A methodology for Automatic Derivation of Cloud Marketplace and Cloud Intelligence

By

ASMA MUSABAH KHAMIS RASHID ALKALBANI

Centre for Artificial Intelligence
SCHOOL OF SOFTWARE
UNIVERSITY OF TECHNOLOGY SYDNEY

A dissertation submitted to the University of Technology Sydney in accordance with the requirements of the degree of DOCTOR OF PHILOSOPHY in the Faculty of Engineering and Information Technology.

OCTOBER 2018

In the name of Allah, most gracious and most merciful

وَقُل رَّبِّ زِدْنِي عِلْمًا

(O my Lord, increase me in knowledge.)
(Al-Quran 20:114)

To my grandfather

Certificate of Original Authorship

I certify that the work in this thesis has not been previously submitted for a degree nor has it been submitted as a part of the requirements for other degree except as fully acknowledged within the text. I also certify that the thesis has been written by me. Any help that I have received in my research work and the preparation of the thesis itself has been acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

SIGNED: Production Note:
Signature removed prior to publication.

DATE: October 2018

ACKNOWLEDGEMENTS

First of all, I would like thank ALLAH (GOD) Almighty for the wisdom he bestowed upon me, the strength, peace of my mind and good health in order to finish this research. I am grateful to my supervisor, Associate Professor Dr.Farookh Hussain, for his supervision and constant support. His invaluable help of constructive comments and suggestions throughout my thesis works has contributed to the success of this research. I am also grateful to my co-supervisor Dr Asif Gill for his support and knowledge regarding this topic.

I would like to express my deepest gratitude to my grandparents, parents, brothers and sisters. Without their continuous support and prayers this work would not have been accomplished. A special thank to my dearest aunt Dr.Salma and uncle Saud, I am grateful for their invaluable advice and unbelievable support they have provided throughout the journey.

I would like to thank all the members of Centre for Artificial Intelligence and School of Software at University of Technology Sydney for their support and friendship.

I would also like to thank all my friends (Belinda, Hayat, Ameera, Maral, Fatima, Ibtisam, Salma, Manal, Lamia, Ly, Lan and Vu) for their support, inspiration, unfaltering belief in my work and having confidence in me, as well as their patience during my time of intense work where without this, this Doctoral degree would never have been completed.

A special thank you goes to my beloved country, the Sultanate of Oman, which has allowed me to realise this beautiful dream of mine, and to achieve this noble goal.

Journal

1. Harvesting as a Service (HaaS): A framework and software for harvesting enterprise cloud services submitted to Enterprise Information Systems (accepted)

Conference

1. A. ALKALBANI, A. SHENOY, F. K. HUSSAIN, O. K. HUSSAIN,ANDY. XIANG,Design and implementation of the hadoop-based crawler for saas service discovery, 29th International Conference on Advanced Information Networking and Applications (AINA), IEEE, 2015, pp. 785–790.

2. A. M. ALKALBANI, L. GADHVI, B. PATEL, F. K. HUSSAIN, A. M. GHAMRY,ANDO. K.HUSSAIN, Analysing cloud services reviews using opining mining, 31st International Conference on Advanced Information Networking and Applications (AINA),IEEE, 2017, pp. 1124–1129.

3. A. M. ALKALBANI, A. M. GHAMRY, F. K. HUSSAIN, AND O. K. HUSSAIN, Blue pages:software as a service data set, 10th International Conference on Broadband and Wireless Computing, Communication and Applications (BWCCA), IEEE, 2015, pp. 269–274.

4. A. M. ALKALBANI, A. M. GHAMRY, F. K. HUSSAIN,ANDO. K. HUSSAIN, Harvesting multiple resources for software as a service offers: A big data study, International Conference on Neural Information Processing, Springer, 2016, pp. 61–71.

5. A. M. ALKALBANI, A. M. GHAMRY, F. K. HUSSAIN,ANDO. K. HUSSAIN, Predicting the sentiment of saas online reviews using supervised machine learning techniques, International Joint Conference on Neural Networks (IJCNN), IEEE, 2016, pp. 1547–1553.

6. A. M. ALKALBANI, A. M. GHAMRY, F. K. HUSSAIN,ANDO. K. HUSSAIN, Sentiment analysis and classification for software as a service reviews, 2016 IEEE 30th International Conference on Advanced Information Networking and Applications (AINA), IEEE, 2016, pp. 53–58.

7. A. M. ALKALBANI ANDF. K. HUSSAIN, A comparative study and future research directions in cloud service discovery, 11th Conference on Industrial Electronics and Applications (ICIEA), IEEE, 2016, pp. 1049–1056.

8. A. M. ALKALBANI ANDF. K. HUSSAIN, Quality cloudcrowd: A crowdsourcing platform for qos assessment of saas services, International Conference on P2P, Parallel, Grid, Cloud and Internet Computing, Springer, 2017, pp. 235–240.

9. A. M. GHAMRY, A. M. ALKALBANI, V. TRAN, Y.-C. TSAI, M. L. HOANG,ANDF. K. HUSSAIN,Towards a public cloud services registry, International Conference on Web Information Systems Engineering, Springer, 2017, pp. 290–295.

Abstract

From a consumer's perspective, a cloud services marketplace is essential for cloud services discovery, selection, and composition. In practice, there are some private cloud services marketplaces, such as the Microsoft Azure marketplace, which are available for consumers belonging to a given vendor only. Nowadays, with the increase in the number of cloud services advertisements, and the adoption of cloud services, the cloud services consumer-base has grown and is projected to expand significantly over time. This increase defines the need for cloud services marketplace to enable effective interaction with cloud services users. A considerable amount of research has conducted in the area of cloud service selection and composition; however, the majority of this research is focused on developing algorithms (such as matching algorithms) and assumes the availability of cloud service information.Furthermore, little attention was given to the efficient discovery of cloud services over the World Wide Web (WWW). According to our literature, no research addresses the need for cloud services marketplace. Hence, this thesis proposes to provide an automatic derivation of cloud marketplace. The design of this marketplace includes a combination of the following modules: 1) cloud services harvesting module; 2) knowledge base for cloud service module; 3) cloud service trust derived intelligence module.

The cloud services harvesting method is designed for harvesting cloud services advertisements from the web and building cloud services dataset. Such a dataset could be used by potential consumers for cloud services discovery and could be useful for future research in cloud selection, composition and recommender systems. Also, the developed cloud services repository could act as a knowledge source for constructing a standard ontology for cloud services. The knowledge base for cloud service module is designed for producing a solution toward cloud services marketplace to organise, publish and retrieve cloud services advertisements. This method involves semantically categories cloud services advertisements grounded on harvested web data to solve the issue of various cloud services advertisements. Also, this method includes the construction of the first commercial cloud services ontology-based repository for cloud services marketing. This repository contains service metadata that can be used to store service advertisements information which annotating to the domain-specific ontology concepts toward retrieving service advertisements more efficiently. The cloud services trust derived cloud Intelligence Module is designed to automatically analyzing the sentiment of cloud reviews to provide the potential consumers with real quality of service (Quality of Experience) information when making the buying decision. Also,

building cloud reviews classifier to automatically classify the reviews: positive, neutral or negative using supervised machine learning algorithms.The result of this thesis will be an intelligent methodology for an automated derivation of the cloud marketplace: cloud services harvester, cloud services knowledge base, and Quality of Experience of cloud services. This methodology will be useful to the potential consumers, cloud providers, and the research community, as it will provide easy access to cloud services advertisements information.

CHAPTER 1

INTRODUCTION

This chapter opens with an overview of Cloud Computing and Service Discovery in Section 1.1. Section 1.2 discusses cloud services advertising and discovery in web environment. Section 1.3 focuses on the difference between Quality of cloud Service and Quality of Experience. Section 1.4 presents the issues related to cloud services discovery. Section 1.5 describes the challenges related to Cloud Service Discovery that will be addressed in this thesis Section 1.6 explains the objectives of this doctoral study. Section 1.7 discusses the scope of this thesis. Section 1.8 presents the contributions of this thesis follow by section 1.9 presents the plan of this thesis.

1.1 Overview of Cloud computing and service discovery

1.1.1 Cloud computing

The term 'cloud computing' refers to the web-based provisioning of computing resources, such as hardware, development platform, and software, that are available on demand as-a-service over the World Wide Web (WWW) [34]. The term 'on-demand-service' means that an end-user can access these web-based resources via an online subscription service. The term cloud has long been used to denote the Internet; hence, the term 'cloud computing' essentially means 'internet-based computing' [83]. According to the National Institute of Standards and Technology (NIST), 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 [66]. In cloud environments, users from across the world share computing resources, so that end-users use these resources over the web without the need to

install or run the application on their computers or servers [26].

Typically, cloud computing offers three primary types of services: 'Software as a Service (SaaS)', 'Platform as a Service (PaaS)', and 'Infrastructure as a Service (IaaS)'. These services provide users with boundless computing resources and allow them to scale resources up or down according to their work requirements. This feature is one of the primary reasons for the widespread use of cloud resources around the world, especially given the need for highly scalable computing resources to cope with dynamically changing web environments [29]. Cloud users leverage the characteristics of cloud services, including resource pooling, measured service, broad network access, rapid elasticity or expansion, and on-demand self-service. These features make cloud services unique, and unlike other web- based services and applications which are designed with boundaries for specific business activities.

1.2 Cloud service advertising and discovery in web environment

According to [51] the term 'service advertising' refers to a service description that is presented via media, which is an essential factor in any marketplace. A typical cloud service advertisement presented online is more than service description itself. The cloud service description describes a total service offer by the cloud service provider to their consumer, and it includes some elements that add additional value to the consumers, such as Quality of Service (QoS) values and technical support details. The cloud service providers are providing their services offers via their websites, whereas these website schema and layout vary from provider to another. For example, the service offering template in Microsoft Azure marketplace website is unlike the one presented in Amazon cloud web marketplace website [3, 12]. Therefore, one of the challenges that the cloud consumers face is how to find a suitable and trustworthy cloud service on the web. In summary, the most popular method for service discovery task across all domains and industries is Google [86]

1.2.1 Features of Cloud services advertisement in web environment

According to [51], the service advertising has the characteristics of heterogeneity, and contextual dependence. Similarly, in the cloud marketplace context, cloud service advertising have the attributes of heterogeneity and contextual dependence, which are explained as follows

1. **Heterogeneity** The simple meaning of 'heterogeneity 'of service advertising is that even if the two service providers are offering the same service, the offer will be different, hence the experience will be totally different. Numerous cloud services providers are offering three types of services, including SaaS provides software application as-a-service, PaaS delivers software development platform-as-a-service; IaaS provides computing resources required to deploy software application and platform, including visual networking components,

hardware, and storage. The cloud services provider presents their services offerings using different schemas and structures across various web portals. Therefore, consumers could be faced a dilemma in having to making a choice from a diverse bunch of service advertisements that offer the same function across various web platforms. Therefore, the heterogeneous of service advertising needs to be controlled to give a uniform marketing and consumer experience.

2. **Contextual Dependence.** A service advertisement may have different content in different contexts [51]. For example, a SaaS service advertisement has different content from that of PaaS or IaaS. Moreover, the service features and the quality of services criteria may vary with the variation of contexts. In the above example, the QoS evaluation criteria for SaaS is different from that of PaaS and IaaS. Similarly, the QoS status of a service offered is dynamic along with its changing contexts. For example, a company may have a good reputation as an IaaS, but not an equally good reputation as a PaaS for business systems.

1.3 Quality of Cloud services / Quality of Experience

Generally speaking, QoS refers to the overall performance of services. In the context of this thesis, the QoS refers to the how reliable and fast the cloud service is defined by cloud service provider and the metrics such as availability and reliability [92]. The QoS is an essential factor for the purchase of a cloud service, whereas different end-users have a different expectation. However, currently the QoS information provided by cloud providers are not sufficient and do not reflect the real value of the service. In the current literature, there has been a considerable amount of research on measuring and monitoring the performance of cloud services to provide the QoS values such as [63]. However, they are all based on network parameters and do not reveal the experience of end-users using the cloud services. The Quality of Experience (QoE) refers to the overall performance of a cloud service as it is perceived by the end-user after using the service [84]. The QoE offers information to the cloud providers which can be used to improve the quality of service. The QoE can be obtained in forms that include surveys, posted reviews, rating, etc. In summary, the QoE can be a better and reliable indicator of the actual consumer experience.

1.4 Issues with Cloud services discovery in web environment

This section will discuss some of the issues related to cloud service discovery in the Web environment. In particular, this section focuses on the pressing issues with cloud service discovery on the web that need to be addressed. The discussion is divided into three parts. Section 1.4.1 discusses some issues related to using general Web Search engine for cloud services discovery. Section 1.4.2 discusses some of the urgent issues related to cloud service discovery in the Web

environment in an in-depth literature reviews. Section 1.5 lists the pressing research issues that will be addressed in this thesis.

1.4.1 Issues with using general web search engine for Cloud services discovery

The internet has become a global marketplace. Service suppliers (including cloud service suppliers) promote their businesses and services through their websites or web portals. The existing mechanisms to retrieve cloud service information are web search engines such as Google and Bing. These search engines are not, however, restricted to finding web information or electronic commerce. Therefore it may retrieve relevant; irrelevant information depending on the descriptive information provided. Furthermore, searching service information via web search engines depends on two factors: the keyword that best indicates the target service (which is the most important aspect in internet marketing and the web search engine's algorithm); and finding a service that is related to cloud services that best match user's needs. The latter can be challenging if the cloud marketplace is vast. Additionally, according to [76] there are a considerable number of websites relating to non-existing services.

Furthermore, using web search engines for service discovery is not an appropriate mechanism for a real-world scenario. For example, consider the following situation: a business owner is searching online for the best cloud providers in the location 'Parramatta, NSW'. She also wants to be able to purchase the best cloud services offers online. Currently, the conventional way to accomplish these tasks requires using a search engine such as Google to locate the websites for cloud services providers. Then, the business owner must search each of these websites for the desired cloud service provider, cloud service, compare the cloud services offers, compare the cloud services providers, choose the best option, and finally, perform online transactions to purchase the service. The challenges of the current approach are clear: finding services this way is time-consuming; the current web search engines are keyword-based, and may not find all the relevant websites; the search may return unrelated websites; and each site may come with an entirely different web page structured.

1.4.2 Issues with Cloud service discovery in research literature

In the research literature, considerable attention has been given to the issue of service discovery on the Web. This research, however, has primarily focused on web services discovery and ignores the cloud services discovery. Conversely, there is a large number of cloud services advertisements and information on the web that do not have methodologies for classification, annotation and discovery. At the time of writing this thesis, most of the research work in the field of cloud service discovery has focused on using web semantic technologies such as semantic search. However, none of the existing research studies take into consideration the development of a reliable registry for cloud services. Furthermore, none of this research takes into account the lack of an intelligent

method for analyzing cloud user experience, which can reveal consumers-oriented insights about the cloud services. This information can be very useful for cloud providers in improving their service offerings.

1.4.3 Issues with the quality of Cloud service

Given the importance of quality of service in purchasing a cloud service, in the research literature, there has been considerable attention given to it. Much of this literature has focused on measuring the quality of services using network parameters, which do not reflect the cloud consumers' experience. Also, the current QoS information that is present in the service advertisements is insufficient. The end-users usually have different expectations depending on factors that include their previous experiences, price, and so forth. None of the current studies, however, measure the overall performance of the services based on users' experiences. Perceived quality information by the end-user who has consumed the service can be an indicator of the actual value of quality and it assists the cloud providers in improving their services. Further details regarding this research issues can found in Chapters 2 and 3. From Chapter 4 to Chapter 7, we will present the solutions for these issues. In the next section, we identified some urgent issues with the cloud services discovery on the Web.

1.5 Issues with Cloud services discovery

This section summarizes some of the urgent questions with cloud services discovery that need to be addressed. These questions are as follows:

1. How we can harvest the information regarding available cloud services advertisements over the web from vast web information?
2. How can a reliable cloud services registry be built based on collected service information?
3. How can we annotate all available service advertisements over the Web based on a specific service domain knowledge?
4. How can a cloud service consumer precisely discover the most relevant cloud service advertisements without relevant domain knowledge about his/her service requested?
5. How can user experience be derived based on the harvested cloud reviews?

1.6 Objectives of the thesis

To address some of cloud services discovery issues previously discussed, the objectives of this thesis are summarized as follows:

- **Research objective1:** To develop an intelligent harvester to collect information from heterogeneous cloud services sources.
- **Research objective 2:** To develop a reliable cloud services registry based on collected cloud services information.
- **Research objective 3:** To develop an intelligent methods for determining the posted reviews intention of the reviewer and also give the overall Quality of Experience (QoE) of a product/service on harvested reviews.
- **Research objective 4:** To validate the above-developed methods by building a prototype system.

1.7 Scope of the thesis

This thesis proposes a methodology for Automatic Derivation of Cloud Marketplace and Cloud Intelligence which has three modules. The first module involves harvesting cloud services across various web portals. The second module involves building cloud services knowledge base. The cloud services knowledge base module has two main tasks: the first task involves constructing a service advertisement meta-data based on the harvested cloud services information from heterogeneous web resources in module 1. The second task involves building a service domain ontology that presents the service knowledge in the specific domain. The third module is Cloud Services Trust Derived Intelligence, which aims to analyze the harvested cloud reviews to determine negative reviews, positive reviews, and neutral reviews. The result of this task is a sentiment analysis that has labels services as being 'positive', 'negative' or 'neutral'. Then, building a machine learning classifier, which can intelligently classify the cloud reviews using machine learning method into positive, negative or neutral. To achieve this, there is a need for a training data which is a sentiment dataset that has been generated in the first task. Also, for the proof-of-concept, in this scope of this study we harvested information from Serchen.com and GetApp.com

1.8 Contributions of the Thesis

This section presents the thesis research contributions. These contributions are classified into 'scientific contributions' and 'social contributions'.

1.8.1 Scientific contributions

1. This thesis presents a thorough and up to date review of the existing work in the field of cloud services discovery.

2. This thesis focuses on providing an intelligent method for harvesting heterogeneous services advertising across different web portals. The results demonstrate that the method in harvesting services is superior to other existing approaches.

3. This thesis focuses on providing a dataset of cloud services offers, along with dataset on consumers' reviews and cloud sentiment dataset, such dataset was not available previously. These datasets can be great resources for researchers in the field of cloud services selection, discovery, and composition.

4. This research focus on providing reliable, cloud services knowledge base that contain collected cloud services information.

5. This research focuses on applying machine learning methods to determine the posted reviews intention of the reviewer (positive, negative or neutral), which gives overall Quality of Experience (QoE) of a product/service on consumers reviews. This is the first work in the existing literature of its type.

6. This research focuses on providing a classifier to predict the sentiment of cloud reviews in the future (positive, negative or neutral).

1.8.2 Social contributions

1. From the consumer's perspective, this thesis provides a public cloud services dataset, along with complete cloud services listings. These listings include information about the services, alongside Quality of Experience based on consumers reviews. This will be useful information for cloud consumers.

2. From the consumer's perspective, this thesis provides knowledge about clouds services that will assist end-users to more accurately locate a useful service.

3. From the provider's perspective, this thesis provides a semantics-based platform for presenting cloud services, which help in discovering cloud services advertisements.

1.9 Plan of the thesis

The thesis is organized into eight chapters. The following are brief summaries of each chapter:

1. **Chapter 2:** provides a thorough review of existing studies in the field of cloud services discovery. This purpose of this review is to understand the current approaches and identify the gaps in the literature.

2. **Chapter 3:** defines and discusses the research problem of cloud services discovery. The defined problem is divided into four research issues, followed by the research questions

formulated that have been based on these research issues. To provide a solution for the research questions, the existing scientific research methods are briefly discussed and the selected research method is described in details.

3. **Chapter 4:** opens by defining the key concepts that are used to describe the solution. The chapter then presents the overview solution to the research questions addressed in Chapter 3. This is followed by an overview of solutions for each of the research issues that are outlined in chapter 3.

4. **Chapter 5:** presents a service-based harvester for crawling cloud services offers across heterogeneously structured web portals. This harvester allows the end-users to harvest the customized data without the need to customize the harvester for each particular website.

5. **Chapter 6:** presents cloud services knowledge base. This module aims to create an ontology-based model to classify service advertisements and generates meta-data to describe service advertisements, which can be perceived as a schema for these advertisements. The validation of this module has been achieved by implementing a prototype which consists of two sub-processes. These sub-processes include structuring service information, and annotating and populating services.

6. **Chapter 7:** presents the module 3 Cloud Services Trust Derived Intelligence, which has two main tasks. The first task involves analyzing the cloud reviews to determine whether the reviewers' attitude (positive, negative or neutral). The result of this task is the sentiment dataset, which is used as a training dataset for the second task. The second task involves applying several machine learning methods to build the cloud reviews classifier. This classifier can predict the intention of reviews positive, negative or neutral. This work is significant because it analyzes the consumers reviews in order to provide the overall quality of experience value.

7. **Chapter 8:** Chapter 8 concludes the thesis and summarises the results of the experiments.

1.10 Conclusion

This chapter has provided an overview about cloud computing and cloud services discovery. The chapter has explored the features of cloud service advertising, namely heterogeneity and contextual dependence. The chapter has discussed the existing issues related to cloud services discovery. There has been a discussion of the thesis' scope and contributions. Finally, there have been summaries of each chapter. The following chapter, Chapter 2, will provide a thorough review of the existing research studies in the current literature.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

This chapter aims to provide a background to the state of the art cloud service discovery approaches by examining how these approaches have been examined in the existing literature. The chapter analyzes each of the contexts, features and methods for each of the approaches. The chapter also discusses the limitations of each approach. Section 2.2 describes the plan followed to conduct this systematic review. Section 2.3 presents an overview of existing work on cloud service discovery. Section 2.4 describes semantic-based approaches, while section 2.5 describes non-semantic-based approaches. Section 2.6 critically evaluates the existing literature, which followed by chapter conclusion. The overall aim of the chapter is to provide a review of existing work on cloud services discovery along different dimensions. Such a review has not yet been undertaken.

2.2 Systematic review planning

[32] propose using the systematic reviews for software engineering area in evaluating and collecting more evidence on a specific research topic. The systematic review undertaken in this thesis is grounded on reviews and synthesis studies related to raised questions in order to make clear the current interest in this subject. We have identified the most related sufficient studies using various resources such as Google Scholar [8] and Scopus [16]. Our objective is to provide a thorough review of existing work on cloud services discovery. We also aim to classify the existing literature based on the proposed approaches and the method used, to achieve the above mentioned. In doing this, we have followed the procedures reported [62, 65]. The procedures

including the following steps: firstly, we formulated the research questions. These questions limit the scope to cloud service discovery. The questions are as follows:

1. How to group the existing studies into useful classifications from a scholar point of view?

2. What are the main concerns of each study?

3. What is the proposed approach context?

4. What evaluation producers has followed to assess the result in each study and to consolidate this approach?

5. What are the limitations of each study?

Secondly, in our study, we used the five following sources: Google Scholar [8], Scopus [16], IEEE Xplore [9], Springer [19], Science Direct [15] and Semantic Scholar [18]. Thirdly, we specified the search keywords and condition within the scope of the research questions mentioned above. Our primary research keywords were 'cloud service discovery' and 'cloud service registry'. We also used some other keywords, including 'cloud services crawler engine', 'cloud service description' and 'service discovery related by Boolean AND'. Fourthly, we selected the criteria to evaluate the relevant studies, and to exclude other studies that did not fit this criteria. Finally, we collected and evaluated the most relevant studies.

2.3 Overview of Cloud service discovery existing approaches

The recent growth in the use of cloud computing technologies has renewed interest in service discovery [80]. At the time of writing, very little attention has been paid in the existing literature to the problem of a lack of mechanism for cloud services discovery in the web environment. A comparative study of existing approach in cloud services discovery shows that most of the existing proposed cloud discovery solutions use semantic technology, especially with rapid development in ontology mark-up language. Other proposed solutions include the semantic agent(s) which combine agent/broker protocol with the using of semantic technology for service discovery. [25, 78, 85]. In this thesis, we have classified the related work into two main approaches based on the technologies used. These approaches are as follows: the semantic-based approach and the non-semantic-based approach. The semantic-based approach was further classified into five categories: semantic service registry, semantic agent(s), semantic service crawling, semantic service annotation, and semantic service matching. In non-semantic-based existing work, there is only one method, and that is the cloud crawler for cloud services repository.

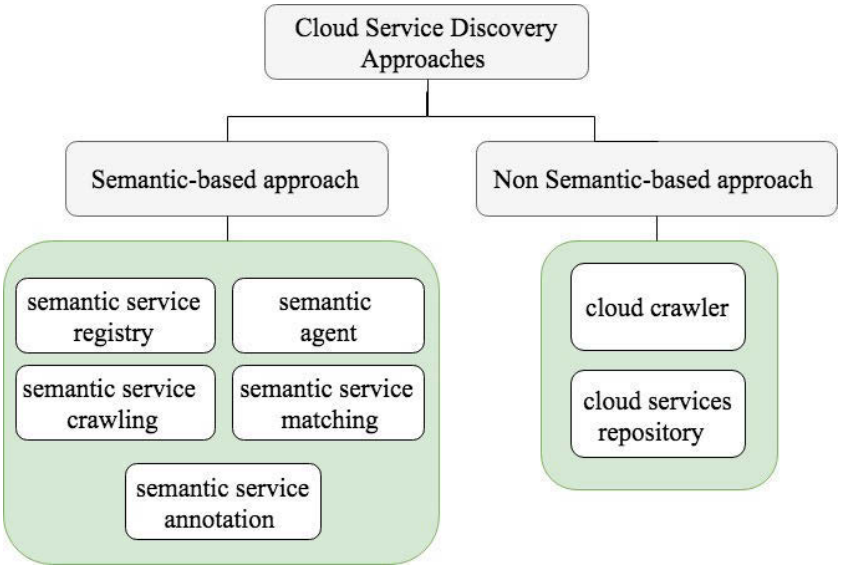


Figure 2.1: Overview of the literature review of the Cloud Service Discovery Approaches

2.4 Semantic-based approaches

Semantic technology such as semantic search and semantic annotation are widely applied in the area of service discovery for web services discovery [89] and ecosystem services discovery [39]. The semantic search aims to improve search accuracy by understanding the intent and contextual meaning of the words the searcher is using in a search [48]. The semantic annotation means providing additional information about various concepts such as organization, people, places etc in any given content [61]. Therefore, this technology can also be applied on cloud services discovery. A thorough review of the current studies shows that most of existing solutions proposed for cloud services discovery use semantic technologies, which we classified in this thesis as follows:

2.4.1 Semantic services registry

The semantic services registry approach is a central services registry that provides a listing of services description using a certain schema. Consumers can use this registry to locate a service that will be of assistance to them [58]. The registry approach usually involves three main parties: the suppliers, the consumers, and the registry. The registry allows any service suppliers or developers to register their list of services, along with a description of these services. In the current literature, the proposed semantic service registry is based on and it extends DAML-S web service annotated description to describe cloud services and their special features. Description elements for cloud services include the required input, the operation of the cloud service, and the service's output [35]. The cloud service description is then published in the web service registry (UDDL) to allow the dynamic discovery of cloud services from one central place. This approach

provides suppliers with an easy way to publish their services in a single location; and it facilitates the service discovery process for the consumer. However, a completely centralized solution that includes both web services and cloud services will have problems coping with the growing market, as well as offering up-to-date services. Moreover, this proposed cloud service description using WSDL does not support some other critical market parameters such as QoS parameters, which are essential factors in deciding whether or not to purchase a particular service.

A study undertaken by [72] highlights a core ontology that can be used to present a cloud service offering in a semantic registry. This ontology, however, focuses only on functional aspects, for example, service capabilities, functional interface, and management interface. This could be useful for selecting a cloud service based on its functionality. The ontology modelled using Web Service Modeling Ontology (WSMO) aims to describe web services. This study does not consider the general information of cloud service offered along with the Quality of Service parameters, which is an essential factor for purchasing a cloud-based service.

2.4.2 Semantic agent(s)

[56] present a cloud service discovery system (CSDS) that assists cloud consumers to find cloud services in the Web environment. The prototype of the CSDS comprises a user interface and three agents: a query processing agent, a filtering agent, and a reasoning agent. The main purpose of CSDS is to find the best cloud services for the consumers by consulting a Cloud services ontology, which comprises a taxonomy of cloud service concepts. The user interface allows the consumer to write a query in the search bar which specifies their preferences, including a service name and service requirements. The processing agent expands the query to improve search accuracy. This query is then sent to Google or any other web search engine to retrieve service information. The results showed that by using semantic search (which based on consulting the cloud ontology for query expansion), the CSDS is more likely to discover the service that has been requested. The rating of the cloud services was partially implemented in the CSDS, but rating the services based on QoS was not considered in the process. Also, the evaluation tool for the CSDS system only included the use of pseudo dataset.

[85] developed Cloudle, which is a multi-agent cloud service discovery system that focuses on matching service requested by a consumer with available advertised cloud services. Cloudle comprises four agents: the cloud search engine agent, the negotiation agent, the cloud commerce agent, and the cloud service composition agent. These self-organizing agents are designed to assist consumer to find advertised cloud services and to manage cloud resources. Cloudle considers that there are different types of cloud services are provided SaaS, PaaS and IaaS. Similar to previous work, the authors in this study are using the semantic search technique to find the requested service. Generally speaking, the semantic search can improve the search accuracy and retrieved information from the web, while the web search engine is a keyword-based search engine that may retrieve irrelevant information. The web search engine also ignores the importance of the

QoS for service discovery and the service purchasing process.

[82] proposed an approach that combines the advantages of cloud service ontology technique with a multi-agent-based protocol for cloud service discovery. This approach consults cloud service ontology concepts to discover and retrieve information about a cloud service. The system was developed with an interface that supports the consumer in locating appropriate cloud services. The results showed that using a semantic search matching mechanism by consulting cloud services ontology can be helpful in meeting consumer requirements and retrieving cloud services information efficiently compared to the traditional approach (a keyword- based search engine). However, this is an offline system and it has not been evaluated in a web environment. Additionally, this system does not provide any solution for retrieving cloud services information in an intelligent manner in the web environment.

[78] introduced a cloud service semantic annotation framework for cloud service description based on a multi-agent approach to support finding cloud services based on their location. This work addressed the optimal discovery of cloud services by referring to a shared cloud ontology to locate an appropriate service. This approach failed, however, to incorporate QoS parameters of cloud services as a factor for the semantic reasoning mechanism.

2.4.3 Semantic service crawling

Very few studies have attempted to develop an intelligent method for crawling cloud service over the internet. [76] proposed a cloud crawler engine to a locate cloud services supported by cloud services ontology. The results from this crawler showed that 5883 cloud services offered online, including SaaS, PaaS and IaaS. These included around 1552 web services. However, the Web portal was crawled through the search engine, not through the developed crawler. The collected dataset included meaningless cloud services information and does not provide sufficient information about cloud services, such as service name, service provider, service description, service rating, and QoS.

2.4.4 Semantic service matching

The semantic matching in search approaches focuses on improving the search accuracy using web search engines by understanding the search query intent using a domain ontology. This approach is one of the efficient solutions that have been found by existing studies to solve the issue of the cloud services discovery in the web environment. For example, [23] developed a service discovery framework that was supported by a unified SaaS ontology to promote semantic-based matching while querying for SaaS services information. The developed framework was composed of three main parts: ‘service registration’, ‘service discovery’, and ‘service ranking’. The ‘service registration’ layer provides mapping between service description and SaaS ontology concepts; and it helps the repository to store service information and the ontology details. This layer also provides a clustering technique to group the services according to their functionality. The

‘service discovery’ layer is responsible for semantic search matching between user,Äôs query and service cluster. In this study, the authors make use of the business ontology model to classify and describe SaaS; while SaaS offers may offer different features and functionalities from business services. Their work does not, though, consider the problem of a lack of an intelligent method for discovering cloud service information in the web. Also, it is offline system and this study does not include a proper evaluation for this approach in a real world environment.

Similarly [22] proposed a semantics-based service discovery approach which utilizes SaaS ontology for storing and retrieving the service information. The SaaS ontology comprises of SaaS domain concepts, SaaS characteristics concepts, SaaS QoS metrics concepts, and SaaS offer concepts. Generally speaking, the proposed framework is comprised of service registration, service e s discovery and service selection. The service registration module is responsible for register ing and clustering service offer into groups using SaaS ontology and Agglomerative Hierarchical Cluttering. The service discovery process is responsible for matching between the users requested query and service cluster. A major shortcoming of this research is that they proposed reusing the existing business services model ontology, such as in [31] to describe cloud services. This could be useful, although the details of cloud resources and cloud services ontologies are different from the existing business services ontology.

Furthermore, to assist business organizations to find an appropriate cloud business service, Tahamtan et al [90] proposed a semantic-based framework for cloud service discovery supported by cloud service ontology. The ontology includes most of the business function concepts and classifications outlined in [31]. In order to locate the right service with the right provider, the ontology is designed to map between the cloud service concepts and business concepts. Additionally, the ontology includes some other important service attributes, such as service characteristics and service delivery model. Unfortunately, they proposed to reuse the existing business services model ontology, which is not applicable in a real world scenario. Additionally, Maheswari et al [67]proposed a semantic-based approach for discovering for cloud-based emergency services such as health care. Their approach (which is also supported by cloud service ontology) is used to find the most relevant services by matching between the user’s query and ontology concepts [55].

2.4.5 Semantic service annotation

A suggested approach to semantically annotate cloud services involves using the extension of WSDL called Semantic Mark-up for web service (DAML-S) [27] and then storing the semantic annotation of the cloud service in web services registry, such as UDDI [38]. This approach was initially proposed by Chen et al [35]. The cloud service is described in the form of three parameters: Operation, Input and Output. This work demonstrates a practical solution for the dynamic discovery of cloud services, although it is an entirely centralized solution that includes web services and cloud services. This cloud service has the ability to cope with the growing marketplace, and it updates in a real-time data manner. Furthermore, QoS parameters were not

included as a feature in the service description representation and service discovery process.

Another study presented a semantic annotation approach for the cloud service profile [93].The service profile consists of the following attributes: service name, service price, service features, and service level of agreement (SLA). The study proposed to have ontology for each attribute and then combines all of them to construct a global service profile ontology. The final global ontology has 64 concepts and 128 properties. Unfortunately, their approach neglects the need to account for QoS attributes in selecting the best service.

2.5 Non Semantic based approach

A few studies have considered developing a solution for cloud services by using techniques other than the semantic technologies, for example, building a repository for cloud services and developing a web crawler specially design for the purpose of retrieving cloud services.

2.5.1 Cloud services crawler

To deal with the lack of a dynamic cloud service discovery method, [46] developed three versions of a cloud crawler to gather cloud service information from three different providers: Amazon Web Services (AWS) [4], Rackspace [14], and GoGrid [7]. The gathered data has two tuples, service specification and service price. Further analysis was applied to the service specification using a K-mean clustering algorithm to cluster cloud services in a category. One of the shortcomings of this work was the need for the crawler to be customized for every website. Therefore, the proposed method in this study failed to crawl cloud services efficiently, since it is a time-consuming for the developers to code and customize the crawler for each Web supplier in order to gather cloud services information in a local repository. This study does not propose any solution to assist the consumer in finding cloud services in the web.

Table 2.1: Summary of studies by cited authors

Author(s)	Approach	Concerns	Contributions	Remarks
[35]	semantic service registry	No registry for cloud services shows	how DAML-S can be used to semantically annotate cloud services in the existing structure of Web service registry (UDDI)	Clearly demonstrates an effective solution for dynamic discovery of cloud services; however, a centralized solution that includes Web services and cloud services has difficulty coping with the growing number of cloud services. Does not consider QoS parameters.
[35]	semantic service registry	No registry for presenting cloud services offerings	how can use Web Service Modeling Ontology (WSMO) to describe the cloud services offer	this approach focused on functional aspects only while selecting cloud services offer and neglect the important of cloud service general information such as description, price,
[49]	semantic agent	No mechanism to support cloud service discovery	integrated semantic technology with an agent-based protocol	Intelligent discovery for cloud services and QoS issues were not considered.
[85]	semantic agent	The lack of QoS information was not considered	Proposes automated discovery and price negotiation of cloud services	Cloudle is still in an early stage of cloud resources discovery and management. The intelligent discovery of services is not considered in the process, and the study fails to incorporate QoS parameters as a factor when finding services.

Continued on next page

Table 2.1 – *Continued from previous page*

Author(s)	Approach	Concerns	Contributions	Remarks
[82]	semantic agent	Locating optimal cloud services based on end-user requirements	integrated semantic technology with an agent-based protocol to retrieve the best cloud service that match user needs	The retrieved service information does not include QoS information about cloud services.
[78]	semantic agent	No standard for publishing cloud services. Use of different vocabulary and terminology by cloud providers for similar features and operations	integrated semantic technology with multi-agent protocol	This approach fails to incorporate QoS parameters of cloud services as a factor in the semantic reasoning mechanism.
[76]	semantic crawler	No mechanism for dynamic discovery of cloud services	Proposes a cloud crawler engine to a locate cloud services supported by cloud service ontology.	Does not add to previous work in the area of cloud service discovery.
[23]	semantic services matching	Lack of mechanism for retrieving SaaS services information on the Web.	An ontology-based SaaS services discovery approach.	The constructed SaaS ontology is based on reusing the existing business ontologies
[22]	semantic services matching	No mechanism for supporting SaaS services discovery	This paper proposed a semantic based approach for SaaS discovery.	Though semantic-based approach was used in this approach, the constructed SaaS ontology is based on re-using of existing ontologies in the literature.

