedures in details:

- *Determine the Search Engine to crawl* The critical element for collecting meta-data for cloud services is through the use of search engines. The reason behind this is, we are unaware of the related business registries, for example, UDDI for Web services. For such Web services, it is recognized that UDDI registries are unsuccessful, and Web service discovery is leaning towards the use of search engines. In 2013 our cloud services crawler explores the Web for cloud services using existing search engines' APIs (e.g., Google, Yahoo, and Baidu) to understand the current state of cloud services. However, in 2017 our cloud services crawler explores only Google and Bing because yahoo uses google search engine and Baidu provides too many non English results which are so difficult to identify because our identifier model works well on English language.
- *Keywords* cloud service crawling choose automatically the first few levels of the Cloud Services Ontology based on defined threshold as keywords to collect seeds which are the URL. Keywords can be used such as Cloud Services, IaaS, PaaS, SaaS, Communication, Storage, Infrastructure, Online Backup, Web Hosting, Virtual Desktop, Virtual Machine, Software, API, etc.
- *Verifier* The verifier is working on Cloud Services Extracting Data Layer. It is responsible to determines whether a given Web source is an active or inactive one. Inactive Web source are kept in the Inactive Cloud Services database

Table 5.1 Cloud service Collection Dataset

Cloud Service Collection		
	2013	2017
Discovered 2013	5,883	3,681
Discovered 2017	3,681	7,461
Unavailable 2017	2,202	0

for another check (i.e., some inactive Web source may be temporarily unavailable) and the error codes are also captured. An active seed will download the startpage and extract the content to pass to the cloud service identifier

- *Identifying* This process is to determine whether a given Web source is cloud service or not. If the Web source provides cloud service it will pass to categorization process to categorize based on cloud service model. Otherwise, we cache the seed at cloud service seed caching repository as not cloud service provider. (chapter 4)
- *Cloud service repository* After we identify and categorize the cloud service will be cached at cloud service repository in order to fast the access to cloud service consumers.

### 5.3 Statistical Analysis and Crawling Results

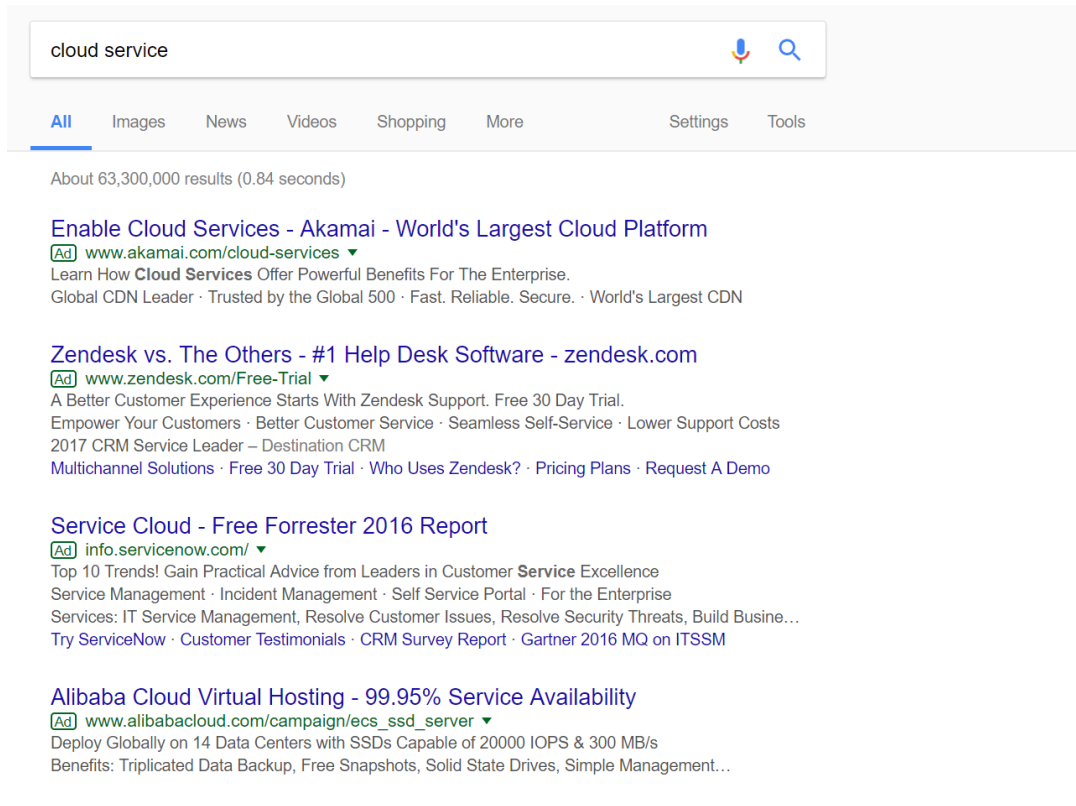
In this section, we comprehensively demonstrate a number of different aspects about discovering results and dataset collection while presenting a statistical analysis of cloud services. In addition, We compare two periods of cloud service crawling: firstly, it has been conducted in 2013 while the second conducted in 2017. Moreover, from table 5.1 we can see only 49.3% of cloud service still available on the Web to provide the services while 3,780 new cloud services we could discover in 2017 and we could not discover or were not started up in 2013. This indicate to dynamic nature of cloud service. More detials about cloud service dataset can be shown below:

### 5.3.1 Cloud Services Identification

The crawler layer collects the possible cloud services' seeds by constantly fetching search results from the search engines indexes. In 2013, the cloud service discovery engine managed to fetch 619,474 possible links and collect 35,601 Web sources for cloud services. In 2017, the cloud service discovery engine managed to fetch 844,696 possible links and collect 55,474 Web sources for cloud services. It is important to use our cloud service identifier to identify the cloud service Web sources collection because this collection is fully of noisy data such (blog, news, wiki, articles, etc..). After the using cloud service identifier, the cloud service discovery engine identified 5,883 unique cloud services in 2013 and 7,461 unique cloud services in 2017. We set several datasets of cloud services in the real environment based on this collected information. These datasets have already been released for research community. It must be mentioned that the datasets collection is tantamount to 2.46 GB of cloud services information, which is available for the research community.

WSDL Based Cloud Services have been investigated in our cloud service discovery to discover the connection between Service-Oriented Computing (SOC) such as WSDL, and cloud computing. This investigation primarily focuses on the number of cloud services that have been implemented using WSDL in order to publish its services. Our cloud service discovery engine is configured to collect the WSDL by searching files with extensions such as WSDL. In 2013, our cloud service discovery engine managed to fetch 1,552 possible links and collected 616 WSDL sources for cloud services in WSDL. However, In 2017 we fetch 652 possible links and only and collected 421 WSDL sources for cloud services in WSDL. After the identification process, our cloud service discovery engine identified 106 valid cloud services implemented using WSDL in 2013 and 87 in 2017.

We observe during the crawling of cloud services, that certain cloud service providers advertise their c services in search engines, which usually appear on the top part



The image shows a search engine interface with the query "cloud service" entered in the search bar. Below the search bar, there are navigation tabs for "All", "Images", "News", "Videos", "Shopping", "More", "Settings", and "Tools". The search results indicate "About 63,300,000 results (0.84 seconds)". Four sponsored advertisements are listed:

- Enable Cloud Services - Akamai - World's Largest Cloud Platform**  
Ad: [www.akamai.com/cloud-services](http://www.akamai.com/cloud-services)  
Learn How **Cloud Services** Offer Powerful Benefits For The Enterprise.  
Global CDN Leader · Trusted by the Global 500 · Fast. Reliable. Secure. · World's Largest CDN
- Zendesk vs. The Others - #1 Help Desk Software - zendesk.com**  
Ad: [www.zendesk.com/Free-Trial](http://www.zendesk.com/Free-Trial)  
A Better Customer Experience Starts With Zendesk Support. Free 30 Day Trial.  
Empower Your Customers · Better Customer Service · Seamless Self-Service · Lower Support Costs  
2017 CRM Service Leader – Destination CRM  
Multichannel Solutions · Free 30 Day Trial · Who Uses Zendesk? · Pricing Plans · Request A Demo
- Service Cloud - Free Forrester 2016 Report**  
Ad: [info.servicenow.com/](http://info.servicenow.com/)  
Top 10 Trends! Gain Practical Advice from Leaders in Customer **Service** Excellence  
Service Management · Incident Management · Self Service Portal · For the Enterprise  
Services: IT Service Management, Resolve Customer Issues, Resolve Security Threats, Build Busine...  
Try ServiceNow · Customer Testimonials · CRM Survey Report · Gartner 2016 MQ on ITSSM
- Alibaba Cloud Virtual Hosting - 99.95% Service Availability**  
Ad: [www.alibabacloud.com/campaign/ecs\\_ssd\\_server](http://www.alibabacloud.com/campaign/ecs_ssd_server)  
Deploy Globally on 14 Data Centers with SSDs Capable of 20000 IOPS & 300 MB/s  
Benefits: Triplicated Data Backup, Free Snapshots, Solid State Drives, Simple Management...

Fig. 5.2 Cloud Service Advertisement

and/or the bottom part of the returned index pages (see Figure 5.2). We collected the contours of advertisement by using the cloud service discovery engine that collected the possible cloud services' Sources by constantly fetching advertisements appearing on the top or bottom side of the search results (i.e., these advertisements are called Ads or sponsored results), and eventually identified 637 unique cloud services in 2013 and 708 in 2017.

### 5.3.2 Cloud Service Extracting Data Error

By looking at table (5.3), we make a comparison of the error that occurred in 2013 and 2017, we can see that the error code 1005 accounts for the highest error percentage (66.95%) (i.e., URL does not exist), which means that the most of the inactive cloud services were discontinued in 2013. In addition, gateway timeout is increased and

Table 5.2 Breakdown of Cloud Services Crawling Results

	Start Page		WSDL/WADL		Ads		Total	
	2013	2017	2013	2017	2013	2017	2013	2017
Links Fetched	617,285	843,312	1,552	652	637	732	619,474	844,696
Possible Web sources	34,348	54,321	616	421	637	732	35,601	55,474
Inactive	366	739	57	106	0	0	423	845
Active	34,619	53,582	559	315	637	732	35,815	54,629
Not cloud service	28,736	46,121	453	228	0	24	29,189	46,373
Is cloud service	5,883	7,461	106	87	637	708	5,883	7,461

Table 5.3 Error Codes for Inactive Cloud Services

Error Code	Description	Percentage	
		2013	2017
101	The connection was reset.	13.66%	9.57%
105	Unable to resolve the server's DNS address.	1.64%	1.89%
107	SSL protocol error.	0.27%	1.48%
118	The operation timed out.	0.27%	0.34%
324	The server closed the connection without sending any data.	0.27%	0.27%
330	Content decoding failed.	0.27%	0.0%
400	Bad request.	0.82%	1.49%
403	Access denied.	3.83%	8.17%
404	The requested URL / was not found on this server.	10.11%	6.27%
408	Request Timeout.	0.0%	9.37%
500	Server Error.	1.37%	12.48%
503	The service is unavailable.	0.27%	3.18%
504	Gateway timeout.	0.27%	48.67%
1005	URL does not exist.	66.95%	0.0%
Total		100%	100%

achieved the highest percentage (48.67%) in 2017 which means the most inactive cloud services are unable to connect during the discovery procedure. However, the comparison made between 2013 and 2017 suggests that the total number of inactive cloud services is a significantly low 423, or about 1.1% in 2013, while it rose slightly by 1.4% in 2007 from the total Web sources.

### 5.3.3 Cloud Service Locations

We conducted studies about the geographical status of cloud services to discover which part of the world the cloud services are provisioned from. In order to visualize the cloud service data in accordance to countries, we extracted the country's domain of each URL. In case a country domain is not present, we used whois-capable online tools such as <http://ipduh.com/ipv6/whois/> and <http://www.sixxs.net/tools/whois/> to determine the location of the URL, tracing back to the geographical location of the hosting data center. The tools only accepted ip addresses as the input. Finally, the percentage of each country is grouped into different regions in order to present a holistic view of computing trends on a world map, as shown in Figures 5.3. and 5.4. The information outlines the details of a specific nation; we paint it in a special color based on the proportional range of the cloud services provisioned by that country. From table 3, we can observe that North America region has the highest rate with regard to the number of cloud service providers, a percentage of 60.45% in 2013 and in 2017. This is followed by Europe 23.27% in 2013 and in 2017. The cloud services are provisioned from Asia 8.7% and 5.27% from Australia while in 2017 The cloud services are provisioned from Asia 8.7% and 5.27% from Australia. The most of cloud service are come from USA and Australia because in our 2017 cloud service discovery we focus on those two region.

continents	Percentage	
	2013	2017
North America	60.45%	60.24%
Europe	23.27%	18.25%
Asia	8.7%	3.44%
Australia	5.27%	17.01%
South America	1.04%	0.0%
Africa	1.27%	1.06%

