PROGRESS REPORT
Ontology-Based Technical Document Retrieval System

Ruvin Yusubov

Supervisor:
Professor Ulrike Sattler

MSc Advanced Computer Science
School of Computer Science
The University of Manchester
May 9th, 2014
# Table of Contents

List of Abbreviations .......................................................................................................................... i  
List of Figures ................................................................................................................................... ii  
Abstract ........................................................................................................................................... iii  

1. Introduction ....................................................................................................................................... 1  
   1.1. Project aims and objectives .............................................................................................. 2  
   1.2. Report Outline .................................................................................................................. 2  

2. Background ....................................................................................................................................... 4  
   2.1. Knowledge Organization Systems ................................................................................... 4  
   2.1.1. Controlled vocabulary .............................................................................................. 4  
   2.1.2. Taxonomy ................................................................................................................ 4  
   2.1.3. Thesaurus ................................................................................................................. 4  
   2.1.4. Ontology and Multi-dimensional modelling ............................................................ 4  
   2.2. Faceted search .................................................................................................................. 7  
   2.3. Information Retrieval System in general .......................................................................... 8  
   2.3.1. Query Language ....................................................................................................... 8  
   2.3.2. Metadata ................................................................................................................... 8  
   2.3.3. Query Answering Process ........................................................................................ 9  
   2.3.4. Evaluation metrics of Information Retrieval System ............................................. 10  
   2.4. Syntax-based approaches and models ............................................................................ 10  
   2.4.1. Boolean model ....................................................................................................... 11  
   2.4.2. Vector Space model ................................................................................................ 11  
   2.4.3. Limitations of syntactic approaches ....................................................................... 11  
   2.5. Semantic-based approaches ............................................................................................ 12  
   2.5.1. Improving Standard Approaches ............................................................................ 12  
   2.5.2. Ontology-based retrieval ........................................................................................ 13  

3. Research Methods and Project Planning ................................................................................. 16
# List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Full Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>OWL</td>
<td>Web Ontology Language</td>
</tr>
<tr>
<td>RDF</td>
<td>Resource Description Framework</td>
</tr>
<tr>
<td>RDFS</td>
<td>Resource Description Framework Schema</td>
</tr>
<tr>
<td>XML</td>
<td>Extensible Markup Language</td>
</tr>
<tr>
<td>IR</td>
<td>Information Retrieval</td>
</tr>
<tr>
<td>IRS</td>
<td>Information Retrieval System</td>
</tr>
<tr>
<td>DRS</td>
<td>Document Retrieval System</td>
</tr>
<tr>
<td>TD</td>
<td>Technical Document</td>
</tr>
<tr>
<td>DITA</td>
<td>Darwin Information Typing Architecture</td>
</tr>
<tr>
<td>DL</td>
<td>Description Logic</td>
</tr>
<tr>
<td>QL</td>
<td>Query Language</td>
</tr>
<tr>
<td>UI</td>
<td>User Interface</td>
</tr>
<tr>
<td>QAP</td>
<td>Query Answering Process</td>
</tr>
<tr>
<td>BK</td>
<td>Background Knowledge</td>
</tr>
<tr>
<td>QAP</td>
<td>Query Answering Process</td>
</tr>
</tbody>
</table>
List of Figures

Figure 2.1: Layers of three different ontology languages ................................................................. 5
Figure 2.2: A picture showing a possible hierarchical structure of documents, and one with applying
the dimensions separated out ............................................................................................................ 7
Figure 2.3: Set of components in standard information retrieval system......................................... 9
Figure 2.4: Information retrieval system without generating metadata (full-text Search) .............. 10
Figure 2.5: Information retrieval system with background knowledge ........................................ 13
Figure 3.1: Gantt Chart for the dissertation schedule ...................................................................... 18
Figure 4.1: Top-level classes of “Toy Ontology” ............................................................................... 20
Figure 4.2: Overall design of the system (including OWL ontology) ............................................. 21
Figure 4.3: Fragment of user interface of N8 browser with one selected and one related item ...... 23
Figure 4.4: Multi-faceted browser of the system, with selected “To Manual Installation” item .... 23
Abstract

The main task of an Information Retrieval System (IRS) is to be able to retrieve relevant information resources. There are many different approaches about how to perform this task. Nowadays, most of IRSs are based on keywords, and we call such approaches syntax-based. However, using these approaches we do not capture the meaning of either keywords or content of information resources. Another kind of approaches in which IRS tries to capture the meaning of these things we call semantic-based. These semantically motivated IRSs usually make use of some Background Knowledge (BK). One of the forms in which BK can be presented is an ontology.

In this project, we will provide a prototype/proof-of-concept of the IRS with the BK in form of an OWL ontology. The system would retrieve Technical Documents (TDs). These documents come from Ericsson equipment manufacturer, and noteworthy, because they are structured and possibly overlapping. In the project, we will investigate and present design choices, which are used to build the ontology as well as the whole system. The OWL ontology is modelled in multi-dimensional way and DITA concepts and ideas will be used to represent the TDs. This kind of modelling will be exposed to the user via a multi-faceted browser. The multi-faceted browser tries to simplify the process of formulating queries where both the domain is complicated as in the case of TDs and queries should be passed to the system in such complex form as OWL class expression.

