Facilitating File Retrieval on Resource Limited Devices

A Thesis submitted for the Degree of
Doctor of Philosophy

By
Sadaqat Jan

Department Electronic and Computer Engineering
School of Engineering and Design
Brunel University
May 2011

Abstract

The rapid development of mobile technologies has facilitated users to generate and store files on mobile devices. However, it has become a challenging issue for users to search efficiently and effectively for files of interest in a mobile environment that involves a large number of mobile nodes. In this thesis, file management and retrieval alternatives have been investigated to propose a feasible framework that can be employed on resource-limited devices without altering their operating systems. The file annotation and retrieval framework (FARM) proposed in the thesis automatically annotates the files with their basic file attributes by extracting them from the underlying operating system of the device. The framework is implemented in the JME platform as a case study. This framework provides a variety of features for managing the metadata and file search features on the device itself and on other devices in a networked environment. FARM not only automates the file-search process but also provides accurate results as demonstrated by the experimental analysis.

In order to facilitate a file search and take advantage of the Semantic Web Technologies, the SemFARM framework is proposed which utilizes the knowledge of a generic ontology. The generic ontology defines the most common keywords that can be used as the metadata of stored files. This provides semantic-based file search capabilities on low-end devices where the search keywords are enriched with additional knowledge extracted from the defined ontology. The existing frameworks annotate image files only, while SemFARM can be used to annotate all types of files. Semantic heterogeneity is a challenging issue and necessitates extensive research to accomplish the aim of a semantic web. For this reason, significant research efforts have been made in recent years by proposing an enormous number of ontology alignment systems to deal with ontology heterogeneities.

In the process of aligning different ontologies, it is essential to encompass their semantic, structural or any system-specific measures in mapping decisions to produce more accurate alignments. The proposed solution, in this thesis, for ontology alignment presents a structural matcher, which computes the similarity between the super-classes, sub-classes and properties of two entities from different ontologies that require aligning. The proposed alignment system (OARS) uses Rough Sets to aggregate the results obtained from various matchers in order to deal with uncertainties during the mapping process of entities. The OARS uses a combinational approach by using a string-based and linguistic-based matcher, in addition to structural-matcher for computing the overall similarity between two entities. The performance of the OARS is evaluated in comparison with existing state of the art alignment systems in terms of precision and recall. The performance tests are performed by using benchmark ontologies and the results show significant improvements, specifically in terms of recall on all groups of test ontologies. There is no such existing framework, which can use alignments for file search on mobile devices.

The ontology alignment paradigm is integrated in the SemFARM to further enhance the file search features of the framework as it utilises the knowledge of more than one ontology in order to perform a search query. The experimental evaluations show that it performs better in terms of precision and recall where more than one ontology is available when searching for a required file.

Acknowledgments

I am grateful to all of my colleagues and friends who supported me in different ways during the Ph.D.

I am heartily thankful to my supervisor, Professor Hamed Al-Raweshidy, who supervised my research and guided me from time to time.

I would like to show my gratitude to Dr. Maozhen (Rick) Li for his tremendous support and guidance throughout my research.

I am thankful to all my colleagues at the WNCC and the all staff at the Department of Electronic and Computing Engineering, Brunel University. I am grateful to the Higher Education Commission of Pakistan and the University of Engineering & Technology, Peshawar for funding my Ph.D. in the U.K. I would also like to thank my friend Ibrar Ali Shah for his help and support over the last four years.

Lastly, I owe my deepest gratitude to my mother, my brothers, my wife and my children, Amna and Rayyan, without whom I would not have made it through my PhD studies.

Table of Contents

List of Figures

List of Tables

List of abbreviations

CLDC	Connected Limited Device Configuration
CDC	Connected Device Configuration
DAML	DARPA Agent Markup Language
GMO	Graph Matching for Ontologies
J2ME	Java2 Micro Edition
JCP	Java Community Process
JSR	Java Specification Requests
JVM	Java Virtual Machine
KVM	Kilo-byte virtual machine
kXML	Kilo-byte Extensible Markup Language
MIDP	Mobile Information Device Profile
OAEI	Ontology Alignment Evaluation Initiative
OEM	Original Equipment Manufacturer
OIL	Ontology Inference Layer
OWL	Web Ontology Language
PAN	Personal Area Network
PBP	Personal Basis Profile
PDA	Personal Digital Assistant
PDAP	PDA Profile
PP	Personal Profile
QOM	Quick Ontology Mapping
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
RMI	Remote Method Invocation
RMIP	Remote Method Invocation Profile
SDP	Service Discovery Protocol
SHOE	Simple HTML Ontology Extensions
SMOA	String Metric for Ontology Alignment

SOA	Service Oriented Architecture
SWRL	Semantic Web Rule Language
URI	Uniform Resource Identifier
W3C	World Wide Web Consortium
WWW	World Wide Web
XML	Extensible Markup Language

Author's Declaration

The work described in this thesis has not been previously submitted for a degree in this or any other university and unless otherwise referenced it is the author's own work.

Statement of Copyright

List of Publications

Journal Papers:

- S. Jan, Maozhen Li, G. Al-Sultany, Hamed Al-Raweshidy and I.A Shah "Semantic file annotation and retrieval on mobile devices" , Mobile Information Systems. vol. 7, no 2, pp. 107-122, (2011).
- S. Jan, Maozhen Li and Hamed Al-Raweshidy, "File Annotation and Retrieval on Mobile Devices", international journal of Personal and Ubiquitous Computing – Springer Link (Accepted).

Conference Papers:

- S. Jan, M. Li, G. Al-Sultany and H. Al-Raweshidy, "*File Annotation and Sharing on Low-End Mobile Devices*", Proc. of IEEE International Conference on Fuzzy Systems and Knowledge Discovery FSKD'10, pp. 2973-2977 (2010)
- G. Al-Sultany, M. Li, S. Jan and H. Al-Raweshidy, "*Facilitating Mobile Communication with Annotated Messages*", Proc. of IEEE CIT 2010, pp. 755-760 (2010).
- S. Jan, S. Khan, M. Li and H.S. Al-Raweshidy, "*Distributed Execution of an Inherently Sequential Network Simulator*", IEEE International Conference on Communication Software and Networks (ICCSN), (2009).
- I. A. Shah, S. Jan, S. A. Mahmud, and Hamed Al-Raweshidy, "*Optimal Path Discovery with Mobility Management in Heterogeneous Mesh Network*", (IEEE-ICFCC), International Conference on Future Computer and Communication., (2009)
- Ibrar Shah, S. Jan, and Kok-Keong Loo, "Selfish Flow Games in Non-Cooperative Multi-RadioMulti-Channel Wireless Mesh Networks With Imperfect Information", Accepted in The Sixth International Conference on Wireless and Mobile Communications (IEEE- ICWMC 2010)-Valencia, Spain.

CHAPTER 1

Introduction

This chapter briefly describes the background to the problems investigated in this thesis, motivation of work, aim of research, major contributions and research methodology. Finally, the structure of the thesis is outlined.

1.1 Background

Information retrieval has been a challenging research issue in recent years because of the huge expansion in information resources and technological advances. One of the most successful approaches in information retrieval systems is to annotate the data to give additional descriptions of the archived information. *Cathro* [1] has defined metadata in the following way:

"*an element of metadata describes an information resource, or helps provide access to an information resource*" and its purpose as "*Whether in the traditional context or in the Internet context, the key purpose of metadata is to facilitate and improve the retrieval of information*". *A. Sen* [2] has also analysed its significance in past and recent projects of data integration. Metadata was found to be valuable in the earlier bibliographic retrieval systems where online information used to be accessed through associated metadata [3]. Its use was further extended in several ways for the management and retrieval of text, images, multimedia repositories [4], [5], [6], [7], [8] web documents [9], [10] and file systems [11].

The realisation of the Semantic Web[1] depends on the availability of semantic annotations to exploit the knowledge about information resources. To achieve the aim of the Semantic Web, the resources, weather text or multi-media, must be semantically tagged by metadata so that heterogeneous applications can exploit them. There are many forms and techniques for the semantic annotation of documents, despite their growing number,

[1] http://semanticweb.org/wiki/Main_Page

complexity and potential impact on retrieval. Despite all the integration efforts, there is still a gap between the representation formats of the linguistic tools used to extract information and those of the knowledge representation tools used to model the ontology and store the instances or the semantic annotations [12].

To achieve the interoperability [13], [14] of the Semantic Web, several research studies have been carried out in the last decade that specifically focus on the key issue of ontology alignment. A range of matchers have been proposed to find the similarity between two entities from different ontologies in order to map them for alignment purposes or to use them in instance based query systems. Semantic technologies have been used widely in retrieval systems to search for required documents or images, but no such real efforts have been reported that facilitate file searching on resource-limited devices.

1.2 Motivation

The number of hand-held computing devices like mobile phones, Personal Digital Assistants (PDA)s and other small computing devices has grown exponentially in recent years and they have become an essential part of our daily life. There were 5 billion mobile phone subscriptions reported[2] in July 2010. These devices are not only used for communication purposes but also for education [15], [16], [17], health [18], business [19], social networking [20], [21], [22], entertainment, personal assistance and for many other reasons. As the technology evolves, these devices are equipped with advanced features like built-in cameras, audio/video recordings, multiple communication interfaces and a variety of software applications including utility programs and games. The technological progression in low-end devices has been significant in recent years and even mobile phones with enhanced computational resources and larger storage capacity are now commonly available. Similarly, the number of mobile phone software applications with different functionalities has also grown rapidly in recent years.

When a device user starts using such features and applications, a large number of files are usually generated. It becomes a challenging issue for users to search efficiently and

[2] BBC News: Technology, http://www.bbc.co.uk/news/10569081

effectively for files of interest on the device itself or in a networked environment where a large number of mobile nodes are involved. The limited input and output resources of such devices even makes the case worse for the users to interact with the device. In order to cope up with such a challenging issue, extensive research is required to provide a realistic and effective approach to handle the following limitations:

- Limited output resources; such devices have comparatively small screens [23].
- Limited computing resources; these devices have limited processing capabilities that are proven to be more time consuming when interacting with them.
- Currently, no straight forward mechanism exists, which can efficiently retrieve a required file on such devices or other devices when connected in a networked environment. Specifically, this is the case where these devices have massive memory capacities (usually in gigabytes) available to store a large number and different types of files. Even if a simple indexing technique is used, the exact information about the required file would still be required for its retrieval. Hence, some intelligence is required to be employed, so that users will not be forced to remember the names and contents of their files.

Generally, files on low-end devices are stored in hierarchical directory structures with application-specific default naming settings. For instance, a mobile phone camera generates an image file and usually names it as "*image001*" and a video recorder names its file as "*video001*". These are non-descriptive and very complicated for users to remember when searching for a required file after some time has elapsed. Furthermore, the output, computing and input limitations of these hand-held smaller devices grounds more complexity in terms of efforts and time consumption. Low-end devices usually have smaller keypads or touch screens which make it even more difficult to manually browse the directories or open all the files to check their contents in order to get the required file. In the case of an ordinary mobile phone, the keypad covers a very small area at the front and the size of a single key itself, is smaller than the size of a normal human being fingertip. When mobile users intend to search for a required file manually by browsing the directories, it becomes a tedious task to use smaller-size navigation keys on such keypads. Although, several research studies are carried out and have analysed text entry methods [24], [25],

[26] but it has been found that this can only maximise typing swiftness. The evolution of touch screens in smart phones has partly facilitated ease of use by replacing the hard buttons with soft ones [27]. The main aim of the research presented in this thesis is to investigate a practical approach, which should be capable of retrieving files on low-end mobile and portable devices with minimum effort required by its users in the various aspects that follow:

- *Compatibility*- To propose and implement a framework which can be employed without radically modifying the underlying operating system of the device and which should be flexible enough to use on various platforms with minimal modifications.
- *User-Friendliness*- The implementation of the framework should be user-friendly and diminish the effort needed in retrieving a required file.
- *Intelligent*- The file retrieval mechanism should be intelligent enough to maximise the ease for its users.
- The file search mechanism should be capable enough to extend the search on other connected devices.

The available XML parsers are computationally expensive for low-end devices and therefore a minimum implementation of the parser should be investigated for use in the proposed mechanism. An ontology should be utilized to give additional knowledge to file search queries in order to make the file retrieval results more efficient and accurate. In addition, there should be a scalable search mechanism, which can accommodate the knowledge of more than one ontology in the retrieval system. For this purpose, ontology alignment techniques should be investigated to achieve more accurate and precise mappings of different ontologies. The ontology alignment process should be capable to deal with uncertainties that rise during the mapping process of entities. Furthermore, these alignments should be exploited to facilitate information retrieval and specifically, on resource-limited devices.

1.3 Contributions to Knowledge

This thesis contributes to knowledge in the research area of file-retrieval on low-end devices by proposing novel file-retrieval frameworks which can be used to retrieve files more efficiently and accurately while requiring less effort from users. The computing limitations of low-end devices, their file systems and the viability of using XML technologies are critically analysed in respect to their capacity for file searching. These findings are then used to propose and design a framework, which exploits the XML structure for storing file information and retrieving files. As a case study, the proposed framework is implemented in Java Micro Edition (JME) and named as FARM. The key contributions are summarised as follows:

1. The File Annotation and Retrieval framework (FARM), which extracts the basic file attributes from the underlying file system of the device, uses attribute information as the annotation tags for the corresponding file and parse it using kXML to store in XML structure. The files are annotated automatically on the first use of FARM on a device. In addition to the basic attributes, additional keywords can also be added to annotate any file. The XML document is then searched for the required field with the file to retrieve any file in search. In addition, FARM also incorporates a Bluetooth module to transfer files between connected and authorised devices.

   The framework provides a variety of options to search for a required file on the device itself or even on the other connected devices, if authorised. The stored meta-data of files in an XML format can also be viewed as a browsing list on the mobile screen. At the same time, FARM also allows users to edit or refresh the meta-data at any time. In order to compare the performance of FARM, an additional framework is implemented in JME which can be used to search for a required file based on its name. The performance of the kXML parser was also evaluated in terms of the time it takes to parse the metadata.

2. Semantic File Annotation and Retrieval framework (SemFARM). The SemFARM

uses a generic domain OWL-ontology which defines the most commonly used keywords. By taking advantage of reasoning about the knowledge of the generic ontology, SemFARM enables the device users to retrieve a file without typing in the exact keywords associated with a file. The SemFARM contribution consists of the following:

(a) The design and implementation of a generic ontology which defines the most commonly used keywords that can possibly be used to annotate a file on the device.
(b) The design and implementation of a converter which takes the XML meta-data as an input and automatically produces its corresponding RDF schema in order to utilise in the inference engine.
(c) The computation of similarity degrees that are based on semantic reasoning and used for matching user queries with the published file descriptions.
(d) A search mechanism which navigates through all the statements inferred by the inference engine for the required information regarding a file search.

The SemFARM framework also supports file retrieval on nearby connected devices and provides all the features which are available in FARM. The search module of SemFARM is scalable which means that any ontology can be used in addition to the predefined one. The predefined ontology can also be expanded at any time, if needed.

3. The Ontology Alignment based on Rough Sets (OARS), uses Rough Sets to aggregate the similarity measures of the un-mapped entities from different ontologies. The three basic matchers namely, string-based, linguistic-based and structural-based are used to compute the similarity between two entities from different ontologies. The structure-based matcher itself consists of three sub-matchers to compare the similarities between the super-classes, sub-classes and

properties of the ontology entities. Various aggregating techniques are analysed and compared focusing on their implication on the overall ontology alignment performance. The key contributions in designing OARS contains the following:

(a) The design and implementation of a structural matcher which computes the similarity of two class entities from different ontologies by comparing their super and sub-classes.
(b) Using rough sets to aggregate the results obtained from three basic matchers for unmapped entities in the process of aligning two different ontologies.
(c) The integration of a lexical database as a semantic matcher.
(d) The analysis and comparison of various techniques used to aggregate the similarity results of the basic matchers.
(e) The comparison of alignments results with existing state of the art alignment systems.

4. Presents a file search mechanism which utilises the ontology alignments to further enhance the capabilities of the framework proposed in contribution 3 (above).

   This leads to an improved file-retrieval capability of the SemFARM search module by exploiting the knowledge of more than one ontology. The performance evaluation shows that the integration of alignments further enhances the efficiency and accuracy of file retrieval.

1.4 Research Methodology

The research methodology used for conducting the research presented in this thesis is summarized as follows:

1. Extensive analysis of the capabilities and limitation of the resources of low-end devices in terms of their memory, processing, input and output was performed. JME was used for employing the framework for MIDP compliant devices as a case

study. The proposed mechanism can also be employed for profiles other than MIDP.

2. Design of the proposed framework FARM using XML and kXML parsers.
3. Implementation of FARM by developing several MIDlets using JME for file annotation and retrieval.
4. Performance evaluation and analysis based on file searching tests using the FARM framework.
5. Evaluation of the kXML parser in terms of time consumption, specifically in the FARM framework.
6. Design and development of a generic ontology to define the general keywords using OWL and Jena APIs.
7. Implementation of the SemFARM framework, which uses the generic ontology in file searching and was developed by extending the search module of FARM.
8. The performance of SemFARM is evaluated from a number of perspectives in comparison to traditional mobile file systems and enhanced alternatives.
9. A detailed review of the followings:
   (a) Ontology heterogeneities and their alignment techniques.
   (b) Ontology matching techniques which include string-based, linguistic-based and structural-based techniques to find the similarities between two entities and their implication for overall alignment performance.
   (c) The repercussions for ontology alignment by using various similarity aggregation techniques.
10. Proposed design and implementation of OARS.
11. The design of structural matchers which compare the super-classes, sub-classes and properties of two entities.
12. The implementation of OARS and its matchers in Java.
13. Performance evaluation and analysis using benchmark datasets and comparison with state of the art alignments systems.
14. Integration of OARS in SemFARM to facilitate file search.
15. Performance evaluation and analysis based on file searching tests after

integrating OARS in SemFARM.

1.5 Thesis Structure

This thesis consists of seven chapters, beginning with this introductory chapter which provides a brief synopsis of the thesis. The fundamental concepts and related research work are presented in Chapter-2. It includes reviews relating to the significance of annotations in the field of information retrieval and recent research enhancements with a special focus on those which takes advantage of semantic web technologies in mobile computing environments. Detailed insights into ontology alignment and existing state of the art alignment systems are also presented in Chapter-2. A brief introduction is presented to the various heterogeneities that can exist amongst different ontologies defined in the same domain of concept. The chapter ends with related work relevant to the different contributions presented in this thesis.

Chapter-3 presents a brief overview of the JME platform and kXML parser which are employed in the file annotation and retrieval framework FARM. The annotation and search modules of the proposed framework are elaborated on in terms of their implemented MIDlets. The later sections of the same chapter give a detailed analysis of the performance evaluation of the framework with various aspects specifically concerning its efficiency and accuracy in file retrieval.

Semantic web technologies and their employment techniques are discussed in Chapter-4 to give a broad idea of the utilization of ontologies and their impact on information retrieval systems, particularly in environments where low-end devices are entailed. A brief introduction to an inference engine and its integration for file searching in the framework is also provided in Chapter-4. The semantic based file searching framework SemFARM is presented and followed by a comprehensive evaluation of its performance in respect to various measures.

The fundamental concepts and various matching techniques used in ontology alignments are explained in Chapter-5. An ontology alignment system (OARS) which uses rough sets to map the entities from two ontologies is proposed and discussed in this

chapter. Three basic similarity matchers and their implications on the performance of ontology alignment are elaborated on, and various aspects of the performance of the OARS are also evaluated in detail. Various aggregating methods are also evaluated in order to signify the employment of rough sets in aggregating final similarity results for mapping two entities from different ontologies.

Chapter-6 is dedicated to the implementation process where the features of ontology alignments are integrated with SemFARM in order to empower the search module to take advantage of the knowledge presented by more than one ontology. A detailed overview is presented to describe various techniques which make use of ontology alignments. This integration is also evaluated to demonstrate the significance of using ontology alignments in file retrieval on low-end devices.

Finally, Chapter-7 concludes the research findings of the thesis and suggests future work that may be carried out in connection with the research presented in this thesis.

CHAPTER 2

Literature Review

The research presented in this thesis deals with the design of various systems, which facilitate file retrieval on resource-limited devices. For this purpose, various technologies have been investigated and it is therefore logical to present and highlight the core concepts and fundamental principles before proceeding into the research presented in the following chapters. In order to propose the file retrieval system, two main approaches namely, XML-based and semantic-based, are considered for searching for a required file.

An overview of semantic web technologies is provided in order to exploit these concepts in file retrieval framework and specifically in respect to resource-limited devices. Ontology alignment is significant in dealing with various heterogeneities in the semantic web and issues of interoperability between the information systems. Various types of heterogeneities between ontologies and their matching techniques are discussed to elaborate the mapping processes in alignment systems. The related literature is reviewed and summarised at the end of this chapter.

2.1 kXML Parser

kXML is a lighter and compact version of XML parser which is specially designed for low-end devices and exclusively used on JME platform. Extensible Markup Language (XML) [30] is a metamarkup language which was endorsed by W3C [28] and became universally supported specification for exchanging document and data across applications and platforms [29]. It has standard syntax for meta-data and standard structure for document and dat. The human readable plaintext form of XML makes it application independent and readable to everyone. In addition, it provides a very simple and standard syntax for encoding. XML documents needs to be accessed and manipulated by processor called XML parsers, which tends to be bulky and requires heavy runtime memory. kXML [31] is widely used pull parser adopting the MIDP requirements. There are three types of parsers, which are stated as below:

(a) Model parsers- they create a representation of the whole document after reading it and hence require more memory than the other types of parsers [32].
(b) Push parsers- they always process data definitions before the document and a complete tree structure is created in the memory.
(c) Pull parsers- they read the document in pieces and the application drives the parser through the document by repeatedly requesting the next piece.

The generation of this tree is memory expensive and thus push parsers are not suitable for low-end devices. The parsers, by using recursive functions, structure the document tree. The size of kXML 2 Jar file is only 43 KB and can further be reduced by using an obfuscator.

2.2 Semantic Web Technologies

Ontologies play a vital role in semantic interoperability as they define basic terms, relations of a domain concept and rules for linking these terms and relations [33], enabling machines to process information between heterogeneous applications. The main reasons for developing an ontology are given as below [34]

- To share common understanding of the structure of information among people or software agents.
- To enable the reuse of domain knowledge.
- To make domain assumptions explicit.
- To separate domain knowledge from the operational knowledge.
- To analyze domain knowledge.

The use of ontologies is evident in almost every field of information system like business, information security, bio-information and knowledge management [35], [36], [37], [38], [39]. Several implementations and context-aware systems have been developed on the Semantic Web technologies [40] such as ontology, RDF [41] and OWL [42]. RDF is a standard model for data interchange which is widely used to share and communicate ontology and it also offers common properties and syntax for describing information. XML only addresses the document structure while RDF provides a data model which can be extended to address ontology representation techniques. RDF does need translation because a domain model can be presented for defining objects and relationships. RDF is also capable enough to share the knowledge between different metadata languages [43].

However, the cardinality constraints cannot be defined by using RDF, which is one of its major limitations. Several ontology languages were proposed which includes SHOE [44] and OIL [45]. OWL was designed to use by applications which need to process the information contents and representing machine interpretable contents on the web. Comparatively, OWL also adds more vocabulary with a formal semantics and allows power that is more expressive.

The main advantage of OWL over the use of RDF is the ability to define cardinality constraints in ontologies. OWL itself is an evolution of DAML+OIL [46] and divided into three sub-languages, OWL-Lite, which provides hierarchy of classification and constraints; OWL-DL have maximum expressiveness with computational completeness and OWL Full has maximum expressiveness with out computational guarantee. The Jena2 [47], [48] toolkit which provides the ability to parse and perform reasoning based on real standards, have

implemented in SemFARM as presented in Chapter-4. It is a leading toolkit for java programmers in semantic web [49] and gives access to a range of inference capabilities.

The reasoning subsystem of Jena2 allows various inference engines to be plugged-in, which are used to derive additional information from base RDF combined with ontology definitions. Types of inference can be divided into two main types, namely standard and rule based. Standard inference includes RDFS and OWL reasoners while in rule based inference, Jena allows the programmers to define their own rules using Jena APIs.

The Jena2 inference structure [50] shown in Figure 2-1, explains that reasoner is accessed through model factory to associate data developing a new model which is called an inference model. The collection of RDF statements, sometimes refers to graphs, are associated with ontology definitions, which gives such additional statements that cannot directly be derived from RDF alone. In the SemFARM framework, OWL reasoner is used which binds the generic ontology definitions with XML metadata dynamically converted to RDF model, as explained in Section 4.2, Chapter-4.

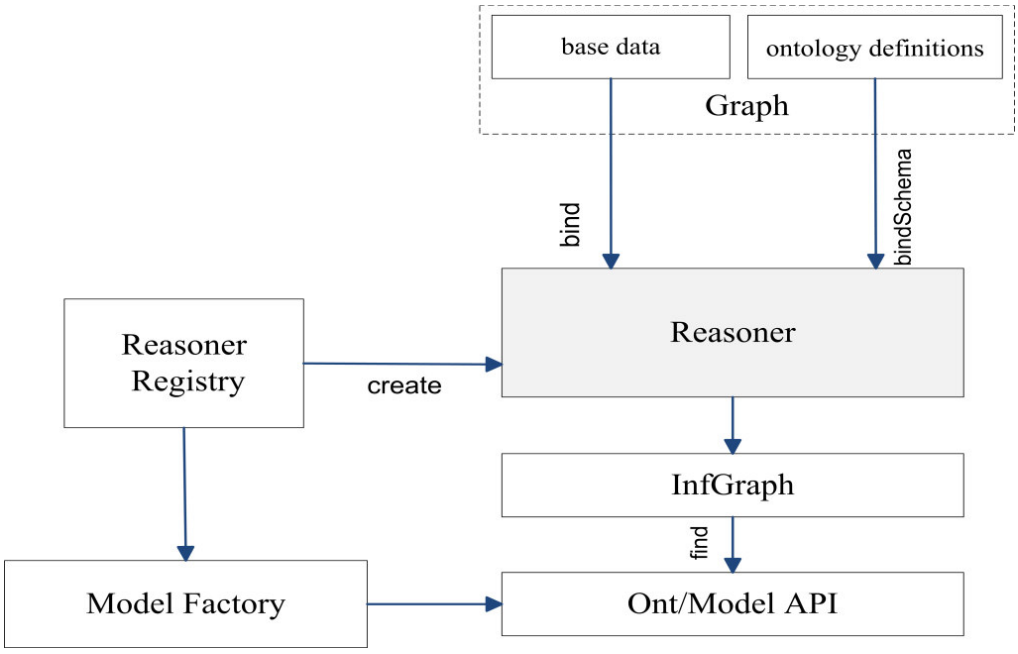

Figure 2-1: Block diagram of Jena inference structure.

2.3 Ontology Alignment Process

This section presents a brief overview of ontologies specifically with respect to their heterogeneity and matching. An ontology is the specification to conceptualize a domain in terms of *concepts*, *attributes* and *relations*. The formally organized set of; *concepts* represent a domain, *relations* describe the relationship amongst the concepts, *attributes* define properties of concepts and the boundary conditions on them are defined by axioms [51], [52]. The concepts are usually organized into hierarchical manner.

2.3.1 Ontology Heterogeneity

Overall ontology heterogeneities have been categorized in many aspects and presented in detail reviews [53], [54], [55], [56]. However, there are two major and most common types of heterogeneity namely *semantic* and *terminological* heterogeneity. *Semantic* heterogeneity occurs due to various reasons like using different axioms or disparity in modelling the same concept. For example, the object property "*address"* may have used for the concept namely *"organization"* in one ontology and may have used for "*Publisher*" in the second ontology. *Terminological* heterogeneity emerges by the using synonyms or different names for the same entity in different ontologies. In Figure 2-2, for example, the entity named as "*Publisher"* in one ontology may have a different name like "*PublishedBy"*in the second but both represent the same entity. The fraction of an ontology shown in Figure 2-2 is taken from one of the ontologies used in OAEI 2010 benchmark tests. The semantic heterogeneity has been the most challenging task in a matching process because it derives from the difference in design or scope of ontology domains in the process of knowledge presentation. Both types of heterogeneities are considered in the ontology alignment system (OARS) proposed in Chapter-5, because they can occur individually, together or in some form of their variations.

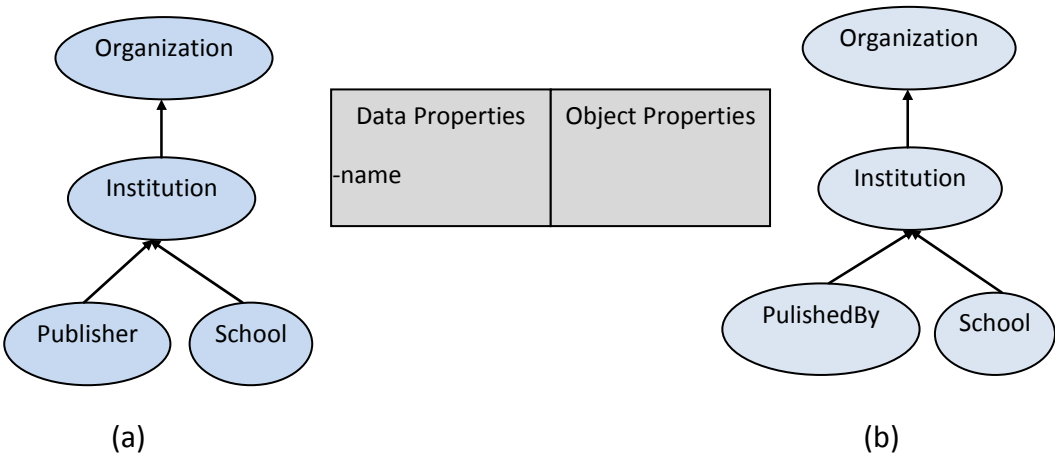

Figure 2-2: An example of ontology mapping.

