THE MANCHESTER SUSHI FINDER

Progress Report
COMP60990 Research Methods and Professional Skills
(June 5, 2014)

Name: Al Abbas, Hani Ahmed
Project Supervisor: Sean Bechhofer

School of Computer Science
The University of Manchester

Table of Contents

LIST OF FIGURES

ABSTRACT:

1 INTRODUCTION:

As the trend nowadays to try making machines more intelligent, sharing knowledge of information instead of sharing the raw information in the web is becoming more desirable. Ontologies are considered the main pillar of the semantic web [1]. Semantic web is all about sharing knowledge ,that is understandable for machines, on the web [1]. Creating ontologies is a way to capture and represent our knowledge of the world [1].

Computers do not understand information stored on the web as xml and html. They are just codes to the machine and they display it to users regardless of what knowledge needed. So, ontologies came along for machines to make sense of information. Ontologies are a way to represent knowledge and make inferences from that knowledge using machines computational capabilities and some reasoning techniques such as description logics.

An intelligent way of representing knowledge needs an intelligent way of browsing it. There are a lot of intelligent browsers that is ontology driven user interfaces. Browsing and constructing queries through such user interfaces would be easy and save time, due to the fact that the user interface act as an interactive manual. It eases the process of constructing the intended query since the process itself is guided by the interface. In addition, it saves the user time by displaying only what is the system intended to do. The user does not need to have previous knowledge about the domain, because the explicit display of the options of constructing a query. Ideas like manuals and the help in the menu bar of a user interface would same absolute comparing to the self-guided user interface.

Additional technique to make the user interface smart is to use faceted browsing. The idea behind faceted browsing is to personalize the search and get more specific results by suggesting some filters. Faceted browsing is very related to ontology driven interface since both provide some information about the query while been constructed [2].

There are systems, that are ontology driven user interface, exist such as Transparent Access to Multiple Bioinformatics Information Sources (TAMBIS) and SEmantic Webs and AgentS in Integrated Economies (SEWASIE). TAMBIS is system that gather and analysis bioinformatics information from different sources through one interactive user interface [3]. While SEWASIE meant to access multiple sources of data and help user through constructing exact query needed [4].

The idea of ontology driven interface is not new. In this project, will try to build an application that has some functionalities of interactive interface.

1.1 Project Aims And Objectives:

The aim of this project is to investigate and demonstrate the benefits of using OWL ontologies and OWL API within ontology driven interface [5]. As well as, making the process of checking and testing ontology easier for students by uploading their ontologies. This will be shown by implementing a configurable and flexible user interface, so that most configurations will lay in the ontology file itself; and the interface could browse other ontologies that contain some specific configurations [5]. The application is called the Manchester sushi finder.

To achieve this aim, project was divided into several of objectives. The objectives of this project are:

- Gather project requirements.
- Look at existing applications that are similar in the basic functionalities. Looking specifically at the Manchester pizza finder.
- Increase the reusability of the user interface by making it configurable to suite content of other conceptual models.
- Increase the accessibility of the system by applying filters on the content of the conceptual model or/and on the result of the search query. By introducing the notion of faceted search the access to specific information.
- Increase the accuracy of the system, so users can only construct valid queries and they get the intended results. Making the user interface driven by ontology and using the faceted browsing along with will increase the accuracy of what needed to be queried.

- Provide more flexible system by saving most of the configurations as annotations within the ontology itself.
- Represent the content of the conceptual model with different views such as tree, and list. Users have more one option to view the content of the model.
- Design user stories and tests evaluations.
- Determine the tools that will be used in the project.
- Start implementing and divide and add functionalities one at a time.

1.2 Project Scope:

This project is going to demonstrate the benefits of ontologies and OWL API using an interactive user interface. The functionalities that will be included in the interface are:

1. The user interface will be built using java.
2. User interface will be configurable for generic use.
3. Flexible query building.
4. Preferences of the user will be saved as configuration for reusability.
5. The sushi finder will be a desktop application.
6. The sushi finder should work for any ontology with standard annotations defined in them (Configuration will be stored in the ontology).
7. User can load ontologies one at a time.
8. User can decides which thing to query about (Sushi-Sushi dishes).
9. User can query for specific sushi or sushi dish type based on wanted or unwanted ingredients.