Finally, an evaluation of the system will be conducted by a different set of parameters, which are inherent to IRSs (performance, scalability etc.). In addition to this, system will be evaluated by twenty queries, which we will get from the Ericsson.
1. Introduction

Nowadays, more and more documents become available for an increasing number of users. Thus, it arises a need to retrieve and provide relevant documents, out of some heap, for a user request. The document can be described as a relevant if it fits user’s needs. The software or the system that gives us an opportunity to get the information needs we call Information Retrieval System (IRS). There are some domains where the issue of retrieving relevant documents plays a significant role. For example, research papers and articles in the domain of biology and medicine. This problem also occurs in the domain of Technical Documents (TDs). For instance, documents which are about telecommunication devices.

There are different kinds of approaches, which are involved in the process of the retrieving relevant documents. Some of them are purely syntactic and based a lot on keywords, other more semantic. The syntax-based approaches are usually performed by finding the keywords according to their spelling in the document or document’s textual representation. This idea of retrieving documents is the most popular nowadays. However, such idea may results in the problems of retrieving relevant documents, because sometimes keyword in a document does not necessary tells that document is relevant. In addition, relevant document may not contain the explicit word, which user can use in formulating query. In case of semantic approaches more important a meaning of the word and not just, how it is spelled. Semantic approaches try to overcome these issues. For instance, synonyms and recognize homonyms of words can help to improve recall and precision respectively.

There is also another common problem in IRSs. Sometimes it is not easy for an ordinary user to use IRS based on commonly used keywords. The user may have problems in expressing his/her information needs and translating these needs into system’s query. This issue can occur because it is difficult to express information needs in the terms, which are used by the system in particular for the systems whose domain is complicated and semantically unknown to some extent. TDs can be an example of such domain.

Thus, there are different ways to hold information retrieval process. In this project, we will make use of background knowledge in form of OWL ontology to support semantic-based approach for TDs. The ontology is used to describe domain knowledge as well as represent TDs. These TDs are from Ericsson equipment manufacturer, about some specific devices, used for some set of tasks, etc. It means that TDs are structured. Other feature, which inherent to most of TDs, and makes them more noteworthy is that they are possibly overlapping. It means that TDs can be buildup of building blocks that possible shared between different documents.

It will also be shown how the proposed modelling of ontology assist in providing to the user a multi-faceted browser to formulate a query, which tries to solve the abovementioned problems of formulating the query. Multi-faceted browser has also been proposed to simplify the process of formulating query that should be done in
complex form such as OWL class expression. Finally, our approach will be evaluated by different set of parameters specified in the Section 3.4.

Thus, the motivation of the project is to show how ontology-based approach can be used and useful in order to build retrieval system for TDs.

1.1. Project aims and objectives

The aim of the project is to build a prototype of an ontology-based TDs retrieval system where design choices need to be consider and applied carefully. Using them, we should maintain different evaluation metrics in an appropriate way. This retrieval system will be with an interface in form of multi-faceted browser. In order to achieve these aims there is a list of tasks and activities need to be done:

1) Research different kinds of IRSs (traditional and with different kinds of background knowledge).

2) Good level of understanding OWL, in order to build the ontology with specific design choices. In order to manage OWL ontology through the programming language, OWL API should also be studied.

3) Understand and apply multi-dimension modelling to present an interface of the system in form of the faceted browser. For this purpose, we will also research N8 browser to get the main ideas of design and implementation. N8 browser is an open source application whose user interface presented in form of multi-faceted browser that based on OWL ontology and thus can serve as a good start.

4) Considering that, our domain is about TDs, whose contents can overlap, DITA ideas need to be understood and used to represent TDs in the ontology. This common standard supposes ideas of how to represent documents with mentioned about them features.

1.2. Report Outline

The report is organized in five chapters: Chapter 2 includes both project preliminaries and works related to the subject area. This begins by presenting an overview of different forms of background knowledge including multi-dimensional modelling of an ontology. Next, faceted search is described. Further section gives a description of components out of which standard IRS consist. Last sections cover different kinds of approaches that are used in IRSs. Most of them presented with examples of other people’s work. Chapter 3 mainly demonstrates research methodology that is used in order to accomplish different tasks of the project as well as project plan, deliverables, and evaluation plan. Chapter 4 shows the current progress in the project. It carried by presenting the design considerations that are taken into account to build the ontology and system as a whole. In addition, the current implementation state is presented there.
Finally, brief summary of what is already done and which tasks still need to be completed indicated in chapter 5.
2. Background

In this chapter, we describe the project preliminaries and terminology as well as literature review of other people’s works related to the subject area.

2.1. Knowledge Organization Systems

Some IRSs use background knowledge, which is why here we review their different forms.

2.1.1. Controlled vocabulary

A controlled vocabulary can be described as a list of predefined terms [34]. These terms are usually used to describe a specific domain. The following knowledge organization systems use controlled vocabulary terms that are organized in different ways.

2.1.2. Taxonomy

A taxonomy allows related terms to be grouped together and categorizes them through hierarchical relations [35]. It means that taxonomy organizes terms and relations on these terms in form of hierarchy, or in a tree structure.