2.3.2 Ontology Matching

Ontology matching process is to find the semantic mapping between two ontologies. Entities of the ontologies are compared to find correspondences between them, however they do not necessarily have to be the same but they should have certain degree of semantic similarity. This degree of semantic similarity can be used as the alignment threshold in the ontology alignment process. It has been a challenging task to find the semantic similarity between the entities of two semantically heterogeneous ontologies. For this purpose, there should be some information available about the internal structure of entities in order to match them. OWL is an emerging language to represent ontologies in semantic web and recommended by World Wide Web (WWW). As its vocabulary is used to describe the semantics of ontology, it can also be used to find some indications for matching entities during the ontology alignment process. In Figure 2-3, we present a part of the OWL syntax, which is used for the same fraction of ontology shown in Figure 2-2. For example, *owl:Class rdf:ID="Institution"* is used to define a class and it's name is Institution. Similarly, the syntax *rdfs:subClassOf* defines a class which is a sub-class of another defined class in ontology. The *owl:ObjectProperty* and *owl:DatatypeProperty* are used to define the object and data properties. Furthermore, properties can also have sub-properties which are define by the syntax *rdfs:subPropertyOf*. The *rdfs:domain* and *rdfs:range* syntax are used to classify

the range and domain of properties, showing that a property is associated to which classes and what type of values a property may have.

```
<owl:Class rdf:ID="Institution">
  <rdfs:subClassOf rdf:resource="http://xmlns.com/foaf/0.1/Organization"/>
  <rdfs:label xml:lang="en">Institution</rdfs:label>
  <rdfs:comment xml:lang="en">An institution.</rdfs:comment>
  <rdfs:subClassOf>
   <owl:Restriction>
        <owl:onProperty rdf:resource="#name"/>
        <owl:cardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">1</owl:cardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
                                                    ……
        <owl:onProperty rdf:resource="#address"/>
        <owl:maxCardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">1</owl:maxCardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
 </owl:Class>
                                                    ……
<owl:ObjectProperty rdf:ID="institution">
  <rdfs:domain rdf:resource="#Report"/>
  <rdfs:range rdf:resource="#Institution"/>
  <rdfs:label xml:lang="en">institution</rdfs:label>
  <rdfs:comment xml:lang="en">The sponsoring institution of a technical report.</rdfs:comment>
 </owl:ObjectProperty>
                                                    ……
–<owl:Class rdf:ID="School">
<rdfs:subClassOf rdf:resource="#Institution"/>
<rdfs:label xml:lang="en">School</rdfs:label>
<rdfs:comment xml:lang="en">A school or university.</rdfs:comment>
</owl:Class>
                                                    ……
–<owl:ObjectProperty rdf:ID="school">
<rdfs:range rdf:resource="#School"/>
<rdfs:label xml:lang="en">school</rdfs:label>
<rdfs:comment xml:lang="en">The name of the school where a thesis was written.</rdfs:comment>
</owl:ObjectProperty>
```

Figure 2-3: A fragment of OWL ontology.

In Figure 2-3, the syntax *owl:ObjectProperty rdf:ID="school"* indicates the object-property named and labeled as "school" while syntax *"<rdfs:range rdf:resource="#School"/"* shows that the property is associated with class "School". This information greatly helps in describing the internal structure of an ontology. There are also a large number of matchers

which are used to deal with terminological heterogeneity. These types of matchers like string-based and linguistic-based, does not take into account the structural position of the entity and operates on element level while comparing. These matchers are mostly used in schema based matching systems (see [56] for more detail). For example, the *Publisher* and *PublishedBy* can be compared by using string based matchers to find the similarity. External resources are always helpful in finding matches where some background knowledge is required about the entity names.

WordNet is an example of the widely used external resource and many ontology alignment systems have exploited its capability in different ways. For example, several mapping systems have translated the entity labels to their respective WordNet senses and then drawn the mapping from there [57] ,[58], [59]. While *J. kwan et al*. [60] exhaustively used the relationships of synsets to measure the lexical similarity between the entities. LOM [61] is another example of alignment tool which make use of lexicon-based matching.

2.3.3 Ontology Alignment

The Ontology alignment process greatly varies and depends on the approach or algorithm used in the system. The process may be varying in degree of mapping automation, the utilization of structural and lexical similarities and the degree of such similarities. Mappings may be completed in one of the three modes, which includes manual, semi-automatic and automatic. In manual mapping, the user does the mapping by hand while in semi-automatic; the system suggests some mappings to the user for rejection or approval. Using automatic mapping, the system does all the process by itself. The manual mapping is the most time consuming but also gives more accurate results compared to the other two modes. The time and accuracy tradeoffs decision is made according to the application and usage scenario.

Alignment systems may also be different in use of external resources in their matching processes such as web resources, external ontologies, dictionaries or semantic resources

like WordNet[3] etc. Some of these systems use learning methods to improve mapping by using previous mapping results. OARS alignment system does not required any user intervention in the alignment process and it is fully automatic. Figure 2-4 shows a typical example of mapping two entities namely *Publisher* in source ontology and *PublishedBy* in the target ontology. Their structural similarity is exactly equal in terms of super-classes while the string-based similarity will not be equal by using any of the widely used string based matching techniques. Semantically, the entities supposed to be aligned by an alignment system, as it is suggested by the snippet of two ontologies given in Figure 2-4; however, it totally depends on the algorithm which is used in an ontology alignment system.

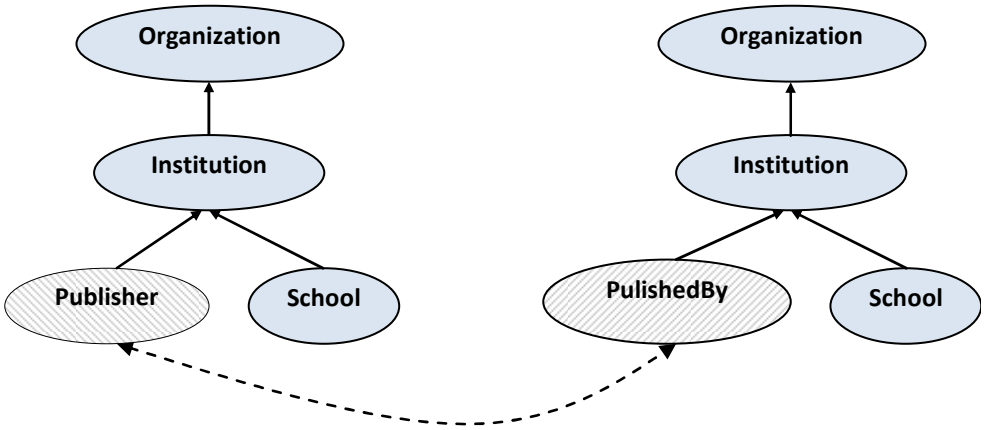

Figure 2-4: Mapping options of two entities from two different ontologies.

2.4 Related Work

2.4.1 Related work on Information Annotation

Several techniques have been proposed and implemented to annotate information on low-end devices and most of these efforts have been made to handle large number of images and videos. However, no real efforts have been made to annotate all types of stored files. Obviously, annotation makes retrieval more efficient not only for images and videos but for any information even if it is on files. For example, Flickr [62] is a special purpose web

[3] http://wordnet.princeton.edu/

service for sharing user uploaded photos and ZoneTag [63] is a tool to annotate camera photos. The ZoneTag mobile application, which is also supported and analyzed by Naaman *et al*. [64], suggests context based tags and some additional tags when a photo is taken on mobile phone camera. The importance of tags and annotation can be determined when retrieving a required photo on Flickr where a photo with more tags can easily be retrieved when compared to a photo with fewer tags. Furthermore, another approach was proposed by Karypidis et al. [65]. It involves annotating photos taken by mobile phone cameras by adding contextual information to them. Devices were operated in a Personal Area Network (PAN) to maintain a shared perception regarding the context to annotate files. The context information was stored on a common repository with the file annotation process being automatic. In order to use tagging in image retrieval, A. Wilhelm et al. [66] proposed a system to annotate images at the capture time by adding Phone ID, username, date and time.

Similarly, a framework was described by Monaghan et al. [67] to use web services, sensors and ontologies to create meaningful annotations. However, most users do not add tags to their information mainly because of the time constraints or a misunderstanding of its importance, especially on devices with limited user interface capabilities. This constraint was considered while proposing the FARM framework where the annotation process was implemented as an automatic process. Even if a user does not enter any information about the file, the process can still annotate files with their basic attributes. A detailed analysis of the motivation for and importance of annotation and tagging was carried out in [68].

However, network coverage is essential for the work proposed in [65] because the information used as metadata was stored on a server and verified by the user before use, and in [67] for web services and access ontologies. To annotate the images a web-based tool was proposed [69], which allows annotating the images and sharing these annotations.

2.4.2 Related Work on Semantic Web Technologies

Various research studies have adopted and proposed several approaches towards efficient file retrieval. Most of these efforts aim at the retrieval of documents/WebPages on

the internet because of the larger volume of information [70][71]; however, as the information volume increases on local resources like the PAN, desktop computers and mobile phones, efforts can now be seen towards efficient information retrieval on all such environments. For example, a semantic file system [72] is proposed for file retrieval using virtual directories and extendable UNIX based file system integrating search functions. Similarly, for supporting semantics in file systems, TagFS [73] is proposed which allows file tagging and the tag-based browsing of information objects on top of an underlying file system.

Information retrieval becomes more challenging task as the restrictions increases in any mobile computing paradigm. A.B. Waluyo et al. [74] presented a survey in which they differentiate the query optimization and processing mechanisms in mobile databases and presented a state-of-the-art in data management for location-dependent query and processing techniques [75]. Various studies shows the importance of tagging [68], concluding that annotation makes the retrieval more efficient, not only for images and videos but for any type of information including files retrieval.

The semantic approach is also extended to mobile devices for picture retrieval, where pictures are annotated with contextual information and used to index each of them [76]. Similarly, the contextual ontology was introduced and successfully implemented in several research efforts. For example, context ontology for mobile devices was developed from embedded mobile sensors [77] for using the resources efficiently, the FLAME2008 platform [78] was successfully developed to support mobile users with personalized context-aware services, and the context ontology was used in a prototype to supervise the health condition of elderly people in runtime [79]. Iwamoto et. al. [80] proposed a design called *uPhoto*, in which context based annotation was implemented by extracting information automatically from embedded sensors and used them as image annotation.

Ontology-based photo annotation was also proposed in [81] to annotate photos using knowledge stored in a RDF schema and the annotation process was not fully automatic. Context based annotation was implemented by extracting information automatically and using it for image annotation in various proposed systems [82]. It is worth mentioning that most of the aforementioned solutions use common repositories to store

metadata. Using a common repository means that network coverage will be required to store and retrieve metadata, making the system reliant on a network medium. However, to the best of our knowledge no such real efforts have been made to annotate or develop ontology for the common keywords which can be used as meta-data for all types of stored files on a mobile phone or other hand held device.

Semantic technologies are used in several research studies supporting pervasive and ubiquities mobile computing. For example, Izumi et al. [83] examined the design of social context-awareness ontology for their implementation of a prototype to supervise elder people in a ubiquitous computing environment and Guo et al. [84] used ontology for dealing with objects in order to search physical artefacts and detect hidden objects in a smart indoor environment.

2.4.3 Related Work on Ontology Alignment

In recent years, research communities from academia and industry have presented many ideas for reducing semantic mismatch problems with the aim of diminishing manual intervention in the matching process. For this purpose, several alignment systems have been proposed. These includes automatic, semi-automatic, application-specific and general-purpose systems. Aspects of these systems are analysed and reviewed in [85], [86], [87].

The schema matching techniques [88] also have been intensely examined by the research community, as the ontology alignment process primarily requires identifying the correspondence between semantically related entities. In these matching systems, the two most widely used techniques are lexical and structural, along with their extensions and variations. In lexical matching, string similarities are measured between two entities, regardless of their hierarchical or internal relation with other elements in the ontology.

The basic intuition behind such techniques is that the more two strings are similar, the greater the probability that they will represent the same concept in different ontologies. The strings, which may be in form of the label or description of an entity defined in an ontology, are treated as sequence of letters. I-Sub [89] is an example of such a matcher which uses string comparison techniques.

It not only utilizes the commonalities between the strings but their differences are also taken into account for comparison. In addition to string-based similarity, language-based similarities are also used as a matching technique where strings are treated as words bearing some meanings depending on the language or resource. These meanings, in turn, are used to compare the concepts in ontologies and lead to precise alignment.

For this purpose, linguistic normalisation techniques or/and external linguistic resources such as dictionaries and lexicons are also used in the matching process. There are several systems, for example WordNet, in which external lexical databases have been exploited to match entities by comparing their label/name information with corresponding synonyms used in a different ontology. The similarity function employed by M. A. Rodrigauz et al. [90] to determine the similar entity classes is based on a matching process [91] which uses synonym sets along with other available information from ontology specifications. Other features of such a lexicon may also be exploited to find the relationships between entities by finding for example hypernym, hyponym, meronym and holonym and so on. in addition to synonyms [92], [93].

Using structural matching techniques, the structural positions or/and relations of the entity with other elements in ontology are compared. The comparison is made between the entities based on their set of properties, domain, data-types, cardinality and so on. The other important and widely used structural matching technique is to compare the relational structures of the entities where the neighbours, super/sub classes and paths are compared. GMO [94] is such an example of a structural matcher which uses RDF graphs to present ontologies and compare their structural similarities. One of the features of GMO is that it can still perform well even without any predefined alignment as input. V-Doc [95] matcher, measures the context of domain entities in terms of their meanings in the Vector Space Model. Words are extracted from descriptions of entities and it neighbours to structure the vectors in word space.

However, it should be pointed out that any technique in isolation like GMO or V-Doc is not adequate enough to give an accurate mapping result and for this reason, we have implemented a combination of string, linguistic and structural-based matchers in OARS to map two entities. The ASMOV [96] is an automatic ontology matching tool which uses both

structural and lexical matchers to calculate a similarity for ontology integration. Its algorithm is designed to automate the alignment process using the weighted average of measurements of similarity to obtain a pre-alignment iteratively which is then verified for semantic inconsistencies. ASMOV mainly considers lexical, hierarchical, restriction and extensional similarities of an entity for a weighted average. The semantic verification process examines the correct correspondences and incompletenesses using predefined inferences. It requires more than one execution to finalize the mapping result and the results of the intermediate iterative executions are employed to refine the subsequent processing phases of alignment. The ASMOV could be computationally expensive because of the iterative nature of its algorithm but gives good results as shown in Ontology Alignment Evaluation Initiative (OAEI)-2010 results [97]. The ASMOV algorithm works well on OWL-DL ontologies.

The SOBOM [98] algorithm is implemented in java and designed for general-purpose ontology alignment. Generally, it finds the anchors in the first step and uses Semantic Inductive Similarity Flooding to flood similarity among concepts. Finally, it utilizes the results of SISF to find relationship alignments. Another example of an ontology alignment system is AgrMaker [99]. It uses three-layer architecture in which a number of different concepts and structural based matchers are included and later combines the results by Linear Weighted Combination using a local confidence quality measure.

Similarly, CODI [100] is based on the Markov logic based probabilistic alignment system, which transforms the alignment to a maximum-a-posteriori optimisation problem. It combines lexical similarity measures with schema information for matching entities in the alignment process. TaxoMap [101] takes into account the labels and sub-class descriptions in ontologies for alignment and employs the Partition Based Matching algorithm [102], which allows the use of predefined equivalence mappings to partition the ontologies into pairs of possible mappings. The ontology alignment is considered as an optimisation problem in MapPSO [103] where the Discrete Particle Swarm Optimization algorithm [104] is applied for solving the problem. Using the MapPSO approach all particles are updated and adjusted iteratively around the best representing particles in the swarm. The quality of alignment in MapPSO is decidedly depends on the selection of matchers and aggregators.

RiMOM [105] is an ontology alignment system that uses combinational approach in

the alignment process. It uses multiple matchers to discover lexical and structural similarities between entities and exploits Bayesian decision theory in order to map them. The basic matchers which are considered as separate strategies compare the taxonomy, constraint, description, name, instance and name-path in mapping process. The user input is also allowed to improve the mapping in alignment process. The enhanced version of RiMOM [106] exploits most of the available ontological knowledge by using these strategies via a strategy selection technique, combines all the similarity values using a sigmoid function, and then initiates an alignment refinement algorithm to finalize the alignment process. However, the parameter settings in RiMOM are highly dependable on the preprocessing step where two similarity factors are compared in ontologies and weights are then assigned to different factors for combining the final results. This means that if two ontologies have more structural similarities, a higher value will be assigned to the weight of structural similarity in combining the final result. Therefore, the mapping of those entities which have other similarities, will suffer, because the same parameters will be used for all entities. The OARS alignment system proposed in Chapter-5, we use Rough Sets classification for each entity individually and the mapping decision is made on entity bases, which does not affect the overall decision of other mappings.

Falcon-AO [107] also uses the combination of linguistic, string based, structural and partition based matchers in the mapping process. These are V-Doc [95], I-Sub [89] and GMO [94]. It requires a similarity combination strategy in order to combine the similarity value resulting from each matcher. A set of coordination rules is also used to reduce structural heterogeneity as a pre- mapping process. The alignment results are returned for the equality and sub-sumption between classes and between properties of ontologies. User intervention is also required and it allows users to evaluate the precision, recall, and F-measure of a matching method given as reference alignment. Using linguistic similarity, Falcon-AO does not differentiate between class and properties while in OARS the linguistic matcher is used for classes and properties separately.

The PROMPT [108] system was developed to support various ontology mediation techniques and it suggests the classes and properties for aligning. It uses linguistic and structural similarity measures to map two entities. PROMPT performs all the changes

automatically and resolves any found conflict by suggesting new mappings to the users. PROMPT is a very useful alignment system where users are involved in the aligning processes. LILY [109] also uses linguistic and structural similarity measures to align the entities from different ontologies. It applies a propagation strategy to generate further alignments and then uses classic image threshold selection algorithm for best suitable threshold. Finally, it extracts the final results, based on the most stable marriage strategy. The QOM [110] ontology alignment system employs the RDF triples as features and it applies heuristic method for mapping the entities. It computes the similarities by using various functions and heuristics but avoids the complete pair-wise evaluation of ontology trees. QOM uses sigmoid function to aggregate the results of various similarity measures. The response time of QOM alignment system is faster than PROMPT. The alignment systems presented in [98], [99], [100], [101] and [103] uses different mapping approaches but have not considered the uncertainty issue during the alignment process of two entities.

2.5 Summary

This chapter has presented basic concepts pertaining to the contributions presented in this thesis. An overview has been given about different types of parsers and specifically kXML, which is designed for resource-limited devices. Various concepts and issues regarding semantic technologies were also presented along with their most common resolutions. The key issue of semantic heterogeneity was explained and a variety of techniques were analysed regarding their significance in overall interoperability between information systems. Finally, literature reviews were presented that relate to the contributions of thesis.

CHAPTER 3

File Annotation and Retrieval on Low-end Devices using XML

To deal with a challenging task of handling large number of files on devices with limited input, output, memory storage and processing capabilities, this chapter presents a practical approach to retrieve the stored files more accurately and efficiently. For this purpose, a framework is proposed and implemented namely (FARM) [111], which primarily exploits the functional features of XML technology for accumulating the meta-data of all the files on a device. The framework automatically traverses the directories and extracts the basic file attributes from the underlying operating system of the device. The metadata is stored locally, which gives this platform a two-fold gain. Firstly, the FARM does not require any common repository and hence do not require any communication medium to store and retrieve metadata. Secondly, the file search query is performed in a distributed fashion when more than one device is searched for files.

FARM is implemented in J2ME by developing several MIDlets to validate its efficiency and accuracy in file retrieval on low-end devices. Furthermore, the framework is equally efficient for searching the required file in networked environment where devices are connected through Bluetooth. Several additional MIDlets have also been integrated to make the FARM a user friendly framework which includes the additional search, annotation and Bluetooth features. The annotation process and search modules are discussed in detail to elaborate their working process. The performance of the framework is evaluated in terms of precision and recall along with the probabilistic evaluations.

3.1 Overview of JME platform

It is essential to discuss the J2ME platform and its MIDP stacks before getting into the details of kXML parser. Users of mobile phone and other hand-held computing devices expect the same high performance and full-featured applications as they find on a desktop or laptop. To meet such expectations and enable the developers to utilize the existing software support, Sun Microsystems developed a platform Java Micro Edition (J2ME) in 1999, which offers a robust environment to support applications on resource-limited devices like mobile phones, PDA, embedded systems etc [112]. J2ME inherits the powerful features of the Java programming language by designing a lightweight virtual machine (KVM) which is capable of providing a secure and efficient execution environment on resource limited devices [113].

It is compatible with all Java enabled devices which runs Java Virtual Machine. Nokia, Ericsson, Motorola, Panasonic, Nextel and many more have Java enabled devices [114]. Different hardware configuration on these small computing devices was a challenging task for Java Community Process (JCP) which is a mechanism to develop standard for Java technology [115], however the challenge was successfully overcome by defining ***Configurations*** and ***Profiles***. *Configuration* is basically the run-time environment and classes operating on a device while *profile* is the set of domain specific classes to implement relevant features on a related group of low-end devices. The J2ME three layered architecture can be depicted in Figure 3-1 and has been the perfect environment for developing applications for small devices [116], [117].

A compact and stripped-down version of virtual machine, called K Virtual Machine (KVM) was developed to make the architecture more modular and scalable. KVM is the smallest possible Java virtual machine that maintains almost all aspects of java programming language and can run on constrained devices with a few hundreds of kilobytes of available memory [118]. As shown in Figure 3-1, the virtual machine directly interact with Configuration layer, there are two configurations available, Connected Limited Device Configuration (CLDC) and Connected Device Configuration (CDC). CLDC is used for devices with small amount of memory and limited computing resources as recommended in its

specification standardized by JSR-139 [119]. These devices usually have 160 KB to 512 KB of memory and are battery powered. The second configuration CDC is specified for devices with 32-bit architecture and having 2MB of memory. CDC devices can implement a complete JVM.

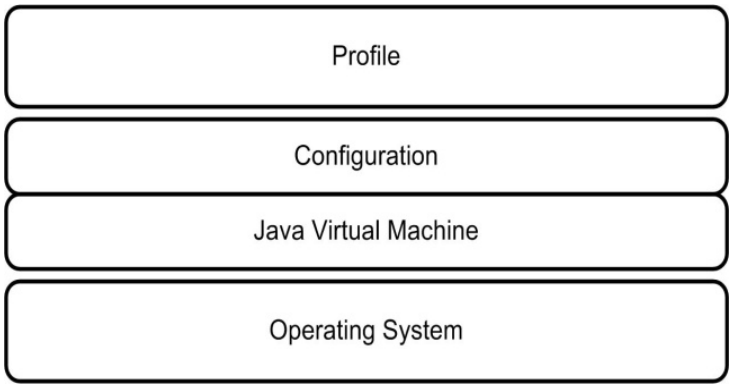

Figure 3-1: Three layered architecture of J2ME platform.

Configuration layer interacts with profile layer in J2ME architecture, which has several profiles consisting of Java classes like Foundation Profile, Game Profile, Mobile Information Device Profile (MIDP), PDA Profile (PDAP), Personal Profile (PP), Personal Basis Profile (PBP) and Remote Method Invocation (RMI) Profile (RMIP) [120]. However, only the MDIP will be elaborated here as it has been used in the FARM framework proposed in this chapter. A complete picture of JME architecture is shown in Figure 3-2 for better understanding of the position of all the required components used in our framework. MID Profile is used with CLDC configuration as it contains classes that give networking, storage and user interface capabilities as shown in Figure 3-3. For some of the Original Equipment Manufacturer (OEM) applications CLDC and MIDP services can be used depending on host operating system of the device. MIDP specification was developed under Java Community Process and is available online [121]. Besides these great advantages of J2ME, there are some weaknesses as well that still need to be addressed. Firstly, the rapid changes in technology should be coping by the JCP's specifications. Secondly, programming with J2ME is not as simple as with standard java language. Finally, still there are some security issues; which are minor but still need to be addressed, which are analysed in [122]. MIDlets are applications which uses MID profile of CLDC specification. The software which implements the MIDP, runs in KVM.

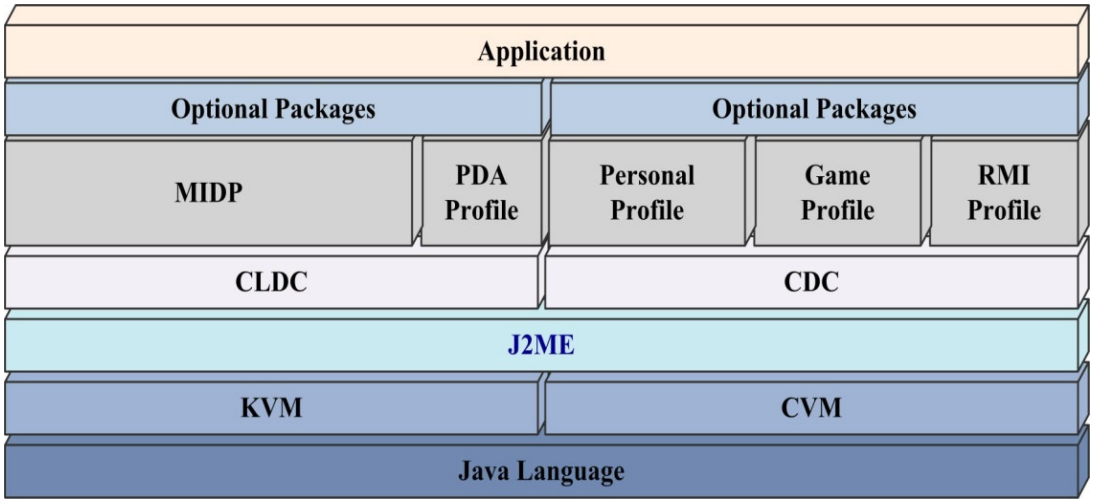

Figure 3-2: J2ME Architecture.

The KVM is supplied by CLDC and provides additional services to facilitate the application code. MIDP requires 128 KB of RAM to implement itself, memory for CLDC, 32 KB for java heap and at least 8 KB of non-volatile memory. The display requirements are the screen size which should be at least 96 pixels wide and 54 pixel high and screen must support at least two colours. The MIDP specifications require that the device should have input mechanism to type 0 to 9 and select keys. Now a days, mobile phones, PDAs and other hand-held computing devices are equipped with much advanced featured not only fulfilling these requirements but beyond.

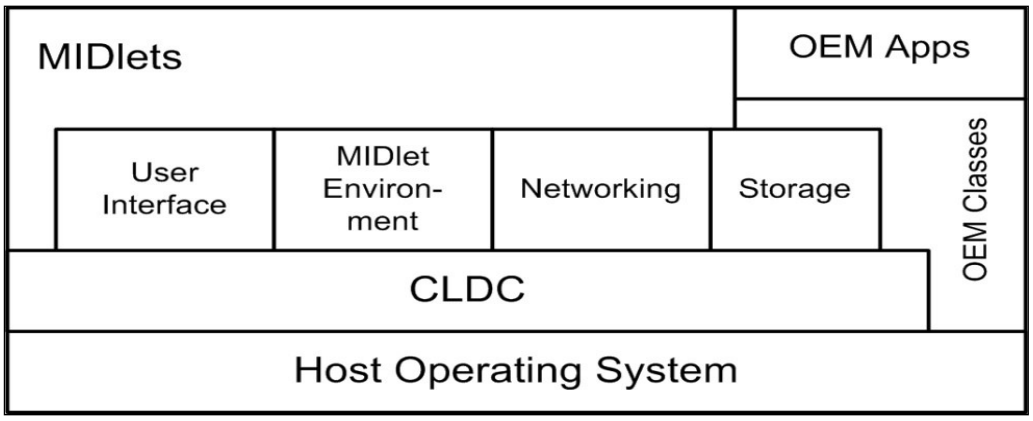

Figure 3-3: Supporting modules in Mobile Information Device Profile (MIDP).

3.2 FARM Implementation

The core idea of FARM is to extract the basic attributes of all the stored files on the device and store them in an XML structure such that each file is associated with its own basic attributes. The same structure can be utilized in the search mechanism where all the XML nodes are navigated for searching a required file. The kXML parser is used to parse the XML data in assembling or searching the meta-data of stored files. There are two main reasons to store the meta-data on the device itself. Firstly, when the search is intended on the device itself, the meta-data will be available for the search module and it will not rely on other storage resources or any connection medium to access its meta-data. Secondly, when the search is intended on other connected devices, each device will execute its search process individually which will not over load the file searching device by processing the meta-data of all devices. In addition, the search will be time efficient because the meta-data of each device will be processed simultaneously by their corresponding search processes.

The logical positions of different components used in FARM are shown in Figure 3-4 to describe the architecture of FARM. For this purpose, several MIDlets have been implemented to support file sharing, search options and Bluetooth connectivity besides file annotation and management.

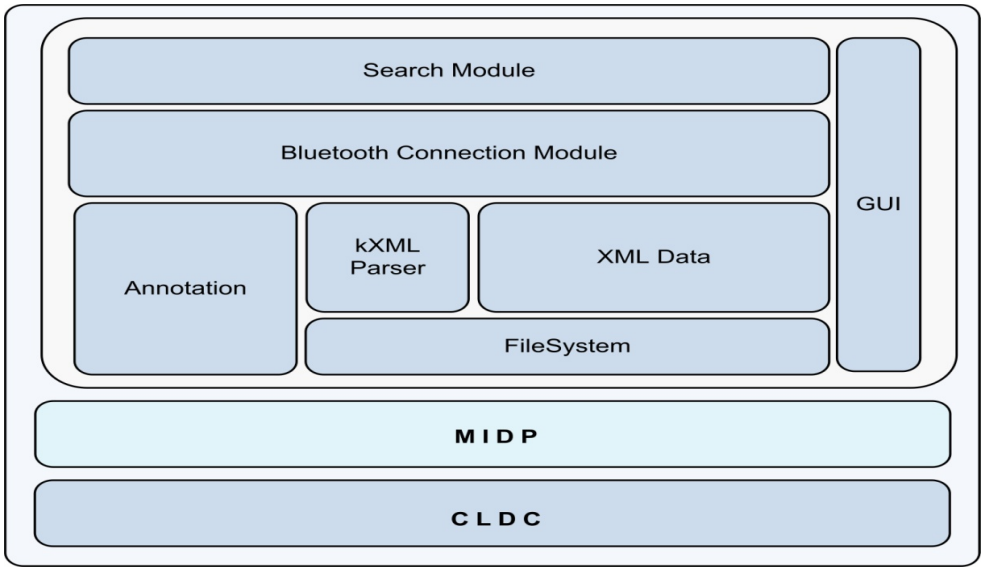

Figure 3-4: The software architecture of FARM.