Table 5.4 Cloud Service Location

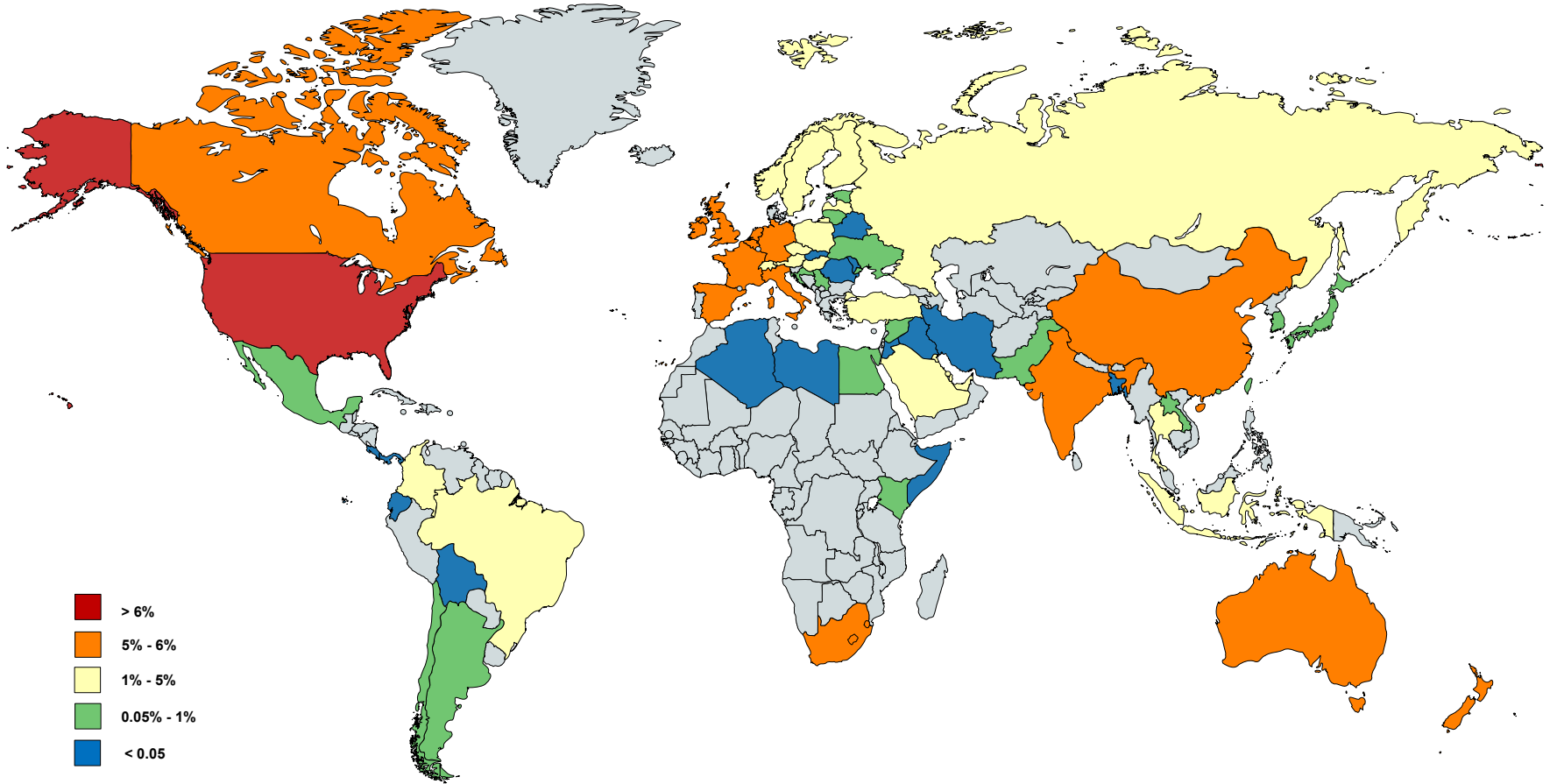


Fig. 5.3 Cloud Service Locations 2013



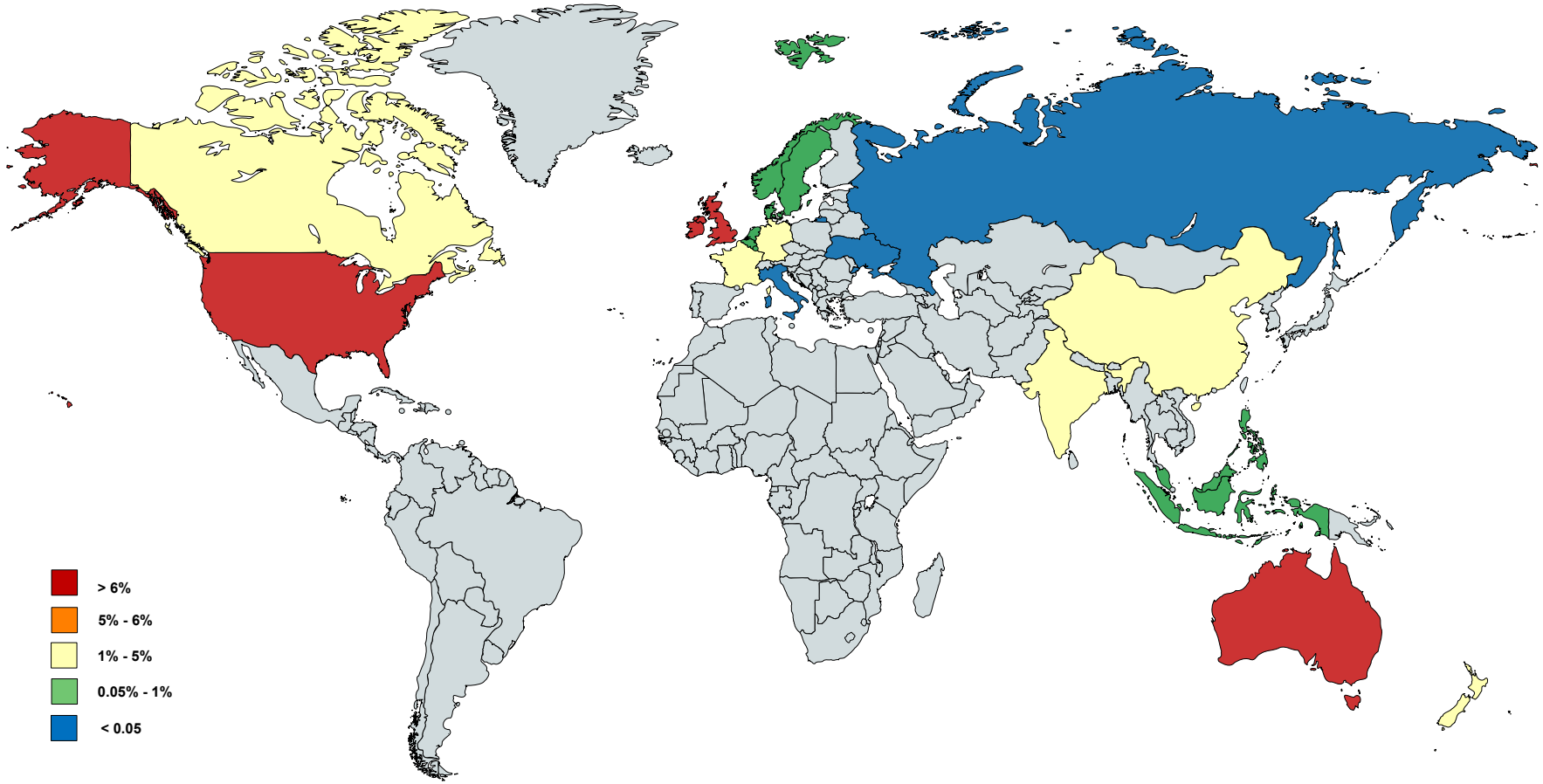


Fig. 5.4 Cloud Service Locations 2016

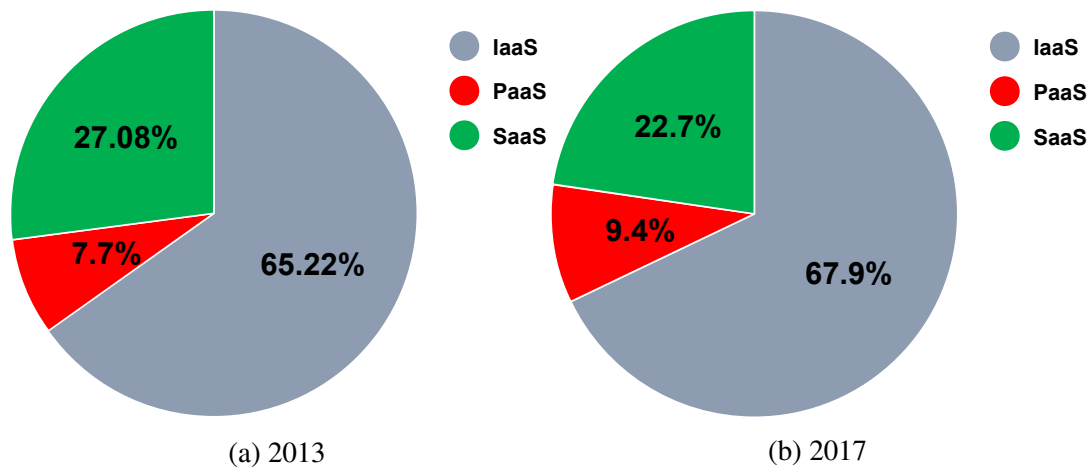


Fig. 5.5 Cloud Service Categorization

### 5.3.4 Cloud Service Provider Categorization

Cloud services are typically classified by different service providers into IaaS, PaaS, and SaaS. It would be interesting to find out the percentages of different kinds of cloud service providers. As described in cloud service crawling procedures, after our cloud services identification finish identifying of cloud service Web source, our cloud service categorization can categorize these cloud services into IaaS, PaaS or SaaS by using our cloud service categorization that has been in described in chapter 3. Figure 5.5 depicts the categorization results wherein cloud service providers are categorized into three different categories, namely, IaaS, PaaS and SaaS. More over, most cloud service providers usually provide more than one services but they focus on service model to provide. Figure 5.5 show that the Infrastructure as a Service dominate the interest of cloud service providers.

### 5.3.5 Cloud Computing and Service-Oriented Computing (SOC)

Service-Oriented Computing (SOC) and Web services play a key role in adopting the desired technologies for cloud computing. Therefore, it is important to understand how to investigate the impact of Service-Oriented Computing (SOC) and Web services in cloud computing. This is about the preliminary studies conducted on the

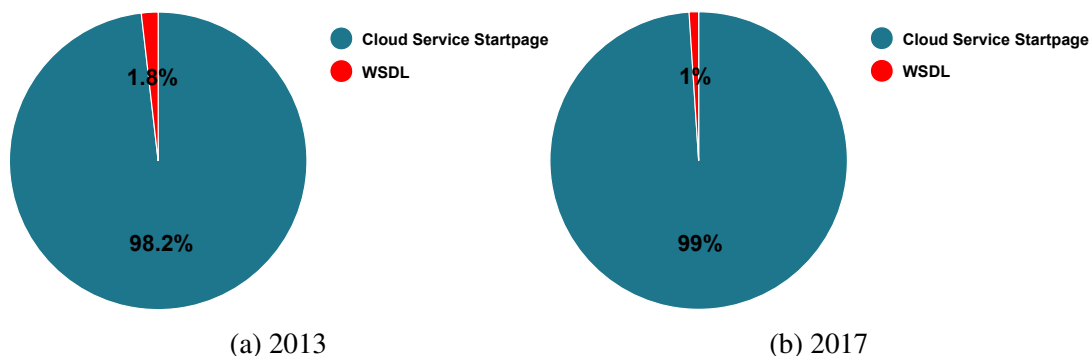


Fig. 5.6 Service-Oriented Computing

information that we collected in 2013 and 2017. Firstly, we investigated how many description languages from Service-Oriented Computing such as Web Services Description Language (WSDL) are used to publish cloud services in a real environment. Figure 5.6 shows the results of comparing the number of cloud services that used WSDL such as Simple Object Access Protocol (SOAP) based Web services. Moreover, WSDL does not play a significant role in cloud computing due to their lowest percentage which stands at less than 1% in the context of implementing Web service interface languages. Due to not all WSDL documents are publicly accessible on the Internet we can not assure that we detect all WSDL documents by our cloud service discovery engine. In addition, the majority of RESTful Web services provides no formal descriptions and are predicated on informal documentation [94] Nevertheless, the extremely low percentage 1 indicates the poor adoption of SOC in cloud computing.

### 5.3.6 Cloud Service Advertisement

Cloud service providers make use of search engines advertisement as a way to market their services so they can gain new clients. With this in mind, our cloud service engine can recognize and hone in on this type of cloud service adverts. Customers may have noticed that most adverts are located either at the top or bottom side of search page results. Figure 5.7 shows roughly 10.8% of all cloud service providers paid for their adverts during 2013, with the aim of gaining new customers. However,

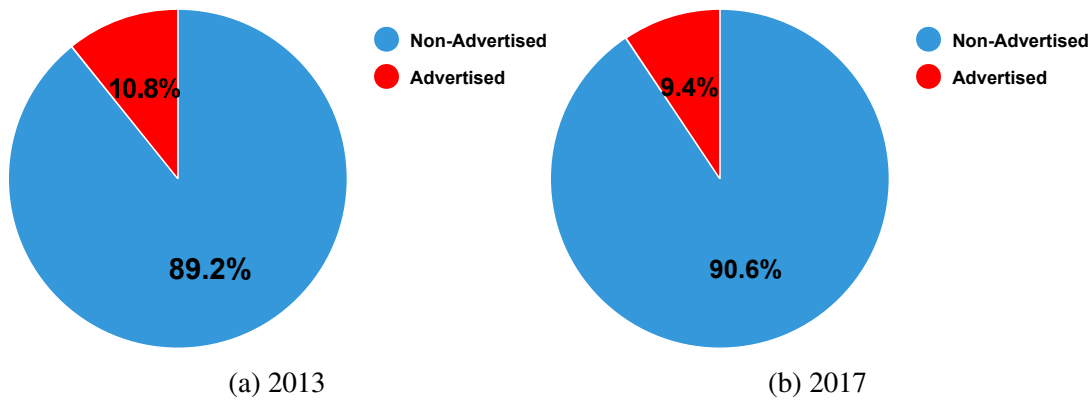


Fig. 5.7 Cloud Service Advertisement

this percentage dropped to 9.4% in 2017. Nonetheless, adverts for cloud services tend to be limited, typically setting out very brief details or purpose. The reason for this being there is only a limited text description below the advert in search engines.

### 5.3.7 Cloud Service IP

We investigated an important aspect of cloud services (i.e., what type of IPs do cloud services use). The process of obtaining ipv4 and ipv6 addresses is ascertained through running the nslookup command that is found in the command prompt of the Windows operating system. By this technique, a simple Java program is written to enable the automatic retrieval of such IP addresses from the URLs of the dataset. As shown in Figure 5.8, the majority of cloud services (97.42%) use IPv4 in 2013 while the usage of IPv6 has increased in 2017. However, it is a fact that IPv4 remains the most widely deployed Internet Layer protocol.

### 5.3.8 Discussion

Crucial analysis can be established based on present of our observations. Initially, we have determined that in 2013, cloud services were 5,833, but they increased in 2017 to 7,461. This reveals that cloud services are incredibly dynamic. Further, during 2013, 37.4% of cloud service analysis is unavailable owing to the extensive

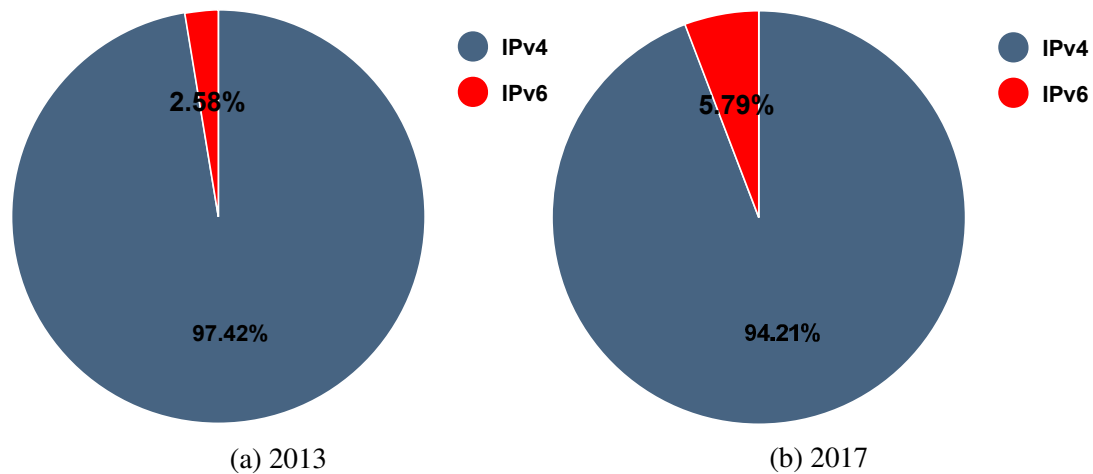


Fig. 5.8 Cloud Service IP

volume of cloud service providers. Furthermore, the use of IPv4 was seen to be one of the leading cloud services, signifying that IPv4 remains one of the dominant Internet Layer Protocol deployments used. Lastly, this highlights the requirement for total standardization, more specifically, regarding description languages to incorporate cloud computing. There is also an urgent need for standardization especially in description languages to fully embrace cloud computing [12]. Further analysis and research should be undertaken to gain a new understanding of reasons behind this, and to facilitate SOC in contributing to cloud computing. The purpose is to benefit from previous R&D (research and development) efforts within the communities of SOC. Likewise, with standardization currently lacking in cloud services, the discovery of cloud services becomes problematic, causing inaccessibility to integrated cloud services.

## 5.4 Challenges of Discovery Cloud Services

Despite detection, authentication, validation, compilation, and classification can be an automated process for cloud services - it is nonetheless a complicated task. For

a detailed dataset to be collected, a cloud service crawler which can overcome many challenges must be designed. These problems include:

### **The Nature of Cloud Services.**

The cloud services market is highly dynamic, and just as one emerges on the Web, another cloud service is discontinued and disappears. This seems to happen on a 24-hour basis! Further, cloud services alter over time, and consequently, the discovery engines must be capable of periodically updating and reexamining cloud services, so that the repository is kept updated. Differentiating which cloud services should be revisited is essential, for example, ones updated frequently and new cloud services and those which can be omitted, such as, ones updated infrequently or discontinued. Owing to the potential numbers of cloud services available on the Web, having the ability to divide them into varying groups or sections will significantly enhance crawling performance.

### **Lack of Standardization.**

Web services can be found by merely by searching the UDDI registries or gathering WSDL documents. However, with cloud services, description languages are not standardized, making them more difficult to locate and in effect, one of the more unusual problems within cloud service discovery. It can be seen that under 2% are describing their services using WSDL. Noisy and redundant data are in abundance, some examples of this are blogs, reviews, news broadcasts and study papers relating to cloud when it comes to cloud services crawling. To illustrate, our discovery search engine, gathered initially from the Web, 47,360 potential Web sources. Out of which, 15.75 percent (7,461) of those were cloud services during 2017, whereas in 2013, it was 16.14

**Crawling Blockage.**

It is clear that high-performance has been substituted for resource use. Resources from other organizations can be consumed by our discovery search engine and certain other Websites, causing the crawler to become obstructed and unable to access those services. Consequently, the performance of the cloud service crawler must be enhanced but devoid of service provider/resource consumption, thereby reducing the load burden on the network (i.e., utilizing the crawler from a number of locations/IP addresses).