2.1.3. Thesaurus

A thesaurus in its turn can be described as a taxonomy with more semantic relations between the terms [35]. Synonyms and hyponyms can be considered as examples of such relations. For instance, the popular thesaurus WordNet [41] groups different English words into sets of synonyms. These sets are connected to each other through a number of semantic relations. For example, the words, “dog” and “canine” from different sets can be connected via hypernym relation.

2.1.4. Ontology and Multi-dimensional modelling

An ontology refers to the representation of the terms and relations between these terms in a format that will be machine understandable [35]. “Ontologies have the potential of modelling information in a way that they can capture the “meaning” of the content by using expressive knowledge representation formalisms” [30]. In this thesis, the term “ontology” will be used as it is adopted in artificial intelligence: “An explicit specification of a shared conceptualization” [20]. “Conceptualization” refers to the abstract model of some subject area. “Shared” means that an ontology tries to give a single and concrete representation of an entity in the domain. “Specification” in its turn tells about the transformation of this shared conceptualization into a formal representation language.

Nowadays, there are different ontology description languages, that including RDF, RDFS, OWL, DAML, etc. As shown in the Figure 2.1, new ontology language usually builds on top of previous one [31]. For instance, RDFS was created as an extension of
Background

RDF vocabulary. OWL, which is based on Description Logics (DL) in its turn, includes all the features of RDFS and provides even larger vocabulary than in RDFS.

For the reason that we have chosen for this project OWL, which is standardize one, most of terminologies and features about ontologies given further relevant just to OWL. We assume the reader is familiar with OWL that is why will not go deep with describing all the notions such as reasoner and how it works, DL, all types of OWL axioms, etc. [38], [39], [40].

Ontologies formally represent domain knowledge as a collection of entities and axioms, using a shared vocabulary. Entities in the ontology are mainly presented as classes, individuals and properties:

1. Classes form a class hierarchy. For example, class “Language” in an ontology can contain the following subclasses: 1) Easy Language 2) Complex Language.

2. Individual is an instance of a class. Example of an instance for “Language” class can be “English”. Sometimes it is difficult accurately distinguish what should be an individual and what a class [17].

3. Properties are used to connect abovementioned entities. For example, if we have two individuals “John” and “Peter”, then the relationship “isFriendOf” between these two individuals can be applied in the following way: “John isFriendOf Peter”.

Finally, axioms connect and show relations between all the aforementioned entities. These axioms divided into T-Box and A-Box. T-Box axioms describe a domain, as set of classes and properties over these classes. An example of T-Box axiom can be the following statement: “All mammals are animals” (i.e. SubClassOf axiom), where both mammal and animal are classes. A-Box axioms are the statements about individuals belonging to these classes. For instance, “John is a Person” (i.e. ClassAssertion), where John is an individual and Person is a class. Combination of all T-Box and A-Box axioms in an ontology make up a knowledge base. These axioms can be asserted explicitly or inferred by a reasoner. Reasoner can be described as piece of software that able to infer logical entailment from a set of asserted axioms. Obtaining all the ancestors of the
specified class can be an example of a simple inference, under the condition that the parent classes are specified. In order to specify some query over OWL ontology we can write DL query (in some standard syntax), which take as input an OWL class expression. As a result of the query, we will obtain all the classes and individuals that are corresponded to the specified class expression.

According to above-mentioned theory of ontology, can be inferred that a taxonomy is a special case of an ontology, where we have only “is–a” relationships between a child node and its parent. In fact, even thesaurus can be implemented with a use of ontology.

Modelling an ontology can be done in different ways. We can apply different strategies and design patterns to conduct it. One of the design choices presented in [9] is modelling an ontology in multi-dimensions. The main idea of such modelling can be described with an example demonstrated in Figure 2.2. Assume that we want to model documents about animals. One of the possible ways to perform it is “trying to squeeze all the given aspects” [9] (types of documents and animals, location of animals, etc.) of the Document and Animal classes into a single hierarchy as it is shown on the left side of Figure 2.2. Therefore, with such modelling, we have just one dimension and we only model the documents. It means that we have documents about animals, about their habitant, etc. The result of such modelling is that we do not have domain of animals modelled independently. We have it just implicitly modelled in our document hierarchy. Modelling all these in multi-dimensions way will change the picture as it is on the right side of Figure 2.2. Here we have rather high-level classes (high in the hierarchy) or parallel taxonomies for different dimensions. So, if our ontology is about documents then rather having one big ontology where everything is a document, we have small document ontology, but also model other dimensions (animals, location of animals, anatomic structure of animals, etc.) that are used to describe properties of the documents. Finally, we use the properties to link separate dimensions.

OWL supports this kind of modelling in a great way. Using OWL, we can model different dimensions along with the relations between them and finally make use of reasoners to put all the dimensions back together again. “We can describe classes and individuals in terms of where they sit in this multi-dimensional space” [9] via complex class expressions. It means that we do not need to introduce names for all the possible combinations of classes. Assume that we have Manual Installation and Printing Device OWL classes. Then in order to describe an individual with type of some class (i.e. ClassAssertion) from Document dimension which is about manual installation of printing device, we do not need do define Manual Installation Of Printing Device class, but can use the following class expression: Document and usedFor some Manual Instalation and about some Printing Device.

