AN ONTOLOGY BASED APPROACH FOR IMPROVING JOB SEARCH IN ONLINE JOB PORTALS

BY

OLUWATOBI SILAS DADA

DEPARTMENT OF COMPUTER SCIENCE,
FACULTY OF PHYSICAL SCIENCE,
AHMADU BELLO UNIVERSITY,
ZARIA, NIGERIA

OCTOBER, 2017

AN ONTOLOGY BASED APPROACH FOR IMPROVING JOB SEARCH IN ONLINE JOB PORTALS

BY

Oluwatobi Silas DADA, B.tech PHYSICS/COMPUTER SCIENCE (FUTMINNA) 2012
P14SCMT8070

A DISSERTATION SUBMITTED TO THE SCHOOL OF POST GRADUATE STUDIES, AHMADU BELLO UNIVERSITY, ZARIA IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF A MASTER OF SCIENCE DEGREE IN COMPUTER SCIENCE

DEPARTMENT OF COMPUTER SCIENCE,
FACULTY OF PHYSICAL SCIENCE,
AHMADU BELLO UNIVERSITY,
ZARIA, NIGERIA

OCTOBER, 2017.

DECLARATION

I declare that the work in this dissertation titled **"AN ONTOLOGY BASED APPROACH FOR IMPROVING JOB SEARCH IN ONLINE JOB PORTALS"** has been carried out by me in the Department of Computer Science under the supervision of Dr. S. E. Abdullahi and Dr. A. F. D. Kana. The information derived from the literature has been duly acknowledged in the text and a list of references provided. No part of this dissertation was previously presented for another degree or diploma at this or any other Institution.

DADA, OLUWATOBI SILAS _____ _____

(Name of Student) Signature Date

CERTIFICATION

This Dissertation entitled **"AN ONTOLOGY BASED APPROACH FOR IMPROVING JOB SEARCH IN ONLINE JOB PORTALS"** meets the regulation governing the award of the degree of M.Sc. Computer Science of Ahmadu Bello University, and is approved for its contribution to knowledge and literary presentation

(Signature) _____

_____ Date _____

Chairman, Supervisory Committee

(Signature) _____

_____ Date _____

Member, Supervisory Committee

(Signature) _____

_____ Date _____

Member, Supervisory Committee

(Signature) _____

_____ Date _____

Head of Department

(Signature) _____

_____ Date _____

Dean, School of Postgraduate Studies

DEDICATION

This Dissertation is dedicated to my beloved Parents Engr. And Mrs. G.I. DADA

ACKNWOLEDGEMENT

I wish to express my profound gratitude to God Almighty, The Most Gracious and Merciful Lord and the custodian of all knowledge, for his unfailing love throughout the rigorous process of this dissertation, without whom this research would have been impossible.

My deepest Gratitude goes to my supervisory committee, chaired by Dr. S.E. Abdullahi for his astute guidance and direction towards the completion of this work. Thank you very much Sir. In the same vein I'll like to say a big Thank you to my supervisor Dr. A.F.D. Kana for your tireless and meticulous supervision of this work. Your foresight, patience and deep knowledge and understanding has made this work a reality. I am most honored to have been supervised by you sir.

I'll also like to acknowledge the H.O.D. Computer Science Prof. S.B. Junaidu, for all your admonishment before and during this dissertation, your encouragement and insight were invaluable to me and I remain Grateful to you Sir. I acknowledge Prof. A.A. Obiniyi for all your encouragement and advice may the Almighty reward you. My warmest gratitude goes to the entire Members of Staff of the Computer Science Department, and the Defunct Mathematics Department A.B.U. Zaria some of which are Prof. O.S. Adewale, Dr. K.A. Bakare, Dr. A. Yahaya (Statistics Dept.) and Mr Olaide Oyelade. God bless you all.

I am thankful to all my colleagues, course mates and friends especially Barnabas, Eric, Gwins, Malachi, Mr. Kola, Engineer Garba, Mela, Hamza, Malam Yahaya, for all your help and support in the course of this program. God Bless you all

Finally I appreciate my parents Engr. And Mrs G.I. Dada, without your support I wouldn't have come this far, I remain eternally grateful to you. To my siblings Niyi, Tosin, Esther, Mr and Mrs Ayo, Mr and Mrs Atteh and the one and only Uche Udensi. God bless you all.

ABSTRACT

Internet has become the primary medium for Human Resource Management, specifically job recruitment and employment process. Most classical job recruitment portals on the internet rely solely on the keyword based search technique in plain text to locate jobs. However, this technique results in high recall with low precision and also without considering the semantic similarity between these keywords. Many researchers have also proposed several semantic matching approaches by developing ontologies as a reference to determine matching accuracy qualitatively, however these approaches do not quantify how closely matched applicants and employers are based on core skills. This dissertation proposes a technique that uses an ontology based approach to enhance keyword searching by leveraging on the similarity between concepts in the ontology, which represent core skills needed and required for a job in order to determine how closely matched an applicant is to a job advertisement and vice-versa . This was achieved by developing a Curriculum Vitae (CV) Ontology, annotating applicant profile and job postings using a common vocabulary and modifying the semantic concept similarity algorithm to accurately compute and rank matching score between profiles when a query is performed. The model was compared with the work of Tran (2016). The results showed that improvements were achieved in overall matching accuracy between core skills supplied by applicants and those required by employers. Improvements of 54% and 36% were obtained for Recall and F-measure respectively, over Tran (2016).

TABLE OF CONTENTS

DECLARATION II

CERTIFICATION V

DEDICATION VI

ACKNWOLEDGEMENT VII

ABSTRACT IX

TABLE OF CONTENTS X

LIST OF FIGURES XII

LIST OF TABLES XIII

LIST OF ACCRONYMS XIV

INTRODUCTION 1

1.1 RESEARCH BACKGROUND 1

1.2 STATEMENT OF RESEARCH PROBLEM 3

1.3 MOTIVATION 4

1.4 AIM AND OBJECTIVES 4

1.5 RESEARCH METHODOLOGIES 5

1.6 DISSERTATION ORGANIZATION 5

LITERATURE REVIEW 7

2.1 CONTEMPORARY HUMAN RESOURCE MANAGEMENT ON THE WEB 7

2.2 THE SEMANTIC WEB 7

2.3 ONTOLOGIES 9

2.3.1 ONTOLOGY BASED APPROACH TO ONLINE RECRUITMENT 10

2.4 BACKGROUND TECHNOLOGIES 11

2.4.1 URIS AND EXTENSIBLE MARK-UP LANGUAGE NAMESPACES (XMLNS) 12
2.4.2 XML - EXTENSIBLE MARK-UP LANGUAGE 13
2.4.3 RDF - RESOURCE DESCRIPTION FRAMEWORK 13
2.4.4 COMPLEX RDF STATEMENTS 15
2.4.5 OWL - WEB ONTOLOGY LANGUAGE 16
2.4.6 THREE FLAVOURS OF OWL 17

2.5 REVIEW OF RELATED WORKS 18

2.5.1 EXISTING SYSTEMS 18
2.5.2 RELATED LITERATURE 22

2.6 LITERATURE GAP 32

ONTOLOGY CONCEPT BASED MATCHING 32

3.1 INTRODUCTION 33

3.2 PROPOSED BUSINESS MODEL 33

3.3 PROPOSED SYSTEM ARCHITECTURE 33

3.4 SEMANTIC SIMILARITY MATCHING ALGORITHM 35

3.4.1 SEMANTIC MATCHING CONCEPT TO CONCEPT 39

3.4.1.1 Weight Allocation 39

3.4.1.2 Node Routing 41

3.4.1.3 Semantic Similarity Computation 41

3.4.2 SEMANTIC MATCHING PROFILE TO PROFILE 42

3.4.3 RANKING 43

IMPLEMENTATION AND RESULTS 44

4.1 INTRODUCTION 44

4.2 SYSTEM SPECIFICATION 44

4.2.1 HARDWARE 44

4.2.2 SOFTWARE 44

4.2.3 PROGRAMMING LANGUAGES 45

4.3 ONTOLOGY DESIGN 45

4.3.1 DESIGNING THE CV ONTOLOGY 45

4.4 THE PORTAL DESIGN 52

4.5 DATA SET 54

4.6 PERFORMANCE EVALUATION CRITERIA 54

4.6.1 Precision, Recall and F-measure 54

4.7 SUMMARY OF RESULTS AND DISCUSSION 66

SUMMARY, CONCLUSION AND RECOMMENDATION 67

5.1 SUMMARY 67

5.2 CONCLUSION 68

5.3 RECOMMENDATION 68

5.4 CONTRIBUTIONS TO KNOWLEDGE 68

REFERENCES 70

APPENDIX 73

LIST OF FIGURES

Figure 2.1: Semantic Web Layer Cake (Bizer *et al.,* 2005)..... 12
Figure 2.2: RDF Triple (Vinaya *et al.,* 2015) 15
Figure 2.3: Variable representation Of an RDF Triple (Vinaya *et al.,* 2015)..... 15
Figure 2.4: Recruitment Process Using Semantic Web Technology (Bizer *et al.,* 2005)..... 23
Figure 2.5: System Architecture Showing Resume and Concept Parser (Vinaya *et al.,* 2015).... 29
Figure 2.6: Online Job Recruitment System Architecture (Geetha and Fenelon, 2015) 30
Figure 2.7: Ontology based Core skills Matching Architecture (Tran, 2016) 32
Figure 3.1: System Architecture 35
Figure 3.2: Ontology Showing Weight Allocation between Concept Nodes 40
Figure 3.3 Applicant and Employer Skills..... 42
Figure 4.1: View of CV Ontology in Protégé Showing Class Hierarchy 48
Figure 4.2: Graph of the Entire Ontology on Ontograph..... 48
Figure 4.3: Graph of the Programming Language Class on OWLviz 49
Figure 4.4: Graph of the Web Development Class on OWLviz 49
Figure 4.5: Graph of the Software Development based Class on Owlviz 50
Figure 4.6: Graph of Web Development Class on OWLviz..... 51
Figure 4.7: Graph of Semantic Web Based Class on OWLviz..... 51
Figure 4.8: Graph of the Hardware Dev. Based class on OWLviz..... 51
Figure 4.9: Graph of HDL Class on OWLviz..... 52
Figure 4.10: GUI showing Search Results for Job search 53
Figure 4.11: GUI Showing Search Results for Applicant Search..... 53
Figure 4.12: Results of Precision Test compared to Tran (2016) 64
Figure 4.13: Results of Recall Test Compared to (Tran 2016)..... 64
Figure 4.14: Results of F-measure Compared to (Tran 2016) 65
Figure 4.15: Aggregate Results Compared to (Tran 2016)..... 65

LIST OF TABLES

Table 4.1: Summary of Search for Two Core Skills 57
Table 4.2: Summary of Search for Three Core Skills 58
Table 4.3: Summary of Search for Four core Skills 59
Table 4.4: Summary of Search for Two Core Skills (Tran, 2016) 60
Table 4.5: Summary of Search for Three Core Skills (Tran, 2016) 61
Table 4.6: Summary of Search for Four Core Skills (Tran, 2016) 62
Table 4.7: Aggregate Results of Proposed System 63
Table 4.8: Aggregate Results (Tran, 2016) 63

LIST OF ACCRONYMS

DAML:	DARPA Agent Markup Language
DARPA:	Defense Advanced Research Agency
DL:	Document Language
FOAF:	Friend of a Friend
HP:	Hewlett Packard
GUI:	Graphical User Interface
HRM:	Human Resource Management
IBM:	International Business Machine
ICT:	Information Communication Technology
IE:	Information Extraction
OIL:	Ontology Interface Layer
JOCI:	Jobs and Contact Intelligence
KIM:	Knowledge Information Management
LIP:	Learning in Project
MIT:	Massachusetts Institute of Technology
RA:	Recruitment Agency
RDF:	Resource Description Framework
SGML:	Standard Generalized Markup Language
SPARQL:	SPARQL Protocol and RDF Query Language
SME:	Small and Medium and Enterprises
TNR:	Total Number Relevant
TNRT:	Total Number Retrieved
TPR:	True Positive Rate
URI:	Uniform Resource Identifier
URL:	Uniform Resource Locator
W3C:	World Wide Web Consortium

CHAPTER ONE

INTRODUCTION

1.1 Research Background

Internet has become the primary medium for recruitment and employment processes. According to Jobberman's Online Recruitment Service Report (2015) (Nigeria's foremost online job portal), applications on its job recruitment portal increased by over 50% between May and September 2015. This clearly indicates an upward trajectory in online job portals being a major player in contemporary job recruitment process. The relevance of the Internet in job recruitment process cannot be overemphasized, more than three-quarter of the age class qualified for recruitment are active internet users and there is an increasing number of companies that publish their job vacancies on the web. (Report, 2015)

There is a large number of online commercial job portals competing to publish job postings for a fee. On the other hand, each company can publish job postings on its company's own Website (Mulder, 2010). However, publishing postings on the corporate website reaches a very limited audience, because the indexing capabilities of current search engines are too imprecise to support searches for open positions. Beside this, meta-search engines are limited in their ability to generate offers that match the precise needs of the clients since job postings are written in free text form using uncontrolled vocabulary (Mochol *et al.,* 2007). Furthermore, some dedicated search engines are entering into the market, allowing detailed queries as opposed to keyword-based search of current search engines. However, the quality of search results depends not only on the search and index methods applied, but on the relevance of the search result to the user's query. Influential factors include the process ability of the used web technologies and the quality

of the automated interpretation of the company specific terms occurring in the job descriptions. The problems of a website's machine process ability result from the inability of classical web technologies to semantically annotate the content of a given website. Semantic Web extends the Web with machine-understandable data, in addition to classic Hypertext Mark- up Language (HTML) pages. This implies that viable improvements can be made for all parties involved in the recruitment process by tapping into the capabilities of semantic web technologies based on the semantic annotation of job postings and profiles.

In human resource management, it is often necessary to locate and match individuals and positions. Examples of such tasks include human resource recruiting, selecting individuals for teams based on different skills and qualifications, and finding the right expert to acquire information or to learn from within an organization. For human resource recruiting, the Internet is being mainly used to place online job advertisements, to perform resume search, and to acquire information about skills and competencies of individuals (Dafoulas *et al.*, 2003). In order to augment and assist this process, the study and development of totally or partially automated techniques and tools have received the attention of both researchers and organizations. To effectively locate and match individuals and positions, within or from outside an organization, it is important to use semantic technology (Coulucci *et al.*, 2003). Semantic descriptions of job offers and applicant profiles allow for qualitative and quantitative reasoning about matching between available and required skills and competencies which is needed to improve the process of deciding whom to hire and assigning individuals to tasks and teams. Furthermore, semantic descriptions of applicant profiles within an organization helps improve the management of individual skills and competencies of available human resources, and provide a global view of the skills available at the organizational level. The World Wide Web is a fast growing medium of

information and services and as such, there is a need for this information and services to be shared between people and applications alike. Ontologies play a major role in supporting the information sharing mechanism by using semantic and extended syntactic interoperability of the web (Coulucci *et al.,* 2003).

1.2 Statement of Research Problem

In most classical search engines and search mechanism adopted by online job recruitment portals, there is a heavy reliance on containment of keywords in free text before search results are returned which may produce a lot of result from a submitted keyword or phrase but many of these results may be irrelevant to user's need therefore, a user may have to navigate through a large number of results to find a domain specific results. Many researchers have proposed several semantic matching approaches and have developed prototype job portals to effectively match job seekers with corresponding job postings. They achieved this by developing human resource or Curriculum vitae (CV) ontologies using controlled vocabularies to determine how applicants are closely related to job positions advertised. however, there exists outstanding problems which this dissertation seeks to address, such as quantitatively and precisely matching job seekers with available job postings based on semantic similarity between their core skills and competences in relation to the core skills and competences required for the advertised jobs and also ranking the search according to the semantic closeness between applicant profile and job profile relative to their respective core skills set.

1.3 Motivation

Increasing unemployment rate in Nigeria necessitates the need to develop a system that will ease recruitment challenges by bridging the gap between employable candidates with available jobs. Developing an approach that matches job applicant profiles with available job vacancies based on semantic similarity between the core skills and competencies of a job seeker and core skills required for a job increases the suitability of candidates to jobs they apply and vice versa and gives priority to hiring primarily based on technical competence and know-how. Furthermore, to adopt an ontology based approach in job portals in order to gain advantages of semantic web technology. The semantic web technology enables proper integration of knowledge in ontology based applications.

1.4 Aim and Objectives

The aim of this research is to enhance job search in online job portals by using an ontology based concept similarity matching algorithm to improve matching by core skills.

The specific objectives are to:

1. develop a Curriculum Vitae (CV) ontology based on core skills
2. modify the concept similarity matching algorithm
3. implement the modified concept similarity matching algorithm
4. analyze and compare the result of this work with that of Tran (2016).

1.5 Research methodologies

The following methods will be employed to achieve the stated objectives of this research.

1. Review of relevant literature in the field of study
2. Determining and Enumerating important core skills in the programming languages domain
3. Defining and modelling classes and class hierarchy for the ontology using Web Ontology Language (OWL)
4. Allocating weight values to the edge between concepts in the ontology using appropriate weight allocation functions
5. Enabling node routing by computing related weight between nodes in every path
6. Determining total sum of all pair concept computation for any two Profiles.
7. Results will be compared with work of (Tran, 2016) in terms of Precision, Recall and F-measure.

1.6 Dissertation Organization

The organization of the rest of this dissertation and a brief outline of the chapters are as follows: Chapter 2 discusses the concept of Human Resource Management on the web and how ontologies have been used to improve it and also reviews related literature. In Chapter 3, the proposed system architecture and modified concept similarity matching algorithm were presented. Chapter 4 shows results obtained from the implementation of the algorithm proposed in chapter 3 as well as evaluation of the results obtained. Chapter five gives a conclusion, summary and recommendation for future studies.