## 5.5 Related Work

Many researchers has been used crawling approach to solve the problem of service discovery. For axample, ArnetMiner [108], a service used to index and search academic social networks. ArnetMiner has crawled the Web to find researchers and publications. In addition, ArnetMiner use a friend-of-a-friend (FOAF) ontology which consists of two concepts which are *researcher* and *publication*, 24 properties of publication and two objects relation to link between the authors, their publication and publication properties.

In addition, crawling the social medias has been used wildly to analyze the characteristics of users in the social media or to detect the spammer. [9, 11].

Web service discovery is a dynamic area of research and has achieved some excellent results. Mahmoud and Al-Masri [2] crawl the UBR (UDDI Business Registries) and gather WSDL documents. They also explore search engines, for example, Baidu, Google, and Yahoo. They gather data which is then analyzed to produce an in-depth statistical report about Web services, for instance, Active vs. Inactive. Li et al. [55] similarly accumulated Web service statistics via Google's Application Programming Interface (API) and produced some thought-provoking results relating to Web services process, scope, distribution and performance diversity.

For cloud services, there have been several attempts to build an ontology for cloud services discovery. For example, Kang and Sim [46] propose a Cloud Service Discovery System using ontology based approach to discover cloud services. This discovery system provides users with cloud services depending on their requirements. However, cloud service providers still need to register at the discovery system in order to publish their cloud services to allow the users to discover them. Furthermore, this ontology uses agent-based methods to find similarities in order to provide the services, including similarity reasoning, equivalent reasoning and numerical reasoning. Zhang et al. [125] propose the Cloud Recommender system to select cloud infrastructure based on their cloud computing ontology called (CoCoO). This ontology defines functional and non functional concepts of infrastructure services with their attributes and relations. However, the ontology has not provided any details about PaaS and SaaS. As well, the validation of ontology concepts is only limited for some cloud infrastructure providers such as Amazon, Microsoft Azure, GoGrid, etc. Other researchers propose to use Distributed Hash Tables (DHTs) for better discovery and load-balancing of cloud services. For example, [91] propose the concept of cloud peer that extends DHT overlay to support indexing and matching of multidimensional range queries



(i.e., the dimensions can include service type, processor speed, available memory, etc.) for service discovery. The proposed approach is validated on a public cloud computing platform (Amazon EC2). Their work focuses on a closed environment [83]. In contrast, we focus on discovering cloud services on an open environment (i.e., the World Wide Web) to allow users to search cloud services that suite their needs. Unlike previous works which did not consider the problem of how to use the cloud service crawling to automatically identify and categorize cloud services on Internet. We propose an automated cloud service search discovery that helps discover cloud services among other Web sources available on the Internet and automatically identify and categorize cloud services.

## 5.6 Summary

In this chapter, we have used a crawler for automatic enhance cloud service discovery engine. Our cloud service discovery engine is able to crawls search engines to collect cloud service information available on the Web and has the capabilities to collect, validate, and categorize cloud services. By continuously crawling resources on the Web, it can continue to maintain an advanced cloud service source for useful and proficient cloud service discovery. We used the cloud service discovery engine to do this task twice in 2013 then 2017. Our automatic crawler managed to fetch 619,474 possible links and collected 35,601 possible Web source for cloud services in 2013 while in 2017 the cloud service discovery engine managed to fetch 844.696 possible links and collect 55,474 possible Web source for cloud services.

In the next Chapter, we conduct extensive experiments to validate our proposed approach. The results demonstrate the applicability of our approach and its capability of effectively identifying and categorizing cloud services from the World Wide Web.



# Chapter 6

## Implementation and Performance Study

This chapter describes the implementation and performance study of our proposed Cloud Service Categorization Framework and Cloud Service Identifier Framework [5, 4, 3]. To validate the feasibility and benefits of our approach, we conduct extensive experimental and performance studies of the proposed techniques using a collection of real-world on cloud services. First, we describe the cloud service dataset that used to conduct the experiments. Secondly, we conduct a set of statistical analysis and present based on the Cloud Service ontology. These statistical results offer an overall view on the our cloud service ontology. Thirdly, we conduct extensive experiments to validate our cloud service categorization approach. The results demonstrate the applicability of our approach and its capability of effectively categorizing cloud services on the Internet. Finally, we validate and study our identifier model from various aspects including accuracy, precision, and recall.

## 6.1 Cloud Service Dataset

To implement our proposed Cloud Service Categorization Framework and Cloud Service Identifier Framework, we use real cloud service Webpages metadata of the 5,883 valid cloud service (see Chapter 5). This data has been chosen due to it was verified and valid. However, these Webpages metadata still required to be filtered to achieve the cloud services sources. More specifically, we cleaned the HTML tags and non-English cloud services in the cloud services metadata. For removing HTML tags we used HTML Parser <sup>1</sup>. As well we used the language detection library<sup>2</sup> to remove non-English cloud services metadata. Finally, we achieved the cloud services sources containing only English language cloud services. We eventually obtained a set of 5,083 cloud services' sources containing only English language. The full data set involves in our experiments.

## 6.2 Cloud Service Ontology

Based on the collected sources of cloud services, we ran the system to generate our cloud services ontology by using the proposed CSO algorithm. The cloud service sources contained 1,935,185 terms. When setting the threshold frequency as 500, we obtained 654 candidate concepts after the first step. Some of the candidate concepts were not considered as the CSO concepts or simply stop words (e.g., numbers and dates). In the validation step, we obtained 105 new concepts for cloud services ontology. For example, `Web_application` was found 510 times. Then, we found that the `host` was the most common concept to appear together with `Web_application` in cloud service sources. Therefore, we linked `Web_application` to `host` and further determined `Web_application` to be the child for `host` because the  $TF*IDF$  weight

---

<sup>1</sup><http://htmlparser.sourceforge.net>

<sup>2</sup><https://code.google.com/p/language>

of `host` is less than the weight of `Web_application`. The concepts with greater frequency are at a higher level in the CSO tree.

Table 6.1 shows the top 10 high frequent concepts in CSO that appears in cloud service source. From the table we can see that the most common concept is cloud infrastructure services. Moreover, the number of cloud service providers that provide infrastructure services is higher than those that offer platform or software services. From our statistics, it is also interesting to note that some cloud service providers do not use the concept of “cloud service” when advertising their services on the Internet.

Table 6.1 Top 10 High Frequency Concepts in CSO

CSOc	# frequency	# appear CSTD	Percentage CSTD
host	23423	2910	57.7%
server	14151	2500	49.6%
application	6704	2160	42.8%
network	6746	2074	41.1%
Cloud computing	3341	1394	27.7%
backup	4562	1289	25.5%
Web hosting	5152	1103	21.8%
compute	2781	926	18.3%
cloud service	1598	836	16.5%
communication	1939	824	16.3%

### 6.2.1 Threshold Identification

To evaluate our cloud service ontology we measure its capability in identifying cloud services in real environment. Therefore, it is important to estimate the threshold value in order to identify cloud services. Thus, we propose using a *confidence interval calculator* to estimate the threshold identification [98] in the cloud service source for identifying cloud services. This is because it provides the identifier for cloud services, which is the ratio between the number of cloud service concepts and the total number of terms appearing in its corresponding document. To calculate the estimated value

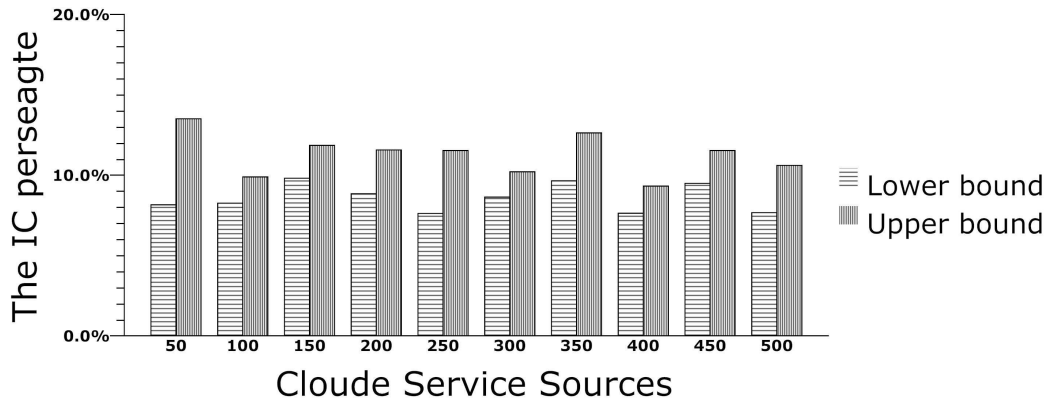


Fig. 6.1 Identifying E Threshold

for identifying cloud service, we use the following equation:

$$E \pm = \mu \pm z \cdot \frac{\sigma}{\sqrt{N}} \quad (6.1)$$

where  $\mu$  represents the mean of  $IC$  in the cloud services source;  $\sigma$  represents the standard deviation of the cloud services source; and  $N$  represents the total number of cloud services sources. In our experiment, we set the value of  $z$  to 95% since this is a common setting and useful in conducting estimations as well as providing close accuracy of a population set for the estimation equation. We ran the experiment randomly using 2,750 cloud service sources to identify an optimal E threshold for identification  $IC$ . In the first round, we randomly selected cloud services sources, then dynamically increased the number of cloud services sources at each round. Fig. 6.1 shows the lower bound, the upper bound for  $IC$  in each set of cloud service sources. We can see from the figure that the lower bound of cloud services ontology is between 7% to 8% of the cloud services source, whilst the upper bound can be between 11% to 14%. This experiment indicates an interesting fact that most cloud services use 8% to 14% of ontology concepts in their service descriptions.

## 6.2.2 Identifying Cloud Services Using Ontology

We conducted an experiment to identify cloud services for proving the robustness of the search engine. In this experiment (see Fig. 6.3), we collected randomly 550 Web sources which are fake cloud services. Additionally, we used 2,200 cloud service sources to run two experiments for evaluation of precision, recall and f-measure. The precision represents the percentage rate of distinguishing between real and fake cloud services, whilst recall represents the percentage of cloud services identified and f-measure shows the mean between recall and precision. At the beginning, the experiment started with 40 real cloud service sources and 10 fake cloud services sources. Then, the number was increased dynamically. The first experiment used the upper bound of the threshold identification while the second used the lower bound of the threshold identification to compare the results. In the experiments, it was concluded that the upper bound threshold identification provides higher precision and average recall, which indicates that the upper bound threshold can distinguish sources better. However, it provides just average performance for identifying cloud services. The lower bound threshold identification provides higher recall and high average precision, which indicates that the lower bound threshold can more effectively identify cloud services. However, it provides more fake cloud services to the search engine.

We conducted another experiment that shows in Fig. 6.4 to study the impact of the noisy data on our cloud service search engine. The noisy data means the fake cloud service results of the cloud service crawler. In this experiment, we compared the cloud service crawler repository which did not use the cloud service identifier and the one which did use the cloud service identifier. Fig. 6.4 shows that the percentage of the noisy data decreased to 10% which gives the cloud service identification repository high robustness for cloud service search engine. However, from Fig. 6.4 we can see that without using the identifier, with the increase of the number of crawled cloud

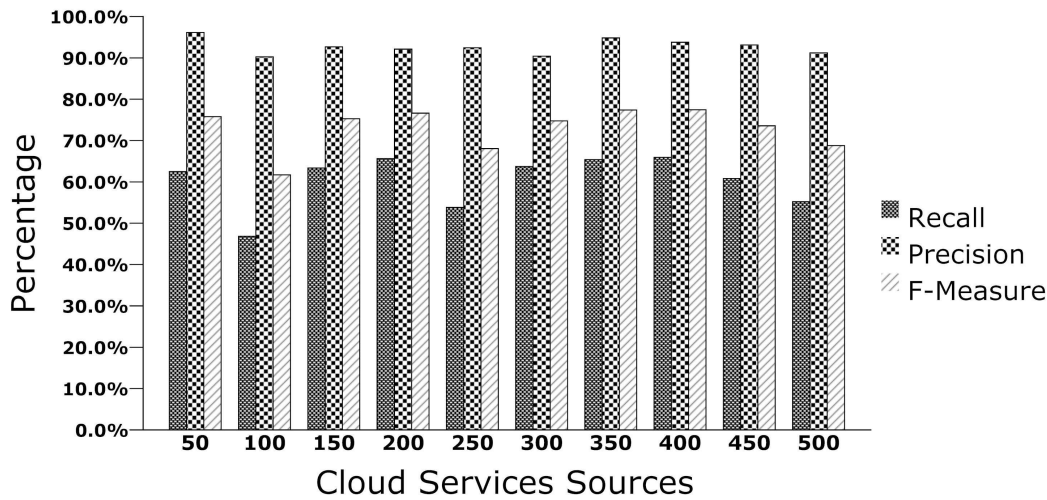


Fig. 6.2 Upper Bound Threshold

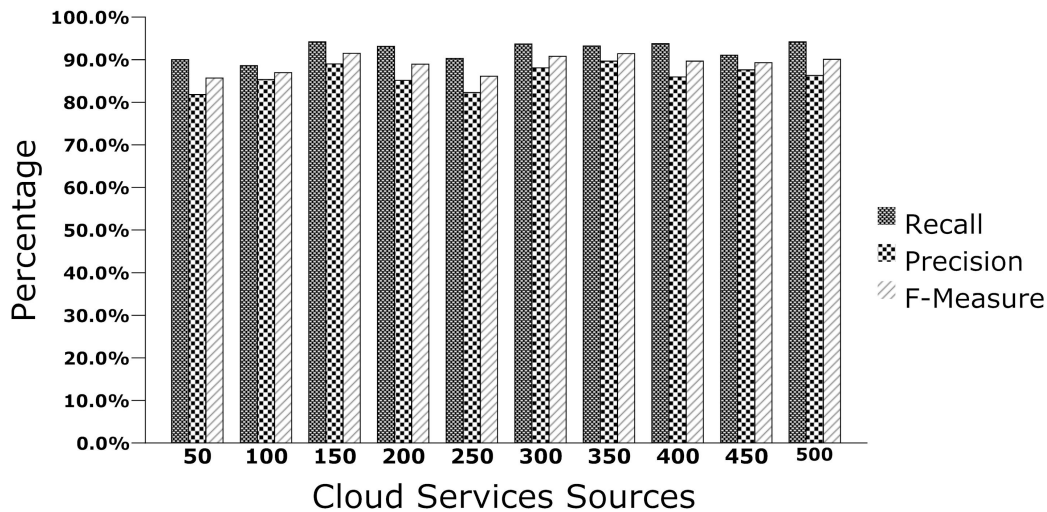


Fig. 6.3 Lower Bound Threshold

services, the number of fake cloud services also increases. On the other hand, by using our identifier, fake cloud services can be successfully identified.

### 6.2.3 Discussion

Current approaches usually focus on using ontologies in order to meet end users requirements. Our approach exploits a cloud service ontology to find cloud service through the Internet. Many of the existing approaches have validated their solutions



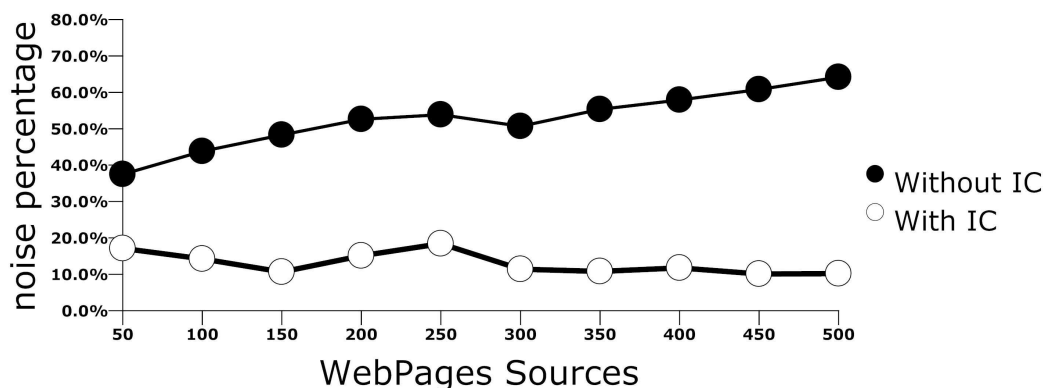


Fig. 6.4 Noisy Data of Cloud Service Search Engine

by using a simulated environment or concentrating on only popular cloud service providers (e.g., Amazon Web Services). In our work, we use a large number of real cloud service sources collected from the Web. An essential technique that should be considered is about automatic building of cloud service ontology. Most of the cloud service ontology has been built manually [46] [125] [58] while Rodríguez-García et al. [28] have used general ICT domain ontology to find the best match for users' requirements. However, the large number of concepts in ICT domain can lead to mistakes when identifying cloud service. Therefore, our approach has used Semi-Automatic technique to enhance cloud service discovery. Furthermore, it concentrates on the most common concepts that appear in cloud service sources to archive high accuracy in identifying cloud service. In the experiments, we use Cloud Service Categorization Similarity to identify cloud service but it shows less accurate results than the lower and upper bound threshold identification. Furthermore, we used the ratio of cloud service identifier instead of considering each concepts as individual feature to identify cloud service, because we found many sparsity that produced high noise data. In addition, the upper bound threshold identification provides higher precision and average recall, which indicates that the upper bound threshold can distinguish sources better. However, it provides just average performance for identifying cloud services. The lower bound threshold identification provides higher recall and high average preci-

sion, which indicates that the lower bound threshold can identify cloud services more effectively. However, it also provides more fake cloud services to the search engine.