1.3 Report Structure:

The rest of the report is structured as follows:

Background and literature review Section:

This section covers background information on what is needed for this project. It elaborates on OWL ontology, OWL API, ontology-based and faceted-based interfaces, visual querying and some information about similar existing systems.

Research methodology and project plan section:

Progress section:

Summary section:

2 BACKGROUND:

2.1 OWL API:

In Wikipedia [6], Application Programming Interface (API) is described as a set of protocols that make sure the software components interact with each other in the right way, and it could take many forms in different areas. It is used in the web as a set of Hypertext Transfer Protocol (HTTP). Also, it has heavy use as libraries of programming language. API is used in different forms such as libraries of programming languages. For example, Java APIs. In object-oriented languages like java, the API is a set of classes and methods to be accessed and used. Basic examples would be like using the inputting and outputting classes e.g. (BufferedReader and BufferedWriter classes in java). Since this project will be built using java-programming language, the API used is a java API which is called OWL API. OWL API is a set of classes and methods that facilitate the access to Web Ontology Language (OWL) ontologies. It creates objects that represent ontologies objects and manages the interactivity between them and any other program. OWL API consists of two terms, so farther clarification is needed. The following two subsections talk about OWL and OWL API separately.

2.1.1 OWL

OWL is a semantic web language that represents things about the world, group of things, and the relations between them [7]. It is a way to represent knowledge such that it is a representation of the world and our knowledge of it and it is accessible to programs and can be used [8]. It is able to represent explicit and implicit things [7]. The intention behind creating OWL is to be used not only by human but also by applications [9]. OWL can be access from machines because it is based on computational logic so that the machine using some software can reason over them [7]. There are two versions of OWL: OWL and OWL 2 [7]. OWL is a W3C recommendation since 2004, and then OWL 2 was published in 2009, followed with a second edition in 2012 [7, 10]. OWL 2 is just an extension and revision of the original OWL publish in 2004 [7]. OWL has several defined syntaxes including Functional Syntax, RDF/XML, OWL/XML and the Manchester OWL Syntax [7, 10].

The information on the web was described by OWL working group [9] as scattered. This information could mean something for humans but not for machines. So, the

semantic web gives explicit meaning for this information. As a result, integrating and processing the information would be easier for machines. There are three sublanguages of OWL, and each one suits some needs. They are just subroutine of the full OWL. The three languages are: OWL Lite, OWL DL, and OWL Full.

In 2012, OWL 2 has been introduced by OWL working group [11]. It is not different than OWL, it could be seen as an extension of OWL with some additional features. OWL 2 have several syntaxes and semantics, usually a developer needs only one syntax and one semantic. Figure 1 shows the structure of OWL 2.