Such feature leads to the idea that multi-dimension modelling with OWL can be exploited by post-coordination [32]. There are also some other features of multi-dimensional modelling that are described in [9].
2.2. Faceted search

Faceted search or faceted browsing is a technique for accessing (searching) information; the following description follows [7]. Information here organized in a way that any information element can be described through multiple dimensions, called facets. Each facet in its turn can be described as a sub-taxonomy, which is formed by some characteristic of division. For instance, in the domain of animals this division can be done by habitat, by prey, by food, etc. It can be noticed that facets are similar to the dimensions that are built in the multi-dimensional modelling of an ontology.

Faceted search allows us to retrieve a collection of information resources by applying multiple filters. This technique should also give an opportunity to retrieve the same information resource in multiple ways, rather than in a single. Facets may seem very similar to taxonomy. However, the restriction to have a “unique path” [7] for each object/node in taxonomy and which is not observed in faceted browsing shows the key difference between these two.

Many popular retrieval systems such as Google, Yahoo, provide a text field for the user to write the desirable query in form of keywords. Formulating a query in this way is appropriate when the user knows which keywords need to be used to get the information need. However, sometimes the subject area can be sophisticated, and it leads to
difficulties for the user to specify a query. Thus, arises the problem to think about which keywords better to use in order to get desirable results. Using multi-faceted search instead of standard text field in such subject areas can be easier due to the idea of getting the same information resource in different ways. User can select the items just from the facets whose content is more or less understandable to him/her.

2.3. Information Retrieval System in general

Information Retrieval (IR) can be described as an activity of getting information resources, which satisfy user’s needs [8]. One of the illustrative examples of IR applications is web search engines. In general, an information retrieval system is a software program that stores, manages and provides information according to the user’s request. The content of information resources provided by the system can be in different forms (e.g. text, video, audio, image).

If the information resources obtained by the system satisfy user’s needs then they are called relevant. A better retrieval system would retrieve more relevant information resources and fewer irrelevant ones.

There are some building blocks out of which standard IRS consist. From system to system, this picture can change. It means that some building blocks may not be included or replaced by others. We grouped some relevant for this project building blocks together into components with the following labels (Figure 2.3):

1) Query Language (QL)
2) Metadata
3) Query Answering Process (QAP)

2.3.1. Query Language

A user formulates a query in a given by the system QL, possibly supported by a suitable User Interface (UI) of the system. Apart from providing a capability to formulate a query, UI is used to observe all the answers that system returns for the user’s query. Examples of the systems with different type of QL will be demonstrated in the subsequent sections of this chapter.

2.3.2. Metadata

Metadata is a representation of an information resource in a way that indicates what the resource is about, but in a more compressed form than original information resource. Metadata helps to optimize the query performance and considerably improve respond time w. r. t. full-text search. The process of generating metadata can be done manually or automatically. In this context “manually” means that people provide metadata by themselves based on their perception of resource’s content. “Automatic” generating metadata can be done by applying different specific tools. For instance, if we want to
generate metadata for textual document, there are available text-mining tools\(^1\). One of the popular form of metadata is index terms. They are used as keywords to summarize the content of an information resource.

2.3.3. Query Answering Process

As shown in Figure 2.3, QAP is a process that takes as an input information from different components and results in providing the required information resources. QL determines the form of user’s query that QAP can take for further proceedings. Thus, the essential input for QAP is a query in the form of QL of the system. This input data afterwards is used by QAP to find all the relevant information resources. Metadata of information resources is another standard input for QAP. Based on these inputs, searching, which is a part of QAP, is performed. The main process of searching is to match user’s query against the specified metadata.

In QAP can also take place different processing tasks such as stemming, matching to the thesaurus, etc. These tasks are also usually applied in the phase of generating metadata out of information resources.

One central problem regarding such retrieval systems is the issue of predicting which documents are more relevant and which are less. Such decision strongly depends on the ranking algorithm, which attempts to establish a simple ordering of the retrieved documents. In this report, ranking will also be considered as a part of QAP.

---

\(^1\) http://www.predictivanalytics.com/top-30-software-for-text-analysis-text-mining-text-analytics/

---

Figure 2.3: Set of components in standard information retrieval system
2.3.4. Evaluation metrics of Information Retrieval System

There are different measures to evaluate IRS. Precision and recall are the most predominant measures for retrieval accuracy. “Precision is the fraction of the retrieved documents which is relevant” [11]. “Recall in its turn is the fraction of the relevant documents which has been retrieved” [11].

Based on how easy and comfortable it is for the user to formulate queries in the system, IRS can also be evaluated by queriability parameter [30]. Performance and scalability are two other common evaluation metrics for IRS. Performance refers to the time needed to get answers for the specified query. Scalability is about number of information resources that can take place and handled by the IRS within a reasonable time.

2.4. Syntax-based approaches and models

Traditional approaches in IR, especially in Document Retrieval (DR) are syntax-based approaches. DR can be considered as a branch of IR where the information resources primarily in the form of text. One of the popular and basic approach here is full-text search. In this approach, all the words of each stored document match the query specified by the user. On the Figure 2.4 can be noticed that this approach do not contain any kind of metadata component and searching is done by matching user’s request to the whole document.