CHAPTER TWO

LITERATURE REVIEW

2.1 Contemporary Human Resource Management on the Web

Human Resource Management (HRM) covers all activities associated with employment relationships within a company, such as recruiting, hiring, time management, payroll, training and skill development, cost planning, performance monitoring and appraisal, etc. (Bizer *et al.*, 2005)

HRM is a large part of the company's responsibilities, and many choose to let Recruitment Agencies (RA) take care of some of these areas. Attracting the right candidates in today's job market is a challenging task, and for online RA businesses, HRM mainly concerns making sure their customers get the most fitting persons in their vacant job positions. It concerns everything from publishing the vacancy online in various job portals, gathering applications, grading the candidates and scheduling interviews between the applicants and the companies. It also concerns post processes like keeping qualified applicants on record for later use, matching candidates with other free vacancies and managing the skills of existing employees. (Niaphruek, 2012)

2.2 The Semantic Web

The concept of the semantic web can best be understood in the words of the father of the semantic web Berners-Lee, who was quoted to have said “I have a dream for the Web in which computers become capable of analyzing all the data on the Web. The contents, links, and transactions between people and computers. A Semantic Web, which should make this possible, has yet to emerge, but when it does, the day to day mechanisms of trade, bureaucracy and our

daily lives will be handled by machines talking to machines" (Sheth and Thirunarayan, 2013). "The 'intelligent agents' people have touted for ages will finally materialize." He further stated that basic idea of the Semantic Web is to extend the Web beyond the classic HTML pages with machine-understandable data. (Sheth and Thirunarayan, 2013). The objective is to use the Web like a global distributed knowledge store which can be leveraged by applications to perform tasks automatically. The base technologies for realizing these vision are:

1. Uniform Resource Identifiers (URIs) as a global identification mechanism for resources and terms used to describe them,
2. The Resource Description Framework (RDF) as basic data model together with its extensible Mark-up Language (XML)-based serialization syntax for publishing data on the Web.
3. The Web Ontology Language (OWL) extending RDF with terms and concepts for expressive knowledge representation. (Berners-Lee *et al.*, 2001).

The development of the Semantic Web is a joint effort of scientific (MIT, Stanford, ILRT etc.) and business institutions (HP, IBM, Nokia etc.) led by the World Wide Web Consortium (W3C). There are substantial research and development investments in semantic technologies by the major players of the IT market. The first large scale prototypes have been deployed in the domains of biomedical and bibliographic data integration. (Quan and karger, 2004).

The Semantic Web makes the web more readable for machines and software agents. Most content on the Web today is designed to be processed by humans and interacted with humans. The Semantic Web is about making the enormous and fast growing amounts of information available on the net more accessible, by letting computers sort out the parts users are interested

in and using the vast knowledge available in a more efficient manner. The semantic side of it, is getting the computer to 'understand' the meaning of the content, and this huge and difficult task is what semantic technologists are trying to address, making the future Web a better one. The Semantic Web as such, is not a particular technology in itself, but a term describing several technologies and standards with the common goal of enabling the computer to extract information from this content. (Quan and karger, 2004).

2.3 Ontologies

An Ontology is a formal and explicit specialization of a shared conceptualization, it is the key element of the semantic web. (Yahiaoui *et al.,* 2006). Tom Gruber at Stanford University, proposed that, the meaning of ontology in the context of computer science is "a description of the concepts and relationships that can exist for an agent or a community of agents." He goes on to specify that an ontology is generally written, "as a set of definitions of formal vocabulary." (Yahiaoui *et al.,* 2006).

Common components of ontologies include:

1. Individuals: instances or objects (the basic objects)
2. Classes: sets, collections, concepts or types of objects
3. Attributes: properties, features, characteristics, or parameters that objects (and classes)
4. Relations: ways that classes and objects can be related to one another
5. Function terms: complex structures formed from certain relations that can be used in place of an individual term in a statement.

6. Restrictions: formally stated descriptions of what must be true in order for some assertion to be accepted as input
7. Rules: statements in the form of an if then (antecedent consequent) sentence that describe the logical inferences that can be drawn from an assertion in a particular form
8. Axioms: assertions (including rules) in a logical form that together comprise the overall theory that the ontology describes in its domain of application. This definition differs from that of "axioms" in generative grammar and formal logic. In these disciplines, axioms include only statements asserted as a prior knowledge. As used here, "axioms" also include the theory derived from axiomatic statements.
9. Events: the changing of attributes or relations. Ontologies at a basic level, should be able to specify "isa" relationships like "A Labrador is a dog". As such, this is a Taxonomy or classification, which is like a relational model and implies "Is subclassification of".

2.3.1 Ontology Based approach to online recruitment

Utilizing Semantic Web technologies in the online recruitment comes with a huge amount of potential advantages. First of all, organizing competencies and skills in an ontology gives a lot of implicit information and relations not easily modelled in regular relational database systems. Other advantages includes the added accuracy when using a controlled vocabulary. An Ontology is crucial for online recruitment because it allows recruiters and job seekers to share a common reference system to describe contents of their documents in a non-ambiguous, simple, semantic

and a formal way. The importance of formalization is to allow for an automatic matching between supply and demand (Yahiaoui *et al.,* 2006).

Data exchange between employers, applicants and job portals is based on a set of vocabularies which provide shared terms to describe occupations, industrial sectors and job skills. In this context, the first step towards the realization of the Semantic Web online recruitment scenario is the creation of a human resources ontology (HR-ontology). The ontology is intended to be used in a job portal, not only to allow for a uniform representation of job posting and job seeker profiles but also to support and improve semantic matching (a technique that combines annotations using controlled vocabularies with background knowledge about a certain application domain).

(Mochol *et al.,* 2006).

2.4 Background Technologies

This section gives a brief highlight of the foundations that stand as a bedrock upon which the realization of the semantic web is dependent

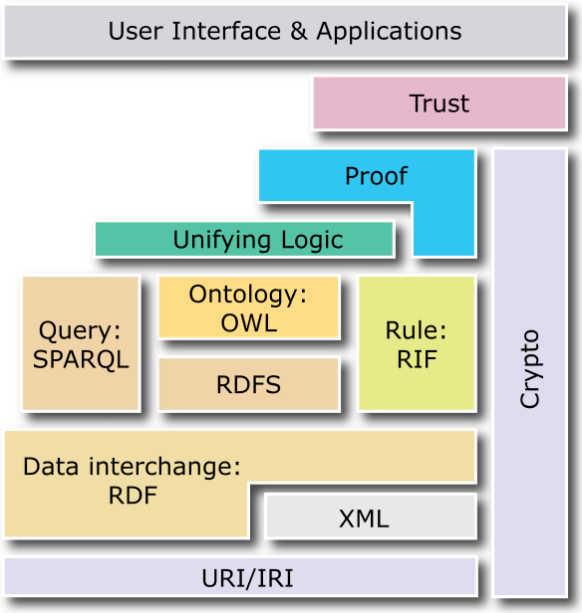


Figure 2.1: Semantic Web Layer Cake (Bizer *et al.*, 2005**)**

2.4.1 URIs and Extensible Mark-up Language namespaces (XMLNS)

Uniform Resource Identifiers (URIs) are used to provide a naming scheme. In order to describe things, there is a need to identify them. Uniform Resource Locators (URLs), are constructs most people who have used the Web are familiar with. A URL is typed into a web browser, and a user is taken to a specific website. URLs are a type of URIs. The URI thus serves as the foundation of both the current and the Semantic Web because of its use in identifying resources, and needless to say, an important principle of the Semantic Web is the ability to name and describe anything. URIs and XML provide the information representation constructs needed, including classes, properties and individuals, provide uniformity extensibility and simplicity, and are thus perfect for naming things (resources) in ontologies. A useful thing about URIs when it comes to OWL

ontologies and SPARQL querying, is the fact that a URI is divided into two parts. These two parts, called the 'base' and 'fragment' are separated by a '#'. The base URI is the same throughout the ontology, whereas the (optional) fragment identifies different parts of the ontology. Hence instead of writing the full URIs

http://folk.uio.no/fredrhal/ontology/Data_IT.owl#Object_Oriented

http://folk.uio.no/fredrhal/ontology/Data_IT.owl#Scripting

http://folk.uio.no/fredrhal/ontology/Data_IT.owl#Teaching

etc., we can use the shorthand notation of relative URIs as '#Object-oriented', '#Scripting' and '#Teaching' using the xml base attribute. (Guissé *et al.*, 2009)

2.4.2 XML - Extensible Mark-up Language

Extensible Markup Language evolved from Standard Generalized Markup Language (SGML) and became a W3C (World Wide Web Consortium) recommendation in 1998. XML is a structured, tag based meta-language made for defining languages. It is a meta-language because it allows authors to create tags that specify the structure and syntax of documents. XML is widely used for a number of applications as well as web data exchange. Some examples are program configuration files, recipe collections, Scalable Vector Graphics, musical notation and RSS web feeds. More importantly, XML provides a concrete syntax for OWL, Web Ontology Language, which is the most used language for creating ontologies for use in the Semantic Web. Although XML provides the syntax for content structure in documents, it doesn't provide any semantics with the meaning of this content. (Guissé *et al.*, 2009),

2.4.3 RDF - Resource Description Framework

The RDF Specifications are built on URI and XML technologies. The W3C RDF suite of specifications consist of:

• RDF/XML Syntax Specification

• RDF Vocabulary Description Language 1.0: RDF Schema

• RDF Primer

• Resource Description Framework (RDF): Concepts and Abstract Syntax

• RDF Semantics

• RDF Test Cases

Represented in XML, the Resource Description Framework uses triplets to make metadata statements about resources on the 'subject predicate object' form. The subject, or first element of the triplet, denotes the *resource*, the predicate denotes *properties* of the resource and the relationship between the subject and the object, the latter representing the 'property value' of the resource. An example can be:

[The Moon][isMadeOf][Cheese]

Here, the resource, the thing to be talked about, is [The Moon]. In RDF, every resource has a URI, and unique identifier. It may be a URL, but not necessarily. The property describes the relation between resources, in this case the resource [The Moon] has the property [isMadeOf], and this value is [Cheese]. Together they form an RDF statement. This triple can also be thought of as a logical formula:

P(x, y) = isMadeOf (theMoon, cheese)

Where P is a binary predicate relating the object (subject) x to the object y. This is a core fact of RDF, it only offers binary predicates.

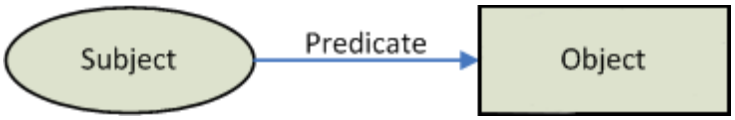


Figure 2.2: RDF Triple (Vinaya *et al.*, 2015)

RDF is often illustrated through figure 2.2, but to provide a standardized syntax that's readable for both humans and computers, RDF is serialized through XML. A viable representation of the triple in Figure 2.2 could be:

```
<rdf:RDF
        xmlns:rdf="http://www.w3.org/1999/02/22rdfsyntaxns#"
        xmlns:theMoon="http://dbooth.org/2014/moon/">
                <rdf:Description rdf:about="http://dbooth.org/2007/moon/">
        <theMoon:isMadeOf>Cheese</theMoon:isMadeOf>
        </rdf:Description>
</rdf:RDF>
```

Figure 2.3: Variable representation Of an RDF Triple (Vinaya *et al.*, 2015)

2.4.4 Complex RDF statements

RDF statements wouldn't be very valuable appearing by themselves. The real power of expressivity comes when joining statements together to form relation structures, tying together resources. Nested descriptions is where a property value is another resource. This is the most common construct, and here is an example depicting the statement "Fred has a friend called John, who lives in Zaria":

[Fred][hasFriend][John]

[John][livesIn][Zaria]

Anonymous nodes, also called blank nodes or bnodes, are resources which are not identified by a URI, and are usually assigned by the RDF processor. The anonymous node can be used to

represent the object in one statement to the subject in another, for statements like “Fred has a friend who lives in Zaria”:

[Fred][hasFriend][_node1]

[_node1][livesIn][Zaria]

RDF containers are used for collection of things, and RDF defines three resource types to hold such collections, along with properties to relate the collection to its members. The different types are *Bags* for unordered resources which may contain duplicates, *Sequences* containing ordered resources, and *Alternatives*, where only one of the members can be selected.

Reification is a mechanism that allows us to create statements of statements such as

[Fredrik][believesIn][moonCheeseStatement]

where the [The Moon][isMadeOf][Cheese] statement is *reified* into a [moonCheeseStatement] resource. This construct is best avoided if possible, as it is easy to wrongfully define resources this way. (Guissé *et al.*, 2009),

2.4.5 OWL - Web Ontology Language

The W3C Web Ontology Language (OWL) is a Semantic Web language designed to represent rich and complex knowledge about things, groups of things, and relations between things. OWL is a computational logic-based language such that knowledge expressed in OWL can be exploited by computer programs, e.g., to verify the consistency of that knowledge or to make implicit knowledge explicit. OWL documents, known as ontologies, can be published in the World Wide Web and may refer to or be referred from other OWL ontologies. OWL is part of the W3C’s Semantic Web technology stack, which includes RDF, RDFS, SPARQL, etc.

The current version of OWL, also referred to as “OWL 2”, was developed by the [W3C OWL Working Group] (now closed) and published in 2009, with a Second Edition published in 2012.

OWL 2 is an extension and revision of the 2004 version of OWL developed by the [W3C Web Ontology Working Group] (now closed) and published in 2004. The deliverables that make up the OWL 2 specification include a Document Overview, which serves as an introduction to OWL 2, describes the relationship between OWL 1 and OWL 2, and provides an entry point to the remaining deliverables via a Documentation Roadmap.

OWL is a W3C project trying to standardize an ontology framework more capable than RDFS. OWL can be seen as the next version of DARPA Agent Markup Language (DAML) + Ontology Inference Layer (OIL), as it is built upon this syntax. DAML+ OIL is an ontology project developed by US Defense Advanced Research Projects Agency (DARPA), and OWL development is done by many of the same people who developed DAML. It is layered on top of RDF, and can be used to describe the set of facts that makes up an ontology. It uses RDF namespaces to integrate the many different and incompatible ontologies that exists. (Guissé *et al.*, 2009),

2.4.6 Three flavours of OWL

The primary goal of OWL is to be able to describe concepts and relationships as unrestricted as possible, but also allow computers to infer and reason about them. If OWL were to put no restrictions on RDF statements, the advantage of computability and decidability would be lost, and so OWL is divided into a family of three languages; Full, (Document Library) DL and Lite. OWL Full is the complete language, while the others are subsets or restrictions of OWL Full. The reason for three versions is that OWL Full is the union of OWL syntax and RDF, and allows an 'RDF datastore' (a collection of RDF statements), and is complex enough to give a logical reasoner serious computational problems. The two subsets are less expressive, but reduces the demands on processing drastically. OWL DL supports Description Logics, a logic family

applying carefully selected restrictions on what can be expressed, to gain computability. This makes sure a (Document Library) DLprocessor can execute a computation, and be sure to get an answer, hence, it is complete and decidable. From the OWL Language Reference document. (Guissé *et al.*, 2009),

2.5 Review of Related Works

2.5.1 Existing systems

Lingway HR Suite solution (Philip and Mazancourt, 2016) is one of the offers currently on the market that enables users to automatically analyze applicants' resumes and match them with the job offers available. LEA 4.0 release is HRXML standards compliant.

The solution is made up of 3 modules:

1. LEA CV (Automated resume acquisition): Automated management of incoming resume flows: allows users to capture resumes in print format by OCR, and resumes in Word and PDF format sent by email or through corporate websites. They are then classified in the resume base.
2. LEA Job Offer (Automated Job offers analysis): Allows users to automatically control the quality of job offers according to marketing and legal (affirmative action/equal opportunity) criteria and to find the most relevant key words for job offers.
3. LEA Search (Semantic search engine on resume and job offers): Semantic search engine for job offers and resumes (matching): enables users to automatically find the resumes that match the job offers, and to offer applicants jobs that correspond to their resumes.

The Ontotext's Jobs & Contacts Intelligence (JOCI) (http://www.innovantage.co.uk) Recruitment Intelligence through Semantic Web Technologies (Innovantage, Fairway Consultants) gathers recruitment related information from websites of United kingdom (UK) organizations, offers services on top of this data to recruitment agencies, job portals, and other. JOCI uses Knowledge and Information Management (KIM) for information extraction (IE), text mining, and makes use of a domain ontology to support the IE process, to structure the knowledge base with the obtained results, and facilitate semantic queries. Innovantage launched in early 2006 to exploit semantic enabled search technologies to deliver actionable business intelligence harvested from the internet. The first product tracks and monitors online recruitment advertising accelerating lead generation, business development and market research for recruitment agencies and job boards. The technology platform aggregates and normalizes jobs posted on over half a million direct employer websites as well as many of the UK's leading job boards. Clients access and manipulate data through an internet browser or at a more sophisticated level through custom XML feeds. Innovantage has invested heavily in next generation Web technologies based on Natural Language Processing. This solves the problem of finding and structuring jobs posted on many thousands of direct employer websites. This is an exciting and evolving technology and will be the basis of many of Innovantage's new products.

The eCCO (eCompetences and Certifications Observatory) System (Mustafa *et al.,* 2007) is an eCompetence Management Tool Based on Semantic Networks. According to several national Institutions', ICT company associations' and the Italian Ministry of Innovation and Technology's requests for standardization of Information and communication Technology (ICT) job profiles, an ICT job profiles model was defined. It is based on ontologies principle to describe "Knowledge Objects", "Skills", "Competences" and their relations. The model takes

into account the several formalizations from national and company ICT job profiles frameworks and refers to the scientific literature about ontology and competences definitions. A software tool has been implemented on the basis of the model defined. It allows collecting individual users' job profiles and analyzing the gaps against a set of ICT standard profiles. The semantic network behind gives the system flexibility. Furthermore, the system enables the dynamical enrichment of the semantic network by uploading new skills by users, who can also suggest linkages between the new nodes. The system administrator will later evaluate and accept them.

The PoCeHRMOM (Project omtrent Competenties en functies in eHRM voor technologische toepassingen op het Semantisch Web door Ontologie en Meertalige terminologie) (http://cvc.ehb.be/PoCeHRMOM/Home.htm) project aims to provide small to medium sized enterprises (SMEs) with competency management possibilities. Existing eHRM applications (e.g. automatic translation of job openings) are mostly relevant for companies with an in place competency management. The development of this knowledge is a tough job, which scares Small and Medium Enterprises (SMEs) away from competency management. As a result, they cannot benefit from existing (and continuously improving) eHRM applications. The main focus of this project is to develop a common database that SMEs can use to build their own competency profiles.

KOWEIN Ontology (Dittman and Zeleweski, 2004) is a German ontology not developed further, focusing not only on HR issues, but also consisting of a generic top level ontology with a domain specific profile of competence and skills management. Its strength lies in using FLogic as an ontology formalism.