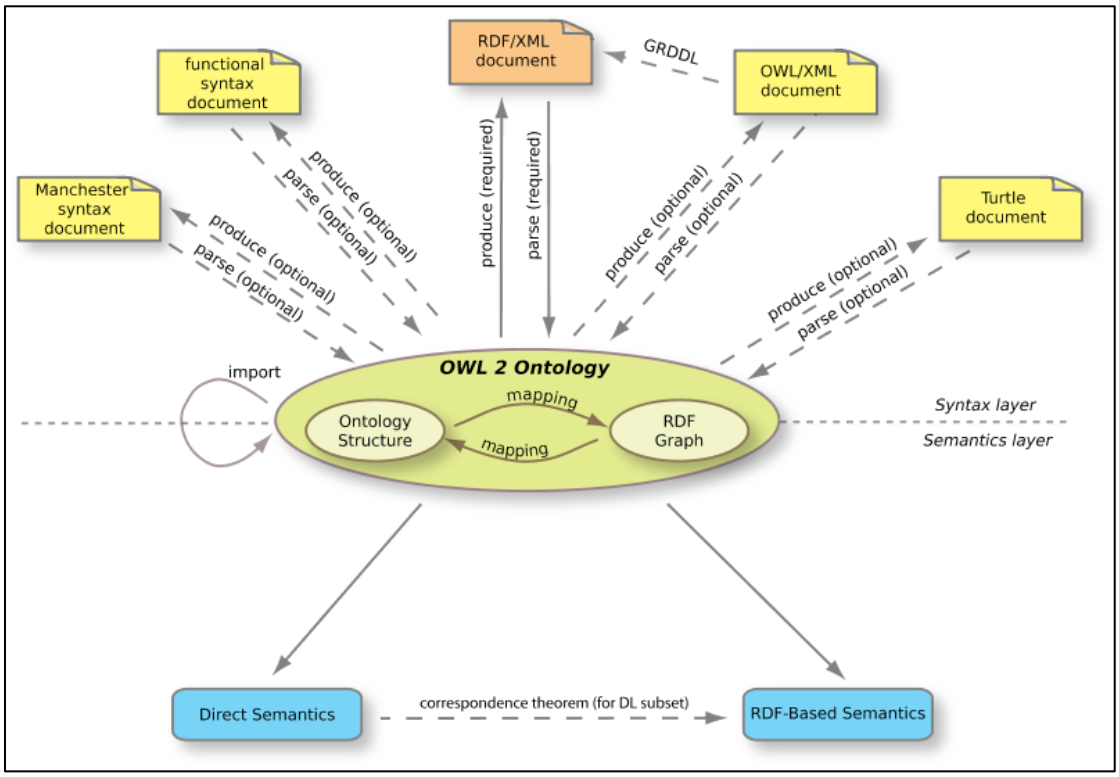

Figure 1: The structure of OWL 2

OWL files or documents are called ontologies [7]. The purpose of these ontologies is to make it easier of machine to access information in the web and preform reasoning on them. These ontologies can be put into the web or into a local computer depending on the need. One of the advantages of ontologies in the web is that they can be referenced from or reference to other ontologies [7]. Ontologies can be placed in a local computer to be used locally.

2.1.2 OWL API

OWL API is described in [10] as an Application Programming Interface for the purpose of specifying how to interact with OWL Ontologies. OWL ontologies can be created, manipulated, and reasoned over using OWL API. It has been available since almost the same time of OWL. OWL API went through several revisions following the development of OWL. OWL API has the ability to parse and serialize OWL ontologies to different syntaxes such as Functional Syntax, RDF/XML, OWL/XML and the Manchester OWL Syntax.

OWL API comes with free java implementation that takes out the burden of parsing and serializing OWL ontologies from the developers back [10]. OWL API comes also with loading and saving ontologies capabilities [10].

OWL ontologies being accessed using OWL API only through OntologyManager interface [10]. OntologyManager interface manage all changes in ontology as seen in Figure 1 below [10].