Continued on next page

Table 2.1 – *Continued from previous page*

Author(s)	**Approach**	**Concerns**	**Contributions**	**Remarks**
[90]	semantic services matching	Lack of mechanism for cloud services discovery	The contribution was a semantic-based framework for cloud service discovery supported by cloud service ontology.	Unfortunately, the constructed cloud service ontology is based on re-using the existing business ontologies.
[67]	semantic services matching	There is no mechanism for supporting cloud-based emergency services, such as health care.	A semantic-based approach for discovery for cloud-based emergency services	The constructed cloud services ontology is based on re-using of existing ontologies.
[35]	semantic services annotation	Lack of semantic an-notation for cloud services on the Web.	The significance of this work was the introduction of extending web service description language (WSDL) to describe cloud services and their special features	A centralized solution that includes both web services and cloud services has problems coping with the growing market, and offering current services. Also, this work did not take into account some other service parameters such QoS attributes.
[93]	semantic services annotation	The issue of lack of semantics annotation for cloud services on the Web.	This work presented semantic annotation approach for cloud service profile.	This approach failed to incorporate QoS parameters of cloud services as factors for cloud services discovery.
[46]	cloud crawler	Lack of dynamic discovery method for cloud services.	Develops a crawler to harvest cloud services information.	The crawler has to be coded and customized for each targeted Web portal.

Shortcomings of existing literature reviews

Based on the above discussion of existing literature, in the area of cloud services discovery, the shortcomings are identified as follows:

1. Researchers suggested using existing ontologies such as Business services ontology to semantically annotate cloud services. This is not sufficient to describe the actual cloud services in real world scenarios;

2. Researchers do not provide any solution for structuring cloud service advertisements and annotating cloud services advertisements according to the published data online in order to solve the problem of heterogeneity in services advertisements;

3. Researchers do not provide any means of evolving a knowledge base or commercial cloud services repository for the cloud services industry;

4. Researchers do not propose any means by which cloud consumers can choose a cloud services with the best QoS;

5. Researchers do not propose any means of providing cloud services dataset along with QoS value for research community.

Table 2.2: Distribution of studies based on publication channel

Publication Source	**Type**	**Year**	**Number**
Springer Conference on Advanced Research on Computer Science and Information Engineering	Conference	2011	1
27th International Conference on Advanced Information Networking and Applications Workshops	Conference	2013	1
IEEE conference on Cyber-Enabled Distributed Computing and Knowledge Discovery	conference	2010	1
IEEE Transactions on Services Computing	Journal	2011	1

Continued on next page

Table 2.2 – *Continued from previous page*

Publication Source	**Type**	**Year**	**Number**
International Conference on Advances in Computing, Communication and Information Science	Conference	2014	1
Springer Intelligent Computing, Communication and Devices	Conference	2015	1
IEEE International Conference on Web Services	Conference	2013	1
Asian Journal of Information Technology	Journal	2016	1
IEEE International Conference on Computer Engineering and Systems	Conference	2013	1
IEEE Eighth World Congress on Services	Conference	2012	1
International Journal of Engineering and Technology	Journal	2014	1
Springer Conference on Advanced Research on Computer Science and Information Engineering	Conference	2011	1
International Journal of Grid and Distributed Computing	Journal	2014	1
IEEE International Conference on Software Engineering and Service Science	Conference	2014	1

2.6 Results Obtained

In this systematic study, we identified the most relevant research papers (see Table 2.1). The publication of papers is spread across journals and conferences proceedings. Out of 13 papers, 4 have been published in journals; the remaining 12 have been published in various international conferences. Furthermore, our result in Table 2.2 shows that the studies equally distributed

across the publication sources. In fact, each publication source has only one paper published in it, which means there is no publication source is preferred by cloud service discovery researchers. Regarding the year of publication, to the best of our knowledge, there is no study significant study related to cloud service discovery prior to 2010. Table 2.2 shows the distribution of reviewed papers, which were published from 2010 to 2016, alongside their publication sources. Table 2.1 indicates that 85 percent of the studies are focused on using semantic techniques to enhance cloud service discovery method. Also, we observed that integrating semantic techniques with agent protocol as well as the semantic search matching are the most common techniques used to solve the issue of cloud service discovery. To conclude, despite a considerable amount of research undertaken on addressing various challenges in Cloud computing such as migration and data processing, cloud services discovery is still largely an untouched area.

2.7 Critical Evaluation Approaches on Cloud Service Discovery

In this section, we present a comparative analysis of the existing work on cloud service discovery. To compare the existing work, the following criteria have been compiled from a thorough review of the reviewed literature. These criteria take into account the various factors that go with cloud service discovery and selection. Moreover, these criteria are important in identifying the most appropriate solutions for selecting the problem at hand; defining the limitations and challenges in the current approaches to improving cloud service discovery; and delivering better solutions in the future.

a. **Ontology Domain :** This criteria defines and describe the all relevant concepts in a single domain of interest.

b. **Knowledge Repository :** If an approach provides a repository for cloud services that is publicly accessed. Such a repository is important because it provides an insight into the cloud service market. This insight could be a very helpful for a potential consumer.

c. **Harvesting Crawler :** If the approach supports harvesting a cloud services information from the web.

d. **Quality of Service (QoS)** : This criteria determines whether an approach provides QoS data at the time of service discovery. Such information is crucial in the selection of the most trustworthy and reliable service. service.

e. **Public Dataset :** If an approach provides the research community with a cloud services dataset and cloud service QoS dataset that are publicly accessible.

f. **Evaluation model :** If the approach proposes any means of comparing their proposed method and results with the relevant approach

Table 2.3: Comparative analysis of Existing Approaches

Source	**Ontology Domain**	**Knowledge Repository**	**Harvesting Crawler**	**QoS**	**Public Dataset**	**Evaluation method**
[35]	NIST	X	X	X	X	X
[49]	NIST	X	X	X	X	X
[85]	NIST	X	X	✓	X	X
[82]	NIST	X	X	X	X	X
[78]	NIST&CMT	X	X	X	X	X
[76]	NIS	X	X	X	✓	X
[23]	SaaS&CMT	X	X	✓	X	X
[22]	SaaS&BS	X	X	✓	X	X
[90]	SaaS&BS	X	X	X	X	X
[67]	MS	X	X	X	X	X
[35]	NIS	X	X	X	X	X
[93]	NIS	X	X	X	X	X
[46]	-	X	✓	X	X	X

The comparative analysis of the reviewed studies demonstrates that a variety of approaches can be used to locate a cloud service. The existing approaches use different semantic techniques such as semantic annotation, semantic search matching, and semantic registry. Also, the majority of the methods are constructing the semantic content of cloud services domain-specific ontologies based on NIST definition and classification, as well as the cloud marketplace. Although, none of the existing studies consider constructing cloud service ontology based on web data. Also, it is apparent from this table that despite the importance of QoS in making a purchase decision, very few studies, such These marketplaces are the major source of cloud computing concepts, as well as terminology that is commonly used by the web community in a real environment. Also, it is apparent from the above table discussion table that despite the importance of QoS in making a purchase decision, very few studies, such as [85], and [23], consider involving the QoS values in in the cloud service discovery process. Additionally, this comparison demonstrates that none of the previous studies proposed a publicly available knowledge repository that could be used by both

[1]NIST:National Institute of Standards and Technology
[2]CMT: Cloud Market Terminologies
[3]SaaS: Software-as-Service
[4]BS: Business Services
[5]MS: Emergency Services e.g. Health-care
[6]x:implies that the study does not address this dimension
[7]✓:implies that the study addresses this dimension
[8]-:implies that the study addresses this dimension

the consumer and business organisations. In the existing literature, there is a lack of information about cloud service dataset or QoS data for cloud services; none of the studies evaluated their model using real cloud data. Also, we found that none of the existing studies were concerned with the evaluation process of their proposed approach, including verification and validation for the proposed system. In summary, the critical issues to emerge from the studies reviewed in this section are as follows:

1. None of the existing approaches propose an intelligent and user-friendly method for harvesting cloud services data from the web. Furthermore, they do not take into account the heterogeneous structures of the websites. By user friendly, we mean people with no technical or programming experiences can run the harvester for collecting information.

2. None of the reviewed works provide an intelligent method with which to engineer cloud services ontology grounded on the web data to deal with the heterogeneous cloud data on the web.

3. There is no method that takes into account the construction of a knowledge repository for cloud services based on real cloud services data.

4. None of the studies focuses on deriving intelligence from cloud reviews data, which is spread across multiple web portals. The derived intelligence could focus on the overall users' experiences.

2.8 Conclusion

This chapter has undertaken a systematic survey of the literature on cloud services discovery. The chapter began by providing an overview of the existing approaches to this topic. The different methods of cloud services discovery have been classified into two approaches-the semantic-based approach and the non-semantic-based approach-based on the technology used. The features and shortcomings of each approach were identified. Finally, the reviewed studies were evaluated based on different aspects that have been compiled from a thorough review of the reviewed literature. This evaluation took into account the various factors that effect with cloud service discovery. We found that there are still many gaps in the area of cloud services discovery that need to be addressed.

CHAPTER 3

PROBLEM DEFINITION

Service discovery on the web has been a long-standing issues for web developers and end-users. Universal Description, Discovery and Integration (UDDI) is the most famous approach for web service retrieval in the web environment. Additionally, scholars such as [28, 59] have proposed a method for service retrieval in several service domains. However, few studies have been undertaken in the area of cloud service discovery; most of the current research focuses on the design of a semantic-based cloud service discovery approach. In the current literature, there is an absence of studies that could assist consumers to obtain a list of services available on the web in a complete and accurate manner. The literature review undertaken in Chapter 2 highlighted the current limitations and challenges in cloud service discovery approaches. Section 3.1 of this chapter presents the key terms and concepts that are used to define the research problems. In section 3.2, we formally define and present the research gaps that we intend to address in this thesis. These gaps were found by conducting a thorough review of the related literature and studies in Chapter 2. Section 3.3 will discuss the research problems, while section 3.4 will outline the research questions. The choice of research method to address the identified research issues is discussed in Section 3.6. Section 3.7 concludes the chapter.

3.1 Key concepts

This section defines the key terms and concepts which are used to formally define a problem in this thesis

3.1.1 Cloud computing

The term “cloud computing” refers to those computing resources (for example, hardware, development platform, and software) that are available on demand “as-a-service” over the web [71]. An end-user can access these web-based resources via an online subscription service.

3.1.2 Cloud services

Cloud services are cloud-based computing resources that are made available to the service consumers on demand through the internet by a cloud service provider. There are three cloud services models: Software-as-a-Service (SaaS), Platform-as-a-Service (PaaS), and Infrastructure-as-a-Service (IaaS).

3.1.3 Cloud service discovery

Cloud service discovery means finding the most reliable/trustworthy service offers available on the web that match the end-users’ needs.

3.1.4 Cloud services consumer

A cloud service consumer is an individual or an organization who/that use cloud services delivered by cloud service providers.

3.1.5 Cloud service provider

A cloud service provider is an entity or organization that offers cloud computing services to other individuals or businesses.

3.1.6 Cloud service advertisements

Cloud service advertisements are a text-based form of marketing that employ messages to promote or sell a cloud service. Cloud advertising is often taken up by sponsored cloud providers who wish to promote their products or services.

3.1.7 Web portal

A web portal is a website that provides a single entry point to access a variety of information and services from diverse sources, such as online forums.

3.1.8 Quality of Service (QoS)

Quality of Service (QoS) means monitoring and measuring the overall performance of the service such as the availability time of the service.

3.1.9 Quality of Experience (QoE)

Quality of Experience (QoE) specifies the overall performance of the service as perceived by the end-users.

3.1.10 Service ontology

Service ontology is a conceptual structure that represents a knowledge of the services in the form of concepts and relationships between service concepts.

3.1.11 Service annotation

Service annotation is the process of adding additional information to the service concepts.

3.1.12 Cloud services semantic marketing

Cloud services semantic marketing is semantically describe cloud services advertisement and marketing phrase using Web Ontology Language (OWL).

3.2 Gaps in the literature

In this chapter, the gaps and shortcomings of existing cloud service discovery approaches that were discovered in Chapter 2 are discussed. These are listed below:

1. None of the existing approaches propose an intelligent or user-friendly method for harvesting cloud services data from the web. These approaches do not take into account the heterogeneous issue of cloud service description in the web portals, or the heterogeneity in the website structures.

2. None of the reviewed works provide an intelligent method with which to engineer cloud services ontology grounded on the web data to deal with the heterogeneous cloud data in the web environment. Such as ontology is needed to enable search of cloud services data.

3. All existing ontologies used to describe cloud services are concerned with technical, operational or functional aspects. These ontologies neglect the general information of the service such as title, features, and price model.

4. There is no method that takes into account constructing a knowledge repository for cloud services that is based on real cloud services data.

5. None of the current studies has considered valuing the service quality based on the consumers' satisfaction. Analyzing consumers' posted reviews could be a useful method to fill the gap and give an indication about the real values of service quality based on consumers' experiences.

3.3 Research overview and problem definition

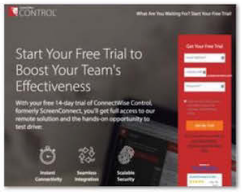
Nowadays, the number of cloud providers (Microsoft Azure, AWS, Google, and so forth.) has increased, thus leading to an increase in the number of cloud services offered online. Each provider is, however, using different marketing platform and techniques to publish their service on the web. Meanwhile, the discovery of cloud services on the web is a challenge for potential consumers. There is, therefore, a need for a solution that could assist consumers in discovering cloud services (SaaS, PaaS, and IaaS) across different web platforms.

As discussed in Chapter 2, several researchers have proposed different approaches for cloud services discovery using multiple technologies such as directories, crawler and semantic techniques [35]. The first approach proposed in the current literature to address this issue was manually maintaining lists of collected information about cloud services providers [38]. The service description in the UDDI is usually written using Web Service Description Language (WSDL), and is commonly used to describe web services such as the input-output parameter. UDDI has, however, failed to gain broad adoption [44]. This is because it is concerned with technical, operational and aspects, and neglects the general information of the service (for example, title, features and price model).

There are four common approaches to cloud services discovery: 1) using web crawlers to create cloud service listings; 2) using semantic technologies for cloud service discovery; 3) integrating agents with semantic technologies for cloud services description and discovery; and 4) combining the above-mentioned approaches. The third approach is the most commonly used approach at this point in time. Some recent studies [22] has focused on semantic searches, while others have focused on semantic annotation [93]. The proposed cloud services ontologies in these studies focused technical and operational aspects only.

The literature review undertaken in Chapter 2 shows that much attention has been given to the field of semantic technologies. However, it also points out that researchers have used various existing ontologies to represent cloud services such as Business ontology [31], which is more likely to support the business and operational aspects. Also, some other researchers have suggested defining the ontology concepts based on the NIST definition and terminology, which classify the cloud services into three delivery models (SaaS, PaaS, and IaaS). Although these are significant approaches for adding semantic meaning to cloud services, the issue of cloud services discovery is still unsolved. There is a need for an ontology that is grounded on web data which takes into account representing the general description of cloud service such as title, price model, and features.

As discussed in Chapter 2, some research has been undertaken in the areas of using UDDI for cloud services discovery and using the WSDL language to represent cloud services, which focus on functional aspects of cloud services (for example, input parameter, output parameter, and service name) [35]. WSDL has been developed to define and represent the web services and it only supports functional description of web services and lacks of non-functional parameters



View Website

ConnectWise Control

ConnectWise Control's remote support solution allows you to remotely view & control devices from anywhere. If you support personal computers or an enterprise, the ability to quickly resolve problems allows you to save time & improve customer satisfaction.

Be the first to review this company

Figure 3.1: ConnectWise a cloud-based service profile in Serchen.com [20]

ConnectWise UK overview

ConnectWise provides a business management and process automation software for technology companies such as cloud service providers, IT service companies, software developers, professional

READ MORE

Pricing

Pricing options

Subscription

Value for money

VIEW PRICING PLANS

Devices

Business size S M L

Markets

Australia, Canada, Europe, United Kingdom, United States

Supported languages

English

Figure 3.2: ConnectWise a cloud-based service profile in GetApp.com [5]

description such as service description, service price model, QoS, service features, and so forth. Furthermore, each cloud provider is uses different cloud service terms and concepts to describe the service they offer. Therefore, the consumer who does not have enough domain knowledge faces problem when trying to choose the service that best matches their needs. For example, as shown in 6.1 and 6.2, ConnectWise (a form of cloud-based service content that is advertised in GetApp) is different different from ConnectWise (which is advertised content in Serchen).

None of the research approaches, however, have taken into account the diversity in the service descriptions (such as in service description and service category) which could result in ambiguities during service discovery process. The lack of a cloud service registry which provides a listings of the general information of cloud services (title, price model, features, and so forth) is one of the main limitations of the cloud marketplace. Such a repository could greatly assist consumers

in locating cloud services. Instead, consumers are currently using general search engines to locate services . A number of private repositories has been made available by cloud providers such as Microsoft, but they are not publicly accessible. Rather, these registries are available for consumers belonging to given vendors only. Because web portals have different structures, most of the proposed crawlers need to customized for each web site which is a time consuming process. Therefore, there is a need for an intelligent method for harvesting cloud data that takes into account the heterogeneity in website structures.

Also, as discussed in Chapter 2, the main reason for developing crawler-based approaches is to provide a listing of cloud services; nevertheless, none of studies provide a complete cloud services dataset that has information of cloud services along with QoS dataset. All studies on QoS-based services selection have been validated by using either demo data or web services dataset. Therefore, the lack of the cloud services dataset is a limitation of the current literature.

QoS is a key component of service offers and it specifies the overall performance of the service. QoS assists the consumer, who is looking for a service with specific requirements. Also, QoS is essential for building trust between service providers and service consumers. QoS can be categorized by two types of parameters: functional and non-functional requirements. As mentioned in Chapter 2 that cloud service offer description lack non- functional QoS parameters such as the availability of the service. Such information is useful and can assist the consumer when they are selecting a service. However, none of the existing studies consider the problem of incomplete QoS information of cloud services. Although there are some web portals for cloud services that provide reviews, there is still a lack of QoS information (such as availability, and reliability) on these portals. Many scholars focus on proposing and developing intelligent methods to calculate QoS. Again, though, these methods assume the availability of relevant and complete QoS information from providers. There is an absence of studies that focus on methods to provide QoS information of cloud services in scenarios where it is incomplete.

Conversely, cloud customers are not only using cloud services online; they are also posting reviews across different web portals. This user-generated content can be very useful when trying to understand the consumers' experience, and it indicates that the consumers will continue using specific service. Also, the potential consumer could be influenced by others experiences and select the service that has received highly positive feedback.

Furthermore, the online reviews are helpful in the sense that they are communicating how the consumers themselves have experienced the services that they have used. In most cases, the online reviews are reflecting the QoS that the service provider offers. Therefore, evaluating the prior consumer experience by analyzing consumers' reviews could be useful for rating the service quality. The QoE is known as the overall performance of the service perceived by the end-users after using the the service. The QoE can be an indicator for the actual value of the QoS; and it can assist the service providers in improving their services. Unfortunately, none of the existing research studies consider providing QoE based on users' experiences.

Finally, to the best of our knowledge, there is no methodology in the literature that takes into account the above-mentioned facets. Based on the above overview and the description of the problem, we formally define the problem that we intend to address in this theses as follows:

"How to develop a cloud service marketplace registry that can assist the consumer in finding the most reliable service advertisement of cloud services?"

The broad question provided above can be broken down into four specific research questions. These questions are discussed in the next section.

3.4 Research questions

This section outlines the research questions which are addressed in this thesis in order to achieve the objective mentioned in chapter 1. The research questions are as follows:

- **Research question 1: How can an intelligent crawler be engineered to collect information from heterogeneous cloud services sources ?**

  There are a number of open source web harvesting platforms especially designed for harvesting the unstructured data from the web. However, all the current methods need to be customized for each website to harvest the web data. Therefore, there is a need for an intelligent method by which the the administrator of the registry or end-user can extract the customized data from any source in the web without the need for customizing the crawler. Therefore, to answer this question, we propose an intelligent method for capturing and harvesting web data that is related to the cloud services; and web data that is related to the cloud services online reviews across multiple web portals. This proposed method considers the heterogeneity of cloud services description in web portals, and the heterogeneity in the websites structure in the harvesting process

- **Research question 2: How can a reliable cloud services registry be built on collected cloud services information ?**

  Once an intelligent crawler has harvested the data from different web sources, it is important to store all the harvested data in the form of a registry. Depending on the web data collected, we are structuring the cloud services knowledge base as a registry.

- **Research question 3: How intelligent business decisions be made, based on the collected cloud services post reviews and cloud services registry ?**

  In our proposed method, we use the sentiment analysis approach to analyze reviews

related to cloud services in order to evaluate consumers' satisfaction. The sentiment in a review can be a good indicator of the consumers' evaluation or assessment. Furthermore, we build a cloud review classifier using different machine learning methods in order to predict the sentiment of cloud reviews in future. Our method has the ability to predict the overall sentiment of given reviews, which in turn can be used to determine QoS.

- **Research question 4: How can the proposed approach be evaluated and validated**

  We need to validate the solution proposed for research question 1 to research question 2. By "validation", we mean building a representation of a prototype system that is based on the proposed methodology. This will allow us to verify the soundness of the proposed methodology. In order to validate the methodology we use a prototype approach. In Chapter 4, we present an overview of the solutions to the research questions. In Chapters 5, 6, and 7 respectively, we present each research solution in details, along with the prototype system used for the validation and evaluation of that proposed solution

3.5 Research objectives:

To address the above research question, the objectives of this thesis are defined as follows:

- **Research objective1:** To develop an intelligent crawler to collect information from heterogeneous cloud services sources.
- **Research objective 2:** To develop a reliable cloud services registry based on collected cloud services information.
- **Research objective 3:** To develop intelligent methods for determining the opinion of online consumers' reviews positive, negative or neutral; and based on that provide the overall Quality of Experience (QoE) of a cloud product/service on harvested reviews.
- **Research objective 4:** To validate the above-developed methods by building a prototype system.

3.6 Approach to problem solving

In order to propose solutions for the research questions that were listed in the previous section, we need to follow a scientific approach to ensure that our development methodology is scientifically based. Therefore, this section provides an overview of the existing scientifically based research methods, and provides the reasons for choosing a particular research methods.

3.6.1 Research methods

There are two information systems research methods: a) the science and engineering approach; and b) the social sciences approach [52]. The science and engineering approach is concerned with using the experimental or measurable information to gain a new knowledge [45], while According to [43], the science and engineering approach has three hierarchical levels. The first level is the conceptual level of creating new concepts or new ideas based on a thorough analysis and reviews of the existing literature. The second level is the Perceptual level, and is concerned with formulating, designing and implementing a new approach or a new method. The third level is the Practical level, and is concerned with carrying out testing and validation of the proposed plan or strategy by using laboratory testing or the real-world cases testing. The social sciences approach is concerned with using of interview and survey to obtain a new knowledge base on a systematic plan [33, 70].This approach has two primary methods. The first is the Quantitative method. This method is focused on the study of the social claims that work with measurable evidence and analysing the raw data using statistical models. The second method is the Qualitative method. This method is concerned with exploring hidden knowledge by using surveys, interviews and personal observations. Unlike the science and engineering approach, the social science approach is not about creating a new method or new device; it is more about understanding the social evidence [57]

This thesis focuses on developing a new methodology for cloud service discovery on the web. The current problems in the cloud services discovery need first to be defined. Following this, a new approach or a new idea for solving these problems needs to be proposed. Finally, an actual prototype system to evaluate the scientific concept needs to be implemented. Therefore, the science and engineering approach is the approach that we follow in this thesis.

3.6.2 Choice of science and engineering research method

The Science and Engineering research method has three levels, which are explained as follows:

Conceptual level: This level consists of four sub-processes: Literature review; Problem formulation; Definition of key concepts; and Conceptual solution.

Literature review: We began this research by conducting thorough reviews for the existing research studies in the field of cloud services discovery. Additionally, we scientifically analyzed the current approach that is used to identify the issues and gaps within current research (3.2). The result of this sub-process is a list of current research gaps that this thesis aims to fill.

Problem formulation: According to the research issues identified in the previous level, we formulated the research questions (3.4) and defined the research objectives (3.5). This thesis concerns with three recent issues in the field of cloud services discovery.

Definition of key concepts: After defining the research questions and research objectives, we identified the key terms and concepts that we use it in this thesis for the purpose of presenting the research problems in Chapter 3 and presenting research solutions in Chapter 4. By applying the

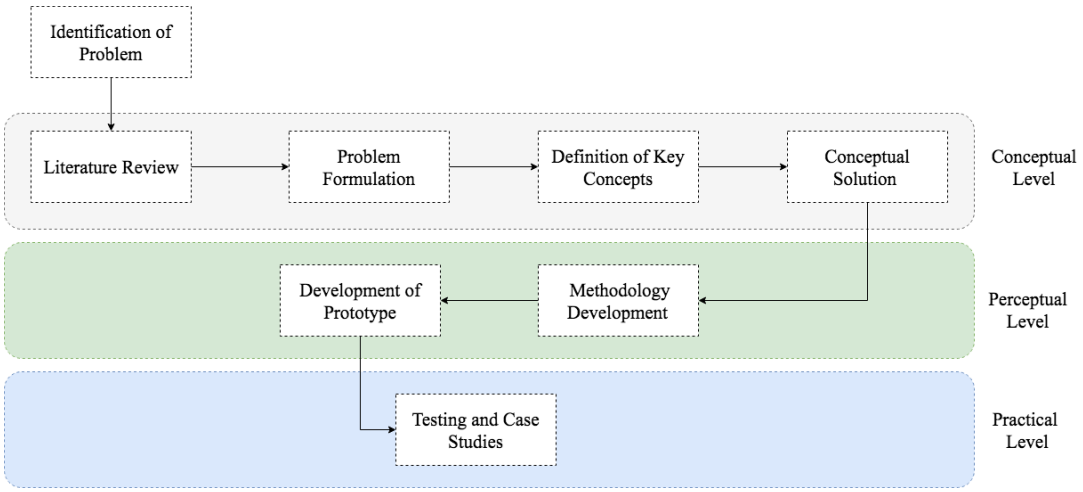


Figure 3.3: A science and engineering research method

sub-process in the conceptual level, we developed a new methodology for cloud service discovery, which is described in Chapter 4. This methodology aims to solve the defined research problems.

Perceptual level: This level is focused on developing the proposed solution in the conceptual level and implementing the actual prototype system. This system has the following two parts:

Methodology development: According to the conceptual level, the proposed methodology in this thesis is divided into three parts: service harvesting (Harvesting-as-a-Service), semantic service annotation, and service business intelligence (analyzing online reviews to obtain the QoE). Information about these methods are presented in Chapters 5, 6 and 7. *Development of Prototype:* According to the proposed methodology presented in Chapter 4, the prototypes of the solution for cloud service harvesting, cloud service annotation, cloud service intelligence were developed. For the validation and evaluation the proposed research methodology in this thesis, a case study was provided. In this thesis, AWS platform, Python, Java programming language, Web Ontology Language (OWL), Protege, PHP,.NET, SQL and MongoDB, Microsoft Excel, Rapid Miner are used to implement the methodology of cloud service discovery.

3.7 Conclusion

This chapter has discussed the research gaps related to cloud services discovery, and has defined the research questions that will be pursued in this thesis. The problems that this thesis investigates can be categorized into three sub-topics: service harvesting, semantic service annotation, and service business intelligence. We have proposed a solution for each of these research problems, thereby enabling the creation of the research methodology presented in Chapter 4. We have defined the fundamental concepts and terms that will be used to discuss and identify the research issues and research solutions. Finally, we have explained why we have chosen the science and

engineering research approach. The following chapter will provide an overview of the solutions proposed to solve the problem addressed in this chapter.

CHAPTER 4

RESEARCH SOLUTION

This chapter will provide an overview of the research methodology that will be used in this thesis. This methodology aims to provide a solution for the research questions that were addressed in Chapter 3, and provide an automatic derivation of the cloud marketplace. The design of this marketplace will include a combination of the following modules and stages: 1) Cloud Services Harvesting Module; 2) Cloud Services Knowledge base Module; and 3) Cloud Services Trust Derived Intelligence Module.

4.1 Overview of the solution for Cloud services discovery across heterogeneous web portals

In this section, we present an overview of the overall solution for cloud services discovery in web environment. We go on to present an overview of individual solutions for each research problems that were discussed in Chapter 3. A representation of overall solution is displayed in Fig 4.1. The core of overview solution is cloud services Knowledge base, which is a repository used to cloud services domain knowledge, and information in regards to cloud services offers, cloud services providers, and cloud services posted reviews. This knowledge base structured is based on the harvested web data using developed Harvesting Tool, namely the Harvesting-as-a-Service (HaaS) that is presented in Chapter 5. In Chapter 6, we introduce the framework for cloud services knowledge base for storing cloud services domain knowledge and cloud services advertisements information. In Chapter 7, we introduce the framework for deriving value of the quality of the cloud services based on reviews that have been posted by consumers. This framework is the Cloud Services Trust Derived Cloud Intelligence. Firstly, the Services Harvesting framework can used for collecting key information about cloud services from the web. Our developed solution

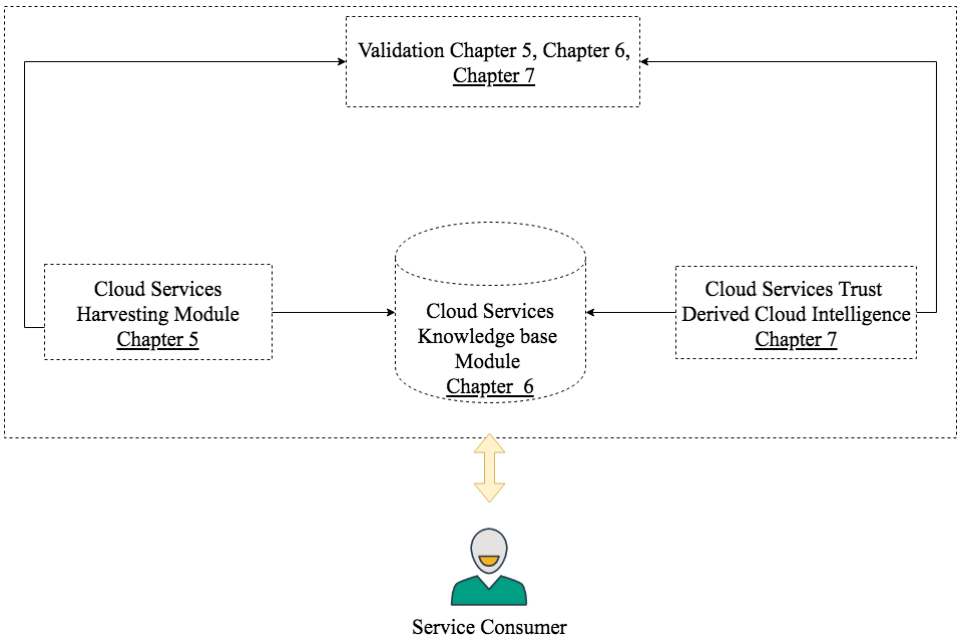


Figure 4.1: Overview of the solution for cloud services discovery across heterogeneous web portals

comprises the following:

1. A solution for developing an intelligent methodology as-a-Service to harvest cloud services from the web supported by an easy to use user-interface;

2. A solution for automatically harvesting heterogeneous cloud services information from heterogeneously structured web sources;

3. A solution for automatically organizing the harvested cloud services information; and providing this datasets containing real cloud services information and actual cloud services reviews;

4. A solution for constructing an open source platform for harvesting cloud services which integrates different types of cloud services information to construct a comprehensive listing of cloud services;

5. A solution for constructing a cloud services repository which could act as knowledge source for a common ontology for cloud services.

Secondly, the solution for services advertisements information annotation and classification (services knowledge base) is comprised of the following sub-solutions:

1. Developing a solution towards cloud marketplace in order to organize, publish and retrieve cloud services advertisements.

2. Semantically classifying cloud services advertisements to be grounded in harvested data from various web resources to solve the issue of heterogeneous cloud advertisements.

3. A solution for constructing the first commercial cloud services ontology-based repository for cloud services marketing. This repository contains service metadata that can be used to store the information of service advertisement that annotating to the cloud services domain ontology concepts toward retrieve cloud advertisements more efficiently.

4. Developing a solution for constructing a knowledge base that is grounded on harvesting service information from various web portals. This knowledge base acts as a knowledge source for the cloud services marketplace.

Thirdly, the solution for Services Trust Derived Cloud Intelligence comprises of the following sub-solutions:

1. A solution for the research community by providing the first polarity dataset based on analyzing real cloud services reviews, which is a very useful to train machine learning classifier.

2. A solution for predicting the sentiment of cloud reviews using machine learning classifier.

Additionally, we present a solution for validation and evaluation for all solutions stated above in Chapters 5, 6 and 7 respectively. In the next section, we present the overview of each of these solutions.

4.2 Overview of the solution for Cloud services harvesting in web environment

In Chapter 3, it was pointed out that, in the current literature, there is no methodology for harvesting cloud services across various web portals that considers the heterogeneity in the descriptions of service advertisements. Furthermore, as stated in Chapter 2, current approaches will more likely be focused on using semantic technologies to retrieve cloud service information, which usually leads to retrieve irrelevant details. Finally, the none of current studies have considered how to construct the cloud services dataset using a real cloud services information. To address these gaps, in Chapter 5, we provide an intelligent method for harvesting cloud service as-a-Service. This method takes into consideration the heterogeneity in the web structure, as well as the diversity in service advertisements across several web portals online. As shown in Figure 4.2, this is the first module in our research methodology, namely Harvesting cloud services. The working process of the proposed module is as follows:

1. **Step 1 Configure the initial URL web page to be visited:** The end users enter the initial URL of the web page to be visited, and they harvest the information.

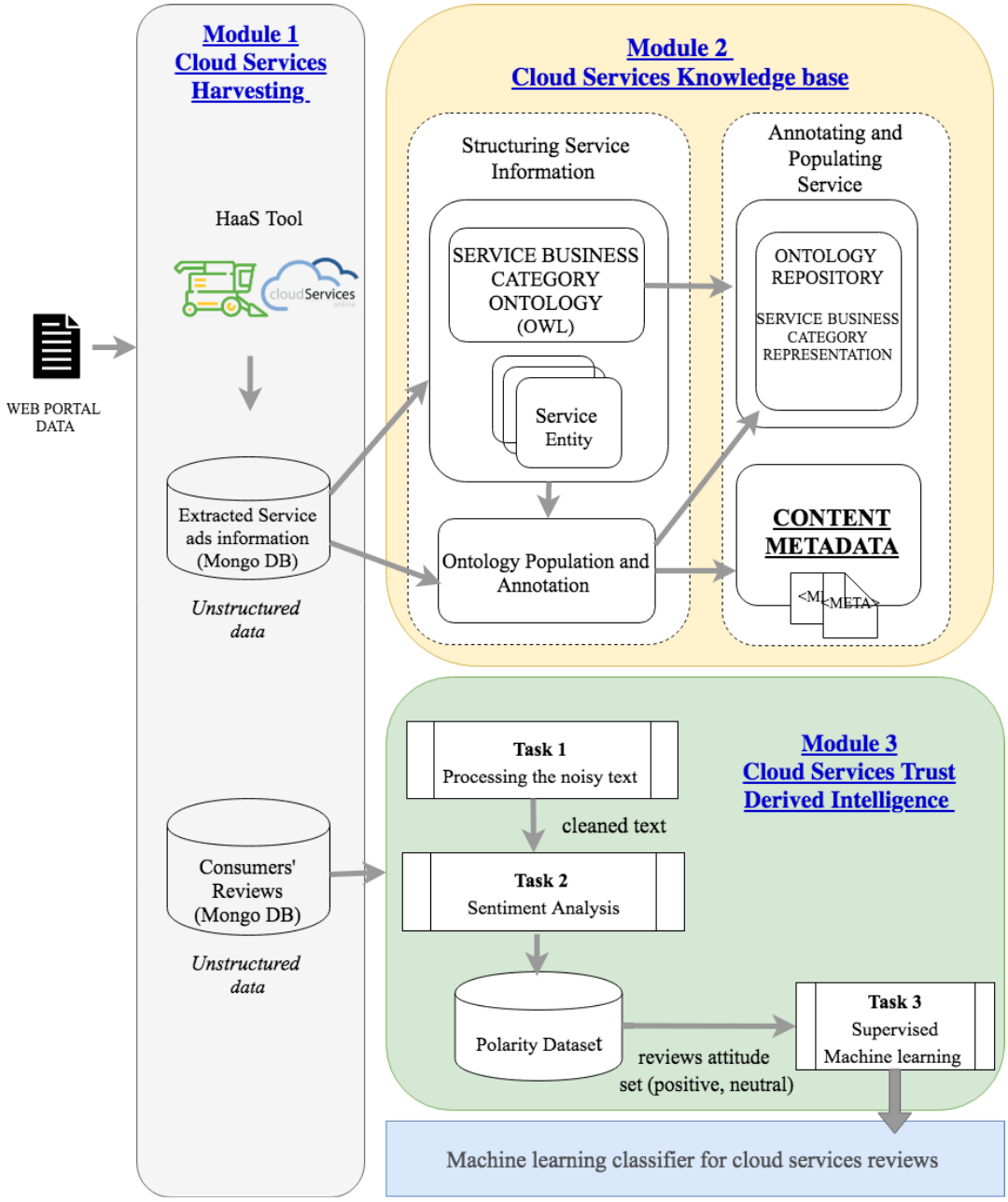


Figure 4.2: Research Methodology Framework

2. **Step 2 Intelligently Learn the structure of the web page:** Once the configuration task has been completed, the configuration details are sent to the intelligent agent for harvesting. The function of the intelligent agent is to learn the structure of web page given in the configuration task using learning algorithm. The outcomes from the intelligent agent is a file containing harvesting commands that obtain details of web page structure. This file is sent to the intelligent harvester.