## 6.3 Cloud Service Categorization

In this section, we ran the cloud Service Categorization experiment based on the Cloud Service Categorization algorithm,

### 6.3.1 Cloud Service Categorization

We generated our cloud service categorization by using the proposed cloud services categorization algorithm under different similarity thresholds (see Chapter 3). We ran the experiments several times for each threshold to obtain the minimum number of clusters for all the cloud service sources. Figure 6.5 shows the number of clusters and the number of runs for each similarity threshold. It also shows the minimum number of clusters that have been obtained using our cloud services categorization algorithm. We can see that the minimum number of clusters increases as the similarity threshold increases. Furthermore, while the similarity threshold is between 0.5 and 0.6, the minimum number of clusters increases slightly. However, when the similarity threshold is greater than 0.6, the minimum number of clusters increases sharply due to two reasons. Firstly, most cloud service providers publish cloud services using their own descriptions, and they share only some basic cloud service concepts. Secondly, the variety of available services does exist in the cloud computing field.

After we finished generating the clusters, we conducted an additional experiment to examine the cluster types in more detail. The cluster types compose of three main cloud services, including Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS). In this experiment, we extracted cloud service ontology concepts from cloud service sources to determine the cloud service cluster

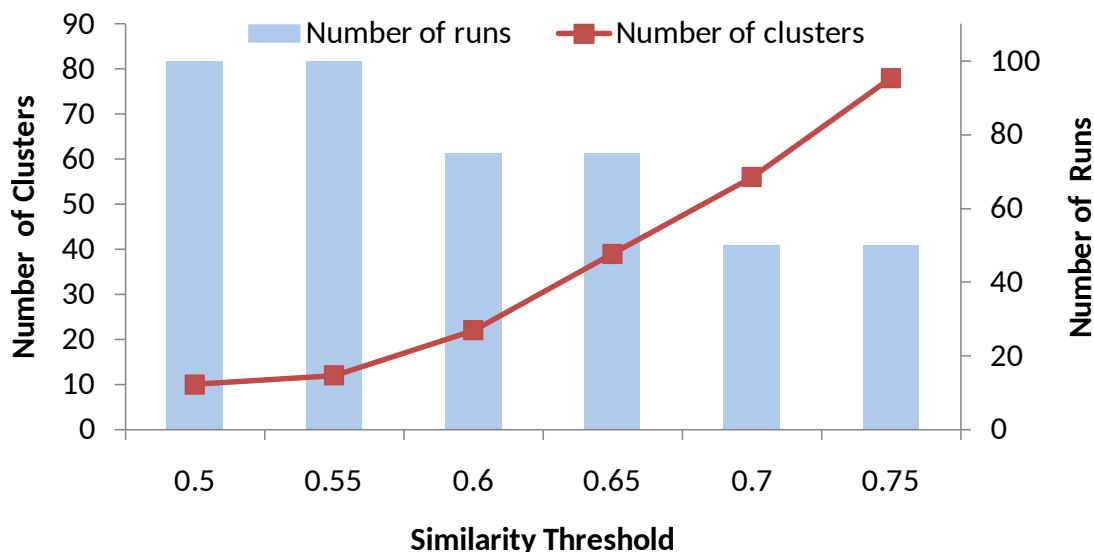


Fig. 6.5 Cloud Service Categorization

types. For example, “Host” is a concept of IaaS while a “Service” is a concept of SaaS. Therefore, we used the type of the majority concepts in the cluster as the cluster type. To be specific, for each cluster that has been generated, we used the variance to decide the cluster type. The type that has the highest variance was considered the cluster type. Figure 6.6 shows the majority cluster type is IaaS, which represents more than half of all clusters. The second popular cloud type is SaaS, which represents 30% of cloud service sources. The third cloud type is PaaS, which is rarely detected because PaaS is middleware between SaaS and IaaS, which leads to having many related concepts from them.

We conducted an experiment to test our cloud service categorization in identifying cloud services for proving the robustness during categorization of cloud service. In this experiment (see Fig. 3), we collected randomly 550 Webpage sources which are fake cloud services. Additionally, we used our test data 1.271 cloud service sources to run this experiment using precision, recall and f-measure with different similarity threshold. The precision represents the percentage rate of distinguishing between real and fake cloud services, whilst recall represents the percentage of cloud services identified and f-measure shows the mean between recall and precision.

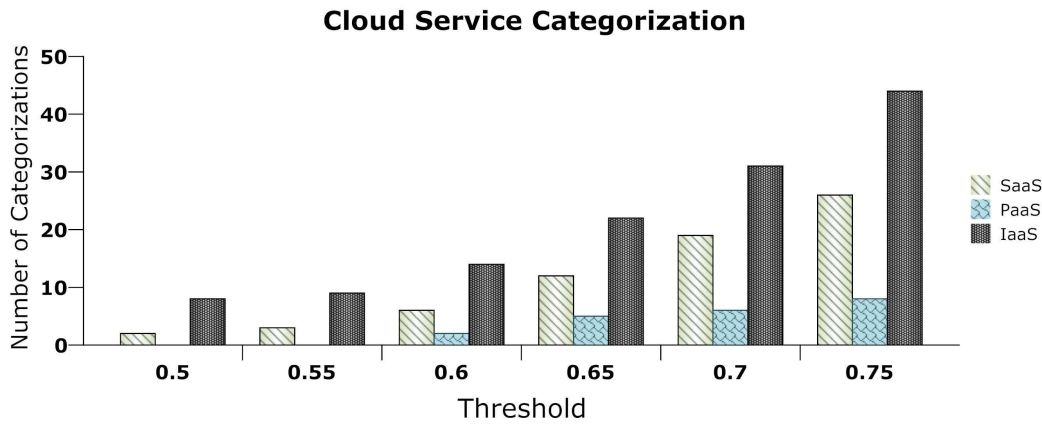


Fig. 6.6 Cloud Service Categorization Type

At the beginning, the experiment started with the similarity threshold 0.5 to compare these Webpages with the real cloud service Webpages under these categorization. However, if any of this Webpages pass the similarity threshold after comparing with the categorization it can be considered as cloud service Webpage and add it to cloud service categorization repository. Then, the numbers of the similarity threshold and clusters were increased dynamically. The experiment can be shown that the high similarity threshold provide high precision and less than average recall due to the huge number of clusters under this threshold. However, the minimum number of clusters is provide high precision and less than average recall which indicate to as minimum number of clusters can achieved as more accurate result can be obtained for cloud service categorization framework.

### 6.3.2 Robustness Cloud Service Categorization

We conducted an experiment to identify cloud services for proving the robustness of the search engine. In this experiment (see Fig. 6.8), we collected randomly 500 Webpages which are fake cloud services. Additionally, we used 1871 cloud service Webpages to run experiment using precision, recall and f-measure with fixed similarity threshold. At the beginning, the experiment started with 40 real cloud service

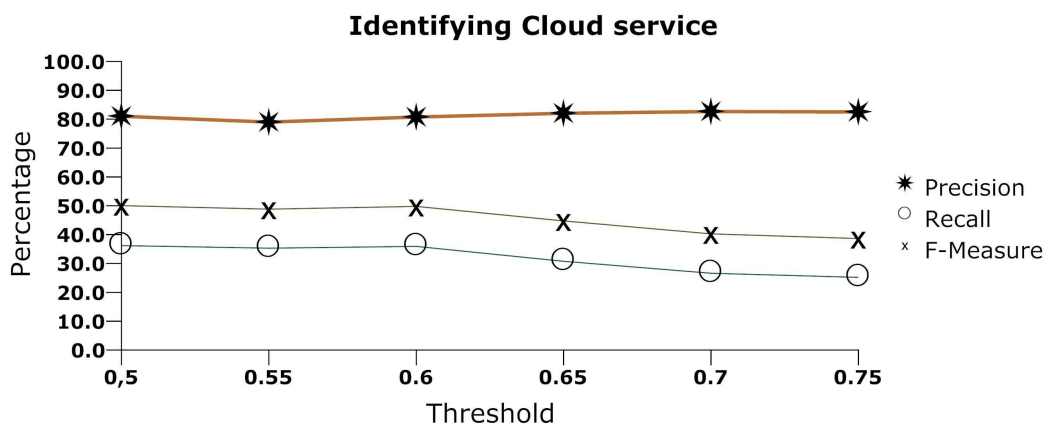


Fig. 6.7 Identifying Cloud service

Webpages and 10 fake cloud services Webpages. Then, the numbers were increased dynamically. The first experiment used the lower bound of the similarity threshold categorization to compare the results, depending on the threshold. In the experiment, it was concluded that the similarity threshold categorization provides high precision and average recall, which indicates that the lower bound threshold can efficiency find cloud services. However, it can provide less fake cloud services to the search engine.

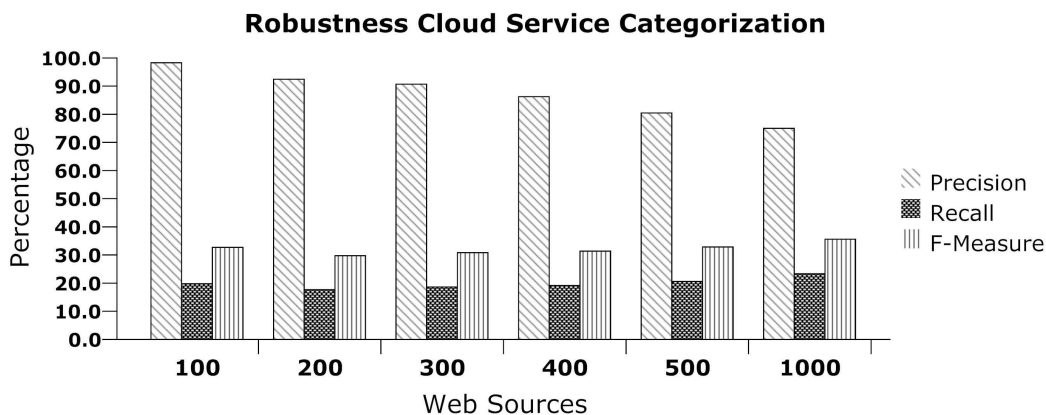


Fig. 6.8 Robustness Cloud Service Categorization

## 6.4 Cloud Service Identifier

In this section, we ran two experiments to identify cloud service based learning features and text features.

### 6.4.1 Cloud Service Identifier Learning Features

Firstly analyzed the effectiveness of different features to learn an identifying function. Given the small size of a cloud service source, our hypothesis is that cloud service features should help build better identifying functions. We started by testing both Jaccard and ontology features using various thresholds. We randomly sampled 1,552 cloud service test data, We ran Jaccard and ontology features 5 different thresholds (0.04, 0.08, 1.2, 1.6, 2.0). Figure 6.9 shows precision and recall under different similarity thresholds to test our approach. A lower identifying threshold gives high precision for both features, but it gives average recall. Furthermore, while we increase the threshold, the recall increases but the precision decreases sharply.

Determining the best threshold value which can be used to identify cloud services for Jaccard Similarity and Semantic features is difficult. Instead of that, we decide to exploit Support Vector Machine SVM [109], which has the capability to identify cloud service without using threshold, to increase the accuracy of identifying real cloud services on the Web environment.

We started by building a cloud service classification model using linear kernel SVM which provides high accuracy result for identifying cloud services. We randomly sampled 3,521 cloud service sources and 4,000 non-cloud service sources from the training dataset to build our cloud service identifier model. We ran a test data cross validation on this sample. Table 1 reports the results in terms of averaged accuracy which is a suitable measure for different feature combinations. From table 6.2, we can see that the accuracy of SVM+Jaccard similarity is very close to the one obtained from SVM+Semantic. Moreover, when we combine SVM+Jaccard similarity and

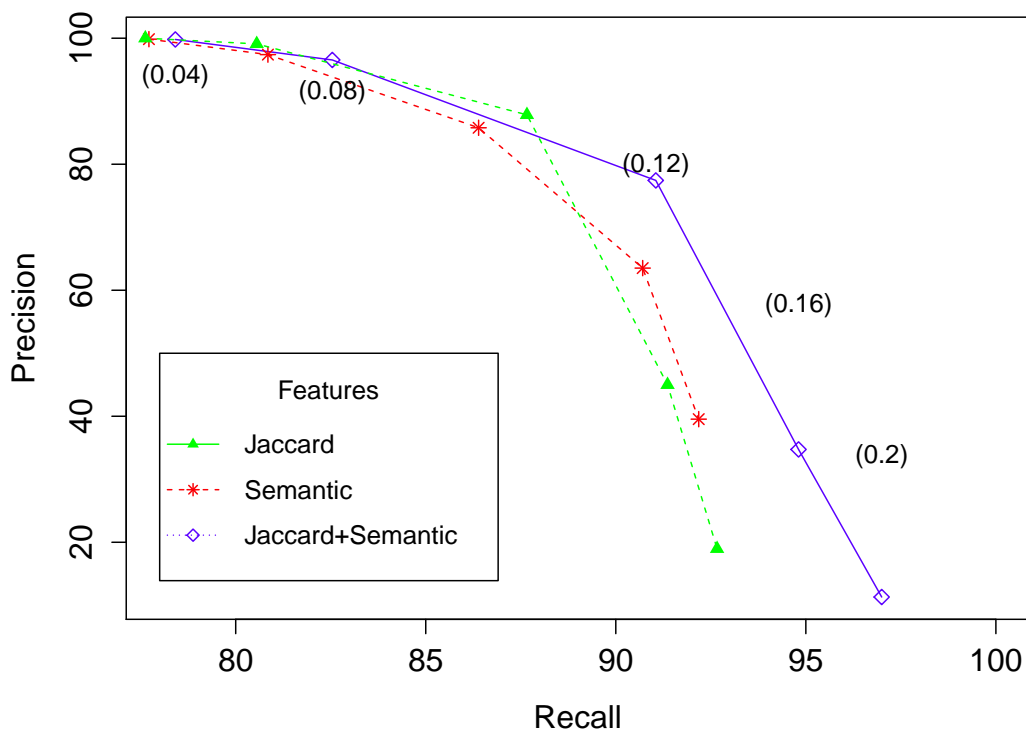


Fig. 6.9 Precision and Recall under Jaccard and Semantic features

Semantic together, it did not improve the accuracy result significantly. However, running all features with SVM increased the accuracy (both precision and recall) of the cloud service identifier model.