Knowledge net Ontology (Bizer *et al.*, 2005) is based on the KOWIEN ontology and the German translation of HRXML, this ontology was developed within the Knowledge Nets project at the Free University of Berlin. It contains national and international classifications for jobs and branches.

ePeople ontology (Biesalski, 2006) was developed at DaimlerChrysler 2003-2006 in KAON, in the context of the ePeople project (in cooperation with FZI Karlsruhe), aiming at established an integrated competence management system at Daimler Chrysler. It primarily represents Skills, Skill Profiles of Employees and Job Skill Requirements in order to allow for exploiting similarity measures on competency profiles for skill profile matching.

Learning in Project (LIP) ontology (Schmidt, 2005) was developed within the EU project learning in Process (2002-2004). In order to support embedding learning processes into work and business processes. Its focus is on the automat ability of on demand learning support and is directed towards relating employees, their organizational context and relevant learning resources (which can range from learning objects up to immature documents or colleagues). It specifically aims at bridging the gap between e-learning, knowledge management, human resource development and performance support.

CommOnCV (Schmidt, 2006) is a project, which aims at dealing with the problems of e-recruitment by considering a new approach based on competency management. The idea consists in allowing a job seeker (respectively a recruiter) to identify and formally represent the competencies underlying its Curriculum Vitae (respectively its job offer). These competencies, which allow to make knowledge, skills, abilities, traits and motives acquired by a person

(respectively required for a job) explicit, are then used to refine the matching process between "supply and demand". From a technical viewpoint, these competencies correspond to annotations, which are formally represented by using Semantic Web languages.

2.5.2 Related Literature

Bizer *et al*., (2005) affirm that using Semantic Web technologies in the domain of online job recruitment could substantially increase market transparency, lower the transaction costs for employers, and change the business models of the intermediaries involved. In their work, they described how online recruitment processes can be streamlined using Semantic Web technologies. They analyzed the realization chances of such a scenario from the technological and business viewpoint, they suggested improvements for several steps by enhancing job postings and job applications with annotations using controlled vocabularies. They further described a prototypical implementation of the required technological infrastructure for the scenario and analyzed the potential economic impacts, the interests of the involved parties, and potential changes to their business models. However their proposed model can only come to full actualization and implementation on a set of assumed scenario, without which the actual technical implementation of their model is impossible. They assumed three stages of the labour market scenario which are information providers, information aggregators and information consumers. They proposed that information providers should publish data in RDF format. However given the current technical realities on ground today very few information providers actually publish their data in RDF format as most information providers have their data in plain HTML formats and as a result, though novel an approach they have proposed, its actual realization may not be feasible at this material time. It is important to note that most of the subsequent researches that have been carried out in the area of using ontologies in the job

recruitment process have taken their bearing from the proposals in the work of Bizer *et al*., (2005). This dissertation is similar to the proposals made by Bizer *et al*., (2005) in that we also seek to enhance job search on a job portal using a controlled vocabulary, However unlike their work, this work intends to propose a new business model upon which our architecture will be founded. This work seeks to incorporate a technique of a more efficient query relaxation which isn't present in their work. This work also proposes an approach that matches and displays a ranked list of job seekers to employers and vice versa strictly based on the semantic similarities between the core skills and competencies supplied in their profiles.

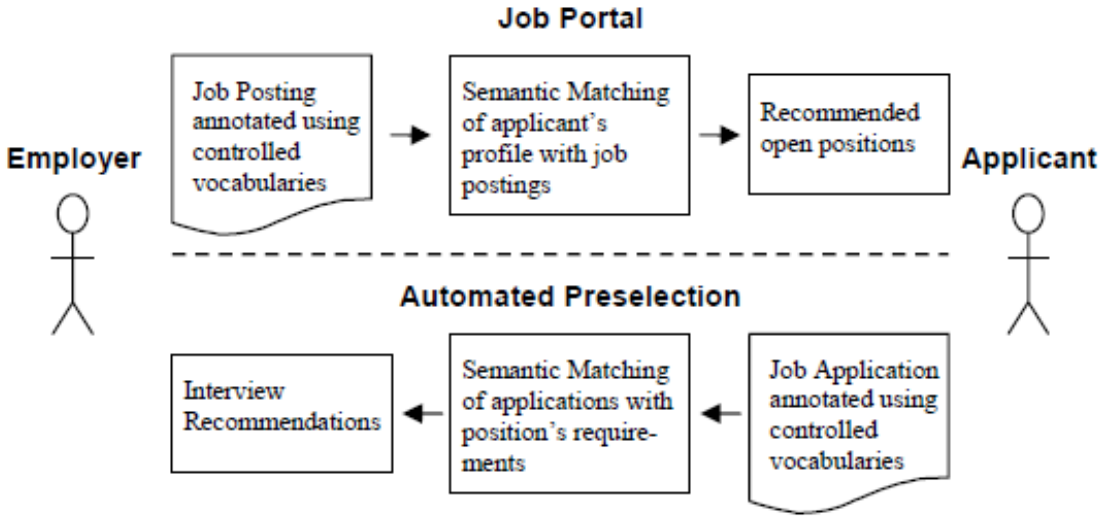


Figure 2.4: Recruitment Process Using Semantic Web Technology (Bizer *et al*., 2005)

Mochol *et al.,* (2007), proposed a prototype job portal which uses semantically annotated job offers and applicants. Using Semantic Web technologies substantially increase market transparency, lower transaction costs and speeds up the procurement process. However adding semantics is not a panacea for everything. They identified some outstanding problems in job search using the system and outline how the technique of query approximation can be the basis

for a solution. Like Bizer *et al*., (2005) they also assumed three stages of the labour market scenario viz; information providers, information aggregators and information consumers. They developed a human resource ontology which was intended to be used in a job portal, not only to allow for a uniform representation of job posting and job seeker profiles but also to support semantic matching (a technique that combines annotations using controlled vocabularies with background knowledge about a certain application domain) in job seeking and procurement tasks.

Semantic Matching is a technique which combines annotations using controlled vocabularies with background knowledge about a certain application domain. In their prototypical implementation, the domain specific knowledge is represented by concept hierarchies like skills, skill level classification, occupational classification, and a taxonomy of industrial sectors. They claim that having this background knowledge of the recruitment domain (i.e. a formal definition of various concepts and specification of the relationships between these concepts) represented in a machine understandable format, allows them to compare job descriptions and applicant profiles based on their semantic similarity instead of merely relying on the containment of keywords like most of the contemporary search engines do. In their Human Resource (HR)-scenario, their matching approach utilizes metadata of job postings and candidate profiles and as the matching result. Inside both job postings and applicant profiles, they grouped pieces of information into "thematic clusters", e.g. information about skills, information regarding industry sector and occupation category, and finally job position details like salary information, travel requirement, etc. Each thematic cluster from a job posting is to be compared with the corresponding cluster from an applicant profile (and the other way round). The total similarity between a candidate profile and a job description is then calculated as the average of the cluster similarities. The

cluster similarity itself is computed based on the similarities of semantically corresponding concepts from a job description and an applicant profile. The taxonomic similarity between two concepts is determined by the distance between them which reflects their respective positions in the concept hierarchy. Their work is similar to this work, in that this work seeks to annotate job postings and job seeker profile using controlled vocabulary from the human resource ontology however, unlike their architecture, the proposed architecture for this work introduces an approach that matches applicants to jobs and jobs to applicant by deploying a modified semantic concept similarity matching algorithm to match applicant profiles with job profiles primarily based on core skills.

Jike and Yuhui (2008) proposed a smart ontology based similarity matching algorithm based on semantic distance. The algorithm takes two concepts as input and computes a semantic similarity as output in four macro steps:

Step 1: Weight allocation. Determine weight according to the relationship between root node of ontology and other concept nodes.

Step 2: Node routing table generation. Record all paths between root node and concept nodes, and generate node routing table.

Step 3: Semantic distance computation. Compute semantic distance according to the node routing table. The semantic distance is the sum of weight between the concepts that have the inheritance relationship.

Step 4: Semantic similarity computation. Constructs similarity function, and computes semantic similarity between Concepts based on semantic distance.

This work adopts and modifies Jike and Yuhui (2008) algorithm, which will for the first time be applied in the CV ontology domain.

Fazel and Fox (2009), combined different kinds of matchmaking strategies to improve the recruitment process. They formally defined job seekers and job advertisements using a skill ontology, a deductive model was used to determine the kind of match between a job seeker and an advertisement. Thus, a description logic-based classification is performed to determine the type of match between a job seeker and an advertisement, and then ranking of applicants with partial match is done based on their similarity degree. Whilst Mochol *et al*., (2007) present a scenario for supporting the recruitment process with semantic web technologies, their approach uses a similarity measure to evaluate the degree of match between job offers and applicants. Fazel *et al*., (2009) approach on the other hand is different from theirs in that in addition to using different similarity measures, it uses logic deductive facilities which present high precision and recall. Also, desired skills and competencies are not considered in their work.

Niaphruek (2012) proposed a semantic job recruitment system that introduced an information model to represent CV (Curriculum Vitae) by creating a CV ontology based on existing well known standards such as Europortfolio, Europass CV description of career, Friend of a Friend (FOAF) and (VIVO). Secondly in his work, he presented a new business model and evaluated it against other popular online recruitment systems in order to prove the feasibility of his approach. He further claims that his system is quite better than others in supporting RDF, Ontologies and Vocabulary. Also supporting SPARQL and enabling linked data by SPARQL endpoint. this work is similar to his work in that a business model similar to his work will be proposed, however this work seeks to propose a business model that is superior and more efficient to his, because, unlike Niaphruek's (2012) business model where all prospective employees are required to develop their own personal website, where they provide their own personal profiles

from which they can share in various formats as need may arise. This dissertation proposes a mechanism which will allow all prospective employee profiles to be created on a single portal, which makes it smatter, and also eliminates overhead incurred in the process of creating and maintaining websites by prospective employees, some of whom might not be technologically savvy nor financially buoyant to maintain one.

Ionecusu *et al.,* (2012) proposed a system that focuses on the semantic modeling of online recruitment documents, then also proposed an ontology for the database field. The proposed ontology is inspired from the common parts, which they considered significant of CVs and job offers in the field of databases. Furthermore they bring clarifications regarding the essential concepts for the semantic modeling of online recruitment systems, field ontology, semantic annotation, semantic indexing, and semantic association of documents.

Kavidha and Saradha (2013) proposed a framework for building intelligent interoperable applications for employment exchange system by collaborating among distributed heterogeneous data models using semantic web technologies. They claim that the objective of the application development using semantic technologies is to provide a better inference for the query against dynamic collection of information in collaborating data models. They further stated that the employment exchange system provides interface for the users to register their details thereby managing the knowledge base dynamically. In their work, semantic server transforms the queries from the employer and jobseeker semantically for possible integration of the two heterogeneous data models to drive intelligent inference. The semantic agent reconcile the syntax and semantic conflicts which exists among the contributing ontologies in different granularity levels and performs automatic integration of two source ontologies and gives better response to the user.

The benefits of building interoperable application using semantic web are; data sharing, reusing the knowledge, best query response, independent maintenance of the model, extending the application for extra features.

May and Yew (2015) proposed an approach to job matching with user provided information, referred to as parameters. They stated that a few common parameters for job matching includes domain of job, job title, position, knowledge, experience, location, salary etc. in their approach, Predefined rules assign weighting factor to each parameter and defining how matching results could be filtered and rank to produce job matching result. They further explained the methodologies used to auto-filling in places where a candidate has missed out certain important information in their resume. The auto-filing utilized the self-learning engine to collect information, analyses the data and auto generates standard template for different categories group. The standard template is categorized based on some parameters such as qualification, education background and job experience. Their proposed self-learning engine uses the advantage of ontology to make inference from data in order to discover missing parameters as well as new relationship among the parameters. The inference techniques also improve the possible inconsistencies of various parameters. The system then performs a final job matching based on direct parameters extracted from user input and dynamically populated parameters from matched standard template. Their approach unlike this work does not annotate applicants profile and job vacancies, also their proposed architecture does not accommodate query approximation, neither does have a mechanism for accurate ranking and matching based on semantic similarities between core skills/ compentencies.

Vinaya *et al.*, (2015) proposed a qualitative assessment of resumes on the basis of different quality parameters using a simple text analytic based approach for a resume collection. The resume collection is assessed for two qualitative aspects, coverage and comprehensibility; and these ratings are transformed into a comprehensive quality rating. All the three parameters are collectively measured into a combined 1 to 5 rating scale for associating a quality metric for resumes. The qualitative evaluation results obtained through the algorithmic approach are congruent to and are hence validated through the wisdom of crowds. They evaluated and combined two qualitative parameters for resume assessment in a systematic and thorough manner, some improvements and extensions may still be possible. The pdf parsing and section identification can also be improved. The standard reference documents can be augmented as well. Transforming the computed values to ranks has been the trickiest part. While for coverage, it is simpler; but in case of comprehensibility, it is a bit complex and tricky to transform computed values to 1 to 5 scale rating. Nevertheless, the algorithmic formulation has the possibility of being used in an annotation and recommendation system.

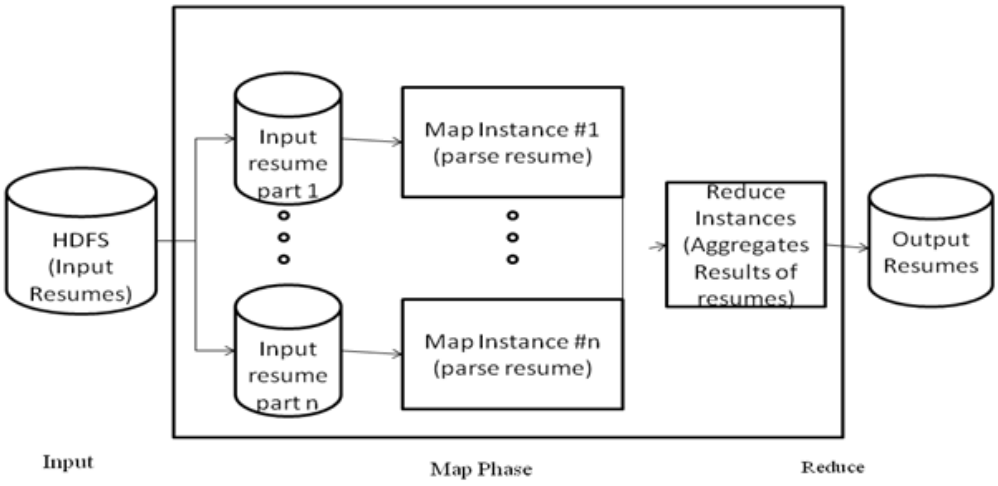


Figure 2.5: System Architecture Showing Resume and Concept Parser (Vinaya *et al.*, 2015)

Geetha and Fenelon (2015) proposed an ontology based approach for job recruitment which focuses on improving security in their proposed system, for selection to be done, the companies (HR) should follow a registration process with enhanced Security features. After this, HR's can search and view candidates based on their requirement like Area-Of-Interest, Aggregate Percentage and so on. The details of the candidates selected by the HRs can be mailed. After the recruitment process done for a particular year, the company profile, candidate details can be scrapped which helps in Memory Management. Their proposed system architecture enhances security, integrity and memory management by sending the security code to user's mail id security is maintained, by refreshing the candidate details periodically memory management is maintained and by logging the details of the user who is visiting the candidate details once again security and integrity are ensured.

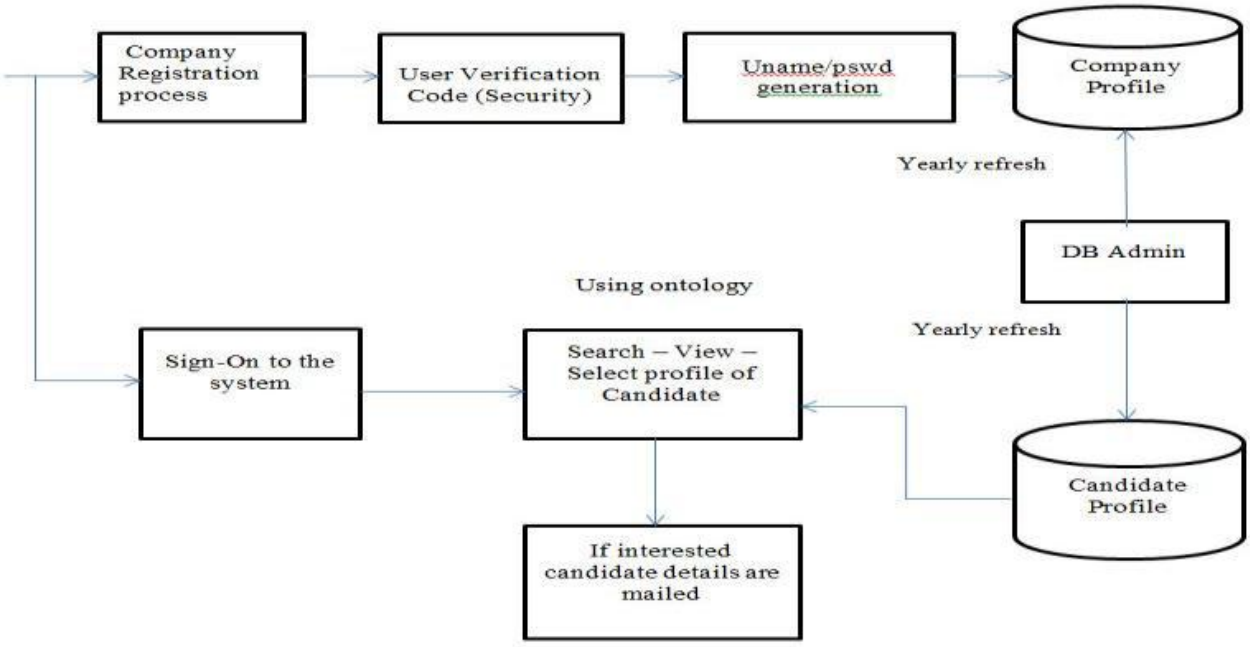


Figure 2.6: Online Job Recruitment System Architecture (Geetha and Fenelon, 2015)

Tran (2016) proposed a technique that uses ontologies to implement a query augmentation that improves defining the context through users adding suggestions of relevant keywords which represent core skills needed for a particular job or task. The intuition is that by searching based on core skills, the context narrows, making it easier to search for any consultant matching a specific assignment or a job seeker searching for a particular job. In his work he claims that the Human Resource Management Ontology was better suited for storing information rather than applying relations between the information. He also suggested that creating his own ontology was considered a better option and gives more freedom when querying with SPARQL. He implemented a query augmentation technique which suggests related words to the already given words in order to improve searching and matching accuracy. The approach in this work unlike his, seeks to implement a semantic similarity matching algorithm which accurately matches applicants with jobs and vice versa based on core skills supplied. He presents empirical results by conducting an experiment that measures the performance of query and the matching accuracy by calculating values for relative precision, relative recall and F-measure. This work seeks to measure the values of the accuracy of our system by using the aforementioned parameters and comparing it with his on the same data set to determine which model performs better.