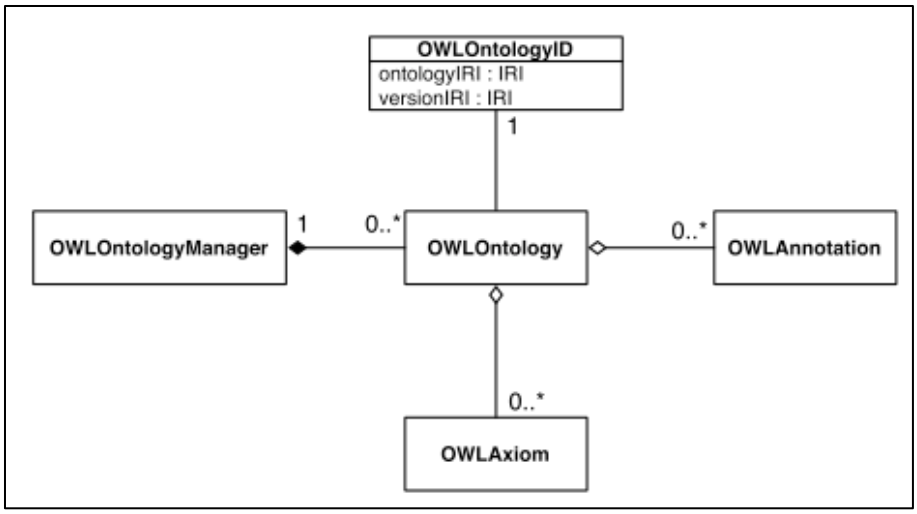

Figure 2: UML diagram showing the management of ontologies using OWL API

Inference is applied on the OWL ontologies using OWLReasoner interface [10]. This interface provides some useful check like consistency, checking computation of class and axiom entailments [10]. Since the reasoning functionality is separate, developers either can use the available or can provide their own implementation [10]. There are some exist implementations of reasoners such as FaCT++, HermiT, and Pellet [10].

As for query using OWL API, it does not offer much as a query mechanism [10]. Since, it provides some sort of basic querying which is based on entailment checking functionality [10].

2.2 Information Retrieval

Information retrieval is the method in which some information is retrieved from a source or multiple sources contains needed data. Every retrieval system needs some mechanisms to retrieval relevant needed information. Nowadays, there is more information probably resides on the cloud than it was in the recent years [12]. As the amount of data on the web grows dramatically, the need to find and retrieve relevant information becomes more important. Getting the wanted results is becoming more problematic because of the amount of the data on the web and the technique used. Usually, in a search engine the results range between relevant and irrelevant. It would be nice for a user to query for something and get the most relevant result that meets his/her needs.

There are different methods of retrieving data from the web. Keyword-based search method could be the oldest among the others.

2.2.1 Keyword-Based Information Retrieval

Keyword-based search method use specific words call "Keyword" that are linked with database records [13]. Keyword-based search method would be the default and the usual choice to use in search, since it has been used over long time. This method seems easy to use, as it is resemble natural language which could be understandable by humans but not by machines [13]. Because of the human factor that exists in writing the search query, things inevitably could go wrong. Simplicity and ease come with a cost, keyword-based search method suffers from some serious issues [13]. (1) One of these issues is the lack of accuracy and recall because of all of the synonyms and the homonyms which are based on memorizing terms rather than concepts [13]. (2) Another major issue is that using keyword-based search add more ambiguity, when the user want just to browse around to find out what is there or the user does not know the right term used in specific content [13].

According to [13], there are solutions for both issues. The lack of precision and recall can be treated by ontology-based information retrieval method. The growth in the ambiguity issue, would be solve be using multi-faceted search method which would guide the user during constructing the search query. So, the use of knowledge base and concept base would be more desirable than just providing arbitrary information. In addition, a sense of Artificial Intelligence is also felt since machines can make inferences based on some rules.

In this project, keyword based method won't be used since it has limitations like recall and ambiguity. As I am trying to overcome these limitations, ontology driven user interface as alternative method will be used.

2.3 Ontology Based User Interface

In general, developing user interfaces hindered by the knowledge of the user [14]. Letting the user known what can he ask for and constructing a meaningful search query using the user interface is the major issue [14]. To remedy this issue, solutions have been proposed. Some of those solutions would be making every option in the user interface available to reduce the recall issue. Another solution would be writing manuals to the user to follow. These solutions might be providing more complexity and other problems. The former solution could overwhelm the user with all of the options available whether needed options or not. The latter solution could increase the load on the user to study and spend time on something that needed to be recalled eventually. Ontology driven user interface would be the most suited solution, since ontologies are based on conceptual model rather than just terms [4, 14]. This conceptual model gives a map for the user to follow upon constructing queries [14].