Table 6.2 Learning cloud service identification: SVM Accuracy, Precision and Recall

Features	Accuracy	Precision	Recall
SVM + Jaccard	82.40%	98.29%	82.41%
SVM + Semantic	81.59%	98.94%	81.35%
SVM + Jaccard + Semantic	82.17%	98.94%	81.86%
SVM + ALL	87.37%	99.29%	86.44%

## 6.4.2 Identifying Cloud Services Based Text Features

We ran another experiment to generate our cloud services identification based text features. We use the K-Nearest Neighbour algorithm to run this experiment. Firstly, we built cloud service corpus that contain 3,297 real cloud services and another non-cloud service corpus that includes 3,023 non-cloud services. Then, we achieved 987,457 terms from cloud service document matrix with high sparsity. However, after we removed the term sparsity the cloud service document matrix is still high dimension. Therefore, we used Latent Semantic Analysis (LSA) to reduce the dimension cloud document matrix and built our term features based on LSA concept. We used different number of K concepts and we ran our classification method. In the beginning, we ran the experiment with small number of K concepts and we gradually increase the K concepts. Fig 6.10 show the accuracy of term features using various k-nearest neighbor. In addition, it obtained high accuracy if we increase the k in k-nearest neighbor which achieved 86% with 6-nearest neighbor and increase the K concepts. In addition, we can see that the 500 concepts are obtained high accuracy with 6-nearest neighbor. However, we can notice that increasing the K of concepts can lead to decrease the identifying process because increasing of terms drives our identification to increase the noise data. We also conducted an experiment to identify cloud services for proving the precision and recall of our proposed cloud service identifier. In this experiment (see Fig 5 for the results), We used LSA terms that extracted from cloud service corpus then we run k-nearest neighbor model to determine the precision and recall. Moreover, the precision represents the percentage rate of distinguishing between cloud service or non-cloud service, whilst recall represents the percentage of cloud services identified. At the beginning, the experiment is started with 100 LSA terms then we increase the number respectively. We can see high relation between the number of LSA terms and precision and recall. we find that the increasing the number of LSA terms can reduce out model precision but it lead to increase the recall



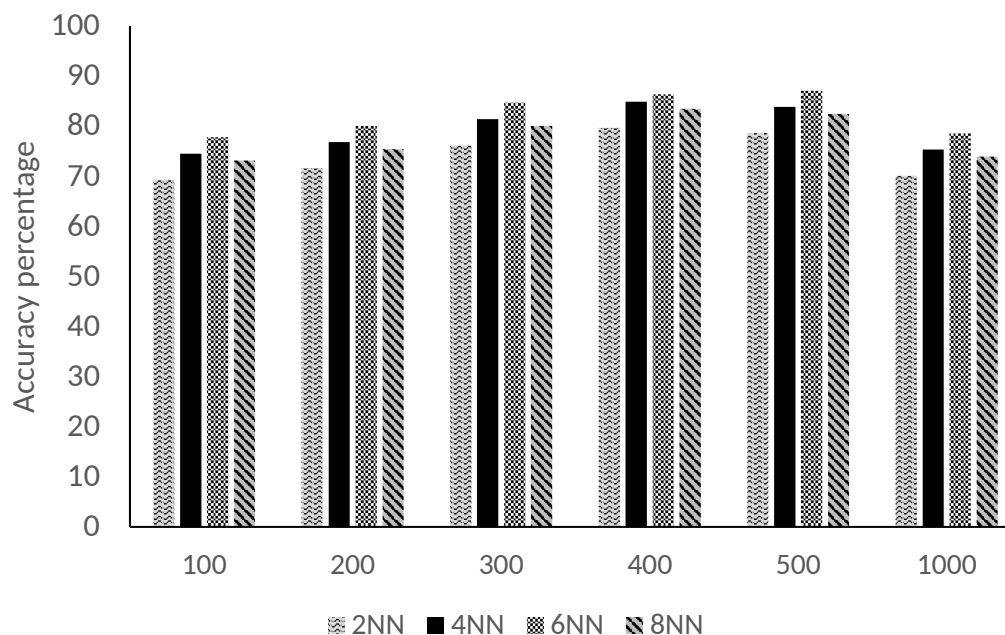


Fig. 6.10 Term features classification accuracy (%) using k-nearest neighbor classification, with LSA dimension reduction cloud service.

percent. Furthermore, we find that increasing the the K number can lead to increase the recall sharply.

### 6.4.3 Identification Robustness of Cloud Service Features

To optimize the cloud service identifier performance, we used our cloud services crawler. Our cloud services crawler explores the Web for cloud services using existing search engines' APIs (e.g., Google, Yahoo <sup>3</sup>, and Bing). We basically chose "cloud service" and "cloud service providers" as keywords to collect new data for each search engines' domain and search type. We used two domains which are Australia and USA and two search engine types which are normal search and advanced search. In the advanced search, we eliminated irrelevant search results such as Blogs, news, Wiki, and articles. The crawler collected possible cloud services' sources by continuously parsing search results from the indices returned by the search engines. The cloud

<sup>3</sup>Current Yahoo search engine uses Google's search engine for search results [https://en.wikipedia.org/wiki/Yahoo!\\_Search](https://en.wikipedia.org/wiki/Yahoo!_Search)

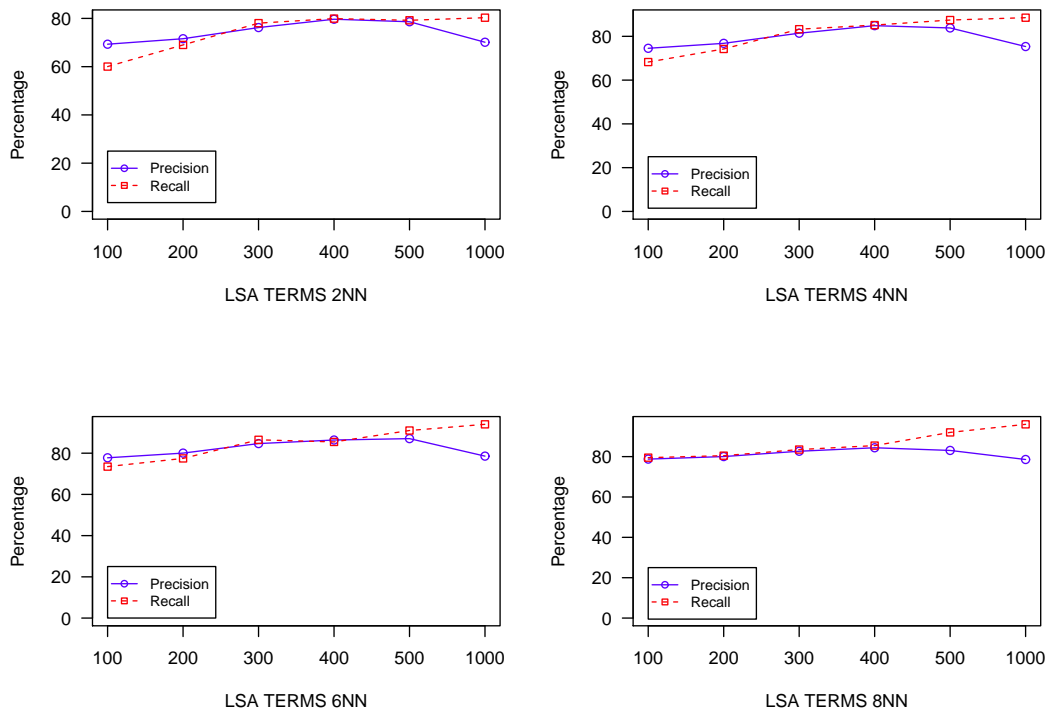


Fig. 6.11 Precision and recall using k-nearest neighbor classification, with LSA dimension reduction cloud service

service crawler managed to parse all cloud service possible URL links. Then, we removed all duplicated links to keep every link unique under its domain.

Table 6.3 shows the statistical analysis of the results that have been collected from search engines. From the table, we can see that the normal search (i.e., N-Search) provides more results than the advanced search (i.e., A-Search). Moreover, 84% of the difference of search results exists between Bing and Google under the USA domain, whilst there are 14% of the similarity of search results between Bing and Google under the Australia domain. In addition, we compared between the normal search and the advanced search for each search engine. In Bing USA, we found 33% similarity between the normal search and the advanced search results, while only 5% similarity under Bing Australia. In Google USA, we found 65% difference between the normal search and the advanced search results, while only 27% similarity under Google Australia. This statistical analysis can be indicated that using different search engines will return different results. Moreover, using several domains inside the same search engine will give various results. Finally, using advanced search can obtain different results from normal search engine interface.

After that, we manually labeled the search results to obtain the valid cloud services. We compared the search engine results with the results from our cloud service identifier model. From Table 6.4, we can see that in Bing USA, the accuracy is between 59% to 61% while our identifier can reach 90% accuracy in identifying cloud services. Also, our identifier can achieve 90% accuracy under Bing Australia which is better than Bing search engine results. As for Google normal search, the accuracy of our model can obtain 87% which is better than Google normal search (about 56% accuracy). Finally, under Google advanced search, we can achieve 57% accuracy by using our cloud service identifier model. However, from Table 6.4, we can conduct that searching about cloud search engine is tedious because there are many results appear have nothing to do with cloud service. In spite of using advanced search can help in filtering search results our cloud service identifier can optimal the advanced

search more than 10%. Therefore, using cloud service identifier is more effectiveness in cloud service discovery.

We also conducted an experiment to identify cloud services for proving the robustness of our proposed cloud service identifier. In this experiment (see Fig. 6.12-6.15 for the results), we randomly collected 1,050 Web sources for each search engine type (normal search and advanced search). Additionally, we labeled each Web source manually as cloud service or non-cloud service. Moreover, the precision represents the percentage rate of distinguishing between cloud service or non-cloud service, whilst recall represents the percentage of cloud services identified. At the beginning, the experiment started with 50 Web sources. During this the experiment, we increased dynamically the Web sources. Fig 6.12 which is Bing Normal search shows high precision (almost 99%) for each time we increase the Web sources in the experiment. From Fig 6.12, we can see the high average recall is shown and it decreases slightly as we increase the number of Web sources.

Similarly, Fig. 6.13 which is Bing Advanced shows high precision for each time we increase the Web sources in the experiment. From the figure, we can see that the average recall is achieved and it decreases as we increased the Web sources. This is because that the search is an advanced search, which is more difficult to recall. Fig. 6.14 which is Google Normal Search shows high precision reaching to 99% for each time we increase the Web sources in the experiment. From Fig. 6.14, we can see that the high recalls are decreased sharply as we increased the Web sources with our identifier. Again, for advanced search, Fig 6.15 which is Google Advanced Search show high precision at almost 99% for each time we increase the Web sources in the experiment. From the figure we can see that the average recall is still stable as we increased the Web sources.

Table 6.3 Comparison results between Google and Bing

<b>Search Engine</b>	<b>N-Search</b>	<b>A-Search</b>	<b>Similarity between Bing and Google</b>	<b>Similarity between N-Search and A-Search</b>
Bing + USA	727	539	15.23%	10.24%
Bing + Australia	700	296	14.54%	5.21%
Google + USA	352	351	15.23%	10.24%
Google + Australia	351	323	14.54%	5.21%

Table 6.4 Comparison results between search engine and our identifier

<b>Search Engine</b>	<b>Accuracy</b>	<b>Accuracy with A-Search</b>	<b>Our Accuracy</b>	<b>Our Accuracy with A-Search</b>
Bing + USA	61.14%	59.58%	90.11%	86.68%
Bing + Australia	58.96%	51.58%	90.62%	85.21%
Google + USA	57.11%	45.04%	87.86%	56.54%
Google + Australia	45.64%	43.04%	86.04%	58.63%

We also conducted an experiment to identify cloud services for proving the robustness of our proposed cloud service identifier. In this experiment (see Fig. 6.12-6.15 for the results), we randomly collected 1,050 Web sources for each search engine type (normal search and advanced search). Additionally, we labeled each Web source manually as cloud service or non-cloud service. Moreover, the precision represents the percentage rate of distinguishing between cloud service or non-cloud service, whilst recall represents the percentage of cloud services identified. At the beginning, the experiment started with 50 Web sources. During this the experiment, we increased dynamically the Web sources. Fig 6.12 which is Bing Normal search shows high precision (almost 99%) for each time we increase the Web sources in the experiment. From Fig 6.12, we can see the high average recall is shown and it decreases slightly as we increase the number of Web sources.

Similarly, Fig. 6.13 which is Bing Advanced shows high precision for each time we increase the Web sources in the experiment. From the figure, we can see that the average recall is achieved and it decreases as we increased the Web sources. This is because that the search is an advanced search, which is more difficult to recall. Fig. 6.14 which is Google Normal Search shows high precision reaching to 99% for each time we increase the Web sources in the experiment. From Fig. 6.14, we can see that the high recalls are decreased sharply as we increased the Web sources with our identifier. Again, for advanced search, Fig 6.15 which is Google Advanced Search show high precision at almost 99% for each time we increase the Web sources in the experiment. From the figure we can see that the average recall is still stable as we increased the Web sources.

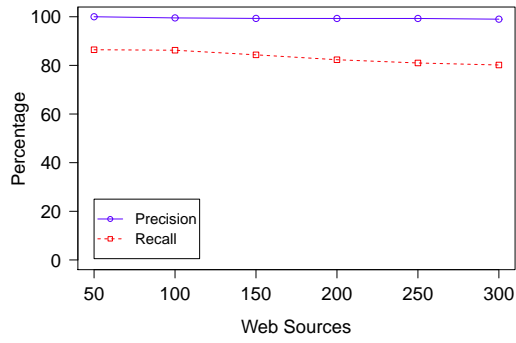


Fig. 6.12 Bing Normal Search

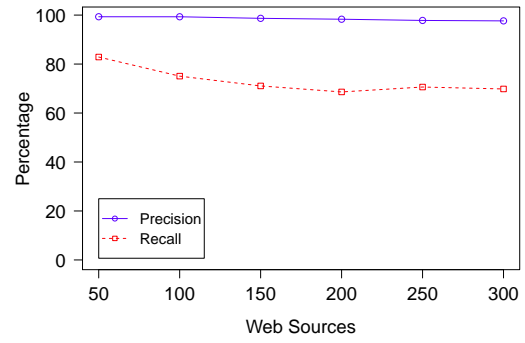


Fig. 6.13 Bing Advanced Search

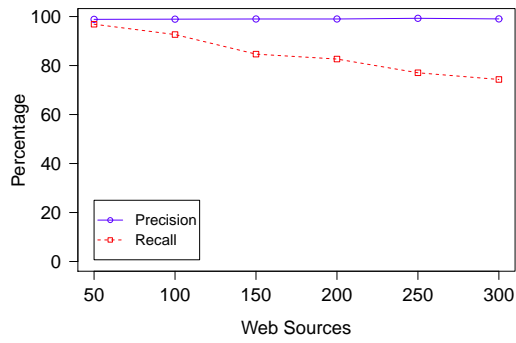


Fig. 6.14 Google Normal Search

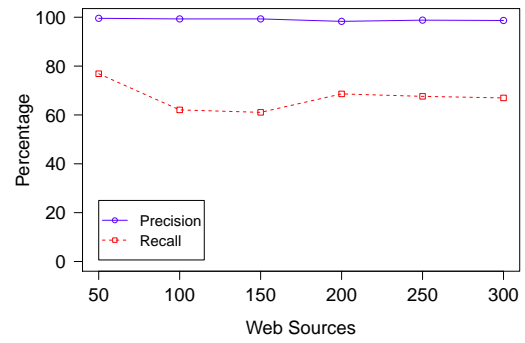


Fig. 6.15 Google Advanced Search

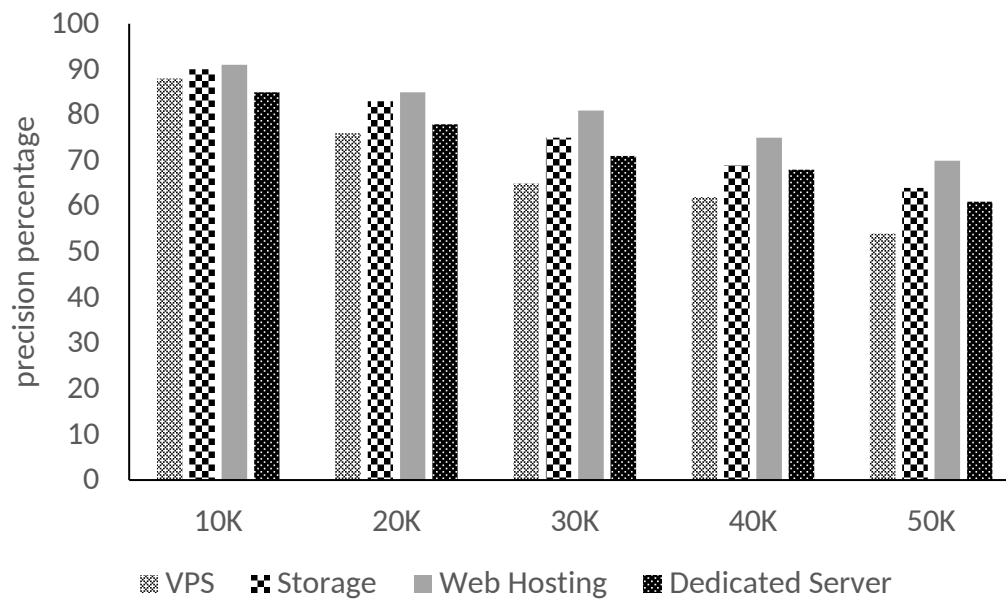


Fig. 6.16 Cloud Service Profile

#### 6.4.4 Cloud Service Profile

In order to determine the service, we ran our cloud profile algorithm to detect and track the service. This algorithm can search for a specific service over a bunch of cloud service providers. To ensure the accuracy of the results, we have labeled the cloud services sources manually before we ran our algorithm. Then, we compared our algorithm with manual processing. Fig. 6.16 shows the precision result of cloud profile algorithm. We can see that if we increase the number of cloud service sources we increase the difficulty of detecting the service features because the precision is decrease. However, we find that service which are popular in real environments such as cloud VPS and Web Hosting can be found precisely by our algorithm. Furthermore, we observed that some cloud service providers show their service features under alternative hyper-links such as plan, features and price.