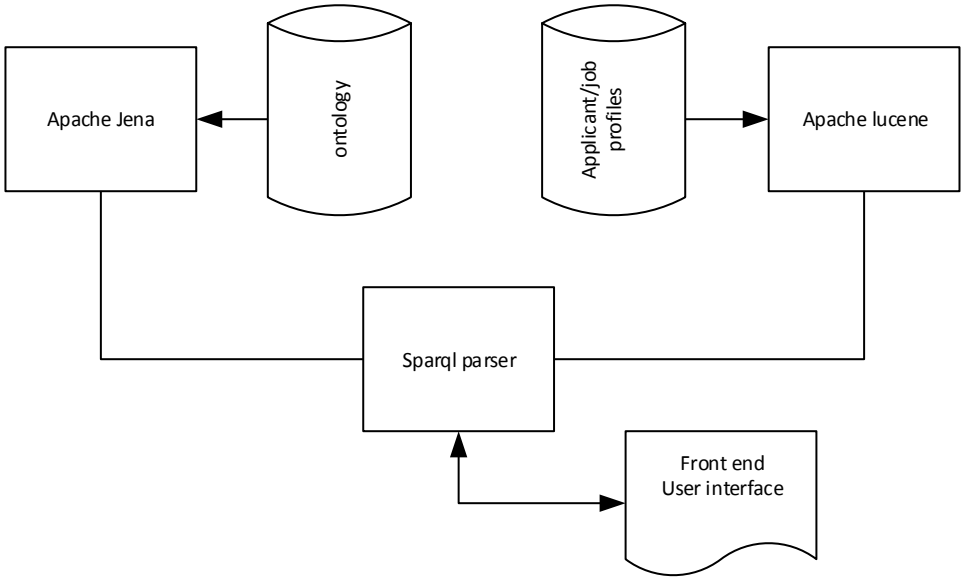


Figure 2.7: Ontology based Core skills Matching Architecture (Tran, 2016)

2.6 Literature Gap

Having completed an extensive literature review on several semantic web techniques and methods that have been employed towards improving the online job recruitment process using the ontology based approach. There is a need to improve matching of job seekers with corresponding job posting primarily based on technical competence. This dissertation seeks to enhance job/applicant search in online job portals by using an ontology based concept similarity matching algorithm to accurately match applicant profiles and job profiles essentially based on core skills and competence.

CHAPTER THREE

ONTOLOGY CONCEPT BASED MATCHING

3.1 Introduction

This chapter describes the proposed Business Model. The architecture of the proposed system is also presented alongside the existing concept similarity matching algorithm and the modifications made to the algorithm.

3.2 Proposed Business Model

The business model is the blueprint upon which the system model is developed and it comprises of the following steps.

1. Applicants can create a personal profile on the job portal using a controlled vocabulary to describe their core skills
2. Employers can also create their profiles and post jobs using a controlled vocabulary to describe the core skills required for the job
3. Applicants can search to get a ranked list of job positions for which they are best suited for based on the core skills supplied in their profile after which they can apply.
4. Employers can also search to get a ranked list of applicants who best fit their advertised positions based on the required core skills supplied in the job advert
5. Employers can request to view detailed profile of the applicant.
6. Applicants reserve the right to grant access to employers who request to view their detailed profile. This measure seeks to protect applicant's detailed information from those who might seek to obtain them for malicious intent.

3.3 Proposed system architecture

The system architecture illustrates the different levels of functionality of the proposed ontology based job recruitment portal. The system comprises of three major components or layers.

Data Layer: At this layer, the job applicant's full profile and the Employer profiles with job adverts originally stored in a MySQL database are converted to RDF format using the D2RQ platform. The D2RQ engine, is a system for accessing relational databases as RDF graphs and also allows data stored in RDF to be queried using SPARQL. When a query is performed, core skills are extracted from the applicant and job profiles. Core skills represents concepts on the ontology. These core skills are accessed by the application logic engine from D2RQ for semantic similarity computation.

Application and Logic Layer: At this layer, classes which represent core skills required in the programming domain initially implemented in an OWL file, are accessed in java using the OWL2Java framework. The Modified Semantic similarity matching Algorithm is deployed to the ontology where the weight values are assigned to the edges between classes. Furthermore the semantic matching of job profiles and applicant profiles is performed by computing the semantic similarity of the core skills and further ranking based on semantic similarity in relation to the ontology.

User interface/Front end layer: On top of the architecture is the front-end layer which offers a browser-based user interface, which accepts input from users by forms: These forms are used for login and entering details on user profiles. Ranked query results are also displayed to the users. The complete system architecture is given figure 3.1

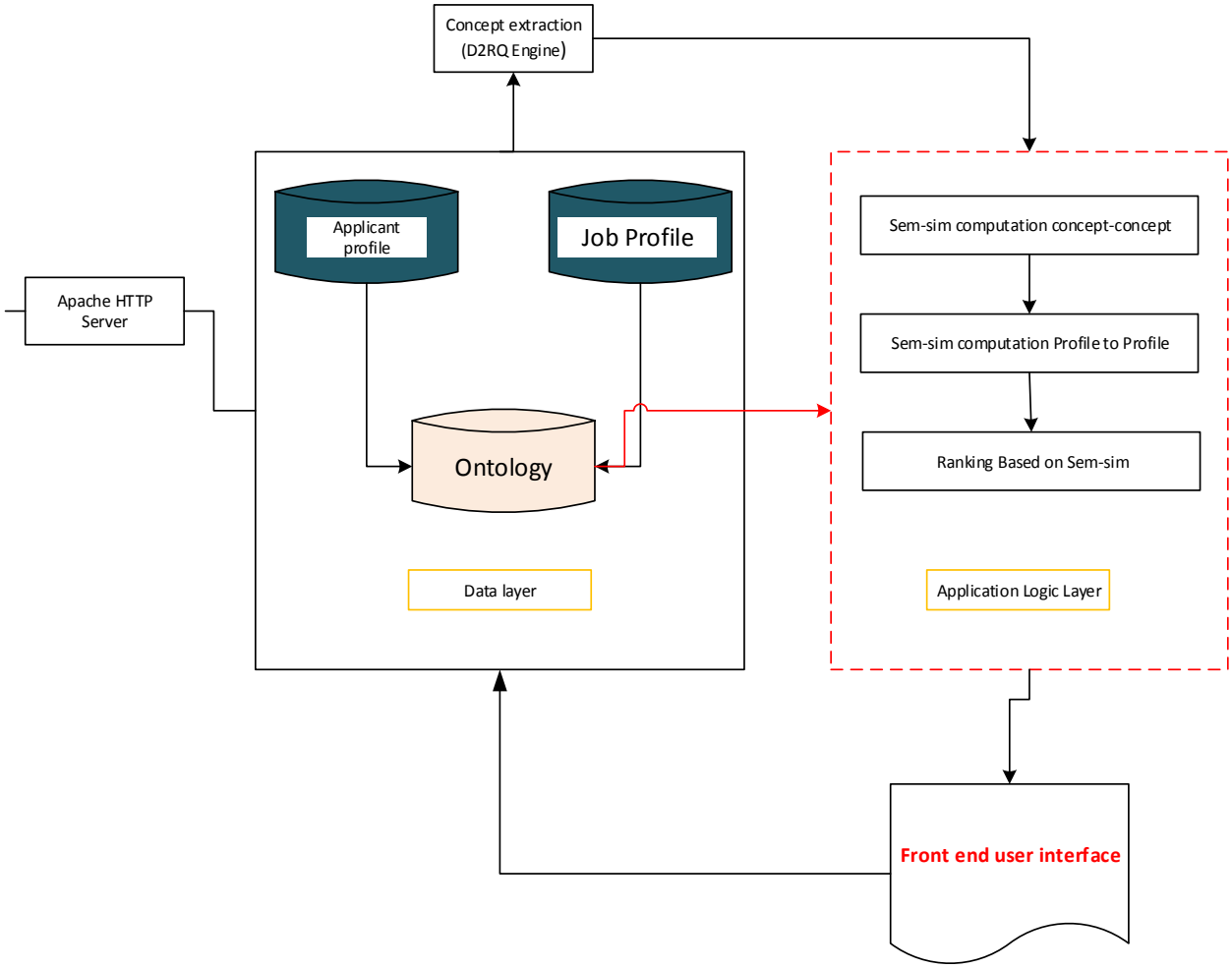


Figure 3.1: System Architecture

3.4 Semantic similarity matching Algorithm

Jike and Yuhui (2008) proposed a concept similarity matching method based on semantic distance. It comprehensively considers the inheritance relations and semantic distance relations between concepts, and measures the degree of matching between concepts through semantic similarity. The algorithm takes two concepts as input and computes a semantic similarity as output in four macro steps:

Step 1: Weight allocation. Determine weight according to the relationship between root node of ontology and other concept nodes.

Step 2: Node routing table generation. Record all paths between root node and concept nodes, and generate node routing table.

Step 3: Semantic distance computation. Compute semantic distance according to the node routing table. The semantic distance is the sum of weight between the concepts that have the inheritance relationship.

Step 4: Semantic similarity computation. Construct similarity function, and compute semantic similarity between concepts based on semantic distance.

Algorithm 1: Existing Concept Similarity Matching algorithm based on Semantic distance Computation

Input: two concepts $c_1 \; and \; c_2$

Output: semantic distance: Sem_Dis ($c_{1,} c_2$)

For two ontological concepts: $c_1 \; and \; c_2$

If $c_1 \;\; c_2$ are the same concept

Sem_Dis ($c_{1,} c_2$) = 0

Else if there exists the direct path relation between $c_1 \; and \; c_2$

Sem_Dis (c_1, c_2) = w [sub (c_1, c_2)]

Else if there exists the indirect path relations between $c_1 \; and \; c_2$

Sem_Dis ($c_{1,} c_2$) = $\sum_{w_c}[sub(c_1, c_2)]$

Where, SPath denotes the shortest path between $c_1 \; and \; c_2$

Else

Sem_Dis ($c_{1,} c_2$) = min {Sem_Dis (c_1, c_0)} + min {Sem_Dis (c_2, c_0)}

Notations used in Algorithm 1 and their meanings

$c_{1,}c_2$ are concepts on the ontology

c_0 is the root concept on the ontology

Sem_Dis (Semantic Distance)

W is the weight value between $c_{1,}c_2$

$\sum w_c$ is the sum of weights between two concepts where there exists an indirect path relation

Algorithm 2: Sub Algorithm for Weight function

```
// C1, C2 are concepts on the ontology
// W is the weight value between C1, C2
// H is an array which holds weights between node edges
Input H, C1, C2
        Sub (H, C1, C2 )
         W;
      W =  H[ C1 ][ C2]
      Return W,
```

Algorithm 2 is called by algorithm 3 to return the weight allocated to the edge between two concepts on the ontology based on their relative positions on the ontology.

Algorithm 3: Modified Concept Similarity Matching algorithm based on Semantic distance Computation

Algorithm Searching ($P,\ Z\{\ z_0 . . . z_n\ }$)

// P is the profile to search

// Z is the set of profiles being searched

// f [0 . . . n] is an array of skills of P

// k [i] is an array of skills of z

// n_0 is the root node

// v is the total number of concepts on the ontology

//Q is an array of sorted profiles based on semantic distance (Sem_Dis)

// H [][] is an array which hold weights between node edges

H [][] = Weights

For each $z\ in\ Z$

k [ ] ← core skills of z

Sum = 0

For i = 0 to k.length - 1

For j = 0 to f.length - 1

If k[i] = f[j] then

Sem_Dis (k[i], f[j]) = 0

Else if there exists the direct path relation between k[i] and f[j]

Sem_Dis (k[i], f[j]) = W_d [sub (H, k[i], f[j])]

Else if there exists the indirect path relations between k[i] and f[j]

Sem_Dis (k[i], f[j]) = $\sum W_i$ [sub (H, k[i], f[j])

Else

Sem_Dis (k[i], f[j]) = min {Sem_Dis($k[i], n_0$)} + min {Sem_Dis(f[j], n_0)}

End for loop

Sum+= sem_Dis

End for loop

Q[]. Profile = Z

Q[]. Sem_Dis = Sum

// sort Q based on the value

End for each

Q[].Sort (Sem_Dis)

The modified concept similarity matching algorithm based on semantic distance also adopts Jike and Yuhui (2008) four macro steps in computing semantic similarity between two concepts. However, in this work the semantic similarity is not between only two concepts, but rather a job profile and an applicant profile which both contain multiple skills (concepts). The algorithm therefore computes the semantic similarity between all the skills in a job profile with all the skills in an applicant profile in pairs and obtains a total. For a particular applicant profile, it repeats this against all the available job profiles and vice versa on the portal. It further sorts and returns in ascending order all the jobs for which a particular candidate is qualified for or all the available candidates qualified for a particular job which represents a ranking mechanism based on semantic similarity.

3.4.1 Semantic matching Concept to concept

3.4.1.1 Weight Allocation

The algorithm allocates the weight value to the edge between concept nodes. Given two concepts c_1 and c_2, the weight allocation function is given as. (Jike and Yuhui, 2008)

W [sub (c_1, c_2)] = $1 + \frac{1}{k^{depth(c_2)}}$…….. (3.1)

W_s [sub $(c_{1,} c_2)$] = $(k + x)$ …… (3.2)

Equations (3.1) and (3.2) are weight allocation functions for Parent-child nodes and sibling nodes respectively. Where, depth(c_2) presents the depth of concept (c_2) from the root concept in ontology hierarchy, k is a predefined factor larger than 1 indicating the rate at which the weight values decrease along the ontology hierarchy. x is a predetermined factor, selected from the given range ($x \geq 0.1 \leq 0.9$), currently we set x = 0.2. These equations have two fundamental properties: (1) The semantic differences between upper level concepts are higher than those

between lower level concepts, in other words, two general concepts are less similar than two specialized ones. (2) The distance between sibling concepts is greater than the distance between parent and child concepts. Specially, the depth of root concept is zero, and the depth of other concepts equal to their path length to root concept node. Figure 3.2 shows an example of the weight values of ontological concepts (where k = 2)

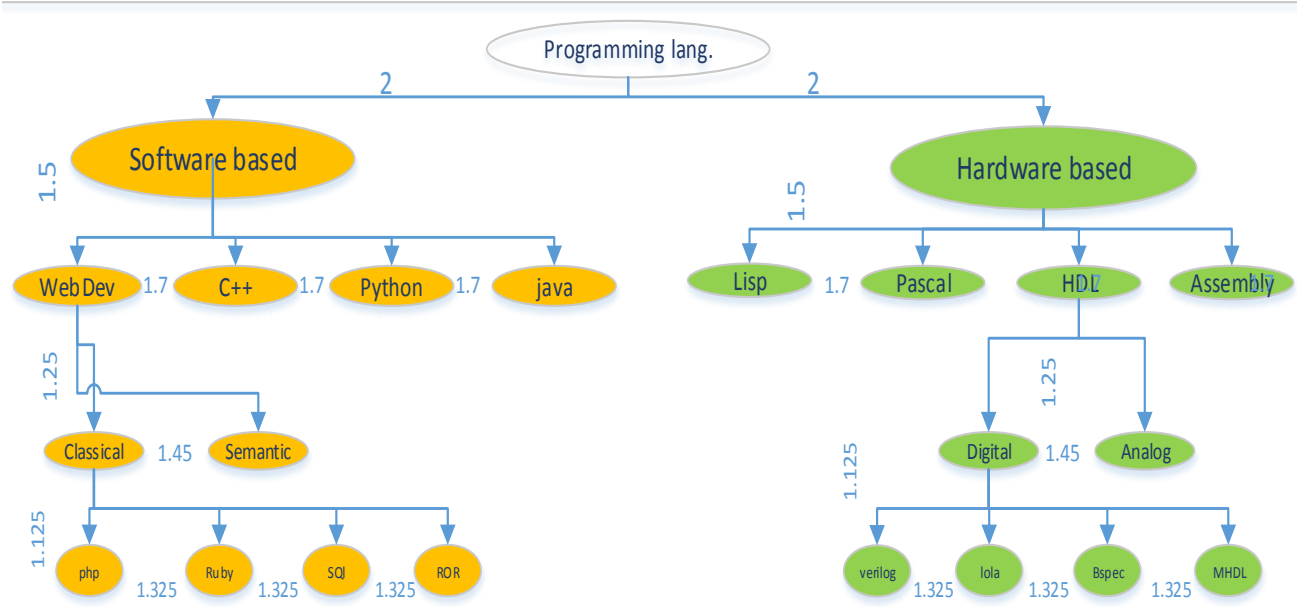


Figure 3.2: Ontology Showing Weight Allocation between Concept Nodes

Figure 3.2 shows how weights are allocated from root concept to other concept nodes. Initially k = 2 (Predefined factor larger than one is assigned), which is the weight between Software based and Hardware based to the root concept (Programming languages). The weight between Hardware based and software based to their respective children node is 1.5, which was arrived at by using equation (3.1) which computes weight values between parent and child nodes based on the depth from a particular node to the root node. Similarly the weight between children nodes of

Software based and children nodes of Hardware based is 1.7 respectively this was arrived at by using equation (3.2) which adds the current value of k to a predetermined constant factor currently set at 0.2. The weight between Classical and Semantic, which are children nodes to Web development is 1.25. This was obtained also by using equation (3.1) whilst also considering the depth to the root node. This Process is repeated for all concepts to compute weights between parent/child and sibling nodes.

3.4.1.2 Node Routing

To each concept node in ontology, we can get all its paths to the root node, and can compute related weight values between nodes in every path, and we can use paths and weight values to generate the node routing. We can compute and compare the length of path between two nodes through the node routing table. It is convenient to obtain the shortest semantic distance not traversing the whole ontology. The path record of child node is composed of the parent node's path record and itself, the child's weight is composed of the parent node's weight list and the weight between the parent-child nodes.