A snapshot showing the main menu of FARM implementation can be seen in Figure 3-5. The annotation MIDlets automatically annotates the files with corresponding file attributes provided by existing file system on the mobile phone and store the metadata locally. The search module provides functionalities to search for files on the device itself or on other devices if connected through Bluetooth.The Bluetooth features include sending search queries, sending back its response, sharing and transferring files to other connected devices. FARM's use case study and its working details are given in the following sub-sections.

Figure 3-5: A mobile screen showing the FARM main menu.

3.3 Use Case Study of FARM

An application scenario is presented to fully understand the working model and significance of FARM. File names on mobile phones are usually named by applications with default settings, for example images are stored with default names *image001*, *imag002* and so on. After some time users tend to forget the information and content of such files stored on his/her phone or other low end devices. In order to retrieve a file on a mobile device without knowing the actual name of the file, the user has to browse or open all the files unless he/she finds the required file. The case worsens if the user has more than one device to store such information and gets complicated when he/she share files with family

members or colleagues. In this case, every device has to be browsed and searched manually for the required file.

Similar situation is presented in this scenario where a user visits a historical fort in summer vocations with his family and takes a group photo on a mobile phone. Three months later, the user finds himself stuck in a situation where he wants to view that particular picture but forgot that which mobile phone was used and what was the exact file name. On top of that, all of his family members have large number of images and files stored on their phones which makes the retrieval more difficult. FARM provides exact solution by allowing all devices to connect through Bluetooth and use the advanced search options to search on all connected devices. The required file can easily be searched through available options. User can view a list of files and its meta-data or can search on other connected mobiles through Bluetooth using advanced options of FARM.

3.4 Annotation Process

Using JSR-75 [123], this core module of the framework interacts with the underlying operating system of device to haul out vital file attributes by traversing the directories stored on the device. These attributes are assembled and used as annotation tags for each corresponding file, which means that each file is annotated with its own basic attributes which are extracted from file system. Its process diagram shown in Figure 3-6 explains the design of annotation module. All attributes are parsed and stored locally in XML format as shown in Figure 3-7, where a fragment of an XML file is presented to show the stored meta-data of two files namely "*nature.PNG*" and "*classnote.doc*". The meta-data consists of two parts namely, *Automated* and *Optional*. In *Automated* meta-data part, files are annotated automatically with three basic attributes which include file-name, file-size and date-of-creation, while two additional tags can be appended through *Optional* meta-data entered by device users. These two optional tags are namely, *Keyword* and *Description.* Thus, each file can be annotated with 5 attributes of which 3 attributes are collected automatically while other two are left optionally to the device users. Annotation is a one time process but it can be edited or updated for any stored file on the device.

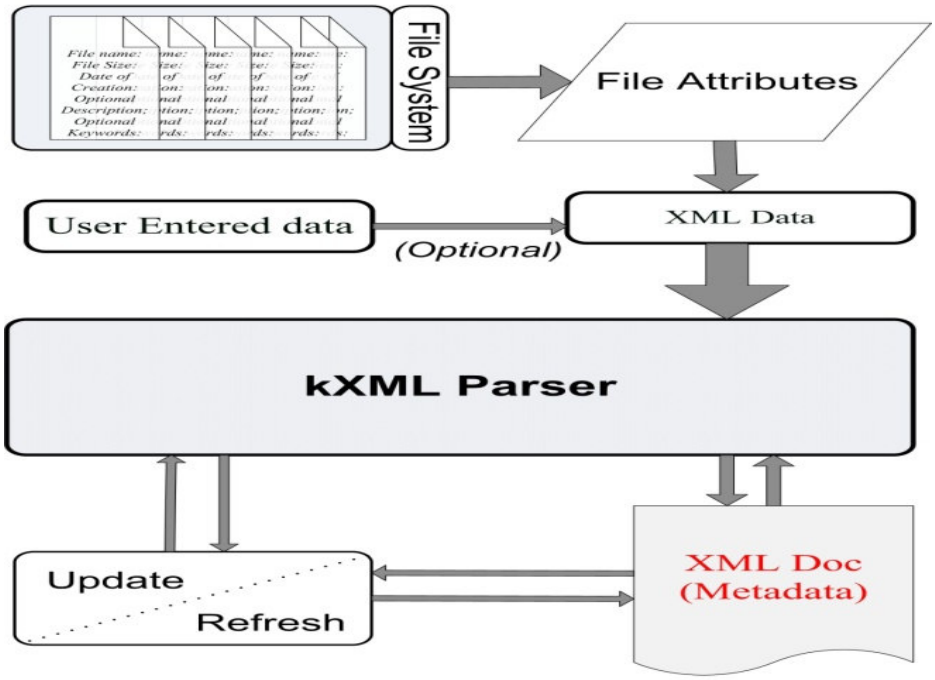

Figure 3-6: Annotation process implemented in FARM framework.

The meta-data is then parsed by kXML parser and stored in an XML structured document so that the parser can later process it for updates and search purposes. Obviously, as the number of files grows on a device the size of the XML file will also increase and the kXML parser will take longer to parse it. However, this parsing time is proved to be negligible which can be depicted from the evaluation results presented in Section 3.7.1.

```
<? xml version="1.0" ?>
- <start>
  :
  :
 - <File>
      <FileName>nature.PNG</FileName>
      <FileSize>360311</FileSize>
      <FileCDate>Sat July 04 22:35:57 GMT 2009</FileCDate>
      <KeyWord>trees, green  ,picture</KeyWord>
      <Description>Visit to a nice place on my 30th Birthday with my family and friends
      </Description>
    </File>
 - <File>
      <FileName>classnote.doc </FileName>
      <FileSize>278</FileSize>
      <FileCDate>Mon Jul 06 11:06:15 GMT 2009 </FileCDate>
      <KeyWord>J2ME, WNCC, notes </KeyWord>
      <Description>Lecture notes</Description>
    </File>
 - <File>
      <FileName>readme.txt </FileName>
      <FileSize>527</FileSize>
      <FileCDate>Mon Jul 06 12:15:35 GMT 2009 </FileCDate>
      <KeyWord>username, password </KeyWord>
      <Description>installation manual </Description>
    </File>
   :
   :
</start>
```

Figure 3-7: A fragment of stored meta-data in XML format.

The implemented MIDlet has additional features to refresh, update or edit the metadata of any file at any time. Figure 3-8 shows a complete list of files on the screen where any file can be selected to update its meta-data. Further more, the metadata of all stored files can be viewed on screen by scrolling up/down the complete list, as shown in Figure 3-9 where the relevant information of a file namely "*nature.png*", can be seen. Besides the automatic annotation, the user-entered keywords can also be stored for much efficient and customized search options. The editing of meta-data for an individual file can be seen in Figure 3-10 where the file "*nature.png*" is selected for updating its meta-data.

Figure 3-8: File selection for annotation.

Figure 3-9: Metadata of "nature.png".

Figure 3-10: Editing of metadata.

3.5 File Retrieval in FARM

The users can search for a specific file through any one or more options or stored keywords. Figure 3-11 shows the snapshot of mobile screen showing different search option provided by the framework. Search module plays a vital role in the proposed framework,

which interacts with nearly all other modules included in FARM. This module is responsible for searching the required information by parsing the whole meta-data using kXML parser. As mentioned in the previous section, that files are annotated with three automatic and two optional tags; search can be performed with any option using available attributes as shown in Figure 3-11. However, the search through using two *optional tags* can only be successful if the required file is annotated with the *optional tags*.

Figure 3-11: Mobile screen showing file-search options in FARM.

If search is intended in a PAN environment, search module uses Bluetooth connection to send search queries to other connected devices and receives back the query results as shown in Figure 3-12. The user first selects a search option according to the attributes of a file for example file name, date of creation, file size, or user entered keyword. A search query is then sent to all connected devices. All query-receiving devices search for the required information in their corresponding local storage and the result is sent back to the query sending device. Individually, all the query-receiving devices use the same approach for searching a required file, which is parsing the XML file using kXML. The Bluetooth module also provides some additional features, which are briefly described in the following sub-section.

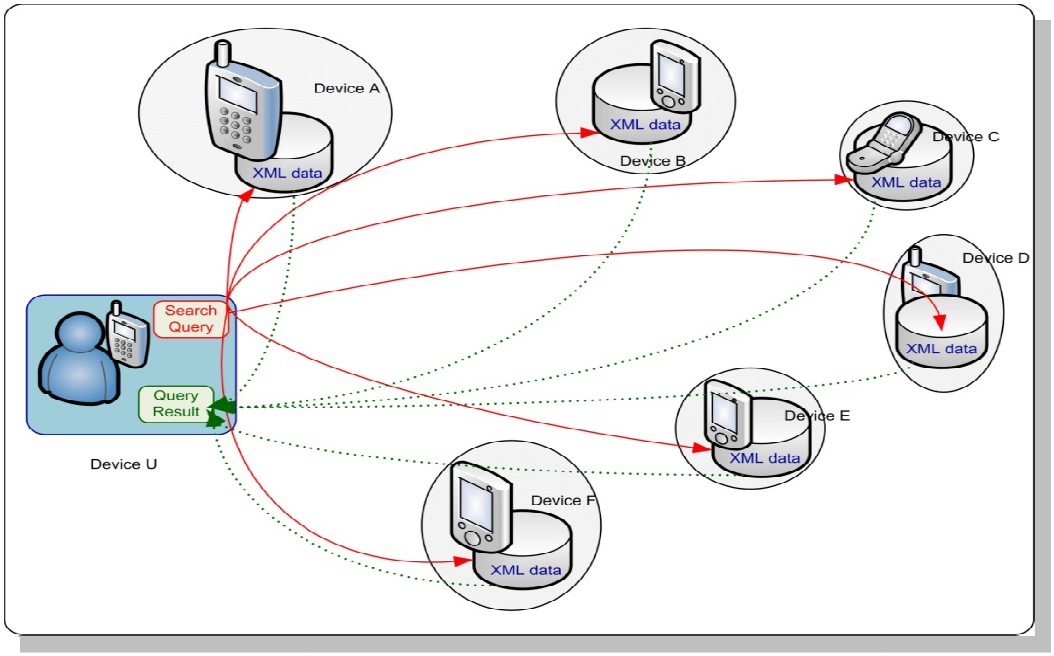

Figure 3-12: File search process in networked envieronment.

3.6 Bluetooth Module

Bluetooth is a short range, low power and low cost radio communication and it is available in the majority of available handheld devices. In FARM framework, this module is used to send the search queries to other connected devices and receive the query results back as shown in Figure 3-12. This module also provides functionalities for sharing, un-sharing and transferring files between devices. The FARM framework fully supports Bluetooth connectivity by implementing Bluetooth protocol stack, which is standardized in JSR-82 [124]. It gives good control for stack initialization, device management, device discovery, service discovery, and communication. The Service Discovery Protocol (SDP) is used to discover the nearby devices and files shared by other users in the network. Figure 3-13 shows a mobile screen with the list of files, which can be selected for sharing with the requesting device.

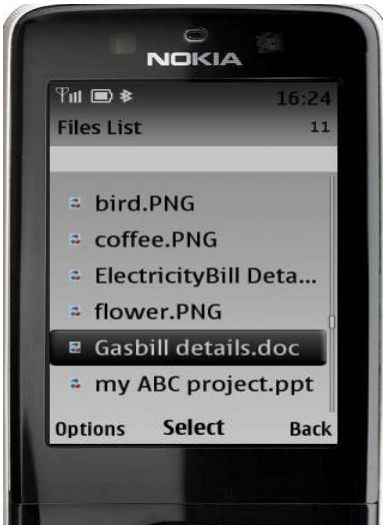

Figure 3-13: File sharing in FARM.

3.7 Performance Evaluation

The performance of FARM is evaluated mainly from the aspects of efficiency and accuracy in file retrieval. For this purpose, a verity of tests and experiment were conducted using FARM implementation to demonstrate the practicality of the proposed approach.

3.7.1 Evaluating the performance of kXML in FARM

The FARM's performance mainly depends on two parameters - time and efficiency in file search. As discussed earlier annotation is a one-time process but file search may be used repeatedly. The search module parse the XML data to get the information about a required file which is the only time consuming task in the framework. To evaluate the performance of kXML in FARM in terms of time consumption, 4 different tests were carried out, each test was performed five times and their mean values were calculated which are presented in Table 3-1. These results indicate that one kilobyte of XML data is parsed in approx. 3.03 milliseconds. In other words, metadata that is required for a single file is parsed in approx. 0.59 milliseconds. These facts show that kXML will not degrade the performance of FARM by taking too long to parse the meta-data. For instance, if a device has stored 1000 files, the

size of XML file will be 197.1 KB to store the meta-data for 1000 files and kXML will take about 578 milliseconds to parse the whole XML file using FARM.

Table 3-1: Performance of kXML parser in FARM framework

XML file Size (KB)	XML Data for number of files	Time taken (in ms)
18.8	100	71
38.1	200	125
97.2	500	291
197.1	1000	578

3.7.2 Probabilistic Evaluation of FARM

To evaluate the efficiency of the search in FARM, a number of decisive parameters are discussed and analyzed. In order to compare and validate the performance of FARM in file retrieval, another MIDlet was implemented with the capability to search for files by sending a query based on file names only, without any annotations. The second MIDlet (which will be referred as untagged for the rest of this thesis), does not annotate any file and hence cannot conduct searches based on annotations. It is expected that the file search based on annotations would be more efficient than searching on file names only. The framework efficiency can also be validated through probabilistic approach, which gives a general idea from the user's perspective about the success probability of searching a required file.

To compare the probability of success for searches in both cases, successive trials were carried out for each and the probability of success is computed. If p is the probability of success for a system, then $q = 1 - p$ can be used to indicate the probability of failure. Trial results obtained from FARM are shown in Table 3-2 using different search options. Number of file based search trials were kept high i.e. 60, to give a fair chance in comparison with untagged system because it supports only file-name based search. The failure of search queries also includes the errors made by users in typing or selecting mismatched options. For example, a user intended to search a file based on its size but instead of selecting the option file-size, he/she selected date of creation.

Table 3-2: Search Query Results.

No of queries	Search Query	Success
60	File Name	52
10	File Size	9
10	Date of creation	6
10	Keywords	8
10	Description	7

Since the result of the search can have two possible outcomes i.e. success or failure, therefore, based on the values of *p* and *q*, a generalized model to compute and quantify the efficiency of search systems can be presented based on the Binomial Distribution *b (x; n, p)* as defined by equation (3.1).

$$b\ (x;n,p) = \binom{n}{x} p^{x} q^{n-x} \tag{3.1}$$

The Binomial Distribution can give a good approximation for the probability of *x* successes for each system as the number of trials *n* increases. Therefore, if x_u, n_u and p_u are the number of successes, number of trials and the success probability respectively. For an untagged search system, b_u (x_u; n_u, p_u) is the Binomial distributed variable for the untagged search system. Similarly, x_f, n_f and p_f are the parameters used for FARM with b_f (x_f; n_f, p_f) being the Binomial distributed variable. In order to calculate a value of *p* for FARM and untagged search system, *n* number of trials is carried out for each system. A comparison of the probability of getting *x* successes for a number of trials for both systems is shown in Figure 3-14.

Another approach in comparing the two systems can be the number of searches a user has to make in order to get to the desired file. Therefore, taking *p* and *q* as the probability of success and failure for *n* independent trials, the distribution for the number of trials until the first success occurs is defined by equation (3.2) as given,

$$g(x;p) = pq^{x-1} \forall\ x = 1, 2, 3 \ldots n \tag{3.2}$$

Where g is a Geometric distributed variable. If p_u and q_u are the success and failure probabilities for the untagged system, then g_u is the distribution for the probability of x_u^{th} trial being the first successful search for an untagged system. Similarly, the notations p_f, q_f, x_f and g_f are used for FARM search system.

Table 3-3: Comparison of values calculated Binomial distribution for FARM and Untagged System.

	n=100		n=200		n=500		n=1000	
X	90	95	180	190	450	475	900	950
$b_f(x_f;n_f,p_f)$	0.07690	0.00377	0.03184	5.63E-05	0.00400	3.29E-10	0.00019	9.40E-19
$b_u(x_u;n_u,p_u)$	3.09E-16	2.01E-21	5.16E-31	1.60E-41	4.24E-75	1.40E-01	2.10E-48	1.74E-210

In Table-3-3, a comparison between FARM and the untagged system is given based on the Binomial Distribution. The number of trials was varied from 100 to 1000 and the probability for x success is calculated for FARM and untagged system when the value of p_f is 0.82 and p_u is 0.52. When the number of successes x is varied keeping the number of trials n constant i.e. n = 100, the comparison of b_f and b_u is given in Table 3-4. It can be noted that the value of b_f increases as the number of successes increases up to 90 while the value of b_u decreases. The reason is based on the value of success probability p which is higher for FARM and relatively lower for untagged system.

Table 3-4: Comparison based on Binomial distribution calculated for X values (30 to 90).

X	30	40	50	60	70	80	90
$P_f = 0.82$	1.01E-20	1.01E-20	2.87E-13	1.50E-07	0.08703	0.08703	0.01082
$P_u = 0.52$	4.31E-06	0.00449	0.07346	0.02228	0.00010	4.30E-09	3.09E-16

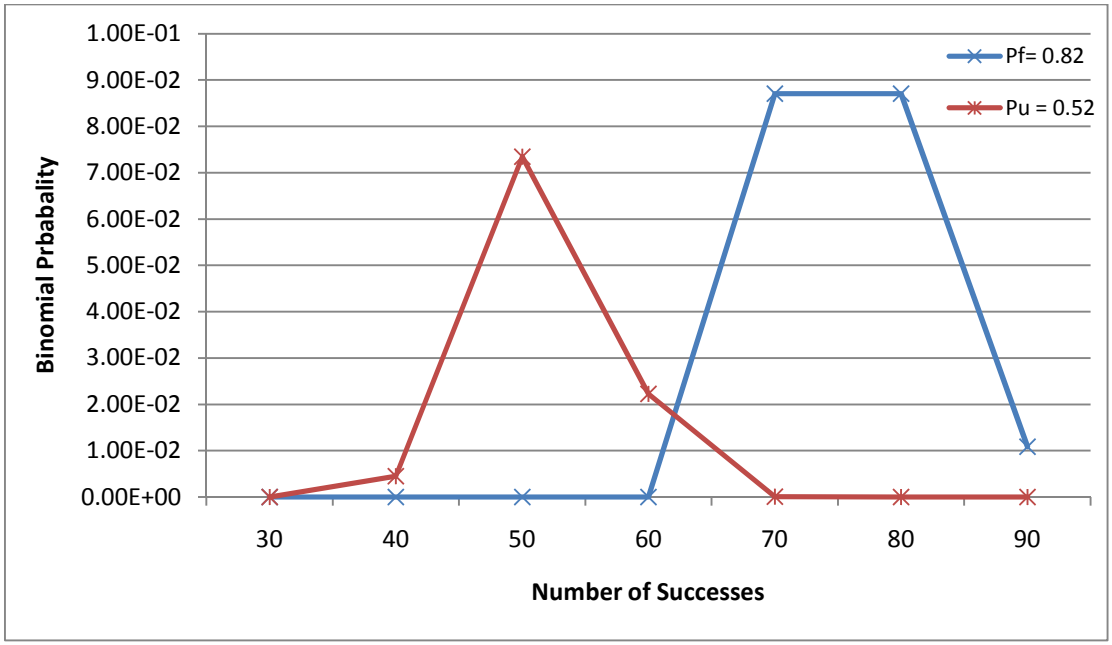

Figure 3-14: Comparison of success probabilities calculated for FARM and Untagged frameworks.

In Figure 3-15, a comparison between the two approaches can be seen based on geometric distribution calculated in Table 3-5. The graph is obtained by varying the number of trials in order to get the probability of first search success.

Table 3-5: Geometric distribution calculated of FARM and Untagged System.

x	1	2	3	4	5
$g_u(x_u; p_u)$	0.52	0.2496	0.1198	0.0575	0.0276
$g_f(x_f; p_f)$	0.82	0.1476	0.0265	0.0047	0.0008

It is evident from the result that the probability of the FARM approach is higher when the number of trials is less i.e. the coverage area under the curve is greater during the first three trials while for the untagged approach it is spread out till trial number 5. This clearly indicates that the probability of success for FARM is higher for lesser number of trials. In other words, the chances for getting a successful query with a small number of trials are high for FARM compared with the untagged approach.

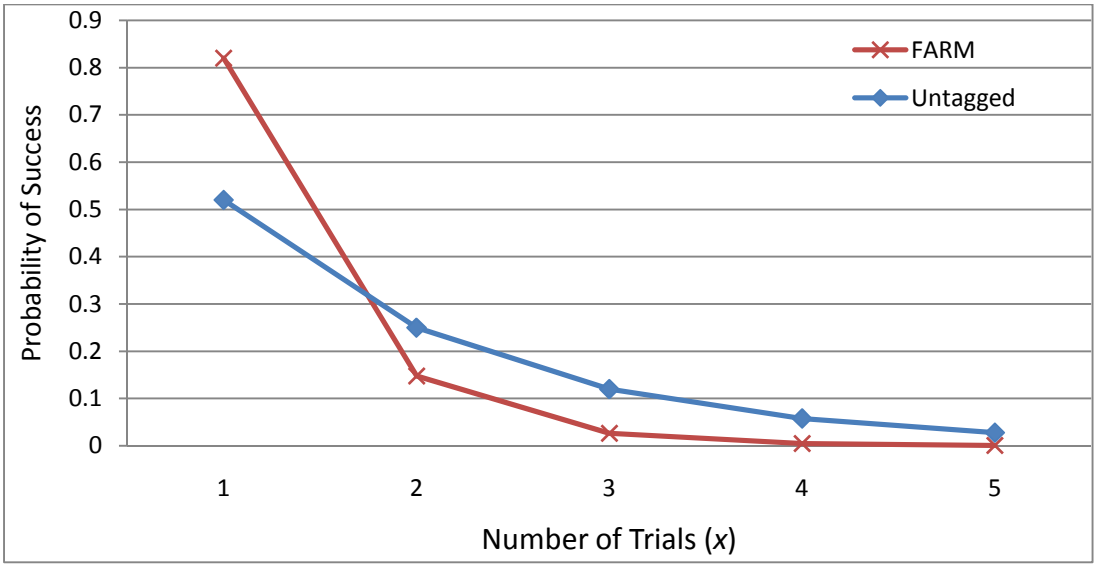

Figure 3-15: Comparison of success probabilities for trials (1 to 5).

Any given geometric distribution for FARM and untagged systems depend on the value of success probability *p.* The value therefore plays an important role in giving us a general intuition about the performance of FARM and untagged systems. We use the maximum likelihood estimator of *p* for Geometric and Binomial distributions and elucidate upon the results presented in Table 3-5.

3.7.2.1 **Parameter Estimation for Geometric Distribution**

For a random sample x_1, x_2, $x_3 \ldots x_n$ from a geometric distribution, the likelihood function is given by;

$$L(p) = (1-p)^{x_1-1}p(1-p)^{x_2-1}$$

$$= p^n(1-p)^{\sum x_i - n}, (0 \le p \le 1)$$

Taking the natural logarithm $L(\theta)L(\theta)$

$$= \ln L(p) = n\ln p + \left(\sum_{i=1}^{n} x_i - n\right)\ln(1-p), (0 \le p \le 1)$$

After taking the derivative with respect to p

$$\frac{d \ln L(p)}{dp} = \frac{n}{p} = \frac{\sum_{i=1}^{n} x_i - n}{1-p} = 0$$

After solving for p, we get

$$p = \frac{n}{\sum_{i=1}^{n} x_i} = \frac{1}{x}$$

The maximum likelihood estimator of p is

$$\hat{p} = \frac{n}{\sum_{i=1}^{n} x_i}$$

$$\hat{p} = \frac{1}{x}$$

$$x = \frac{1}{\hat{p}} \tag{3.3}$$

Since *x* is the number of trials in which the first success is expected to happen, therefore, a smaller value of *x* suggests that the first success is expected in smaller number of trials when compared with a case when the value of *x* is larger. As an example, if the value of $\hat{p}$ for the FARM is 0.82 while for untagged system it is 0.52, then the number of trial on which the first success is expected i.e. *x* = 1.21 while for the untagged system, the value of *x* is 1.92. It can be clearly deduced that the higher value of $\hat{p}$ in case of FARM system has a higher probability of getting a successful search sooner than the untagged system.

3.7.2.2 **Parameter Estimation for Binomial Distribution**

In order to evaluate the parameter estimation for binomial distribution of FARM and untagged system, the maximum likelihood function is

$$f(x) = \frac{n!}{x!(n-x)!} p^{x} (1-p)^{n-x}$$

the function can be written as

$$L(p)=\prod_{i=1}^{n} f(x_i)=\prod_{i=1}^{n}\left(\frac{n!}{x_i!(n-x_i)!}p^{x_i}(1-p)^{n-x_i}\right)=\left(\prod_{i=1}^{n}\frac{n!}{x_i!(n-x_i)!}\right)p^{x_i}(1-p)^{n-\sum_{i=1}^{n}x_i}$$

taking log of both sides to get the log likelihood function

$$\ln L(p)=\sum_{i=1}^{n}x_i\ln p+\left(n-\sum_{i=1}^{n}x_i\right)\ln(1-p)$$

$$\frac{d\ln L(p)}{dp}=\frac{1}{p}\sum_{i=1}^{n}x_i-\left(n-\sum_{i=1}^{n}x_i\right)\frac{1}{1-p}=0$$

$$\frac{\left(1-\hat{p}\right)\sum_{i=1}^{n}x_i-\left(n-\sum_{i=1}^{n}x_i\right)\hat{p}}{\hat{p}\,(1-\hat{p})}=0$$

$$\sum_{i=1}^{n}x_i-\hat{p}\sum_{i=1}^{n}x_i-np+\sum_{i=1}^{n}x_i\hat{p}=0$$

$$\hat{p}=\frac{\sum_{i=1}^{n}x_i}{n}$$

If *k* is the number of successes in *n* trials, then $\hat{p}$ = *k/n* and therefore

$$k = n\,\hat{p} \tag{3.4}$$

Equation (3.4) simply indicates that if the number of trials *n* is kept constant, the increase in the value of $\hat{p}$ can also lead to an increase in the value of *k*. Generally, if k_1 and p_1 are the number of successes and success probabilities for FARM while k_2 and p_2 are the number of successes and success probabilities for an untagged system then if $p_1 > p_2$, then $k_1 > k_2$. Since in the previous section $p_1 = 0.82$ while p_2 = 0.52, therefore, the number of successes for FARM is greater than that of the untagged system for any number of trials as long as the condition $p_1 > p_2$ stands.

3.7.3 Calculating Precision and Recall

Precision and Recall have been widely used in information retrieval to evaluate the accuracy of a search mechanism [125], [126], [127]. Its use in the context of document

retrieval, C. J. Van Rijsbergen [125] followed Cleverdon [128] and defined the terms *Precision* and *Recall* as follows:

- The Recall of the system, that is, the proportion of relevant material actually retrieved in answer to a search request.
- The Precision of the system, that is, the proportion of retrieved material that is actually relevant.

Two sets of filenames namely, *target-set* and *retrieved-set,* were selected randomly from a set of 500 filenames. The target-set consisted of 10 filenames obtained to declare the relevancy of retrieved files while the number of files in the retrieved-set was kept different from 1 to 10. The relevancy is then checked by comparing both sets of files. At least one file was kept relevant for all groups of tests to make sure the number of retrieved files will not be zero. This test-process was implemented through a Java program and average was taken for the 1000 tests. The same process was used for FARM but the comparison of both sets was extended to three times i.e. after the first comparison of both sets, another set of retrieve-set is picked up and compared with the same target-set and then a third set is picked up and compared. As discussed in Section 3.4, the annotation process annotates each file with five tags in which three of them are being annotated automatically. For this reason, the target-set was compared thrice with the retrieve-set as it has at least three times more chances to retrieve a relevant file.

Let $\mathrm{Re}\,trel$ be the number of relevant files retrieved, $\mathrm{Re}\,l$ be the total number of relevant files and $\mathrm{Re}\,t$ be the number of retrieved files, then recall (Rc) and precision ($PREc$) can be calculated as defined by equation (3.5) and (3.6):

$$Rc = \frac{\left|\mathrm{Re}\,trel\right|}{\left|\mathrm{Re}\,l\right|} \tag{3.5}$$

$$PREc = \frac{\left|\mathrm{Re}\,trel\right|}{\left|\mathrm{Re}\,t\right|} \tag{3.6}$$

Figure 3-16 shows that precision is 100% when recall is 10% for both FARM and untagged system. However, FARM performs better than the untagged system as the recall goes higher. When the recall is 100%, the precision of FARM is 40% while the precision of the untagged system is just above 20%.