3. **Step 3 Harvesting the sample web page:** Once the harvester has received the file harvesting commands, it starts harvest the web pages. The outcome of this task is the sample of harvested data.

4. **Step 4 Result displayed to the end-user for data validation:** In this step, once harvesting the above sample has been completed, the result is displayed to the end-user for the data validation and modification. The end-user verification aims to ensure that the sample data is correct and complete.

5. **Step 5 Remove redundant data:** In this step, once the validation task has been completed, the file is sent to the Optimizer tool, which will remove all redundant data.

6. **Provide the harvester with complete URLs:** Once the sample file is ready, the end-user provides the system with complete set of URLs for harvesting.

7. **Step 6 Harvesting:** Finally, the harvester starts harvesting the target web pages by taking into consideration the polite harvesting. This process is explained in greater detail in Chapter 5.

4.3 Overview of the solution for Cloud service knowledge base

In Chapter 2 and Chapter 3, it was pointed out in research gaps that, in existing literature, all existing ontologies used to describe cloud services concern with technical, operational or functional aspects and neglect the general information of the service such as title, features, price model, etc. Additionally, there is a lack of domain knowledge grounded on the web that classifies cloud services. Finally, it was pointed that there is no methodology proposed to engineer cloud services ontology grounded on the web data to deal with the various cloud data in the web environment.

In order to address this research gaps:

1. In Chapter 6, we propose a framework for cloud services knowledge (Module 2), which is used to store service advertisements information and service domain knowledge. With the two objectives stated above, we divide the service knowledge into two parts: 1) service meta-data and 2) service ontology. The service meta-data stores meaningful information about service advertisements data harvested. This meta-data is represented by the service entity, which describes the general information about cloud services. The service ontology is the representation of the service knowledge with regards taxonomy in a specific service domain, and is concerned with representing the abstract concepts and their relationships from that domain. In this study, we focus on one service domain, namely Software-as-a-Service (SaaS). Our proposed methodology can be a solution for service discovery in all cloud services domains. Examples of service concepts in SaaS domain are management and

customer management. These can be viewed as two concepts in the SaaS domain, with the former being the the sub-class of the later.

2. In Chapter 6, we propose a user interface for the cloud services knowledge base that is proposed above. This interface will discover and retrieve service offers. The working process of the proposed search module is as follows:

   a. End-user enters can choose from the tree view displays, which represents the general categories and concepts of the SaaS enterprise.

   b. For each category concept, there are other levels of sub-categories. Some of these levels have one level, while others have up to three levels.

   c. If a bottom level concept of a service ontology is finally determined by the consumer, the service entity metadata associated with the concept is displayed to the end-user.

3. In Chapter 6, we present all the tasks involved in the cloud services Knowledge base module in order to classify and annotate the service advertisements.

4.4 Overview of the solution for services trust derived Cloud intelligence

The previous chapter established that the existing literature in cloud services discovery contains an absence of studies that focus on methods to provide QoS information in scenarios where it is incomplete. Additionally, cloud customers are not only using cloud services online; they are also posting reviews across different web portals. These posted reviews are helpful in the sense that they are communicating the actual value of service from consumers' perspective. In most cases, the posted reviews are reflecting the QoS that the service provider offers. However, none of the existing studies considers the importance of analyzing reviews that have been posted by consumers in determining the real value of the QoS. To address this issue, in Chapter 7, we propose a QoE method (Module 3), which uses a harvesting tool to harvest the consumers posted reviews across various web review portals. We then apply the sentiment analysis to determine the emotional tone of certain words that have been used by reviewers. The aim of this step is to gain an understanding of the consumer attitudes, opinions and emotions expressed in the reviews. The outcome of this analysis will be label each review as being either "positive", "negative", or "neutral". We need this dataset as the training data for building cloud reviews classifier using supervised machine learning. Further information about this model is provided in Chapter 7.

4.5 Overview of the solution for the validation of proposed methodology

In this thesis, we make use of the evaluation method of prototype and functional testing to validate and evaluate the proposed research methodology for cloud service discovery on the web. Specifically, in relation to the three solutions above (section 4.2 and section 4.3), we propose the following evaluation process:

1. Module 1 - Cloud Services Harvesting: The objective of this module is to harvest cloud services across various web portals. In doing this, we will take into account the heterogeneity in the structure, as well as the diversity in service descriptions on the web. We build a prototype for service harvester, namely Harvesting-as-a-Service (HaaS) and run the prototypes in three web portals that are each structured differently to one another. Subsequently, we test and validate and the harvester function. The objective of the functional testing is to verify the functionality, performance and reliability of harvested data. For the evaluation, we have compare the HaaS platform with Scrapy [17], which is an open source application platform for harvesting structured data from the web. We will compare the HaaS platform and the Scrapy platform using multiple dimensions, such as performance user interface, and output files. To thoroughly and objectively evaluate the performance of the HaaS platform, we will employ the following indicators: harvesting time, and data quality.

2. Module 2 - Cloud Services Knowledge base: We will validate this module by implementing a prototype which consists of two sub-processes similar to the conceptual design that is described in Figure 4.2: structuring service information and annotating and populating service.

3. Module 3 - Cloud Services Trust Derived Intelligence: The objective of this module is to build a cloud reviews classifier using a supervised machine learning method. For implementation, we used two forms of open softwares: Knime [10] and Rapid Miner [11]. To validate and evaluate the cloud reviews classifier, we have used well know parameters, accuracy, recall, precision. Also, we have applied 3-fold, 5 fold, and 10 fold cross validation.

4.6 Conclusion

This chapter has presented an overview of the solutions to the research problems that are being addressed in this thesis. This chapter has also provided an overview of the proposed solutions to the four research issues that were identified in Chapter 3. In the following chapter, we will present in details the fundamentally of the propose Module 1 cloud services harvesting, which

has been identified in this chapter as being an important part of the cloud service discovery methodology that is used in this thesis.

CHAPTER 5

HARVESTING-AS-A-SERVICE (HAAS): A FRAMEWORK AND SOFTWARE FOR HARVESTING ENTERPRISE CLOUD SERVICES

5.1 Introduction

Cloud Services Discovery (CSD) is emerging as a new trend for service discovery across distributed and heterogeneous environments online. It is a process for locating a cloud service that best matches the end-user's requirements [87]. Since the emergence of cloud technologies, cloud providers have provided their service offers online through their official websites, and end-users are making use of general search engines such as Bing and Google to discover cloud services [47]. However, the cloud consumers may get lost among the massive number of possibly irrelevant search results. As mentioned in the literature reviews chapter 2, a number of authors have recently attempted to provide a solution for cloud services discovery.

In [24] reviewed the literature on the discovery of cloud services and found that most of the studies used semantic web technologies for the dynamic discovery of cloud services. The studies reviewed in this work recommended an ontology-driven for cloud services discovery in the web and all of the studies are using existing ontologies, such as business ontology to semantically describe cloud services functions and improve queries precision. Similarly, [74] suggested adding semantic annotation to cloud services profile online to automate the discovery of cloud services. The objective of using semantic annotation is to allow search engines (such as Google) to semantically identify and retrieve service information based on user's objectives [47]. A key issue with the semantic-based approach is that the semantic search could vary depending on the ontology domain and terminologies covered [30]. Constructing an ontology which contains all relevant domain concepts, such as service classification, service type, etc., is not an easy task, given the fact that cloud providers use different terminologies and vocabularies to describe their

services' offers, even though they have the same features [95]. On the other hand, other studies such as [56] suggested that using a multi-agent based approach protocol could enhance the process of cloud services discovery, however these studies fail to incorporate Quality of Services information.

From the above discussion, we note that the researchers' have spent a great effort on enhancing keyword-based search engines with annotation, or developing semantic-based systems. Also, they have proposed the agent-based approach which is still in the conceptual phase without enough practical applications in the real environment. The shortcomings of the current studies are summarized as follows:

1. they do not provide any intelligent method to enhance cloud services information retrieval across heterogeneously structured Web portals;
2. they do not propose a means for generating cloud services datasets and metadata;
3. they suggested reuse of existing ontologies such as business ontology to describe the service functions and characteristics;
4. they do not propose extracting a cloud services knowledge from the Web data;
5. they do not provide a registry to categories, organize and publish cloud services.

To address the shortcomings, in this chapter we present an intelligent methodology to harvest data from the World Wide Web (WWW) to build a comprehensive listing of cloud services. Also, no dataset of cloud services (IaaS, PaaS SaaS) exists. Such a dataset could be used by potential cloud consumers for cloud services discovery and could be very useful for future research in cloud service selection, composition and recommender systems. Also, the developed datasets could be a knowledge source for designing a cloud services ontology. Our proposed method for harvesting cloud services across several Web portals, termed as Harvesting as a Service platform (HaaS). The objectives of this methodology include:

1. Developing an intelligent methodology "as a Service" to harvest cloud services data from the Web supported by an easy to use user-interface;
2. Automatically harvest heterogeneous cloud services information from heterogeneously structured web sources;
3. Automatically organizing the harvested cloud services information and provides dataset containing real cloud services information, and actual cloud services reviews;
4. Constructing an open source platform for harvesting cloud services information which integrates different types of cloud services information to construct a comprehensive listing of cloud services;

5. Constructing a cloud services repository which could act as a knowledge source for constructing a common ontology for cloud services in the future.

This chapter is organized as follows: Section 5.2 describes the proposed system architecture, Section 5.3 outlines the system workflow, Section 5.4 presents the system implementation, Section 5.5 conducts the system evaluation and presents the results and discussion; and conclusion are drawn in section 5.6.

5.2 Proposed System Architecture

In this section, we present the architecture of our Harvesting as a Service (HaaS) system, which harvests information about cloud services and their QoS across heterogeneous Web portals. The benefits of HaaS are as follows:

1. HaaS is the first service-based intelligent harvester that harvests cloud services information across heterogeneous Web portals over the WWW. By "intelligent", we mean that it has the ability to harvest heterogeneously structured websites without the need for coding.

2. HaaS is the first intelligent harvester that is supported by an easy to use user interface. No programming experience is required by users who wish to use the harvester to harvest cloud services portals.

In the next section, we present the architecture of the HaaS system for cloud service discovery, which is illustrated in Figure 2. Harvester as a Service (HaaS) consists of the following elements: Policy Centre, Configuration Manager, Intelligent Learning Agent, Intelligent Harvester Agent, Semi-structured Harvested Data, Harvested Data Optimizer and Cloud Services Repository. We explain the function of each component in the system architecture below:

1. **Policy Centre:** The Policy Centre provides the essential procedures for coordinating the data harvesting process. The initial policies set for the HaaS system are as follows:

    a. *Configuration Policy:* This policy delineates the boundary for carrying out the harvesting process by providing details of the structure of the data and the target data to be harvested. We set two configuration policy rules in the HaaS framework, as follows:

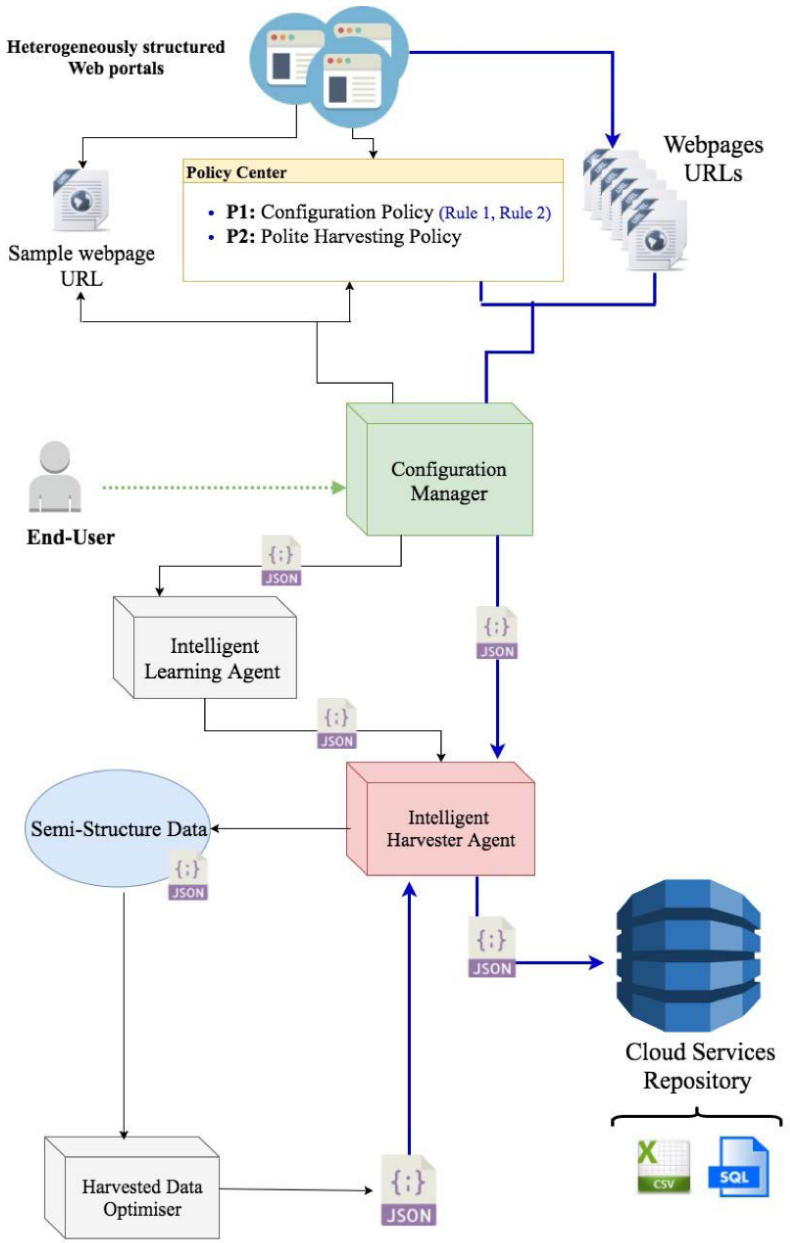


Figure 5.1: HaaS system architecture

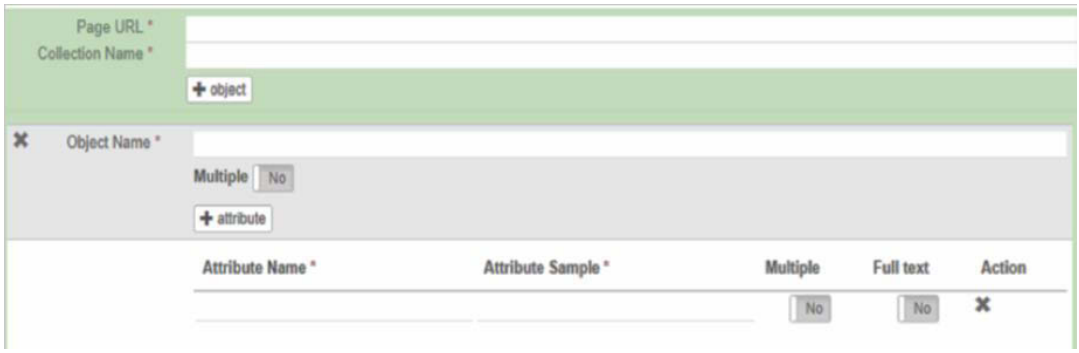


Figure 5.2: Details of HaaS configuration interface

- Rule 1: The objective of this policy is to choose a sample page from the target website, and then identify the data that need to be harvested from that sample page. In our scenario, we have a list of data attributes (custom attributes), for example, attribute name, attribute example, etc., as shown in Figure 1.
- Rule 2: The objective of this rule is to organize the related custom attributes of a custom object. The custom object can have one or many custom attributes. For example, an object called review has attributes that are related to consumers,Äô reviews, as shown in Figure 3.

b. *Polite Harvesting Policy:* This policy regulates the maximum time for each harvesting session, to avoid overloading the website being harvested. The harvesting process can be divided into multiple sessions, depending on the total number of related links provided. After each session, the process is paused for a certain period before continuing, so that the harvested portal is not overloaded. Figure 4 shows our two rules for crawling the website being harvested politely as follows:

- Records per session: Indicates how many links can be harvested per session.
- Waiting time interval: Indicates how many seconds to wait before moving on to the next session.

2. **Configuration Manager:** The Configuration Manager personalizes the harvesting process and provides essential information for collecting data. The harvesting process is comprised of two phases: phase one is the setup phase and phase two is the harvesting phase. Phase one is conducted in four steps, using the Configuration Manager user interface. During this setup phase, the Configuration Manager utilizes three levels of the configuration structure: the web page level, the custom object level, and the custom attribute level. The web page level provides a sample Uniform Resources Locator (URL) of a web page, while the custom object and the custom attribute levels define the data targeted for collection. To personalize the harvesting process, the Configuration Manager consults the Policy Centre to ensure defined rules/policies are followed during the procedure.

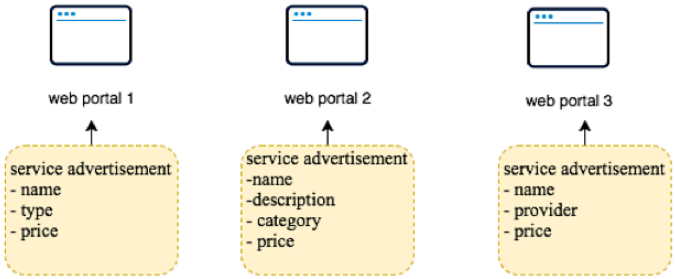
3. **Web Page Intelligent Harvester Agent:** The task of the Web Page Intelligent Harvester is to extract meaningful information about cloud services from the Web, as specified by the end-user in the data configuration step. To handle data heterogeneity when dealing with a large number of web pages, we define policies within the Policy Centre. The Intelligent Harvester Agent then carries out the harvesting process based on the defined policies.

4. **Intelligent Learning Agent:** The task of the Intelligent Learning Agent is to learn the web page layout/structure of the target Web portal. We propose an intelligent algorithm for learning the HTML structure of a web page, as shown in Figure 2.

5. **Semi-structured Harvested Data:** This module structures the incoming harvested information from the Web Page Intelligent Harvester. This information is structured as a JSON object, using the parameters specified by the user during the configuration phase. At this stage, the file includes harvested data with redundant configured attributes. The expected output file from the harvesting process is displayed in Figure 5.

6. **Harvested Data Optimizer:** The objective of this module is to remove redundant configured attributes from the harvested data. These attributes are useful for learning the true HTML structure of the sample web page. The redundant data can be removed after the learning process.

7. **Cloud Services Repository:** The Cloud Services Repository is designed to store the cloud services information collected and downloaded by the HaaS harvester in No SQL database in Mongo DB [21]. It contains all the cloud service information, including cloud service offer details such as service URL, service name, service type, service category, and details of consumers' reviews such as reviewer name and comments. Also, this information is made available to users as a CSV, PDF or SQL. Deploying the No SQL database would ensure coping with challenges of heterogeneous cloud service data, especially with growing number of services which would require more efficient and scalable database. The NoSQL database provides the flexibility to scale the database such as defining any new attribute or adding missing values as demonstrated in a figure 5.3. The NoSQL approach is more efficient in knowledge representation and provides scalability in modelling a knowledge base. Additionally, this study is not only just concerned with storing the heterogeneous cloud services data itself but also cared for sharing of the data, so for sharing the different cloud services data, a semantic knowledge base proposed in the next chapter [chapter 6]. We used a formal concept analysis in Mongo DB for knowledge construction and building the ontology.



No SQL Approach

	Name	Provider	Type	Description	category	price
Service 1	√		√			√
Service 2	√			√	√	√
Service 3	√	√				√
.....						

Figure 5.3: NoSQL approach

5.3 System workflow

Several components collaborate to realize the automatic discovery of cloud services information and construct the Cloud Services Repository, as described in Section 3. In this section, we present the workflow of the process in detail.

- **Step 1.** Before the *Intelligent Harvester Agent* starts to work, the end-user needs to configure the initial URL of Web pages to be visited (usually the URL of a Web page from the target website), and the harvesting data details in the Policy Centre. As previously mentioned, there are two levels of data configuration, the object level and the attribute level. At the attribute level, the end-user provides the system with the harvesting information details such as attribute name, attribute example, etc. (Figure 1). At the object level, the user groups the related attributes in a theme called *Object*. Once the configuration is complete, the *Configuration Manager* sends the configuration commands to the *Intelligent Learning Agent*.

- **Step 2.** *The Intelligent Learning Agent* runs the learning algorithm to learn the structure of the web page. On completion of the learning process, the result is generated as a JSON file and the *Configuration Manager* sends the file to the Intelligent Harvester Web page.

- **Step 3.** When *Intelligent Harvester Agent* receives the JSON file from the *Configuration Manager*, the harvester starts to harvest the webpage based on the commands in the JSON file. The JSON file holds the details of the data structure and sample data. The output of this step is the sample of harvested data.

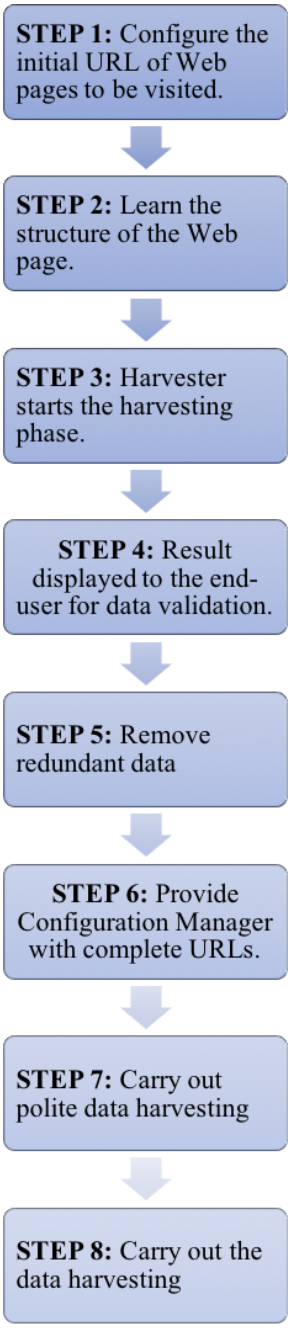
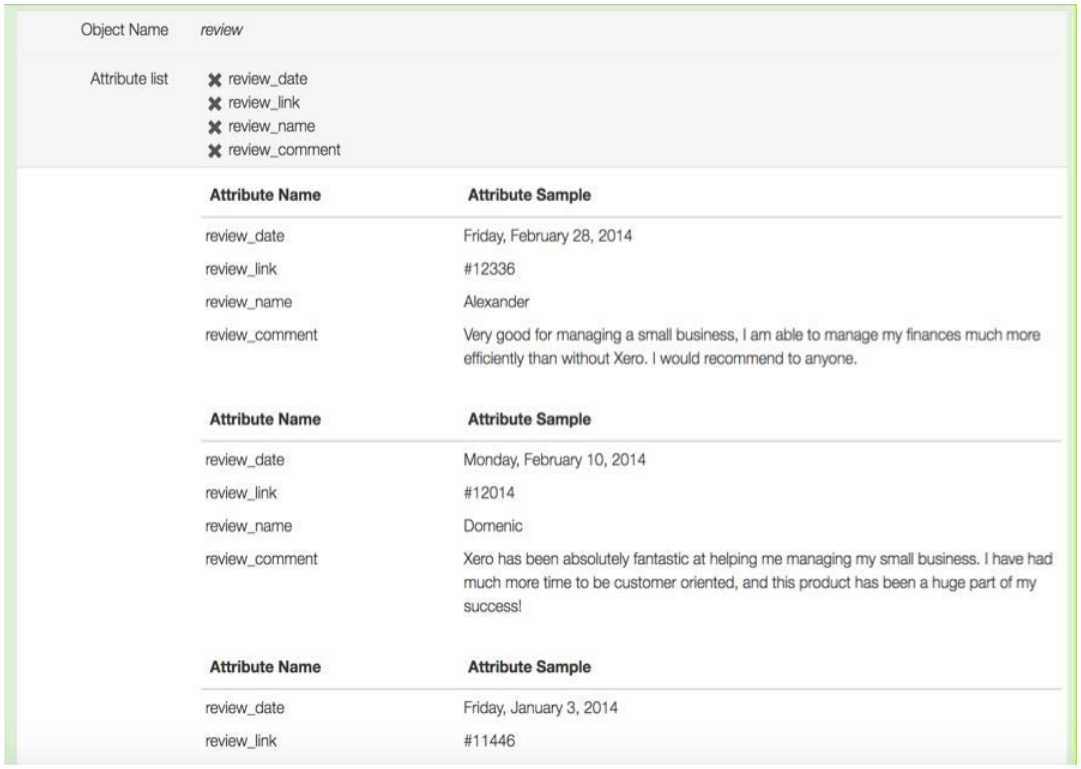


Figure 5.4: The steps of the HaaS harvesting process



Object Name *review*

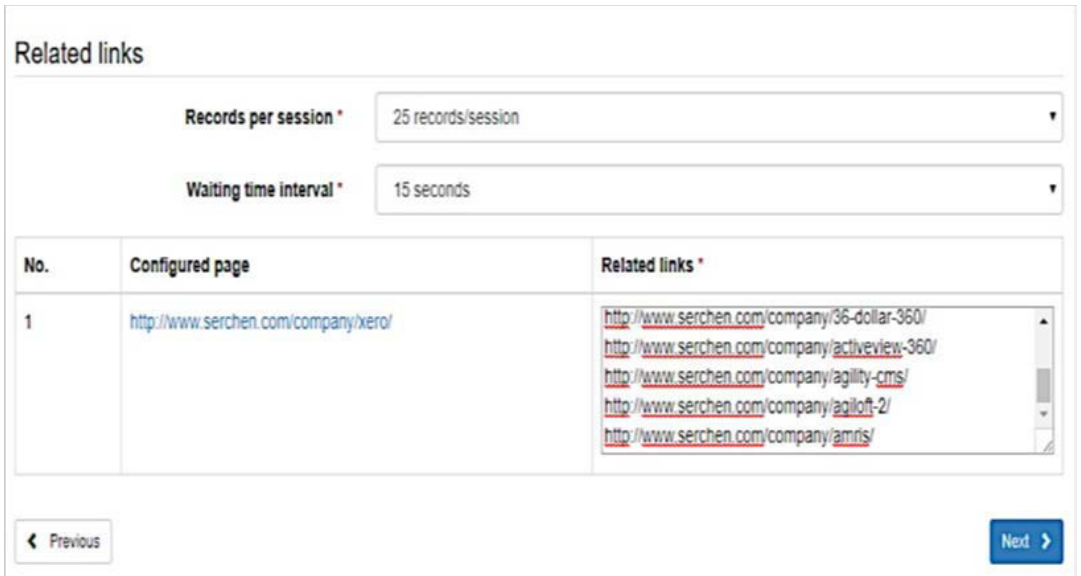
Attribute list
- review_date
- review_link
- review_name
- review_comment

Attribute Name	Attribute Sample
review_date	Friday, February 28, 2014
review_link	#12336
review_name	Alexander
review_comment	Very good for managing a small business, I am able to manage my finances much more efficiently than without Xero. I would recommend to anyone.

Attribute Name	Attribute Sample
review_date	Monday, February 10, 2014
review_link	#12014
review_name	Domenic
review_comment	Xero has been absolutely fantastic at helping me managing my small business. I have had much more time to be customer oriented, and this product has been a huge part of my success!

Attribute Name	Attribute Sample
review_date	Friday, January 3, 2014
review_link	#11446

Figure 5.5: A sample of harvested data



Related links

Records per session * 25 records/session

Waiting time interval * 15 seconds

No.	Configured page	Related links *
1	http://www.serchen.com/company/xero/	http://www.serchen.com/company/36-dollar-360/ http://www.serchen.com/company/activeview-360/ http://www.serchen.com/company/agility-cms/ http://www.serchen.com/company/agiloft-2/ http://www.serchen.com/company/amris/

Previous Next

Figure 5.6: Screenshot of the configuration set-up for serchen.com

```
{
   "configWithStructure":{
      "website_url":"http://www.serchen.com",
      "analyse_structure":"no",
      "pages":[
         {
            "has_structure":"1",
            "collection_name":"services_collection",
            "objects":[
               {
                  "multiple":"no",
                  "parent_tag":{
                     "position":0,
                     "name":"div",
                     "attributes":{
                        "class":"reviews-container"
                     }
                  },
                  "name":"detail",
                  "attributes":[
                     {
                        "expected_result":{
                           "text":"1",
                           "attributes":[

                           ]
                        },
                        "full_text":"yes",
                        "name":"company_name",
                        "filter_tag":{
                           "position":0,
                           "name":"span",
                           "attributes":{
```


Figure 5.7: Sample of JSON file result semi-structured data

- **Step 4.** Once harvesting of the sample Web page has been completed, the result is displayed to the end-user for data validation and modification. The end-user verification is to ensure that the sample data is correct and complete. Steps 1, 2 and 3 are a recursive process until the end-user-defined harvesting boundary has been reached.

- **Step 5.** Once the harvested data and attributes have been passed to the *Harvested Data Optimizer*, all redundant data will be removed and the result will be stored in the form of a JSON file. The output of this step is semi-structured data in JSON format based on end-user configuration in Step 1.

- **Step 6.** Once the sample file is ready, the end-user provides the *Configuration Manager* with the complete set of URLs for harvesting. The *Configuration Manager* sends the URLs and a sample JSON file from Step 5 to the *Intelligent Harvester Agent*

- **Step 7.** Before the *Intelligent Harvester Agent* starts working, the end-user sometimes needs to consider Rule 2 of the configuration policy with regard to crawling the data from

the web pages politely by defining the number of harvesting links per session and the required waiting time before moving to the next harvesting session.

- **Step 8.** When the *Intelligent Harvester Agent* receives the URLs and the sample *Semi-Structured Harvested Data* (JSON file) from the *Configuration Manager*, the *Intelligent Harvester Agent* starts to harvest data from the target URLs.

5.4 HaaS Implementation

The HaaS platform was developed on the AWS platform using Amazon Elastic Compute Cloud (Amazon EC2) [2], a web service that provides computing capacity in the cloud. In addition, we used the following underlying technologies for HaaS implementation: Python 3, Mongo DB, and BeautifulSoup4 library [73]. The HaaS user interface structure provides instructions to guide end-users through the harvesting process. End-users do not need to install or configure any applications locally because HaaS data management, software, and hardware is delivered from the cloud as a service to users. We implemented the HaaS framework for validation and evaluation as follows.

The HaaS Configuration Manager module is built during the first phase of constructing the HaaS platform. It assists the end-user to define the harvested data and then invokes the Intelligent Harvester. By “end-user”, we mean a cloud end-user such as a consumer, organization, or developer who has knowledge about offers of cloud services and where to find them over the WWW. In addition, the HaaS platform also contains a Policy Centre module to manage and control the harvesting process. With the help of the Policy Centre module, the Configuration Manager module is able to define the harvesting elements and customize the harvesting process according to the end-user’s objectives. As mentioned in Section 3, the Intelligent Harvester starts to harvest a sample of harvest data once the harvest elements have been properly defined.

Algorithm 1: Structure Algorithm

Input: Configuration data in JSON format following the structure:

- S = {*multiple*, *objects* = (o_1, o_2…o_n)} is the structure of the configuration, containing a sequence of configured objects, *multiple* ∈ {yes, no} (get one or multiple HTML data for objects of the same structure)
- o_i (i ∈ N) = {*attributes* = (a_1, a_2…a_m)} is a detail content of each object including a sequence of configured attributes.
- a_j (j ∈ {1, 2, 3…m}) = {*sample*, *multiple*, *fullText*}: *sample* (sample text for the attribute), *multiple* ∈ {yes, no} (get one or multiple HTML data of similar HTML sibling tags) and *fullText* ∈ {yes, no} (sample text is in full content or partial content).

Output: Metadata structure in JSON format with the core is the configuration data with extended information to navigate the position of attributes during the harvesting process. The output for objects and attributes is as follow:

- o_i (j ∈ N) = {*attributes* = (a_1, a_2…a_m), *parentTag* = {*position*, *tagName*, *className*, *idName*}}. *Position*: the position of o_i *tagName* in relation to the whole HTML document, *tagName*: HTML tag name, *className*: HTML class name, *idName*: HTML ID name.
- a_j = {*sample*, *multiple*, *fullText, filterTag* = {*position*, *tagName*, *className*, *idName*}}. *Position*: the position of the *tagName* in relation to the *parentTag* of the object containing a_j, *tagName*: HTML tag name, *className*: HTML class name, *idName*: HTML ID name.

Procedure: Begin Algorithm

For i = 1 to n
  Fetch the sample of the first attribute a_1 in o_i
  Compute HTML parent tag of a_1 and stores *parentTag* in o_i using BeautifulSoup
  For j = 1 to m
    If *parentTag* does not contain a_j then
      Compute HTML parent tag of the *parentTag* and stores *parentTag* in o_i using BeautifulSoup
      Repeat step 4
    End if
  End for
  For j = 1 to m
    Compute HTML tag of a_j based on computed *parentTag* in o_i and stores *filterTag* in a_j using BeautifulSoup
  End for
End for

End Algorithm

Figure 5.8: Structure algorithm

Algorithm 2: Harvest URLs algorithm

Input: The algorithm Harvest URLs has three types of input:

1. Metadata structure from the output of Algorithm 1: Structure algorithm
2. A sequence of URLs that have the same HTML structure with the configured Web page. URL = (url_1, url_2…url_p)
3. Polite harvesting parameters {*recordsPerSession*, *waitingTimeInterval*}. *recordsPerSession*: the number of URLs to be harvested over one session, *waitingTimeInterval*: the time (seconds) the harvest process will pause between each session.

Output: Dataset includes data of configured attributes for all inputted URLs. The dataset is stored in MongoDB and exported to CSV format.

Procedure: Compute the number of sessions NS based on the number of URLs and parameter *recordsPerSession*. NS = Number of URLs/ *recordsPerSession*
Set session to 1
While session <= NS then
  For k = 1 to p
    For i = 1 to n
      Compute all possible *parentTag* of o_i inside url_k using BeautifulSoup and store these tags into ts
      If *multiple* of o_i is "yes" then
        Repeat step 12 to step 18 for all *parentTag* inside ts
      Else then
        Perform step 12 to step 18 for the *parentTag* that has the position aligned with o_i position
      End if
      For j = 1 to m
        If *multiple* of a_j in o_i is "yes" then
          Parse the content of a_j for all similar HTML siblings based on *parentTag* and a_j *filterTag* (*tagName*, *className*, *idName*) using BeautifulSoup and store the data in MongoDB for url_k
        Else then
          Parse the content of a_j for the HTML tag based on *parentTag* and a_j *filterTag* (*position*, *tagName*, *className*, *idName*) using BeautifulSoup and store the data in MongoDB for url_k
        End if
      End for
    End for
  End for
  Pause the process based on *waitingTimeInterval*
  Increment session to 1
End while

End Algorithm

Figure 5.9: Harvest URLs algorithm

Most importantly, the sample of harvested data is displayed to the end-user for the data validation task, which is a significant advantage of our HaaS approach compared Without a data validation process, errors in the data could go undetected until all the data has been harvested. To counter this scenario, we propose a data validation process in which a sample of harvested data is presented to the user for verification or modification. If the user verifies the provided sample data, the harvesting process is executed and all the data are harvested. To enable the removal of redundant data from the harvested data, we deploy the Harvested Data Optimizer module, which removes all redundant data to support the harvesting process. Both the data verification task and data optimization task are implemented to guarantee the quality and reliability of the harvested information throughout the harvesting process.

The core of the HaaS system is the Intelligent Harvester module. As noted in Section 3, "intelligent" refers to a service-based harvester application supported by a user interface that has the ability to harvest Web pages with different structures. Therefore, by following the HaaS user interface instructions, end-users are able to harvest several Web portals without the need for developers. As stated in Section 1, we have chosen to harvest cloud services (IaaS, PaaS, and SaaS) advertised across the following three Web portals: CloudReviews, GetApp, and Serchen. Each Web portal has a different website structure, so we harvested each website individually and separately. The results are stored in MongoDB in JSON format, which can be converted to CSV format. The novelty of HaaS is threefold: (a) it can harvest cloud services information (including QoS information) from heterogeneously structured Web pages; (b) it has the ability to harvest several Web portals and collected data using the developed HaaS user interface without the need for developers; and (c) the data verification step and data optimization step can be utilized by end-users throughout the harvesting process to ensure data quality.

5.5 System Evaluation, Results and Discussion

The system evaluation is separated into three subtasks: 1) evaluating the whole HaaS framework; 2) analysis of case studies; and 3) comparing HaaS with similar existing system.