![Diagram of Information retrieval system without generating metadata (full-text Search)](image)

Figure 2.4: Information retrieval system without generating metadata (full-text Search)

However, to make retrieval process more efficient, documents or any other kind of information resources are not involved directly in QAP. Instead of it, they are represented in the system by their surrogates or metadata. Traditional approaches based on the metadata, which in the form of keywords. A keyword usually a word in natural language. For example, for textual document it can be a word, which appears mostly in
the document. User’s query can also be described as a list of keywords, as it is in popular search engines (Google, Yahoo, etc.). Eventually, keywords from user’s query, match against the keywords that represent a document.

There are many different retrieval strategies or models that are used in IRSs. They represent components of Figure 2.3 in different ways. Despite the fact that there are many models, only some of them traditional. Based on the traditional models new models are proposed until nowadays [1], [2].

The following two subsections will describe two of the most basic and traditional models [10].

2.4.1. Boolean model

This model does not consider ranking, and relates to the exact matching models. It means whether documents are or are not retrieved. Therefore, QAP does not contain any ranking sub-process. In this model query can be specified with using logical operators such as AND, OR, NOT. Therefore, QAP of the system with such model can take a query with QL in form of Boolean expressions. Metadata of the information resources in this model presented as a list of keywords. Hence, the query “clever OR beautiful” should provide all the documents whose metadata contains either clever or beautiful keywords, or both.

2.4.2. Vector Space model

In this model, metadata of source document is also usually in form of keywords. However, both metadata of documents and QL are presented as vectors in the term space. It means that each dimension corresponds to a separate term. This method promotes ranking which is conducted in QAP and thus more relevant documents should be at the top of all retrieved documents. The model proposes framework in which partial matching is possible. Such idea is realized due to the non-binary weights of index terms. In classic variant of the model proposed by Salton [12] weights are calculated by applying term frequency-inverse document frequency method. Then in order to make ranking a geometric similarity measure between document and query used. Similarity measure can also be realized in different ways. For example, one way is to calculate cosine of the angle between the metadata and query vectors.

2.4.3. Limitations of syntactic approaches

As we might have noticed these standard approaches with or without metadata based on matching the keywords just according to their spelling. Such approach in general may result to bad precision because document can contain some words, which do not relate to its content. Recall can be decreased, because sometimes even relevant document may not contain keywords, which user indicate as a part of his/her query. Syntax-based approaches at least contain the following drawbacks:

- “Synonyms, homographs, homophones and inflection of words can all fool algorithms which see search terms only as a sequence of characters” [3].

Ontology-Based Technical Document Retrieval System
• Semantic relations and taxonomical information of things cannot be captured.

• As in the case of synonyms of words, time can also be specified in different ways. If we search documents about the “XX century” using exactly this phrase, relevant resources containing the characters sequence like “1945” or “1956” will not be found by simple keyword matching” [3].

From user perspective, these drawbacks can lead to the problem in formulating query. The user starts to think which keywords better to use in formulating query in order to get his/her information needs.

2.5. Semantic-based approaches

In the following two subsections, we will observe how different kinds of Background Knowledge (BK) can be used and improve IRS.

2.5.1. Improving Standard Approaches

As was shown in previous section, QAP of standard approaches based a lot on keywords and thus acts on word-level and do not capture meaning of the keywords.

Abovementioned models are often improved with additional mechanisms. These mechanisms help to achieve better results. Stemming can help to find the root of the word through morphological relations of the words. For example, “if the query term running is used with use of stemming tools” [5] we can force QAP, especially part of it which is responsible for finding and matching the keywords, to search for run, runner etc.

Use of taxonomy and thesaurus helps to make standard keyword-based retrieval more robust. This idea involves a keyword expansion using BK. For example, the same term running can be expanded with two new keywords such as sprinting and jogging and therefore recall should be improved. The use of the BK results with a new component in Figure 2.3, which connected with QAP and can be demonstrated as it is in Figure 2.5

In [22] WordNet thesaurus is used to improve IRS, which uses standard Boolean model. Thesaurus and stemming are used here over queries and metadata, which are in form of keywords. This work also describes experiments of how adding WordNet thesaurus improves the system w. r. t. precision, recall and other measures.

However, keyword expansion using thesaurus still has some valuable drawbacks some of which presented in [13]. For instance, synonym expansion may actually decrease precision. An example for it can be word bark or any another word which has more than one certain meaning (homonym). All synonyms of such keywords will be used in the expansion, which leads to retrieve many irrelevant documents.

There are also other more sophisticated methods to expand a set of query terms. Usually they based on keywords co-occurrence statistics and their semantic relations [14], [15], [16].
2.5.2. Ontology-based retrieval

A lot of research has been done in applying BK in form of ontology for IRS. However, different systems make use of it in different ways.

As it was with taxonomy and thesaurus, one of the popular ideas is to make use of ontologies for the process of expanding queries.