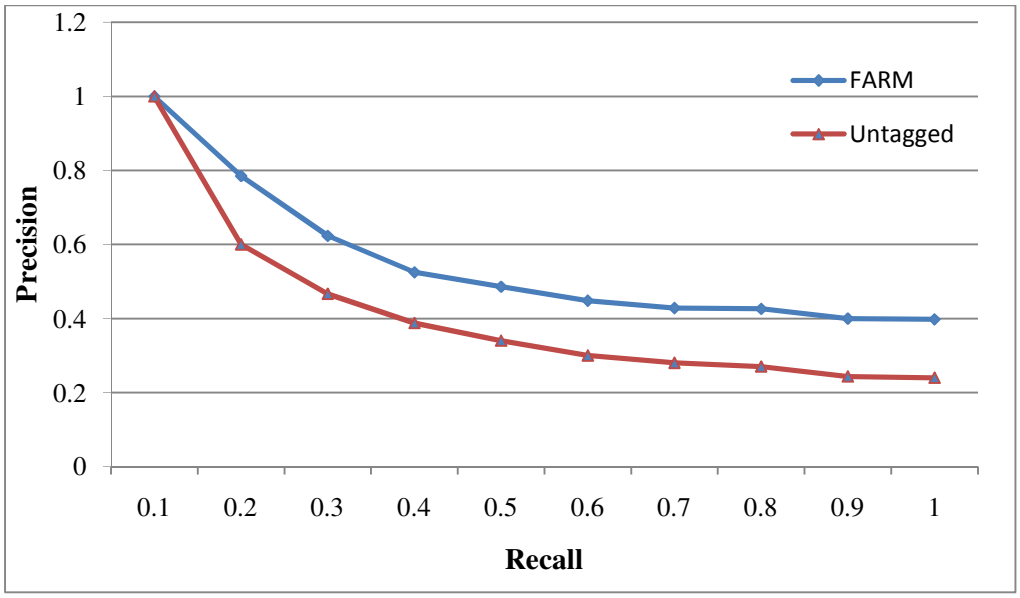

Figure 3-16: Precision and Recall for FARM and Untagged system

3.7.4 Evaluation of Automated and Optional Metadata

To evaluate the significance of automated and optional meta-data in FARM, two sets of tests were formulated, each comprising of 25 search queries. For the first set, all files were automatically annotated and the optional tags were also added, however for the second test, files were only annotated automatically and additional tags were left empty. Results shown in Table-3-6 indicate that success rate is 84% for the set which was fully annotated and 72% for the second set.

The significance of user-typed additional tags can be measured by the difference between two sets of results, which is 12% in this case. Using (1), the binomial distribution is calculated to approximate the probability of *x* successes as the number of trial *n* increases for automated annotation set and optional annotated sets of test. The successes probability

can be denoted by P for the first set and P_a for the second set (which is automated annotation only).

Table 3-6: File search query types and results

No of queries	Query types	Success
25	File Name, File size, Date of creation, keywords and description.	21
25	File Name, File size and Date of creation	18

When the number of successes x is varied keeping the number of trails n constant i.e. n = 100, the comparison of P and P_a is given in Table 3-7. A probability comparison of getting x successes for a number of trials for both sets of test is shown in Figure 3-17.

Table 3-7: Comparison of values calculated by using geometric distribution.

x	50	60	70	80	90
P=0.84	2.65E-15	5.75E-09	1.95E-04	5.67E-02	2.92E-02
Pa=0.72	1.69E-06	0.0029137	0.07869	0.01815	7.41E-06

The graph shows that the probability of getting a successful result is higher for P which is 70% and above as compared to P_a. It is more likely to get a successful result if files are annotated with the optional tags along with *automated* meta-data tags. By using maximum likelihood estimator we can calculate the *first* expected success in both cases and can compare the results. In order to calculate the parameter estimation for geometric distribution of P and P_a can be computed using equation-3. Where x gives the number of trials in which the first successes occurs.

Calculating for both sets, the value of p for the *automated* annotation and *optional* annotation is 0.84 and 0.72 respectively, and the value of x is 1.190 and 1.388. It can clearly be concluded that if files are tagged with automated meta-data only, user will need 1.388 trials to get the required file and 1.19 trials will be required if files are tagged with optional tags along with automated metadata.

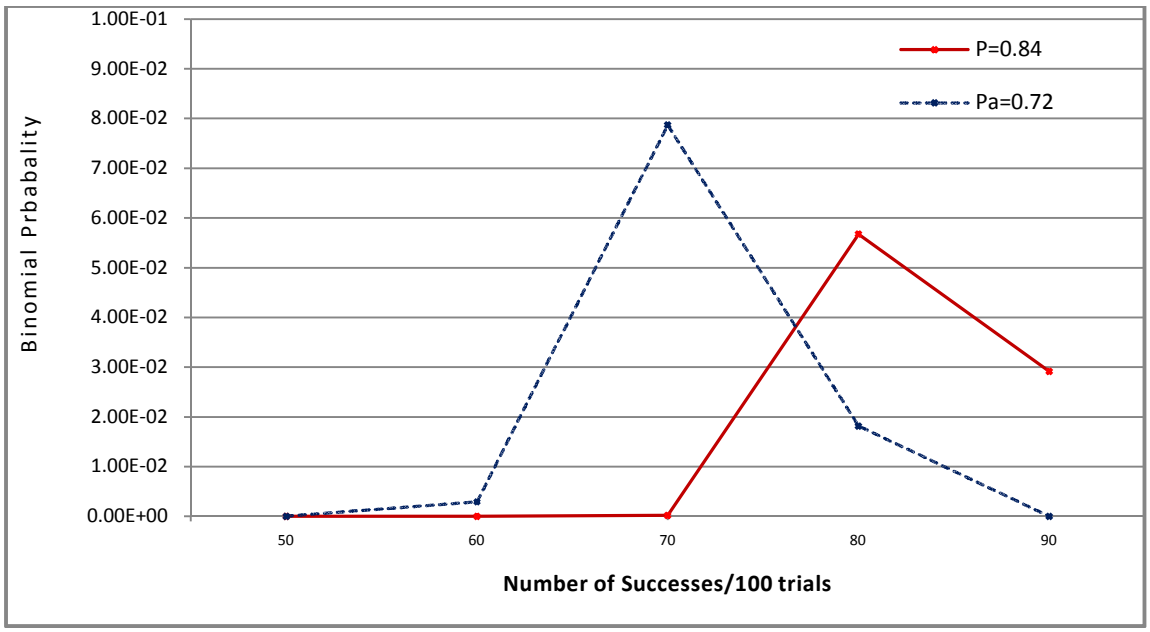

Figure 3-17: Comparison of successes per 100 trials.

3.8 Summary

This chapter presented FARM framework, which automatically annotate files on low-end devices, and provides an efficient mechanism to search a required file. It briefly described the FARM software architecture specifically focusing on the exploitation of XML structure, which is used to store the meta-data and retrieve the required information associated with a file in search. The annotation and search processes are highlighted along with some additional features provided by FARM. The performance of the FARM has been evaluated with various aspects and detailed analyses were presented.

CHAPTER 4

Semantic-based Retrieval of Files on Low-end Devices

This chapter presents a framework namely SemFARM [129], which further enhances the search capabilities of FARM, a framework presented in Chapter-3 to annotate and retrieve the files on low-end devices. The SemFARM framework is built on semantic web technologies in support of file retrieval on low-end mobile devices. A generic ontology is developed which defines a number of keywords, their possible domains and properties. Based on semantic reasoning, similarity degrees are computed to match the user queries with published file descriptions. The SemFARM prototype is implemented using the Java mobile platform (JME). The performance of SemFARM is evaluated from a number of aspects in comparison with traditional mobile file retrieval systems and enhanced alternatives. Experimental results are encouraging showing the effectiveness of SemFARM in file retrieval and demonstrate that the use of semantic web technologies have facilitated file retrieval in mobile computing environments by maximizing user satisfaction in searching for files of interest.

4.1 Semantic-based file retrieval framework

SemFARM uses the same annotation process, which is proposed and implemented for the FARM framework as explained in Chapter-3, Section 3.4. Similarly, all other modules of FARM are fully compatible with SemFARM. However, the search mechanism of SemFARM is different as it is extended to employ the semantic technology for improving the file search capabilities. For this purpose, a generic ontology is developed in OWL to define the meanings of most widely used keywords which can possibly be used to annotate a stored file on a device. The ontology itself is scalable and can be further expanded, if needed. The meta-data stored in XML format by annotation process, are automatically converted to RDF model. Alternatively, the meta-data of files can be directly stored in RDF model but to ensure the compatibility with FARM framework, the SemFARM uses an XML to RDF converter. For example, the file named "*image.08*" is annotated with two tags "*building"* and "*Brunel University*" as its keyword and description respectively, will be converted in RDF model accordingly as shown in Figure 4-1 where the text in <> tags represent the XML arrangement for the same file.

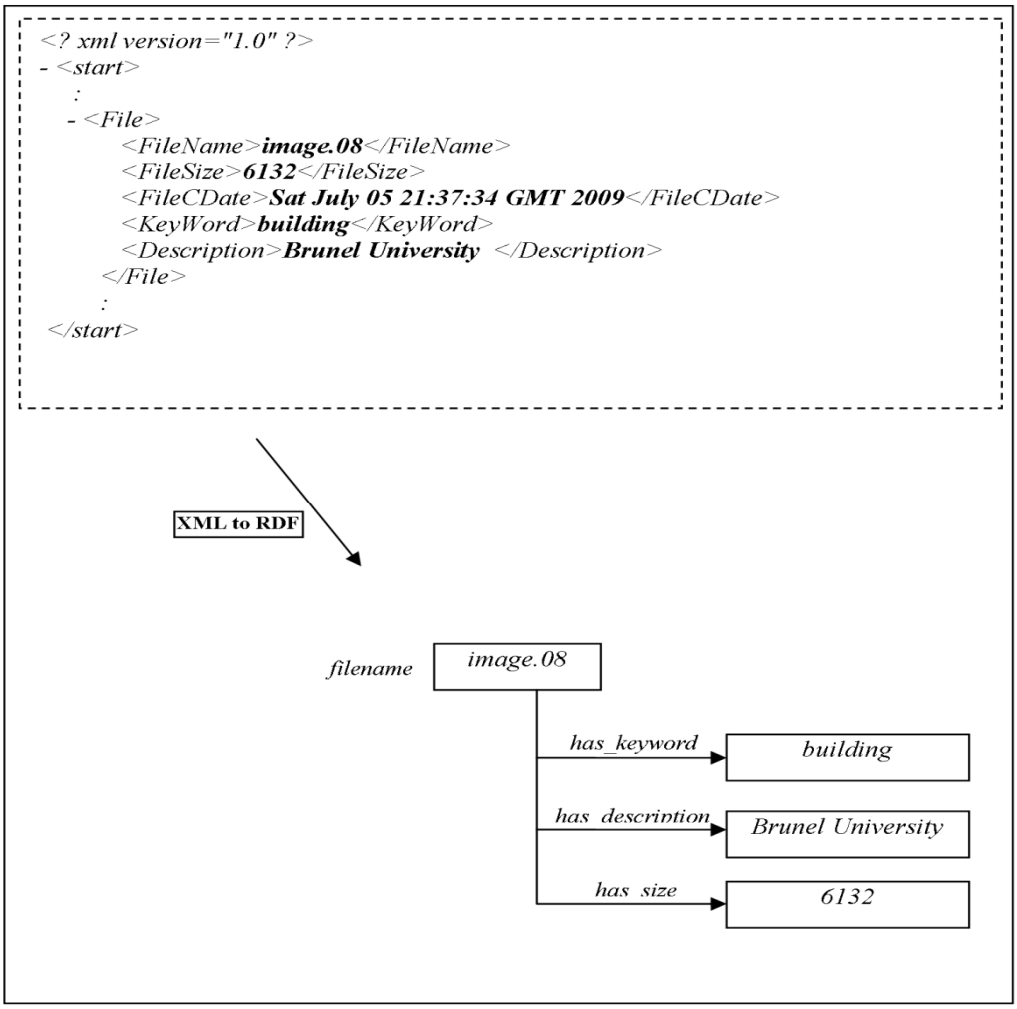

Figure 4-1: XML to RDF conversion of the metadata associated with "image.08".

4.2 SemFARM Search Module

To perform a file search, the XML data created in the annotation process is sent to the search agent where the file is parsed to create an RDF document dynamically. In the creation of RDF model process, the XML structure is strategically dealt and exploited in such a way that tags associated with a file, are used as relationships in the RDF schema for a particular file. The RDF schema is then passed to the Jena Inference engine along with the OWL ontology definitions to derive additional statements as depicted in Figure 4-2.

All statements and resources are searched for the required information about a file in query. The search is performed by navigating the inference model using the Jena APIs for a

specific property associated with a resource. The same process is repeated for all connected devices on which the search is intended. SemFARM supports Bluetooth connectivity to share and transfer files between connected and authorized devices. The reasoning task is performed on a network server as low-end devices are unlikely to perform reasoning in rational time because of their limited computing resources. Therefore, an XML file containing the meta-data of files is sent to the server where the reasoning is performed by binding it with ontology definitions. Currently, the SemFARM search module employs single ontology for extracting additional information about query keywords however; the search module can be extended to take advantage of multiple ontologies.

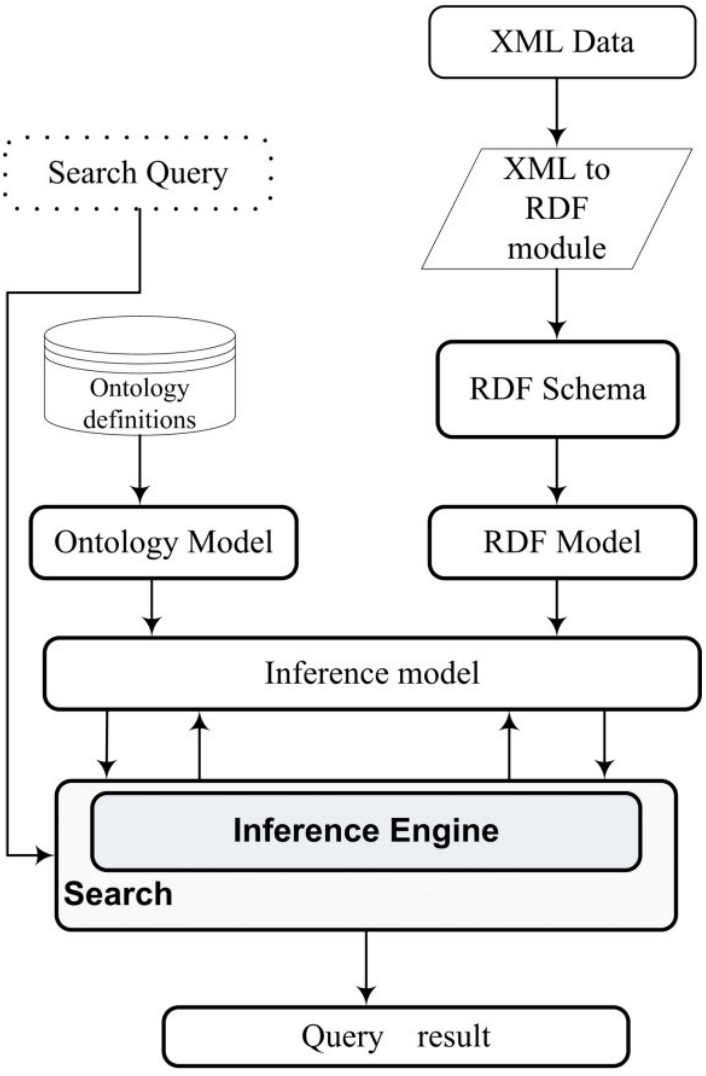

Figure 4-2: File search process in SemFARM framework.

For this purpose, various approaches have been proposed in recent years for example, M. Bhatt et al. [130] proposed and demonstrated the prospect of sub ontology extraction from a base-ontology. Similarly, A. Flahive et al. [131] demonstrated the sub-ontology extraction and extending it with new features through Service Oriented Architecture (SOA) and proposed a distributed framework [132] for ontology tailoring.

4.3 Matching Degree in SemFARM

The matching degree can be computed between properties assigned to files with the properties used in file retrieval queries based on ontology definitions. Let

- p_Q is a property used in a file retrieval query
- p_A is a property associated with a file.

The following relationships between p_Q and p_A are based on the work proposed by Paolucci et al. [133].

- *Exact match:* p_Q and p_A are equivalent, or p_Q is a subclass of p_A.
- *Plug-in match:* p_A subsumes p_Q.
- *Subsume match:* p_Q subsumes p_A.
- *Nomatch:* There is no subsumption between p_Q and p_A.

Li *et al*. [134] further defined match degrees by considering the semantic distance between properties in an advertisement and query, which they used for service discovery to quantify the relationships. Similarly, we can also quantify the match degrees between a property associated with a file and the properties used in a file retrieval query. For this purpose, a numerical degree is assigned for each match to quantify the relationship between p_Q and p_A. To consider the semantic distance between p_Q and p_A in assigning a match degree,

Let

- $dom(p_Q, p_A)$ be the degree of a math between p_Q and p_A and
- $||P_Q, P_A||$ be the semantic distance between p_Q and p_A in terms of domain ontology Ω.

Following the proposed work in [134], $dom(p_Q, p_A)$ is defined for a match degree calculation as follows:

$$dom(P_Q,P_A) = \begin{cases} 1 & exact \text{ match}, \\ \frac{1}{2} + \frac{1}{e^{(||PQ,PA||-1)}} & plugin \text{ match}, \left|\left|PQ, PA\right|\right| \geq 2, \\ \frac{1}{2 \times e^{(||PQ,PA||-1)}} & subsume \text{ match}, \left|\left|PQ, PA\right|\right| \geq 1, \\ 0.5 & uncertain \text{ match}, \\ 0 & nomatch. \end{cases} \quad (4.1)$$

According to equation (4.1), for a plug-in match between p_Q and p_A, $dom(p_Q, p_A) \in (0.5, 1)$. For a subsume match between p_Q and p_A, $dom(p_Q, p_A) \in (0, 0.5)$.

4.4 Use Case Study of SemFARM

Files are usually stored on mobile devices with the application default settings which are not descriptive enough to be used for file retrieval. The case worsens if users have larger storage capacities on their devices, which is very likely. A similar scenario is presented where a mobile phone user, assuming his name is *Michael,* took a few snapshots of family members using his mobile phone's built-in camera on a birthday party for his niece. On the same occasion, his wife and son also took snapshots using their own mobile phones.

Four months later, Michael wanted to view one of the group pictures taken at the party but he forgot the file name, as the pictures were stored with application default name settings. Michael had to browse and view all the stored pictures on his mobile phone with 16GB of memory making his job more tedious, particularly on a limited keypad and screen. SemFARM can facilitate users in similar situations by providing various search options. It is expected that a user can even forget keywords associated with a file; but still the semantic

support enables the framework to successfully accomplish the retrieval. User can utilize the search options of SemFARM by simply entering the date, the keyword associated with that particular file or any similar keyword for example *birthday*, *birthdayparty*, *party*, *niece* etc. The SemFARM will first search those files which have exact matches and then the ontology will be used to find similar keywords or meaning of the keyword which the user has entered to find the required file. If Micheal's phone is connected with mobile phones of his wife and son using Bluetooth, SemFARM will automatically perform the same search operation simultaneously on all mobile phones provided his son or wife have authenticated the connection and operation.

4.5 Performance Evaluation

Various tests and comparisons are outlined in the following sub-sections to measure the efficiency of SemFARM in terms of file retrieval and search accuracy. Generally, it is expected that file retrieval will be more convenient in terms of effort and time, using a keyword based search compared to browsing all the directories manually.

4.5.1 Computing Matching Degree

The match degrees used for relationships between properties p_Q and p_A to retrieve a file in SemFARM can be *exact*, *plug-in* or *subsume* which are described in Section 4.3. Figure 4-3 shows the ontology definitions used in this case study describing the classifications of health, entertainment, academic, event, health properties and personal properties fragments. Each file on a device is annotated with two keywords. To evaluate the match degree we performed two groups of tests; namely *set-1* and *set2.* In the first group of tests all queries were relevant to health related files to find the matching degree in one fragment, while in the second group of tests, queries were related to the personal related fragment. The queries selected for the first group includes the keywords "*treatment", "gp", "hospital", " tablets"* and *"health".* It can be seen in Figure 4-3 that properties having an exact match to these queries includes *c3, c4, c2, c5* and *c1.* The match

degree can be calculated between the associated keywords and query keywords using equation (4.1) described in Section 4.3 and the keyword definitions given in Figure 4-3. Table 4-1 shows the matching degree calculated for four returned files as an example.

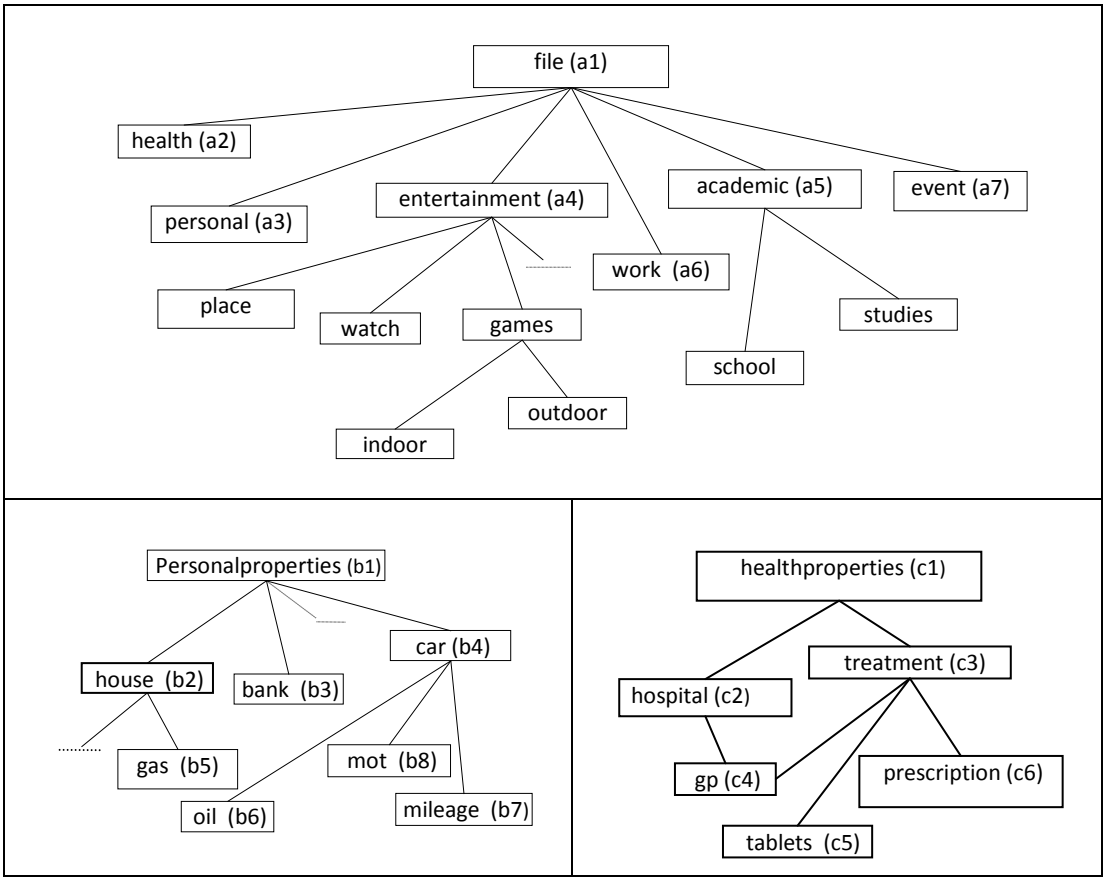

Figure 4-3: Ontologies used in an example for computing match degrees.

Similarly, in the second group of tests, the selected keywords as a query were, *"mileage", "car", "bank", "house"* and *"personal"*. Figure 4-3 shows the matching properties which include *b4, b7, b3, b2* and *b1*. The matching degree calculated matching degree in this fragment of personal related keywords could be seen in Table 4-2.

Table 4-1: Match degrees calculations for test set-1.

Properties / Files names	c3	c4	c2	c5	c1
file10	87% (P)	18%(S)	50%(S)	100%(E)	50%(S)
file11	50%(S)	87% (P)	100%(E)	18%(S)	87% (P)
file12	87% (P)	100%(E)	50%(S)	18%(S)	50%(S)
file13	100%(E)	87% (P)	18%(S)	50%(S)	87% (P)

Table 4-2: Match degrees calculations for test set-2.

Properties / Files names	b4	b7	b3	b2	b1
file1	100%(E)	87% (P)	18%(S)	18%(S)	50%(S)
file2	50%(S)	100% (E)	6%(S)	6%(S)	50%(S)
file3	18%(S)	18%(S)	100% (E)	50%(S)	87% (P)
file4	18%(S)	6%(S)	18%(S)	100% (E)	87% (P)
file5	87% (P)	64%(S)	87% (P)	87% (P)	100% (E)

4.5.2 Calculating Precision and Recall

Precision and recall are widely used in information retrieval to evaluate the accuracy of a search mechanism [125], [126] and [127]. To evaluate the precision and recall in SemFARM, 15 files were randomly selected and annotated with relevant keywords which were not necessarily defined by our ontology. After executing a search query, the list of returned files was checked for relevant files and the number of relevant files was noted. The process was repeated, varying dissimilar search queries to ensure a different number of returned files for calculating the recall value from 0.1 to 1. The same test was repeated for 10, 12 and 15 randomly selected files and the final precision and recall for SemFARM was calculated by the mean values of all three tests. The results obtained from three sets are presented in Table 4-3 and the mean values are plotted in Figure 4-4 for comparison.

The precision and recall for SemFARM is computed using equation (3.5) and (3.6) defined in Chapter-3, Section 3.7.3. The results are then compared with the untagged system and FARM as their precision and recall are already computed in Chapter-3, Section 3.7.3 shown in Figure 3-16 using the same equations, which are (3.5) and (3.6).

All results are plotted in Figure 4-4, which shows that precision is 1 for all three systems at 10% of recall; however, the precision is higher at most values of recalls for SemFARM followed by FARM and the untagged systems. For example, the precision of SemFARM, FARM and the untagged systems are 0.81, 0.62 and 0.46 respectively at the recall of 30%.

Table 4-3: Results of Precision and Recall calculated for three tests.

Test-1		Test-2		Test-3		Ave.	
Precision	**Recall**	**Precision**	**Recall**	**Precision**	**Recall**	**Precision**	**Recall**
1	0.083333	1	0.133333	1	0.1	1	0.1
0.75	0.25	1	0.266667	1	0.2	0.9167	0.2
0.8	0.333333	0.833333	0.233333	0.8	0.4	0.8111	0.3
0.714286	0.416667	0.777778	0.4	0.714286	0.5	0.7354	0.4
0.666667	0.5	0.8	0.533333	0.6	0.6	0.6889	0.5
0.636364	0.583333	0.692308	0.6	0.636364	0.7	0.655	0.6
0.615385	0.666667	0.642857	0.6	0.615385	0.8	0.6245	0.7
0.6	0.75	0.611111	0.733333	0.642857	0.9	0.618	0.8
0.647059	0.916667	0.619048	0.866667	0.5625	0.9	0.6095	0.9
0.545455	1	0.6	1	0.588235	1	0.5779	1

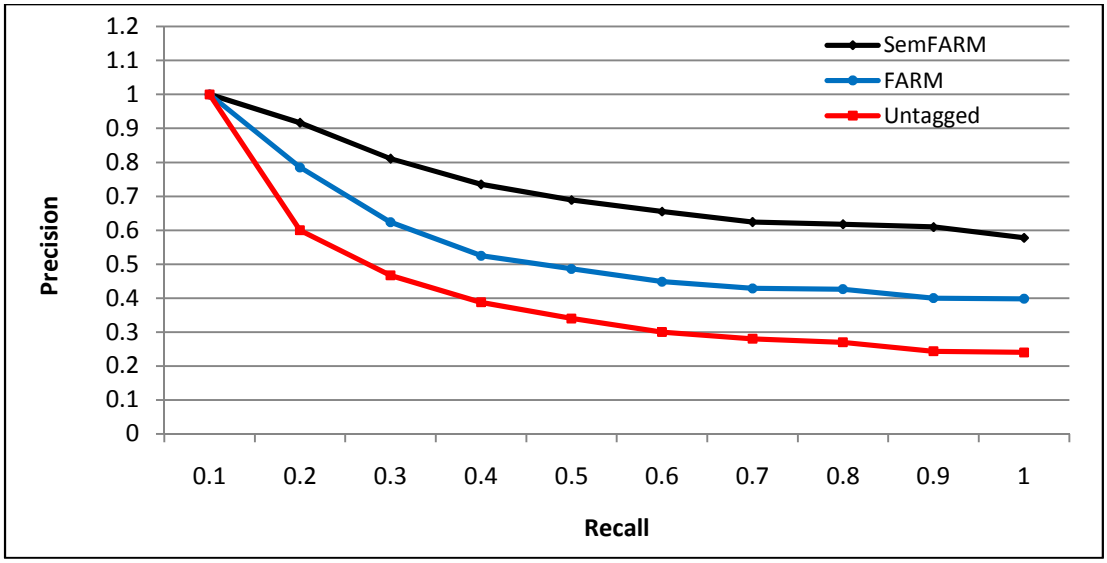

Figure 4-4: Comparison of Precision and Recall for Untagged, FARM and SemFARM

4.5.3 Probabilistic Evaluation

A generalized comparison is carried out by computing the probability of a successful file search for the following approaches:

(i) *SemFARM:* semantic-based search is performed using ontology definitions on metadata which consists of file attributes and keywords.

(ii) *FARM*: search is performed on metadata which consists of file attributes and keywords.

(iii) *Untagged*: search is performed on metadata consists of filenames only.

To compare the number of searches a user has to make in order to get the desired file let *p* and *q* as the probability of success and failure for *n* independent trials, the distribution for the number of trials until the first success occurs is defined by equation 3.2 presented in Chapter-3. Where g is the geometric distributed variable and redefining the p and q,

Let,

- p_{sf} represent the success probability in SemFARM
- q_{sf} represent the failure probability in SemFARM

- g_{sf} is the distribution for probability of x_{sf}^{th} trail being the first successful search for SemFARM

Similarly, using equation (3.2) for FARM and Untagged, Let,

- p_f represent the success probability in FARM
- q_f represent the failure probability in FARM
- g_f is the distribution for probability of x_f^{th} trail being the first successful search for FARM
- p_u represent the success probability in untagged system
- q_u represent the failure probability in untagged system
- g_u is the distribution for probability of x_u^{th} trail being the first successful search for

In order to find the success probability, 100 file search trials were carried out for each of the three systems in which SemFARM, FARM and untagged system returned 88, 82 and 52 queries successfully. Table 4-4 presents the geometric distribution calculated for first 5 trials of each system. It is evident from Figure 4-5 that the success probability of SemFARM is higher for the first trial as compared to FARM and untagged systems. This indicates that the probability of success for SemFARM is higher for lesser number of trials as compared to the rest of two systems.