5.5.1 Prototype implementation and evaluation

A prototype of HaaS was constructed using the Python Beautiful Soup package [79] and Mongo DB. To evaluate the prototype, we executed the HaaS system to crawl Web portals with cloud services information, including IaaS, PaaS and SaaS. We crawled three Web portals: GetApp (`https://www.getapp.com/`), Serchen (`http://www.serchen.com/`) and CloudReviews (`https://www.cloudreviews.com`). The crawler was able to generate metadata of 17657 cloud services and 17337 cloud service reviews in total (Table 4). Both datasets of cloud services and cloud reviews are available online via our cloud service registry website, `http://cloudmarketregistry.com/cloud-market-registry/home.html`.

Table 5.1: Parameters of the HaaS testing prototype

Web portal	**No. of harvested cloud services meta-data**	**No. of harvested cloud reviews**
serchen.com	12511	11078
getapp.com	5146	6259
cloudreviews.com	149	-
Total	17806 services	17337 reviews

5.5.2 Case studies using GetApp, Serchen and cloud reviews

In this section, we present case studies for verification of our proposed framework, then discuss and compare our system with similar studies. As previously mentioned, we harvested cloud services information from three Web portals: getapp.com, serchen.com and cloudreviews.com. For verification purposes, we present the process of harvesting serchen.com, getapp.com and cloudreviews.com step by step in this section. To harvest serchen.com, the HaaS system takes the end-user through a number of steps, as follows:

- *Step 1: Configuration* This step takes users to the home page of the HaaS system, and the purpose of this step is to help users to configure what they want to harvest on their target Web portal. First, the user needs to investigate serchen.com thoroughly to understand the serchen.com sitemap and the layout of serchen.com web pages with repetitive HTML structure that they wish to focus on. One of the cloud service offer web pages will be selected as a sample page. To set up the configuration, users need to provide the following details on the configuration form, as shown in Figure 7:

  1. PAGE:
     a. *Website URL:* Original URL of the harvested website.
     b. *Page URL:* Sample page URL to obtain sample data. A page can have one or many objects.
  2. OBJECT:
     a. *Object Name:* Custom object name. An object can have one or many attributes.
     b. *Object Multiple:* This option indicates whether to harvest the same object multiple times
  3. ATTRIBUTE:
     a. *Attribute Name:* Custom attribute name.
     b. *Attribute Sample:* Attribute sample copied from the sample page. If the text is too long, partial text can be used.

c. *Multiple:* Indicate whether to harvest the same attribute multiple times (e.g. harvest multiple li in an ul tag).

d. *Full text:* Indicate whether the attribute sample is in full text or partial text (check HTML structure)

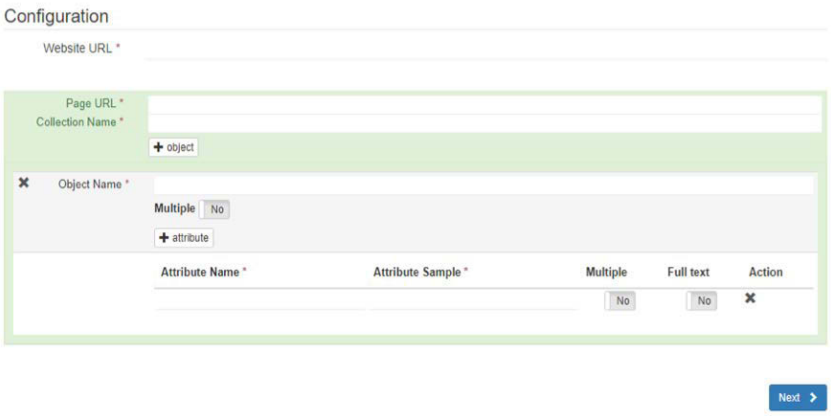


Figure 5.10: Screenshot of interface for Step 1: Configuration

The *Add Object button* (+ object) inside the page panel is used to add a new empty object to the Configuration page. The *Add Attribute button* (+ attribute) inside the Object panel for the collection of attributes is used to add a new empty attribute to the Object on the Configuration page. Users are able to remove Objects/Attributes using the delete buttons. Clicking the Next button takes users to *Step 2: View Sample*. The screenshot in Figure 8 shows the configuration setup screen for serchen.com based on sample pre-defined data from the system.

- *Step 2: View Sample* The purpose of this step is to learn the HTML structure of all configured objects and their attributes in the configured page. It also assists users to validate the results to ensure that the required data is sampled, per the user's request at *Step 1: Configuration.* If the data is not correct, users can move back to Step 1 using the Previous button to adjust their configuration of the error object/attribute. If this is the case, users simply click button to trigger the system to re-learn the structure and re-get the sample data. Removing redundant attributes to customize the harvested result is carried out in *Step 4: Start Harvesting.* Figure 9 is a screenshot of the first part of the result section interface. Users click the Next button to move to *Step 3: Related Links.*

- *Step 3: Related Links* This step assists users to copy the URLs of all web pages that have the same HTML structure as the configured sample web page from *Step 1: Configuration.* The cloud services offerings are located in a unique web page URL but the all pages have the same structure. The polite harvesting feature is implemented in the HaaS system with

constraints that include records per session and waiting time interval (refer to Section 3). In Figure 8 is a screenshot of Step 3 after the related links have been provided. Users click the Next button to move to the next step, *Start Harvesting*.

- *Step 4: Start Harvesting* is the final step, in which users only need to press the Start Harvesting button to start the harvesting process. The system stores harvested data in MongoDB with JSON syntax. When the harvesting process is finished, the system generates data in CSV file format and makes it available to the end-user. The web browser pops up another Tab that enables users to download the data in CSV file format (save file name as <filename>.csv). If automatic popups are blocked on the user's browser, the user needs to allow popups for the web tool and re-start harvesting to download the file. The structure of the exported file is as follows:

  a. All Attributes (A) belong to an Objects (O) of multiple values and assigned to No, they are combined and considered as columns in the top table (Main Table of output Datasets).

  b. Attributes (A) belong to the Objects (O) of multiple values and assign to Yes, they are combined and considered as a separate table inside the file with an empty row on top of the table. Figure13 displays the result file in CSV format.

To harvest the getapp.com Web portal, we followed the same steps. Figure 11 and Figure 12 are screenshots of the harvesting process of getapp.com

5.5.3 Comparison and discussion

To evaluate the features of the HaaS platform, we compare it with the Scrapy platform (Table 5). Scrapy is an open source application platform for harvesting structured data from the Web. A comparison of the HaaS platform with Scrapy along multiple dimensions is shown in Table 5. This table demonstrates how different dimensions of the HaaS platform and the Scrapy platform compare, such as performance, user interface, output files, etc. The results of this comparison show that there are a number of similarities between HaaS and Scrapy; for example, both have the polite harvesting feature, and both output files in CSV and JSON format. One of the most important features of HaaS is the user interface, which enables end-users to easily employ the harvesting process without the need for coding. End-users follow the HaaS user interface instructions to harvest the data from the target Web portals, whereas the Scrapy crawler has to be coded and customized for each target website. In addition, the HaaS platform is cloud-based and provided as a Service to the end-user. To thoroughly and objectively evaluate the performance of the HaaS system, we employ the following two indicators, *Harvesting Time* and *Harvested Data Quality*, to compare HaaS with Scrapy.

Figure 5.11: Screenshot of Step 1: Configuration interface for serchen.com

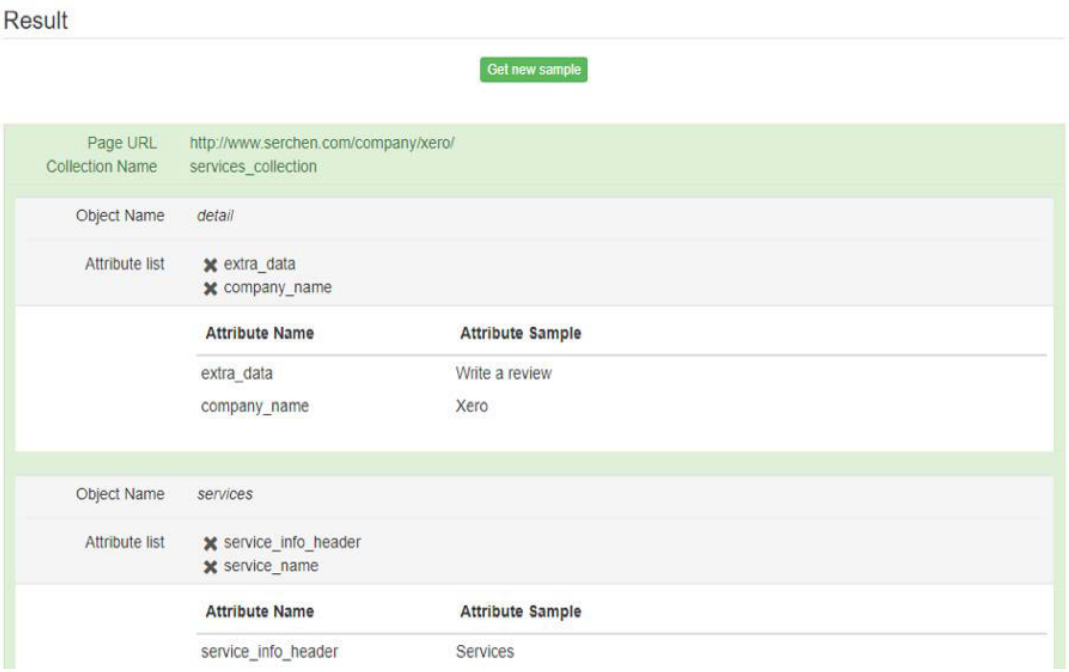


Figure 5.12: Sample of data harvested from serchen.com

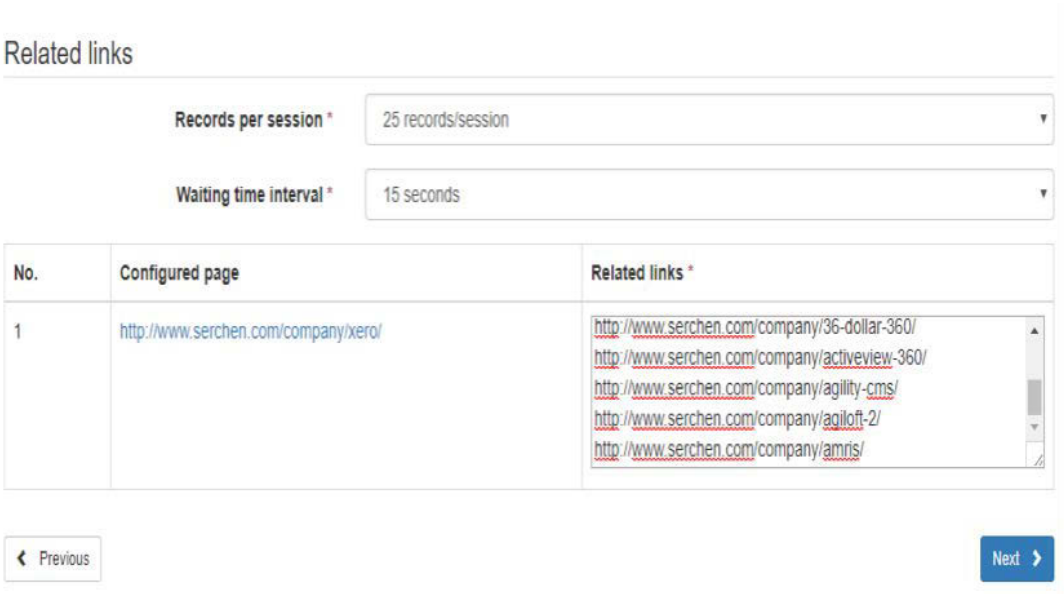


Figure 5.13: Screenshot of Step 3: Related Links showing links from serchen.com

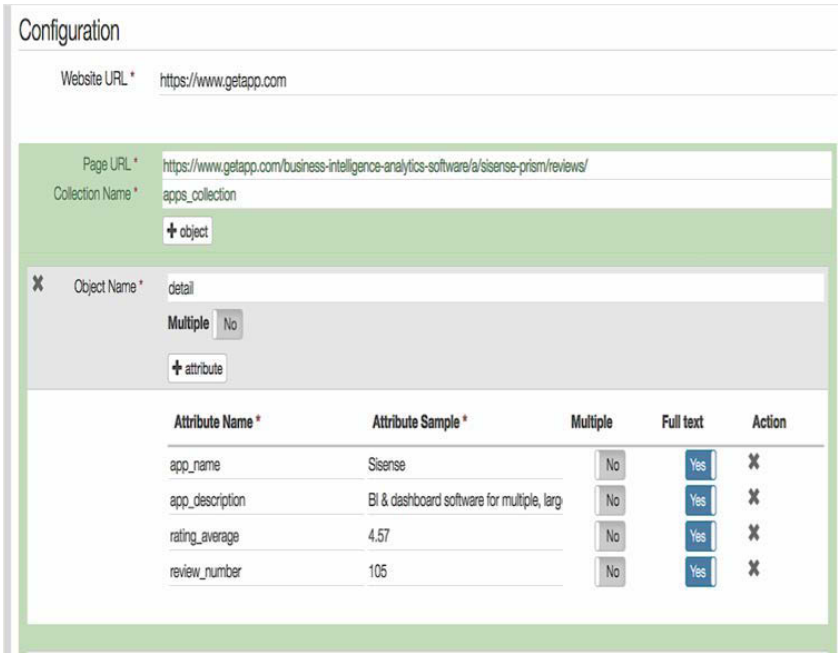


Figure 5.14: Screenshot of Step 1: Configuration interface for getapp.com

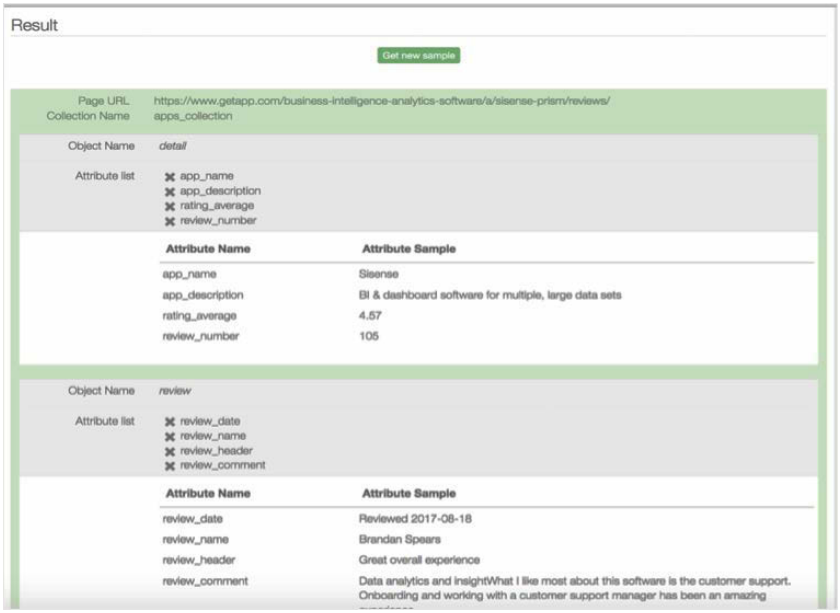
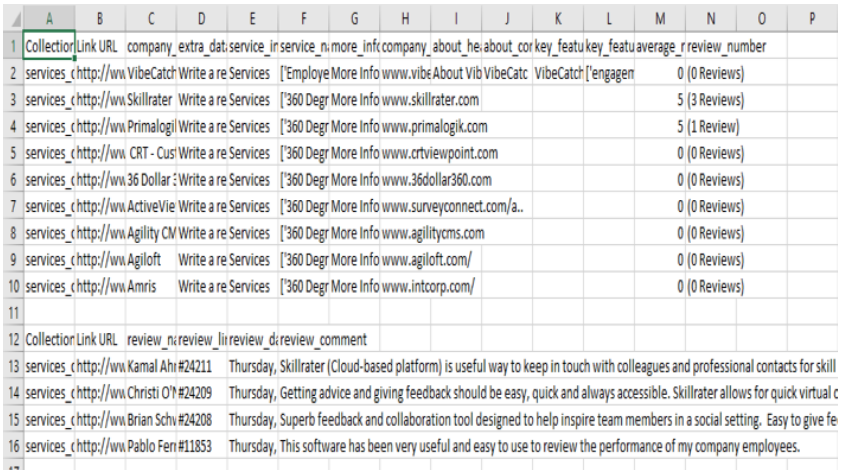


Figure 5.15: Sample of data harvested from getapp.com



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	Collection	Link URL	company_	extra_dat	service_in	service_n	more_infc	company_	about_he	about_cor	key_featu	key_featu	average_r	review_number		
2	services_c	http://ww	VibeCatch	Write a re	Services	['Employe	More Info	www.vibe	About Vib	VibeCatc	VibeCatch	['engagen	0	(0 Reviews)		
3	services_c	http://ww	Skillrater	Write a re	Services	['360 Degr	More Info	www.skillrater.com					5	(3 Reviews)		
4	services_c	http://ww	Primalogil	Write a re	Services	['360 Degr	More Info	www.primalogik.com					5	(1 Review)		
5	services_c	http://ww	CRT - Cus	Write a re	Services	['360 Degr	More Info	www.crtviewpoint.com					0	(0 Reviews)		
6	services_c	http://ww	36 Dollar :	Write a re	Services	['360 Degr	More Info	www.36dollar360.com					0	(0 Reviews)		
7	services_c	http://ww	ActiveVie	Write a re	Services	['360 Degr	More Info	www.surveyconnect.com/a..					0	(0 Reviews)		
8	services_c	http://ww	Agility CM	Write a re	Services	['360 Degr	More Info	www.agilitycms.com					0	(0 Reviews)		
9	services_c	http://ww	Agiloft	Write a re	Services	['360 Degr	More Info	www.agiloft.com/					0	(0 Reviews)		
10	services_c	http://ww	Amris	Write a re	Services	['360 Degr	More Info	www.intcorp.com/					0	(0 Reviews)		
11																
12	Collection	Link URL	review_n	review_lir	review_d	review_comment										
13	services_c	http://ww	Kamal Ahı	#24211	Thursday,	Skillrater (Cloud-based platform) is useful way to keep in touch with colleagues and professional contacts for skill										
14	services_c	http://ww	Christi O'I	#24209	Thursday,	Getting advice and giving feedback should be easy, quick and always accessible. Skillrater allows for quick virtual c										
15	services_c	http://ww	Brian Schv	#24208	Thursday,	Superb feedback and collaboration tool designed to help inspire team members in a social setting. Easy to give fe										
16	services_c	http://ww	Pablo Fern	#11853	Thursday,	This software has been very useful and easy to use to review the performance of my company employees.										

Figure 5.16: Sample of Harvested Data

a. Harvesting Time In this section, we examine the efficiency of the proposed HaaS system compared to the Scrapy in harvesting heterogeneously structured websites. We first harvested the serchen.com Web portal using both systems (HaaS and Scrapy) without applying the polite harvesting feature, and compared both tools in relation to crawl time. Tables 6, 7 and 8 show this comparison of harvesting time across three rounds of harvesting. To validate the harvesting results, we measured the percentage of change in the harvesting time as a function of the number of harvested services (20, 40, 60, 80 and 100). Polite harvesting was not applied in any of the three rounds. The harvesting results of serchen.com show that the proposed HaaS system performs better than Scrapy. The harvesting time usually depends on network bandwidth, CPU capacity at the time of running, server response time at the time of running, and the polite harvesting configuration for both HaaS and Scrapy. The graph in Figure 10 illustrates a significant difference between HaaS and Scrapy in harvesting serchen.com.

Table 5.2: Comparison of Scrapy and HaaS

Comparison Aspects	**Scrapy**	**HaaS**
Tool	Application Framework	Web Application
Programmer support	Need Programmer Intervention	it does not need programmer Intervention
Polite Harvesting	support polite harvesting	support polite harvesting
Ease of Installation	Need installation	Cloud-based (as a Service)
Output files and format	JSON, CSV and XML	JSON and CSV
User Interface	It does not has a user interface	It has a user interface

Table 5.3: Harvesting of HaaS and Scrapy as a function of the number of services - Serchen Round1

Serchen (1st round)		
Services	**HaaS**	**Scrapy**
20	3.093	2.306
40	8.472	17.388
60	10.992	21.346
80	15.514	17.602
100	19.103	43.101

Table 5.4: Harvesting of HaaS and Scrapy as a function of the number of services - Serchen Round2

Serchen (2nd round)		
Services	**HaaS**	**Scrapy**
20	3.064	3.003
40	8.292	10.481
60	11.147	18.89
80	14.482	29.803
100	18.053	29.022

Table 5.5: Harvesting of HaaS and Scrapy as a function of the number of services - Serchen Round3

Serchen (3rd round)		
Services	**HaaS**	**Scrapy**
20	3.233	3.338
40	8.574	12.468
60	12.752	20.271
80	14.927	23.583
100	19.282	36.923

Table 5.6: Average of three rounds of harvesting using HaaS and Scrapy - Serchen

Serchen (3rd round)		
Services	**HaaS**	**Scrapy**
20	3.13	2.882
40	8.446	13.445
60	11.630	20.169
80	14.974	23.663
100	18.813	36.3487

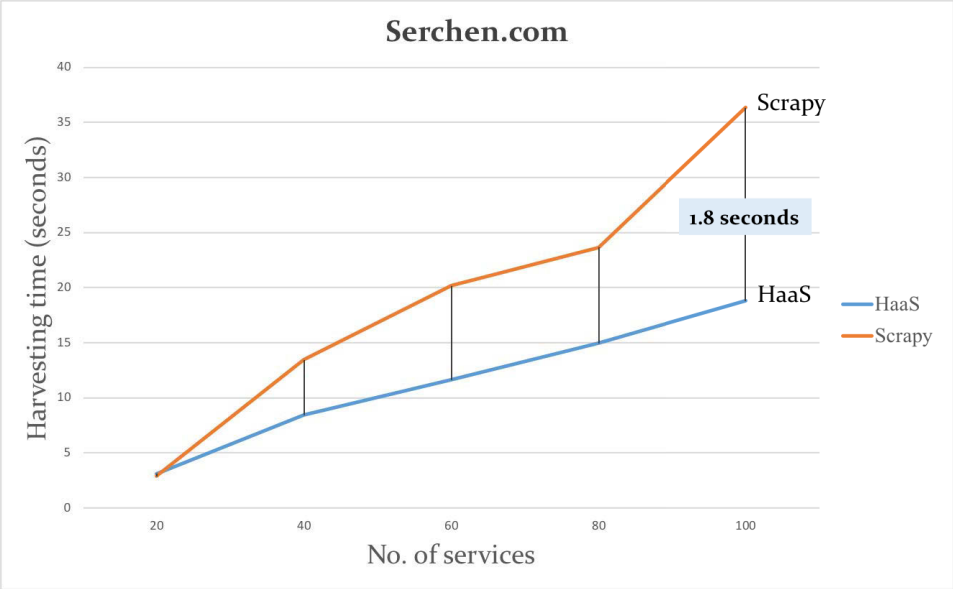


Figure 5.17: Comparison of HaaS and Scrapy harvesting time for serchen.com

We next harvested getapp.com using HaaS and Scrapy. We applied the polite harvesting feature for both tools (Table 10). Tables 11, 12 and 13 present the comparison of harvesting time for HaaS and Scrapy across three rounds. Similar to Serchen, we validate the GetApp harvesting results by measuring the percentage of change as a function of the number of harvested services (20, 40, 60, 80 and 100). Polite harvesting was not applied in any of the three rounds. The GetApp harvesting results show that the proposed HaaS system performs better than the existing system, Scrapy. The graph in Figure 11 illustrates that the harvesting time for HaaS is less than the time achieved by Scrapy for 20 and 40 services; however, for 60, 80 and 100 services, the harvesting time of HaaS is greater than that of Scrapy.

Table 5.7: Average of three rounds of harvesting using HaaS and Scrapy - Serchen

Harvesting Tool	**Record per session**	**Waiting time interval (in secs)**
Scrapy	10 pages	3
HaaS	10 records	3

Table 5.8: Harvesting of HaaS and Scrapy as a function of the number of services - GetApp Round1

GetApp (1st round)		
Services	**HaaS**	**Scrapy**
20	8.388	70.481
40	18.799	142.716
60	161.953	222.71
80	226.724	285.584
100	292.045	386.883

Table 5.9: Harvesting of HaaS and Scrapy as a function of the number of services - GetApp Round2

GetApp (2nd round)		
Services	**HaaS**	**Scrapy**
20	35.199	23.637
40	98.448	68.676
60	176.898	121.934
80	225.692	169.508
100	292.384	221.106

Table 5.10: Harvesting of HaaS and Scrapy as a function of the number of services - GetApp Round3

GetApp (2nd round)		
Services	**HaaS**	**Scrapy**
20	34.697	23.264
40	119.225	75.473
60	162.028	115.385
80	227.983	167.352
100	289.653	221.374

Table 5.11: Average of three rounds of harvesting using HaaS and Scrapy - GetApp

GetApp (2nd round)		
Services	**HaaS**	**Scrapy**
20	26.0946	39.127
40	78.824	95.621
60	166.959	153.343
80	226.799	207.481
100	291.361	276.454

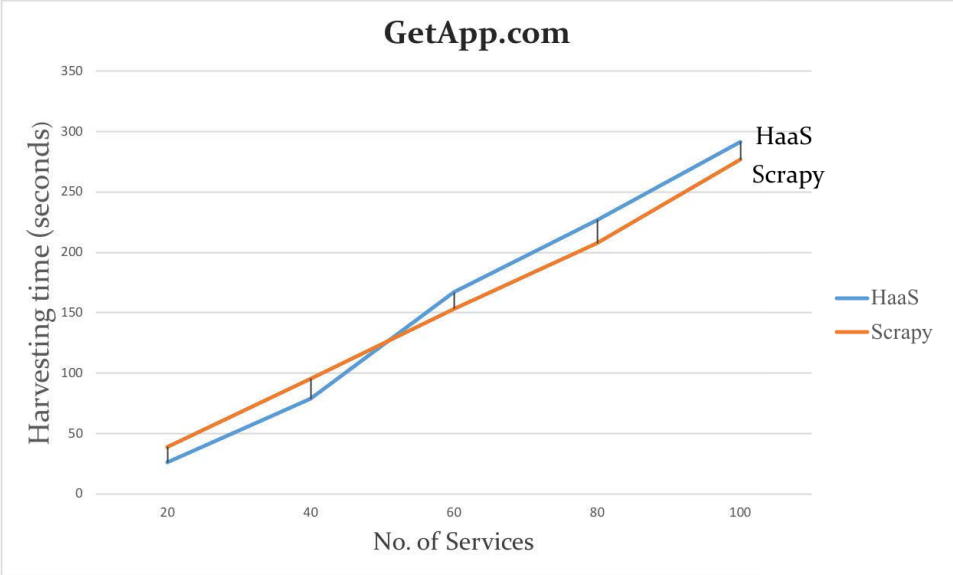
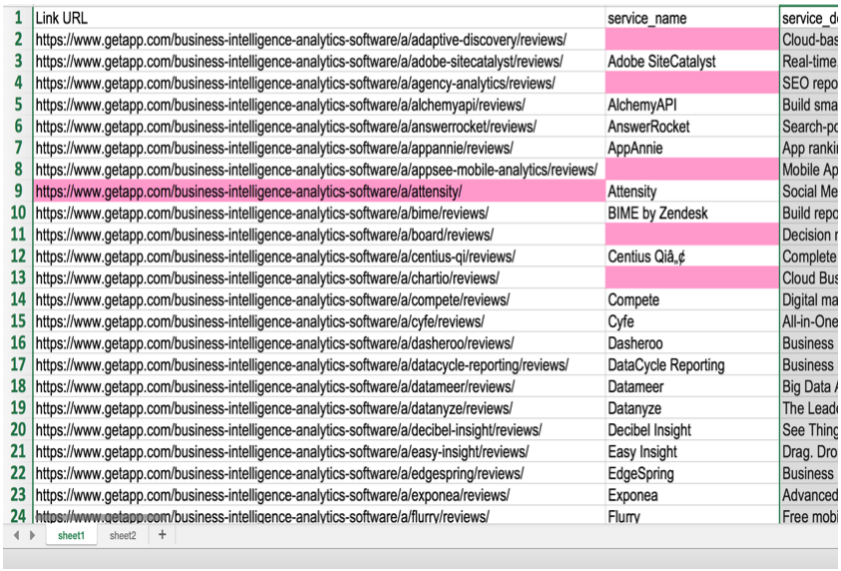


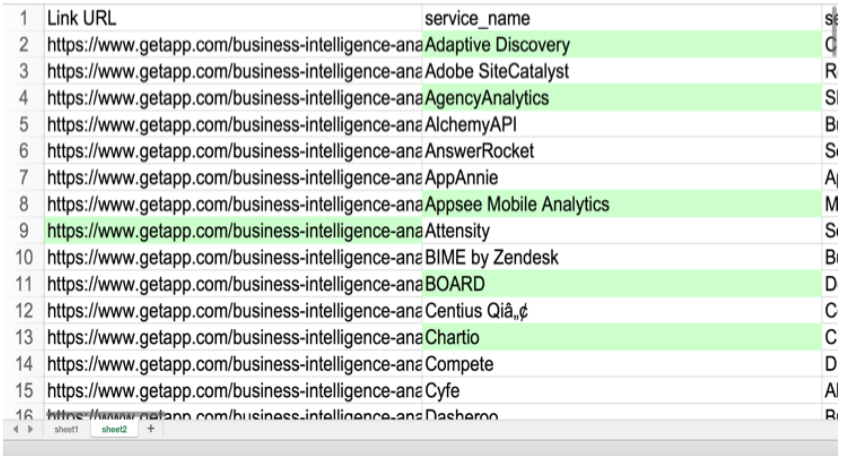
Figure 5.18: Comparison of HaaS and Scrapy harvesting time for GetApp

b. Harvested Data Quality In this section, we examine and compare the quality of the harvested data of both tools, HaaS and Scrapy. Scrapy harvesting quality depends on the programmer's skills in analyzing HTML structure and writing Scrapy scripts, whereas HaaS is a well-structured design with no variation in HTML structure. Text processing in both tools is slightly different (e.g. Unicode characters, new line vs space), but this does not affect the quality of the data. However, we have compared the results of harvesting 100 cloud services from serchen.com using both tools (Scrapy and HaaS). Figures 16 and 17 present the harvested data in two columns service URL, service name. The results in both figures indicate that a number of values were missing from the service description column in the case of the Scrapy results. By "missing value", we mean that the corresponding value was not present in the harvested data. There are 14 missing service name values out of 100 harvested services, with a successful harvesting quality rate of 86%. Interestingly, there are no missing values in the HaaS file (Figure 17), indicating the successful harvesting quality rate of the 100.



	Link URL	service_name	service_d
2	https://www.getapp.com/business-intelligence-analytics-software/a/adaptive-discovery/reviews/		Cloud-bas
3	https://www.getapp.com/business-intelligence-analytics-software/a/adobe-sitecatalyst/reviews/	Adobe SiteCatalyst	Real-time
4	https://www.getapp.com/business-intelligence-analytics-software/a/agency-analytics/reviews/		SEO repo
5	https://www.getapp.com/business-intelligence-analytics-software/a/alchemyapi/reviews/	AlchemyAPI	Build sma
6	https://www.getapp.com/business-intelligence-analytics-software/a/answerrocket/reviews/	AnswerRocket	Search-pc
7	https://www.getapp.com/business-intelligence-analytics-software/a/appannie/reviews/	AppAnnie	App rankir
8	https://www.getapp.com/business-intelligence-analytics-software/a/appsee-mobile-analytics/reviews/		Mobile Ap
9	https://www.getapp.com/business-intelligence-analytics-software/a/attensity/	Attensity	Social Me
10	https://www.getapp.com/business-intelligence-analytics-software/a/bime/reviews/	BIME by Zendesk	Build repo
11	https://www.getapp.com/business-intelligence-analytics-software/a/board/reviews/		Decision r
12	https://www.getapp.com/business-intelligence-analytics-software/a/centius-qi/reviews/	Centius Qiâ„¢	Complete
13	https://www.getapp.com/business-intelligence-analytics-software/a/chartio/reviews/		Cloud Bus
14	https://www.getapp.com/business-intelligence-analytics-software/a/compete/reviews/	Compete	Digital ma
15	https://www.getapp.com/business-intelligence-analytics-software/a/cyfe/reviews/	Cyfe	All-in-One
16	https://www.getapp.com/business-intelligence-analytics-software/a/dasheroo/reviews/	Dasheroo	Business
17	https://www.getapp.com/business-intelligence-analytics-software/a/datacycle-reporting/reviews/	DataCycle Reporting	Business
18	https://www.getapp.com/business-intelligence-analytics-software/a/datameer/reviews/	Datameer	Big Data A
19	https://www.getapp.com/business-intelligence-analytics-software/a/datanyze/reviews/	Datanyze	The Leade
20	https://www.getapp.com/business-intelligence-analytics-software/a/decibel-insight/reviews/	Decibel Insight	See Thing
21	https://www.getapp.com/business-intelligence-analytics-software/a/easy-insight/reviews/	Easy Insight	Drag. Dro
22	https://www.getapp.com/business-intelligence-analytics-software/a/edgespring/reviews/	EdgeSpring	Business
23	https://www.getapp.com/business-intelligence-analytics-software/a/exponea/reviews/	Exponea	Advanced
24	https://www.getapp.com/business-intelligence-analytics-software/a/flurry/reviews/	Flurry	Free mobi

Figure 5.19: Screenshot of data harvested from serchen.com by Scrapy



	Link URL	service_name	se
2	https://www.getapp.com/business-intelligence-ana	Adaptive Discovery	C
3	https://www.getapp.com/business-intelligence-ana	Adobe SiteCatalyst	R
4	https://www.getapp.com/business-intelligence-ana	AgencyAnalytics	S
5	https://www.getapp.com/business-intelligence-ana	AlchemyAPI	B
6	https://www.getapp.com/business-intelligence-ana	AnswerRocket	S
7	https://www.getapp.com/business-intelligence-ana	AppAnnie	A
8	https://www.getapp.com/business-intelligence-ana	Appsee Mobile Analytics	M
9	https://www.getapp.com/business-intelligence-ana	Attensity	S
10	https://www.getapp.com/business-intelligence-ana	BIME by Zendesk	B
11	https://www.getapp.com/business-intelligence-ana	BOARD	D
12	https://www.getapp.com/business-intelligence-ana	Centius Qiâ„¢	C
13	https://www.getapp.com/business-intelligence-ana	Chartio	C
14	https://www.getapp.com/business-intelligence-ana	Compete	D
15	https://www.getapp.com/business-intelligence-ana	Cyfe	A
16	https://www.getapp.com/business-intelligence-ana	Dasheroo	B

Figure 5.20: Screenshot of data harvested from serchen.com by HaaS

5.6 Conclusion

In this chapter, we have presented a service-based harvester called *Harvesting as a Service* (HaaS) for crawling cloud services information from heterogeneously structured Web portals. The key contribution of the proposed system is the HaaS user interface, which allows end-users to harvest websites without the need for developers or coding, unlike other harvesting tools in the literature. Experiments were carried out and the results show that compared to the existing system, Scrapy, our proposed system demonstrates a significant improvement in the quality of harvested data. The research indicates that the uses of HaaS support intelligent cloud service harvesting and the complexity of heterogeneously structured Web portals.

CHAPTER 6

CONSTRUCTING A DOMAIN-SPECIFIC ONTOLOGY FOR CLOUD SERVICES FROM WEB SOURCES (CLOUD SERVICES KNOWLEDGE BASE)

In the recent years, semantic web represented in ontologies plays an essential role in knowledge reasoning and knowledge representation. Researchers focused on constructing cloud services ontologies to discover cloud services over the web. However, as mentioned in chapter 2 most of these studies proposed constructing cloud service ontology based on existing ontology. For example, in [90] the authors used business ontologies to represent cloud services. However, these ontologies described classes (concepts) and individuals (instances) from business aspects and it does not have concepts to describe cloud services

Moreover, the number of cloud services advertisements has significantly increased, resulting in an increasing need for an intelligent method that can assist in discovering cloud service advertisements on the web. Cloud services provided their services offers online across several web portals; therefore, it is a challenge for the cloud consumers to find cloud service advertisements that match their needs within these different web platforms. Also, using web search engine to find service information is very common; though, the web search engines are using keyword-based search manner which usually retrieves relevant and irrelevant service information [50]. Additionally, the heterogeneity of cloud services data in marketing makes it hard for cloud consumers to read, realise, and compare the cloud services advertisements, while cloud providers are using different terminologies and vocabularies to describe cloud services.