3.4.1.3 Semantic Similarity Computation

The greater the semantic distance between two concepts, the lesser the semantic similarity between them. That is, semantic similarity and semantic distance have an inverse relation

$$semantic\ distance = \frac{1}{semantic\ similarity} \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3.3)$$

3.4.2 Semantic matching profile to profile

The algorithm obtains a single semantic distance value between a job profile and an applicant profile by computing semantic distance between all the skills in the job profile with those in the applicant profile in pairs and returning a total. Figure 3.3 shows skills possessed by an applicant and skills required by the employer. The profile to profile computation is done by calculating the semantic distance with reference to the ontology. The example in Figure 3.3 uses the sample ontology shown in Figure 3.2

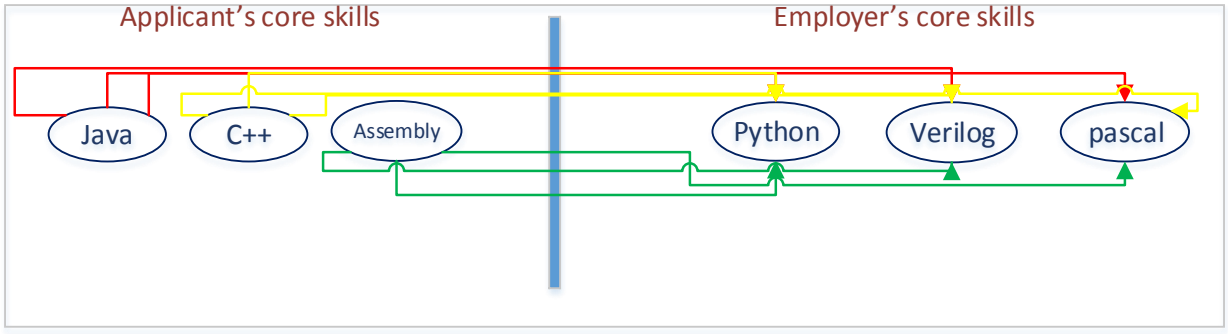


Figure 3.3 Applicant and Employer Skills

Thus semantic distance between (Java, Python) = 1.7 (Java, Verilog) = 1.5+2+1.125+1.25+1.5+2 = 9.36 (Java, Pascal) = 1.5+2+2+1.5 = 7 (C++, Python) = 1.7 (C++, Verilog) = 1.5 + 2 + 1.5 + 1.125 + 1.25+ 1.5+2=9.36 (C++, Pascal) = 1.5+2+2+1.5=7 (Assembly, Python) =1.7 (Assembly, Verilog) = 1.125 + 1.25 + 1.7 = 4.08 (Assembly, Pascal) = 1.7. After computing the semantic distance between all the skills in pairs, the next stage is to compute the total semantic distance between the two profiles which will be the sum of all the pair to pair skills initially computed, which will be 1.7 +9.36+7+1.7+9.36+7+1.7+4.08 = 42. Thus the value of the semantic distance between the applicant profile and the job profile in Fig 3.3 is 42.

3.4.3 Ranking

Ranking is achieved by returning values of semantic distance computed in ascending order. This implies that profiles with the greatest semantic similarity (i.e. the smallest value of semantic distance) when compared to the job profile are returned first.

CHAPTER FOUR

IMPLEMENTATION AND RESULTS

4.1 Introduction

This chapter presents the implementation of the Semantic Job Portal. Testing of the developed system was done by analyzing and evaluating the performance graphically. After describing the design elements of the system, it is necessary to know relevant measurements that determine how well the search performs. Three performance criteria; Precision, Recall and F-measure were adopted. Finally, we present the experimental results and discussions as compared with Tran (2016).

4.2 System Specification

The hardware and software specification used for this work are as follows:

4.2.1 Hardware

a Hewlett Packard (HP) laptop with a (Tm)i5-6200U Processor running at 2.40GHz

b 6.00GB of RAM and

c 500GB of hard disk

4.2.2 Software

a Windows 8 operating system

b Protégé 4.3

c Java NetBeans IDE, SE 8 Implementation

d Owl2Java API Version 2

4.2.3 Programming Languages

a. PHP

b. Java

4.3 Ontology Design

For this research work, re-using existing standard ontology was considered, however, the open source ontologies found online do not suit our objectives. Therefore a new ontology which is also an extension of Tran, (2016) was created from scratch based on the guide for development of Ontologies given by (Noy *et al.*, 2016) where the following was stated:

1. Determine the domain and scope of the ontology.
2. Consider reusing existing ontologies.
3. Enumerate important terms in the ontology.
4. Define the classes and the class hierarchy.
5. Define the properties of classes-slots.
6. Define the facets of the slots.
7. Create instances.

4.3.1 Designing the CV Ontology

The first step requires that the domain and scope of the ontology to be designed must be determined. For this ontology, the domain is the computer programming languages and the scope is to model the basic information as regards core skills/concepts that are associated with Job positions in the aforementioned domain. The second step requires the consideration of reusing existing ontology. There was no access to any suitable one at the time of carrying out this research that could completely be re-used. However, the ontology was an expansion of that created by (Tran 2016), which only considered the domain of Web Programming. Figure 4.1

shows the classes and their hierarchy as required by step four in ontology design. In step five relations (properties) are created to connect entities or classes and literal values. There are two types of properties (relations) that are used in OWL. These are DataProperty and ObjectProperty. Some of the properties used here are, hasSkills, requiresSkill and inCategory (ObjectProperty). As mentioned earlier, the domain Ontology in this work is the computer programming languages which represents core skills/concepts that are associated with Job positions that require competence in the aforementioned domain.

Creating Classes and class Hierarchy

Two super classes were created for the ontology in this work. Firstly is the "Person class" which comprises of two subclasses which are the employer and the applicant class. The second is the "Programming language class" which comprises of two major subclasses which are, Hardware development based programming languages subclass and Software development based programming language subclass.

Hardware development based programming languages subclass: This class consists of programming languages often required by employers who need programmers for the purpose of hardware development. These programs are commonly used in coding electronic circuits and digital logic circuits. Some of the sub classes used under this category include Assembly, Lisp, Matlab, Pascal and also Hardware Development Language (HDL). HDL class comprises of two sub classes which are Analog and Digital.

Software development based programming languages subclass: This class was populated with major programming languages required in the software development circles relative to the needs of the employer. They include C++, C#, Delphi, Java, Ruby, Perl, Objective C, Python,

MacRuby, Swift and Visual Basic. The Software Development class had one major sub-class which is Web development based programming languages which is further broken down into two other major sub-classes these are Classical web programming languages and Semantic web programming languages. Classical web based programming languages are programming languages required by employers for the development of classical web systems and applications, This was further subdivided into two main classes often times as required by employers, which are Front-end Path and Full-Stack Path. These two subclasses were populated with sub classes as shown in the onto-graph and owlviz below. Semantic Web based programming languages includes Languages often required by employers for the development of semantic web based systems and applications basically OWL RDF and XML

It is important to note that the classes and sub-classes created in this ontology which represents core skills and concepts required for employment in the programming languages domain are not exhaustive but good enough for our proposed modified concept similarity matching algorithm to operate upon. Figures 4.2 to 4.9 show the classes and sub classes on OWLViz in Protégé version 4.3.