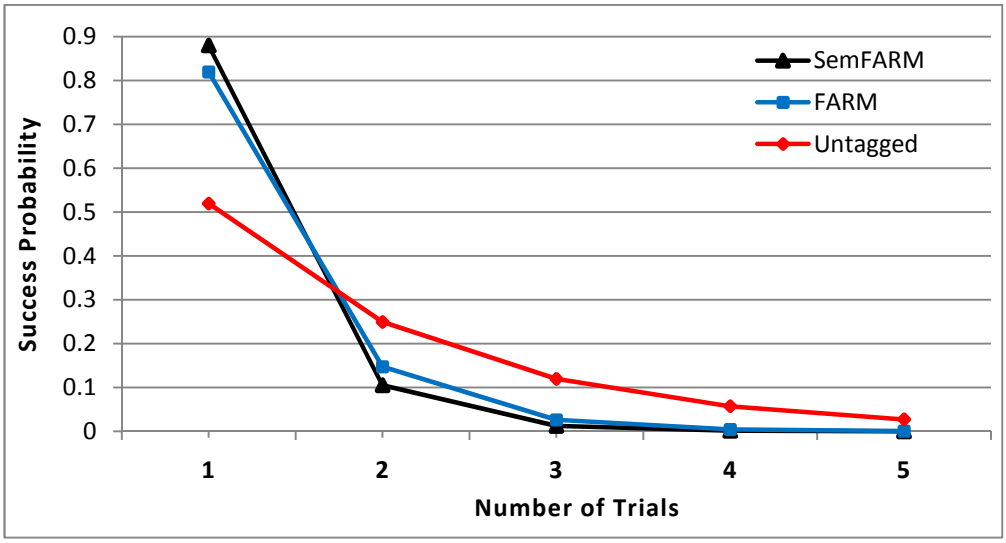

Figure 4-5: Success probability of trials for FARM, SemFARM and Untagged frameworks

In other words, the chances for getting a successful query in *lesser number of trials* are greater for SemFARM when compared to FARM and untagged approaches. The geometric

distribution presented in Table 4-4 is based on success probability *p,* which can be used to symbolize the general perception about the performance of SemFARM by calculating its maximum likelihood estimation.

Table 4-4: Comparison of values calculated by using geometric distribution.

x	1	2	3	4	5
$g_u(x_u; p_u)$	0.52	0.2496	0.1198	0.0575	0.0276
$g_f(x_f; p_f)$	0.82	0.1476	0.0265	0.0047	0.0008
$g_{sf}(x_{sf}; p_{sf})$	0.88	0.1056	0.01267	0.00152	0.00018

4.5.3.1 **Parameter Estimation for Geometric Distribution**

The maximum likelihood estimator of *p* for geometric distribution is based on results presented in Table -3. For a random sample x_1, x_2, $x_3 \ldots x_n$ from a geometric distribution, the likelihood function is as given in Chapter- 3, Section 3.7.2.1.

Using equation (3.3), the required number of trials on which the first success is expected can be calculated. For example, the value of $\hat{p}$ is 0.88 for SemFARM which means the first success is expected in 1.13 trials. Similarly, the first success is 1.21 and 1.92 for FARM and untagged systems respectively.

4.6 Summary

SemFARM was presented in this chapter to utilize the semantic technologies in file retrieval on low-end devices. The implementation of the proposed framework was presented to validate its feasibility in terms of efficiency and accuracy in file retrieval. A generic ontology was developed to define the most commonly used keywords to annotate a stored file. The matching degrees were defined in order to match the closest relevant keywords while searching for a required file. The framework was evaluated in terms of precision and recall to measure its accuracy in file retrieval on a mobile device. In addition, probabilistic evaluations were also presented.

CHAPTER 5

Aggregating Similarity Measures using Rough Sets in Ontology Alignment

Semantic Web technologies were employed in a file retrieval framework presented in Chapter-4. There are several challenging issues, which emerged in Semantic Web technologies due to heterogeneous knowledge resources. This Chapter presents an ontology alignment system (OARS) to deal with ontology heterogeneities and facilitate the interoperability in a semantic environment. OARS uses Rough Sets to aggregate the results obtained from different similarity measures during the ontology alignment process. Three main similarity matchers are used in OARS to map two entities from different ontologies including string-based, linguistic-based and structural-based matchers. This Chapter also presents the sub-matchers which are used to compute the similarities between the super-classes, sub-classes and properties of the two entities. The performance of the proposed alignments system is evaluated through various experimental tests and compared with existing state of the art alignments systems. In addition, various aggregating methods are also implemented and tested to compare the results with the rough sets based aggregating method. All the experimental tests are performed by using the benchmark data sets. The comparison results indicate that OARS gives improved recall values on different groups of data sets.

5.1 Similarity Measures

This section discusses various similarity matchers and specifically those which are used in OARS to map two entities from different ontologies. Each individual matching technique is treated as a matcher while the result obtained from its process is considered as the similarity between two entities. It should be pointed out that any matching technique (in isolation) is not adequate enough to give an accurate match between two entities and hence they are used as the combination of two or more, depending on the algorithm used in alignment system. The main and effectual similarity matching techniques include structural and lexical. Structural similarity techniques are used to find the similarity between the structural appearance of classis, properties and their instances in the ontology structure while lexical techniques are used to compute the similarity between the entities regardless of their structural appearance in the ontologies for example URI, classname and annotation etc. of entities.

The lexical similarity techniques may consider the entity name or label as sequence of characters, string or word as a whole. The combination of structural and lexical matching techniques gives much better idea about the overall similarity of a concept defined in ontology. OARS utilized the results of various similarity matching techniques which include string-based, linguistic-based and structural-based similarities. The structural based similarities compare the super-classes, sub-classes and properties for two entities. Most of the existing ontology alignment systems compare the similarities using more than one elementary techniques and then results of these techniques are aggregated by using an aggregation strategy depending on the implemented algorithm.

5.1.1 String-based Similarity

In string-based similarity calculation the entities are considered as strings, regardless of their structures or other associated properties defined in ontology. The string normalization process is made after the basic comparison of entity names. Both entity strings are

converted to lower-case and punctuations, dashes and blank character are eliminated. The normalization process play important role in string comparison techniques. For example, "*MasterThesis"*, "*Master-Thesis"* and "*Master thesis*" are normalized to *"masterthesis"*. There is a variety of techniques proposed to calculate the string similarities depending on characteristics of measurements. These techniques include sub-string distance [135], [56],

Levenstein [136], Jaro-Winkler[137], [138], Needleman-Wunsch [139] and n-gram similarity [140], [141]. *Cohen et al.* provided a good survey of the different methods to calculate string distance using various functions [142]. *Stoilos et al.* [89] proposed *Smoa* string metric which is based on intuitions about similarities presented in [143]. It computes the string similarity based on their commonalities as well as their differences. The *Smoa* metric is calculated by subtracting the sum of *differences* and *winkler* similarity from the *commonalities* of strings. The commonalities are calculated by the substring string metric. Let Sim_strng denote the string similarity between e_i and e'_i, then $Sim\_strng\ (e_i, e'_i)$ can be calculated using equation (5.1) as given below,

$$Sim\_strng\ (e_i, e'_i) = Smoa\ (e_i, e'_i) \tag{5.1}$$

To calculate the substring metric between two strings, a process to find and remove the biggest common substring is continued until no further common substring can be found. The lengths of these substrings are then added and scaled with the length of strings. The differences used in Smoa are computed by the length of unmatched strings. We use Smoa as string-based matcher in our proposed ontology alignment work.

5.1.2 Linguistic Similarity

Linguistic similarities are computed using external resources like language dictionaries, thesauri or specific databases. Such similarities are very useful when string-based similarities are not easy to find between entities and it happens when synonyms are used for the same concept in ontologies. For example, the names "*brochure*" and "*booklet*" refers to the same concept but the string-based similarity between them is low enough (*which is 6,*

using the Levenshtein distance) to be ruled out for selection as a mapping candidate. The WordNet is a similar kind of lexical database which provides a repository of lexical items defined as set of semantic vocabulary. OARS uses WordNet to exploit the information encoded in the names, labels or descriptions to a deeper extent, by looking up the synsets as defined by equation (5.2). In WordNet, different meanings of the same concept are grouped together as sets of synonyms (*synsets*) in terms of nouns verbs, adjectives and adverbs. In hierarchical manner, synsets are interlinked by means of various conceptual-semantic and lexical relations. For example, nouns have relationships of hypernym, hyponym, holonym, meronym and coordinate term. Similarly verbs are linked through relationships of hypernym, troponym, entailment, and coordinate terms.

Now considering the same example of entity names "*brochure*" and "*booklet*", this time using WordNet, the same entities would be selected as good candidates for mapping by using WordNet where the *brochure*, *folder*, *leaflet* and *pamphlet* are defined as synonyms. To compute the linguistic similarity of two words w_i and w'_i as defined by equation (5.2), denoting names/labels of entities e_i and e'_i from two ontologies O and O',

let

- $Sim\_lin(w_i, w'_i)$ is the linguistic similarity between w_i and w'_i
- Σ be the external resource *(WordNet)*
- $s(w_i)$ is the set of synonyms,
- $h(w_i)$ is the set of hyponyms and hypernyms and
- $t(w_i)$ is the set of antonyms of w'_i, we define

$$Sim\_lin(w_i, w'_i) = \begin{cases} 1 & if \ \ w'_i \in s(w_i) \\ 0.5 & if \ \ w'_i \in h(w_i) \\ 0 & if \ \ w'_i \in t(w_i) \end{cases} \quad (5.2)$$

The similarity relations for hyponyms and hypernyms are set to 0.5 and they are further investigated by considering the same result by using structural similarities defined in equations (5.3), (5.4), (5.5) and (5.6). However, it is inferred from the synonym and

antonyms that the words are totally *similar* or *dissimilar* respectively. One possible drawback of using such resources is that it may find more possible matches for the same concept. However, this issue can be tackled by investigating some structural information of such possible matches for entitles [144].

5.1.3 Structural Similarity

The structural similarity information plays vital role in situation where the linguistic or string based similarity between two entities proved to be insufficient or incomplete. This information between two entities comes from their structural features like, their relation with other entities and their direct properties. The main intuitions behind the structural similarity are given below;

- *If two classes from different ontology have similar upper-classes in hierarchy, it is likely that they define the same concept*.
- *If two classes from different ontology have similar sub-classes in hierarchy, it is likely that they define the same concept.*
- *If two classes from different ontology have similar properties, it is likely that they define the same concept.*
- *Two entities having any combination of two or all the three above mentioned similarities suggest more likelihood to be the similar concept.*

The structure similarity of the two entities $e_i$and e'_i from ontologies O and O'respectively, is computed by considering the similarities in terms of super-classes, sub-classes and properties. We have defined equation (5.3) to calculate the similarity between super-classes of two entities e_i and e'_i from two ontologies O and O'.

Let,

- $Sim\_hsp(e_i, e'_i)$ be the similarity between super-classes e_i and e'_i,
- $K_{sup}(e_i)$ be the set of super classes for entity e_i ,

- $K_{sup}(e'_i)$ be the sets of super classes for entity e'_i ,
- $|K_{sup}(e_i)|$ be the cardinality of $K_{sup}(e_i)$,
- $|K_{sup}(e'_i)|$ be the cardinality of $K_{sup}(e'_i)$, we have

$$Sim\_hsp(e_i, e'_i) = \frac{1}{2}\left(\frac{|(K_{sup}(e_i) \cap K_{sup}(e'_i))|}{|K_{sup}(e_i)|} + \frac{|(K_{sup}(e_i) \cap K_{sup}(e'_i))|}{|K_{sup}(e'_i)|}\right) \quad (5.3)$$

To compare the structural hierarchy, we need also to compare the similarity between the sub-classes of e_i and e'_i which is defined by equation (5.4) as follows,

Let,

- $Sim\_hsb(e_i, e'_i)$ be the similarity between sub-classes e_i and e'_i,
- $K_{sub}(e_i)$ be the set of sub classes for entity e_i,
- $K_{sub}(e'_i)$ be the sets of sub classes for entity e'_i,
- $|K_{sub}(e_i)|$ be the cardinality of $K_{sub}(e_i)$,
- $|K_{sub}(e'_i)|$ be the cardinality of $K_{sub}(e'_i)$, we have

$$Sim\_hsb(e_i, e'_i) = \frac{1}{2}\left(\frac{|(K_{sub}(e_i) \cap K_{sub}(e'_i))|}{|K_{sub}(e_i)|} + \frac{|(K_{sub}(e_i) \cap K_{sub}(e'_i))|}{|K_{sub}(e'_i)|}\right) \quad (5.4)$$

The similarity between the properties also plays an important role in suggesting the overall similarity of two entities in different ontology. We use $Sim\_pr(e_i, e'_i)$ as defined by (5.5) to represent the similarity between the properties and calculated as below,

Let,

- $Pr(e_i)$ be the set of properties of entity e_i,

- $Pr\ (e'_i)$ be the set of properties of entity e'_i ,
- $|Pr(e_i)|$ be the cardinality of $Pr(e_i)$,
- $|Pr(e'_i)|$ be the cardinality of $Pr\ (e'_i)$,

$$Sim\_pr(e_i, e'_i) = \frac{1}{2}\left(\frac{|(\ Pr(e_i) \cap Pr(e'_i))|}{|\ Pr(e_i)|} + \frac{|(\ Pr(e_i) \cap Pr(e'_i))|}{|Pr(e'_i)|}\right) \quad (5.5)$$

Finally, the overall structural similarity is computed by the average of three matchers which includes $\_hsp(e_i, e'_i)$, $Sim\_hsb(e_i, e'_i)$ and $Sim\_pr(e_i, e'_i)$ defined by equation (5.6) as given below,

$$Sim\_strc(e_i, e'_i) = {}^{1}/_{3}\left((Sim\_hsp(e_i, e'_i) + Sim\_hsb(e_i, e'_i) + Sim\_pr(e_i, e'_i)\right) \quad (5.6)$$

5.2 Using Rough Sets for Similarity Aggregation

Rough Sets theory is based on the indiscernibility relation of objects with respect to the available information which partitions the universe into sets of similar objects called elementary sets. Elementary sets can further be used to build knowledge about the real or abstracted world where the use of indiscernibility relation leads to information granulation [145]. The Rough Sets theory has proved to be a useful mathematical technique for analysing object descriptions. It assumes that every object of the universe is associated with a certain amount of information, represented by some attributes which express the description of objects [146], [147]. The concept of objects and their attributes in Rough Sets can be exploited to deal with uncertainties during the mapping process of ontology alignment when results of matchers does not give obvious indication either the entities are similar or vice versa and when such issue of uncertainty arises. When results of similarity matchers are considered as the attributes of the entities, they can be classified by employing

the techniques of Rough Sets theory. These classifications can further be used to decide the similarities between the entities based on their attributes, which holds the values of similarity matchers. The OARS uses Rough Sets techniques by defining the ontology entities as the objects in Rough Sets while the similarities of the entities are defined as the attributes of the objects in Rough Sets theory. Rough Sets theory provides techniques to analyze the object descriptions by assuming that every object of the universe is associated with certain amount of information, represented by some attributes expressing the description of objects. Similarly, each entity in ontology can be described by its available similarity measures obtained from different matchers during the process of comparisons. To formally derive the alignment uncertainty problem,

Let

- U be the set of entities as $U = \{e_1, e_2, e_3 \dots, e_n\}$,
- F is the set of matching factors as $F = \{f_1, f_2, f_3\}$,
- X be the subset of U,

The entities having similarities amongst them with respect to given matching factors are denoted by$[x]_F$. To approximate X with respect to matching results in set F, the lower and upper approximations are given below,

Let

- $\underline{F}(X)$ represent the lower approximation of set X with respect to F is the set of entities which are certainly belong to X.

$$\underline{F}(X) = \{ x \mid [x]_F \subseteq X \} \tag{5.7}$$

- $\bar{F}(X)$ represent the upper approximation of set X with respect to F is the set of entities which may possibly belong to X.

$$\bar{F}(X) = \{ x \mid [x]_F \cap X \neq \emptyset \} \tag{5.8}$$

$$\alpha_F(X) = \frac{|\underline{F}(X)|}{|\bar{F}(X)|} \tag{5.9}$$

The ratio of the accuracy will be $0 \leq \alpha_F(X) \leq 1$. The main objective is to compare the similarity of a selected entity with the unmapped entities from target ontology included in U. Considering two entities in set X which is definable [148], [149] with respect to *F*, when the accuracy of rough set is equal to 1. Thus, entities are considered for mapping when the accuracy of rough sets results in 1 after computations using equation (5.7), (5.8) and (5.9). During the alignment process where the $Sim\_strng\ (e_i, e'_i)$, $Sim\_lin\ (w_i, w'_i)$ and $Sim\_strc(e_i, e'_i)$ do not find exact matches between the entities, the similarity results from these matchers are recorded for each of unmapped element. The set *F* defines three matching factors as given below,

Let

- f_1 represent the value returned by comparing e_i and e'_i using string matcher; $Sim_{strng}(e_i, e'_i)$ as defined by equation (5.1) given in Section 5.1.3.

- f_2 represent the result which is computed by the mean of two results returned by comparing e_i and e'_i entities with $Sim\_hsp(e_i, e'_i)$ and $Sim\_hsb(e_i, e'_i)$ matchers according to equation (5.3) and (5.4) respectively as described in Section 5.1.3. Similarly,

- f_3 represent the result returned by comparing e_i and e'_i entities through $Sim\_pr(e_i, e'_i)$ matcher defined by equation (5.5) given in Section 5.1.3.

Based on experimental results and comparison presented in Section 5.4.3.2, the values are normalized to the nearest decimal values which are returned by individual matchers before computing the accuracy of Rough Sets. The confidence degree of 1 or 0.75 is assigned to the mapping relation when Rough Sets accuracy is computed with respect to full or any combination of two elements from set *F* respectively.

To explain the use of Rough Sets in OARS to compare the similarities between entities, few computations are presented as an example shown in Figure 5-1. Let us assume that both cases have 5 unmapped entities namely e'_1, e'_2, e'_3, e'_4 and e'_5 in the target ontology and the result values obtained using three matchers are given against each entity after comparing with entity e_i from the source ontology. In real world alignment process, number of entities and number of similarity factors between each entity may differ but we present the calculations for comparing e'_i with e'_1 and e'_4 only. We present two separate cases namely *Case-1* and *Case-2. Case-1* is presented to demonstrate the similarity calculation between the source and target entities explicitly based on 2 factors f_1 and f_2. While the Case-2 is presented to demonstrate the similarity calculation explicitly based on 3 factors f_1, f_2 and f_3. For computing the accuracy of Rough Sets for e_i ,based on the values given by three matchers, the set *F* should be $\{1,1,1\}$ for e'_i in an ideal case to map e_i and e'_i. Consider Case-1 as given in Figure 5-1;

- for $X = \{e'_i, e'_1\}$, with respect to $F = \{f_1, f_2\}$, the $\bar{F}(X)$ = $\{e'_i, e'_1, e'_3, e'_4\}$, and $\underline{F}(X)$ = ∅ suggests that e'_i and e'_1 are indefinable based on the given results and left unmapped.

- for $X = \{e'_i, e'_4\}$, with respect to F= $\{f_1, f_2\}$, the $\bar{F}(X)$ and $\underline{F}(X)$ = $\{e'_i, e'_4\}$, now the $\underline{F}(X) \neq \emptyset$ and the $\alpha_F(X) = 1$ suggests that e'_i and e'_4 are considered for mapping this time. The confidence degree value of 0.75 is assigned to mapping relation because the set *F* contain two elements in this case. Consider *case-2* as given in Figure 5-1;

- for $X = \{e'_i, e'_1\}$, with respect to $F = \{f_1, f_2, f_3\}$, the $\bar{F}(X) = \{e'_i, e'_4\}$, and $\underline{F}(X) = \emptyset$ again suggests that e'_i and e'_1 are indefinable based on the given results and left unmapped.

- for X = $\{e'_i, e'_4\}$, with respect to $F = \{f_1, f_2, f_3\}$, the $\bar{F}(X)$ and $\underline{F}(X)$= $\{e'_i, e'_4\}$, the $\underline{F}(X) \neq \emptyset$ suggests that e'_i and e'_4 are definable with respect to F, the

$\alpha_F(X) = 1$ and entities are considered for mapping with the confidence degree of 1.

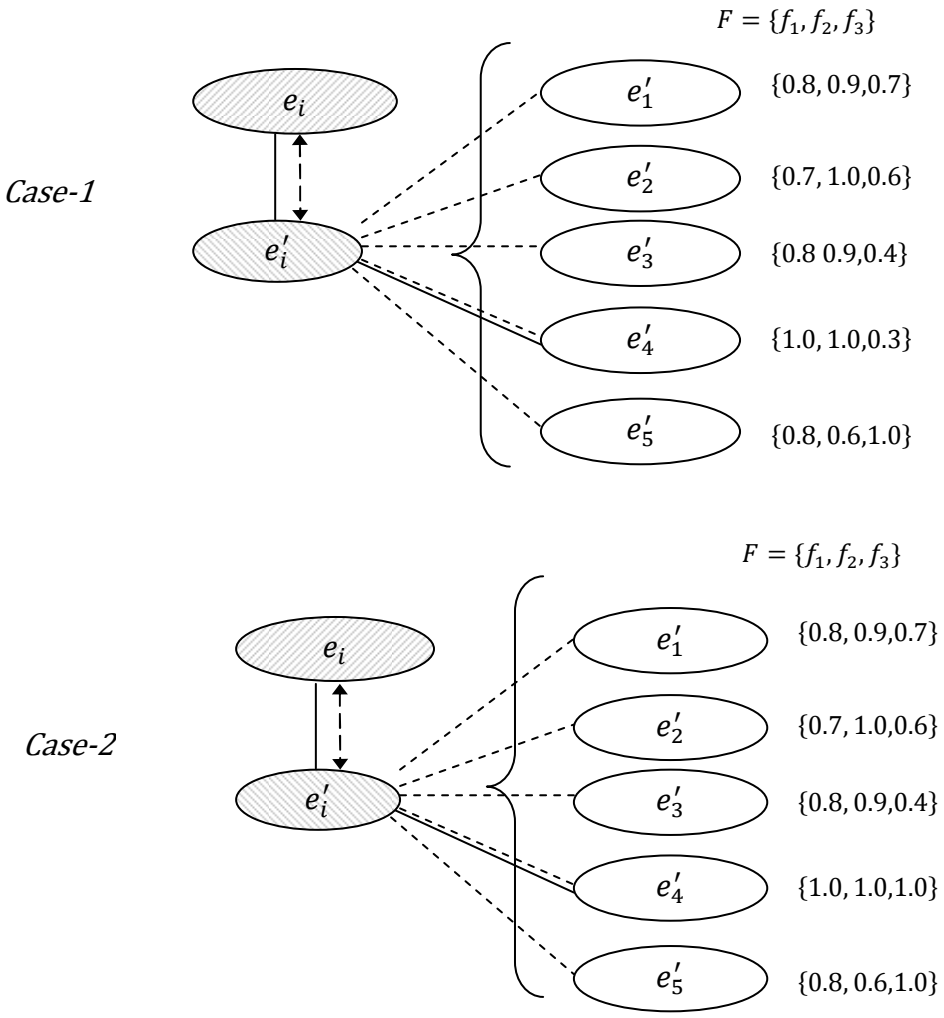

Figure 5-1: Example of Rough sets based comparison of similarities.

5.3 Alignment Process

There are several ontology definitions used in literature to define the ontology alignment process but the most commonly used definition is "*An ontology is the formal explicit specification of a shared conceptualization*" [150]. In this thesis, an ontology is defined as the following tuple given by equation (5.10);

$$O = \{ C, R, A, I \} \tag{5.10}$$

where *C, R, A* and *I* are sets of *concepts*, *relationships*, *axioms* and *instances*. The output of the matching process usually results in an alignment. The ontology alignment is defined by the process by equation (5.11) *as a correspondence function f with accuracy confidence c on the set of semantic relationships r between the entities* e_i *and* e'_i *belonging to source ontology* O *and target ontology* O' *respectively.*

Thus,

$$A' = f\{ e_i, c, r, e'_i \} \tag{5.11}$$

The OARS is implemented in Java and alignment APIs [151] are used for basic functions like input of source and target ontologies and alignment output. Therefore the aliment output is according to the specifications of OAEI as a fragment of alignment output is shown in Figure 5-2 where the two entities namely "*lastName*" are mapped from ontologies 101 and 205. The relationship " = " indicated the exact match relation between the two entities. The ontology matching process locates the best corresponding entity in target ontology O', for each entity of source ontology O. The two entities e_i and e'_i from source ontology O and target ontology O' respectively, are first compared by $Sim\_strng\ (e_i, e'_i)$, $Sim\_lin\ (w_i, w'_i)$ and $Sim\_strc(e_i, e'_i)$. If any of the matcher gives an exact match, the entities are selected as mapping candidate.

Before finalizing the mapping candidates, the mappings are further verified through other matchers. The entities having inexact matching results through base matchers are further processed through rough set approach, which is explained in Section 5.2. The overall main working model of OARS is presented in Figure 5-3. The process starts with taking two ontologies as input one as the source and other as the target ontology resulting in the third ontology which is considered as the alignment output of source and target ontologies. Therefore, leaving the source and target ontologies unchanged. The pre-processing of entities mainly includes string based normalization techniques as discussed in Section 5.1.1.

The three basic matchers used in OARS are defined by (5.1), (5.2) and (5.6) given in Sections 5.1.1, 5.1.2 and 5.1.3 respectively. The rough sets are computed for unmapped entities as described in Section 5.2.

```
<map>
  <Cell>
    <entity1
rdf:resource='http://oaei.ontologymatching.org/2010/benchmarks/101/onto.rdf#lastName'/>
    <entity2
rdf:resource='http://oaei.ontologymatching.org/2010/benchmarks/205/onto.rdf#lastName'/>
    <relation>=</relation>
    <measure rdf:datatype='http://www.w3.org/2001/XMLSchema#float'>1.0</measure>
 </Cell>
</map>
<map>
 <Cell>
    <entity1
rdf:resource='http://oaei.ontologymatching.org/2010/benchmarks/101/onto.rdf#country'/>
    <entity2
rdf:resource='http://oaei.ontologymatching.org/2010/benchmarks/205/onto.rdf#country'/>
    <relation>=</relation>
    <measure rdf:datatype='http://www.w3.org/2001/XMLSchema#float'>1.0</measure>
  </Cell>
</map>
```

Figure 5-2: A fragment of ontology alignment output.

The mappings are processed through verification process before being included in the final alignment. The mapping found with exact matches by basic matchers, are verified by other similarity measures. For example if two entities are found exactly similar through string-based matcher, their structural similarity is compared to verify the overall similarity of the entities. When two entities are selected for mapping through the rough sets based computation, a confidence value is associated as explained in Section 5.2.

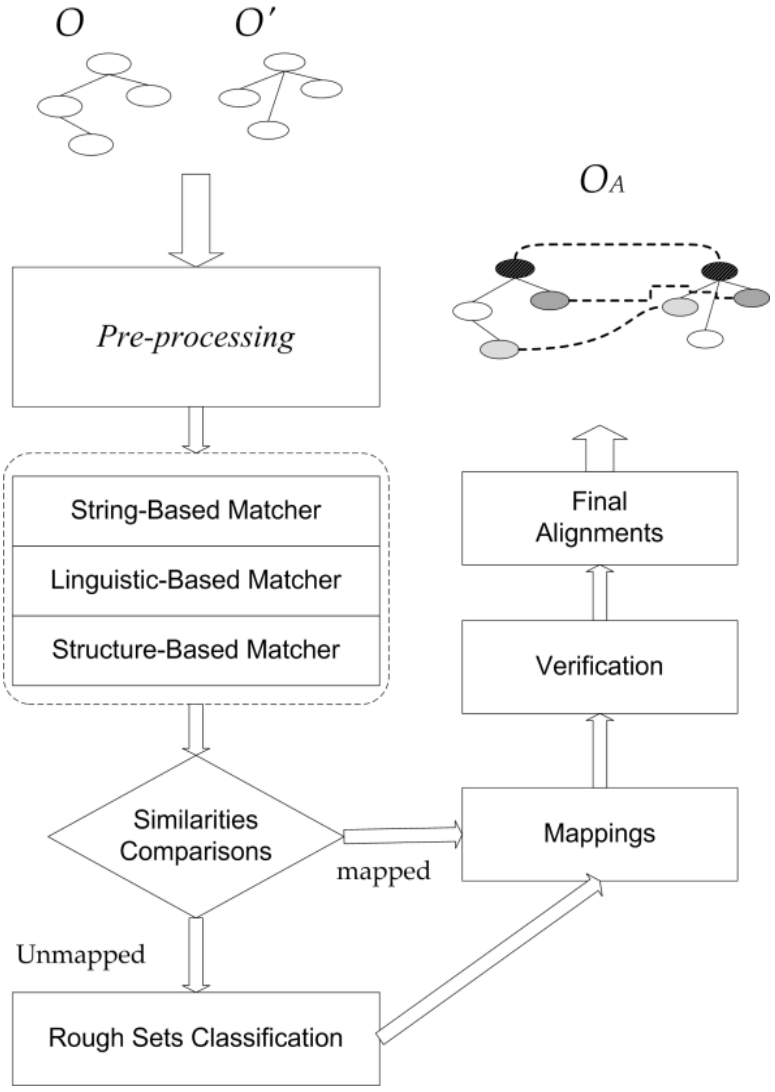

Figure 5-3: OARS alignment process.