Fig 6.17 shows that where mostly we can find that the cloud service features inside the cloud services. From Fig 6.17 we found the 36% of cloud service describe their services under the service name hyper-links while 21% describe under the features



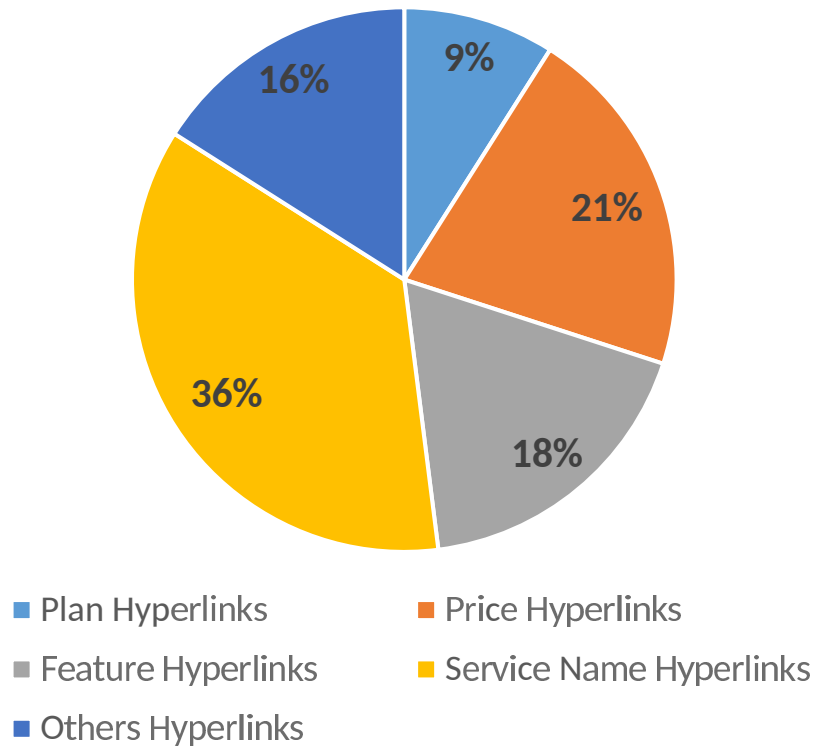


Fig. 6.17 Cloud Service Profile Features

hyper-links. More over, 18% of cloud service providers described their services under price hyper-link while only 9% of cloud services used plan hyper-links to describe their services. However, 18% of cloud service providers describe their services under specific name for the service which leads to difficulty in finding the service and its features.

## 6.5 Summary

In this chapter, we conducted extensive experimental and performance studies of the proposed techniques using a collection of real-world on cloud services. Firstly, we describe the cloud service dataset that is used to conduct the experiments. Secondly, we conduct a set of statistical analysis and present based on the Cloud Service ontology. These statistical results offer an overall view on the our cloud service ontology.

Thirdly, we conducted extensive experiments to validate our cloud service categorization approach. The results demonstrate the applicability of our approach and its capability of effectively categorizing cloud services on the Internet. Finally, we validate and study our identifier model from various aspects including accuracy, precision, and recall.

# Chapter 7

## Conclusion

In this chapter, we summarize the contributions of this dissertation and discuss some future research directions for on cloud services discovery.

### 7.1 Summary

Over the past few years, cloud computing has been more and more attractive as a new computing paradigm due to high flexibility for provisioning on-demand computing resources that are used as services through the Internet. The issues around cloud service discovery have considered by many researchers in in the recent years. However, in cloud computing, with the highly dynamic, distributed, the lack of standardized description languages, diverse services offered at different levels and non-transparent nature of cloud services, this research area has gained a significant attention. Robust cloud service discovery approaches will be essential in promoting and growing cloud service consumers and providers and will significantly contribute to the adoption and growth of cloud computing. In this dissertation, we have proposed an automated cloud service discovery of cloud services. We also conduct extensive experiments to validate our proposed approach. The results demonstrate the applicability of our approach and its capability of effectively identifying and categorizing cloud services on

the Internet. Firstly, we develop a novel approach to build the cloud service ontology. Cloud service ontology initially is built based on the National Institute of Standards and Technology (NIST) cloud computing standard. Then, we add new concepts to the cloud service ontology by automatically analyzing real cloud services based on cloud service ontology algorithm. We also propose cloud service categorization that use Term Frequency to weigh cloud service ontology concepts and calculate cosine similarity to measure the similarity between cloud services. The cloud service categorization algorithms is able to categorize cloud services to clusters for effective categorization of cloud services. In addition, we use Machine Learning techniques to identify cloud service in real environment. Our cloud service identifier is built by utilizing cloud service features extracted from real cloud service providers. We determine several features such as similarity function, semantic ontology, cloud service description and cloud services components, to be used effectively in identifying cloud service in the Web. Also, we build a unified model to expose the cloud service's features to a cloud service search user to ease the process of searching and comparison among a large amount of cloud services by building cloud service's profile. Furthermore, we particularly develop a cloud service discovery Engine that has capability to crawl the Web automatically and collect cloud services. The collected datasets include meta-data of nearly 7,500 real-world cloud services providers and nearly 15,000 services (2.45GB). The experimental results show that our approach i) is able to effectively build automatic cloud service ontology, ii) is more robust in identifying cloud service in real environment and iii) is more scalable in provide more details about cloud services. We can summarize our main research contributions in the following:

### **7.1.1 Cloud Service Ontology**

We built a comprehensive ontology for reasoning during cloud service discovery, which was semi-automatically based on mining new concepts, after analysing real-world cloud services. Our approach involved two steps: creating a base ontology by following the US National Institute of Standards and Technology's (NIST) cloud computing standards, which we also called a cloud service ontology roadmap because there are limited concepts in this ontology, and developing an algorithm to analyse these cloud services and identify new cloud service concepts that could generate our cloud service ontology. Our ontology automatically enhances crawling procedures and both the identification and categorisation models. [3].

### **7.1.2 Cloud Service Categorization Model**

We built a cloud services categorization model that was based on the cosine similarity approach. Our model contains a processor that automatically categorizes given cloud service providers based on the service model (IaaS, PaaS and SaaS). This model relies on the cluster method to build the categorization, which can be updated automatically after categorizing a new cloud service provider to increase categorization knowledge. Furthermore, the model uses term frequency to weigh cloud service ontology concepts and calculate cosine similarity to measure the similarities between cloud services. We then designed and developed cloud service categorization algorithms that categorize cloud services into clusters [4].

### **7.1.3 Cloud Service Identifier Model**

The cloud services identifier model identifies features to determine whether a given source is a cloud service. This model relies on the classification method to build the identifier, which can be updated automatically after identifying a new cloud service provider to enhance identifier knowledge. To identify cloud services in a real envi-

ronment, we focused on learning features that can be shared by large cloud service providers on the web to predict whether the given web source provides cloud services. We discovered a wide range of features between cloud service providers inside their sources, such as similarity of function, cloud service semantic concepts and cloud service components, and used those as identification features to train learning models. Learning models are fed into the classification algorithm to detect cloud service providers. In addition, our identifier model can build a cloud service profile, which will include details about the service and its attributes (e.g. price, memory, CPU, storage, OS etc).

#### **7.1.4 Dataset of Cloud Service Collection**

From what we have seen, we consider that in order to gain invaluable knowledge of the technologies used in the field of cloud services, there is an urgent need to identify the present state of cloud services. Therefore, we collect, identify and analyze all cloud services that are available on the web. We have used the cloud service discovery engine to undertake this task twice: in 2013 and 2017. Our automatic discovery managed to fetch 619,474 possible links and collected 35,601 possible Web sources for cloud services in 2013; by contrast, in 2017, the cloud service discovery engine managed to fetch 844,696 possible links and collect 55,474 possible Web sources for cloud services. From the data collected, the analysis was carried out, and the results achieved provided a comprehensive overview of the status of cloud services. Also, we prepared a large dataset of real-world cloud services that available on the Web. Also, we released the collected dataset to the research community. These dataset includes 7,461 cloud services and nearly 15,000 services attributes(2.45 GB) [84].

### **7.1.5 Implementation and Performance Study**

We implemented our proposed frameworks for discovering cloud services via cloud service crawling. We developed a unified platform for automatic cloud service discovery to identify and categorize cloud services. We conducted extensive experiments and performance studies of the proposed approaches using real-world cloud services available on the World Wide Web in order to validate the feasibility and benefits of our approaches. The results demonstrated the applicability of our approach and its ability to effectively identify and categorize cloud services on the Internet[5, 4, 3].

## **7.2 Future Work**

Although cloud services discovery issues attracted many researchers, several research issues still need to be addressed. In particular, we identify the following directions for future research in cloud service discovery.

### **7.2.1 Cloud Service Standards**

Based on a comparative study that we conduct in chapter 2, we conclude that building a unified standards is the most appropriate solution which will support cloud service discovery [31, 71]. Cloud service standards need to be able to considers two aspects of cloud service (functional, non functional). In addition, the standard needs to be generic which should be able to describe all cloud services (SaaS, PaaS, and IaaS) and cover technical, operational, business, and semantic aspects.

### **7.2.2 Cloud Service Recommendation**

Quality-of-Service (QoS) is widely employed to represent the non-functional characteristics of cloud service and has been considered as the key factor in service selection [16]. We argue that there is a need to recommend cloud services currently available on

the Web, which will support customer to select appropriate cloud services to based on his demands. The recommendation system should be able to support in finding nearest cloud service with high quality attributes to cloud service customers. Collaborative filtering [99] and content based recommendation [57] techniques can be used to build reliable cloud service recommendation system [64].

### **7.2.3 Cloud Service Personalization**

Cloud services can offer many solutions on different aspects of security. The decision is which one to choose: VPN (Virtual Private Network), SSL (Secure Socket Layer) for IaaS (Infrastructure as a Service) or PaaS (Platform as a Service) and SaaS (Software as a Service), for example, like in IBM, SSH (Secure Shell, i.e. in Amazon) password-related protection.

It is irrelevant whether it is a service provider or requester that is the consumer of cloud services (note: the technology is not always suitable for every consumer of cloud services). Hence, we believe that techniques should be adaptable, to assist cloud service customers in customizing the relevant technologies to their specific requirements.

Likewise, although the volume of technologies offered by cloud services is vast, meaning consumers using cloud services may be confronted with issues regarding configuration (i.e., the quantity and type of virtual machines, contract length and policies on control access). Accordingly, there is a greater need for smart-technologies which will enable the cloud platform to study and learn cloud service consumer patterns.



# References

- [1] Yasmine M Afify, Ibrahim F Moawad, NL Badr, and M Fahmy Tolba. A semantic-based software-as-a-service (saas) discovery and selection system. In *Computer Engineering & Systems (ICCES), 2013 8th International Conference on*, pages 57–63. IEEE, 2013.
- [2] Eyhab Al-Masri and Qusay H Mahmoud. Investigating web services on the world wide web. In *Proceedings of the 17th international conference on World Wide Web*, pages 795–804. ACM, 2008.
- [3] Abdullah Alfazi, Talal H Noor, Quan Z Sheng, and Yong Xu. Towards ontology-enhanced cloud services discovery. In *Advanced Data Mining and Applications*, pages 616–629. Springer International Publishing, 2014.
- [4] Abdullah Alfazi, Quan Z Sheng, Yongrui Qin, and Talal H Noor. Ontology-based automatic cloud service categorization for enhancing cloud service discovery. In *Enterprise Distributed Object Computing Conference (EDOC), 2015 IEEE 19th International*, pages 151–158. IEEE, 2015.
- [5] Abdullah Alfazi, Quan Z Sheng, Wei Emma Zhang, Lina Yao, and Talal H Noor. Identification as a service: Large-scale cloud service discovery over the world wide web. In *Big Data (BigData Congress), 2016 IEEE International Congress on*, pages 485–492. IEEE, 2016.
- [6] James Allan. *Topic detection and tracking: event-based information organization*, volume 12. Springer Science & Business Media, 2012.
- [7] Flora Amato and Francesco Moscato. A modeling profile for availability analysis of composite cloud services. In *P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), 2014 Ninth International Conference on*, pages 406–413. IEEE, 2014.
- [8] Farnoush Banaei-Kashani, Ching-Chien Chen, and Cyrus Shahabi. Wspds: Web services peer-to-peer discovery service. In *Proceedings of the International Conference on Internet Computing*, pages 733–743, 2004.
- [9] Fabricio Benevenuto, Gabriel Magno, Tiago Rodrigues, and Virgilio Almeida. Detecting spammers on twitter. In *Collaboration, electronic messaging, anti-abuse and spam conference (CEAS)*, volume 6, page 12, 2010.
- [10] Enrico Blanzieri and Anton Bryl. A Survey of Learning-Based Techniques of Email Spam Filtering. *Artificial Intelligence Review*, 29(1):63–92, 2008.

- [11] Matko Bošnjak, Eduardo Oliveira, José Martins, Eduarda Mendes Rodrigues, and Luís Sarmiento. Twitterecho: a distributed focused crawler to support open research with twitter data. In *Proceedings of the 21st International Conference on World Wide Web*, pages 1233–1240. ACM, 2012.
- [12] Athman Bouguettaya, Munindar Singh, Michael Huhns, Quan Z. Sheng, Hai Dong, Qi Yu, Azadeh Ghari Neiat, Sajib Mistry, Boualem Benatallah, Brahim Medjahed, Mourad Ouzzani, Fabio Casati, Xumin Liu, Hongbing Wang, Dimitrios Georgakopoulos, Liang Chen, Surya Nepal, Zaki Malik, Abdelkarim Erradi, Yan Wang, Brian Blake, Schahram Dustdar, Frank Leymann, and Michael Papazoglou. A service computing manifesto: The next 10 years. *Commun. ACM*, 60(4):64–72, March 2017. ISSN 0001-0782.
- [13] Ivona Brandic, Schahram Dustdar, Tobias Anstett, David Schumm, Frank Leymann, and Ralf Konrad. Compliant cloud computing (c3): Architecture and language support for user-driven compliance management in clouds. In *Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on*, pages 244–251. IEEE, 2010.
- [14] Hong Cai, Ke Zhang, MiaoMiao Wang, JiaLin Li, Lei Sun, and XinSheng Mao. Customer centric cloud service model and a case study on commerce as a service. In *Cloud Computing, 2009. CLOUD'09. IEEE International Conference on*, pages 57–64. IEEE, 2009.
- [15] Fei Chen, Xiaoli Bai, and Bingbing Liu. Efficient service discovery for cloud computing environments. *Advanced Research on Computer Science and Information Engineering*, pages 443–448, 2011.
- [16] Xi Chen, Zibin Zheng, Qi Yu, and Michael R Lyu. Web service recommendation via exploiting location and qos information. *IEEE Transactions on Parallel and Distributed Systems*, 25(7):1913–1924, 2014.
- [17] Trieu C Chieu, Ajay Mohindra, Alexei A Karve, and Alla Segal. Dynamic scaling of web applications in a virtualized cloud computing environment. In *E-Business Engineering, 2009. ICEBE'09. IEEE International Conference on*, pages 281–286. IEEE, 2009.
- [18] Kassidy P Clark, ME Warnier, Frances MT Brazier, and Thomas B Quillinan. Secure monitoring of service level agreements. In *Availability, Reliability, and Security, 2010. ARES'10 International Conference on*, pages 454–461. IEEE, 2010.
- [19] Alejandro Cornejo, Saira Viqar, and Jennifer L Welch. Reliable neighbor discovery for mobile ad hoc networks. *Ad Hoc Networks*, 12:259–277, 2014.
- [20] Amir Vahid Dastjerdi, Sayed Gholam Hassan Tabatabaei, and Rajkumar Buyya. An Effective Architecture for Automated Appliance Management System Applying Ontology-based Cloud Discovery. In *Proc. of 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing (CCGrid 2010)*, pages 104–112, 2010.
- [21] Yu Deng, Michael R Head, Andrzej Kochut, Jonathan Munson, Anca Sailer, and Hidayatullah Shaikh. Introducing semantics to cloud services catalogs.