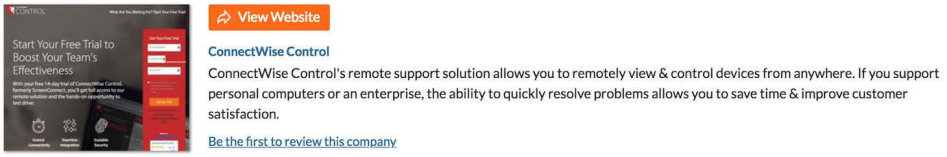


Figure 6.1: ConnectWise a cloud-based service profile in Serchen.com [20]

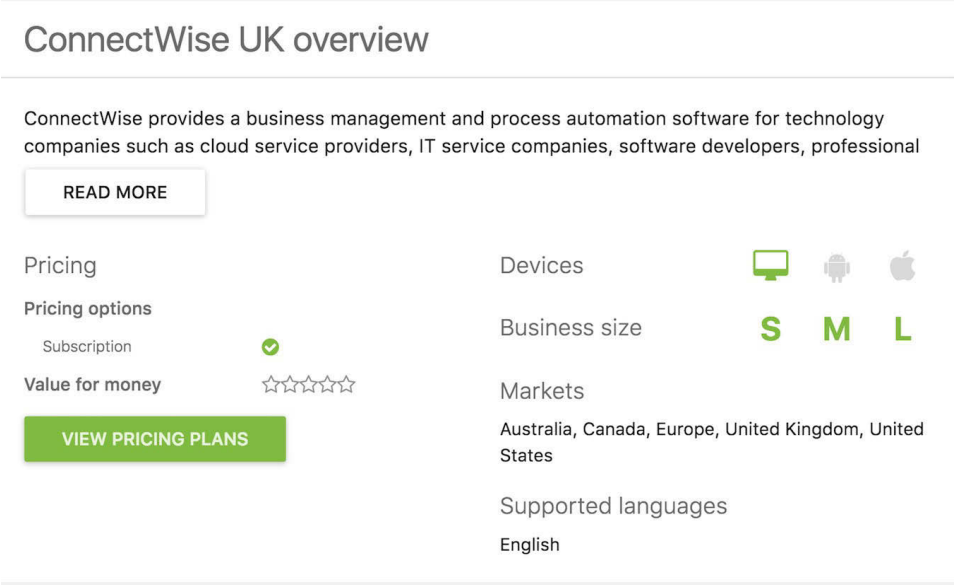


Figure 6.2: ConnectWise a cloud-based service profile in GetApp.com [5]

Moreover, as mentioned in chapter 2, ‘semantic search ’and ‘semantic annotation ’are well-known methods used to enhance the keyword-based searching technique of web search engines to retrieve accurate information in several domains. In regards, cloud service information retrieval online, thorough reviews of existing studies highlight that both methods have been used to enhance the process of discovering and retrieving cloud services information on the web. The trend of the semantic search is to extract more accurate information about cloud services from the internet by takes into account the context meaning of the keywords research in the web search engine. Semantic annotation is to add more information to describe cloud service entity in the web, which could support the automatic discovery of cloud services.

A thorough literature reviews in Chapter 2 shows that most of recent attention has focused on constructing cloud services ontology, then using these ontologies as reference for enhancing the searching capabilities and enhancing the retrieving information from the web [56, 76]. However, all of these studies referring to the existing ontologies such as Business ontology and NIST cloud service terminologies to semantically annotate the cloud services data which is not sufficient to describe the actual cloud services in real world scenario, the functional details of cloud services differ from other services advertised in the World Wide Web (WWW) [44]. Additionally,

researchers in other studies suggested the use of Web Service Description Language (WSDL) for cloud services semantic annotation which include the following elements input, operation and output [93]. A key issue with this semantic-based approach is that it does not include the essential elements for describing cloud service entity such as service type and service price.

Moreover, semantic searches and semantic annotations are well-known methods used to enhance the keyword-based searching technique of web search engines to retrieve accurate information in several domains. In regard to cloud service information retrieval online, a thorough review of the existing studies highlights that both methods have been used to enhance the process of discovering and retrieving cloud service information online. The objective of a semantic search is to retrieve more accurate information about cloud services from the web search engines by taking into account the context or meaning of the search term whereas a semantic description refers to the process of enriching cloud services with a semantic annotation that adds more information to describe the cloud services, thereby assisting their discovery.

From the above discussion, we note that a great deal of previous research into cloud services discovery has focused on using semantic technologies for cloud service discovery, though the shortcomings of current studies are summarized as follows:

1. The quality of information retrieval using web search engines relies on the quality of the domain ontologies.

2. Researchers suggested referring to existing ontologies such as Business ontologies to semantically annotate and describe cloud service entity, which is not sufficient to describe the actual cloud services in real world scenario.

3. Researchers do not provide any solution for structuring and annotating cloud service advertisement based on the organized cloud service information represents in the web sources. This could be a useful solution to solve the problem of heterogeneity in services advertisement in the web.

4. Researchers do not provide any means of constructing a knowledge base or commercial cloud services repository for listing cloud services.

To address the shortcomings, in this chapter we propose innovation solution for construing domain-specific ontology for cloud service from web sources. Many Tools developed to assist in designing and constructing the ontology, for example, OntoEdit and Protege[88],[13]. However, none of these tools can automatically build an ontology. Ontology knowledge including concepts, instances and relations has to be defined by the end-users and then using the existing tools to develop and organize the ontology. Therefore, we propose to harvest ontology concepts, instances and relations from the web sources.

With a large amount of cloud service information in the web, which dramatically increased and while most of this information is organized manually, in this study we propose to take advantages of structured services information in the web for constructing a domain-specific ontology. Our proposed 'cloud services knowledge base 'method involves the construction of domain-specific ontology for cloud service, which represents the cloud service classifications based on the classification given to the cloud services in multiple web sources, such as GetApp and Serchen. The significance of this research is that aiming at collecting the ontology knowledge: concepts, instances, and relations from the web sources, which is different from all existing studies in the literature that building the cloud services ontology based on existing ontology in the body of the literature.

To construct the cloud services Business classification (CS BCLAS) ontology based on structured service information in the web. We adopted an approach called 'a social classification '[68]. This approach has successfully applied in previous research related to ecosystem services classification [40], in this study the authors collected ecosystem services classification from yellow pages, and then they constructed ontology based on this classification within yellow pages. The developed ontology in this study has been used as knowledge base for classifying the ecosystem services. The idea of a social classification approach is to involves the users and community in the process of organizing information and classification. Therefore, we consider the cloud services classification provided by online community such as service providers, agents and consumers as the knowledge source for constructing cloud service classification ontology. The objectives of constructing a cloud services knowledge base method include:

1. Constructing a domain-specific ontology for cloud services based on web sources.

2. Constructing ontology-based cloud services knowledge base grounded on harvested web sources. This knowledge base contains the service metadata that can be used to store service advertisements, which annotating to specific-domain ontology concepts toward retrieving cloud services advertisement more efficiently.

3. Constructing a cloud services knowledge base, which acts as a cloud services knowledge source marketplace.

This chapter organized as follows: Section 6.2 the methodology developed is described; Section 6.3 presents the workflow of proposed approach; Section 6.4 presents the prototype and Section 6.5 explains the experiments conducted to evaluate our methodology; finally, some conclusion is put forward in Section 6.6

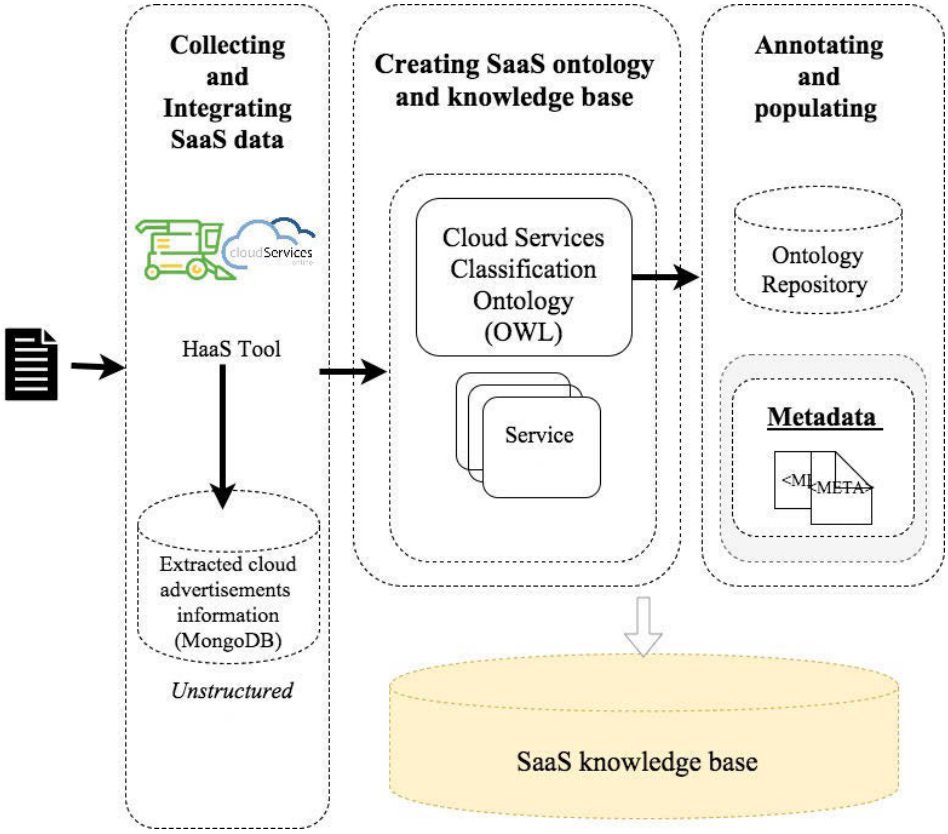


Figure 6.3: Approach for creating SaaS ontology and knowledge base

6.1 The proposed approach for creating SaaS ontology and knowledge base

In this section, we introduce an innovative solution for cloud service discovery that takes into account the heterogeneity in services advertisements context published in the World Wide Web (WWW). The significant of this work is that the construction of the SaaS ontology based on web data collected and integrated from various web sources. To classify cloud services based on web data from multiple web sources. we present and describe in detail the proposed approach as shows in figure 6.3 1), which composed of three layers: (1) Collecting and Integrating service data, (2) Creating service ontology and knowledge base, and (3) Annotating and populating the structured ontology. To explain, implement and validate the proposed framework in this research, we focus on Software as a Service, whereas our proposed structure is designed to apply to all types of cloud services. Hence, the remaining sections of this paper focus on SaaS.

6.1.1 Collecting and Integrating SaaS data

In this layer, the method used to discover and select the websites for SaaS concepts and taxonomies for creating SaaS ontology, then a web crawler used to collected the data. In this work, we developed a crawler especially designed to collect service information from various web sites called Harvesting as a Service (HaaS). The HaaS has the capability to harvest HTML structured web data from heterogeneous web portals, by heterogeneous we mean different structured web site. Then, the harvested data store in in a repository in a semi-structured manner. In our work, our target is web pages which have published SaaS advertisements, therefore we manually choose to harvest three heterogeneous web portals that publish and market cloud services, namely `www.cloudreviews.com`, `www.getApp.com` and `www.serchen.com`. We collected HTML structured data related to SaaS advertisements on these web portals and then exporting the collected data in JSON file or CSV file. The collected data in the files has same HTML structure. To apply an exhaustive analysis and obtain useful information from the collected data, we organised the data in relation database and then perform some analysis to extract important service concepts and taxonomies related to SaaS business domain. The candidate concepts are processed in order to select the most adequate ones by performing a statistical analysis. The selected concepts and taxonomies are finally incorporated to the ontology. The resulting taxonomy use to guide a search for cloud services. More details about this shown on fig 6.10 and explains in section 6.1.2

6.1.2 Creating SaaS ontology and knowledge base

In this section, the approach to discover, collect, and select representative concepts for a cloud services business classification is described. This method is based on analysing several cloud services advertisements in order to find important concepts for cloud services advertisements domain by studying the initial keywords used in advertisements for service classification. By performing a statistical analysis, the most adequate concepts are selected. The selected concepts are finally incorporated to the ontology. The resulting taxonomy of terms can be the base for discovering cloud services information.

More details, the work-flow shown in fig 6.10, has the following steps:

1. It starts with select the publicly available web portals to obtain the most represented web sites that contain the cloud services advertisements information.

2. For each website returned, an exhaustive analysis is performed to obtain useful information from each one, as following:

    a) Different structured and layout for each website.

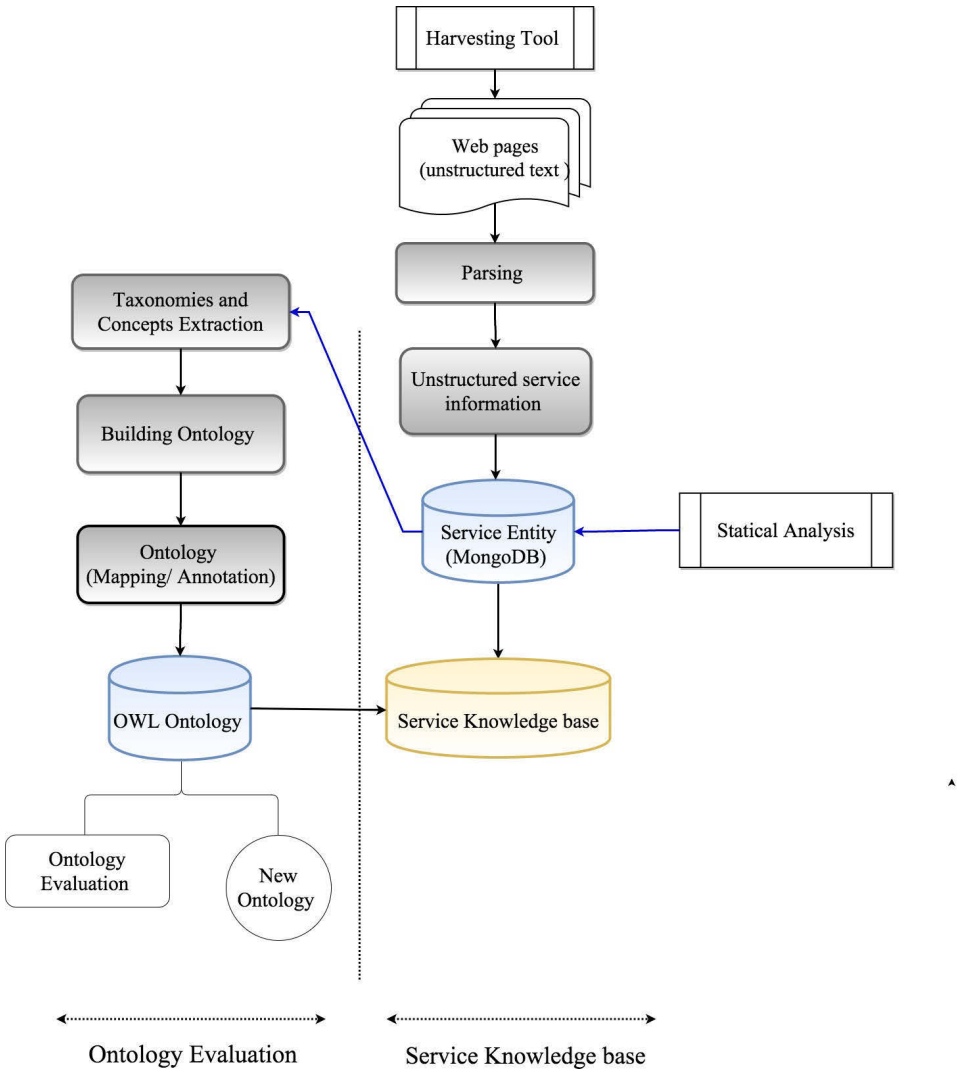


Figure 6.4: Ontology building method

b) For each website, the parser "HaaS" learns the structured using Intelligent Algorithm and storing the result JSON script. Then, the parser uses this script as guide to collected data from all webpages in the websites.

c) All websites obtain has HTML formats.

3. The parser returns useful text data from each site related to cloud services advertisement and tries to find the initial keyword used to classify the advertisement (e.g. Marketing) and selected as candidate concepts.

a) Word must be represented with a standard ASCII character set (not Chinese for example).

a) For each candidate concept selected, as shown in table 6.1, statistical analysis is performed to select the most representative ones. Our method considers the following attributes:

   i. Total number of term (candidate) appearances which represents a measure of the concept‚Äôs relevance for domain and allows to eliminate very specific ones (e.g. lease accounting).

   ii. Number of different websites that contains the term at least one time: this gives a measure of the concept‚Äôs relevance for the domain and allows to eliminate very specifics ones (e.g. insurance policy)

4. To obtained result is a hierarchy that is stored as an ontology. Each class name is represented as it is in the table 6.1

Table 6.1: Candidate concepts for the SaaS business ontology

Term (concept)	**Root**	**Appear**	**Different pages**
Communication	SaaS	3	1106
Sales	SaaS	3	1059
Finance Accounting	SaaS	3	1328
Integration solution	SaaS	2	681
HRM	SaaS	3	478
Business Intelligence	SaaS	3	4291
Collaborates	SaaS	3	1620
CRM	SaaS	3	143
Marketing	SaaS	3	2103
Management	SaaS	3	3790
Project Management	Management	3	1389
Operation Management	Management	3	577
IT Management	Management	3	1824

[1] Concepts: means SaaS advertisements classification concepts appears in the websites
[2] Root: means the root of the service taxonomy used in the website
[3] Appear: refers to the number of web portals that used the classification term
[4] Different pages: refers to the number of appearances of classification in different pages

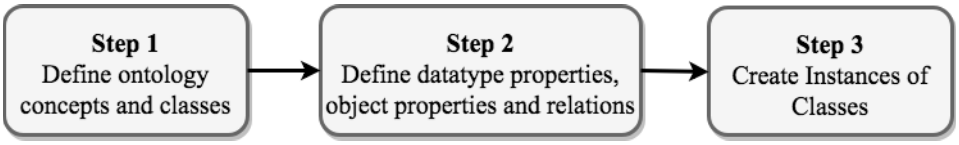


Figure 6.5: Modeling SaaS BCLAS ontology

6.1.2.1 Ontology representation

To represent and store the ontology, we use a standard representation language: Web Ontology Language (OWL), which is a semantic language for publishing and sharing ontologies on the World Wide Web (WWW). Once the ontology created it is easy to obtain each service advertisement for each service category (concept), because their URLs are stored on each leave (subclass) fram. The ontology concepts are covered the most important service taxonomies used in the web sources. The updating of this concepts could be easily by harvesting more web portals and performing a statistical analysis for each concept keyword. To design and store the ontology, we applied the following:

1. **Define ontology concepts and classes:** this step refers to presenting the most general concepts in the domain knowledge and subsequent specialisation of the concepts. Figure 6.6 shows the hierarchy ontology model for SaaS business categorisation that has been developed. For example, as shown in the model, we have a general concept, such as Management, which is further specialised into subclasses, for example, management SubClassOf (SaaS) and project management subclass-of (management). The development of ontology concepts and classes is explained in detail in Section 6.1.3

2. **Define datatype and object properties:** this step presents the logical association between classes and individuals. There are two types of properties: an object property and a datatype property. The object property refers to the relationship between individuals, whereas the datatype property refers to the data value of the individual, for example, Marketo Analytics:hasProviderLink `http://www.marketo.com/`

3. **Create instances of classes:** this step refers to entities or objects and is also known as instances which are the main component of the ontology, for example, the concept business intelligence could be described as a set of Bime Analytics, Marketo Analytics and Yellowfin Analytics which are entities of business analytics services.

After structuring the SaaS business categories into a conceptual ontology model, we then structured the harvested data related to the SaaS advertisement description, such as service name, service provider link and service category etc. into a relational database which represents the service advertisement description as attributes. These attributes are the annotation content

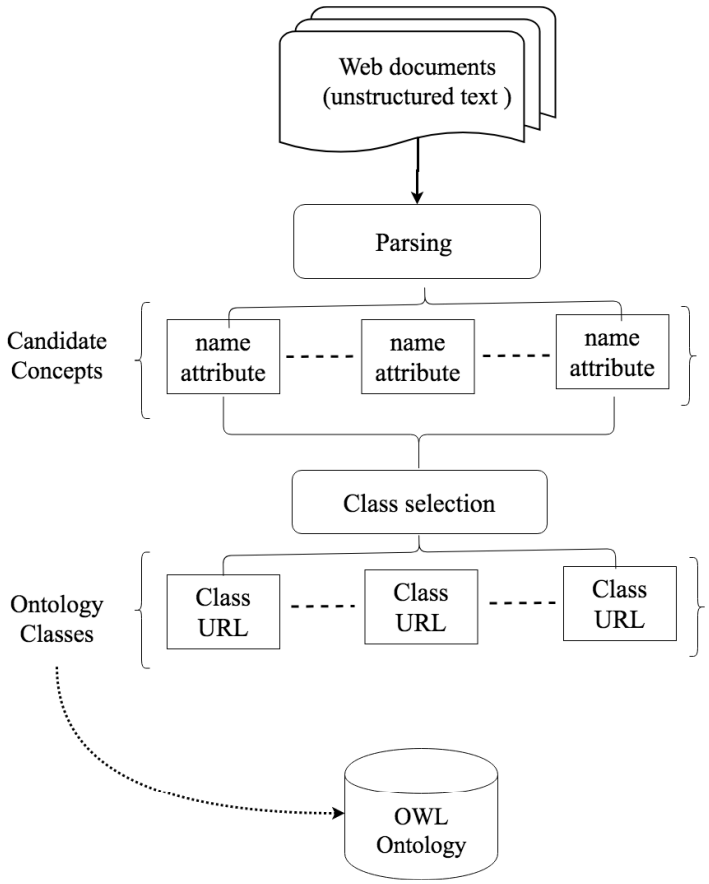


Figure 6.6: Modeling SaaS BCLAS ontology

for the SaaS advertisements which will be used to create the service metadata description in the ontology.

6.1.3 Annotating and Populating SaaS advertisement domain ontology (A/P SaaS)

This layer represents the SaaS advertisement knowledge base and comprises three parts: service ontology repository (OWL), service annotation (service meta- data OWL) and populating the service metadata. The service metadata (M) provides descriptive information on the content of the advertisements. For exam- ple, for a service, the service name, the service category, the service description, the provider link, the starting price, the rating, the free trial and mobile app are: Marketo Analytics , Business Intelligence , Marketo offers a marketing software platform to help drive the success of small-sized enterprises and large firms. The software it offers is complete, powerful and user-friendly. Marketo sales effectiveness and marketing automation software rationalise the marketing processes, sends out a larger number of campaigns, enhances sales performance and creates more leads , `http://www.marketo.com/` , $1195/month, Yes, Yes .

Ontology population aims at semi-automatically inserting instances of SaaS business category concepts, proper ties and relations to the knowledge base as defined in the domain ontology. Once the SaaS advertisement annotation and ontology population are performed, the end-users of an application can exploit the resulting annotations and instances to query, to share, to access and to publish SaaS advertisements, metadata and knowledge.

6.1.3.1 Hierarchy of SaaS Business Category Ontology (SaaS BCLAS)

The SaaS BCLAS ontology is used to represent the knowledge of services in a particular domain. The knowledge of services comprises basic SaaS business categories which may relate to each other. We utilise information on SaaS advertisement business categories from the semi-structured data (CSV file format). The SaaS business categories are represented by SaaS categories concepts, and also the relations among SaaS categories are also defined. We propose a hierarchical structure to describe the SaaS BCLAS ontology which consists of SaaS category concepts and the relationships between them.

The structure of the SaaS BCLAS ontology concepts is a four-layer hierarchy (fig 2). The first layer is the root of the hierarchy, which represents the abstract concepts of all the service categories in the SaaS domain. The second layer is the preliminary specialisation for the abstract SaaS category concept, which classifies ten categories of SaaS service concepts - [*Business Intelligence, Collaboration, Communication, CRM, Finance Accounting, HRM, Integration Solution, Management, Marketing and Sales*]. The third layer is the further specialisation for some of the abstract SaaS category concepts in the second layer, which represents the services in each basic sub-domain of SaaS services. The service category concepts execute the function of the SaaS business category domain definition, which corresponds to the actual SaaS services in the real world. In conclusion, each service concept has the properties of concept description, which refers to the detailed description of the corresponding service. These properties can be used to semantically match with the SaaS service metadata, which will be discussed later. Finally, the ontology details are stored in a repository in OWL format.

6.1.3.2 SaaS service advertisement metadata

The ontology is used to add semantics to harvest data from the cloud portals. We consider that SaaS service advertisement S is represented by service metadata M which describes the general knowledge of the SaaS services such as service ID, service name and service details. In this work, we utilise the service metadata descriptive information from the relational database which has the harvested data organised by attributes, such as service name, service description and service category. To identify the service concepts that are relevant to a certain service category concept, Fig 3 illustrates the format of the SaaS services metadata which can be represented as a tuple where the elements of the tuple can be defined as follows:

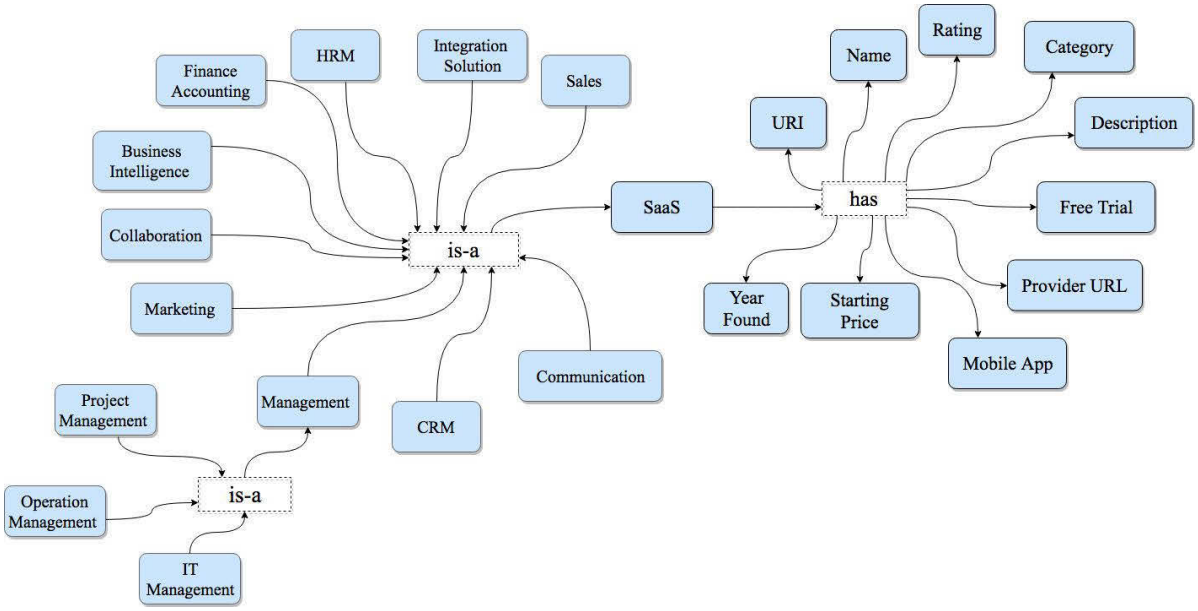


Figure 6.7: Conceptual Model of SaaS Business Category Ontology (SaaS BCLAS).

[*service id, service name, service category, service description, provider link, free trial (yes, no), mobile app (yes, no), rating, starting price, year founded*]
Service ID : is the URI of the service which is the reference to the semantically linked concepts.
Service Name : is the name of the service.
Service Category : is the category to which the service belongs.
Service description : is the detailed text description of the service features and facilities.
Provider link : is the URL link of the service provider.
Starting Price : is the starting price of the service per month.
Rating : is the score that a consumer gives to a service after purchasing and using it.
Free Trial : indicates if the service is available for free a trial or not.
Mobile App : indicates if the service is a mobile application or not

6.1.4 Maintaining the knowledge base

To maintain the accuracy of the KB, we deployed the Time-driven method for updating the KB. The time-driven method is based on running/triggering the crawler engine (HaaS) at regular intervals of time schedule once a month based on how frequent cloud services providers are updating or adding new services. The pseudocode of the time-driven method explains below:

Algorithm: Time-driven algorithm for updating the KB

```
1.  Function SetCrawlerServiceRun(15):
2.  BEGIN:
3.  Creates a schedule process to run RunServicesCrawlerAgent() every 15 days
4.  END:

5.  Function RunServicesCrawlerAgent():
6.  BEGIN:
7.  Call CrawlerAgentStart()    #Starts cloud services crawling
8.  Sleep(30)    #thread sleep for 30 secs
9.  While (IsCrawlerProcessRun):
10. BEGIN:
11. Print "Your knowledge base updating process still running ..... "
12. Sleep(30)    #thread sleep for 30 secs
13. END:

14. VAR NumberOfRecordsAdded= Call FetchNumberOfRecordsAdded(Today)
15. Print Print "Your knowledge base has been updated with "+NumberOfRecordsAdded +
    "new services"
16. END:
17. While (Check)

18. Function IsCrawlerProcessRun():
19. BEGIN:
20. IF (crawler engine process IS running):
21. Return TRUE
22. ELSE IF (crawler engine procees finished)
23. Return FALSE
24. END:
25. Funtion FetchNumberOfRecordsAdded(Datetime):
26. BEGIN:
27. return Query database to get number of records updated
28. END:
```

End Algorithm

Figure 6.8: Time-driven algorithm for updating the KB algorithm

6.2 Approach Workflow

Several components collaborate to harvest web pages and building the SaaS knowledge base involving service ontology (O) and service metadata (M) as described in Section 3. We present the workflow of the process in detail as follows:

- *Step 1 Harvesting Data:* This task involves collecting the SaaS advertisement descriptions from the web pages. Usually, the data on web pages is non-structured though web pages are great sources for SaaS service information. Therefore, we developed a harvester called HaaS to harvest SaaS information among heterogeneous web portals and the collected data

is stored in a repository in a semi-structured manner (CSV file).

- *Step 2 Structuring Data:* This task represents the construction of the SaaS advertisement knowledge base which consists of two parts. Firstly, we refer to the semi-structured repository contents to construct the SaaS advertisement entity, which has descriptive information of the SaaS advertisements. Each advertisement entity has the following attributes [service ID, service name, service category, service description, provider link, starting price, rating, free trial, mobile app]. In this context, the advertisement entity represents the descriptive structured information on the SaaS advertisements which will be used in Task 3 to annotate and add information to the SaaS domain ontology. Service entities are stored in a relational database (SQL file) for future reference. The SaaS BCLAS ontology is developed by referring to the relational database attributes and extracting the ontology concepts. The main purpose of SaaS BCLAS is to semantically categorise SaaS advertisements as defined by the domain ontology (SaaS BCLAS). After the ontology has been structured, we add the instances of the SaaS advertisements to the concepts and annotate each element of the SaaS advertisements which will be discussed in detail in Task 3.

- *Step 3 Mapping and populating the ontology:* The aim of this task is to semi-automatically insert new instances of SaaS advertisement concepts, properties and relations to the knowledge base as defined by the domain ontology (SaaS BCLAS). The final result of this task is the SaaS advertisement knowledge base which the end-user can use to query, share, access and publish SaaS advertisement metadata and knowledge.

6.3 Prototype Implementation

The prototype implementation phase consists of three sub-process similar to the conceptual model design phase described in the previous sections, which are harvesting web data , Structuring Data' and 'Ontology Mapping and Population .

The prototype implementation phase consists of three sub-processes, similar to the conceptual model design phase described in the previous sections, which are harvesting web data, structuring data and ontology mapping and population. The first process is realised by using the developed harvesting tool, called Harvesting as a Service (HaaS), to crawl the SaaS service data among heterogeneously structured web portals. HaaS is capable of collecting heterogeneous data among different web portals and storing the result as semi-structured data in various format files such as CSV, SQL or PDF. A discussion of the HaaS tool is outside the scope of this paper, as in this work, the focus is on the issue of a lack of knowledge sources for constructing a service knowledge base.

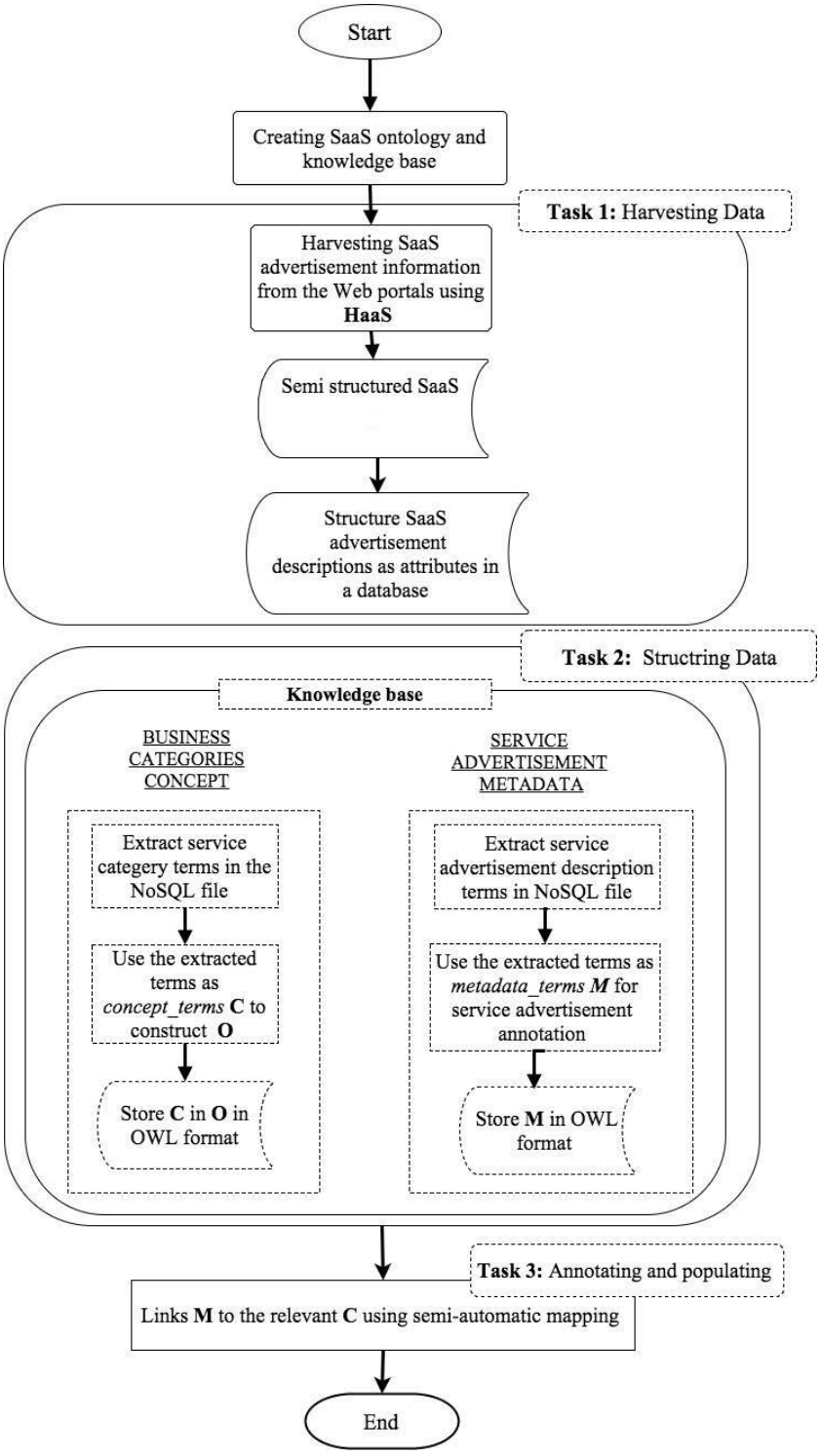
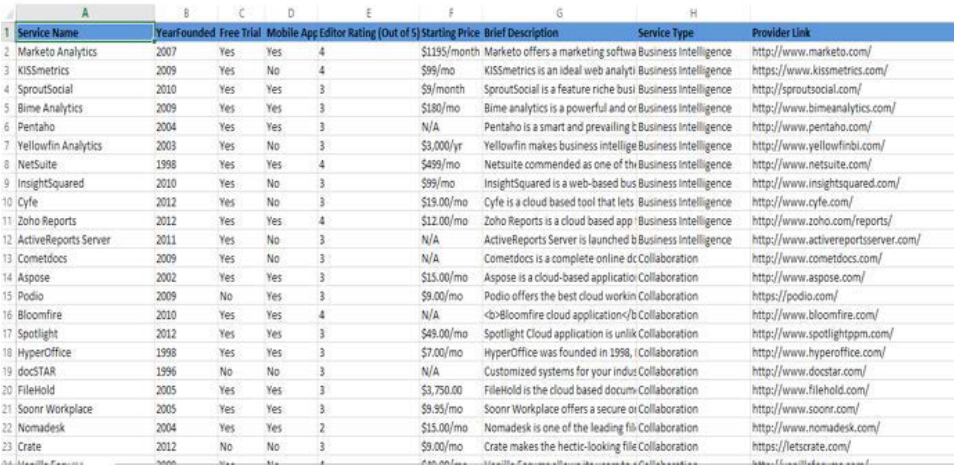


Figure 6.9: Approach Workflow

The results in Figure 6.10 show that HaaS has the ability to retrieve accurate data as well as organise the harvested data in a semi-structured manner which would be useful for creating the



	A	B	C	D	E	F	G	H	
1	Service Name	YearFounded	Free Trial	Mobile App	Editor Rating (Out of 5)	Starting Price	Brief Description	Service Type	Provider Link
2	Marketo Analytics	2007	Yes	Yes	4	$1195/month	Marketo offers a marketing softwa	Business Intelligence	http://www.marketo.com/
3	KISSmetrics	2009	Yes	No	4	$99/mo	KISSmetrics is an ideal web analyti	Business Intelligence	https://www.kissmetrics.com/
4	SproutSocial	2010	Yes	Yes	3	$9/month	SproutSocial is a feature riche busi	Business Intelligence	http://sproutsocial.com/
5	Bime Analytics	2009	Yes	Yes	3	$180/mo	Bime analytics is a powerful and or	Business Intelligence	http://www.bimeanalytics.com/
6	Pentaho	2004	Yes	Yes	3	N/A	Pentaho is a smart and prevailing t	Business Intelligence	http://www.pentaho.com/
7	Yellowfin Analytics	2003	Yes	No	3	$3,000/yr	Yellowfin makes business intellige	Business Intelligence	http://www.yellowfinbi.com/
8	NetSuite	1998	Yes	Yes	4	$499/mo	Netsuite commended as one of th	Business Intelligence	http://www.netsuite.com/
9	InsightSquared	2010	Yes	No	3	$99/mo	InsightSquared is a web-based bus	Business Intelligence	http://www.insightsquared.com/
10	Cyfe	2012	Yes	No	3	$19.00/mo	Cyfe is a cloud based tool that lets	Business Intelligence	http://www.cyfe.com/
11	Zoho Reports	2012	Yes	Yes	4	$12.00/mo	Zoho Reports is a cloud based app	Business Intelligence	http://www.zoho.com/reports/
12	ActiveReports Server	2011	Yes	No	3	N/A	ActiveReports Server is launched b	Business Intelligence	http://www.activereportsserver.com/
13	Cometdocs	2009	Yes	No	3	N/A	Cometdocs is a complete online dc	Collaboration	http://www.cometdocs.com/
14	Aspose	2002	Yes	Yes	3	$15.00/mo	Aspose is a cloud-based applicatio	Collaboration	http://www.aspose.com/
15	Podio	2009	No	Yes	3	$9.00/mo	Podio offers the best cloud workin	Collaboration	https://podio.com/
16	Bloomfire	2010	Yes	Yes	4	N/A	<b>Bloomfire cloud application</b	Collaboration	http://www.bloomfire.com/
17	Spotlight	2012	Yes	Yes	3	$49.00/mo	Spotlight Cloud application is unlik	Collaboration	http://www.spotlightppm.com/
18	HyperOffice	1998	Yes	Yes	3	$7.00/mo	HyperOffice was founded in 1998,	Collaboration	http://www.hyperoffice.com/
19	docSTAR	1996	No	No	3	N/A	Customized systems for your indus	Collaboration	http://www.docstar.com/
20	FileHold	2005	Yes	Yes	3	$3,750.00	FileHold is the cloud based docum	Collaboration	http://www.filehold.com/
21	Soonr Workplace	2005	Yes	Yes	3	$9.95/mo	Soonr Workplace offers a secure or	Collaboration	http://www.soonr.com/
22	Nomadesk	2004	Yes	Yes	2	$15.00/mo	Nomadesk is one of the leading file	Collaboration	http://www.nomadesk.com/
23	Crate	2012	No	No	3	$9.00/mo	Crate makes the hectic-looking file	Collaboration	https://letscrate.com/

Figure 6.10: A screenshot of the SaaS advertisement harvested data in CSV format.

knowledge representation of SaaS services.