5.4 Evaluation

This section presents an evaluation and analysis of the proposed and implemented ontology alignment system. The benchmark test ontologies from OAEI alignment campaign[4]-2010 were used to evaluate the performance of OARS. These benchmark tests offer various sets of ontologies to evaluate a wide range of features regarding the strengths and

[4] http://oaei.ontologymatching.org/2010

weaknesses of the matchers and alignment systems. The reference alignments are also available for tests which have been aligned manually and regarded as correct alignments.

5.4.1 Benchmark Data Sets

The OAEI 2010 benchmark data sets include number of artificial ontologies providing certain level of difficulties to test the alignment systems analytically. These ontologies are built from one *OWL* ontology on the bibliography topic. The base ontology is test-101 which is considered as reference ontology, containing 33 named classes, 24 object properties, 40 data properties and 76 individuals in which 20 of them are named while the rest are anonymous.

The descriptions of these tests are shown in Table 5-1, mainly containing three groups. These groups are namely *simple tests* (1xx), *systematic tests* (2xx) and *real-life* ontologies (3xx). Simple tests have 4 ontologies with minor variations that are aimed to compare with reference ontology. Ontologies in the systematic tests have been built to test the ability of the alignment systems when specific information is eliminated from the ontologies. The eliminated information may include the following;

- Classes are replaced with several classes, expanded or flattened.
- The entity names are replaced with synonyms, strings from other languages than English or even some random strings.
- Comments at different levels are translated other foreign language than English or suppressed at all.
- Properties are suppressed or their restrictions on classes are discarded.
- Instances are suppressed.
- Specialization hierarchies are expanded, suppressed or flattened.

Furthermore, ontologies in the third group are real world ontologies from different institutions and left unchanged in benchmark data set.

Table 5-1: Benchmark data set[5] description

Test sets	Description (regarding *source* and *test* ontology)
101-104	The hierarchical structure is similar Entity name is same or totally different
201-210	The hierarchical structure is similar Different linguistic used in some levels
221-247	Different in structure Label linguistic is similar
248-266	Hierarchical structure and linguistics are different
301-304	Real world ontologies

5.4.2 Evaluation Measures

The performance of OARS is evaluated through *Precision*, *Recall* and *F-measure*. Precision and recall are the most widely accepted and well-known principles in the research area of information-retrieval [152] and ontology-alignment [153]. The precision and recall explained in Chapter-3, Section 3.7.3 are redefined here to evaluate the OARS,

Let A_d be the set of discovered alignments from the set of total accurate alignments A_t.The precision, recall and *F-measures* are defined below;

$$Prec = \frac{|A_d \cap A_t|}{|A_d|} \tag{5.12}$$

$$Rec = \frac{|A_d \cap A_t|}{|A_t|} \tag{5.13}$$

Thus, precision defined by equation (5.12), is the ratio of correctly returned alignments over the total number of returned alignments while recall defined by equation (5.13), and is the ratio of correctly returned alignments over the total number of correct

[5] http://oaei.ontologymatching.org/2010/benchmarks/index.html

alignments. The harmonic mean of precision and recall is computed by $F-measure$ as defined by equation (5.14),

$$F-measure = \frac{2\times Prec \times Rec}{Prec+Rec} \quad (5.14)$$

5.4.3 Experimental Results

This section presents the performance evaluation of OARS in different aspects. The evaluation of similarities aggregation is presented to underline its effect on the results of overall performance in ontology alignment. The comparison of OARS with other existing alignment systems is also outlined in this section.

Critical analyses are presented to highlight the advantages and limitation of the proposed alignment system based on the results obtained by running OARS on benchmark data set ontologies. The alignment process we have implemented is totally automatic and hence, no user intervention involved in any tests during the alignment process.

5.4.3.1 Effect of Similarities Aggregation Algorithms

To evaluate the performance of the alignment system comprehensively, several test scenarios are formulated using benchmark data sets and evaluation criteria defined by equations (5.12), (5.13) and (5.14). The main purpose of these test scenarios is to assess the following,

- The efficacy of basic matchers
- The outcome of various results aggregating combinations
- The effect of using rough sets in aggregating results basic matchers

Four test scenarios are designed, each scenario uses different combination of matchers to aggregate the final mapping results. For this purpose, four algorithms are implemented

separately in the alignment system, namely *A1*, *A2*, *A3* and *A4* as defined by (5.15), (5.16), (5.17) and (5.18) respectively. The details of these four algorithms are given as below;

- *A1* represents the method where alignment is derived from the mean value of two results returned by string and linguistic matchers such that

$$A1 = (Sim_{strng}(e_i, e'_i) \ + Sim\_lin\ (w_i, w'_i))/2 \quad (5.15)$$

- *A2* represents the method where the mean value of results obtained by structural and linguistic matchers is used to compute the alignment such that

$$A2 = (Sim\_lin(w_i, w'_i) + \ Sim\_strc(e_i, e'_i))/2 \quad (5.16)$$

- Similarly, *A3* represents the method where the mean value of string and structural based matchers' results is considered for alignment computation. Thus *A3* is given as below

$$A3 = (Sim\_strng\ (e_i, e'_i) \ + \ Sim\_strc(e_i, e'_i))\ /2 \quad (5.17)$$

- Finally, *A4* represents the method where the mean value of string, linguistic and structural matchers is considered for alignment computation. *A4* is computed as given below

$$A4 = (Sim\_strng\ (e_i, e'_i) \ + Sim_{lin}(w_i, w'_i) \ + \ Sim\_strc(e_i, e'_i))/3 \quad (5.18)$$

Group 3xx of test ontologies from the benchmark data sets are used in these tests because it contains the real world ontologies as described in Section 5.4.1. Figure 5.4 shows the comparison of results obtained from all four algorithms defined by (5.15), (5.16), (5.17) and (5.18) in terms of precision, recall and F-measure. The set of ontologies in group 3xx have more string similarities as compared to structural and linguistic similarities with the

reference ontology. It is also evident from Figure 5.4 that those algorithms in which string-based matchers are incorporated in combinations with other matchers, show improvement in F-measure.

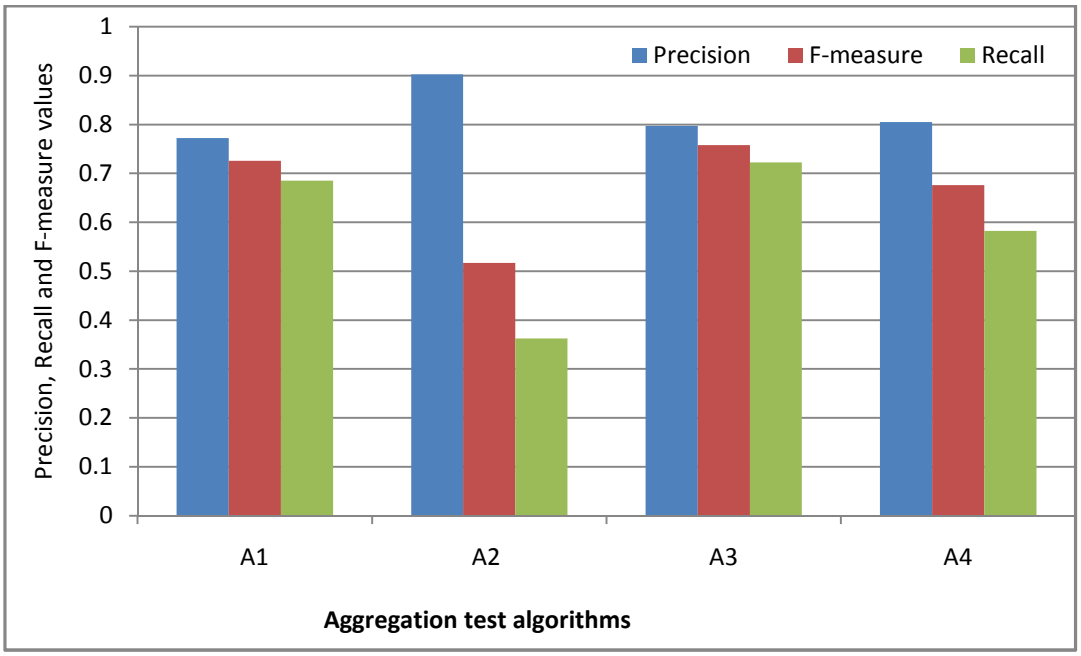

Figure 5-4: Precision, Recall and F-measure for various aggregation algorithms.

For example, the value of F-measure for A3 is the highest followed by A1 and A4 which is 0.758, 0.726 and 0.675 respectively and the only algorithm which does not use string-based similarity is A2 with the F-measure value of 0.516. This also shows the significance of single matcher in improvement of overall mapping performance. Similarly, there is comparatively less linguistic similarities in group 3xx ontologies with the reference ontologies.

For example, some correct alignment from one of the ontology include "*abstract*" = "*hasAbstract*", "*volume*" = "*hasVolume*" and "*copyright*" = "*hasCopyright*" but using WordNet synsets as linguistic matcher will not give good similarity results. Such results degrade the overall mapping performance of other matchers when the mean value of all matchers is taken in aggregation. In Figure 5-4, the A3 algorithm does not consider the result from linguistic matcher which has improved F-measure value than other algorithms.

Using the same group of tests i.e. 3xx, we run the tests on OARS and the overall results in terms of precision, recall and F-measure are shown in Figure 5-5. The main difference

remains in the aggregation method, where OARS uses Rough Sets classification on unmapped entities and utilizes the similarity values of basic matchers while A4 considers the mean value of results obtained from the basic matchers. There is a significant improvement in the performance of OARS as compared to A4 in terms of all three evaluating factors. The precision, recall and F-measure values of A4 are 0.805, 0.582 and 0.675 respectively while for OARS these values are 0.862, 0.845 and 0.83 respectively. The overall improvement achieved by OARS in F-measure is 22.96% over A4 algorithm.

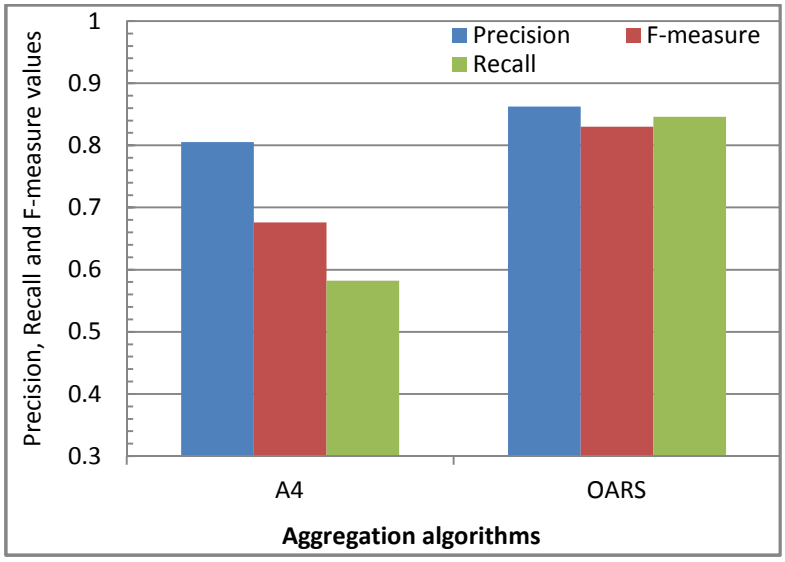

Figure 5-5: Comparison Precision, Recall and F-measure calculated for aggregation algorithms.

These results confirm that, firstly, no single matcher is sufficient enough to count on for achieving good mapping results in an alignment system. Secondly, any single matcher that achieves the higher similarity results will improve the overall performance of alignment system. Thirdly, aggregating the results from different matchers by taking their mean value is not only insufficient but can also reduce the overall mapping performance when some matchers present low similarities.

Finally, it is the technique which utilizes the results from different matchers, play a vital role in the overall performance of an ontology alignment system

5.4.3.2 Selection of Normalization Value used in Rough Sets

To select the most appropriate value for the boundary region used in Section 5.4.3.2, various tests are performed considering the values including 50, 33.33, 25, 20, 10 and 5. These tests are performed on group 2xx of benchmark data sets. Figure 5-6 shows that the using the value "50" it achieves the highest recall value but on the other hand it gives the lowest precision value. Similarly, using the value "5" gives highest precision value but gives the lowest recall value as compared to other values. We found the value"10" as the most appropriate value with the highest value in terms of F-measure and therefore selected for using as the rough set boundary region value in OARS.

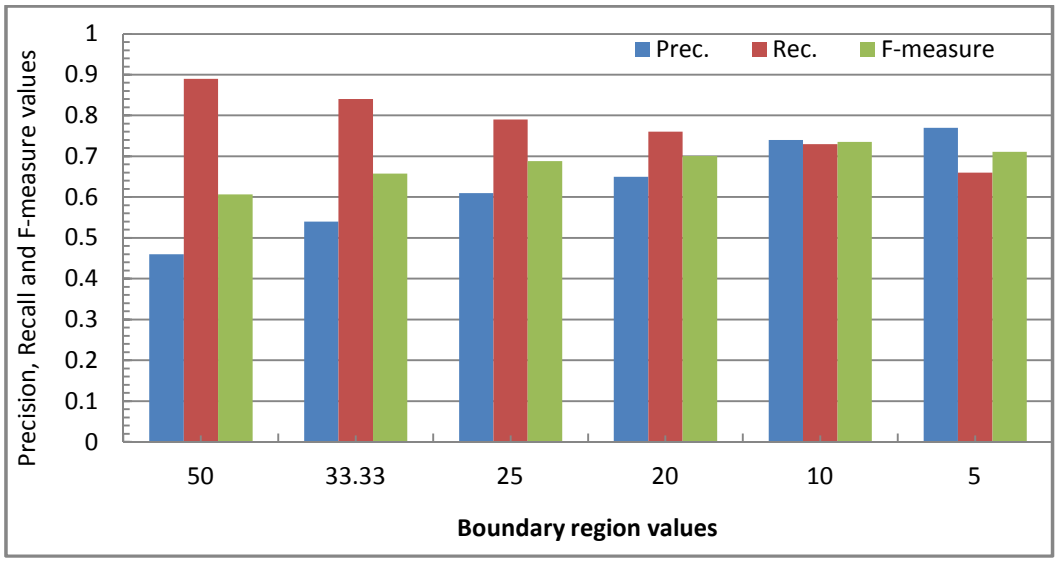

Figure 5-6: comparison of Precision, Recall and F-measure calculated for different boundary region values.

5.4.3.3 Comparison of OARS with Representative systems

The group wise analysis of results obtained by OARS running on benchmark data sets is given as below,

Group-1xx

Almost all of the systems in comparison achieved perfect results for test 1xx in terms of precision and recall with the exception of TaxoMap where it has achieved noticeable lower

recall value of 0.34. These are the basic tests to align two ontologies with almost similar entities or little language generalization is required. OARS use string-based and linguistic-based matchers to cope with such heterogeneities and hence these modifications have perfectly tackled by our alignment system.

Group 2xx

Ontologies in this group having altered the comments and labels were mostly aligned rightly and the use of WordNet as a linguistic matcher in OARS proved imperative dealing with synonyms (for example in test 205) and provided good alignments. Similarly, minor string modifications were also successfully detected by using the smoa in our string-based matcher. The linguistic matcher proved helpful in ontologies where linguistics were used for example in test ontologies 201, 202 and 248-266. However, the system performance was not very satisfactory where foreign language was used instead of English as the OARS currently do not support or implemented any dictionary resources for such languages.

Furthermore, ontologies with only structural changes were also tackled successfully because when this information was suppressed, the linguistic or string similarities were still available in the ontologies. We found the most challenging alignment task was to deal with those ontologies where both structural and labels modifications were made. However, ontologies with little structural changes were verified by the other available structural information. OARS system achieved better recall value than most of systems which are compared in Table 5-2. Other systems which have achieved higher recall for this test group include ASMOV, AgrMaker and RiMOM with the values of 0.89, 0.83 and 0.84 respectively. This is because of the fine classification of entities regarding the similarity values when the basic matchers did not suggest complete similarities. Such a similarity values are processed by Rough Sets and the computation of its accuracy can deal with analogous situations. The mappings suggested by rough sets are further strengthened by verifying with the value of other basic matchers.

There are only four systems which has the recall value higher than 0.8 leading by OARS followed by ASMOV, AgrMaker and RiMOM while only five systems in comparison have the precision value lower than 0.9 as shown in Figure 5-7, which includes TaxoMap, MapPSO, Falcon, CODI and edna.

Table 5-2: Comparison of results achieved by alignments systems on benchmark datasets.

Benchmark test	1xx		2xx		3xx		H-mean	
System	Prec	Recall	Prec	Recall	Prec	Recall	Prec	Recall
refalign	1	1	1	1	1	1	1	1
edna	1	1	0.43	0.57	0.51	0.65	0.45	0.58
AgrMaker	0.98	1	0.95	0.84	0.88	0.58	0.95	0.84
AROMA	1	0.97	0.94	0.46	0.83	0.58	0.94	0.48
ASMOV	1	1	0.99	0.89	0.88	0.84	0.98	0.89
CODI	1	1	0.83	0.42	0.95	0.45	0.84	0.44
Ef2Match	1	1	0.98	0.63	0.92	0.75	0.98	0.65
Falcon	1	1	0.81	0.63	0.89	0.76	0.82	0.65
GeRMeSMB	1	1	0.96	0.66	0.9	0.42	0.96	0.67
MapPSO	1	1	0.67	0.59	0.72	0.39	0.68	0.6
RiMOM	1	1	0.99	0.83	0.94	0.76	0.99	0.84
SOBOM	1	1	0.97	0.74	0.79	0.75	0.97	0.75
TaxoMap	1	0.34	0.86	0.29	0.71	0.32	0.86	0.29
OARS	1	1	0.92	0.90	0.86	0.83	0.90	0.87

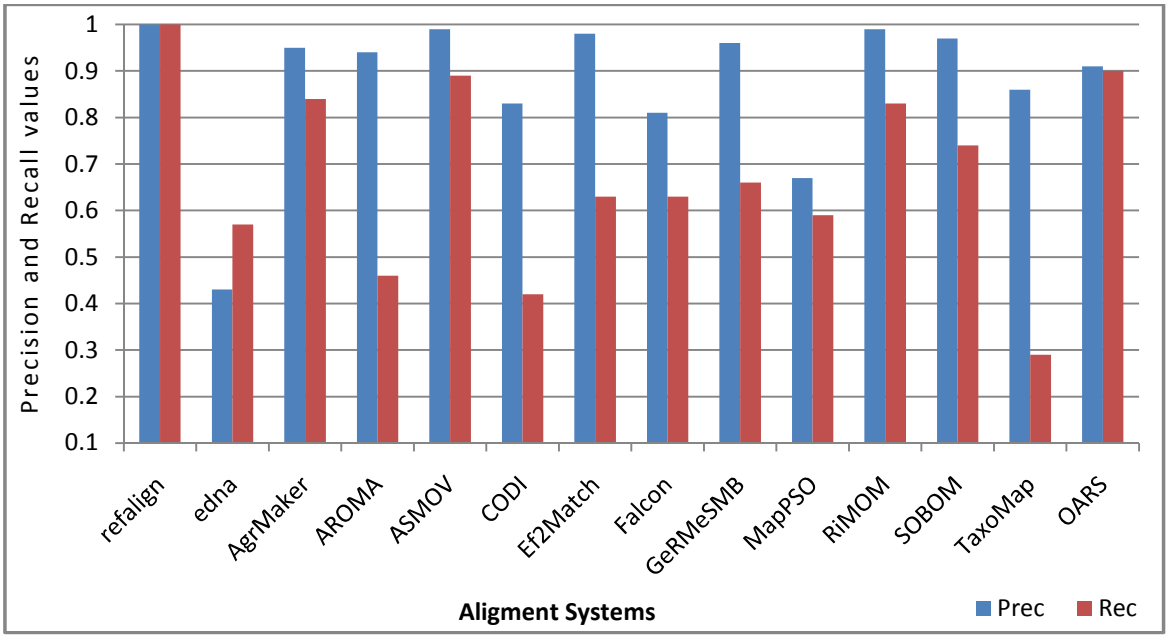

Figure 5-7: Comparison of Precision and Recall results achieved by alignment system for test group-2xx.

Group 3xx

These real world ontologies have the blend of obscurities found in group 2xx. In this group of tests, the string-based matcher performed very well as there are little structural information available in these ontologies. For example in ontology 302 there is no structural information about ontology so the system totally relied on other matchers. However, relying on more than one sort of ontology characteristics and using them in appropriate sequence somehow gives us reasonable results in terms of precision and recall. The test results for this group suggests that the performance of OARS has attained better recall value of 0.86 than most of the others while the precision value of 0.83 is better than MapPSO, SOBOM and TaxoMap only. As shown in Figure 5-8, only OARS and ASMOV have a recall value higher than 0.8 while TaxoMap, MapPSO and GeRMeSMP having lower recall values which are less than 0.5. The same performance trend in term of precision and recall can also be observed in this group of tests that improvement in recall is higher than improvement in terms of precision.

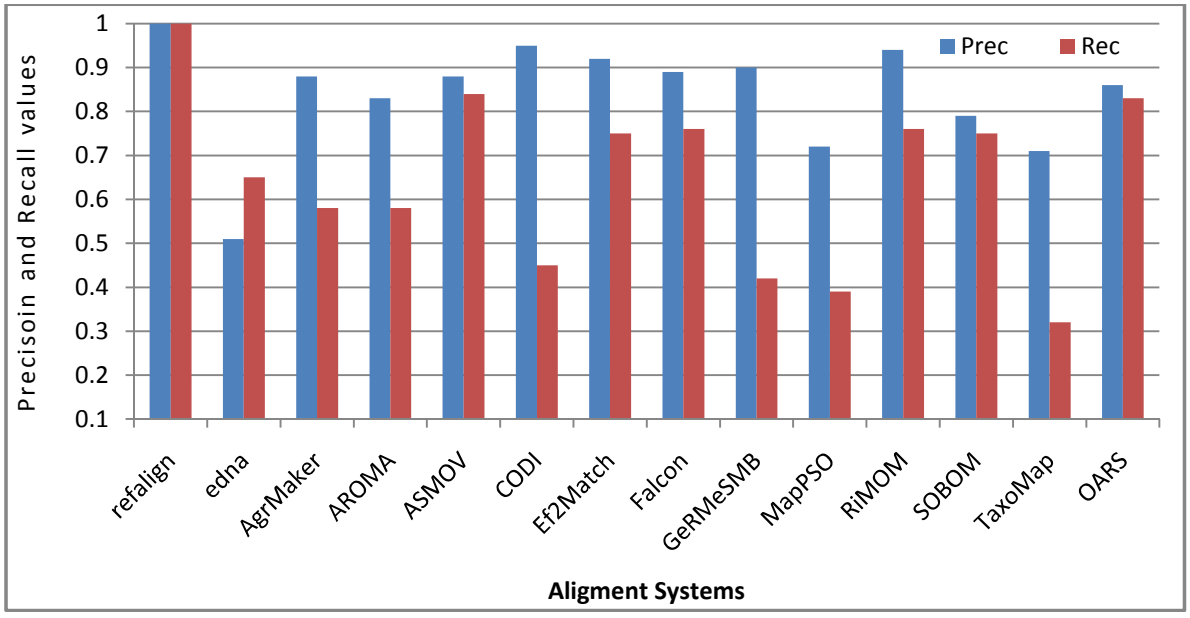

Figure 5-8: Comparison of Precision and Recall results achieved by alignment system for test group-3xx

5.4.4 Result Analysis

Using Rough Sets in alignment process, gives the notch to make use of little available information about an ontology and then trying to verify as much as possible by all other available information. By using rough sets in OARS alignment system, the overall recall value has improved but there is no significant improvement in the precision. For example, if we examine Table 2, it is evident that ASMOV, Ef2Match, GeRMeSME, RiMOM and SOBOM have achieved superior precision than ours and have attained the values which are above 0.95. However, the improvement in terms of recall is achieved at a little of cost decrease in the value of precision. It is worth mentioning here that both precision and recall are considered when the performance of an alignment system is evaluated. It is believed that the verification process after calculating rough sets on entities needs more investigation to improve the performance of OARS in terms of precision. It should be noted that the results of all other alignment systems are accessed from the OAEI website (http://oaei.ontologymatching.org/2010/). These alignment systems have not used *exactly similar* settings for testings because of their different requirements used in their working approach.

5.5 Summary

This chapter has presented an ontology alignment system OARS, which uses a combinational approach in order to map two entities from different ontologies. The alignment system uses rough sets to aggregate the results obtained from different similarity matchers. Various similarity matchers were reviewed and their working processes were explained. A brief introduction to rough sets and its implementation in the proposed alignment system was presented in Section 5.2. Three measures were used namely, precision, recall and F-measure, to examine the accuracy of the alignments obtained by OARS. The performance of OARS was evaluated and compared with some existing alignment systems and it showed good results, notably an improvement of recall. Various other tests were performed to evaluate the effectiveness of using rough sets in OARS.

CHAPTER 6

Semantic-based File Retrieval on Low-End Devices with Ontology Alignment Support

This chapter presents the augmentation in file-retrieval framework by utilizing the efficacy of ontology alignment. A semantic-based file retrieval system was presented in Chapter-4, where a generic ontology was utilized to define the meanings of general keywords used to annotate files on low-end devices. Furthermore, an ontology alignment system was presented in Chapter-5 to map two different ontologies developed in the same domain in order to extract extra information by utilizing both ontology definitions at the same time. To further augment the search potential of semantic-based file retrieval on low-end devices, the search module has been extended to exploit the ontology alignment facilities. This extension enables the device users to take advantage of all ontologies, which might have been developed in the same domain. For instance, if two generic ontologies are developed to define two different concepts in such a way, that the second ontology defines a concept which is a sub-concept in the first ontology. In this case, each query, which has the keywords defined by the second ontology, will have more information available to retrieval system as compared to the information available form first ontology.

This chapter also presents various approaches to integrate the alignment system and their implications. The overall improvement in semantic-based file retrieval search module is analysed by implementing and integrating ontology alignment system and comparing the results obtained from different experimental tests.

6.1 Alignment Utilization

The ontology alignments can be utilized by various techniques, which totally depend on the necessity of applications and scenarios. A brief overview of the most widely used techniques is presented with respect to their feasibility by employing them in SemFARM.

6.1.1 Ontology Transformation

The transformation process on two ontologies is unidirectional processes in which the alignments are used to describe the entities of one ontology in context of the second ontology. Similarly, the process may also be used in reverse direction where the second ontology is used to describe the entities of first ontology. This characteristic of ontology transformation leads to the fact that the semantic of ontology may get changed during the transformation process. The overall transformation can also be carried out by transforming the models. In model transformation, one model is taken as an input and another model is generated as the output, while both conforming to a given *metamodel*. Similarly, the transformation can be achieved by detecting ontology pattern detections [154]. Initially, the results of ontology pattern detection are taken from the corresponding query along with the ontology transformation pattern which is related to the results. Then a set of operations are generated according to ontology transformation pattern. A general process of ontology alignment is shown in Figure 6-1 where ontology A and B are aligned and their resultant aligned ontology is C.

6.1.2 Ontology Merging

Generally, the ontology merging process uses the alignments to generate a new ontology by combining two or more conceptually different ontologies. There are two main types of the merging process in terms of input ontologies. In first type, the input ontologies are replaced with the merged ontology while in the second type, the merging ontologies

remain unchanged in the process and the newly generated ontology is considered as the union of merging ontologies. The merging process varies depending on the application and specifically on the language used for defining the ontologies to be merged. The bridging axioms are generated when the two merging ontologies are expressed in the same language. In this process, all such entities from both ontologies are included, which had no match during the alignment process. Thus, the individual unmatched entities remain unchanged within the new ontology along with alignments which represents the common entities from different ontologies involved in merging process. Ontology merging is very useful when reasoning is required from more than one ontology.

Ontology merging process can be accomplished manually, semi-automatically or in fully automatic fashion. However, the manual process of merging proved to be more time consuming and not viable for many applications. Therefore, several systems and frameworks have been proposed and implemented in recent years as discussed in Chapter 2. In order to facilitate the file retrieval on low-end devices using SemFARM, the alignments are used in merging the ontologies in the first step.

The OntoBuilder is an ontology merging tool [155] [156] which generates an ontology after the merging process and then maps the ontology to the query form. It extracts the information from the visited websites for answering the query. The ontology is merged with a global ontology in its adaptation phase where the query is answered. Similarly, the approach which is presented in this chapter to integrate ontology alignments with SemFARM, use a merged ontology to search a required file. However, the algorithms used for alignment and the working model of both systems are different.

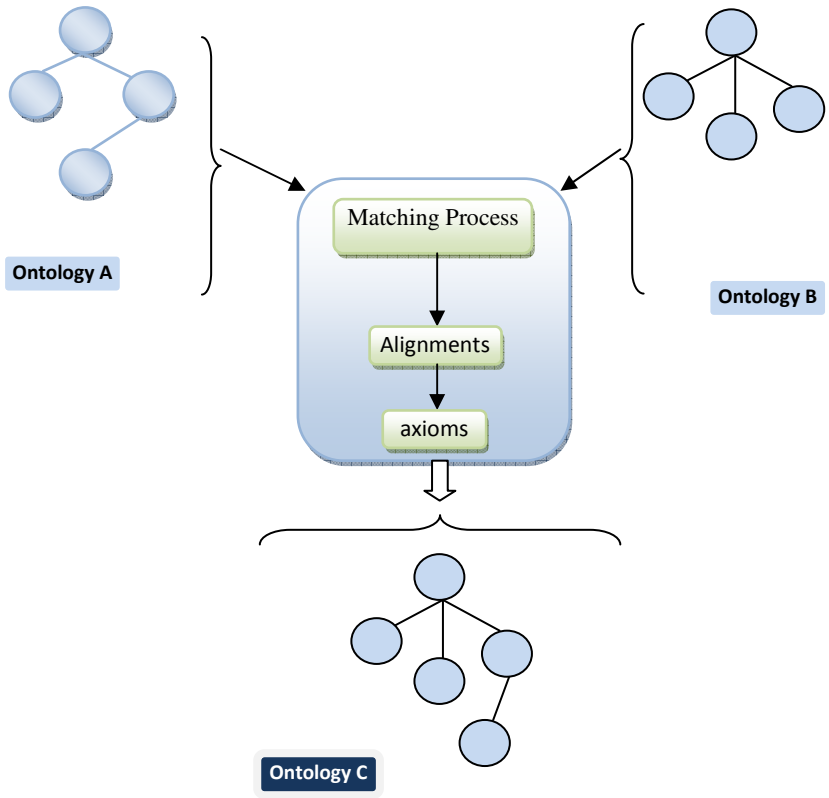

Figure 6-1: A general process of ontology alignment.