The project presented in [23] uses the ontology in a way that it assists in expanding a query. A user formulates the query in the QL that is in form of keywords. The metadata of resources is also in form of keywords. QAP works in a way that keywords specified in QL afterwards is processed by different Natural Language Processing (NLP) mechanisms. As a result of these processes the query, especially the keyword stems are converted to the class names that are found in the ontology. Next step which take place in QAP is an expansion process, which is done by adding subsuming and even superseding classes (with a decreased weight factors) of the class that was a part of the query. The weight factor estimates the relevance of the term. Finding subsuming and superseding becomes possible due to the fact that classes were sorted by category (high level class) in a hierarchical way. Eventually, the list is obtained by the process described above will be matched against the metadata.

The method proposed in [26], apart from inferred class hierarchy, uses and defines other semantic relations between the entities of ontology to expand the user’s query. QL here has form of both keywords and Boolean expressions. Therefore, QAP in its turn apart from a list of keywords, as it was in [23], can also take a query in the form of Boolean expression. Metadata in the same form as it was in the previous approach. The number of keywords for the query maximum of two. QAP maps each of the keywords of the
query to the single term in the ontology. However, due to the fact that ontology was written in OWL, there are different type of entities that can reflect the keyword. Based on the type of entity or entities (if the number of terms were two) nine different combinations are possible. For instance, if the keyword of the query corresponds to the individual in terms of OWL, then the query will be expanded by adding all the individuals, which are strongly related to the given one via A-Box axiom.

Work presented in [29] shows other ways of using ontology in IRS. Although QL here is in the form of OWL class expressions, UI of the system supports to formulate queries in the form of keywords. For the system presented in this work, which was used to query for information about the football matches, ontology already contains all the necessary information/data to provide the answer for the user query. It was done with representing XML files, which contained all the information about football matches, into correspondent OWL files. These OWL files afterwards connected through a Protégé server. Each object from XML files represented as individual and relationship with other individuals represented in form of A-Box axioms. Thus, we can tell that metadata of the objects which will be retrieved is in form of A-Box axioms. Moreover, the experiment conducted in the study shows that the method proposed by them surpasses the simple query expansion within the same domain, background knowledge and information resources.

The image retrieval system described in [4] shows other features which BK in form of ontology can bring. One of the distinctive features w. r. t. the works described above is that user can use faceted browsing to specify a query. Such facets are the reflection of class hierarchy that was specified in RDFS ontology. Apart from class hierarchy, instances of these classes are also used in the facets. They occur only as leaves of the hierarchy and under the class to which they belong. Another noticeable moment is a way of providing metadata for information resources. For each image file, correspondent instance of “Image Element” class was created in the ontology. These instances of images are associated with a set of other instances of the same ontology. For example, if we have facets “Event” and “Persons” and image is about “Divine Service” event and “Linus Torvalds” then this image will be associated with at least two individuals (divine service and Linus Torvalds). The association or metadata in this system is provided in form of RDF triple. So, clicking on the “Linus Torvalds” item in the “Persons” facet will make a query and based on the provided metadata, will retrieve desirable image. Moreover, interesting feature of this work is to make use of ontology to recommend images, which are not necessary match the user’s query. For instance, system can recommend the images where the person specified by the query occurs in image but in the event, which differ from the event indicated by user.

Ontology-based approach can also be merged with some syntax-based approaches in order to increase precision. One of the popular ideas is to make use of classic vector-space model together with ontology. As an example in [18] classic vector-space model including annotation weighting algorithm used in order to provide ranking. The method uses weighted annotation when associating documents with ontology instances. VICODI project [3], [19] is another example, which combines usage of ontology with
vector-space model. With use of keywords to formulate query, QAP works in a way that system first does full-text search based on the labels specified for each entity in ontology. It brings feature that, at the moment when user types some text system provides all the possible entities in ontology with such characters. Such feature means that ontology can be used to disambiguate queries.
3. Research Methods and Project Planning

3.1. Development Process

The project aims to design, implement and analyze/evaluate an application prototype of an ontology-based retrieval system for TDs. It is necessary to define a process that helps us to understand requirements and steps that should be done in order to accomplish these tasks. The process described further is specific to the project and was supposed out of discussion with supervisor. Such process helped to start implementing and experimenting different aspects of the project earlier. Four phases were defined and addressed below, where three (2-4) of them iterative and together form an iteration. It is also noticeable that supposed process has some similarities with a standard iterative and incremental software development [36].

1) **Background research and General requirements acquisition.** In the first phase, research was carried out to get a basic idea about the project background and requirements. It means that main objectives are laid out to give start to the next phase.

2) **Requirements analysis.** In this phase, requirements are identified, addressed and added after discussion about some components of the system with supervisor. It is also possible that some of the original requirements a bit change, and those changes should be added on the next iteration.

3) **Design and Implementation.** This is the phase where most of the activities take place. One of the main activities is implementation of defined requirements. This process based on iterations so that it is intended to implement all requirements in each iteration which can continues for 2-3 weeks. Choosing, analysing and investigating different aspects of design choices also take place in this phase.

4) **Testing, Discussion and Evaluation.** In this phase, the implementation conducted in third phase should be tested as it is defined in Section 3.4. Then, discussions between the student and the supervisor about possible solutions of some tasks and implemented piece of the system take place. Discussion also includes how useful such kind of approach according to different evaluation metrics. Once this phase is finished, we decide whether we will continue with implemented piece or will try a bit another method.