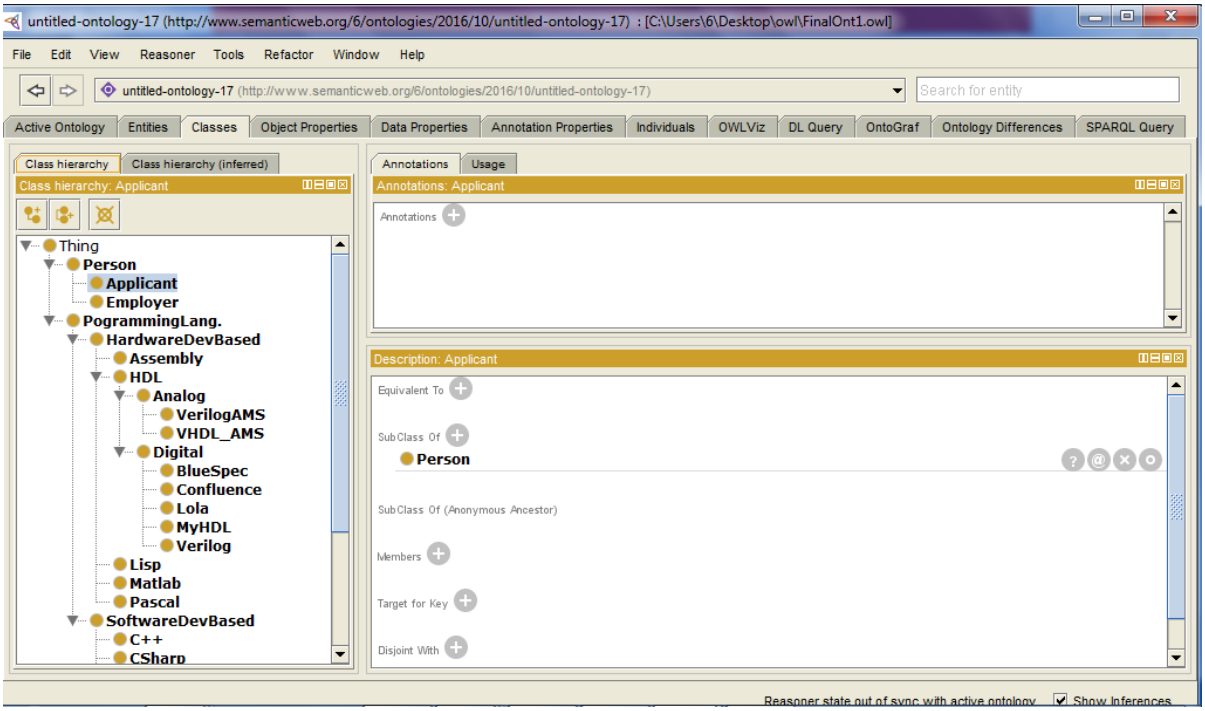


Figure 4.1: View of CV Ontology in Protégé Showing Class Hierarchy

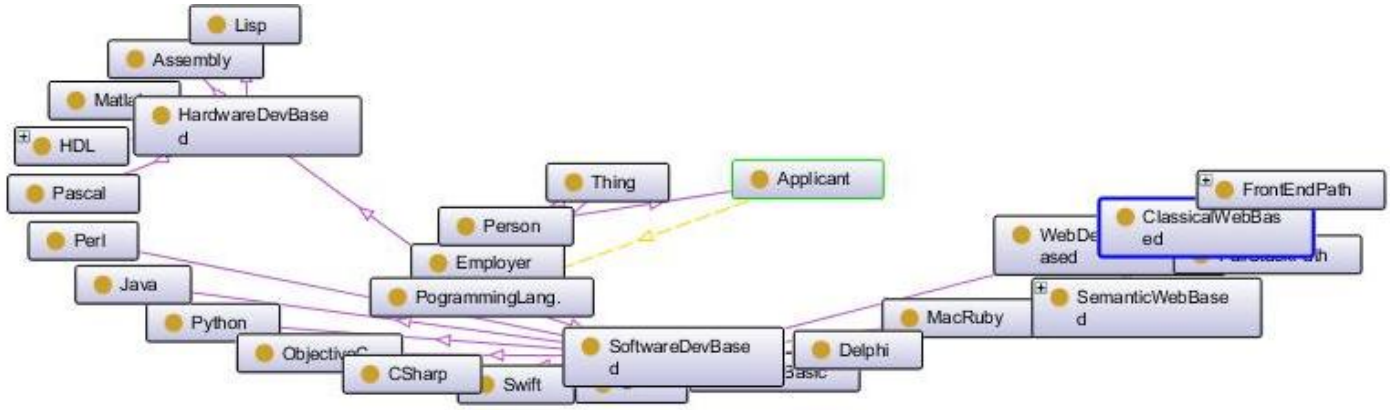


Figure 4.2: Graph of the Entire Ontology on Ontograph

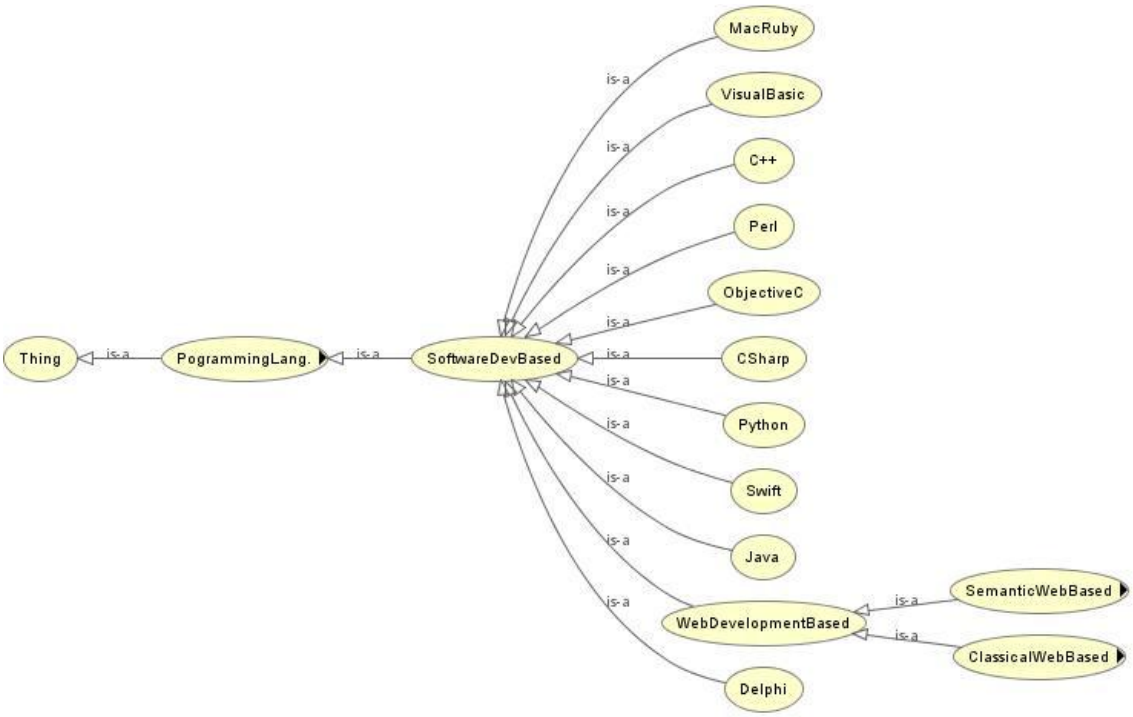


Figure 4.3: Graph of the Programming Language Class on OWLviz

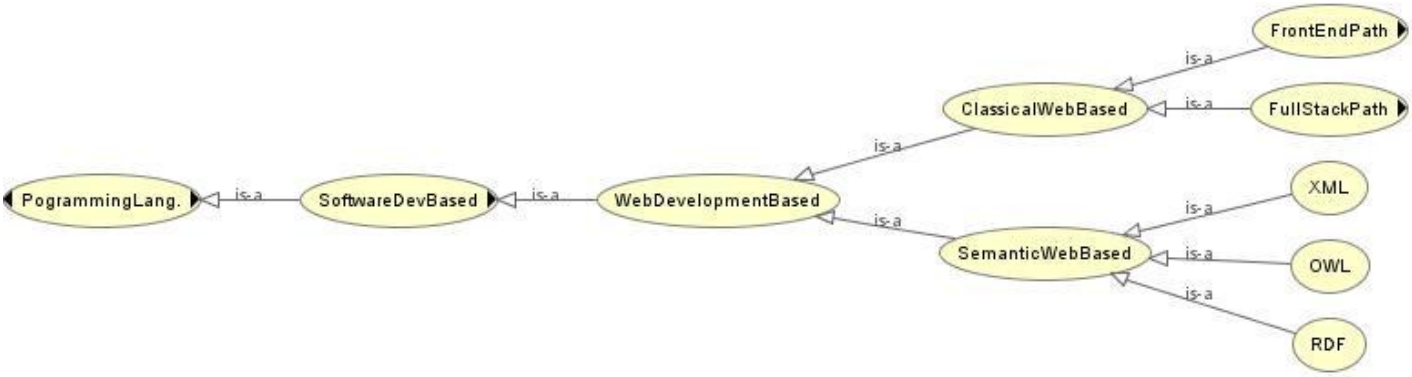


Figure 4.4: Graph of the Web Development Class on OWLviz



Figure 4.5: Graph of the Software Development based Class on Owlviz

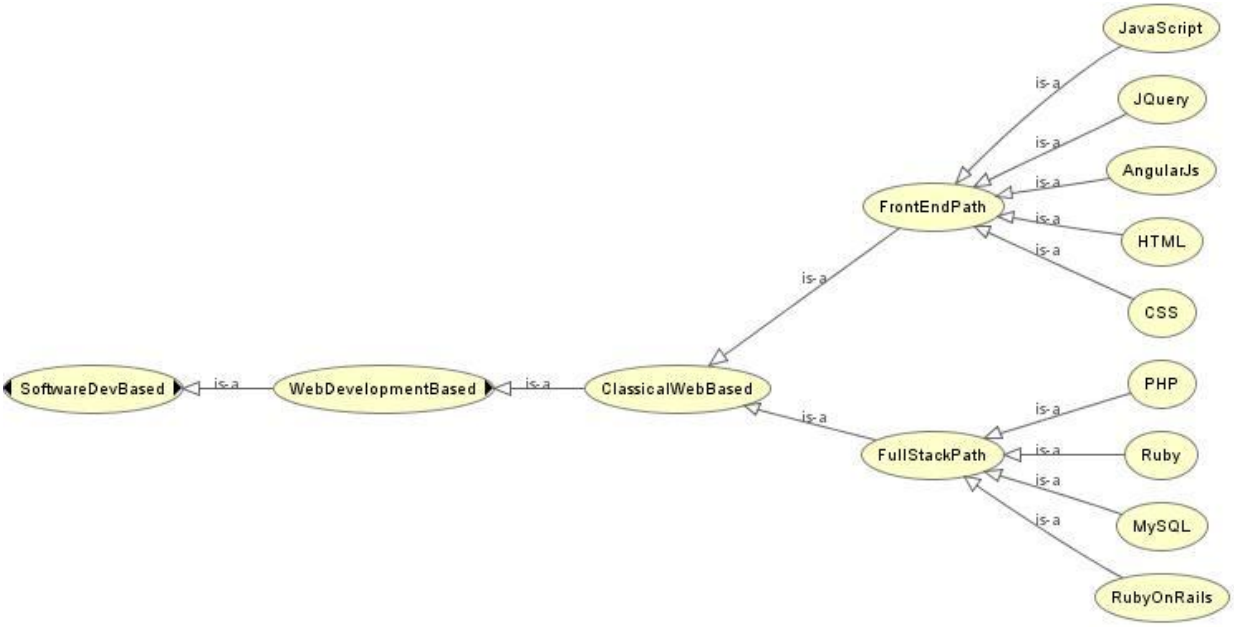


Figure 4.6: Graph of Web Development Class on OWLviz

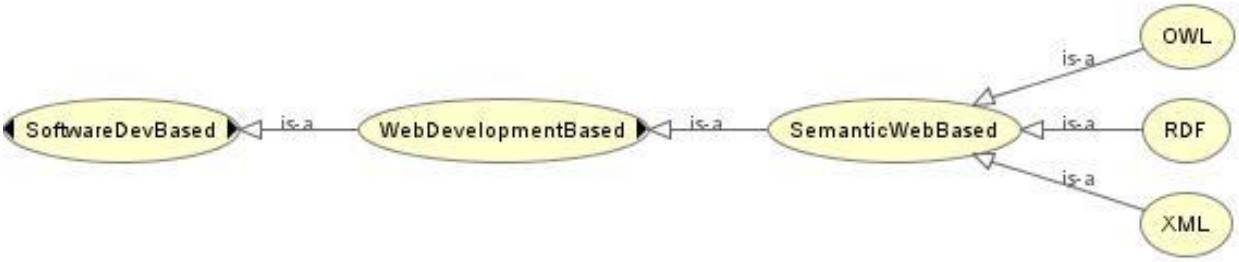


Figure 4.7: Graph of Semantic Web Based Class on OWLviz

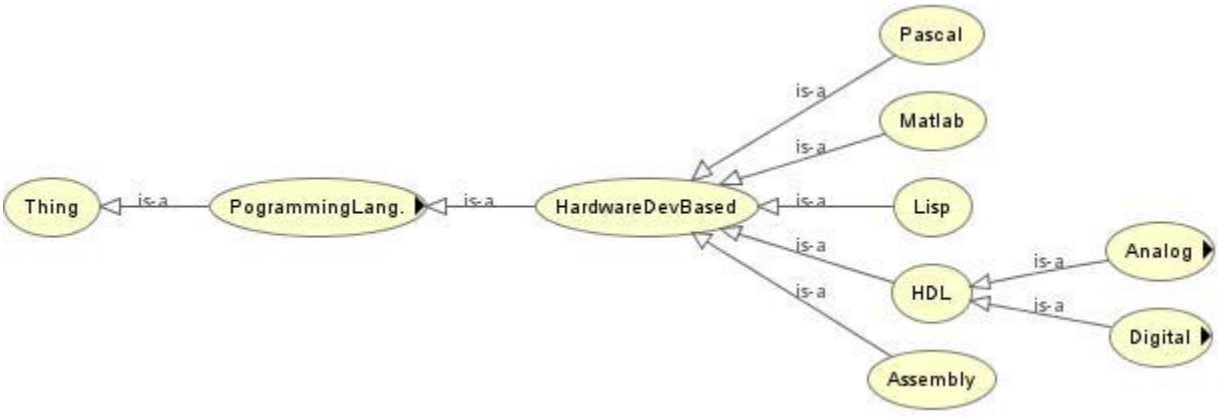


Figure 4.8: Graph of the Hardware Dev. Based class on OWLviz

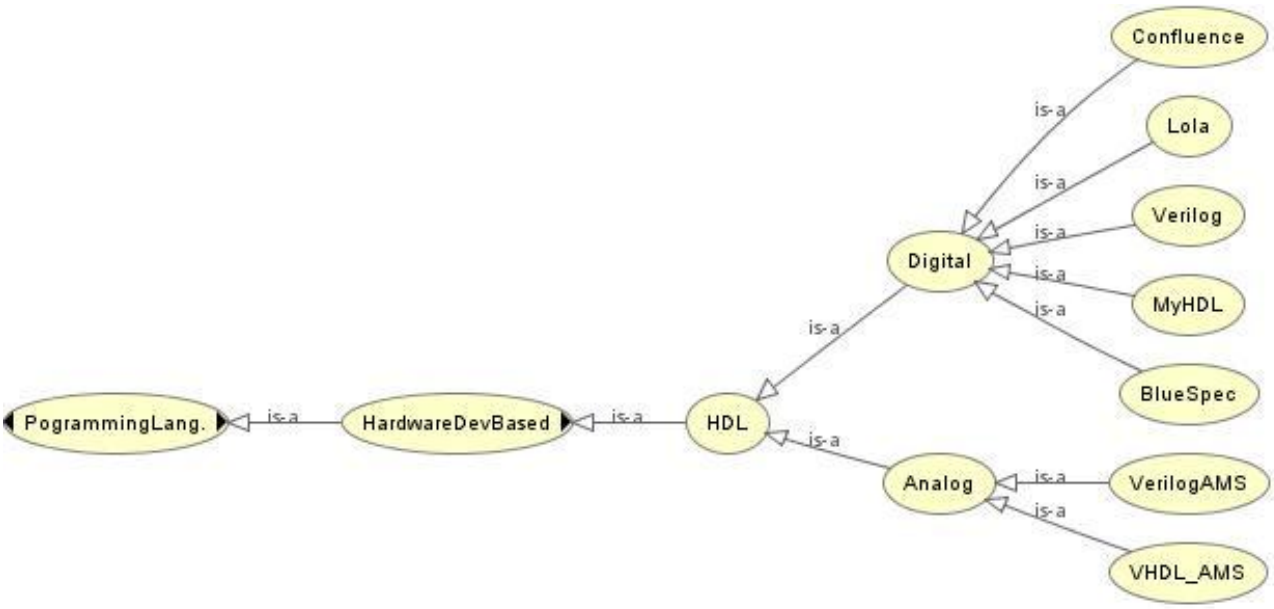


Figure 4.9: Graph of HDL Class on OWLviz

4.4 The Portal Design

The frontend/user interface was developed using HTML while CSS was used for styling. Javascript was used for the client side script while the server side scripts were written using PHP which is a widely-used open source general-purpose scripting language that is especially suited for web development and can embed HTML and also interacts well with the MySQL database.

There are several components such as forms, buttons and dialog boxes. The search button has the facility that makes it possible to search. Java programming language which is a versatile, open source and one of the most popular object oriented language was used to implement the concept similarity matching algorithm, and also to assign weight values to the concepts (skills) on the ontology imported into the OWL2Java API.

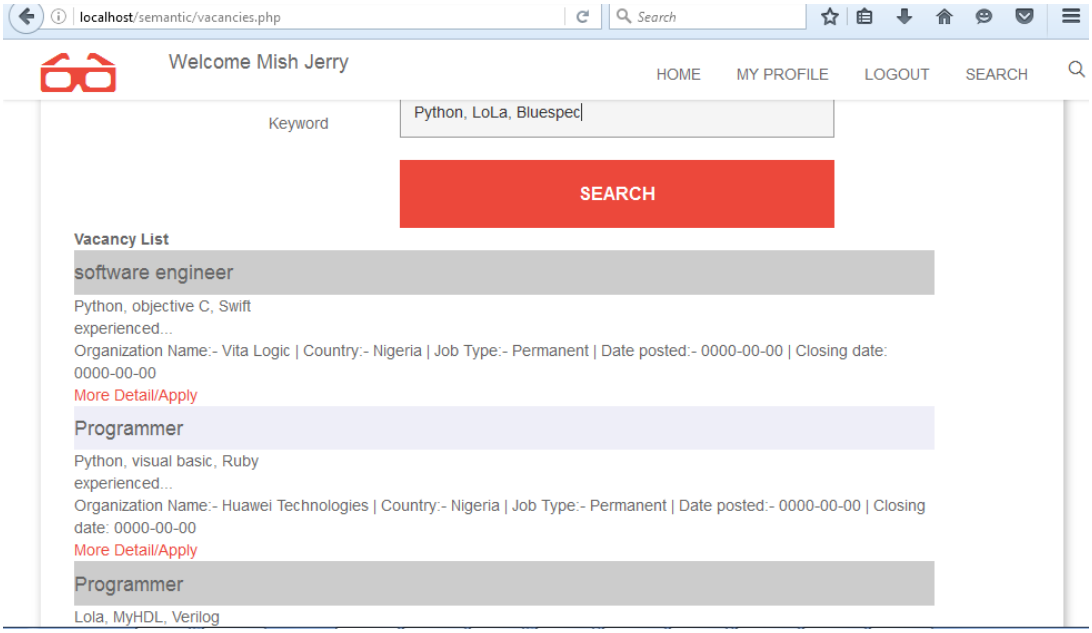


Figure 4.10: GUI showing Search Results for Job search

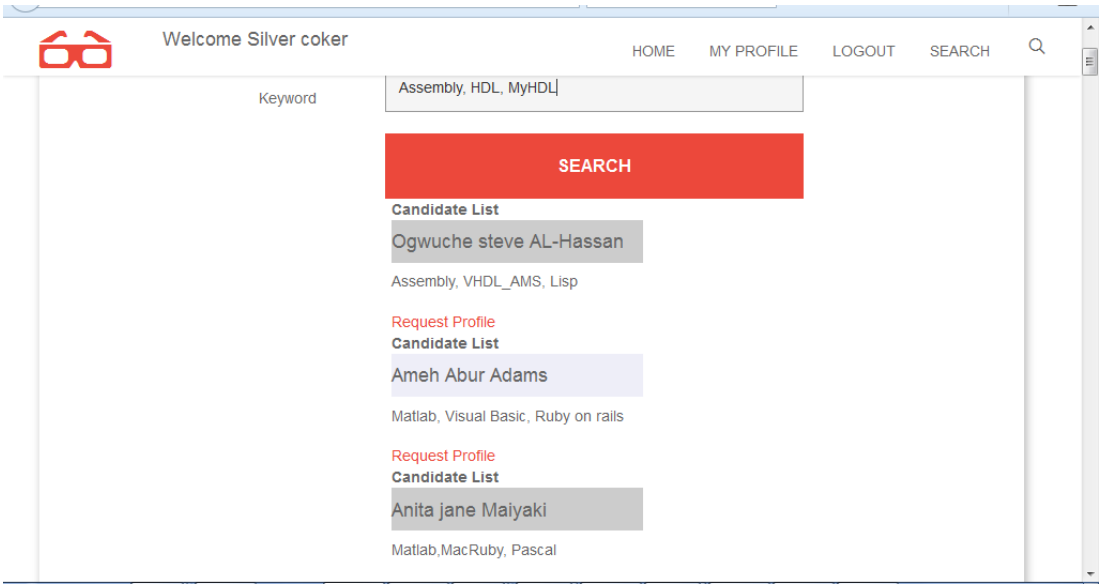


Figure 4.11: GUI Showing Search Results for Applicant Search

4.5 Data Set

A Total of One hundred and sixty (160) applicant and employer profiles were created. There were eighty (80) profiles for two core skills, three, and four respectively, all relative to the ontology. A total of sixty (60) profiles were used to test the Precision, Recall and F-measure of the search. The same set of data set were also used to test the performance of (Tran, 2016).

4.6 Performance Evaluation Criteria

The performance evaluation indices used to determine the performance of our system are Precision, Recall and F-measure which are broadly explained in the next section.

4.6.1 Precision, Recall and F-measure

Precisions primarily is used to show how useful search result are. Precision and recall are units of measurement that show an indication of how well a system performs when querying for relevant documents. Given a set of documents divided into relevant and non -relevant documents, Precision measures how many of the documents retrieved by a search are relevant. Recall on the other hand measures how many of the relevant documents in the whole set of documents get retrieved.

The strict definition of Precision is given as: (Tran, 2016)

$$\text{Precision} = \frac{n(relevant\ documents\ \cap\ retrieved\ documents)}{n(retrieved\ documents)} \quad \ldots\ldots\ldots\ldots\ldots\ldots (4.1)$$

The precision, also known as Positive Predictive Value (PPV), is given by taking all the relevant documents into consideration and see how many of the retrieved documents actually were relevant. If the result is that all the retrieved documents are relevant, the value of precision would

be 1. A low precision value would indicate that the set of retrieved documents mostly diverges from the relevant document set. A value of 1 in precision may indicate a good retrieval of the search engine but can be misleading. A simple way of looking at this is that it is quite easy to reach a value of 1 if the search engine have a very strict condition resulting in only one document that form the relevant set. The problem is that the relevant set may be a lot larger than only the one relevant document, leading to incorrect conclusions. This is why Recall also needs to be taken into consideration. Recall is given as: (Tran, 2016)

$$\text{Recall} = \frac{n(relevant\ documents\ \cap\ retrieved\ documents)}{n(relevant\ documents)} \quad \ldots\ldots\ldots\ldots\ldots\ldots (4.2)$$

Recall, sometimes also known as Sensitivity or True Positive Rate (TPR), is similar to precision but looks at how many of the relevant documents are actually retrieved. A high value of recall would indicate that most of the documents in the retrieved set coincide with the relevant set. Again, the recall value by itself could be misleading as a high value can be reached by retrieving all the documents. As recall only checks if the documents from the relevant set are contained in the retrieved set, retrieving the whole set would inherently also retrieve all of the relevant documents resulting in recall getting the value of 1. Recall by itself may give a skewed result. Recall shows how complete search result are.

F-measure is a combination of both precision and recall, and is a harmonic mean of both the measurements. It tries to give a better measurement of "effectiveness" than recall and precision alone are able to accomplish. Generally, "effectiveness" is defined as:

Precision - Ability to get results that are closely grouped.

Accuracy - Ability to get results that are as close as possible to the correct one.

Trueness - Average of all the results closeness to the target.

F-measure, sometimes also called F-score, has multiple definitions but its main one is defined as follows: (Tran, 2016)

$$\text{F-measure} = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad \text{..... (4.3)}$$

Table 4.1: Summary of Search for Two Core Skills (Proposed System)

Applicants	Core skills searched	TNR_T	TNR	Precision	Recall	F-measure
1	Java, PHP	80	76	0.95	1	0.97
2	MacRuby, Ruby	80	71	0.89	1	0.94
3	Verilog, VHDL-AMS	80	70	0.88	1	0.93
4	HTML, CSS	80	60	0.75	1	0.86
5	Python, Lola	80	70	0.88	1	0.93
6	PHP, MySQL	80	75	0.94	1	0.96
7	Java, Python	80	77	0.96	1	0.98
8	C++, Matlab	80	75	0.94	1	0.96
9	C++,C#	80	76	0.95	1	0.97
10	CSS, JavaScript	80	69	0.86	1	0.92
11	Visual Basic, C++	80	72	0.90	1	0.95
12	Matlab, Pascal	80	70	0.88	1	0.93
13	MySQL, JavaScript	80	74	0.93	1	0.96
14	PHP, JavaScript	80	75	0.94	1	0.96
15	Blue spec, Verilog	80	70	0.88	1	0.93
16	Java, C++	80	77	0.96	1	0.98
17	Ruby on rails, Mac ruby	80	70	0.88	1	0.93
18	Ruby, Ruby on rails	80	70	0.88	1	0.93
19	VHDL-AMS, Bluesspec	80	70	0.88	1	0.93
20	Java, Swift	80	71	0.89	1	0.94
	Average			**0.91**	**1**	**0.95**

TNR_T = Total number of retrieved profiles

TNR = Total number of relevant profiles

Table 4.2: Summary of Search for Three Core Skills (Proposed system)

Applicants	Core skills searched	TNRT	TNR	Precision	Recall	F-measure
1	Java, PHP, C++	80	77	0.96	1	0.98
2	MacRuby, Ruby, Ruby on rails	80	73	0.91	1	0.95
3	Verilog, VHDL-AMS, Bluespec	80	70	0.88	1	0.93
4	HTML, CSS, Python	80	71	0.89	1	0.94
5	Python, Lola, Bluespec	80	75	0.94	1	0.96
6	PHP, MySQL, JavaScript	80	75	0.94	1	0.96
7	Java, Python, Swift	80	74	0.93	1	0.96
8	C++, Matlab, Pascal	80	70	0.88	1	0.93
9	C++,C#, Visual Basic	80	73	0.91	1	0.95
10	HTML, CSS,PHP	80	70	0.88	1	0.93
11	XML, OWL,RDF	80	68	0.85	1	0.91
12	Swift, Python, Visual basic	80	70	0.88	1	0.93
13	Matlab Verilog, Bluespec	80	70	0.88	1	0.93
14	CSS, PHP, Java	80	71	0.89	1	0.94
15	C++, C#, Python	80	74	0.93	1	0.96
16	Python, Verilog, Matlab	80	73	0.91	1	0.95
17	OWL,RDF,PHP	80	71	0.89	1	0.94
18	Ruby, Swift, Lola	80	72	0.90	1	0.95
19	Ruby on rails, Swift, Verilog	80	72	0.90	1	0.95
20	JavaScript, Jquery, C#	80	75	0.94	1	0.96
Average				**0.90**	**1**	**0.94**

Table 4.3: Summary of Search for Four core Skills (Proposed System)

Applicants	Core skills searched	TNRT	TNR	Precision	Recall	F-measure
1	Swift, Python, Visual Basic, java	80	70	0.88	1	0.93
2	MySQL, Jquery, JavaScript, C++	80	75	0.94	1	0.96
3	Assembly, HDL, MyHDL, Pascal	80	68	0.85	1	0.91
4	C++,C#, Python, Java	80	77	0.96	1	0.98
5	Lola, Bluespec, Verilog, VHDL	80	67	0.85	1	0.90
6	Delphi, Java, Matlab, Ruby	80	70	0.88	1	0.93
7	C++, C#, Delphi, Ruby on rails	80	70	0.88	1	0.93
8	Perl, Python, Swift, Lisp	80	69	0.86	1	0.92
9	Python, Swift, C#, Perl	80	73	0.91	1	0.95
10	C++, C#, Macruby, Objective C	80	73	0.91	1	0.95
11	CSS, JavaScript, Objective C, C++	80	75	0.94	1	0.96
12	Visual Basic, C++,C#, JavaScript	80	76	0.95	1	0.97
13	Matlab, Pascal, Bluespec, C#	80	70	0.88	1	0. 93
14	MySQL, JavaScript, Jquery, AngularJs	80	76	0.95	1	0.97
15	PHP, JavaScript, AngularJs, Jquery	80	76	0.95	1	0.97
16	Blue spec, Verilog, Pascal, Assembly	80	70	0.88	1	0.93
17	Java, C++, Python, Delphi	80	76	0.95	1	0.97
18	Ruby on rails, Mac ruby, Perl, Objective C	80	70	0.88	1	0.93
19	Ruby, Ruby on rails, Perl, Lisp	80	71	0.89	1	0.94
20	VHDL-AMS, Bluesspec, Assembly, HDL	80	70	0.88	1	0.93
Average				**0.90**	**1**	**0.94**