6.1.3 Ontology Mediation

The process of ontology mediation uses the alignment for bridging different ontologies in order to achieve interoperation between different applications. Therefore, the mediation process requires not only the alignments but also their specifications in order to use them in a specific scenario. Some of the available literature terms the ontology mediation as an independent software component, which mediates between two other components in such a way that mediation is achieved by ontology transformation and data translation. The ontology transformation transforms a query definition from the perspective of one ontology to another, as discussed in Section 6.1.1. If required, the data translation is

performed on the answered query by inverting the alignments to ensure the compatibility amongst heterogeneous applications.

6.1.4 Translation

The alignments are used to generate the transformation program for the process of translation to extract data without importing the ontology concerned. The translation is defined as an operation in context of data translation including ontology language [157]. For example, translating the ontology language RDFS to OWL or other possible axioms. The translation process ensures to maintain the semantics of ontology even if the syntax is changed.

6.1.5 Reasoning

This process uses alignment results as the rules for reasoning amongst different ontologies. The performance of the reasoning is based on the fact that how the heterogeneity between alignments and rules is resolved. For example, the SWRL which is a de-facto standard for defining OWL language based rules is considered as the knowledge-driven querying language. The OntoEngine[158] is an inference system which performs automated reasoning in merged ontologies in order to translate ontology. It has the key feature of indexing structures for managing multiple ontologies and controls the rules for ordering both forward and backward chaining operations.

The PROMPT framework is included in Protégé [108],[159] as an extension which is used to manage multiple ontologies and provide users the key tasks like ontology aligning, merging and translating between different formalisms. iPROMPT is a component of PROMPT which is a semi-automatic tool providing a user interface to interact with the merging process. It merges the ontologies by relating the concept structures in ontologies in interactive fashion.

6.2 Integration in SemFARM

To exploit the ontology alignment capabilities in semantic-based file retrieval on low-end devices, a new search module was implemented in java. The same file annotation module was used as described in Chapter-3, Section 3.4, which automatically annotates the files with three basic attributes and two user entered fields. Similarly, the meta-data are automatically parsed and stored in XML structured document. The search module presented in this chapter, utilizes the ontology alignment work as presented Chapter-5, in order to search a required file. Figure 6-2 shows the overall working processes of the search module, where the receiving file queries are answered after merging the two existing ontologies. When multiple OWL ontologies are found on the query answering system, they are first aligned and these alignments are then converted to bridging axioms in order to utilize the alignments as single ontology.

For this purpose, initially one of the renderer class OWLAxiomsRendererVisitor was used from the ontology alignment API package [151] The API provides OWL axioms for expressing the equivalence, sub-sumption and exclusively relations. This renders the obtained alignments as a merged ontology of the two input ontologies. Once the merged ontology is acquired, the ontology model and the RDF model are bind together to form an inference model. The RDF model is automatically created from the XML document by the XML to RDF converter module as shown in Figure 6-2. Finally, the file-search query is answered by navigating the inference model for query-word as explained in Chapter-4 Section 4.2. The list of the names of files is then sent back to the corresponding sending device as the query result.

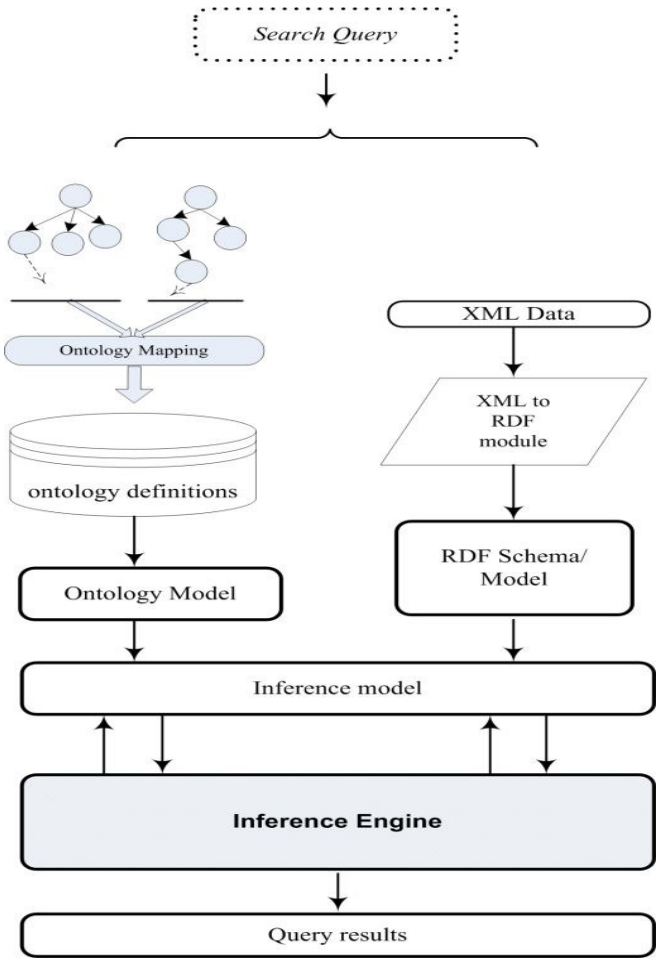

Figure 6-2: Search module of SemFARM framework with ontology alignment support.

6.3 Evaluation

A supplementary ontology was developed for evaluating the performance of SemFARM after integrating the ontology alignment features. The domain concept of the supplementary ontology was selected from a sub-concept of the main generic ontology which was used in the implementation of SemFARM. The main idea was to evaluate the integration of ontology alignment and how it leverages the file retrieval search module in SemFARM by employing the ontology alignments which were obtained from the newly developed and main ontology.

6.3.1 Performance Evaluation Environment

Two case studies are provided for evaluation purposes, which are defined as given below;

- *Case-1: SemFARM without ontology alignment*

  One generic ontology was utilized in the setup (SemFARM) to retrieve the required files. In this case, the search module of SemFARM utilized the knowledge extracted from the main ontology only. Hence, a single ontology was utilized in this setup therefore the alignments were not required and used.

- *Case-2: SemFARM with ontology alignment*

  Two ontologies were utilized in the setup to retrieve the required files. The search module of SemFARM utilized ontology alignments which were obtained by aligning the main and second ontologies. In this case, more knowledge was obtainable in the form of two ontologies and the capability of the search module by employing the ontology alignments.

Three set of tests were formulated in order to demonstrate the efficacy of ontology alignment in file retrieval on low-end devices. For this purpose, precision and recall was used as the performance measures which were described in Chapter-3, Section 3.7.3. The same measures were used to evaluate the performance of SemFARM framework presented in Chapter-4, Section 4.5.2.

The final precision and recall values were calculated by the average of three tests. In each test set, different numbers of files were annotated with such keywords, which were considered as relevant to the file-searching query. The numbers of relevant files were kept different in order to obtain the values of Recall. It should be noted that some of the files were also annotated with such keywords, which were not defined by the main generic

ontology. However, these keywords were defined in the second ontology, which was only aimed to develop for a limited concept domain.

Furthermore, the number of query-words were also kept different at each test set but it was made assured that the query-words contains keywords from both ontologies to give a fair chance to both cases in each test. Similarly, the same query-words were used for both cases in each corresponding test. However, different query-words were used in different set of tests. In order to elaborate it more clearly, Figures 6-3 and 6-4 are presented which shows the fragments from two ontologies used for the evaluation purpose.

For instance, the general ontology used in SemFARM defines the concept "*subjects"* as shown in Figure 6-3, while the sub-concepts of *subjects* are defined in the second ontology like, "*physics*", "*java*" and "*chemistry*" etc. as illustrated in Figure 6-4. In this case, when the file searching query-word is defined in the supplementary ontology which defines the sub-concept of main ontology, the file may not be retrieved without the integration of ontology alignment. Thus, it is the integration of ontology alignment which takes advantage of the knowledge veiled in more than one ontology.

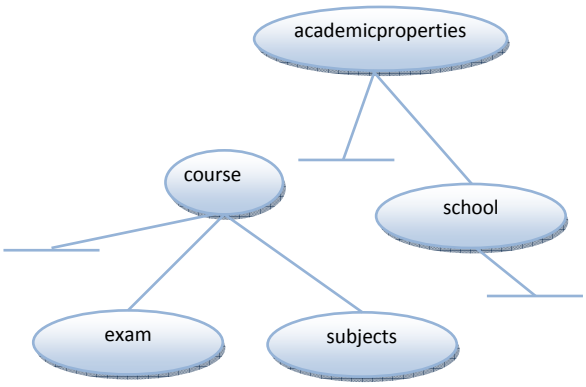

Figure 6-3: A fragment of the generic ontology used in SemFARM.

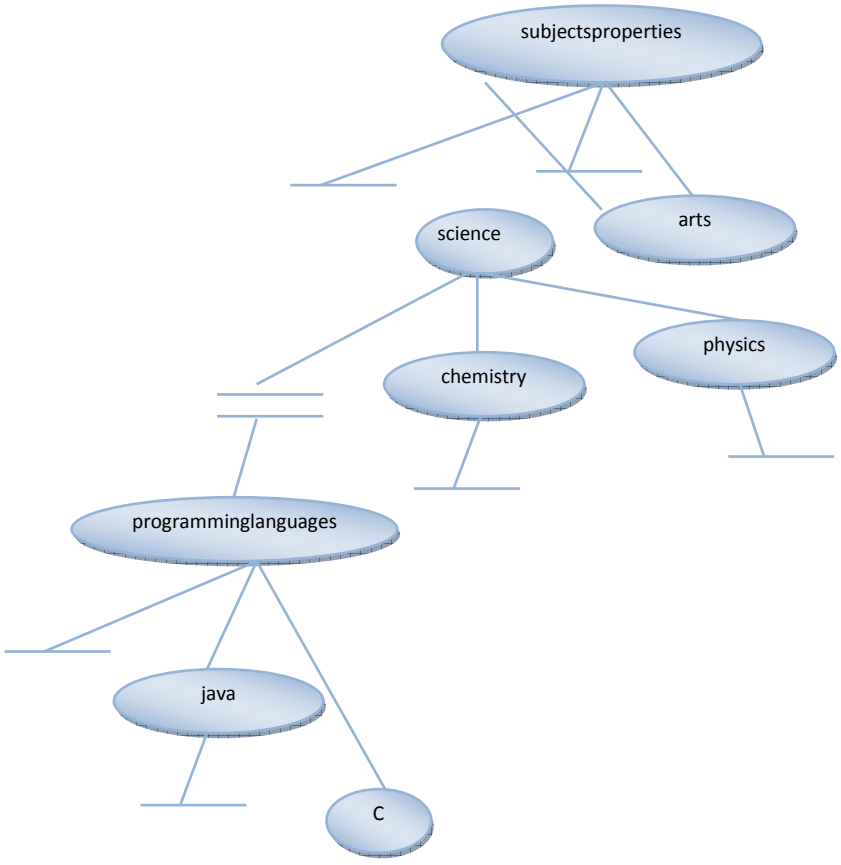

Figure 6-4: A fragment of the supplementary ontology.

6.3.2 Computing Precision and Recall

The experimental result obtained from all three tests for case-1 and case-2 are given in table 6.1 and 6.2 respectively in terms of their precision and recall values. The overall comparison of both tables suggests the improvement of case-2 in terms of precision against the same values of recall. For example, the average precision values of case-1 and case-2 are 0.65 and 0.72 respectively against the same recall value of 0.5. It can also be observed that the decrease in precision values for case-2 is lesser than the decrease in precision values for case-1 as the recall value goes higher from 0.1 to 1. This can be further elucidated by the statement that the precision (*average*) values decreases from 1 to 0.49 for case-1 and 1 to 0.61 for case-2 as the corresponding recall values increases from 0.1 to 1. To give an overall idea about the comparison which is based on the average precision values and recall computed for both cases is shown in Figure 6-5. The figure suggests that the integration of ontology alignment improves the overall performance in file retrieval system. For example the precision values for case-2 and case-1 are 0.616 and 0.492 respectively at the recall value of 1.

Table 6-1: Comparison of Precision and Recall values calculated for case-1.

Test set-1		Test set-2		Test set-3	
Precision	*Recall*	*Precision*	*Recall*	*Precision*	*Recall*
1	0.083	1	0.133	1	0.1
0.75	0.25	0.75	0.2	1	0.2
0.8	0.333	0.667	0.267	0.8	0.4
0.714	0.417	0.667	0.4	0.714	0.5
0.667	0.5	0.7	0.467	0.6	0.6
0.636	0.583	0.538	0.467	0.636	0.7
0.615	0.667	0.571	0.633	0.538	0.7
0.534	0.797	0.556	0.767	0.571	0.8
0.5294	0.85	0.524	0.833	0.563	0.9
0.458	0.947	0.519	0.963	0.5	1

Table 6-2: Comparison of Precision and Recall values calculated for case-2.

Test set-1		Test set-2		Test set-3	
Precision	*Recall*	*Precision*	*Recall*	*Precision*	*Recall*
1	0.0833	1	0.1333	1	0.1
0.75	0.25	1	0.26667	1	0.2
0.8333	0.3167	0.8333	0.3333	0.8	0.3333
0.8571	0.4	0.7778	0.4	0.7142	0.499
0.7778	0.5	0.8	0.5333	0.6	0.6
0.7273	0.61	0.6	0.6	0.6363	0.7
0.6923	0.75	0.6428	0.6	0.6153	0.8
0.6667	0.8333	0.6111	0.7333	0.6428	0.9
0.6875	0.9167	0.6667	0.9333	0.5625	0.92
0.5714	1	0.6521	0.9999	0.625	1

The precision values are the same at the recall value of 0.1 in both cases because as the number of retrieved files is lower, there are lesser chances that the retrieved file will be irrelevant. It should also be noted the overall precision of case-1 has decreased slightly as compared to the precision of SemFARM which is presented in Chapter-4. The main reason behind this decrease is the mandatory inclusion of keywords (from the second ontology) in the file retrieval query-words. While computing the overall precision for evaluating the performance of SemFARM in Chapter-4, the query-words were selected randomly.

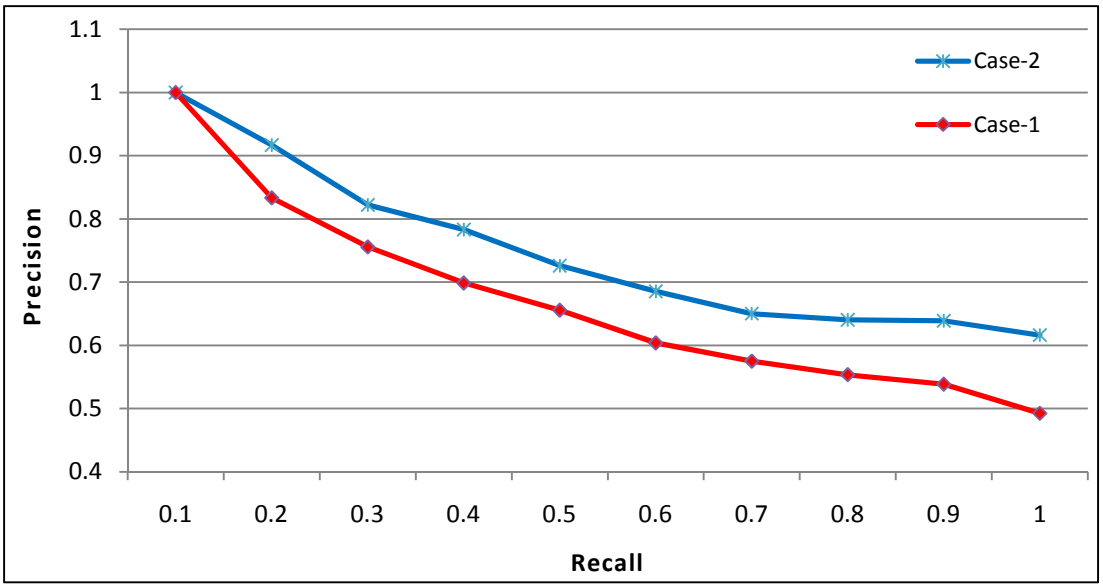

Figure 6-5: Comparison of precision and recall calculated for both cases.

6.3.3 Probabilistic Evaluation

The performance of both cases can also be evaluated through probabilistic evaluation by calculating their geometric probability distribution, which can give a general intuition about the success probability of searching a required file. In order to find the success probability 30 file-search trials were carried out for both cases, with the success rate of case-1 and case-2 was 0.82% and 833 % respectively. Each time, a file search query-word was selected randomly from a predefined set of query-words, which were collected from the keywords defined by the two ontologies in use. To calculate the geometric probability distribution for both cases the equation (3.2) which is presented in Chapter-3, Section 3.6.2, is further defined an given below;

Let

- g_{c1} represents the geometric distribution for the probability of x_{c1}^{th} trial being the first successful search for case-1
- p_{c1} represents the success probability for case-1

- g_{c2} represents the geometric distribution for the probability of x_{c2}^{th} trial being the first successful search for case-2
- p_{c2} represents the success probability for case-2

$$g_{c1}(x_{c1};\ p_{c1}) \tag{6.1}$$

$$g_{c2}(x_{c2};\ p_{c2}) \tag{6.2}$$

The probability distribution calculated for the first 5 trials of both cases is computed by equation (6.1) and (6.2) and results are presented in Table 6-3. The difference between the values computed for both cases is lesser but the general trend about the success probability can be observed in Figure 6-6.

Table 6-3: Probability distribution for Case-1 and Case-2.

x	1	2	3	4	5
$g_{c1}(x_{c1};\ p_{c1})$	0.82	0.1476	0.02656	0.00478	0.00086
$g_{c2}(x_{c2};\ p_{c2})$	0.833	0.13911	0.02323	0.00387	0.00064

The success probability of retrieving a file on the very first trial is higher for case-2 as compare to the probability of retrieving a file in case-1. When the first two values are considered jointly for both cases, it can be depicted that there are 97.21% and 96.76% chances for case-2 and case-1 respectively, that a file will be retrieved on the first two attempts.

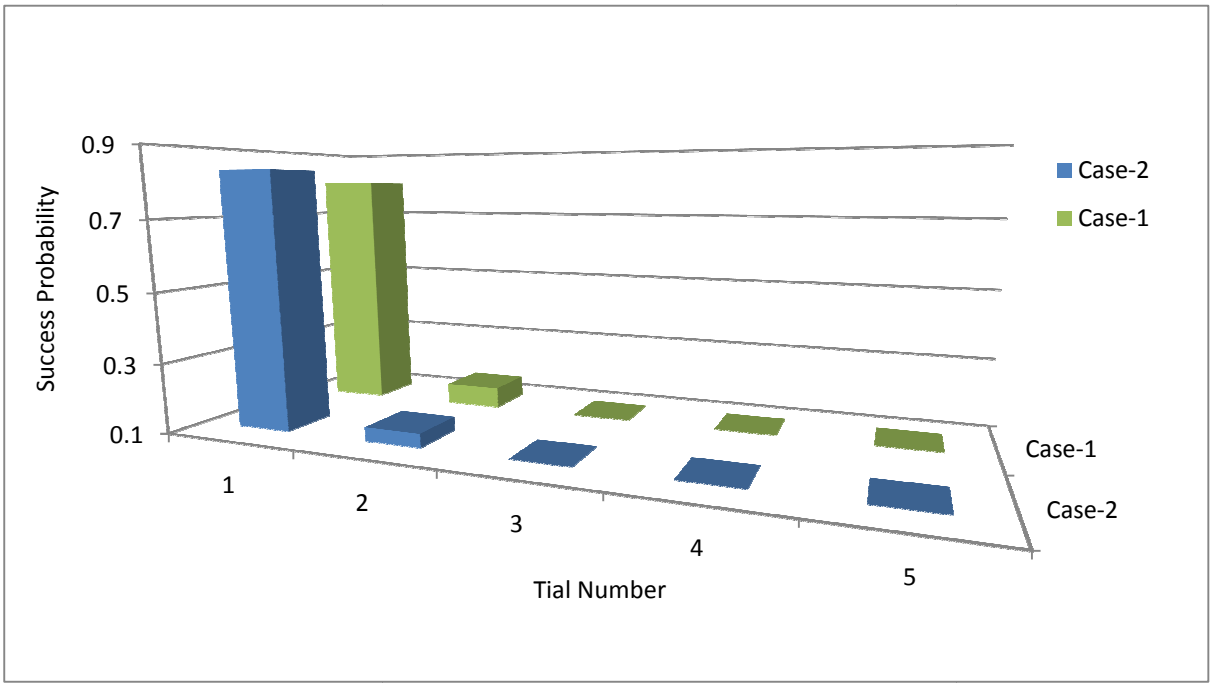

Figure 6-6: Comparison of success probabilities calculated for both cases.

6.4 Summary

This chapter presented the integration of ontology alignments with the semantic-based file retrieval framework for low-end devices, which was presented in Chapter-4. For this purpose, the alignments were obtained by the ontology alignment process which was proposed and implemented in Chapter-5. An overview was presented in Section 6.1 to highlight different techniques exploiting the ontology alignments for semantic interoperability between applications. The implementation was presented in Section 6.2 to augment the practicality of the proposed idea, followed by its performance evaluation in Section 6.3. The performance was evaluated with respect to the overall improvement in file retrieval framework in terms of the most acceptable measures namely, precision and recall. Probabilistic evaluations were also presented in Section 6.3.3 to measure the performance with respect to device users.

CHAPTER 7

Conclusions and Future Work

This chapter presents the main conclusion and summarises the contributions proposed in this thesis. The future work section highlights those research areas where the findings of this research can further be used and suggests new research directions.

7.1 Conclusion

This thesis has investigated the emerging issue of managing large number of files on low-end mobile devices and proposed practical frameworks for facilitating the file retrieval. The main reason for creating metadata in any information retrieval system is to facilitate the discovery of the required information and its use has proven to be very effectual. The FARM framework presented in this thesis automatically annotates stored files with their corresponding attributes. These attributes are considered as meta-data of files and parsed through a tiny parser to assemble them in XML structure. The annotation and file retrieval mechanisms proposed in FARM were cautiously designed keeping the resource limitations of the devices in mind. The proposed mechanism was further investigated for employment in a networked environment where all the connected devices can be searched for the required file using the same approach and without its modification. The proposed framework was implemented as a case study, by developing several MIDlets in the JME platform to validate its feasibility. However, the proposed framework can also be implemented on other mobile platforms by investigating their file systems to extract the file information. The performance of FARM was evaluated by performing different experimental tests and a significant improvement was noted in file retrieval. Improvements were also noted in terms of accuracy in file retrieval and accessibility in the use of the framework.

The semantic web technologies were investigated in detail by specifically focusing on different techniques, which can be utilized for file retrieval on resource-limited devices. Based on the findings, a semantic-based framework (SemFARM) was proposed which makes use of a generic ontology to formally define the most commonly used keywords. SemFARM makes it possible to give additional knowledge to the keywords which can be associated with files as their corresponding metadata. The generic ontology was developed using OWL language.

Matching degrees were defined in order to match the closest relevant keywords while searching for a required file. The performance evaluation of SemFARM was presented in terms of precision and recall to determine its accuracy and efficiency in file retrieval on resource limited devices. In addition to precision and recall, probabilistic evaluations were also presented to demonstrate the efficacy of SemFARM from the perspective of a device user.

The thesis has presented an ontology alignments system OARS, which utilizes rough sets to aggregate the results obtained from different matchers when computing the similarities between two entities from different ontologies. State of the art ontology alignment systems were reviewed and analysed in order to investigate different matching techniques and their implications for the performance of overall ontology alignment. Three matching techniques were implemented to find the similarity between the super-classes, the sub-classes and the properties of two entities e_i and e'_i. The most common heterogeneities in ontologies were also investigated and reviewed while proposing the OARS.

The performance of the OARS was evaluated and compared with state of the art alignment systems. While aggregating the results of different similarity matchers, various techniques were examined and evaluated by implementing separate algorithms in the OARS. The implications of using rough sets in the proposed alignment system as the aggregation method was also evaluated by comparing the alignments results with other implemented algorithms. The overall results were evaluated in terms of most acceptable measures in this filed of research, namely *Precision*, *Recall* and *F-measure.* The results

showed that the OARS gives good alignment result when compared with other alignments systems, notably in terms of recall.

The thesis has presented a framework which utilizes the alignments to facilitate file retrieval on low-end devices. The main goal of ontology alignment is to overcome the issue of semantic interoperability between heterogeneous application environments and to share the knowledge presented by different ontologies in the same domain concept. Similarly, the proposed framework has enabled the file searching framework to extract the knowledge from different ontologies to empower the file search capabilities. For this purpose, the ontology alignment system, presented in Chapter-5 was used to perform the mapping between two ontologies defined in the same domain. These alignments were employed in SemFARM framework which was presented in Chapter-4, and allowed its search module to exploit the knowledge of more than one ontology to retrieve a required file efficiently. The framework was assessed by designing two test-case scenarios. In the first test-case, the SemFARM framework utilizes one ontology in order to retrieve a required file, while in the second test-case, the search module of SemFARM was enabled to make use of more than one ontology by aligning them before answering a search query. The results were measured in terms of precision and recall and showed improvements in both cases.

7.2 Future work

There are several recommendations which can be used for future research directions in facilitating file retrieval and semantic operability, specifically on low-end devices.

7.2.1 File Annotation

Metadata plays a vital role in any information retrieval system because it explains and describes the stored information. Obviously, the more knowledge a system has about the stored information, the better are the chances of retrieving it successfully. The FARM framework presented in Chapter-3, annotates the stored files with their three basic

attributes namely, filename, file size and date of creation. These attributes are extracted from the underlying operating system of the device. The file systems can further be investigated and more attributes can easily be associated automatically as the metadata of the files. For example, the types of files can be guessed through their extensions like file names "*mypic.jpg*" and "*titanic.avi*" suggest that it is an image and video file respectively. The addition of such information to metadata may further improve the file searching mechanism.

7.2.2 Semantic-based Search

The ontology used in the SemFARM framework was developed in OWL language and the search module implemented in SemFARM is also fully compatible with OWL. However, the interoperability between different ontology languages can also be integrated to further improve the overall performance of file retrieval in a connected environment. The design of a light reasoner for low-end devices is another research area which can further be investigated to facilitate semantic applications on such devices. One of the possible solutions might be the design of an application specific reasoner which only employs the required part, instead of implementing the whole reasoner. The SemFARM framework may further be investigated for its potential employment on other mobile platforms other than JME. Furthermore, the framework may also be investigated to deploy as a web service where mobile users can be benefited without downloading the application on their mobile phones. For this purpose, an application specific browser can be developed which can handle the annotation and search processes of the SemFARM.

7.2.3 Ontology Alignment

Investigation of the following issues has the potential to improve the ontology alignment system:

- An alignment-database can be designed such that all the alignments can be stored and used in future alignment processes.

- The alignments which are obtained from aligning those ontologies which are defined for the same domain can be grouped together. These alignments can also be utilized in the following ways:

    (a) The alignments systems can be trained to learn from the previous alignments stored in the database.

    (b) Any query based system may take advantage of the stored alignment without repeating the alignment processes.

    (c) These alignments can be used as upper ontologies.

    (d) The domain specific stored alignment can be effectively used in resolving the conceptual heterogeneity between ontologies.

- An effective learning mechanism can be investigated to take full advantage of the previously stored alignments, specifically from the alignments in similar domains.

- The integration of an additional matcher which uses foreign dictionaries to translate the words between different languages for example English to French, or vice versa, or any other language can be investigated. This may prove very useful for aligning ontologies which are defined in different languages.

- The integration of an independent matcher in the alignment process where syntactic heterogeneity of ontologies can be resolved, can be investigated.

7.3 References

[1] W. Cathro, “Metadata: An Overview,” in Standards Australia Seminar: Matching Discovery and Recovery, (1997), http://www.nla.gov.au/nla/staffpaper /cathro3.html, last accessed May (2011).

[2] A. Sen, “Metadata management: past, present and future”, Decision Support Systems 37 (1), pp. 151–173. (2004).

[3] G. Salton. “Introduction to Modern Information Retrieval”, McGraw-Hill, New York, (1983).

[4] S. Handschuh, St. Staab, F. Ciravegna, “S-CREAM-semi-automatic creation of metadata”, in: A. Gomez-Perez (Ed.), The 13th International Conference on Knowledge Engineering and Management (EKAW-2002), Springer Verlag, (2002).

[5] D. Petrelli, V. Lanfranchi, F. Ciravegna, ”Working Out a Common Task: Design and Evaluation of User-Intelligent System Collaboration”, In Proceedings of Tenth IFIP TC13 International Conference on Human-Computer Interaction (INTERACT 2005) Rome, September (2005).

[6] A. Kuchinsky, C. Pering, M.L. Creech, D. Freeze, B. Serra, Gwizdka J., “FotoFile: A Consumer Multimedia Organization and Retrieval System”, In Proceedings of ACM CHI99 Conference on Human Factors in Computing Systems, 496-503, (1999).

[7] M. Tuffield, S. Harris, D. Duplaw, A. Chakravarthy, C. Brewster, N. Gibbins, K. O'Hara, F. Ciravegna, D. Sleeman, N. Snadbolt, Y. Wilks, “Image Annotation with Photocopain” in the Proceedings of the fifteenth world wide web conference (www06), Edinburgh, May (2006).