- In *Services Computing (SCC), 2011 IEEE International Conference on*, pages 24–31. IEEE, 2011.
- [22] Beniamino Di Martino, Giuseppina Cretella, Antonio Esposito, and Raffaele Giulio Sperandeo. Semantic representation of cloud services: a case study for microsoft windows azure. In *Intelligent Networking and Collaborative Systems (INCoS), 2014 International Conference on*, pages 647–652. IEEE, 2014.
- [23] Giuseppe Di Modica, Giuseppe Petralia, and Orazio Tomarchio. A business ontology to enable semantic matchmaking in open cloud markets. In *Semantics, Knowledge and Grids (SKG), 2012 Eighth International Conference on*, pages 96–103. IEEE, 2012.
- [24] W Keith Edwards. Discovery systems in ubiquitous computing. *IEEE Pervasive Computing*, 5(2):70–77, 2006.
- [25] Daren Fang, Xiaodong Liu, Imed Romdhani, Pooyan Jamshidi, and Claus Pahl. An agility-oriented and fuzziness-embedded semantic model for collaborative cloud service search, retrieval and recommendation. *Future Generation Computer Systems*, 56:11–26, 2016.
- [26] Ian Foster, Yong Zhao, Ioan Raicu, and Shiyong Lu. Cloud computing and grid computing 360-degree compared. In *Grid Computing Environments Workshop, 2008. GCE'08*, pages 1–10. Ieee, 2008.
- [27] Adrian Friday, Nigel Davies, Nat Wallbank, Elaine Catterall, and Stephen Pink. Supporting service discovery, querying and interaction in ubiquitous computing environments. *Wireless Networks*, 10(6):631–641, 2004.
- [28] Francisco García-Sánchez, Rafael Valencia-García, Rodrigo Martínez-Béjar, and Jesualdo T. Fernández-Breis. An ontology, intelligent agent-based framework for the provision of semantic web services. *Expert Systems with Applications*, 36(2, Part 2):3167 – 3187, 2009. ISSN 0957-4174.
- [29] Saurabh Kumar Garg, Steve Versteeg, and Rajkumar Buyya. A framework for ranking of cloud computing services. *Future Gener. Comput. Syst.*, 29(4): 1012–1023, June 2013. ISSN 0167-739X. doi: 10.1016/j.future.2012.06.006. URL <http://dx.doi.org/10.1016/j.future.2012.06.006>.
- [30] Souad Ghazouani and Yahya Slimani. A survey on cloud service description. *Journal of Network and Computer Applications*, 2017.
- [31] Souad Ghazouani and Yahya Slimani. Towards a standardized cloud service description based on usdl. *Journal of Systems and Software*, 132:1–20, 2017.
- [32] Glauco Goncalves, Patricia Endo, Marcelo Santos, Djamel Sadok, Judith Kellner, Bob Melander, and Jan-Erik Mangs. Cloudml: An integrated language for resource, service and request description for d-clouds. In *Cloud Computing Technology and Science (CloudCom), 2011 IEEE Third International Conference on*, pages 399–406. IEEE, 2011.

- [33] Andrzej Goscinski and Michael Brock. Toward dynamic and attribute based publication, discovery and selection for cloud computing. *Future generation computer systems*, 26(7):947–970, 2010.
- [34] Nicola Guarino, Daniel Oberle, and Steffen Staab. What is an ontology? In *Handbook on ontologies*, pages 1–17. Springer Berlin Heidelberg, 2009.
- [35] Stefan Gudenkauf, Mirco Josefiok, Andre Goring, and Oliver Norkus. A reference architecture for cloud service offers. In *Enterprise Distributed Object Computing Conference (EDOC), 2013 17th IEEE International*, pages 227–236. IEEE, 2013.
- [36] Dominique Guinard, Vlad Trifa, Stamatis Karnouskos, Patrik Spiess, and Dominic Savio. Interacting with the soa-based internet of things: Discovery, query, selection, and on-demand provisioning of web services. *IEEE transactions on Services Computing*, 3(3):223–235, 2010.
- [37] Hatem Hamad, Motaz Saad, and Ramzi Abed. Performance evaluation of restful web services for mobile devices. *International Arab Journal of e-Technology*, 1(3), 2010.
- [38] Mohammad Hamdaqa, Tassos Livogiannis, and Ladan Tahvildari. A reference model for developing cloud applications. In *CLOSER*, pages 98–103, 2011.
- [39] Taekgyeong Han and Kwang Mong Sim. An ontology-enhanced cloud service discovery system. In *Proceedings of the International MultiConference of Engineers and Computer Scientists*, volume 1, pages 17–19, 2010.
- [40] Michael Hausenblas, Robert Grossman, Andreas Harth, and Philippe Cudré-Mauroux. Large-scale linked data processing-cloud computing to the rescue?. *CLOSER*, 20(2):246–251, 2012.
- [41] Qiang He, Jun Yan, Yun Yang, Ryszard Kowalczyk, and Hai Jin. A decentralized service discovery approach on peer-to-peer networks. *IEEE Transactions on Services Computing*, 6(1):64–75, 2013.
- [42] Patrick Hoberg, Jan Wollersheim, and Helmut Krcmar. Service descriptions for cloud services-the customer’s perspective. 2012.
- [43] CN Höfer and G Karagiannis. Cloud computing services: taxonomy and comparison. *Journal of Internet Services and Applications*, 2(2):81–94, 2011.
- [44] Karuna P Joshi, Yelena Yesha, and Tim Finin. Automating cloud services life cycle through semantic technologies. *IEEE Transactions on Services Computing*, 7(1):109–122, 2014.
- [45] Frederic Junker, Jürgen Vogel, and Katarina Stanoevska. Aggregating price models for composite services in cloud service marketplaces. In *Parallel and Distributed Processing with Applications (ISPA), 2012 IEEE 10th International Symposium on*, pages 479–486. IEEE, 2012.
- [46] Jaeyong Kang and Kwang Mong Sim. Cloudle: An Ontology-Enhanced Cloud Service Search Engine. In *Proc. of the 2010 International Conference on Web Information Systems Engineering (WISE 2010)*, pages 416–427, 2010. ISBN 978-3-642-24395-0.

- [47] Jaeyong Kang and Kwang Mong Sim. Cloudle: an ontology-enhanced cloud service search engine. In *International Conference on Web Information Systems Engineering*, pages 416–427. Springer, 2010.
- [48] Jaeyong Kang and Kwang Mong Sim. Ontology and search engine for cloud computing system. In *System Science and Engineering (ICSSE), 2011 International Conference on*, pages 276–281. IEEE, 2011.
- [49] Jaeyong Kang and Kwang Mong Sim. Towards agents and ontology for cloud service discovery. In *Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2011 International Conference on*, pages 483–490. IEEE, 2011.
- [50] Jaeyong Kang, Cloud Computing, et al. Cloudle: An agent-based cloud search engine that consults a cloud ontology. In *Proceedings of the International Conference on Cloud Computing and Virtualization*. Citeseer, 2010.
- [51] Matthias Klusch, Benedikt Fries, and Katia Sycara. Automated semantic web service discovery with owls-mx. In *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, pages 915–922. ACM, 2006.
- [52] Yue Kun, WANG Xiao-Ling, and ZHOU Ao-Ying. Underlying techniques for web services: A survey. In *Journal of software*. Citeseer, 2004.
- [53] Peep Kungas and Mihhail Matskin. Semantic web service composition through a p2p-based multi-agent environment. In *International Workshop on Agents and P2P Computing*, pages 106–119. Springer, 2005.
- [54] Bu Sung Lee, Shixing Yan, Ding Ma, and Guopeng Zhao. Aggregating iaas service. In *SRII Global Conference (SRII), 2011 Annual*, pages 335–338. IEEE, 2011.
- [55] Yan Li, Yao Liu, Liangjie Zhang, Ge Li, Bing Xie, and Jiasu Sun. An exploratory study of web services on the internet. In *Web Services, 2007. ICWS 2007. IEEE International Conference on*, pages 380–387. IEEE, 2007.
- [56] Dongxi Liu and John Zic. Cloud#: A specification language for modeling cloud. In *Cloud Computing (CLOUD), 2011 IEEE International Conference on*, pages 533–540. IEEE, 2011.
- [57] Pasquale Lops, Marco De Gemmis, and Giovanni Semeraro. Content-based recommender systems: State of the art and trends. In *Recommender systems handbook*, pages 73–105. Springer, 2011.
- [58] Yong Beom Ma, Sung Ho Jang, and Jong Sik Lee. Ontology-based Resource Management for Cloud Computing. In *Intelligent Information and Database Systems*, pages 343–352. Springer, 2011.
- [59] PC Magazine and API In. Pcmag.com encyclopedia, 2017.
- [60] D Marshall. Cloud storage provider nirvanix is closing its doors, 2013.

- [61] Toni Mastelic, Ivona Brandic, and Andrés García García. Towards uniform management of cloud services by applying model-driven development. In *Computer Software and Applications Conference (COMPSAC), 2014 IEEE 38th Annual*, pages 129–138. IEEE, 2014.
- [62] Peter Mell and Timothy Grance. The NIST Definition of Cloud Computing, Sep 2011. Accessed: 05/06/2012, Available at: [http://csrc.nist.gov/publications/drafts/800-145/Draft-SP-800-145\\_cloud-definition.pdf](http://csrc.nist.gov/publications/drafts/800-145/Draft-SP-800-145_cloud-definition.pdf).
- [63] Peter Mell and Timothy Grance. The NIST Definition of Cloud Computing (draft). *NIST special publication*, 800(145):7, 2011.
- [64] Prem Melville, Raymond J Mooney, and Ramadass Nagarajan. Content-boosted collaborative filtering for improved recommendations. In *Aaai/iaai*, pages 187–192, 2002.
- [65] Michael Menzel and Rajiv Ranjan. Cloudgenius: decision support for web server cloud migration. In *Proceedings of the 21st international conference on World Wide Web*, pages 979–988. ACM, 2012.
- [66] E. Meshkova, J. Riihijärvi, M. Petrova, and P. Mähönen. A Survey on Resource Discovery Mechanisms, Peer-to-Peer and Service Discovery Frameworks. *Computer Networks*, 52(11):2097–2128, 2008.
- [67] Khaled M Metwally, Abdallah Jarray, and Ahmed Karmouch. Two-phase ontology-based resource allocation approach for iaas cloud service. In *Consumer Communications and Networking Conference (CCNC), 2015 12th Annual IEEE*, pages 790–795. IEEE, 2015.
- [68] A.N. Mian, R. Baldoni, and R. Beraldi. A Survey of Service Discovery Protocols in Mobile Ad Hoc Networks. *IEEE Pervasive Computing*, 8(1):66–74, 2009.
- [69] Ryszard S Michalski, Jaime G Carbonell, and Tom M Mitchell. *Machine learning: An artificial intelligence approach*. Springer Science & Business Media, 2013.
- [70] Gilad Mishne, David Carmel, and Ronny Lempel. Blocking Blog Spam with Language Model Disagreement. In *Proc. of the First International Workshop on Adversarial Information Retrieval on the Web (AIRWeb 2005)*, volume 5, pages 1–6, 2005.
- [71] Rafael Moreno-Vozmediano, Rubén S Montero, and Ignacio M Llorente. Key challenges in cloud computing: Enabling the future internet of services. *IEEE Internet Computing*, 17(4):18–25, 2013.
- [72] Francesco Moscato, Rocco Aversa, Beniamino Di Martino, Teodor-Florin Fortiș, and Victor Munteanu. An analysis of mosaic ontology for cloud resources annotation. In *Computer Science and Information Systems (FedCSIS), 2011 Federated Conference on*, pages 973–980. IEEE, 2011.

- [73] Hassina Nacer and Djamil Aissani. Semantic web services: Standards, applications, challenges and solutions. *Journal of Network and Computer Applications*, 44:134–151, 2014.
- [74] V Sashi Kanth Nagireddi and Shakti Mishra. An ontology based cloud service generic search engine. In *Computer Science & Education (ICCSE), 2013 8th International Conference on*, pages 335–340. IEEE, 2013.
- [75] Nidal Nasser and Yunfeng Chen. Enhanced intrusion detection system for discovering malicious nodes in mobile ad hoc networks. In *Communications, 2007. ICC'07. IEEE International Conference on*, pages 1154–1159. IEEE, 2007.
- [76] Falak Nawaz, Kamran Qadir, and H Farooq Ahmad. Semreg-pro: a semantic based registry for proactive web service discovery using publish-subscribe model. In *Semantics, Knowledge and Grid, 2008. SKG'08. Fourth International Conference on*, pages 301–308. IEEE, 2008.
- [77] Suphakit Niwattanakul, Jatsada Singthongchai, Ekkachai Naenudorn, and Supachanun Wanapu. Using of jaccard coefficient for keywords similarity. In *Proceedings of the International MultiConference of Engineers and Computer Scientists*, volume 1, 2013.
- [78] Talal H. Noor and Quan Z. Sheng. Credibility-Based Trust Management for Services in Cloud Environments. In *Proc. of the 9th Int. Conf. on Service Oriented Computing (ICSOC'11)*, Paphos, Cyprus, Dec 2011.
- [79] Talal H Noor, Quan Z Sheng, and Abdullah Alfazi. Detecting occasional reputation attacks on cloud services. In *Web Engineering*, pages 416–423. Springer Berlin Heidelberg, 2013.
- [80] Talal H Noor, Quan Z Sheng, and Abdullah Alfazi. Reputation attacks detection for effective trust assessment among cloud services. In *Trust, Security and Privacy in Computing and Communications (TrustCom), 2013 12th IEEE International Conference on*, pages 469–476. IEEE, 2013.
- [81] Talal H Noor, Quan Z Sheng, Sherali Zeadally, and Jian Yu. Trust management of services in cloud environments: Obstacles and solutions. *ACM Computing Surveys (CSUR)*, 46(1):12, 2013.
- [82] Talal H Noor, Quan Z Sheng, and Athman Bouguettaya. *Trust management in cloud services*. Springer, 2014.
- [83] Talal H. Noor, Quan Z. Sheng, Anne H.H. Ngu, and Schahram Dustdar. Analysis of Web-Scale Cloud Services. *IEEE Internet Computing*, 18(4):55–61, 2014.
- [84] T.H. Noor, Q.Z. Sheng, A. Alfazi, A.H.H. Ngu, and J. Law. CSCE: A Crawler Engine for Cloud Services Discovery on the World Wide Web. In *Proceedings of IEEE 20th International Conference on Web Services (ICWS), 2013*, pages 443–450, 2013. doi: 10.1109/ICWS.2013.66.