N8 multi-faceted browser and constructed for this system ontology are strongly influenced the development process. This system gave an opportunity to start looking, considering and thinking about different aspects of designing and implementation rather early than it will be with starting from the scratch.
3.2. Deliverables

The main deliverable of the project, besides the dissertation, is to design and build the document retrieval system that is capable to retrieve TDs based on the user query. This system will be considered as a prototype/proof-of-concept rather than a production-level application, in order to understand design choices and to implement the proposed system with the selected technologies. In order to build the system we will need to design and implement an OWL ontology of TDs. Thus, the ontology with some specific design choices will be another deliverable. Finally, an evaluation of the approach by various metrics will be conducted.

3.3. Out of the Scope Issues

The project does not consider how metadata for the document was obtained. It means we do not solve the task of obtaining suitable A-Box axioms from TDs. Thus, to simplify this process and due to the fact that for the project corpus will not contain more than 50 documents, metadata will be provided manually. In addition to this, current system will not be designed to function with accessing by multiple users.

3.4. Evaluation

At the end of the project dissertation, it will need to provide working piece of software, especially ontology-based TDs retrieval system. Considering that the project will use Java as server-side programming language, we will apply JUnit\(^1\) tests to check whether software works correctly or not. An interface of the system will likely be tested with Selenium\(^2\). The main question related to the testing of the user interface is whether the system works correctly for multiple selection of the items of different facets. As was mentioned before, the system will be presented in form of multi-faceted browser. Therefore, in order to retrieve TDs user will need to click/select the item(s) of some facet(s).

Considering the fact that we will build the system for no more than 50 documents, in order to check scalability, we will multiply the metadata (individual with A-Box axioms) of these 50 documents. One of the possible way to achieve it is to loop over existence metadata through the OWL API\(^3\). So, by multiplying the metadata, we can check, how long does the system work for different number of documents and for different kind of queries.

Another question on which our system should give an answer is whether the browser allows us to express a query. It means that if the user has a query and wants to retrieve a certain set of documents can s/he get them? For example, can the user search for all

\(^1\) http://junit.org/
\(^2\) http://docs.seleniumhq.org/
\(^3\) http://owlapi.sourceforge.net/
the documents that are about installation or repair? In order to evaluate the system by this parameter we will ask the member of Ericsson equipment manufacturer to provide us twenty realistic queries with correspondent answers for these queries. Using these queries, we will also be able to check precision and recall of the retrieval system.

3.5. Project Planning

The project dissertation plan has been divided in numerous subtasks following the research method proposed in Section 3.1. The Gantt chart for the overall project plan is defined in the Figure 3.1.

![Gantt Chart for the dissertation schedule](image)
4. Project Progress

4.1. Selection of technology

To develop the logic of required retrieval system and present everything in the web, Java programming language including JSP and Java Servlet technologies are used, currently with Eclipse as an IDE. The creation of ontology is done in OWL, which is the most powerful for today’s time language for authoring ontology. In order to get inferences out of axioms that are specified in OWL, HermiT 1.3.8 reasoner was selected. Protégé\(^1\) ontology editor is used to simplify knowledge representation process due to the graphical user interface. OWL API library is used to manage ontology through the Java programming language. Moreover, to publish the proposed system into Web apart from Servlet and JSP, web-programming technologies as HTML, CSS, JavaScript, JSON and Ajax are used. For example, Ajax is used in order to provide dynamic interface to specify a query and get answers back.

4.2. Ontology Design

For the project, together with supervisor, was decided to build two ontologies. First of them is “Toy Ontology” which is used during the first third of the entire project. This decision was made, for the reason that domain of Technical Documents (TDs) is sophisticated due to a lot of specific terminology what makes it inconvenient for building basic prototype and making experiments. However, “Toy Ontology” already includes some features, which will be used in the target ontology and can be exploited through OWL API.

The domain of “Toy Ontology” is also about TDs. The top-level ontological categories are depicted in Figure 4.1. As can be noticed modelling was done in multi-dimension way, which will also be applied to the target ontology. The classes of the ontology represent documents through the Task, Device and Reader dimensions. The Document dimension connects with these three dimensions via different properties. For instance, class “Repair_Document” from Document connects with class “To_Repair” from Task via “usedFor” property. So, Background Knowledge (BK) of ontology is in the form of T-Box. Apart from class hierarchy of documents, there are also concrete individuals of these classes in the ontology, which associated with target TDs.

As was mentioned before, TDs can overlap. Such feature will be reflected in the modeling of our target ontology. The idea of how such feature can be realized was inspired by DITA framework [37]. The core idea of DITA architecture is that we do not consider all documents as atomic entity, but some of them are made up of textual snippets. We can take these snippets and link them together in a right order and what is more important we can do it as many levels as we want. Last feature will be achieved

\(^1\) http://protege.stanford.edu/
by specifying property *contains* in the target ontology as transitive. Assume that we have *section1, section2* and *section3* individuals, which are building blocks for some TDs. Additionally we asserted that *section2 contains section1* and *section3 contains section2*. Then, if some document *doc1 contains section3* and property *contains* is transitive, everything that is relevant for *section1* it should also be relevant for *doc1*. For instance, if we asserted that *section1* is used for task “A” and specified the query that should provide us all the TDs, which are used for this task, then we should get *doc1*. It can be achieved using the following DL query: “*contains value section1*”. As a result of this query we will obtain *section2, section3* and *doc1* individuals. In DITA, building block of the document calls *topic* and the entire document or compound (contains in itself other *topics or maps*) building block calls *map*. Therefore, while the classes are taken from one of the dimensions, individuals represent either *topic* or *map*.