The second process is realised by using SQL database and Protege-OWL. We extracted the meaningful data from the semi-structured repository (CSV) and we utilise SQL database to transform it to SaaS ads attributes, as shows in Figure A screen-shot of SaaS ads harvested data in CSV format.. Next, we utilise Protege OWL as the main tool for domain ontology construction. The ontology defines as a shared vocabulary used to model a specific domain, so for the purpose of this work we choose a particular service domain for SaaS services marketing namely business categories as the boundary within which the ontology built. Figures 6.10, 6.11, 6.12 and 6.13 show the screen-shot of the implementation of SaaS BCLAS in Protege-OWL.

The third process is realised by adding annotations and populating the SaaS BCLAS ontology. Once the ontology is defined as having classes and properties, the next task is to create mappings that tell the reasoner how the service advertisement entity data in the relational database (SQL) relates to the classes and properties in the SaaS CATGE ontology. In this context, the On-Top Protege Plugin is used to generate class individuals. The main reasons for choosing OnTop Protege Plugin [13] are as follow:

1. DB-Ontology for editing the mappings.

2. Mapping language is quite powerful and can be surprisingly intuitive.

3. Quest query engine is integrated into Protege.

4. SPARQL is provided in the query interface with the help of Quest.

We describe the process of inserting SaaS advertisement instances and adding an annotation to each element defined in the SaaS BCLAS ontology, as following:

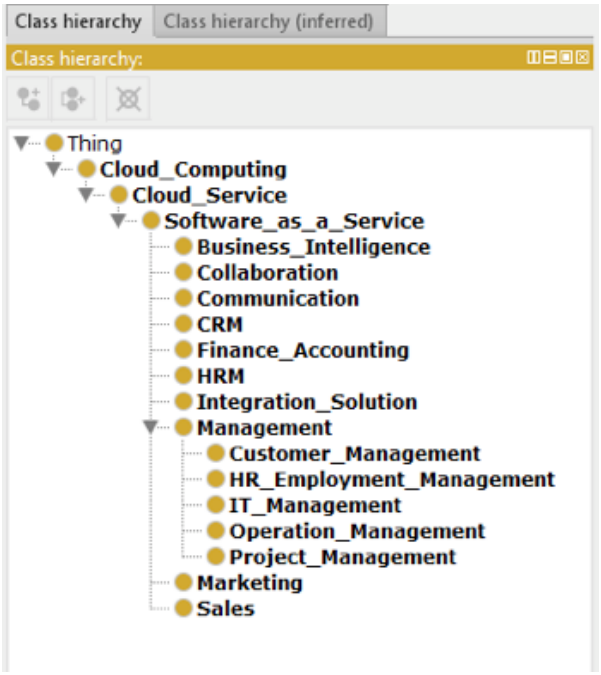


Figure 6.11: A screenshot of SaaS BCAGT ontology hierarchy.

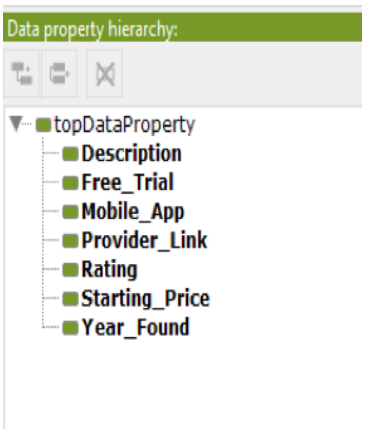


Figure 6.12: A screenshot of SaaS BCLAS ontology data properties.

- Step 1. Data Source Manger: in this step, we create a connection between the data source and Protege. The OnTop Plugin requires a JDBC driver, there- fore we import it with Protege as shown in Figure 6.15

- Step 2. Mapping Manger: the second step is to define the mapping. The source field within the mapping manager supports a standard SQL query. With the source, we need to determine the type of records we need to retrieve, and according to the source, the target is defined. To define the target (triple template), the RDF Turtle syntax is considered. Figure 6.16 shows the source and the target in detail.

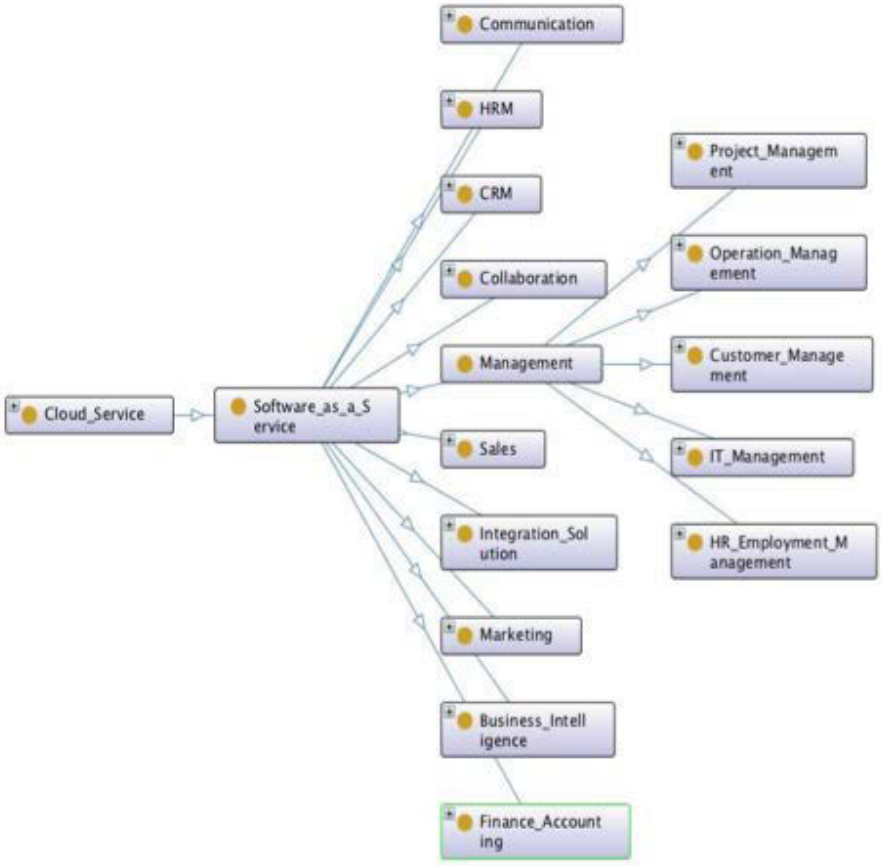


Figure 6.13: A screenshot of the SaaS BCLAS hierarchy.

```
PREFIX: http://www.semanticweb.org/smarkt
/ontologies/2017/11/saasontology#
Select * WHERE {
?p a:Business_Intelligence.
?p :Free_Trial?Free_Trial.
?p :Mobile_App?Mobile_App.
?p :Rating ?Editor_Rating_Out_of_5.
?p :Provider_Link ?Provider_Link.
?p :Year_Found ?YearFounded.
?p :Description ?brief_ Description.
?p :Starting_Price ?Starting_Price.
}
```

Figure 6.14: SPARQL code for mapping between the relational database and the ontology.

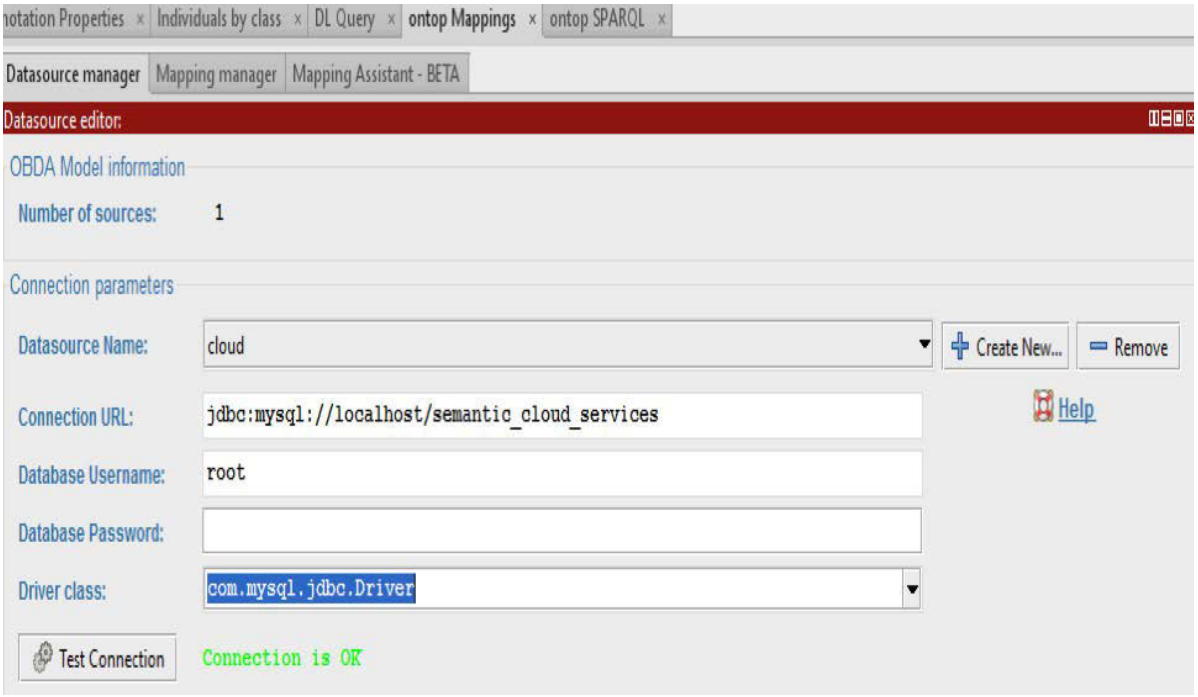


Figure 6.15: A screenshot of the data source connection.

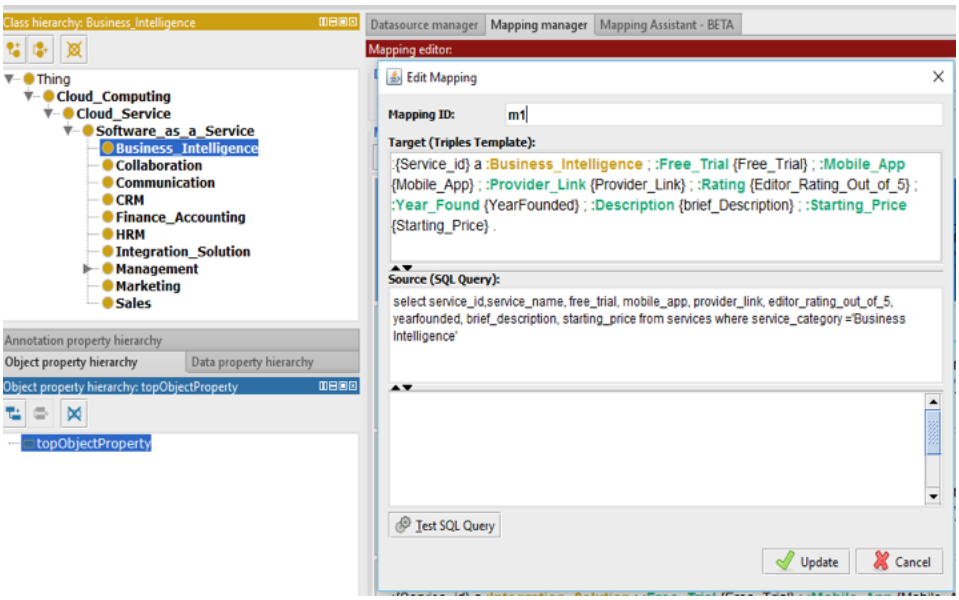


Figure 6.16: A screenshot of defining the mapping.

As seen in Figure 6.16, the mapping is defined for one of the classes, Business Intelligence . Firstly, the source is defined to retrieve the records (services) whose categories are Business Intelligence. To convert this source into the target, the Turtle Syntax is used. In the target, : denotes the default prefix (the URI of our ontology), a is a predefined alias for the predicate

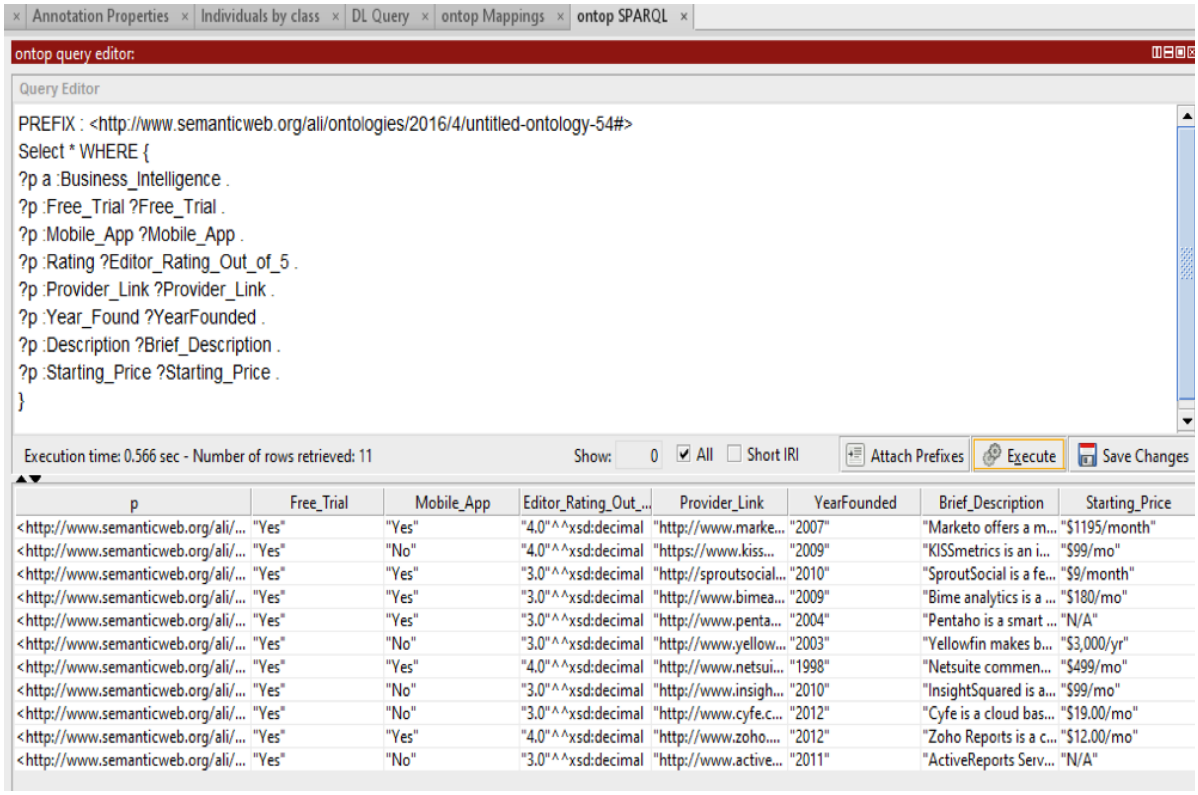


Figure 6.17: A screenshot of the implementation of SPARQL

rdf:type, and Business Intelligence is a class name in our ontology. The triple, :service id a: Business Intelligence states that the individuals identified by the string :service id are instances of the class Business Intelligence. Moreover, :Free Trial, :Mobile App, :Provider Link, :Rating, :Year Found, :Description and :Starting Price are data properties in our ontology. Similarly, we generate mappings for other classes as well.

- Step 3. OnTop SPARQL: After generating the mapping, we test all of the map- pings through the OnTop SPARQL interface. We follow the SARQL query syntax to test the mappings. Below is Figure 6.17 shows an example of one of the mapping tests and a screenshot of the implementation of OnTop SPARQL which shows the SAPRQL query test applied for Business Intelligence.

- Step 4. Materialise Triple: this is the final step to add SaaS advertisement individuals (instances) in the SaaS BCLAS ontology . Once the SARQL query test is performed, the individuals are generated and added to the respective classes defined by the mapping. Figures 6.18 and 6.19 show examples of how the materialise triple is per-formed and how the SaaS advertisement individuals are generated.

In order to evaluate the ontology, we used the JOWL Plugin for the interface, which displays the tabs, classes, object properties, individuals (data), and SPARQL - DL test for query the data. The classes tab displays all of the classes in a semantic tree view. Figure 6.20shows the plugin interface.

Figure 6.18: A screenshot of individuals generated from the materialisation.

In order to make the front end user-friendly as an interface for SaaS SMARKT, we changed the front end of the JOWL plugin so the data can be displayed in a more elegant way. Figures 6.21, 6.22 and 6.23 show the new interface of the plugin. The SaaS SMARKT is available online via `http://52.37.193.247/saas-semantic-market/`

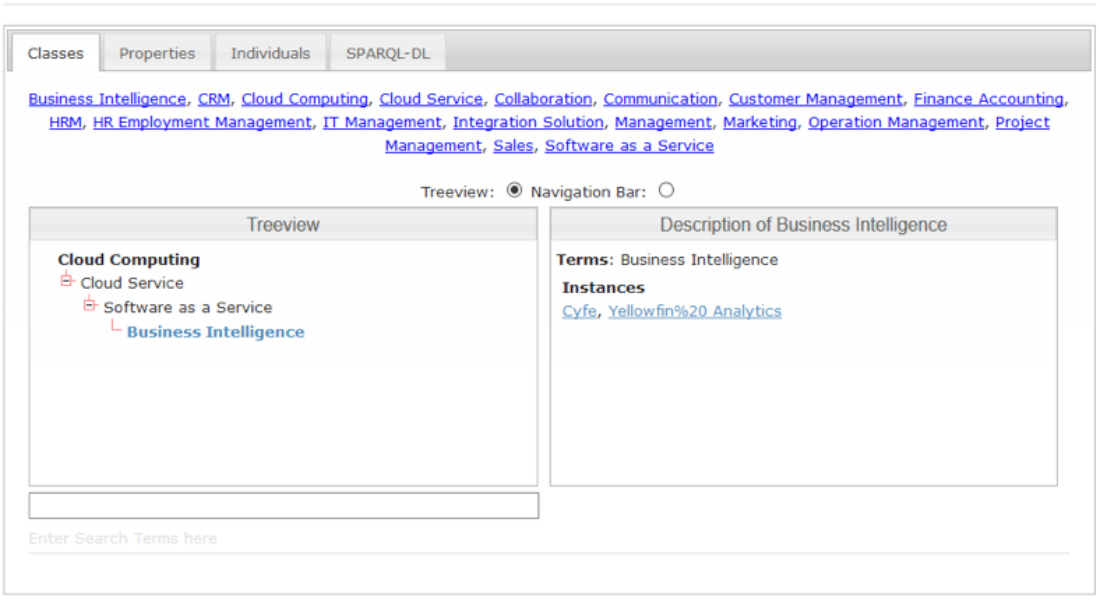


Figure 6.20: A screenshot of the default interface of the JOWL plugin.

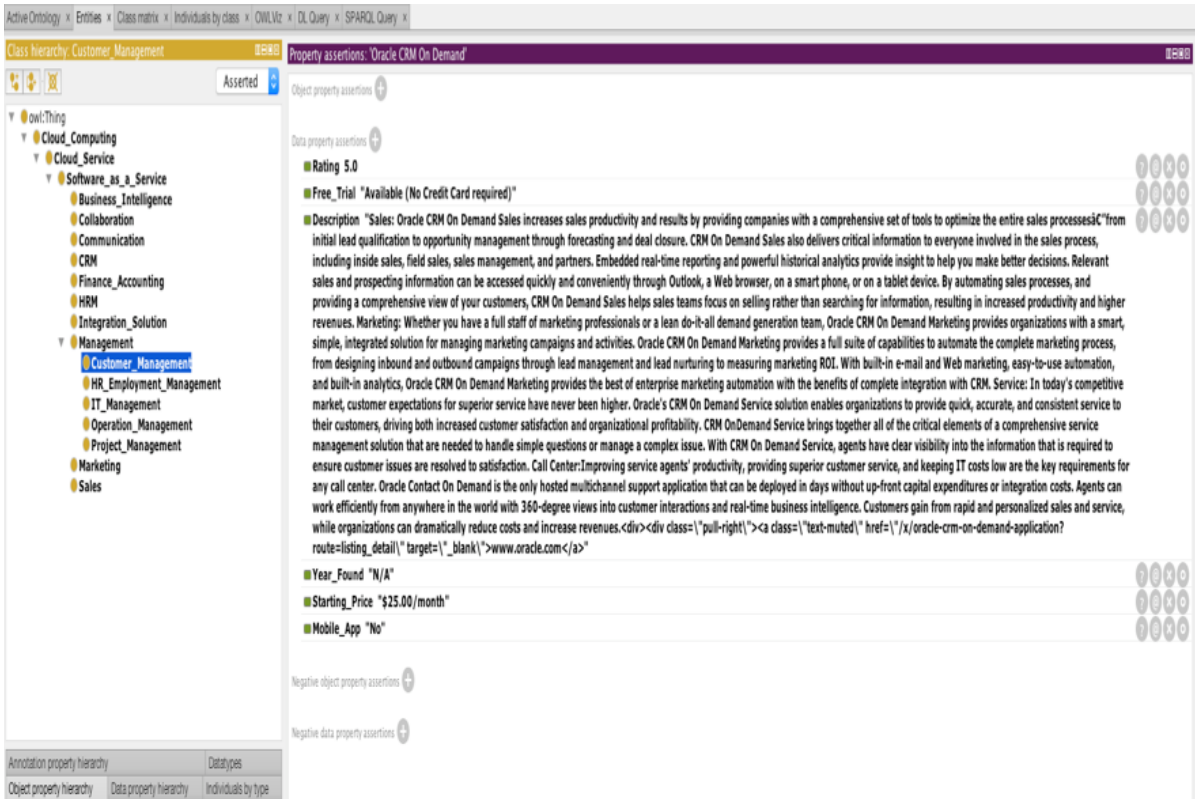


Figure 6.19: The detail of the Oracle CRM on Demand instance populated into the Customer Management sub-class of the Management class

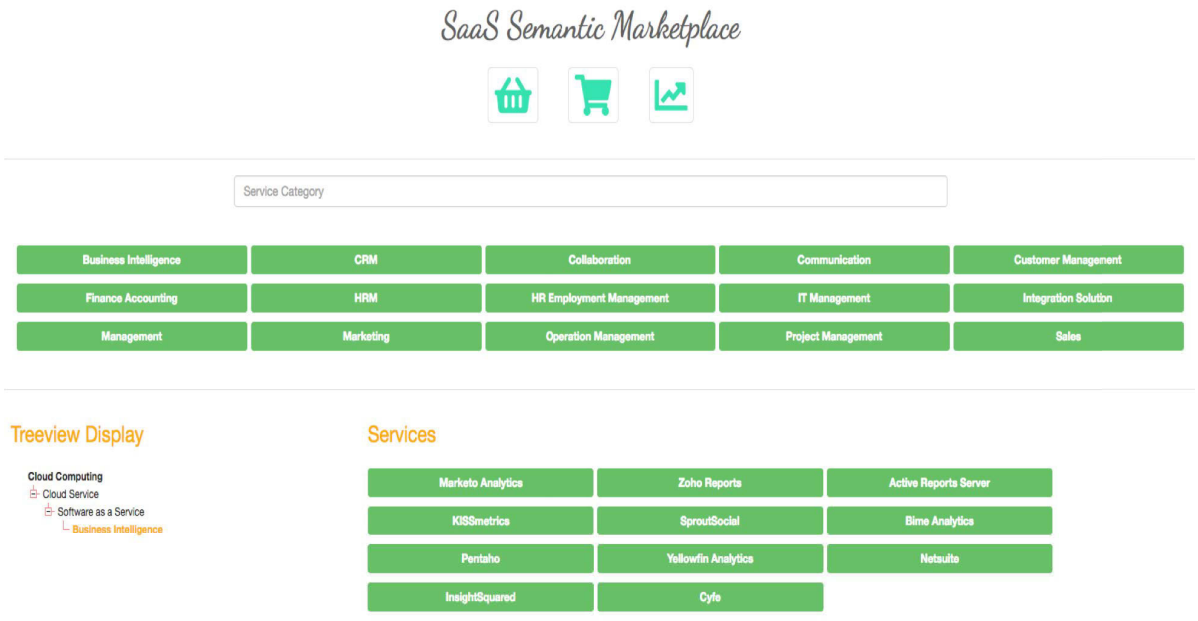


Figure 6.21: A screenshot of the modified user interface (homepage)

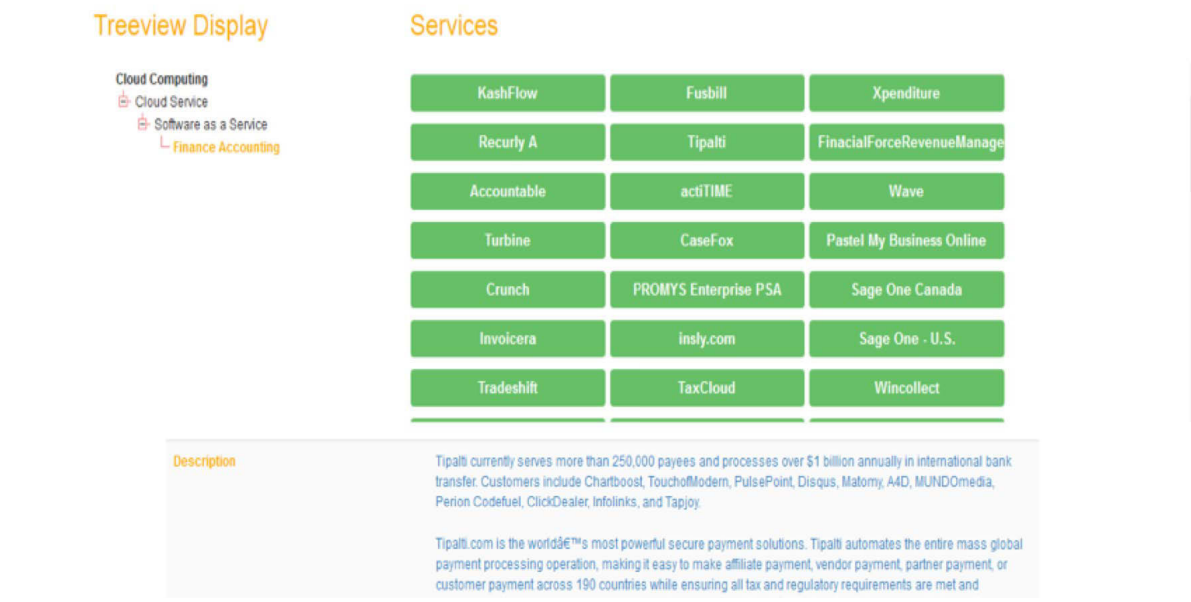


Figure 6.22: A screenshot of the modified user interface (service catalogue page)

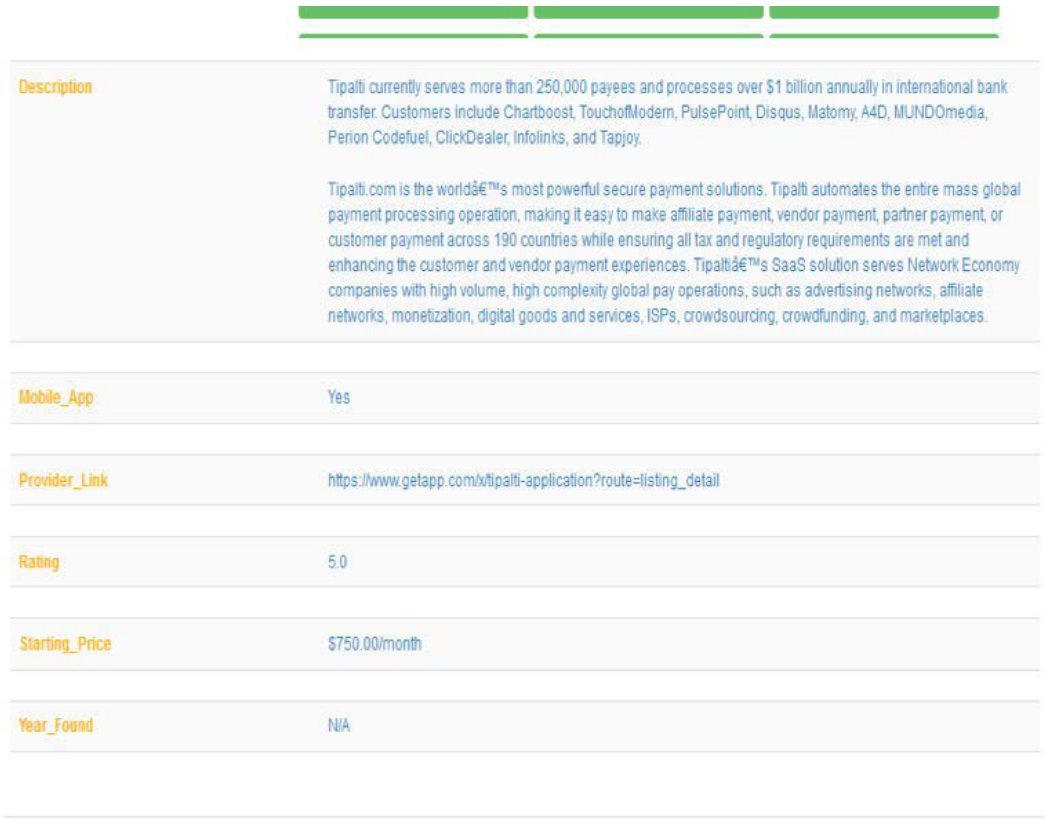


Figure 6.23: A screenshot of modified user-interface (service profile page)

6.4 Experiment results and analysis

The success of the SaaS SMARKT methodology is directly related to both the amount of data extracted from the web portals during the first phase of the methodology using HaaS, and the system accuracy in resolving the issues caused by heterogeneous web data, which is not the focus of this work. This work focuses on the development of the SaaS SMARKT knowledge base, which is the second phase of developing the SaaS architecture. The success of the SaaS SMARKT knowledge base is based on the success of the ontology mapping and population. In order to evaluate the results obtained after the ontology population phase, the relevant ontological entities (individuals, object properties and datatype properties) in the repository were gathered manually. Information on the number of service concepts retrieved and the number of service concepts correctly retrieved by our methodology was also obtained. Additionally, it is necessary to check whether the individuals and properties are properly created, and the instances correctly instantiated within the ontology, which will be discussed in detail in the following sections.

6.4.1 Evaluation

Precision, recall, and F-score measures are widely used in the area of information retrieval [42] therefore, we used these measures to evaluate the results of the ontology population. In our case, we applied these measures to the tagged extractions with regard to the instantiated concepts in the knowledge base. The equations used are as follows:

$$precision = \frac{numberofinstancescorrectlyacquired}{thenumberofinstancesacquired} \tag{6.1}$$

$$recall = \frac{numberofinstancescorrectlyacquired}{numberofinstancesexistingintheconceptualtree} \tag{6.2}$$

$$f\text{-}measure = \frac{2 * recall * precision}{recall + precision} \tag{6.3}$$

The harvested 17806 service entities are collected from three different web portals in the relational database as a service entity using the aforementioned equations. Table 2 presents the results of the set of individuals and concepts extracted from the repository to map and populate the SaaS BCLAS ontology. A set of 17806 individual mappings according to the conceptual tree of the SaaS BCLAS ontology was produced. Of these concepts, 17793 were correctly instantiated by the rules of the category and 13 were not instantiated. We thus obtain a recall of 99.59 and precision of 98.98. These accuracy measures are shown in Table 6.4

Table 6.2: Experiment results of 17806 concepts mappings based on the conceptual tree of the SaaS BCLAS ontology

Concept type	**Number of concepts correctly instantiated**	**Number of concepts not instantiated**
Instances	17793	13
Data type properties	124,551	91
Total	142344	104

Table 6.3: Average recall, precision and F-measure for the experiments.

Concept type	**Precision**	**Recall**	**F-measure**
Instances	99.87	99.06	99.46
Data type properties	99.32	98.91	99.11
Total	99.59	98.98	99.28

Table 6.4: Benchmark data summary

Key performance indicators	**Benchmark average**		
	Recall	Precision	F-measure
Information retrieval	99.59%	98.98%	99.28%
	Individuals	Datatype property	Object property
Information correctly instantiated	99.92	99.91	-

To summarise, significant values for precision and recall were achieved for the retrieved concepts. The main reasons for this are: 1) the domain is specific; and 2) the harvested data and mapping process has allowed the creation of fitted knowledge resources. Similarly, the F-measure value of the individuals and relationships shows that the ontology was correctly instantiated with a rate of 99.28%, hence, we can be confident that our proposed architecture delivers superior performance in this experiment. Table 6.4 provides a benchmark summary for the data test.