- [85] Daniel Oberle, Alistair Barros, Uwe Kylau, and Steffen Heinzl. A unified description language for human to automated services. *Information systems*, 38(1):155–181, 2013.
- [86] Justice Opara-Martins, Reza Sahandi, and Feng Tian. Critical analysis of vendor lock-in and its impact on cloud computing migration: a business perspective. *Journal of Cloud Computing*, 5(1):4, 2016.
- [87] Aabhas V Paliwal, Nabil R Adam, and Christof Bornhovd. Web service discovery: Adding semantics through service request expansion and latent semantic indexing. In *Services Computing, 2007. SCC 2007. IEEE International Conference on*, pages 106–113. IEEE, 2007.
- [88] Pankesh Patel, Ajith Ranabahu, and Amit Sheth. *Service Level Agreement in Cloud Computing*. 2009.
- [89] Siani Pearson and Azzedine Benameur. Privacy, security and trust issues arising from cloud computing. In *Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on*, pages 693–702. IEEE, 2010.
- [90] Clément Quinton, Daniel Romero, and Laurence Duchien. Automated selection and configuration of cloud environments using software product lines principles. In *Cloud Computing (CLOUD), 2014 IEEE 7th International Conference on*, pages 144–151. IEEE, 2014.
- [91] Rajiv Ranjan, Liang Zhao, Xiaomin Wu, Anna Liu, Andres Quiroz, and Manish Parashar. Peer-to-Peer Cloud Provisioning: Service Discovery and Load-Balancing. In *Cloud Computing*, pages 195–217. Springer, 2010.
- [92] Jinghai Rao and Xiaomeng Su. A survey of automated web service composition methods. In *SWSWPC*, volume 3387, pages 43–54. Springer, 2004.
- [93] Molka Rekik, Khoulood Boukadi, and Hanène Ben-Abdallah. Cloud description ontology for service discovery and selection. In *Software Technologies (ICSOFT), 2015 10th International Joint Conference on*, volume 1, pages 1–11. IEEE, 2015.
- [94] Dominik Renzel, Patrick Schlebusch, and Ralf Klamma. Today’s top “restful” services and why they are not restful. *Web Information Systems Engineering-WISE 2012*, pages 354–367, 2012.
- [95] Miguel Ángel Rodríguez-García, Rafael Valencia-García, and Francisc García-Sánchez. Creating a Semantically-enhanced Cloud Services Environment Through Ontology Evolution. *Future Gener. Comput. Syst.*, 32:295–306, March 2014. ISSN 0167-739X.
- [96] Miguel Ángel Rodríguez-García, Rafael Valencia-García, Francisco García-Sánchez, and J Javier Samper-Zapater. Ontology-based annotation and retrieval of services in the cloud. *Knowledge-Based Systems*, 56:15–25, 2014.



- [97] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Inf. Process. Manage.*, 24(5):513–523, August 1988. ISSN 0306-4573. doi: 10.1016/0306-4573(88)90021-0. URL [http://dx.doi.org/10.1016/0306-4573\(88\)90021-0](http://dx.doi.org/10.1016/0306-4573(88)90021-0).
- [98] Myra L Samuels, Jeffrey A Witmer, and Andrew Schaffner. *Statistics for the Life Sciences*. Pearson education, 2012.
- [99] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, pages 285–295. ACM, 2001.
- [100] Cristina Schmidt and Manish Parashar. A peer-to-peer approach to web service discovery. *World Wide Web*, 7(2):211–229, 2004.
- [101] Aviv Segev and Quan Z Sheng. Bootstrapping ontologies for web services. *IEEE Transactions on Services Computing*, 5(1):33–44, 2012.
- [102] Jyothi Shetty and Demian Antony D’Mello. An xml based data representation model to discover infrastructure services. In *Smart Technologies and Management for Computing, Communication, Controls, Energy and Materials (IC-STM), 2015 International Conference on*, pages 119–125. IEEE, 2015.
- [103] Kwang Mong Sim. Agent-based cloud computing. *IEEE Transactions on Services Computing*, 5(4):564–577, 2012.
- [104] Subashini Subashini and Veeraruna Kavitha. A survey on security issues in service delivery models of cloud computing. *Journal of network and computer applications*, 34(1):1–11, 2011.
- [105] Le Sun, Jiangan Ma, Hua Wang, and Yanchun Zhang. Cloud service description model: An extension of usdl for cloud services. *IEEE Transactions on Services Computing*, 2015.
- [106] Yih Leong Sun, Terence Harmer, Alan Stewart, and Peter Wright. Mapping application requirements to cloud resources. In *European Conference on Parallel Processing*, pages 104–112. Springer, 2011.
- [107] Amirreza Tahamtan, Seyed Amir Beheshti, Amin Anjomshoaa, and A Min Tjoa. A cloud repository and discovery framework based on a unified business and cloud service ontology. In *Services (SERVICES), 2012 IEEE Eighth World Congress on*, pages 203–210. IEEE, 2012.
- [108] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. Arnetminer: Extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD ’08*, pages 990–998, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-193-4. doi: 10.1145/1401890.1402008. URL <http://doi.acm.org/10.1145/1401890.1402008>.
- [109] Simon Tong and Daphne Koller. Support vector machine active learning with applications to text classification. *Journal of machine learning research*, 2 (Nov):45–66, 2001.

- [110] Hong-Linh Truong, Schahram Dustdar, and Kamal Bhattacharya. Programming hybrid services in the cloud. In *International Conference on Service-Oriented Computing*, pages 96–110. Springer, 2012.
- [111] Alexander JAM Van Deursen and Jan AGM Van Dijk. Using the internet: Skill related problems in users’ online behavior. *Interacting with computers*, 21(5-6):393–402, 2009.
- [112] Martha Varguez-Moo, Francisco Moo-Mena, and Victor Uc-Cetina. Use of classification algorithms for semantic web services discovery. *JCP*, 8(7):1810–1814, 2013.
- [113] Cong Wang, Qian Wang, Kui Ren, and Wenjing Lou. Privacy-preserving public auditing for data storage security in cloud computing. In *Infocom, 2010 proceedings ieee*, pages 1–9. Ieee, 2010.
- [114] Xueshan Wang, Fuzan Chen, and Minqiang Li. Web service classification approach with an integrated similarity measure. In *Proceedings of the 23rd International Conference on Industrial Engineering and Engineering Management 2016*, pages 251–255. Springer, 2017.
- [115] Yi Wei and M. Brian Blake. Service-Oriented Computing and Cloud Computing: Challenges and Opportunities. *IEEE Internet Computing*, 14(6):72–75, 2010. ISSN 1089-7801. doi: <http://doi.ieeecomputersociety.org/10.1109/MIC.2010.147>.
- [116] Christof Weinhardt, Arun Anandasivam, Benjamin Blau, and Jochen Ster. Business Models in the Service World. *IT Professional*, 11(2):28–33, March 2009. ISSN 1520-9202. doi: 10.1109/MITP.2009.21.
- [117] Guo Wen-yue, Qu Hai-cheng, and Chen Hong. Semantic web service discovery algorithm and its application on the intelligent automotive manufacturing system. In *Information Management and Engineering (ICIME), 2010 The 2nd IEEE International Conference on*, pages 601–604. IEEE, 2010.
- [118] Yan Wu, Chungang Yan, Zhijun Ding, Guanjuan Liu, Pengwei Wang, Changjun Jiang, and MengChu Zhou. A multilevel index model to expedite web service discovery and composition in large-scale service repositories. *IEEE Transactions on Services Computing*, 9(3):330–342, 2016.
- [119] Feng Ye, Zhijian Wang, Zhenyu Yue, Xinkun Xu, and Yuansheng Lou. Cloududdi: An extended uddi model for cloud services. In *Cloud Computing and Intelligent Systems (CCIS), 2012 IEEE 2nd International Conference on*, volume 1, pages 499–503. IEEE, 2012.
- [120] Hyunjeong Yoo, Cinyoung Hur, Seoyoung Kim, and Yoonhee Kim. An Ontology-Based Resource Selection Service on Science Cloud. In *Grid and Distributed Computing*, volume 63, pages 221–228. 2009.
- [121] Seongwook Youn and Dennis McLeod. A comparative study for email classification. *Advances and innovations in systems, computing sciences and software engineering*, pages 387–391, 2007.

- [122] L. Youseff, M. Butrico, and D. Da Silva. Toward a Unified Ontology of Cloud Computing. In *Proc. of 2008 Grid Computing Environments Workshop (GCE 2008)*, pages 1–10, 2008.
- [123] Shoujian Yu, Jianwei Liu, and Jiabin Le. Decentralized web service organization combining semantic web and peer to peer computing. *Web Services*, pages 116–127, 2004.
- [124] Cheng Zeng, Xiao Guo, Weijie Ou, and Dong Han. Cloud Computing Service Composition and Search Based on Semantic. In *Proc. of the first IEEE International Conference on Cloud Computing Technology and Science (CloudCom 2009)*, pages 290–300, 2009.
- [125] M. Zhang, R. Ranjan, A Haller, D. Georgakopoulos, M. Menzel, and S. Nepal. An Ontology-Based System for Cloud Infrastructure Services’ Discovery. In *Proc. of the 8th International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom 2012)*, pages 524–530, Oct 2012.
- [126] Jianming Zhou, Tianlei Zhang, Hui Meng, Liping Xiao, Guisheng Chen, and Deyi Li. Web service discovery based on keyword clustering and ontology. In *Granular Computing, 2008. GrC 2008. IEEE International Conference on*, pages 844–848. IEEE, 2008.
- [127] Jing Zhou, Nor Aniza Abdullah, and Zhongzhi Shi. A hybrid p2p approach to service discovery in the cloud. *International Journal of Information Technology and Computer Science (IJITCS)*, 3(1):1, 2011.
- [128] Alejandro Zunino and Marcelo Campo. A survey of approaches to web service discovery in service-oriented architectures. *Innovations in Database Design, Web Applications, and Information Systems Management*, 107(1), 2012.



# Appendix A

## CV

### **Abdullah Alfazi Ph.D. Candidate**

School of Computer Science

Faculty of Engineering Computer & Mathematical Sciences

The University of Adelaide

Adelaide, SA 5005, Australia

abdullah.alfazi@adelaide.edu.au

Office: +61 88 313 4487

Mobile: +61 468 310 994

### **Research Interests**

Cloud Computing, Service-oriented Computing, Trust Management, Data Management, Information Retrieval, Data Mining and Quality of Assurance.

### **Residency**

Saudi citizen

## Highlights

- Awarded *King's Abdullah* selective **Postgraduate Scholarship**, 2013 - (**Ph.D.**).
- Awarded *King's Abdullah* selective **Postgraduate Scholarship** 2009 - 2012 (**M.Sc.**).

## Awards and Honours

- Awarded *King's Abdullah* selective **Postgraduate Scholarship**, (Saudi Arabian Ministry of Higher Education), Ph.D., The University of Adelaide (UoA), Adelaide, Australia, August, 2013-.
- Awarded *King's Abdullah* selective **Postgraduate Scholarship**, (Saudi Arabian Ministry of Higher Education), **M.Sc.**, in Computer Science, The University of Adelaide (UoA), Adelaide, Australia, 2010 -2012.

## Education

Jan 2013

The University of Adelaide (UoA) Adelaide, Australia

M.Sc. in Computer Science

Thesis: *Identifying Credible Feedback for Effective Trust Management in Cloud*

*Environments*

Principle Supervisor: Prof. Michael Sheng

CGPA: 4.88 / 7

January 2006

Teachers College Teachers College Jeddah, Saudi Arabia

B.Sc., Computer

CGPA: 4,05 / 5

---

## Research Experience

August 2013 - Current

PhD student

School of Computer Science, UoA      Adelaide, Australia

- Worked on, *Cloud Service Discovery and Analysis: A unified Framework*.

Feb 2011 - Dec 2012

Master student

School of Computer Science, UoA      Adelaide, Australia

- Worked on, *Trust Management for Services in Cloud Environments*; I designed and participated in implementing, the Cloud-Armor.
- Worked on, *a project for Robot detects mines*; I designed and implemented the mines field map.

## Industrial Experience

April 2007 - April 2009

Officer in quality and service management of IT Department.

Bank Aljazeera (www.baj.com.sa)

## Trainings

April 2007 to November 2007

E-commerce systems analysis

Contents of the course

Management servers.

Network (CCNA Introduction)

Database (SQL Language - SQL server)

Security Information (ISA server)

## Presentation and Workshop

1. I presented *Ontology-based Automatic Cloud Service Categorization for Enhancing Cloud Service Discovery*. The 19th IEEE International EDOC Conference (EDOC 2015). Adelaide, Australia, September 21-25, 2015
2. I presented *Towards Ontology-Enhanced Cloud Services Discovery*. the 10th International Conference on Advanced Data Mining and Applications (ADMA 2014). Guilin, China, December 19-21, 2014
3. I presented *Identifying Fake Feedback for Effective Trust Management in Cloud Environments* research paper at the 10th International Conference on Service Oriented Computing (ICSOC 2012), Shanghai, China
4. I participated in the *1st International Workshop on Analytics Services on the Cloud (ASC 2012)* at the 10th International Conference on Service Oriented Computing (ICSOC 2012), Shanghai, China

## Publications

### Refereed Conferences Publications

1. Identification as a Service: Large-Scale Cloud Service Discovery Over the World Wide Web. **Abdullah Alfazi**, Quan Z. Sheng, Wei Emma Zhang, Lina Yao, and Talal H. Noor. *The 5th IEEE International Congress on Big Data*. San Francisco, USA, June 27-July 2, 2016.
2. Ontology-based Automatic Cloud Service Categorization for Enhancing Cloud Service Discovery. **Abdullah Alfazi**, Quan Z. Sheng, Yongrui Qin, and Talal H. Noor. *The 19th IEEE International EDOC Conference (EDOC 2015)*. Adelaide, Australia, September 21-25, 2015



3. Towards Ontology-Enhanced Cloud Services Discovery. **Abdullah Alfazi**, Talal. H. Noor and Quan Z. Sheng, and Yong Xu. *the 10th International Conference on Advanced Data Mining and Applications (ADMA 2014)*. Guilin, China, December 19-21, 2014.
4. Cloud Armor: A Platform for Credibility-based Trust Management of Cloud Services. Talal. H. Noor, Quan Z. Sheng, Anne H.H. Ngu, **Abdullah Alfazi**, and Jeriel Law. *The 22nd ACM Conference on Information and Knowledge Management (CIKM 2013)*. San Francisco, CA, USA, October 27-November 1, 2013.
5. CSCE: A Crawler Engine for Cloud Services Discovery on the World Wide Web. Talal. H. Noor, Quan Z. Sheng, **Abdullah Alfazi**, Anne H.H. Ngu, and Jeriel Law. *The IEEE 20th International Conference on Web Services (ICWS 2013)*. Santa Clara Marriott, CA, USA, June 27 to July 2, 2013.
6. Detecting Occasional Reputation Attacks on Cloud Services. Talal. H. Noor, Quan Z. Sheng, and **Abdullah Alfazi**. *The 13th International Conference on Web Engineering (ICWE 2013)*. Aalborg, Denmark, July 8-12, 2013.
7. Reputation Attacks Detection for Effective Trust Assessment of Cloud Services. Talal. H. Noor, Quan Z. Sheng, and **Abdullah Alfazi**. *The 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom 2013)*. Melbourne, Australia, July 16-18, 2013.
8. Identifying Fake Feedback for Effective Trust Management in Cloud Environments. Talal H. Noor, Quan Z. Sheng, **Abdullah Alfazi**, Jeriel Law and Anne H.H. Ngu. *The 1st International Workshop on Analytics Services on the Cloud (ASC 2012)*. Shanghai, China, November 12-16, 2012.

## Thesis

9. Identifying Credible Feedback for Effective Trust Management in Cloud Environments, **Abdullah Alfazi**. Master Research project, in Computer Science, International the University of Adelaide (UoA), Adelaide, Australia, 2012. (Excellent mark)

## Professional Services

- PC member, Journal of Personal and Ubiquitous Computing
- External Reviewer, the 8th IEEE International Conference on Cloud Computing Technology and Science (CloudCom 2016)
- External Reviewer, the 5th IEEE (pending) International Conference on Big Data and Cloud Computing (BDCloud 2015)
- External Reviewer, the 13th Australasian Symposium on Parallel and Distributed Computing
- External Reviewer, The International Journal of Web Portals, 2014.
- External Reviewer, the 4th International Workshop on Adaptive Services for the Future Internet 2<sup>nd</sup> September 2014, Manchester, UK.

## Research Projects

- Cloud-Armor (2013 University of Adelaide, Australia)

<http://cs.adelaide.edu.au/~cloudarmor>

## Referees

**Prof. Michael Sheng,**  
Department of Computing

Faculty of Science and Engineering  
Macquarie University  
Sydney, NSW 2109, Australia  
Email: Michael.sheng@mq.edu.au

**Dr. Lina Yao,**

Lecturer  
School of Computer Science and Engineering  
The University of New South Wales  
Australia  
Email: lina.yao@unsw.edu.au

**Dr. Talal H Noor,**

School of Computer Science  
College of Computer Science and Engineering  
Taibah University  
Saudi Arabia  
Email: tnoor@taibahu.edu.sa

**Prof. Ali Babar**

School of Computer Science  
Faculty of Engineering Computer & Mathematical Sciences  
The University of Adelaide  
Adelaide, SA 5005, Australia  
Email: ali.babar@adelaide.edu.au