Table 4.4: Summary of Search for Two Core Skills (Tran, 2016)

Applicants	Core skills searched	TNR_T	TNR	Precision	Recall	F-measure
1	Java, PHP	20	20	1	0.26	0.41
2	MacRuby, Ruby	19	19	1	0.28	0.44
3	Verilog, VHDL-AMS	15	15	1	0.21	0.38
4	HTML, CSS	20	20	1	0.33	0.50
5	Python, Lola	14	14	1	0.20	0.33
6	PHP, MySQL	21	21	1	0.28	0.44
7	Java, Python	20	20	1	0.26	0.41
8	C++, Matlab	17	17	1	0.23	0.37
9	C++,C#	14	14	1	0.18	0.31
10	CSS, JavaScript	12	12	1	0.17	0.30
11	Visual Basic, C++	13	13	1	0.18	0.31
12	Matlab, Pascal	16	16	1	0.23	0.37
13	MySQL, JavaScript	14	14	1	0.19	0.32
14	PHP, JavaScript	17	17	1	0.23	0.37
15	Blue spec, Verilog	10	10	1	0.14	0.25
16	Java, C++	18	18	1	0.23	0.37
17	Ruby on rails, Mac ruby	14	14	1	0.20	0.33
18	Ruby, Ruby on rails	13	13	1	0.19	0.32
19	VHDL-AMS, Bluesspec	9	9	1	0.13	0.23
20	Java, Swift	19	19	1	0.28	0.44
	Average			**1**	**0.22**	**0.36**

Table 4.5: Summary of Search for Three Core Skills (Tran, 2016)

Applicants	Core skills searched	TNRT	TNR	Precision	Recall	F-measure
1	Java, PHP, C++	48	48	1	0.62	0.76
2	MacRuby, Ruby, Ruby on rails	30	30	1	0.41	0.58
3	Verilog, VHDL-AMS, Bluespec	35	35	1	0.50	0.67
4	HTML, CSS, Python	40	40	1	0.56	0.71
5	Python, Lola, Bluespec	37	37	1	0.49	0.66
6	PHP, MySQL, JavaScript	46	46	1	0.61	0.76
7	Java, Python, Swift	44	44	1	0.59	0.74
8	C++, Matlab, Pascal	49	49	1	0.70	0.82
9	C++,C#, Visual Basic	49	49	1	0.67	0.80
10	HTML, CSS,PHP	45	45	1	0.64	0.78
11	XML, OWL,RDF	33	33	1	0.49	0.66
12	Swift, Python, Visual basic	32	32	1	0.46	0.63
13	Matlab Verilog, Bluespec	38	38	1	0.54	0.70
14	CSS, PHP, Java	45	45	1	0.63	0.77
15	C++, C#, Python	48	48	1	0.69	0.81
16	Python, Verilog, Matlab	45	45	1	0.62	0.77
17	OWL,RDF,PHP	38	38	1	0.54	0.70
18	Ruby, Swift, Lola	30	30	1	0.42	0.59
19	Ruby on rails, Swift, Verilog	34	34	1	0.47	0.64
20	JavaScript, Jquery, C#	38	38	1	0.51	0.66
Average				**1**	**0.56**	**0.71**

Table 4.6: Summary of Search for Four Core Skills (Tran, 2016)

Applicants	Core skills searched	TNRT	TNR	Precision	Recall	F-measure
1	Swift, Python, Visual Basic, java	51	51	1	0.72	0.84
2	MySQL, Jquery, JavaScript, C++	51	51	1	0.68	0.81
3	Assembly, HDL, MyHDL, Pascal	34	34	1	0.50	0.67
4	C++,C#, Python, Java	50	50	1	0.65	0.77
5	Lola, Bluespec, Verilog, VHDL	37	37	1	0.55	0.71
6	Delphi, Java, Matlab, Ruby	40	40	1	0.57	0.73
7	C++, C#, Delphi, Ruby on rails	44	44	1	0.63	0.77
8	Perl, Python, Swift, Lisp	46	46	1	0.67	0.80
9	Python, Swift, C#, Perl	45	45	1	0.62	0.77
10	C++, C#, Macruby, Objective C	49	49	1	0.67	0.80
11	CSS, JavaScript, Objective C, C++	50	50	1	0.66	0.79
12	Visual Basic, C++,C#, JavaScript	50	50	1	0.66	0.79
13	Matlab, Pascal, Bluespec, C#	34	34	1	0.49	0.66
14	MySQL, JavaScript, Jquery, AngularJs	45	45	1	0.59	0.74
15	PHP, JavaScript, AngularJs, Jquery	47	47	1	0.62	0.77
16	Blue spec, Verilog, Pascal, Assembly	40	40	1	0.57	0.73
17	Java, C++, Python, Delphi	48	48	1	0.63	0.77
18	Ruby on rails, Mac ruby, Perl, Objective C	31	31	1	0.44	0.61
19	Ruby, Ruby on rails, Perl, Lisp	33	33	1	0.46	0.63
20	VHDL-AMS, Bluesspec, Assembly, HDL	37	37	1	0.53	0.70
Average				**1**	**0.60**	**0.74**

Table 4.7: Aggregate Results of Proposed System

No of Skills	Precision	Recall	F-Measure
Two skills	0.91	1	0.95
Three Skills	0.90	1	0.94
Four skills	0.90	1	0.94
Average	**0.90**	**1**	**0.94**

Table 4.8: Aggregate Results (Tran, 2016)

No of Skills	Precision	Recall	F-Measure
Two skills	1	0.22	0.36
Three Skills	1	0.56	0.71
Four skills	1	0.60	0.74
Average	**1**	**0.46**	**0.60**

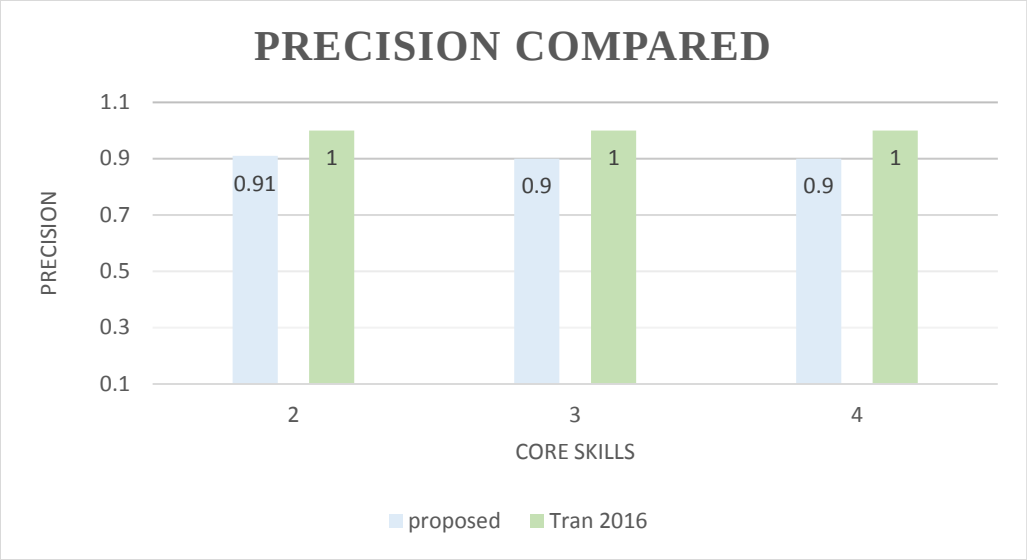


Figure 4.12: Results of Precision Test compared to Tran (2016)

The result shows that Tran (2016) has a better search precision with a 100% average as against an average of 90% precision in this work.

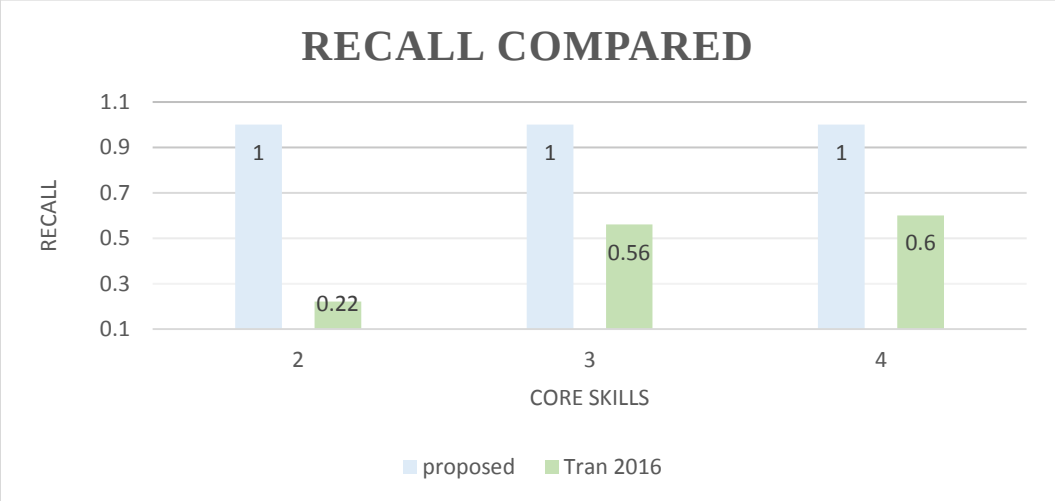


Figure 4.13: Results of Recall Test Compared to (Tran 2016)

The result shows that there is a 100% recall rate in this work due to the fact that all profiles compared are returned as against an average recall of 46% in the work of Tran (2016).

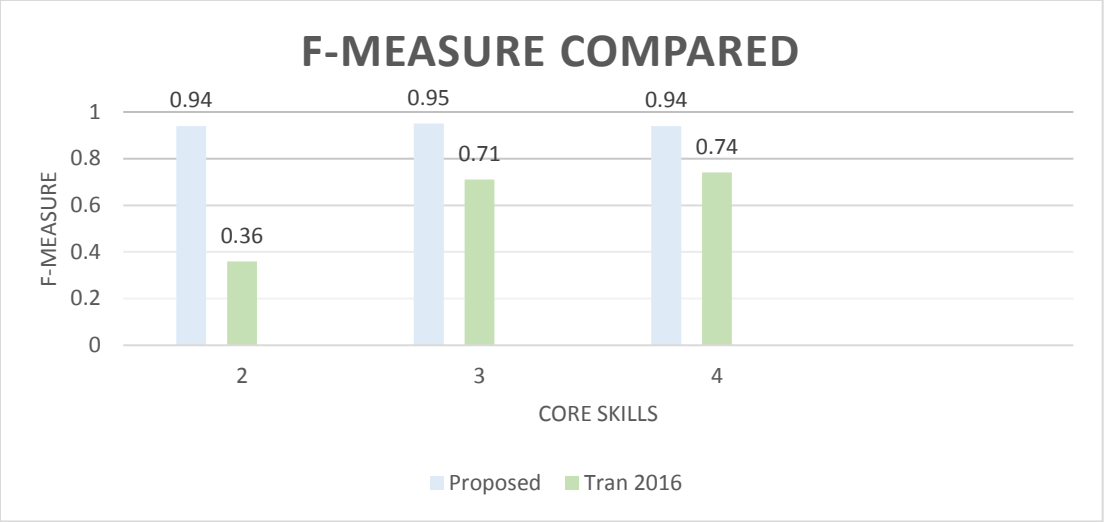


Figure 4.14: Results of F-measure Compared to (Tran 2016)

The result shows that there is a 91% average F-Measure in this work as against 60% in the work of Tran (2016).

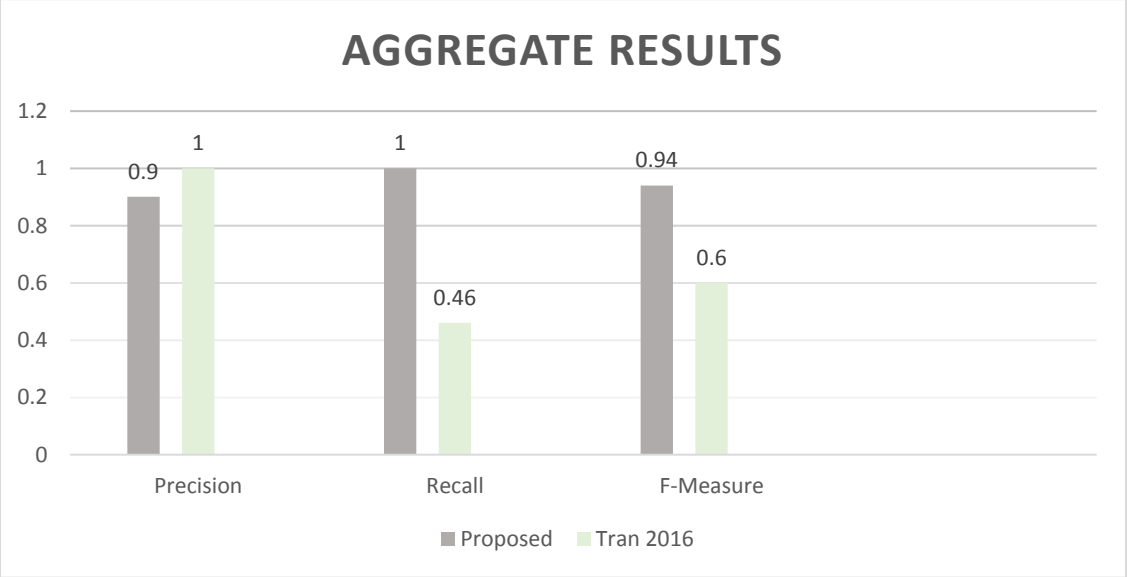


Figure 4.15: Aggregate Results Compared to (Tran 2016)

The aggregates of the results obtained in this work as compared to Tran (2016) shows that an improvement of 54% and 36% were obtained over Tran (2016) for Recall and F-Measure Respectively.

4.7 Summary of results and Discussion

Some of the observations made from the results obtained are summarized as follows:

i.) Our experiments indicate that the search and matching is performed to specification since it was able to get a ranked list of all jobs available on the portal relative to the ontology according to the core skills of the applicant/employer.

ii.) From the results obtained from this work there were relative improvements of 54% and 36% over Tran (2016) for Recall and F-Measure respectively these improvements were obtained as a result of the implementation of the concept similarity matching algorithm deployed on the ontology as against just leveraging on SPARQL query as used in Tran (2016). However in the work of Tran (2016) a 100% Precision is recorded as against an average of 90% in this work. This is because SPARQL query retrieves only profiles that exactly match the query or profiles with one or more skills belonging to the same superclass with the skills in the searching profile, thereby making all retrieved profiles relevant to the search.

iii.) Query relaxation was achieved as evident in the recall rate, which is 100%. This means that all candidate profile on the portal are compared against all the jobs and are indexed in the search result. This is contrary to (Tran 2016) which places strict restrictions on search results returned.

CHAPTER FIVE

SUMMARY, CONCLUSION AND RECOMMENDATION

5.1 Summary

The Internet has evolved as the primary medium for Human Resource Management, specifically job recruitment and employment process. Most classical job recruitment portals on the internet rely solely on the keyword based search technique to locate jobs. However, this technique results often times in high recall with low precision and also without considering the ontological closeness/similarity between these key words. In this work we proposed a technique that uses an ontology based approach to enhance keyword searching by leveraging on the similarity between concepts in the ontology (which represent core skills needed and required for a job) to determine how closely matched an applicant is to a job advertisement and vice-versa . This was achieved by developing a CV Ontology, annotating applicant profile and job postings using a common vocabulary and modifying the semantic concept similarity algorithm to compute matching accuracy and to also rank. The results obtained from this work was compared against similar work by Tran (2016) whose system was implemented and supplied with the same data set used in this work. The results showed that improvements were achieved in overall matching accuracy between core skills supplied by applicants and those required by employers Improvements of 54% and 36% were obtained for Recall and F-Measure respectively, over Tran (2016). Although (Tran, 2016) obtained a better Precision due to the strict usage of SPARQL, Precision and Recall taken in isolation do not determine how effective a search is, but rather the F-Measure, which is the Harmonic mean between Precision and Recall. In this work, an F-measure value of 94% which represent a 36% improvement over Tran (2016) was obtained. This indicates that the

search and matching technique Proposed in this work is very effective and accurate and represents a marked improvement over the approach in the work of Tran (2016).

5.2 Conclusion

In this research work, a modified concept similarity matching algorithm was deployed to a CV ontology in order to match core skills of applicants to employers more accurately. From the results of experiments carried out on the implemented system, the searching and matching technique recorded an improvement in accuracy of matching. Results affirm the relevance of the concept similarity matching algorithm in determining ontological closeness/similarities between concepts on an ontology and how it can be applied to improve the employment process.

5.3 Recommendation

Although the result obtained from this work showed an improvement in accuracy and overall effectiveness of search as compared to Tran (2016), the ontology can be expanded to also take into consideration other parameters that improve recruitment such as “years of working experience” possessed by applicants and required by employers. The algorithm can be further modified to factor in the aforementioned in computing the matching score between Job profiles and applicant profiles. Future work could be looked into in this regard, in other to develop a more holistic and robust job search recruitment portal.

5.4 Contributions to knowledge

A lot of research has been done on enhancing job search in online job portals using ontologies and semantic web technologies, however this work significantly contributes to knowledge as follows:

1. Modifying and Implementing the Concept Similarity Matching Algorithm in job portals to match applicant profiles to job profiles based on core skills. This has introduced the added advantage of quantifying matching accuracy based on ontological closeness which defines semantic similarity between core skills needed and required by applicants and employers alike.
2. Introduction of a ranking technique which returns to users search for positions they are best suited for based on their core skills set, which represents a marked improvement over approaches used in existing literature.

REFERENCES

Abrouk, L. (2006). *Annotation de documents par lecontexte de citation basée sur une ontologie.* M.Sc. Thesis, Universite Montpelier II, France. Available from http://tel.archives-ouvertes.fr/docs/00/14/25/68/PDF/these.pdf Retrieved May 17, 2016

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The Semantic Web. A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities. Available from http://www.scientificamerican.com Retrieved March 14, 2016

Bizer, C., Heese, R., Mochol, M., Oldawaski, R., Tolksdorf, R., & Ecstein, R. (2005). *The Impact of semantic web technologies on job recruitment process.* In: proceedings of International Conference Wirtschaftsinformatik (WI), Bamberg, Germany.