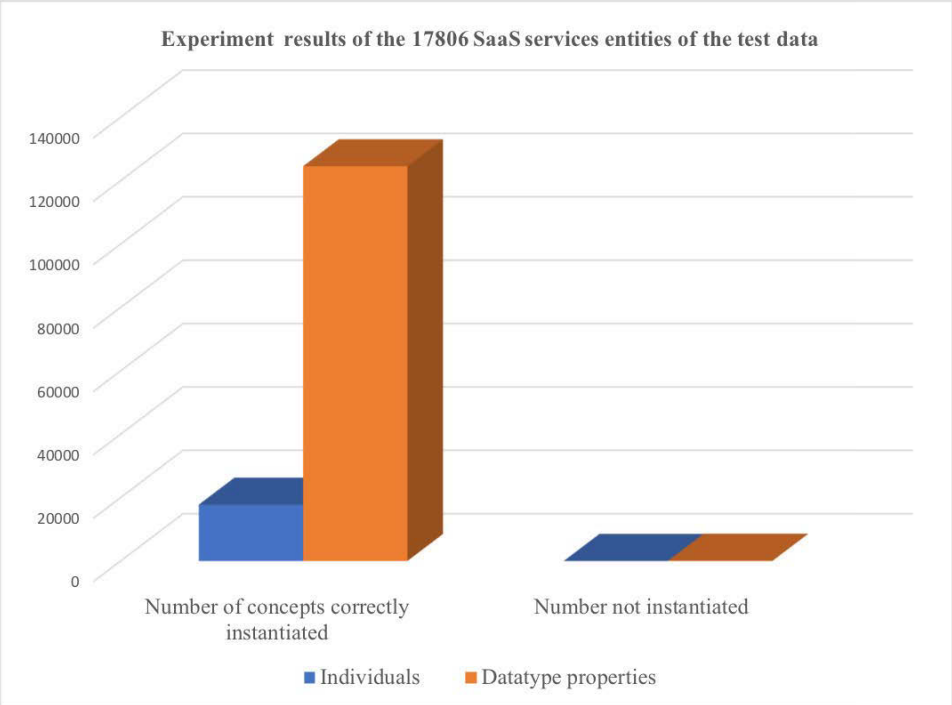


Figure 6.24: Experiment results for the 17806 SaaS service entities of the test data

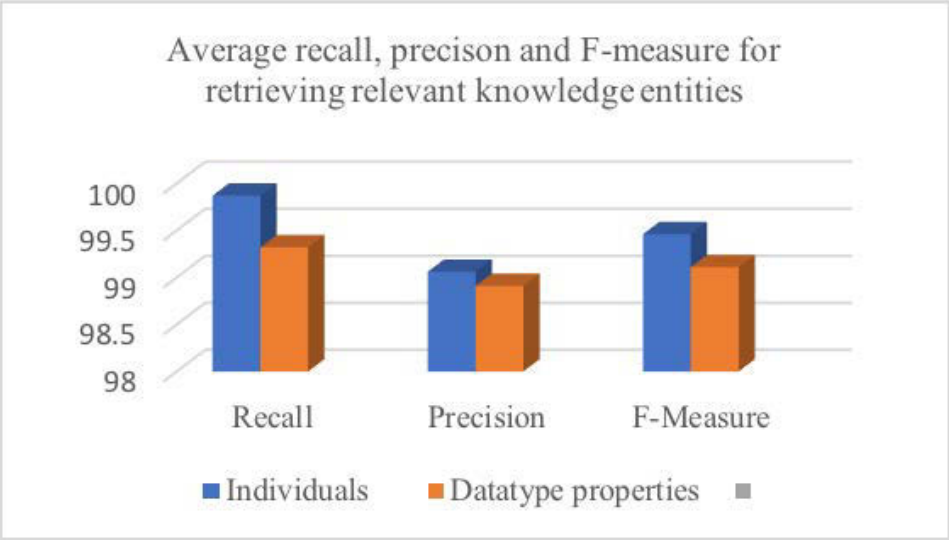


Figure 6.25: Average of recall, precision and F-measure

6.5 Discussion

This chapter proposes cloud services knowledge base architecture that offers an innovative solution for cloud services advertising and discovery. Traditionally, cloud services are advertised across different web portals which end-users discover using general search engines such as

Google. Although several approaches to cloud service discovery exist, none consider the concept of constructing cloud services ontology based on web sources or constructing cloud services knowledge base.

We conducted a thorough exploration of the literature which is summarized in Table 6.5. All the reviewed studies support the use of semantic technology, however, there is a lack of a knowledge source for cloud services. Furthermore, all the existing semantic approaches are based on the using of existing ontologies, hence there is a need for an intelligent method to construct cloud services ontology based on web sources. Therefore, in this chapter, we developed the cloud services knowledge base architecture which involves three steps: 1) collecting data ; 2) structuring data; and 3) data annotating and populating. For the validate purpose in this research, we focus on Software as a Service, whereas our proposed structure is designed to apply to all types of cloud services. Our approach creates an ontology model to classify SaaS advertisements and creates metadata to describe SaaS service offers, which can be perceived as a schema for SaaS offers.

Furthermore, we emphasise the importance of constructing domain knowledge and an ontology for SaaS advertisements, taking into account the web data. For example, the techniques used by Wikipedia for better matches rely on the domain knowledge in Wikipedia [1]. In our context, the SaaS BCLAS ontology is a language that can be used to describe a SaaS offering in precise terms. It is a schema that defines SaaS service features and prices, and it can be a shared metadata for multiple SaaS providers. The SaaS service knowledge base forms the basis for successful semantic e-commercial SaaS service data exchange and discovery capabilities online.

One significant benefit of the SaaS knowledge base architecture is that it provides a standard representation for advertising SaaS services through using vocabulary and shared metadata for many SaaS service providers and the semantic markup which provides more structured and detailed information about SaaS offers. The results showed that the SaaS knowledge base and the friendly user interface can be used to extract SaaS offers according to the domain knowledge and ontology (SaaS BCLAS).

Table 6.5: Related work summary

Author(s)	**Approach**	**Source of data**	**Outcomes**	**Service**	**Knowledge base**	**Aspect**
[55]	semantic matching	grounded on existing ontology	semantic-based discovery method	cloud services	x	technical and operational

Continued on next page

Table 6.5 – *Continued from previous page*

Author(s)	**Approach**	**Source of data**	**Outcomes**	**Service**	**Knowledge base**	**Aspect**
[22]	semantic matching	grounded on existing ontology	semantic-based discovery method	SaaS	x	technical and operational
[90]	semantic matching	grounded on existing ontology	semantic-based discovery method	cloud services	x	technical and operational
[67]	semantic matching	grounded on existing ontology	semantic-based discovery method	cloud emergency services	x	technical and operational
[35]	semantic annotation	grounded on existing ontology, using of existing web service profile and registry to publish cloud service	Using web service profile to publish cloud services which includes three attributes (input, process and output)	cloud services	Web service (UDDI registry)	functional

Continued on next page

Table 6.5 – *Continued from previous page*

Author(s)	**Approach**	**Source of data**	**Outcomes**	**Service**	**Knowledge base**	**Aspect**
[93]	semantic annotation	grounded on existing ontology	Standard cloud service profile with four attributes service name, service price, service features, and service level of agreement (SLA)	Cloud services	x	technical and operational

6.6 Conclusion

In this chapter, we have presented cloud services knowledge base architecture for cloud services advertising and discovery. The study focus on one type of cloud services namely SaaS, while it design to apply in all type of cloud services. The key contribution of this architecture is that constructing SaaS ontology based on web sources and developing SaaS knowledge base. Experiments were carried out and the results show that the ontology was correctly instantiated with a rate of 99.28%, our proposed system demonstrates a significant of the knowledge base architecture for organizing cloud services advertisements and cloud services discovery.

[1]x:implies that the study does not address this dimension

CHAPTER 7

CLOUD SERVICES TRUST DERIVED CLOUD INTELLIGENCE

Opinion mining is a growing field of research that is concerned with identifying the opinion from the text written by the human in natural language using Artificial Intelligence (AI) [77]. Recently, opinion mining has attracted the attention of the researchers in the different domains, such as stock market prediction and marketing [60]. Opinion mining (which is also known as sentiment analysis) includes the use of machine learning algorithms, which is a type of AI, to classify opinions related to a service/product, such as consumer review. The opinion mining method is essential for potential consumers to get sufficient information about the quality of services provided by the service provider; and ensuring the competitiveness of their products. Potential consumers can benefit from opinion mining method by having access to the experiences of the previous consumers that allows them to make a better choice when buying a service or a product.

Online consumer review (which is also known as business review) is an essential element of a marketing strategy as it impacts on the buying decision. [75] stated that the second most trusted source of product information is the consumer review after the recommendations from friends and family. According to [36], the consumer review is more user-oriented provides the evaluation of product/service from consumer's perspective. However, it is becoming an increasingly difficult task to read online comments and understand the consumers' opinions due to the large number of online reviews posted across different web platforms. Finally, it is challenging for businesses to derive business insights based on these reviews.

In the field of cloud computing, there is an increasing number of cloud consumer reviews for different cloud services. These reviews are posted online across various reviews web portals, such as getApp.com and serchen.com. Such web portals can be an excellent source to understand the

cloud consumers' satisfaction. Therefore, there is a need for automating the analysis of consumers' online reviews of cloud services.

As mentioned in Chapter 2, none of the existing studies considers proposing any means by which cloud consumers can choose a cloud service with the best QoS. According to [92], the Quality of Service information provided by cloud providers are insufficient and cannot guarantee the Quality of Services. With absence of QoS information at the time of making the buying decision by potential consumers, it is important to develop an intelligent yet reliable method to analyze consumers' reviews and determine the intention of the cloud reviewers. Such an approach can give the potential consumer access to the previous buyers experiences when buying the cloud service. Additionally, it could indicate the real Quality of Service perceived based on previous consumers' experience. That is, this approach could indicate the Quality of Experience (QoE). There is no research study done in this area of using sentiment of online reviews to assist cloud consumers in buying decision or deriving QoE of cloud products based on previous users' experiences.

To address these shortcomings, in this chapter, we present 'Cloud Trust Derived Cloud Intelligence' methodology to analyze cloud consumers' reviews that reflect the user's experience with cloud services. Such analysis can assist potential consumers in buying cloud services. The objectives of this methodology include:

1. Automatically harvesting cloud services reviews from several web portals.
2. Constructing the world's first cloud reviews dataset.
3. Automatically analyzing the sentiment of cloud reviews and generating the cloud polarity dataset. The cloud polarity dataset has a collection cloud reviews labeled with positive, negative, neutral, which is a training dataset for supervised machine learning.
4. Building machine learning classifiers for automatic prediction of consumer,Äôs review intention: positive, neutral or negative in the future. Knowing the review intention can indicate the real quality of service based on previous users' experiences.

The rest of this chapter is organized as follows: Section 7.1 presents the methodology used in this work. Section 7.2 demonstrates the workflow of data analysis tools. Section 7.3 explains the data analysis phases. Section 7.4 explains the experiments conducted to evaluate our methodology. Section 7.5 presents the results and the evaluations. A discussion of the findings is presented in section 7.6 Section 7.7 concludes the chapter.

7.1 Cloud services trust derived intelligence framework

In this section, we introduce an intelligent method, as shown in Figure7.1: 'Cloud services trust derived intelligence' framework for collecting online consumers' reviews related to cloud

services. We then analyze and classify the sentiment of the reviews. This framework is the first work done in this area and it tackle a fundamental issue of automating the sentiment analysis for cloud services, which is called sentiment polarity classification. In our framework, we follow the data analysis phases, which are described in detail in section 7.2. The data analysis consists of five main phases: identifying the problem; designing data requirements; pre-processing data; performing the data analysis; and visualizing data. As shown in figure 7.1, Task 1 is to identify the data analysis problem that this research considers which is related to automating sentiment analysis for cloud consumers' reviews. Task 2 involves creating the cloud services reviews dataset. Task 3 involves cleaning the data in order to avoid misleading results. Task 4 involves using sentiment analysis to generate the polarity dataset, in which each review is labelled as 'positive','negative' or 'neutral'. This polarity dataset is used as training dataset for the data classification process. Finally, task 5 is the classification process for building machine learning classifiers to automatically classify the consumer reviews into (positive, neutral or negative) in the future and drawing the conclusion about the overall consumers' experiences with cloud services.

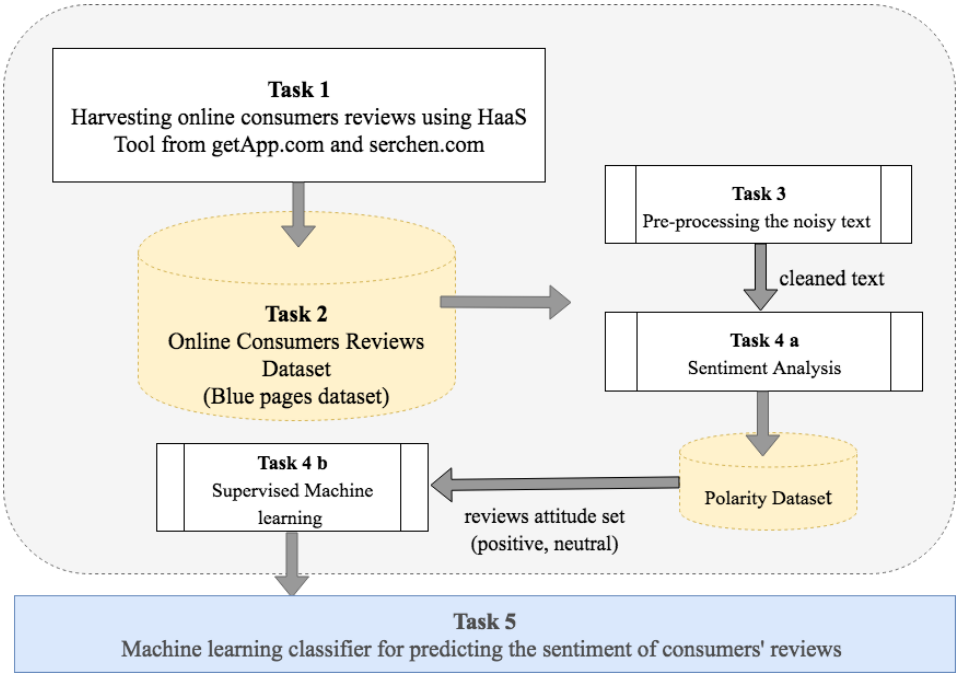


Figure 7.1: Cloud services trust derived intelligence framework

Sentiment analysis is a well-known natural language processing method which classifies the intention of the reviews as either 'positive' (admiration), 'negative' (criticize), or 'neutral'. To determine the sentiment of online consumers' reviews, Machine Learning (ML) is essential. ML techniques can be categorized as either supervised or unsupervised. This research focuses on applying supervised ML, which is based on the fact that the dataset acts as a guide to teach

the ML algorithms what conclusions it should come up with. The dataset usually consists of known input data and known expected outputs for guiding and training the algorithm. Because this research aims to predict the sentiment of online consumers reviews, in the first phase, we need a training dataset. This is a polarity dataset of reviews that has the reviews' text along with the sentiment label (positive, neutral or negative). All existing data sentiment features are combined in a prediction model that can predict whether the attitude of new reviews is positive, neutral or negative. To conclude, the sentiment for each review will determine, and the sentiment for all reviews will be aggregate as shows in figure 7.2. Then, this will be used to build intelligent classifiers to derive the satisfaction of cloud consumers. The pseudocode of the proposed framework demonstrates in figure 7.3

Algorithm 1: Pseudocode for the proposed framework

Input: Cloud service review text
Output: The sentiment polarity of Cloud service review text

1. *X= 0;*
2. *A review text is divided into M sentences, each sentence is divided into N Cases*

3. *For (i=1; i++; i<= M)*
4. *{ X_i = 0;*
5. *For (j=1; j++; j<=N)*
6. *{ $X_i = X_i + O_i$; }*

7. *$S = S + X_i$; }*
8. *If (S > 0)*
9. *Output the sentiment of the reviews is positive;*
10. *Else if (S <0)*
11. *Output the sentiment of the reviews is negative;*
12. *Else*
13. *Output the sentiment of the reviews is neutral;*

14. *End if*
15. *End if*
16. *End for*
17. *End for*

18. *Train Models using SVM classifier*

End Algorithm

Figure 7.2: Cloud intelligence

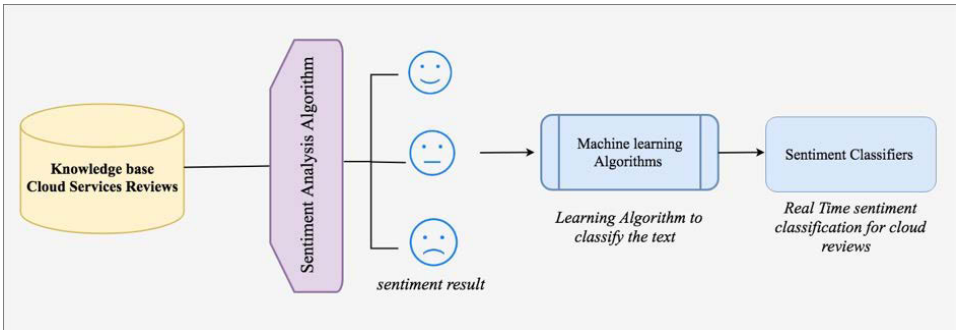


Figure 7.3: Pictorial representation of the proposed framework

7.2 Data analysis tools

Nowadays, there are several tools available for data analytics, such as R, Weka, RapidMiner and KNIME. A comparative study was undertaken by [41] to compare RapidMiner, Weka, R and Knime from different aspects, including volume of data used, response time, ease of use, price tag, and analysis algorithm and handling. The result of this study shows that KNIME is recommended for the beginner users, and that Weka is very similar to KNIME. The study shows that RapidMiner is more for experts who need programming skills and less visualization; while the R Tool is the leading tool for visualization. Also, RapidMiner is the only tool which has statistical and predictive analytics capabilities, so it can be easily used and implemented on any system.

Moreover, RapidMiner integrates the maximum algorithms of the other mentioned tools. A study undertaken by [54] stated that compared with the other data analysis tools such as R and KNIME the RapidMiner is more famous to use in business and marketing analysis. Therefore, in our research, we used RapidMiner for conducting the experiments as our focus is to analyse business reviews related to cloud services. RapidMiner provides an interactive user environment for machine learning and data mining processes and it is open source. Also, it has more than hundreds learning schemes for clustering, classification, and regression tasks.

7.3 Data analysis phases

There are five sequential phases in data analysis process which are performed to get the analysis result [64]. The phases are as follows:

1. Identifying the problem
2. Designing data requirements
3. Pre-processing data

4. Performing the data analysis

5. Visualizing data.

In our research, we follow the above mentioned phases to carried out the analysis of online consumers reviews:

1. **Identifying the problem:** We perform the data analytic techniques in order to automate the sentiment analysis of online consumers reviews related to cloud services/products. This analysis can provides the potential consumers with consumers' experiences as well as assisting them in making the most appropriate purchase decision.

2. **Designing data requirements:** To perform the data analysis process, we need a dataset of online consumer reviews related to cloud services. Therefore, we collected 9270 reviews over a period of 3 months (October-December 2017) from getApp.com and serchen.com, using HaaS Tool. Each posted review reflects the cloud consumers experience with cloud services written in English. Each post contains the text of the review and other attributes, for example, the names of reviewers and the dates that reviews were posted online.

3. **Pre-Processing data:** Before using the cloud reviews dataset, we need to pre-process the data to translate it into a specific format before applying the data analysis process. This pre-processing tasks include data cleaning and data sorting. This step will help avoid generating misleading results.

4. **Performing data analysis:** In this step, we are apply the sentiment analysis to the cloud reviews data, to generate the polarity dataset, which has each reviews labeled to positive, negative or neutral. We then use the polarity dataset as training dataset for data classification process. In the data classification process, we use four data classification algorithms in our experiment: Decision Tree, Naive Bayes, K-Nearest Neighbor and Support Vector Machines.

5. **Visualizing Data:** Deriving these steps we visualize the data.

7.4 Experiments

7.4.1 Sentiment analysis using RapidMiner

In this study, RapidMiner Studio 8.1.001 is used to apply document-level sentiment analysis on consumer text reviews related to cloud services/products. RapidMiner supports various data analytic API, one of which is AYLIEN API. The AYLIEN Text Analysis API easily extracts and analyses insights from text. The API is capable of performing document-level sentiment analysis, as well as feature-based or aspect-based sentiment analysis. The API supports four

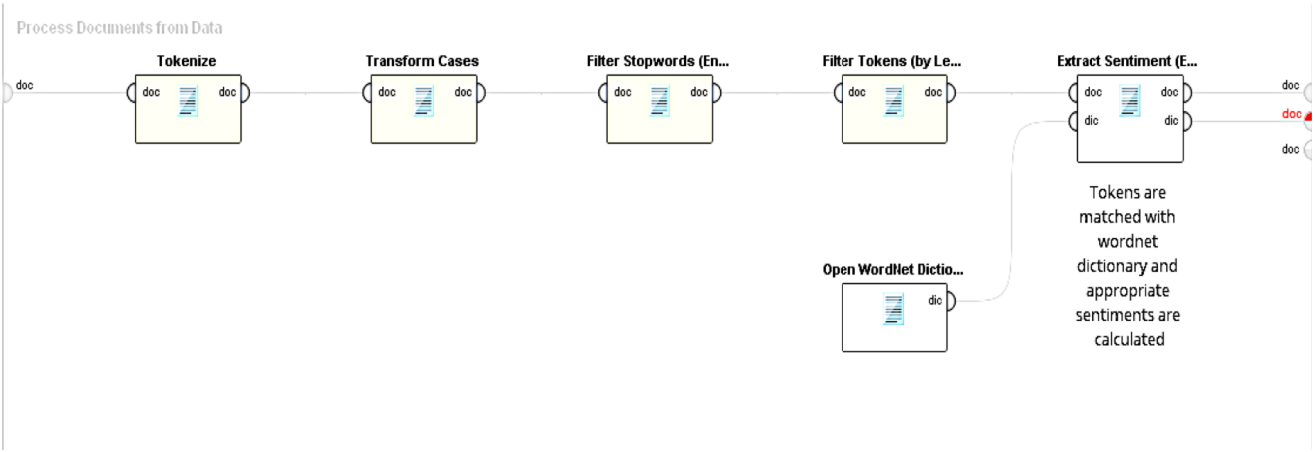


Figure 7.4: Part of sentiment analysis workflow in RapidMiner

different domains for aspect-based analysis such as cars, hotels, airlines and restaurants. For this research, AYLIEN Text Analysis by AYLIEN 0.2.0 extension is installed inside RapidMiner Studio. The primary operator (which is named Analyze Sentiment) is used to perform document-level sentiment analysis on consumer reviews. The operator extracts sentiments as being positive, neutral or negative from the consumer reviews that we supply as an input to this operator. The operator requires the following mandatory input parameters:

1. Connection: An application ID and application key needs to be set up to authenticate the user;

2. Input attribute: The customer review needs to be analysed is set in this parameter.

The pre-processing data phase is a very important phase for improving the model's accuracy. This phase primarily involves formatting and cleaning data. For the data preparation and processing phase, we used two RapidMiner extensions: "Text processing" and "wordnet". The sentiment analysis workflow in RapidMiner is shown in Figure 7.2, which is explained as follows:

1. **Step 1 Tokenize:** This operator splits the review text into a sequence of tokens. There are various ways to split the sentence into words like non-letters, specify character, and regular expression. For this research, the non-letter mode is used to split review text.

2. **Step 2 Transform Cases:** This operator transforms all the characters of the tokens into lowercase or uppercase. For example, if the review text contains words like amazing or "Amazing" or "AMAZING", then all these words are converted into the same case and are all treated the same.

3. **Step 3 Filter Tokens (by length):** This operator filters tokens based on a specified minimum and maximum character limit. Using this operator, we can filter out unnecessary

tokens such as “at”, “for”, “an” and “of”. In this study, we have filtered all small words with length less than four characters to improve the model execution time.

4. **Step 4 Extract Sentiment (English):** This operator extracts sentiments based on the SentiWordNet dictionary. The output value of the sentiment is between -1 and 1, where -1 means the text is very negative, 1 means the text is very positive, and 0 means the text is neutral. Document-level sentiment is calculated based on the average sentiment value of all tokens.

5. **Step 5 Open WordNet Dictionary:** This operator seeks the path of the dictionary so that the Extract Sentiment operator can use this dictionary to calculate sentiment level.

For this experiment, a free plan of AYLIEN API is used. This plan allows 1,000 hits per day, which is not enough for our research purpose as our review dataset has around 10000 records. Moreover, the plan performs only 60 hits per minute, which is very slow, and there is a missing domain related to software services or IT-related products for performing aspect-based sentiment analysis. We have used 10000 reviews and a system of AMD6 processor with 4GB of RAM to make this experiment.

7.4.2 Classification using RapidMiner

In this section, we present the experiments which have been undertaken hy the by RapidMiner software. Our basic mechanism for predicting the sentiment of online consumers' reviews involves using the learning supervised techniques shown in Fig 7.3. Our model consists of three main steps: Step 1: preparing the dataset, which has been described in details in data collection section; Step 2: the training process; and Step 3: the prediction process.

Firstly, we downloaded the polarity dataset. We then examined the dataset manually to ensure the quality of the data. This dataset has three sentiment labels (positive, neutral and negative). In our experiments, we used the negative, neutral and positive reviews. Secondly, we trained different algorithms using the training dataset with the known data (reviews) and known responses (labels) as input, and the prediction model as the output. Finally, we tested the prediction models to evaluate their ability to predict the sentiment of any new reviews.

7.4.2.1 Data Classification Algorithms used in the Experiments

In our experiments, we applied five different supervised machine learning algorithms: k-Nearest Neighbor, Naive Bayes, Naive Bayes (kernel), Support Vector Machines, Decision Tree, and Random Forest. A brief description of each algorithm is below:

A. **k-Nearest Neighbor Model /classifier:** k-Nearest Neighbor algorithm (which is also known as lazy learning algorithm) is the simplest algorithm of all the machine learning

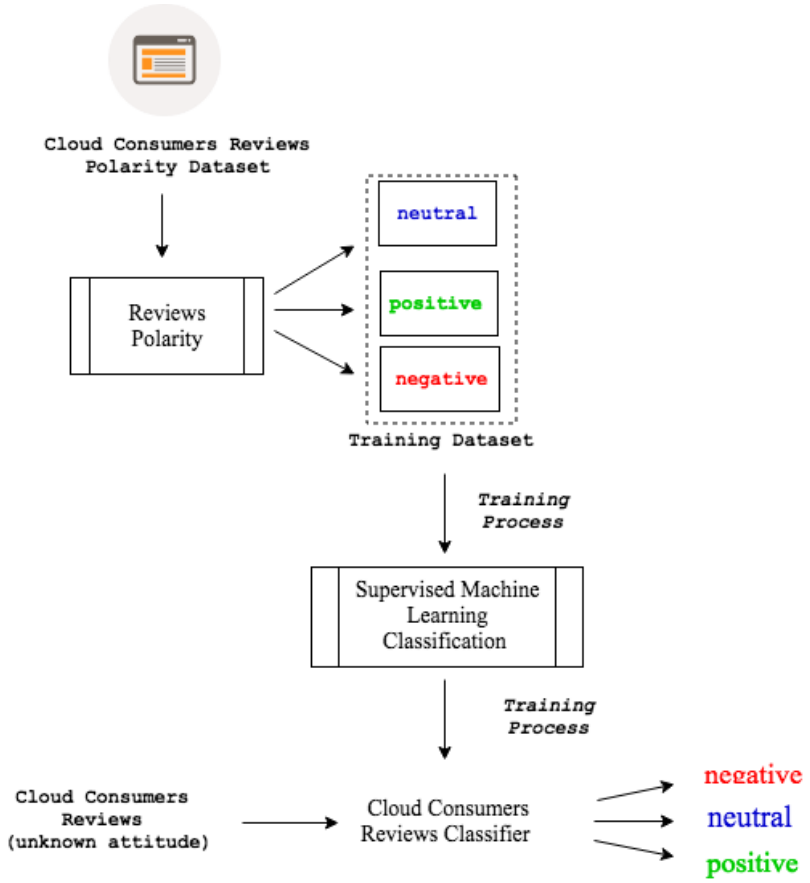


Figure 7.5: The Experiment Framework

models. This algorithm compares a given example "x" with training examples which are similar to it [94]. The training dataset is stored in a n-dimensional pattern space and the algorithm searches the area for the nearest example (k) from the training dataset that is close to the given example x. This algorithm usually is used for classification and regression. Regarding classification, it classifies according to the majority vote of it is nearest k. K is a positive integer. When k = 1, this means it is assigned to the nearest neighbor. In this research, as previously mentioned, we used the cloud polarity dataset as the training dataset, and we implemented algorithm using the K-NN operator in RapidMiner, which can be either a classification model or regression model [6]. For a classification model, the label type of the training examples is polynomial or binomial, however, if the label is numerical, this generates a regression model. The training dataset was used as input for the operator, and it also has k-parameters that specify the number of the nearest neighbors.

B. **Naive Bayes Model / classifier:** Naive Bayes classifier is a popular algorithm that is used for text classification in different domains [53]. It is a simple probabilistic classifier applying Bayes theorem that is used to predict the class of a new document. In contrast

to the other classifiers, a Naive model is efficient since it only requires a small training dataset to estimate the variance and means that need for document classification [69]. The probabilistic analysis of Naive Bayes for a document d and class c is as follows: for a document d and class c:

$$P(c-d) = P(d-c)P(c)P(d) \tag{7.1}$$

The Naive Bayes classifier is:

$$c = argmaxP(c-d) \tag{7.2}$$

In this research, we implemented the Naive Bayes model using the Naive Bayes operator in RapidMiner. The training dataset was used as input for this operator that includes document d (online consumers reviews), and class c (label: negative, or positive). The output provides the Naive Bayes sentiment classification model with online reviews.

C. **Support Vector Machines Model classifier:** This algorithm is a traditional text classification model that performs well in different domains. It was developed by [91] for binary classification, however, it has been applied successfully to many applications (e.g [37]). In this study, the Linear Support Vector Machine algorithm is used for sentiment classification. It is a hyperplane represented by the vector that separates the negative and positive training vectors with a maximum margin. In this experiment, we made use of the SVM operator in Modeling package in RapidMiner to build the SVM classifier. To train and test the classifier, we applied 3-fold cross-validation, 5-fold cross-validation and 10-fold cross-validation, so this process ran with different folds as the training set.

D. **Decisions Tree Model classifier:** The Decision Tree model is similar to an inverted tree as it has a root at the top and it grows downwards. This algorithm identifies different ways to split a dataset into segments, like branches, and these segments form the decision tree. Decision trees have been widely used for data mining and text classification, such as in [81]. In this research, we use the decision tree algorithm to build a model to predict the sentiment of online consumers reviews. We implement our model using a decision tree operator within the tree induction package in RapidMiner. The expected output is a classifier model that will assist in determining the sentiment of new cloud consumers reviews.

7.5 Results and Evaluation

Table 7.1 sshows distribution of reviews into positive, neutral and negative, with level of threshold. The results show that there is a specific range threshold for each polarity category as follows:

positive between (0 and 1), negative between (-1 and 0) and neutral is zero. The result of this analysis demonstrates that cloud services reviews are more likely to share positive experience in relation to cloud products (80.9%). Also, a less that 20% of reviewers gave negative feedback about their experience with cloud services (13.7%), whereas a small number of reviewers posted neutral comments (5.3%). In summary, the result of this stage is a polarity data set, which used in the next stage as training dataset to build machine learning classifiers.

We use the Rapidminer application to build supervised machine learning classifiers k-Nearest Neighbor Algorithm, Naive Bayes Algorithm, Support Vector Machines, Decisions Tree Algorithm and Random Forest Algorithm were performed in the data set. For the validation purpose of each classifier, we conducted a 3-fold cross validation. This means that the data is divided into three, one being the testing set and the other two being the training sets. Then, we conducted 5-fold cross validation, which means the data is divided into five, one being the testing set and the other four being the training sets.

Also, we conducted 10-fold cross validation. This means that the data is divided into five, one being the testing set and the other nine being the training sets. We used TF-IDF approach to generate the word vectors. To evaluate the sentiment classification of our classifier, we used the common index for text classification including accuracy, classification error, precision, and recall for each classifier. The results are presented in the form of tables and figures shown below. Best accuracy was achieved by two classifiers: SVM and Naive Bayes.

Table 7.1: Summary of Sentiment Analysis using Rapid Miner

Polarity Type	No. of Reviews	Threshold Level	Polarity %
Positive	7499	(0, 1]	80.9%
Neutral	494	[-1,0)	5.3%
Negative	1277	0	13.7%
Total	9270 reviews		

Table 7.2: Performance of Linear SVM Classifier

k folds cross validation	Recall	Precision	Accuracy	Classification error
3 folds	61.53% +/- 1.17%	69.39% +/- 1.49%	85.92% +/- 0.41%	14.08% +/- 0.41%
5 folds	65.52% +/- 1.92%	72.90% +/- 1.61%	87.26% +/- 0.54%	12.74% +/- 0.54%
10 folds	75.50% +/- 1.85%	68.52% +/- 2.66%	88.29% +/- 1.02%	11.71% +/- 1.02%

Table 7.3: Confusion Matrix of Linear SVM Classifier (3 fold)

	True Negative	True Neutral	True Positive	Class Precision
Pred. Negative	668	38	235	70.99%
Pred. Neutral	46	183	155	47.66%
Pred. Positive	556	271	7089	89.55%
Class Recall	52.60%	37.20%	94.79%	

Table 7.4: Confusion Matrix of Linear SVM Classifier (5 fold)

	True Negative	True Neutral	True Positive	Class Precision
Pred. Negative	730	44	197	75.18%
Pred. Neutral	30	216	164	52.68%
Pred. Positive	510	232	7118	90.56%
Class Recall	57.48%	43.90%	95.17%	

Table 7.5: Confusion Matrix of Linear SVM Classifier (10 fold)

	True Negative	True Neutral	True Positive	Class Precision
Pred. Negative	776	32	180	78.54%
Pred. Neutral	31	241	157	56.18%
Pred. Positive	463	219	7142	91.28%
Class Recall	61.10%	48.98%	95.49%	

Table 7.6: Performance of K-NN Classifier

k folds cross validation	Recall	Precision	Accuracy	Classification error
3 folds	46.58% +/- 1.07%	70.52% +/- 1.30%	24.29% +/- 0.25%	75.71% +/- 0.25%
5 folds	49.73% +/- 1.96%	70.33% +/- 0.57%	26.69% +/- 0.66%	73.31% +/- 0.66%
10 folds	52.21% +/- 2.36%	70.44% +/- 1.12%	28.45% +/- 1.40%	71.55% +/- 1.40%

Table 7.7: Confusion Matrix of K-NN Classifier (k=8) (3 folds)

	True Negative	True Positive	True Neutral	Class Precision
Pred. Negative	1266	346	6639	15.34%
Pred. Neutral	1	142	3	97.26%
Pred. Positive	3	4	837	99.17%
Class Recall	99.69%	28.86%	11.19%	

Table 7.8: Confusion Matrix of K-NN Classifier (k=8) (5 folds)

	True Negative	True Positive	True Neutral	Class Precision
Pred. Negative	1261	308	6447	15.73%
Pred. Neutral	1	178	5	96.74%
Pred. Positive	8	6	1027	98.66%
Class Recall	99.29%	36.18%	13.73%	

Table 7.9: Confusion Matrix of K-NN Classifier (k=8) (10 folds)

	True Negative	True Positive	True Neutral	Class Precision
Pred. Negative	1260	279	6310	16.05%
Pred. Neutral	1	206	6	96.71%
Pred. Positive	9	7	1163	98.64%
Class Recall	99.21%	41.87%	15.55%	

Table 7.10: Performance of Decision Tree Classifier

k folds cross validation	Recall	Precision	Accuracy	Classification error
3 folds	46.32% +/- 1.25%	70.49% +/- 1.25%	24.20% +/- 0.20%	74.90% +/- 0.20%
5 folds	49.43% +/- 1.66%	70.13% +/- 0.47%	26.62% +/- 0.63%	72.98% +/- 0.66%
10 folds	51.98% +/- 2.16%	70.28% +/- 1.10%	28.35% +/- 1.40%	71.50% +/- 1.38%

Table 7.11: Confusion Matrix of Decision Tree Classifier

	True Negative	True Positive	True Neutral	Class Precision
Pred. Negative	665	35	233	70.12%
Pred. Neutral	44	181	153	47.02%
Pred. Positive	555	269	7087	88.05%
Class Recall	51.90%	36.99%	94.02%	

Table 7.12: Confusion Matrix of Decision Tree Classifier

	True Negative	True Positive	True Neutral	Class Precision
Pred. Negative	729	43	195	74.85%
Pred. Neutral	28	214	162	52.02%
Pred. Positive	508	230	7116	90.15%
Class Recall	57.07%	43.25%	94.78%	

Table 7.13: Confusion Matrix of Decision Tree Classifier

	True Negative	True Positive	True Neutral	Class Precision
Pred. Negative	774	30	179	78.07%
Pred. Neutral	30	240	155	55.76%
Pred. Positive	460	217	7140	90.55%
Class Recall	60.46%	48.38%	95.08%	

Table 7.14: Performance of Naive Bayes Classifier

k folds cross validation	Recall	Precision	Accuracy	Classification error
3 folds	61.53% +/- 1.17%	69.39% +/- 1.49%	85.92% +/- 0.41%	14.08% +/- 0.41%
5 folds	65.52% +/- 1.92%	72.90% +/- 1.61%	87.26% +/- 0.54%	12.74% +/- 0.54%
10 folds	68.52% +/- 2.66%	75.50% +/- 1.85%	88.29% +/- 1.02%	11.71% +/- 1.02%

Table 7.15: Confusion Matrix of Naive Bayes Classifier (3 folds)

	True Negative	True Positive	True Neutral	Class Precision
Pred. Negative	668	38	235	70.99%
Pred. Neutral	46	183	155	47.66%
Pred. Positive	556	271	7089	89.55%
Class Recall	52.60%	37.20%	94.79%	

Table 7.16: Confusion Matrix of Naive Bayes Classifier (5 folds)

	True Negative	True Positive	True Neutral	Class Precision
Pred. Negative	730	44	197	75.18%
Pred. Neutral	30	216	164	52.68%
Pred. Positive	510	232	7118	90.56%
Class Recall	57.48%	43.90%	95.17%	

Table 7.17: Confusion Matrix of Naive Bayes Classifier (10 folds)

	True Negative	True Positive	True Neutral	Class Precision
Pred. Negative	776	32	180	78.54%
Pred. Neutral	31	241	157	56.18%
Pred. Positive	463	219	7142	91.28%
Class Recall	61.10%	48.98%	95.49%	

Table 7.18: Classifiers performance evaluation and comparison

	Accuracy	Classification error	Recall	Precision
SVM	88.29%	12.74%	75.50%	68.52%
K-NN	28.45%	71.55%	52.21%	70.44%
Decision Tree	28.35%	71.50%	51.98%	70.28%
Naive Bayes	88.29%	11.71%	68.52%	75.50%