So, apart from real data that will come from the Ericsson, final version of ontology will use this compositional representation (Lego approach) of documents. As was shown, in our case building blocks can be even in form of sections from TDs.

![Figure 4.1: Top-level classes of “Toy Ontology”](image)

### 4.3. System Design

For clear understanding of the whole picture of the proposing retrieval system, it is helpful to divide our system into components as it is presented in Figure 4.2, which is similar to the Figure 2.5, but more detailed.

**Query Language (QL).** QL of the proposed system is in the form of OWL class expression. It seems not to be easy task for the user to formulate queries in such QL. Due to this fact and considering that the domain of TDs is also complex, the process of
formulating a query is supported by multi-faceted browser. One of the facets here is *Instances* (Figure 4.4) which provides an opportunity to observe all the retrieved documents. Every such document corresponds to the individual of ontology with the type of some class from *Document* hierarchy. Each of the other facets corresponds exactly to one of the top-level classes of the ontology. It means that each facet reflects exactly one dimension of the source ontology. As a result of such modelling, user’s query is done by selecting or clicking on the items from the facets (Figure 4.3, 4.4), where each of the items corresponds to the class name in the ontology.

**Metadata.** As was noticed from Section 4.2, individuals of the *Document* dimension uniquely represent the target TDs. So, describing these individuals similar to the providing metadata for the documents. Each such individual can be associated with a set of classes or individuals of the ontology through A-Box axioms, which characterize the content of the documents. Also, need to be noticed that for the project metadata is provided manually (by annotating all the individuals).

![Diagram](image)

**Figure 4.2: Overall design of the system (including OWL ontology)**

**Query Answering Process (QAP).**

QAP of the system takes an OWL class expression from QL. Then this class expression is used to make a DL query using Manchester syntax. The DL query, in its turn, is carried out through the OWL API. For instance, clicking on the item “To Manual
Installation” from Figure 4.4 will be transformed into the following class expression: “\textit{usedFor some To Manual Installation}”. The result of this DL query is the list of classes and individuals (with the type of some class from Document hierarchy of the ontology). Thus, the items that reflect these classes as well as documents that reflect these individuals should be reflected. According to our example, we will get two such individuals: \textit{doc4} and \textit{doc6}.

Finally, OWL ontology, which was, described Section 4.2, consists of T-Box and A-Box axioms, which form BK and metadata of TDs respectively.

### 4.4. Implementation

Apart from modelling and creating “Toy Ontology” which was described in Section 4.2 some analysis and experiments were done in order to decide which of the two possible implementations take for further development. The idea of the first one, which was used in N8 multi-faceted browser, is to precompute all the related entities (items and instances) for each single item from each dimension and save it as some Java object, in particular as String object (Figure 4.3).

So, selecting any item from the browser after we get the interface of the system will not invoke reasoning anymore. All the queries and reasoning over axioms are done before we get the interface. Of course, such implementation can affect performance if we have thousands of entities. On the other hand, we will feel this drawback only once, especially in the phase of loading the project.

Another method which was applied for “Toy Ontology” and which we plan to use with the target one we call “on the fly”. This method also find all the related items including documents of \textit{Instances} facet. Such name means that only necessary DL queries are conducted based on the selected item and nothing more. This approach should work better in the case when we have big number of documents, as it is in Ericson.

With current progress and implementation, the system provides a multi-faceted browser, with facets indicated in 4.4. The system uses web-design template adopted from N8 browser. A user can select one of the items from one of the facets (excluding \textit{Instances} facet) and get all the related items as well as a list of retrieved documents. Each of these documents, as was mentioned before, should reflect the relevant TD document.
Figure 4.3: Fragment of user interface of N8 browser with one selected and one related item

Figure 4.4: Multi-faceted browser of the system, with selected “To Manual Installation” item.
5. Summary and Further Work

In the Chapter 4, we showed that our current progress includes:

1) “Toy Ontology” that was modelled in multi-dimensional way as well as the extent to which this ontology incorporates the target one. It was also shown the design choices that will be applied to the target ontology.

2) Multi-faceted browser where we can specify a query by single clicking on the item from some facet and get all the related items from the other facets as well as documents that should be retrieved based on the user’s query.

Therefore, according to this progress, the remaining effort on the project will focus on creating the target ontology with design choices mentioned in Section 4.2 and finalizing the design and implementation of the current (“on the fly”) approach of the system. Moreover, it will need to add an opportunity for the user to make multiple selection of the items from the facets. Due to the fact that each selected item results in class expression, multiple selection will be equivalent to the conjunction of the class expressions. Development, testing and evaluation (according to the evaluation metrics and techniques presented in Section 3.4) will continue to follow the process defined in Section 3.1.
List of References


[39] *OWL 2 Web Ontology Language Primer: http://www.w3.org/TR/owl2-primer/*