[8] J. Tang, X.-S. Hua, G.-J. Qi, Y. Song, and X. Wu, “Video Annotation Based on Kernel Linear Neighborhood Propagation,” IEEE Transactions on Multimedia, vol. 10, no. 4, pp. 620–628, (2008).

[9] V. Lanfranchi, F. Ciravegna, D. Petrelli, “Semantic Webbased Document: Editing and Browsing in AktiveDoc”, Proceedings of the 2nd European Semantic Web Conference , Heraklion, Greece, May 29-June 1, (2005).

[10] M. Vargas-Vera, E. Motta, J. Domingue, M. Lanzoni, A. Stutt, F. Ciravegna, “MnM: Ontology driven semi-automatic or automatic support for semantic markup”, In Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management, EKAW02. Springer Verlag, (2002).

[11] C. A. N. Soules, G. R. Goodson, J. D. Strunk, and G. R. Ganger, “Metadata efficiency in versioning file systems”, Conference on File and Storage Technologies, pages 43–58. USENIX Association, (2003).

[12] F. Amardeilh, “Semantic Annotation and Ontology Population”. Semantic Web Engineering in the Knowledge Society, ISI Global, (2008).

[13] A. Ouksel, and A. Sheth, “A Brief Introduction to the Research Area and the Special Section”, SIGMOD Record (Special Section on Semantic Interoperability in Global Information Systems) 28(1): 5–12, (1999).

[14] J. Heflin and J. Hendler, “Semantic interoperability on the web”. In Extreme Markup Languages-2000, (2000).

[15] P. Thornton, & C. Houser, “Using mobile phones in education”. In Proceedings of the 2nd international workshop on wireless and mobile technologies in education (WMTE ’04), Taiwan, (2004).

[16] C. Markett, I. Arnedillo Sánchez, S. Weber and B. Tangney, “Pls turn ur mobile on: Short message service (SMS) supporting interactivity in the classroom”. In: Kinshuk, D.G. Sampson and P. Isaias, Editors, Cognition and exploratory learning in digital age, International Association for Development of the Information Society Press, Lisbon, pp. 475–478, (2004).

[17] A. Holzinger, A. Nischelwitzer, M. Meisenberger, “Mobile Phones as a Challenge for m-Learning: Examples for Mobile Interactive Learning Objects (MILOs)”. In: Tavangarian, D (ed.) 3rd IEEE PerCom, pp. 307–311, (2005).

[18] I. Juzang, T. Fortune, S. Black, et al. “A pilot programme using mobile phones for HIV prevention”, J Telemed Telecare. Journal of Telemedicine and Telecare; 17: 150–153, (2011).

[19] Parikh, S. Tapan, Javid, Paul, Sasikumar, K., Ghosh, Kaushik, and Toyama, Kentaro, ”Mobile phones and paper documents: Evaluating a new approach for capturing microfinance data in rural India” In Proceedings of the SIGCHI conference on human factors in computing systems, Montr´eal, Qu´ebec, Canada, pp. 551–560. NewYork: ACM Press, (2006).

[20] L. Srivastava, “Mobile phone and evolution of social behaviour”, Behav Inf Technol 24(2): 111–129, (2005).

[21] J. Vincent, “Emotional Attachment to Mobile Phones: An Extraordinary Relationship”, In Mobile World: Past, Present and Future, Springer, pp. 95-104, (2005).

[22] J. Vincent, “Emotional Attachment and Mobile Phones”, Knowledge, Technology, & Policy, Vol. 19, No. 1, pp. 39-44, (2006).

[23] R. Hardy and E. Rukzio, “Touch & Interact: Touch-Based Interaction of Mobile Phones with Displays,” in Proceedings. 10th Int’l Conf. Human-Computer Interaction with Mobile Devices and Services (MobileHCI 08), ACM Press, pp. 245–254, (2008).

[24] M. Silfverberg, I. MacKenzie, S.,&Korhonen,P., “Predicting text entry speed on mobile phones”, Proceedings of the CHI 2000 Conference on Human Factors in Computing Systems. New York: ACM, (2000).

[25] I. S. MacKenzie, and R.W. Soukoreff, “Text entry for mobile computing”, Models and methods, theory and practice. Human-Computer Interaction, 17(2&3), p.147—198, (2002).

[26] K. Lyons, T. Starner, D. Plaisted, J. Fusia, A. Lyons, A. Drew, and E. Looney, “Twiddler typing: One-handed chording text entry for mobile phones”, In Human Factors in Computing Systems (CHI 2004 Proceedings), pages 671 – 678, Vienna, Austria, ACM Press.April 27-29, (2004).

[27] L. Seungyon, Shumin Zhai, “The performance of touch screen soft buttons”, Proceedings of the 27th international conference on Human factors in computing systems, April 04-09, Boston, MA, USA (2009).

[28] W3C: http://www.w3.org/, last accessed May (2011).

[29] Extensible Markup Language, http://www.w3.org/XML/, last accessed May (2011).

[30] M. C. Daconta, L. J. Obrst, and K. T. Smith, ”The Semantic Web: a guide to the future of XML”, Web services, and knowledge management. Indianapolis, Ind.: Wiley Pub, (2003).

[31] kXML, http://kxmlrpc.objectweb.org/, Last accessed May (2011).

[32] J. Knudsen, “Parsing XML in J2ME”, http://developers.sun.com/techtopics/mobility /midp/articles/parsingxml/ Last accessed May (2011).

[33] R. Neches, R Fikes, T Finin, T Gruber, R Patil, T Senator, and W Swartout. “Enabling Technology for Knowledge Sharing”, AI Magazine, 12(3):36–56, (1991).

[34] N. F. Noy and D.L. McGuinness. “Ontology development 101: A guide to creating your first ontology”. Technical Report SMI-2001-0880, Stanford Medical Informatics, (2001).

[35] P Lord, S Bechhofer, MD Wilkinson, G Schiltz, D Gessler, D Hull, CA Goble, and L Stein. “Applying Semantic Web Services to Bioinformatics Experiences Gained, Lessons Learnt”. In Proceedings of the 3rd International Semantic Web Conference, volume 3298 of LNCS, pages 350–364. Springer-Verlag, (2004).

[36] A. Herzog, N. Shahmehri, and C. Duma. An ontology of information security. International Journal of Information Security and Privacy, 1(4):1--23, October-December (2007).

[37] MC. Daconta, LJ Obrst, and KT Smith. The Semantic Web: A Guide to the Future of XML, Web Services, and Knowledge Management. Wiley, (2003).

[38] D. Trastour, C Bartolini, and C Preist. Semantic Web Support for the Business-to-Business E-Commerce Lifecycle. In Proceedings of the 11th International Conference on World Wide Web, pages 89–98. ACM, (2002).

[39] C. Pedrinaci, J. Domingue, and A. K. Alves de Medeiros. A Core Ontology for Business Process Analysis. In 5th European Semantic Web Conference, (2008).

[40] T. Berners-Lee, James Hendler, Ora Lassila. The Semantic Web. Scientific American, May (2001).

[41] Resource Description Framework (RDF), http://www.w3.org/RDF/, Last accessed May (2011).

[42] M. Smith, C. Welty, and D. McGuiness," Web Ontology Language (OWL) Guide", August (2003).

[43] S. Decker , Sergey Melnik , Frank Van Harmelen , Dieter Fensel , Michel Klein , Jeen Broekstra , Michael Erdmann , Ian Horrocks, "The Semantic Web: The Roles of XML and RDF", IEEE Internet Computing, v.4 n.5, p.63-74, September (2000).

[44] J. Heflin, J Hendler, and S Luke. "SHOE: A knowledge representation language for internet applications". Technical report, Dept. of Computer Science, University of Maryland at College Par, 1999. Technical Report CSTR- 4078 (UMIACS TR-99-71), (1999).

[45] D. Fensel, FV Harmelen, I Horrocks, DL McGuinness, and PF Patel- Schneider. "OIL: An Ontology Infrastructure for the Semantic Web" IEEE Intelligent Systems, 16(2):38–45, (2001).

[46] D. McGuinness, L., Fikes, R., Hendler, J. and Stein, L. A., " DAML+OIL: An Ontology Language for the Semantic Web". IEEE Intelligent Systems, 17(5): 72-80 (2002).

[47] J.J. Carroll et al., "Jena: Implementing the Semantic Web Recommendations, tech. report HPL-2003-146, Hewlett Packard Laboratories Bristol, (2003).

[48] Jena2. http://www.hpl.hp.com/semweb/jena2.htm, Last accessed May (2011).

[49] B. McBride "Jena", IEEE Internet Computing, July/August, (2002).

[50] Jena Inference, http://jena.sourceforge.net/inference/, Last accessed May (2011).

[51] D. Fensel. Ontologies: Silver Bullet for Knowledge Management and Electronic Commerce. Springer-Verlag, (2001).

[52] T. R.Gruber, "Toward Principles for the Design of Ontologies Used for Knowledge Sharing", Int. J.Hum. Comput. Stud. , 43, 907–928, (1995).

[53] M. Benerecetti, P. Bouquet and C. Ghidini, Contextual reasoning distilled, Journal of Experimental and Theoretical Artificial Intelligence 12 , pp. 279–305 (2000).

[54] C. Ghidini, Giunchiglia F "A semantics for abstraction". In: M´antaras Rde, Saitta L (eds) Proceedings of ECAI'2004, including PAIS 2004, pp 343–347, (2004).

[55] H. Wache, T. Voegele, U. Visser, H. Stuckenschmidt, G. Schuster, H. Neumann, and S. Huebner. "Ontology-based integration of information - a survey of existing approaches", In Proceedings of the workshop on Ontologies and Information Sharing at the International Joint Conference on Artificial Intelligence (IJCAI), pages 108–117, (2001).

[56] J. Euzenat, Shvaiko, P., "Ontology matching" , Springer, Heidelberg (2007)

[57] P. Bouquet, L. Serafini, and S. Zanobini, "Peer-to-Peer Semantic Coordination", Journal of Web Semantics, 2(1), (2005).

[58] F. Giunchiglia, P. Shvaiko, and M. Yatskevich, “Semantic Schema Matching”. in Proceedings of CoopIS’05, volume 3760 of LNCS, pages 347 – 360, (2005).

[59] Giunchiglia, P. Shvaiko, and M. Yatskevich, “Discovering Missing Background Knowledge in Ontology Matching”. in Proceedings of ECAI, (2006).

[60] J. Kwan and Hwan-Sung Yong, “Ontology Matching based on hypernym, hyponym, holonym, and meronym sets in word net” , International journal of Web & Semantic Technology (I West), Vol.1, No.2, April (2010).

[61] J. Li, "LOM: A Lexicon-based Ontology Mapping Tool", Proceedings of the Performance Metrics for Intelligent Systems (PerMIS. '04), (2004).

[62] Flickr Photo Sharing , http://www.flickr.com, Last accessed May (2011).

[63] ZoneTag, http://zonetag.research.yahoo.com/, Last accessed May (2011).

[64] M. Naaman, Nair R, “ZoneTag’s collaborative tag suggestions: what is this person doing in my phone?”, IEEE Multimed 15(3):34–40 (2008).

[65] A. Karypidis and S. Lalis., “Automated context aggregation and file annotation for PAN-based computing”, Personal and Ubiquitous Computing, 11(1):33–44, (2007).

[66] A. Wilhelm , Yuri Takhteyev , Risto Sarvas , Nancy Van House , Marc Davis, “Photo annotation on a camera phone”, CHI '04 extended abstracts on Human factors in computing systems, Vienna, Austria, (2004).

[67] F. Monaghan, O'Sullivan, D, “Automating Photo Annotation using Services and Ontologies”, Proceedings of Mobile Services and Ontologies Workshop, (2006).

[68] M. Ames , Mor Naaman, “Why we tag: motivations for annotation in mobile and online media”, Proceedings of the SIGCHI conference on Human factors in computing systems, San Jose, California, USA, (2007).

[69] A. Russell, B.C., Torralba, A. Murphy, K.P. and Freeman, W.T. “LabelMe: a database and web-based tool for image annotation”. MIT AI Lab Memo AIM-2005-025, September, (2005).

[70] A. Ricardo, Baeza-Yates , Berthier Ribeiro-Neto, “Modern Information Retrieval”, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, (1999).

[71] M. Kobayashi , Koichi Takeda,” Information retrieval on the web”, ACM Computing Surveys (CSUR), v.32 n.2, p.144-173, June (2000).

[72] P. Mohan, Raghuraman, Venkateswaran S and Arul Siromoney, "Semantic File Retrieval in File Systems Using Virtual Directories," in the Poster Session of the 13th Annual IEEE International Conference on High Performance Computing (HiPC), Bangalore, India, Dec (2006).

[73] S. Schenk, Olaf G¨orlitz, and Steffen Staab. “TagFS: Bringing semantic metadata to the filesystem”. In Poster at the 3rd European Semantic Web Conference (ESWC), (2006).

[74] A. Borgy Waluyo, Bala Srinivasan, David Taniar: “Research in mobile database query optimization and processing”, Mobile Information Systems, 1(4): 225-252, (2005).

[75] A. Borgy Waluyo, Bala Srinivasan, David Taniar: “Research on location-dependent queries in mobile databases”, International Journal of Computer Systems: Science and Engineering, 20(2): (2005).

[76] W. Viana, S. Hammiche, B.Moisuc,M. Villanova-Oliver, J. Gensel, and H.Martin. “Semantic keyword-based retrieval of photos taken with mobile devices”. InMoMM, pages 192–199, (2008).

[77] A P. Korpipää and J. Mäntyjärvi, “An Ontology for Mobile Device Sensor- Based Context Awareness,” Modeling and Using Context: in Proceedings of 4th Int’land Interdisciplinary Conf. (Context 2003), LNCS 2680, Springer-Verlag, pp. 451–458, (2003).

[78] N. Weißenberg, A. Voisard, and R. Gartmann. “Using ontologies in personalized mobile applications,” in D. Pfoser and I. Cruz (Eds.), in Proceedings of the Intl. ACM GIS Symposium, ACM Press: New York, (2004).

[79] S. Izumi, Kazuhiro Yamanaka, Yoshikazu Tokairin, Hideyuki Takahashi, Takuo Suganuma, Norio Shiratori: “Ubiquitous supervisory system based on social contexts using ontology”. Mobile Information Systems 5(2): 141-163 (2009).

[80] T. Iwamoto, G. Suzuki, S. Aoki, N. Kohtake, K. Takashio, and H. Tokuda, “uPhoto: A Design and Implementation of a Snapshot Based Method for Capturing Contextual Information”, presented at The Second International Conference on Pervasive Computing , Advances in Pervasive Computing, Linz/Vienna, Austria, (2004).

[81] G. Schreiber, B. Dubbeldam, J. Wielemaker, and B. Wielinga, "Ontology-Based Photo Annotation”, IEEE Intelligent Systems, vol. 16, pp. 66-74, (2001).

[82] M. R. Koivunen, and Swick, R., “Metadata Based Annotation Infrastructure offers Flexibility and Extensibility for Collaborative Applications and Beyond”, in Proceedings of the K-CAP 2001 Workshop on Knowledge Markup and Semantic Annotation, Victoria, British Columbia, (2001).

[83] Izumi, S., Yamanaka, K., Tokairin, Y., Takahashi, H., Suganuma, T., Shiratori, N.: “Ubiquitous Supervisory System based on Social Contexts using Ontology”. Mobile Information Systems (MIS) 5(2), 141–163 (2009).

[84] B. Guo, Satake S, Imai M “Home-explorer: ontology-based physical artifact search and hidden object detection system”. Mobile Inf. Syst. 4(2):81–103 (2008) .

[85] Y. Kalfoglou and M. Schorlemmer. Ontology mapping: the state of the art. The Knowledge Engineering Review, 18(1):1–31, (2003).

[86] P. Shvaiko, Euzenat, J.,”A survey of schema-based matching approaches. Journal on Data Semantics”, IV, 146-171 (2005).

[87] N. F. Noy, “Semantic integration: a survey of ontology-based approaches”, ACM SIGMOD Record, v.33 n.4, December (2004).

[88] E. Rahm , Philip A., “Bernstein, A survey of approaches to automatic schema matching”, The VLDB Journal — The International Journal on Very Large Data Bases, v.10 n.4, p.334-350, December (2001).

[89] G. Stoilos, G. Stamou, S. Kollias, “A string metric for ontology alignment”, in: Proceedings of the 4th International Semantic Web Conference, LNCS, vol. 3729, Springer, pp. 624–637, (2005).

[90] M. Rodr´ıguez and M. Egenhofer. “Determining semantic similarity among entity classes from different ontologies”. IEEE Trans. on Knowledge and Data Eng., 15(2):442–456,(2003).

[91] A. Tversky, “Features of similarity”, Psychological Review 84 , pp. 327–352, (1977).

[92] M. Yatskevich and F Giunchiglia, “Element level semantic matching using WordNet”, in Proceedings of Meaning Coordination and Negotiation Workshop, ISWC, (2004).

[93] N. Asanoma, “Alignment of ontologies: WordNet and Goi-Taikei”, Proceedings of the Second Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL 2001): WordNet and Other Lexical Resources Workshop Program (pp. 89–94). East Stroudsburg, PA: ACL, (2001).

[94] W. Hu, Jian, N.S., Qu, Y.Z., and Wang, Y.B. “GMO: A Graph Matching for Ontologies. K-Cap 2005 Workshop on Integrating Ontologies”, 43—50, (2005).

[95] Y. Qu, Hu, W., and Cheng, G., ”Constructing virtual documents for ontology matching”. in Proceedings of the 15th International World Wide Web Conference (WWW’06), 23–31, (2006).

[96] Y. R. Jean-Mary, E. P. Shironoshita, M. R. Kabuka: “Ontology matching with semantic verification”, Journal of Web Semantics, (2009).

[97] J. Euzenat, A. Ferrara, C. Meilicke, J. Pane, F. Schar e, P. Shvaiko, H. Stuckenschmidt, O. Svab-Zamazal, V. Svatek, and C. Trojahn dos Santos. ,"Results of the ontology alignment evaluation initiative 2010", In P. Shvaiko, J. Euzenat, F. Giunchiglia, H. Stuckenschmidt, N. Noy, and A. Rosenthal, editors, in Proceedings of 5th ISWC workshop on ontology matching (OM), Shanghai (Chine), pages 1-35, (2010).

[98] P. Xu, Y. Wang, L. Cheng and T. Zang, "Alignment Results of SOBOM for OAEI 2010," Ontology Matching, pp. 203, (2010).

[99] F. Isabel, Cruz, Flavio Palandri Antonelli, and Cosmin Stroe., "AgreementMaker: Efficient Matching for Large Real-World Schemas and Ontologies", PVLDB, 2(2):1586–1589, (2009).

[100] J. Noessner and M Niepert., "CODI: Combinatorial Optimization for Data Integration–Results for OAEI 2010". In Proceedings of the 5th International Workshop on Ontology Matching, page 142, (2010).

[101] F. Hamdi, Safar, B., Niraula, N., Reynaud, C. "TaxoMap in the OAEI 2009 alignment contest", In: ISWC Workshop on Ontology Matching, Chantilly (VA US), pp. 230–237, (2009).

[102] W. Hu, Y Zhao, and Y Qu. "Partition-based block matching of large class hierarchies", In R Mizoguchi, Z Shi, and F Giunchiglia, editors, Proceedings of the 1st Asian Semantic Web Conference, volume 4185 of LNCS, pages 72–83. Springer-Verlag, (2006).

[103] J. Bock and J. Hettenhausen, "MapPSO Results for OAEI 2008", The 7th International Semantic Web Conference, Karlsruhe, (2008).

[104] J. Bock, J. Hettenhausen, "Discrete Particle Swarm Optimisation for Ontology Alignment", Information Sciences Article in Press ,(2010).

[105] J. Tang, Li, J., Liang, B., Huang, X., Li, Y. & Wang, K., "Using Bayesian decision for ontology mapping" ,Journal of Web Semantics 4(1), 243–262, (2006).

[106] J. Li, J. Tang, Y. Li, Q. Luo, RiMOM: A dynamic multi-strategy ontology alignment framework, IEEE Transactions on Knowledge and Data Engineering 21 (8), 1218–1232, (2009).

[107] W. Hu, Qu, Y., “Falcon-AO: A Practical Ontology Matching System”, Journal of Web Semantics, 6(3), 237–239, (2008).

[108] N. F. Noy and M. A. Musen, "The PROMPT suite: interactive tools for ontology merging and mapping." International Journal of Human-Computer Studies 59(6): 983-1024, (2003).

[109] P. Wang, Xu, B. LILY: The Results for the Ontology Alignment Contest OAEI 2007. In Proceedings of ISWC 2007 Ontology Matching Workshop. Busan, Korea, (2007).

[110] M. Ehrig, and S. Staab, “QOM: Quick Ontology Mapping” , Proceedings of the 3rd International Semantic Web Conference (ISWC), (2004).

[111] S. Jan, Maozhen Li, G. Al-Sultany, H. Al-Raweshidy, "File annotation and sharing on low-end mobile devices", Seventh International Conference on Fuzzy Systems and Knowledge Discovery (FSKD-10), pp.2973-2977, 10-12, (2010).

[112] Sun Developers Network (SDN), “Java Micro Edition” (now Oracle); (http://www.oracle.com/technetwork/java/javame/overview/index.html). Accessed May (2011).

[113] G. Lawton, “Moving Java into Mobile Phones,” Computer, vol. 35, no. 6, pp. 17–20, June (200).

[114] J. Keogh, “The Complete Reference J2ME”, (chapter 1) - published by Osborne/McGraw-Hill, (2003).

[115] Java Community Process, Community Development of Java Technology Specification (JCP); http://jcp.org/en/home/index. Accessed May (2011).

[116] J. White, " An introduction to Java 2 micro edition (J2ME)", Java in small things Proceedings of the 23rd International Conference on Software Engineering, p.724-725 Toronto, Ontario, Canada, (2001).

[117] A. Isakow, and Shi, H. "Review of J2ME and J2MEbased Mobile Applications", International Journal of Communication and Network Security, Vol. 8 No. 2, pp. 189-198, (2008).

[118] KVM Porting Guide, http://www.mobilejava.co.kr/bbs/temp/portingboard/KVM_porting.pdf Last accessed May (2011).

[119] Connected limited device configuration (CLDC) specification. JSR 139, http://jcp.org/aboutJava/communityprocess/final/jsr139/index.html, Accessed May (2011).

[120] J2ME in a nutshell: A desktop quick reference. Topley K (2002) Nutshell handbook O'Reilly, (2002).

[121] Java Specification Requests, Mobile Information Device Profile (MIDP) JSR 118, http://jcp.org/en/jsr/detail?id=118, Accessed May (2011).

[122] C. Talhi Mourad Debbabi, Mohamed Saleh and Sami Zhioua. "Security Evaluation of J2ME CLDC Embedded Java Platform", Journal of Object Technlogy, 5(2):125—154, (2006).

[123] Java Specification Requests ,(JSR-75 Specifications), http://www.jcp.org/en/jsr/detail?id=75, last accessed May (2011).

[124] Java Specification Requests, Bluetooth Wireless Technology (JSR-82 Specifications) http://jcp.org/en/jsr/detail?id=82, last accessed May (2011).

[125] C. J. van Rijsbergen, "Information Retrieval", Butterworths, London, (1979).

[126] D.A. Buell and D.H. Kraft, "Performance Measurement in a Fuzzy Retrieval Environment", in Proceedings of ACM SIGIR '81, pp. 56-62 (1981).

[127] L. SU, “The relevance of recall and precision in user evaluation”, Journal of the American Society for Information Science, 45, 207-217, (1994).

[128] CLEVERDON, C.W., MILLS, J. and KEEN, M., Factors Determining the Performance of Indexing Systems, Volume I - Design, Volume II - Test Results, ASLIB Cranfield Project, Cranfield (1966).

[129] S. Jan, Maozhen Li, G. Al-Sultany, Hamed Al-Raweshidy and I.A Shah “Semantic file annotation and retrieval on mobile devices” , Mobile Information Systems. vol. 7, no 2, pp. 107-122, (2011).

[130] M. Bhatt, Andrew Flahive, Carlo Wouters, Wenny Rahayu, David Taniar: “MOVE: A Distributed Framework for Materialized Ontology View Extraction”, Algorithmica, 45(3): 457-481, (2006).

[131] A. Flahive, David Taniar, Wenny Rahayu, Bernady O. Apduhan: “Ontology tailoring in the Semantic Grid”, Computer Standards & Interfaces, 31(5): 870-885, (2009).

[132] A. Flahive, Wenny Rahayu, David Taniar, Bernady O. Apduhan: “A Distributed Ontology Framework in the Semantic Grid Environment”, Proceedings of the 19th International Conference on Advanced Information Networking and Applications (AINA 2005), IEEE Computer Society, pp: 193-196, (2005).

[133] M. Paolucci, T. Kawamura, T. Payne, and K. Sycara, “Semantic Matching of Web Service Capabilities,” in Proceedings of First Int’l Semantic Web Conf. (ISWC ’02), pp. 333-347, June (2002).

[134] M. Li, B. Yu, O. Rana and Z. Wang, “Grid Service Discovery with Rough Sets”, IEEE Transactions on Knowledge and Data Engineering, vol.20, no.6, pp.851-862, ISSN: 1041-4347, June (2008).

[135] J. Euzenat, Le Bach, T., Barrasa, J., Bouquet, P., De Bo, J., Dieng-Kuntz, R., Ehrig, M., Hauswirth, M., Jarrar, M., Lara, R., Maynard, D., Napoli, A., Stamou, G., Stuckenschmidt, H., Shvaiko, P., Tessaris, S., Van Acker, S., Zaihrayeu, I.: State of the art on ontology alignment. deliverable 2.2.3, (2004).

[136] I. Levenstein, “Binary codes capable of correcting deletions, insertions and reversals”, Cybernetics and Control Theory, 707-710, (1966).

[137] M. Jaro, M., “Probabilistic linkage of large public health data files” (disc. p687-689).Statistics in Medicine 14, 491–498, (1995).

[138] W. Winkler, “The state record linkage and current research problems”, Technical report, Statistics of Income Division, Internal Revenue Service Publication, (1999).

[139] S.B. Needleman, Wunsch, C.D.,” A general method applicable to the search for similarities in the amino acid sequence of two proteins”. Molecular Biology 48, 444–453, (1970).

[140] W. Cavnar and J. Trenkle, “N-Gram-Based Text Categorization,” in Proceedings of Symp. Document Analysis and Information Retrieval, Las Vegas, pp. 161-169, April (1994).

[141] G. Kondrak, “N-gram similarity and distance”. Twelfth Int. Conf. on String Processing and Information Retrieval SPIRE, pp. 115-126, (2005).

[142] W. W. Cohen, P. Ravikumar, and S. E. Fienberg.,” A comparison of string distance metrics for name-matching tasks” , In IIWEB, pages 73--78, (2003).

[143] D. Lin, “An information-theoretic definition of similarity” , In Proceedings of 15th International Conf. on Machine Learning, Morgan Kaufmann, San Francisco, CA, 296–304, (1998).

[144] A. Budanitsky , Graeme Hirst, “Evaluating WordNet-based Measures of Lexical Semantic Relatedness”, Computational Linguistics, v.32 n.1, p.13-47, March (2006).

[145] S. Greco, Matarazzo, B., S_lowi´nski, R., “Rough sets theory for multicriteria decision analysis” , European J. of Operational Research 129(1), 1–47, (2001)

[146] Z. Pawlak, “Rough sets”, International Journal of Information & Computer Sciences 11, 341±356, (1982).

[147] N. Shan, W. Ziarko, H. J. Hamilton, and N. Cercone, “Using rough sets as tools for knowledge discovery,” in Proceedings of 1st Int. Conf. Knowledge Discovery Data Mining, U. M. Fayyad and R. Uthurusamy, Eds. Menlo Park, CA, pp. 263–268, (1995).

[148] Z. Pawlak, Rough Sets: Theoretical Aspects of Reasoning About Data. , Kluwer Academic, Dordrecht, The Netherlands (1991).

[149] Y. Yao, “ A note on definability and approximations”, Transactions on Rough Sets, pp. 274-282, (2007).

[150] T.R. Gruber, “A Translation Approach to Portable Ontology Specification”, Knowledge Acquisition 5: 199-220, (1993).

[151] J. Euzenat, “An API for ontology alignment”, In McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) ISWC 2004. LNCS, vol. 3298, pp. 698–712. Springer, Heidelberg (2004).

[152] Makhoul, John, Francis Kubala; Richard Schwartz; Ralph Weischedel, “ Performance measures for information extraction” in Proceedings of DARPA Broadcast News Workshop, Herndon, VA, February (1999).

[153] H. H. Do, S. Melnik, and E. Rahm. Comparison of schema matching evaluations. In Proceedings of the workshop on Web and Databases, (2002).

[154] S.v´ab-Zamazal, V. Sv´atek, and F. Scharffe, “ Pattern-based Ontology Transformation Service” , in Proceedings of the 1st International Conference on Knowledge Engineering and Ontology Development, (2009).

[155] A. Giovanni, Modica , Avigdor Gal , Hasan M. Jamil, “The Use of Machine-Generated Ontologies in Dynamic Information Seeking” , in Proceedings of the 9th International Conference on Cooperative Information Systems, p.433-448, September 05-07, (2001).

[156] A. Gal, G. Modica and H. Jamil. OntoBuilder, “Fully Automatic Extraction and Consolidation of Ontologies from Web Sources”, In ICDE Conference, (2004).

[157] P. Lucian, Velegrakis, Yannis, Miller, Renee J., Hernandez, Mauricio A., and Fagin, Ronald “Translating web data”. In Bernstein et al. , pages 598-609, (2002).

[158] D., McDermott, Qi, P., “Ontology translation by ontology merging and automated reasoning”, In EKAW’02 workshop on Ontologies for Multi-Agent Systems. Sigüenza, Spain, (2002).

[159] N. Noy. & Musen, M., “Ontology Versioning in an ontology-management framework”, IEEE Intelligent Systems, 19(4), 6–13, (2004).