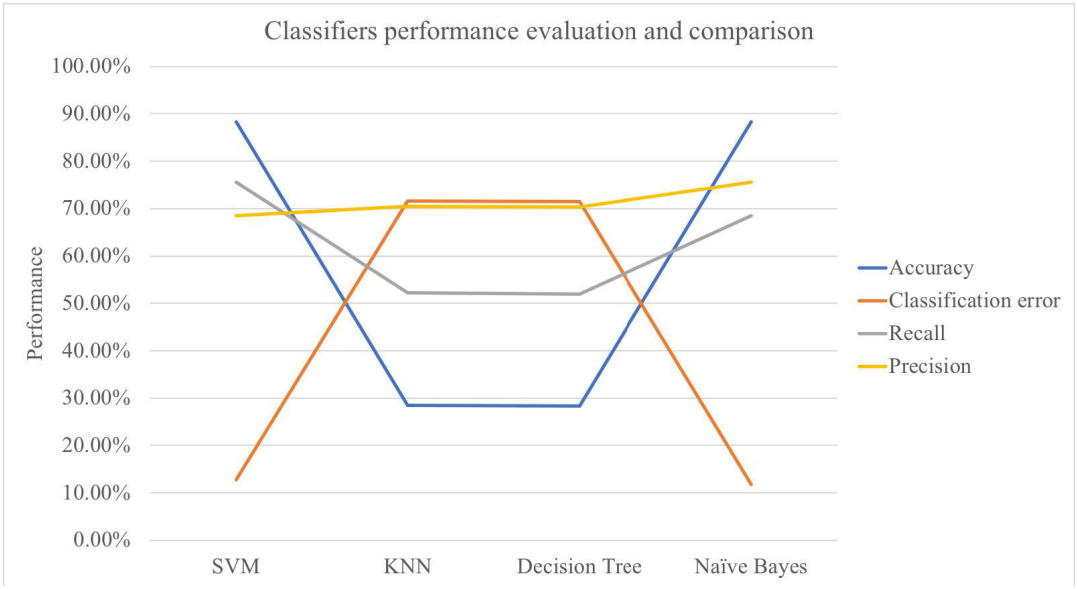


Figure 7.6: Comparison of classifiers performance

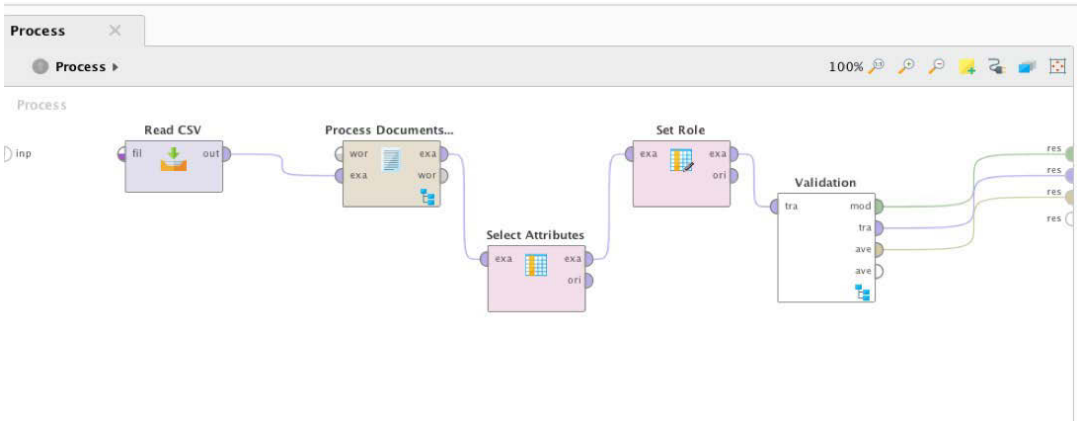


Figure 7.7: A screenshot of Rapid Miner implementation for classification process

7.6 Discussion

In this chapter, we propose that "Cloud Services Trust Derived Cloud Intelligence" will fill the research gap in the current literature in regards to a method that focuses on deriving intelligence from cloud reviews data, which spread across multiple web portals. Our proposed methods aim to provide the overall users' experiences: quality of experience. As mentioned in Chapter 2, none of the existing studies consider proposing any means by which cloud consumers can choose a cloud service with the best QoS. According to [84], quality of cloud services provided online is not sufficient and cannot guarantee the quality of services. As shown in Table 1.19, here are some existing approaches proposed regarding measuring the quality of services based on functional

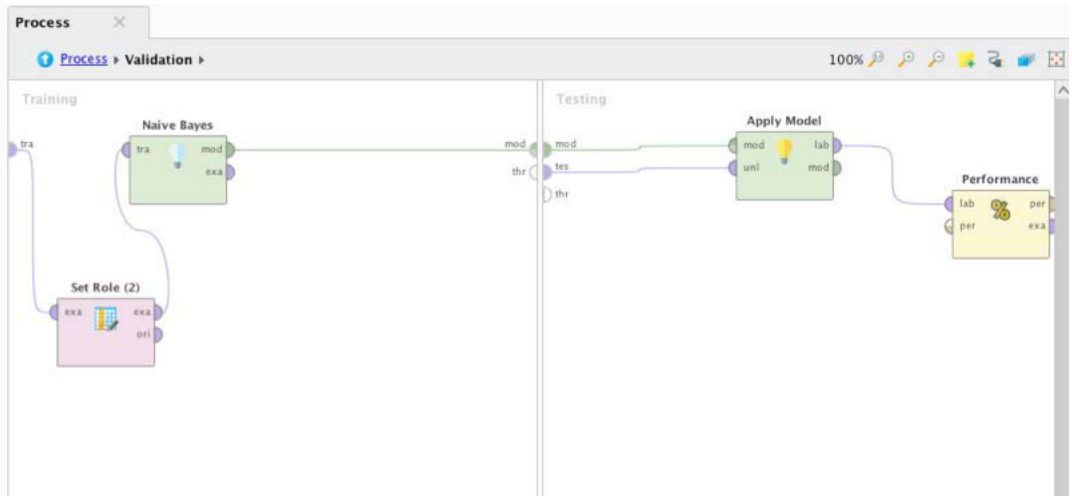


Figure 7.8: A screenshot of Naive Bayes classifier implementation

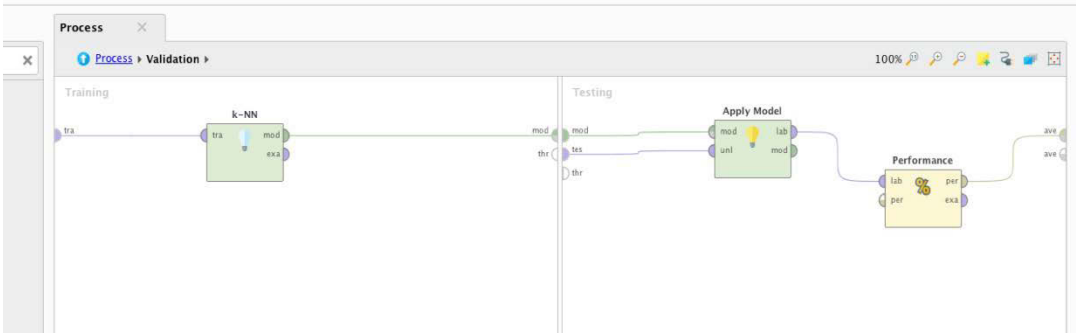


Figure 7.9: A screenshot of K-NN classifier implementation

Figure 7.10: A screenshot of Decision Tree classifier implementation

and technical aspects only. However, none of the current studies consider performing opinion mining and sentiment analysis on cloud consumer reviews to measure the real quality of service based on users' experience in order to indicate the quality of the cloud services.

We conducted a thorough exploration of the literature reviews; the summary of this is presented in table 1.19. All the studies reviewed suggest the need to provide missing information

Figure 7.11: A screenshot of SVM classifier implementation

regarding the quality of the service, which focused on technical and functional aspects. However, there is a lack of the real quality of service value at the time of making the buying decision by consumers. One way of determining the actual quality of service is by analyzing the consumers' reviews. Overall, the summary table highlights the need for an intelligence method for deriving the quality of cloud services based on consumers' experience: quality of experience. It is essential to know the previous consumers' experiences. This knowledge could assist the potential consumers to make the right decision when buying a cloud service.

Table 7.19: Comparative analysis of Existing Approaches

Source	**QoS approach**	**Service**	**Users' experience**
[85]	technical and functional aspects	cloud services	X
[76]	technical and functional aspects	cloud services	X
[23]	technical and functional aspects	SaaS	X
[22]	technical and functional aspects	SaaS	X

Therefore, in this chapter, we introduce an intelligent method as shown in figure 1.1: "cloud services trust derived intelligence". This method uses sentiment analysis and supervised machine learning to analyze the consumer reviews and determine the real quality of service based on consumers' experience. To our knowledge, there is no similar existing approach to cloud services trust derived intelligence framework in the field of cloud services. The cloud services trust derived intelligence framework consists of a five step process: 1) preparing the cloud consumers' reviews dataset; 2) processing the dataset to clean the data and make it ready for further analysis; 3) applying the sentiment analysis on the dataset, classifiying each review as "negative", "neutral" or "positive".

[1]QoS:Quality of Service
[2]QoE:Quality of Evaluation

The result of this task is polarity dataset which use as tanning dataset for the next task, and finally 4) using the supervised machine learning algorithms to build a prediction model for predicting the sentiment of cloud services reviews in the future. Furthermore, in this chapter we emphasize the important of consumers' reviews and their relation to the Quality of Services (QoS) and how exploring the consumers' reviews could solve the issue of a lack of QoS values. In our context, reviews sentiment is a measure used to predict the real quality of services based on users' experiences and recommendations. A significant of cloud services trust-derived intelligence is that provides an intelligent classifier to determine the cloud reviews sentiment; and provides the potential consumers' with information about the QoS. This information will assist the consumer with the buying decision. Such an analysis also could assist the cloud providers in improving their service based on the consumers evaluation and feedback after using the service.

The sentiment results show that 80.9% of the reviews are positive. This indicates that consumers are most likely satisfied with cloud services. The results show that the prediction accuracy of the SVM-based TF-IDF approach (10-fold cross validation testing) and Naive Bayes TF-IDF approach (10-fold cross validation testing) is 88.29%. This indicates that Naive Bayes and SVM perform better in determining sentiment than in determining other classifiers. This work also provides valuable insight into online cloud services reviews and offers the research community the first SaaS polarity dataset.

7.7 Conclusion

In this chapter, we introduce the "Cloud services trust derived intelligence" framework. This framework will be used to collect online consumers' reviews related to cloud services, and then analyze and classify the sentiment of the reviews. This framework is the first work done in this area and it tackles a fundamental issue of automating the sentiment analysis, called sentiment polarity classification. In our framework, we follow the data analysis phases. These phases (which are described in details in section 1.2) are identifying the problem, designing data requirements, pre-processing data, performing the data analysis, and visualizing data. As displayed in the figure 1.1, Task 1: involves identifying the data analysis problem that this research consider which automating sentiment analysis for cloud consumers' reviews. Task3 involves cleaning the data in order to avoid misleading results. Tasks 4 and 5 involve applying sentiment analysis to generate polarity dataset, which has each reviews labeled to positive, negative or neutral. We then use the polarity dataset as a training dataset for data classification process. The classification process is for building machine learning classifiers to automatically classify the consumer reviews into "positive", "neutral" or "negative" in the future.

CHAPTER 8

CONCLUSION AND FUTURE WORK

In this concluding chapter, we summarize the contributions to cloud service discovery that this thesis has made based on the stated objectives in chapter 1. We also discuss the important directions of future work that could be undertaken in this area of research.

8.1 Research issues addressed in this thesis

The aim of this thesis is to provide an automatic derivation of the cloud marketplace methodology that can assist potential cloud consumers to find the most reliable service advertisements of cloud services in the web environment. To efficiently locate cloud services advertising in the web environment, we developed an intelligent harvester to collect data about cloud services from various cloud services web sources. We established a reliable knowledge base for cloud services based on collected cloud services data. We then developed an intelligent method for determining whether consumer reviews are positive, negative or neutral. The intelligent method indicates the real quality of services based on users' experience: that is, the Quality of Experience (QoE) of cloud product/service. Providing the service advertising along with the QoE could assist potential consumers to make the correct decision when buying the service. The research issues that were addressed in this thesis are summarized as follows:

1. Existing research studies did not consider the importance of providing an intelligent method for harvesting cloud services advertising data from heterogeneous web portals.

2. Existing research studies did not propose constructing of cloud services ontology grounded on the web data to deal with heterogeneous cloud data on the web.

3. None of the existing research proposes building a knowledge repository for cloud services on real cloud services data.

4. None of the existing research studies focuses on deriving intelligence from cloud reviews data, which spread across multiple web portals. The derived intelligence could focus on the users' experiences.

8.2 Summary of thesis contributions

In this section, we present a summary of the thesis contributions. The major contribution of this thesis to the existing body of literature is that it proposes a methodology for an automatic derivation of cloud marketplace. The cloud marketplace has a combination of the following modules: 1. cloud services harvesting module; 2. knowledge base for cloud services module; and 3. cloud services trust-derived intelligence module. Before developing the complete solution for cloud services discovery based on the proposed methodology, this thesis presents a systematic literature review of the various proposed approaches in the existing body of literature for cloud services discovery. That review is itself an additional contribution of this thesis to the existing body of literature. In the next section, we provide a brief overview of the four contributions made by this thesis to this literature.

8.2.1 Contribution 1: A systematic literature review of Cloud services discovery

In Chapter 2, we presented an extensive literature review in area of cloud services discovery. To the best of our knowledge there is no existing survey of cloud services discovery approaches. For the purpose of discussion and evaluation, we classified the existing research approaches into two main classes based on the technology used. These classes are as follows:

- Semantic-based approach.

- Non semantic-based approach.

We further classified the existing literature on semantic-based cloud service discovery approach into five classes based on the functionality, which are as follows:

1. semantic service registry.

2. semantic service crawling.

3. semantic agent.

4. semantic service matching.

5. semantic service annotation.

We further classified the existing literature on non-semantic-based cloud service discovery approach into two classes based on the functionality, which are:

1. cloud crawler.

2. cloud services repository.

8.2.2 Contribution 2: A method for harvesting enterprise Cloud services: Harvesting-as-a-Service (HaaS)

The second significant contribution of this thesis is that it proposes an intelligent method to harvest cloud services advertisements in order to build a comprehensive listing of cloud services (cloud service dataset). Potential cloud consumers could use this dataset to locate cloud services. The method proposed in this thesis here was presented in Chapter 5. To the best of our knowledge, the existing body of literature on cloud services discovery does not propose any means for harvesting cloud services data from the web environment, and it does not suggest any means for creating the cloud services dataset.

The benefits of the harvesting method we used are as follows:

1. This method enables to harvest cloud services data from the web supported by an easy to use user-interface;

2. The method enables automatically harvest heterogeneous cloud services information from heterogeneously structured web sources;

3. The method enables the automatic organization of the harvested cloud services information; and provides a dataset containing real cloud services information, and actual cloud services reviews;

4. This method involves constructing an open source platform for harvesting cloud services which integrates different types of cloud services to construct a comprehensive listing of cloud services. To this extent, the method can be contrasted with the existing literature on cloud service discovery, which mostly focuses on semantic crawling of the web for cloud services;

5. This method involves constructing a cloud services repository which could act as a knowledge source for constructing a common ontology for cloud services in the future.

8.2.3 Contribution 3: A method for constructing a knowledge base for Cloud services semantic marketing

The third significant contribution of this thesis is that it proposes a method a knowledge base for cloud services. The method for doing this was presented in Chapter 6. To the best of our knowledge, the existing literature on cloud services discovery does not propose any means for building a knowledge base for cloud services. The features of this method are as follows:

1. The method takes into account context-dependent and heterogeneous of cloud services advertising in the web environment;

2. The method enables automatic discovery of cloud services advertisements in terms of a domain-specific service knowledge base. This stands in contrast to the existing literature on cloud service discovery, which focuses only on using semantic technologies to enhance crawling for cloud service on the web environment;

3. The method enables annotation for cloud services advertisements. We use the term "annotation" to refer to ontology mark-up languages;

4. The method enables automatic organizing for cloud services advertisements based on a domain-specific knowledge base;

5. The method proposes the first commercial repository for the cloud services marketplace.

8.2.4 Contribution 4: A method for Cloud services trust derived cloud intelligence

The fourth significant contribution of this thesis is that it proposes a method for automatically analyzing the sentiment of cloud services consumers' reviews. This method was introduced in Chapter 7. To the best of our knowledge, the existing literature on cloud services discovery does not suggest any methods for solving the issue of lack of QoS information that potential consumers can rely on it while taking the buying decision. The features of this method are as follows:

1. The method takes into account the lack of QoS information.

2. The method proposes the first polarity dateset for cloud services.

3. The method proposes deriving intelligence from cloud reviews data. This data is spread across multiple web portals. The derived intelligence focuses on the overall users' experiences.

4. The method proposes reliability of quality value of cloud services based on end-users' feedback after using the service. This is called Quality of Experience (QoE).

5. It proposes four machine learning classifiers to automatically classify the consumers' reviews and opinions as being "positive", "negative", or "neutral" to determine the polarity level.

8.3 Summary and outlook

The technological upgrading capability will require for the proposed approach to moving toward offering cloud services marketplace from which cloud consumer may purchase the cloud services products. Cloud consumer demand easy and simple way which 'one-stop marketplace' to shop for cloud services and securely conduct online transactions. While the proposed approach maintains a platform with cloud services listing and descriptions, it does not currently allow for purchasing to be done online. This functionality must consider and integrated into the proposed approach to allow for the secure purchase of cloud service to be made. Also, a new feature must consider for coping with the growing cloud services market through methods such as cloud service data classification intelligent algorithm.

8.3.1 Feasibility of the thesis in the multi-Cloud contexts

1. The proposed methodology provides the multi-cloud providers with a semantic platform for presenting multi cloud services which assist in retrieving cloud services offers.

2. The proposed methodology provides the multi-cloud providers with knowledge base which assist potential consumers to more accurately allocate a useful service.

3. The proposed methodology provides a publicly available cloud services dataset along with a complete cloud services listing, and Quality of Experience (QoE) information based on consumer reviews. The proposed QoE model is not representing the quality of service providers, however, it represents the actual value of cloud service in the market by investigating the cloud consumers comments after using the service. Therefore, the proposed QoE model will not be affected in multi-cloud provider contexts.

8.3.2 Limitation of the thesis

1. This thesis only focused on the development of a methodology for automatic derivation cloud marketplace and cloud intelligence. The proposed approach covered just the cloud services discovery issues that have defined in this thesis.

2. The proposed methodology is designed to apply to all types of cloud services, however, for evaluating the proposed methodology, we focused on one type of cloud service: Software as a Service.

8.4 Conclusion and future work

From the start of this thesis and to date, we have published nine international conference papers and submitted two journal articles on the topic under investigation. The list of publications arising as a result of the work documented in this thesis is attached in the front pages. Although we have undertaken extensive research on cloud services discovery, there is still work that needs to be done. This work is as follows:

1. An advanced algorithm to update the knowledge in the cloud services knowledge base on the real-time basis. As mentioned previously, cloud services advertising information is context-dependent and therefore has different content in different contexts. Also, the content of the service advertisement can be changed at any time by service providers. Thus, updating the knowledge basis in the real-time basis is extremely important. Our future work will involve developing an intelligent method to have a real-time knowledge base for cloud services.

2. Advanced service domain knowledge (ontology) updating methodology. In this thesis, we proposed a method to construct a domain-specific ontology for cloud services which takes into account the various and context-aware characteristic of service advertisements. However, we do not propose any means updating the ontology and adding new cloud services-related concepts using the cloud services dataset. Thus, as a part of our future work, we intend to develop a methodology to automatically generate the ontology by mining the new concepts from web data.

3. Unsupervised Text Mining: In this thesis, we proposed four classifiers for cloud reviews classification based on supervised machine learning methods. Two classifiers (SVM model and Naive Bayes model) in two experiments provided a good level of accuracy. As a part of future work, we intend to design a cloud reviews classifier using unsupervised machine learning algorithms and compare their performance against supervised methods.

References

[1] *Wikipedia, the free encyclopedia*.
https://en.wikipedia.org/wiki/Main_Page.
(Accessed on 08/22/2018).

[2] *Amazon ec2*.
https://aws.amazon.com/ec2/, April 2018.
(Accessed on 04/07/2018).

[3] *Amazon web services (aws) - cloud computing services*.
https://aws.amazon.com/, April 2018.
(Accessed on 04/07/2018).

[4] *Amazon web services (aws) - cloud computing services*.
https://aws.amazon.com/, July 2018.
(Accessed on 08/22/2018).

[5] *Connectwise uk pricing, features, reviews & comparison of alternatives | getapp*.
https://www.getapp.com/operations-management-software/a/connectwise-uk/,
March 2018.
(Accessed on 03/08/2018).

[6] *Data mining tools | rapidminer*.
https://rapidminer.com/data-mining-tools-try-rapidminer/, April 2018.
(Accessed on 04/05/2018).

[7] *Datapipe acquired by rackspace*.
https://www.rackspace.com/datapipe, March 2018.
(Accessed on 03/02/2018).

[8] *Google scholar*.
https://scholar.google.com.au/, March 2018.
(Accessed on 03/07/2018).

[9] *Ieee xplore digital library*.
https://ieeexplore.ieee.org/Xplore/home.jsp, April 2018.

(Accessed on 04/22/2018).

[10] *Knime - open for innovation*.
https://www.knime.com/, March 2018.
(Accessed on 03/05/2018).

[11] *Lightning fast data science platform | rapidminer*.
https://rapidminer.com/, March 2018.
(Accessed on 03/03/2018).

[12] *Microsoft azure cloud computing platform & services*.
https://azure.microsoft.com/en-au/, April 2018.
(Accessed on 04/07/2018).

[13] *protege*.
https://protege.stanford.edu/, March 2018.
(Accessed on 03/20/2018).

[14] *Rackspace: Managed dedicated & cloud computing services*.
https://www.rackspace.com/, August 2018.
(Accessed on 08/22/2018).

[15] *Sciencedirect.com | science, health and medical journals, full text articles and books.*
https://www.sciencedirect.com/, April 2018.
(Accessed on 04/07/2018).

[16] *Scopus | the largest database of peer-reviewed literature | elsevier*.
https://www.elsevier.com/solutions/scopus, April 2018.
(Accessed on 04/11/2018).

[17] *Scrapy | a fast and powerful scraping and web crawling framework*.
https://scrapy.org/, March 2018.
(Accessed on 03/13/2018).

[18] *Semantic scholar - an academic search engine for scientific articles*.
https://www.semanticscholar.org/, April 2018.
(Accessed on 04/11/2018).

[19] *Springer - international publisher science, technology, medicine*.
https://www.springer.com/gp/, April 2018.
(Accessed on 04/11/2018).

[20] *Top customer management software reviews 2018*.
https://www.serchen.com/customer-management/, March 2018.

(Accessed on 03/08/2018).

[21] *What is mongodb? | mongodb*.
`https://www.mongodb.com/what-is-mongodb`, August 2018.
(Accessed on 08/26/2018).

[22] Y. M. AFIFY, I. F. MOAWAD, N. BADR, AND M. F. TOLBA, *A semantic-based software-as-a-service (saas) discovery and selection system*, in Computer Engineering & Systems (ICCES), 2013 8th International Conference on, IEEE, 2013, pp. 57–63.

[23] Y. M. AFIFY, I. F. MOAWAD, N. L. BADR, AND M. TOLBA, *Ontology-based saas catalogue for cloud services publication and discovery*, Asian Journal of Information Technology, 15 (2016), pp. 4900–4915.

[24] Y. M. AFIFY, I. F. MOAWAD, N. L. BADR, AND M. F. TOLBA, *Cloud services discovery and selection: Survey and new semantic-based system*, in Bio-inspiring Cyber Security and Cloud Services: Trends and Innovations, Springer, 2014, pp. 449–477.

[25] A. AKINWUNMI, E. OLAJUBU, AND G. ADEROUNMU, *A multi-agent system approach for trustworthy cloud service discovery*, Cogent Engineering, 3 (2016), p. 1256084.

[26] H. ALHAKAMI, H. ALDABBAS, AND T. ALWADA,ÄÔN, *Comparison between cloud and grid computing: review paper*, International journal on cloud computing: services and architecture (IJCCSA), 2 (2012).

[27] A. ANKOLEKAR, M. BURSTEIN, J. R. HOBBS, O. LASSILA, D. MARTIN, D. MCDERMOTT, S. A. MCILRAITH, S. NARAYANAN, M. PAOLUCCI, T. PAYNE, ET AL., *Daml-s: Web service description for the semantic web*, in International Semantic Web Conference, Springer, 2002, pp. 348–363.

[28] A. ANKOLEKAR, M. BURSTEIN, J. R. HOBBS, O. LASSILA, D. L. MARTIN, S. A. MCILRAITH, S. NARAYANAN, M. PAOLUCCI, T. PAYNE, K. SYCARA, ET AL., *Daml-s: Semantic markup for web services*, 2001.

[29] D. K. BARRY, *Web services, service-oriented architectures, and cloud computing: The savvy manager's guide (the savvy manager,Äôs guides)*, 2012.

[30] D. BONINO, F. CORNO, L. FARINETTI, AND A. BOSCA, *Ontology driven semantic search*, WSEAS Transaction on Information Science and Application, 1 (2004), pp. 1597–1605.

[31] M. BORN, A. FILIPOWSKA, M. KACZMAREK, I. MARKOVIC, M. STARZECKA, AND A. WALCZAK, *Business functions ontology and its application in semantic business process modelling*, ACIS 2008 Proceedings, (2008), p. 110.

[32] D. BUDGEN AND P. BRERETON, *Performing systematic literature reviews in software engineering*, in Proceedings of the 28th international conference on Software engineering, ACM, 2006, pp. 1051–1052.

[33] F. BURSTEIN AND S. GREGOR, *The systems development or engineering approach to research in information systems: An action research perspective*, in Proceedings of the 10th Australasian Conference on Information Systems, Victoria University of Wellington, New Zealand, 1999, pp. 122–134.

[34] R. BUYYA, C. S. YEO, S. VENUGOPAL, J. BROBERG, AND I. BRANDIC, *Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility*, Future Generation computer systems, 25 (2009), pp. 599–616.

[35] F. CHEN, X. BAI, AND B. LIU, *Efficient service discovery for cloud computing environments*, in Advanced Research on Computer Science and Information Engineering, Springer, 2011, pp. 443–448.

[36] Y. CHEN AND J. XIE, *Online consumer review: Word-of-mouth as a new element of marketing communication mix*, Management science, 54 (2008), pp. 477–491.

[37] N. CRISTIANINI AND J. SHAWE-TAYLOR, *An introduction to support vector machines and other kernel-based learning methods*, Cambridge university press, 2000.

[38] F. CURBERA, M. DUFTLER, R. KHALAF, W. NAGY, N. MUKHI, AND S. WEERAWARANA, *Unraveling the web services web: an introduction to soap, wsdl, and uddi*, IEEE Internet computing, 6 (2002), pp. 86–93.

[39] H. DONG AND F. K. HUSSAIN, *Focused crawling for automatic service discovery, annotation, and classification in industrial digital ecosystems*, IEEE Transactions on Industrial Electronics, 58 (2011), pp. 2106–2116.

[40] H. DONG, F. K. HUSSAIN, AND E. CHANG, *A framework for discovering and classifying ubiquitous services in digital health ecosystems*, Journal of Computer and System Sciences, 77 (2011), pp. 687–704.

[41] S. DWIVEDI, P. KASLIWAL, AND S. SONI, *Comprehensive study of data analytics tools (rapidminer, weka, r tool, knime)*, in Colossal Data Analysis and Networking (CDAN), Symposium on, IEEE, 2016, pp. 1–8.

[42] J. EUZENAT, *Semantic precision and recall for ontology alignment evaluation.*, in IJCAI, vol. 7, 2007, p. 348353.

[43] R. D. GALLIERS AND F. F. LAND, *Choosing appropriate information systems research methodologies*, Communications of the ACM, 30 (1987), pp. 901–902.

[44] S. GHAZOUANI AND Y. SLIMANI, *A survey on cloud service description*, Journal of Network and Computer Applications, 91 (2017), pp. 61–74.

[45] A. S. GOLDHABER AND M. M. NIETO, *Photon and graviton mass limits*, Reviews of Modern Physics, 82 (2010), p. 939.

[46] S. GONG AND K. M. SIM, *Cb-cloudle and cloud crawlers*, in Software Engineering and Service Science (ICSESS), 2014 5th IEEE International Conference on, IEEE, 2014, pp. 9–12.

[47] A. GOSCINSKI AND M. BROCK, *Toward dynamic and attribute based publication, discovery and selection for cloud computing*, Future generation computer systems, 26 (2010), pp. 947–970.

[48] R. GUHA, R. MCCOOL, AND E. MILLER, *Semantic search*, in Proceedings of the 12th international conference on World Wide Web, ACM, 2003, pp. 700–709.

[49] T. HAN AND K. M. SIM, *An ontology-enhanced cloud service discovery system*, in Proceedings of the International MultiConference of Engineers and Computer Scientists, vol. 1, 2010, pp. 17–19.

[50] M. R. HENZINGER, R. MOTWANI, AND C. SILVERSTEIN, *Challenges in web search engines*, in ACM SIGIR Forum, vol. 36, ACM, 2002, pp. 11–22.

[51] D. J. HILL AND N. GANDHI, *Service advertising: a framework to its effectiveness*, Journal of Services Marketing, 6 (1992), pp. 63–76.

[52] R. HIRSCHHEIM, *Information systems epistemology: An historical perspective*, Research methods in information systems, (1985), pp. 13–35.

[53] T. JOACHIMS, *Text categorization with support vector machines: Learning with many relevant features*, in European conference on machine learning, Springer, 1998, pp. 137–142.

[54] A. JOVIC, K. BRKIC, AND N. BOGUNOVIC, *An overview of free software tools for general data mining*, in Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2014 37th International Convention on, IEEE, 2014, pp. 1112–1117.

[55] J. KANG AND K. M. SIM, *Cloudle: an ontology-enhanced cloud service search engine*, in International Conference on Web Information Systems Engineering, Springer, 2010, pp. 416–427.

[56] ——, *Towards agents and ontology for cloud service discovery*, in Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2011 International Conference on, IEEE, 2011, pp. 483–490.

[57] B. KAPLAN AND J. A. MAXWELL, *Qualitative research methods for evaluating computer information systems*, in Evaluating the organizational impact of healthcare information systems, Springer, 2005, pp. 30–55.

[58] T. KAWAMURA, J.-A. DE BLASIO, T. HASEGAWA, M. PAOLUCCI, AND K. SYCARA, *Preliminary report of public experiment of semantic service matchmaker with uddi business registry*, in International Conference on Service-Oriented Computing, Springer, 2003, pp. 208–224.

[59] M. KERRIGAN, A. MOCAN, M. TANLER, AND D. FENSEL, *The web service modeling toolkit-an integrated development environment for semantic web services*, in The Semantic Web: Research and Applications, Springer, 2007, pp. 789–798.

[60] Y. KIM, S. R. JEONG, AND I. GHANI, *Text opinion mining to analyze news for stock market prediction*, Int. J. Advance. Soft Comput. Appl, 6 (2014), pp. 2074–8523.

[61] A. KIRYAKOV, B. POPOV, I. TERZIEV, D. MANOV, AND D. OGNYANOFF, *Semantic annotation, indexing, and retrieval*, Web Semantics: Science, Services and Agents on the World Wide Web, 2 (2004), pp. 49–79.

[62] B. KITCHENHAM, *Procedures for performing systematic reviews*, Keele, UK, Keele University, 33 (2004), pp. 1–26.

[63] M. KLEMS, D. BERMBACH, AND R. WEINERT, *A runtime quality measurement framework for cloud database service systems*, in Quality of Information and Communications Technology (QUATIC), 2012 Eighth International Conference on the, IEEE, 2012, pp. 38–46.

[64] J. LEPPING, *Wiley interdisciplinary reviews: Data mining and knowledge discovery*.

[65] Y. LEVY AND T. J. ELLIS, *A systems approach to conduct an effective literature review in support of information systems research.*, Informing Science, 9 (2006).

[66] F. LIU, J. TONG, J. MAO, R. BOHN, J. MESSINA, L. BADGER, AND D. LEAF, *Nist cloud computing reference architecture*, NIST special publication, 500 (2011), p. 292.

[67] J. U. MAHESWARI AND G. KARPAGAM, *Ontology based comprehensive architecture for service discovery in emergency cloud*, Int J Eng Technol, 6 (2014), pp. 242–51.

[68] A. MATHES, *Folksonomies-cooperative classification and communication through shared metadata*, 2004.

[69] A. MCCALLUM, K. NIGAM, ET AL., *A comparison of event models for naive bayes text classification*, in AAAI-98 workshop on learning for text categorization, vol. 752, Citeseer, 1998, pp. 41–48.

[70] D. G. MCTAVISH AND H. J. LOETHER, *Social research*, Addison-Wesley Educational Publishers, 1999.

[71] P. MELL AND T. GRANCE, *The nist definition of cloud computing*, (2011).

[72] C. MINDRUTA AND T.-F. FORTIS, *A semantic registry for cloud services*, in Advanced Information Networking and Applications Workshops (WAINA), 2013 27th International Conference on, IEEE, 2013, pp. 1247–1252.

[73] R. MITCHELL, *Web scraping with Python: collecting data from the modern web*, " O'Reilly Media, Inc.", 2015.

[74] N. A. NABEEH, H. A. EL-GHAREEB, AND A. RIAD, *Review of cloud services discovery*, Advances in Information Sciences and Service Sciences, 7 (2015), p. 28.

[75] A. NIELSEN, *Nielsen: Global consumers‚Äô trust in ‚Äòearned‚Äôadvertising grows in importance*, Business Wire, (2012).

[76] T. H. NOOR, Q. Z. SHENG, A. ALFAZI, A. H. NGU, AND J. LAW, *Csce: a crawler engine for cloud services discovery on the world wide web*, in Web Services (ICWS), 2013 IEEE 20th International Conference on, IEEE, 2013, pp. 443–450.

[77] B. PANG, L. LEE, ET AL., *Opinion mining and sentiment analysis*, Foundations and Trends® in Information Retrieval, 2 (2008), pp. 1–135.

[78] M. PARHI, B. K. PATTANAYAK, AND M. R. PATRA, *A multi-agent-based framework for cloud service description and discovery using ontology*, in Intelligent Computing, Communication and Devices, Springer, 2015, pp. 337–348.

[79] K. PEFFERS, T. TUUNANEN, M. A. ROTHENBERGER, AND S. CHATTERJEE, *A design science research methodology for information systems research*, Journal of management information systems, 24 (2007), pp. 45–77.

[80] C. PETTEY, *Gartner says worldwide public cloud services market to grow 18 percent in 2017*.

[81] J. R. QUINLAN, *Induction of decision trees*, Machine learning, 1 (1986), pp. 81–106.

[82] V. RAJENDRAN AND S. SWAMYNATHAN, *A novel approach for semantic service discovery in cloud using broker agents*, in International Conference on Advances in Computing, Communication and Information Science (ACCIS-‚Äô14), 2014, pp. 242–250.

[83] RECORDS.NSW.GOV.AU, *Using cloud computing services - implications for information and records management ‚Äî state records nsw*, 2015.

[84] F. SAFDARI AND V. CHANG, *Review and analysis of cloud computing quality of experience*, (2014).

[85] K. M. SIM, *Agent-based cloud computing*, IEEE transactions on services computing, 5 (2012), pp. 564–577.

[86] H. SONG, D. CHENG, A. MESSER, AND S. KALASAPUR, *Web service discovery using general-purpose search engines*, in Web Services, 2007. ICWS 2007. IEEE International Conference on, IEEE, 2007, pp. 265–271.

[87] L. SUN, H. DONG, F. K. HUSSAIN, O. K. HUSSAIN, AND E. CHANG, *Cloud service selection: State-of-the-art and future research directions*, Journal of Network and Computer Applications, 45 (2014), pp. 134–150.

[88] Y. SURE, M. ERDMANN, J. ANGELE, S. STAAB, R. STUDER, AND D. WENKE, *Ontoedit: Collaborative ontology development for the semantic web*, 2002.

[89] K. SYCARA, M. PAOLUCCI, A. ANKOLEKAR, AND N. SRINIVASAN, *Automated discovery, interaction and composition of semantic web services*, Web Semantics: Science, Services and Agents on the World Wide Web, 1 (2003), pp. 27–46.

[90] A. TAHAMTAN, S. A. BEHESHTI, A. ANJOMSHOAA, AND A. M. 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, IEEE, 2012, pp. 203–210.

[91] V. VAPNIK, *The nature of statistical learning theory*, Springer science & business media, 2013.

[92] M. VARELA AND J. SANKALA, *Understanding quality issues in the cloud*, (2010).

[93] M. VASUDEVAN, P. HALEEMA, AND N. C. S. IYENGAR, *Semantic discovery of cloud service catalog published over resource description framework*, International Journal of Grid and Distributed Computing, 7 (2014), pp. 211–220.

[94] K. Q. WEINBERGER AND L. K. SAUL, *Distance metric learning for large margin nearest neighbor classification*, Journal of Machine Learning Research, 10 (2009), pp. 207–244.

[95] L. YOUSEFF, M. BUTRICO, AND D. DA SILVA, *Toward a unified ontology of cloud computing*, in Grid Computing Environments Workshop, 2008. GCE'08, IEEE, 2008, pp. 1–10.