Colucci, S., Di Noia, T., Di Sciascio, E., Donini, F. M., Mongiello, M., Mottola, M. A. (2003). Formal Approach to Ontology-Based Semantic Match of Skills Descriptions. *Journal of Universal Computer Science., 9*(12), 1437-1454.

Dafoulas, G. A., Nikolaou, A. N., Turega, M., & (2003). *E-Services in the Internet Job Market.* In: proceedings of 36th Hawaii International Conference on System Sciences, Hawaii, USA.

Dittmann, L., & Zelewski, S. (2004). *Ontology-based Skills Management.* In: proceedings of 8th World Multi-conference on Systemics, Cybernetics and Informatic, Oslo, Norway.

Fazel, Z., & Fox, M. (2009). *Semantic Matchmaking for Job Recruitment: An Ontology Based Hybrid Approach.* In: proceedings of the 3rd International Workshop on Service Matchmaking and Resource Retrieval in the Semantic Web at the 8th International Semantic Web Conference (ISWC), Washington D.C., USA.

Geetha, G., & Fenelon, S. (2015). Recruitment Based On Ontology with Enhanced Security Features. *The International Journal of Engineering and Science (IJES), 4*(2), 28-32.

Guissé, A., Lévy, F., Nazarenko, A., & Szulman, S. R. (2009). Annotation sémantique pour l'indexation de règles métiers. Available from http://www.irit.fr/TIA09/thekey/articles/guisse-levy-nazarenko-szulman.pdf Retrieved March 16, 2016

Ionecusu, B., Ionecusu, I., Florescu, V., & Tinca, A. (2012). Semantic Annotation and Association of Web Documents: A Proposal for Semantic Modelling in the Context of E-recruitment in the IT Field. *Journal of accounting and management information system, 11*(1), 76-96.

Jike, G., & Yuhui, Q. (2008). *Concept Similarity Matching Based on Semantic Distance.* In: proceedings of Fourth International Conference on Semantics, Knowledge and Grid. IEEE computer society, Beijing, China.

Kavidha, A., & Saradha, A. (2013). Intelligent interoperable application for employment Exchange using ontology. *Webology, 10*(2).

May, F., & Yew, C. (2015). *Intelligent job matching with self-learning recommendation engine.* In: proceedings of 6th International Conference on Applied Human Factors and Ergonomics (AHFE 2015) and the Affiliated Conferences,

Mochol, M., Jentzsch, A., & Wache, H. (2007). *Suitable employees wanted? Find them with semantic techniques.* In: proceedings of Workshop on Making Semantics Web For Business at European Semantic Technology Conference (ESTC2007), Vienna, Austria.

Mochol, M., & Paslaru, B. (2006). *Practical Guidelines for Building Semantic eRecruitment Applications.* . In: proceedings of International Conference on Knowledge Management, Special Track: Advanced Semantic Technologies, Berlin, Germany.

Mulder, W. (2010). Personal informations system e -Entwicklungsstand, Funktionalitat und Trends. Wirtschafts informatik. *Journal of Universal Computer Science., 16*(2), 98–10.

Mustafa, J., Luk, V., & Diana, M. (2007). HR-Semantic Roadmap: Semantic Challenges and Opportunities in Human Resource Domain. Available from http://www.jarrar.info/publications/OOAHR-SemanticsRoadmapV025.pdf Retrieved April 14, 2016

Niaphruek, P. (2012). *Job Recruitment System Using Semantic Web Technology.* In: proceedings of 15th International conference of international academy of physical science Pathumthani, Thailand.

Noy, N. F., & McGuinness, D. L. (2016). Ontology Development 101: A Guide to Creating Your First Ontology. Available from URL: http : / / protege . stanford . edu / publications /ontology%7B%5C_%7Ddevelopment/ontology101- noy- mcguinness.htm Retrieved May 5, 2016

Philip, S., & Mazancourt, H. (2007). Use of HR-XML for a CV Parser: A Software Editors Perspective. Available from http://www.eife-l.org/publications/proceedings/cv07/Lingway.pdf Retrieved May 20, 2016

Quan, D., & Karger, D. (2004). *How to Make a Semantic Web Browser*. In: proceedings of 13th International World Wide Web Conference.

Report., J. s. O. R. S. (2015). Available from http://www.jobberman.com/INFOGRAPHIC-Online-Recruitment-Services-Report-2015.htm Retrieved April 21, 2016

Schmidt, A. (2005). *Bridging the Gap Between E-Learning and Knowledge Management with Context-Aware Corporate Learning Solutions. Professional Knowledge Management.* . In: proceedings of Third Biennial Conference WM, Revised Selected Papers, Lecture Notes in Artificial Intelligence (LNAI).

Schmidt, A. (2006). *Context-Aware Workplace Learning Support: Concept, Experiences, and Remaining Challenges* In: proceedings of European Conference on Technology-Enhanced Learning (EC-TEL 06).

Sheth, A., & Thirunarayan, K. (2013). Multimedia Semantics empowered Web 3.0 Available from https://books.google.com.ng/books?id=IK4v4fluDJgC&pg=RA2PA7&lpg=RAA7 Retrieved April 20, 2016

Tran, H. (2016). *Human Resource Matching Through Query Augmentation for Improving Search Context.* (M.Sc. thesis), Linkopings University Sweeden. Available from https://liu.diva-portal.org/smash/get/diva2:941053/FULLTEXT01.pdf

Vinaya, R., Munjula, R., & Sidna, N. (2015). An unstructured text analytics approach for qualitative evaluation of resumes. *International Journal of Innovative Research in Advanced Engineering (IJIRAE), 8*(2), 2349-2163.

Yahiaoui, L., Boufaida, Z., & Prié, Y. (2006). Semantic Annotation of Documents Applied to e-recruitment. Available from https://www.semanticscholar.org/paper/Semantic-Annotation-of-Documents-Applied-to-E-Yahiaoui-Boufa/pdf Retrieved April 14, 2016

APPENDIX

CODE SAMPLES

SOURCE CODE FOR ONTOLOGY

```
<?xml version="1.0"?>


<!DOCTYPE Ontology [
    <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
    <!ENTITY xml "http://www.w3.org/XML/1998/namespace" >
    <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
    <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
]>



<Ontology xmlns="http://www.w3.org/2002/07/owl#"
     xml:base="http://www.semanticweb.org/6/ontologies/2016/10/untitled-ontology-17"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:xml="http://www.w3.org/XML/1998/namespace"
     ontologyIRI="http://www.semanticweb.org/6/ontologies/2016/10/untitled-ontology-17">
    <Prefix name="" IRI="http://www.w3.org/2002/07/owl#"/>
    <Prefix name="owl" IRI="http://www.w3.org/2002/07/owl#"/>
    <Prefix name="rdf" IRI="http://www.w3.org/1999/02/22-rdf-syntax-ns#"/>
    <Prefix name="xsd" IRI="http://www.w3.org/2001/XMLSchema#"/>
    <Prefix name="rdfs" IRI="http://www.w3.org/2000/01/rdf-schema#"/>
    <Declaration>
        <Class IRI="#Analog"/>
    </Declaration>
    <Declaration>
        <Class IRI="#AngularJs"/>
    </Declaration>
    <Declaration>
        <Class IRI="#Applicant"/>
    </Declaration>
    <Declaration>
        <Class IRI="#Assembly"/>
    </Declaration>
    <Declaration>
        <Class IRI="#BlueSpec"/>
```

```
</Declaration>
<Declaration>
    <Class IRI="#C++"/>
</Declaration>
<Declaration>
    <Class IRI="#CSS"/>
</Declaration>
<Declaration>
    <Class IRI="#CSharp"/>
</Declaration>
<Declaration>
    <Class IRI="#ClassicalWebBased"/>
</Declaration>
<Declaration>
    <Class IRI="#Confluence"/>
</Declaration>
<Declaration>
    <Class IRI="#Delphi"/>
</Declaration>
<Declaration>
    <Class IRI="#Digital"/>
</Declaration>
<Declaration>
    <Class IRI="#Employer"/>
</Declaration>
<Declaration>
    <Class IRI="#FrontEndPath"/>
</Declaration>
<Declaration>
    <Class IRI="#FullStackPath"/>
</Declaration>
<Declaration>
    <Class IRI="#HDL"/>
</Declaration>
<Declaration>
    <Class IRI="#HTML"/>
</Declaration>
<Declaration>
    <Class IRI="#HardwareDevBased"/>
</Declaration>
<Declaration>
    <Class IRI="#JQuery"/>
</Declaration>
```

```
    <Declaration>
        <Class IRI="#Java"/>
    </Declaration>
    <Declaration>
        <Class IRI="#JavaScript"/>
    </Declaration>
    <Declaration>
        <Class IRI="#Lisp"/>
    </Declaration>
    <Declaration>
        <Class IRI="#Lola"/>
    </Declaration>
    <Declaration>
        <Class IRI="#MacRuby"/>
    </Declaration>
    <Declaration>
        <Class IRI="#Matlab"/>
    </Declaration>
    <Declaration>
        <Class IRI="#MyHDL"/>
    </Declaration>
    <Declaration>
        <Class IRI="#MySQL"/>
    </Declaration>
    <Declaration>
        <Class IRI="#OWL"/>
    </Declaration>
    <Declaration>
        <Class IRI="#ObjectiveC"/>
    </Declaration>
    <Declaration>
        <Class IRI="#PHP"/>
    </Declaration>
              <DataProperty IRI="#Years_of_Experience"/>
        <Class IRI="#Person"/>
    </DataPropertyDomain>
    <DataPropertyRange>
        <DataProperty IRI="#Years_of_Experience"/>
        <Datatype abbreviatedIRI="xsd:integer"/>
    </DataPropertyRange>
</Ontology>

<!-- Generated by the OWL API (version 3.4.2) http://owlapi.sourceforge.net -->
```

SAMPLE JAVA SOURCE CODES

```
package semantic;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.HashMap;
import java.util.TreeMap;
import java.util.Iterator;

/**
 *
 * @author Edem's Family
 */
public class Semantic
{
    public HashMap< String, String > path = new HashMap<>();
    public double tree[] = { 0, 2, 1.5, 1.25, 1.125, 1.0625 };
    public double sibbling[] = { 0, 2.2, 1.7, 1.45, 1.325, 1.2625 };
    public static final String DATABASE_URL = "jdbc:mysql://localhost/jobs";

    {
        path.put("matlab", "programming language/hardware based/matlab" );
        path.put("assembly", "programming language/hardware based/assembly" );
        path.put("lisp", "programming language/hardware based/lisp");
        path.put("pascal", "programming language/hardware based/pascal");
        path.put("vhdl-ams","programming language/hardware based/hdl/analog/vhdl-ams");
        path.put("verilog-ams","programming language/hardware based/hdl/analog/verilog-ams");
        path.put("verilog", "programming language/hardware based/hdl/digital/verilog");
        path.put("blue spec", "programming language/hardware based/hdl/digital/blue spec");
        path.put("mhdl", "programming language/hardware based/hdl/digital/mhdl");
        path.put("confluence", "programming language/hardware based/hdl/digital/confluence");
        path.put("lola", "programming language/hardware based/hdl/digital/lola");
        path.put("java", "programming language/software based/java");
        path.put("c++", "programming language/software based/c++");
        path.put("python", "programming language/software based/python");
        path.put("c#", "programming language/software based/c#");
        path.put("perl", "programming language/software based/perl");
        path.put("vb", "programming language/software based/vb");
```

```
        path.put("obj c", "programming language/software based/obj c");
        path.put("mac ruby","programming language/software based/mac ruby");
        path.put("swift","programming language/software based/swift");
        path.put("delphi", "programming language/software based/delphi");
        path.put("java", "programming language/software based/java");
        path.put("xml", "programming language/software based/web dev based/semantic/xml");
        path.put("rof", "programming language/software based/web dev based/semantic/rof");
        path.put("owl", "programming language/software based/web dev based/semantic/owl");
        path.put("php", "programming language/software based/web dev based/classical/full
stack/php");
        path.put("mysql", "programming language/software based/web dev based/classical/full
stack/mysql");
        path.put("ruby", "programming language/software based/web dev based/classical/full
stack/ruby");
        path.put("ruby on rails", "programming language/software based/web dev
based/classical/full stack/ruby on rails");
        path.put("html", "programming language/software based/web dev based/classical/front
end/html");
        path.put("css", "programming language/software based/web dev based/classical/front
end/css");
        path.put("jscript", "programming language/software based/web dev based/classical/front
end/jscript");
        path.put("jquery", "programming language/software based/web dev based/classical/front
end/jquery");
        path.put("angular js", "programming language/software based/web dev based/classical/front
end/angular js" );
    }
    /**
     * @param args the command line arguments
     */
    public void testSem( String search )
    {
        TreeMap< Double, Integer > job = jobSearch( search );
        try ( Connection c = DriverManager.getConnection( DATABASE_URL, "root", "alex1234" ) )
        {
            Iterator jobid = job.values().iterator();
            while   ( jobid.hasNext() )
            {
                Statement s = c.createStatement();
                ResultSet r = s.executeQuery( "select * from job where jobid = " + jobid.next()
);
                System.out.println( r.getString("requirements") );
            }
        }
        catch   ( Exception e ){}
    }
```

```
    public TreeMap< Double, Integer > jobSearch( String search )
    {
        TreeMap< Double, Integer > job = new TreeMap<>();
        try ( Connection c = DriverManager.getConnection( DATABASE_URL, "root", "alex1234" ) )
        {
            Statement s = c.createStatement();
            ResultSet r = s.executeQuery( "select * from job" );
            do {
                job.put( jobDistance( search, r.getString("requirements") ), r.getInt("jobid") );
            } while ( r.next() );
        }
        catch   ( Exception e ){}
        return job;
    }
```

SAMPLE PHP CODES

```
<?php require_once('Connections/jobs.php'); ?>
<?php
if (!function_exists("GetSQLValueString")) {
function GetSQLValueString($theValue, $theType, $theDefinedValue = "", $theNotDefinedValue = "")
{
  if (PHP_VERSION < 6) {
    $theValue = get_magic_quotes_gpc() ? stripslashes($theValue) : $theValue;
  }

  $theValue = function_exists("mysql_real_escape_string") ? mysql_real_escape_string($theValue) :
mysql_escape_string($theValue);

  switch ($theType) {
    case "text":
      $theValue = ($theValue != "") ? "'" . $theValue . "'" : "NULL";
      break;
    case "long":
    case "int":
      $theValue = ($theValue != "") ? intval($theValue) : "NULL";
      break;
    case "double":
      $theValue = ($theValue != "") ? doubleval($theValue) : "NULL";
      break;
    case "date":
      $theValue = ($theValue != "") ? "'" . $theValue . "'" : "NULL";
```

```
      break;
    case "defined":
      $theValue = ($theValue != "") ? $theDefinedValue : $theNotDefinedValue;
      break;
  }
  return $theValue;
}
}

$maxRows_Jobs = 10;
$pageNum_Jobs = 0;
if (isset($_GET['pageNum_Jobs'])) {
  $pageNum_Jobs = $_GET['pageNum_Jobs'];
}
$startRow_Jobs = $pageNum_Jobs * $maxRows_Jobs;

mysql_select_db($database_jobs, $jobs);
$query_Jobs = "SELECT * FROM job ORDER BY dateposted DESC";
$query_limit_Jobs = sprintf("%s LIMIT %d, %d", $query_Jobs, $startRow_Jobs, $maxRows_Jobs);
$Jobs = mysql_query($query_limit_Jobs, $jobs) or die(mysql_error());
$row_Jobs = mysql_fetch_assoc($Jobs);

if (isset($_GET['totalRows_Jobs'])) {
  $totalRows_Jobs = $_GET['totalRows_Jobs'];
} else {
  $all_Jobs = mysql_query($query_Jobs);
  $totalRows_Jobs = mysql_num_rows($all_Jobs);
}
$totalPages_Jobs = ceil($totalRows_Jobs/$maxRows_Jobs)-1;

$maxRows_Applicants = 10;
$pageNum_Applicants = 0;
if (isset($_GET['pageNum_Applicants'])) {
  $pageNum_Applicants = $_GET['pageNum_Applicants'];
}
$startRow_Applicants = $pageNum_Applicants * $maxRows_Applicants;

mysql_select_db($database_jobs, $jobs);
$query_Applicants = "SELECT * from applicant left join users on users.userid =
applicant.applicantid where applicant.fname <> 'NULL' and applicant.mname <> 'NULL' and
applicant.surname <> 'NULL' order by users.dateregistered desc";
$query_limit_Applicants = sprintf("%s LIMIT %d, %d", $query_Applicants, $startRow_Applicants,
$maxRows_Applicants);
$Applicants = mysql_query($query_limit_Applicants, $jobs) or die(mysql_error());
```

```
$row_Applicants = mysql_fetch_assoc($Applicants);


if (isset($_GET['totalRows_Applicants'])) {
  $totalRows_Applicants = $_GET['totalRows_Applicants'];
} else {
  $all_Applicants = mysql_query($query_Applicants);
  $totalRows_Applicants = mysql_num_rows($all_Applicants);
}
$totalPages_Applicants = ceil($totalRows_Applicants/$maxRows_Applicants)-1;


$maxRows_Employers = 10;
$pageNum_Employers = 0;
if (isset($_GET['pageNum_Employers'])) {
  $pageNum_Employers = $_GET['pageNum_Employers'];
}
$startRow_Employers = $pageNum_Employers * $maxRows_Employers;


mysql_select_db($database_jobs, $jobs);
$query_Employers = "SELECT * from employer left join users on users.userid = employer.employerid
where employer.employerid <> 'NULL' order by users.dateregistered desc";
$query_limit_Employers = sprintf("%s LIMIT %d, %d", $query_Employers, $startRow_Employers,
$maxRows_Employers);
$Employers = mysql_query($query_limit_Employers, $jobs) or die(mysql_error());
$row_Employers = mysql_fetch_assoc($Employers);


if (isset($_GET['totalRows_Employers'])) {
  $totalRows_Employers = $_GET['totalRows_Employers'];
} else {
  $all_Employers = mysql_query($query_Employers);
  $totalRows_Employers = mysql_num_rows($all_Employers);
}
$totalPages_Employers = ceil($totalRows_Employers/$maxRows_Employers)-1;?>
```