Ontology based user interface is a user interface that allows the user to construct and manipulate queries based on some domain concept stored in ontology [4, 14]. The domain concept drives the user interface [14]. There is no need for manuals or shove all available options in the user interface, since the ontology based one which should act as a guide for the user [4, 14]. It depends on recognizing knowledge instead of memorizing keywords [14]. It allows the user to build a complex and meaningful queries and return the needed results [14]. In addition, it offers the user the option of

browsing around to find out what he/she needs [14]. The user does not have to any thing about the underlying conceptual knowledge [14]. TAMBS give the illusion of retrieving from single source while it read from multiple sources and convert selected options to appropriate query languages that match sources' [14].

The user interface offers choices and some scenarios for the user, so that the user would be guided toward constructing meaningful queries that return the intended results [4, 14]. Users would not face the no-result status after running queries. Query expressions are Description Logics (DLs) expressions and they are incremental and compositional [4, 14]. DL is a way for knowledge representation used by the conceptual model [15]. It provides hierarchal model based on conceptual model that represent classes of specific domain and the relationships between the instances of those classes [4, 14]. DL model is not easy because of the need knowledge about the DL syntax along with understanding it, so a friendly user interface need to build to separate the user from dealing with DL [14].

According to [14], there are two kind of concepts that the DL model support. The concepts definitions and the assertions made on the concepts definitions, like the subsumption relationship between two classes. In a way, assertions on the original concepts considered as defining new concepts definitions. Compositional concept can be formed using some services provided by DL. Reasoning about the concept definitions is done through the services provided by DL. These services are [14]:

- **Satisfiability**: make sure that the concepts are consistent.
- **Subsumption**: create composite concepts definition from assertion made on the original concepts definition.
- **Classification**: make new classification hierarchy based on the subsumption relationship.
- **Retrieval**: retrieve any individual that is part of concept definition whether it is original or generated from the subsumption relationships.

Ontologies support creating annotation properties within themselves and associate them with entities. Annotations properties consist of the name and the value. Annotations within ontologies play a major role in driving the user interface.

Annotations would form some set of rules for the user interface to follow and interact based on. A tool was developed based on some Animals ontology and annotations would be a good example of what could annotations do in term of user interface interactivity [16].

In this project, I am trying to develop an application to find out different information about sushi. The application is ontology based. Annotations play major role in driving the user interface to make more flexible. This application is based on The Manchester Pizza Finder application, which will be elaborated on a separate section, except this one would have some enhancements.

2.4 Faceted-Based Search

Ontology based user interface only provides the taxonomy and conceptual hierarchy and broad search capabilities. With the ontology conceptual hierarchy, user still can get broad search results. Transition from general to more specific results needs some kind of smart retrieval mechanism. Facet-based search along with ontology based user interface would guide the user toward constructing valid search queries and personalizing the search queries to suite the user needs. As using ontology in user interface development eliminates the recall element, using faceted-based search eliminates the ambiguity constructing the query and gets the intended results. So, ontology helps in returning relevant results. But faceted-based search assists in taking those relevant results and returning the most exact results. Ontogator is a system that combines the two methods: ontology driven interface and faceted based search [13]. The intent of Ontogator to search of particular image with specific annotations [13].

Ontology support faceted classification system as it provides a taxonomic order. Taxonomic order allows multiple way of viewing results rather that pre-determined one [13, 17]. Faceted search based on faceted classification system where information element are dimensions called facets [17]. Faceted search is an intelligent and efficient retrieval mechanism that allows the users to filter a collections of information based on some facets [17, 18]. Faceted search is known also as “faceted navigation” or “faceted navigation” [17]. Users can get more accurate and relevant results by applying filters (Facets) [17]. Facets here are derived from the ontology itself based on some annotations as metadata [13].

There are faceted browsing and faceted search. Faceted browsing is constructing search queries by selecting some provided filters (Facets) [13]. In faceted browsing, user is provided with choices to select from to form the valid search query [13]. The query language is hidden from the user, so the burden of knowing the syntax is lifted. On other hand, faceted search is more in personalizing the search result to suit what is needed. Faceted search is used heavily in e-commerce websites like Amazon[1] or eBay[2]. Figure 2 shows the use of faceted search in Amazon.

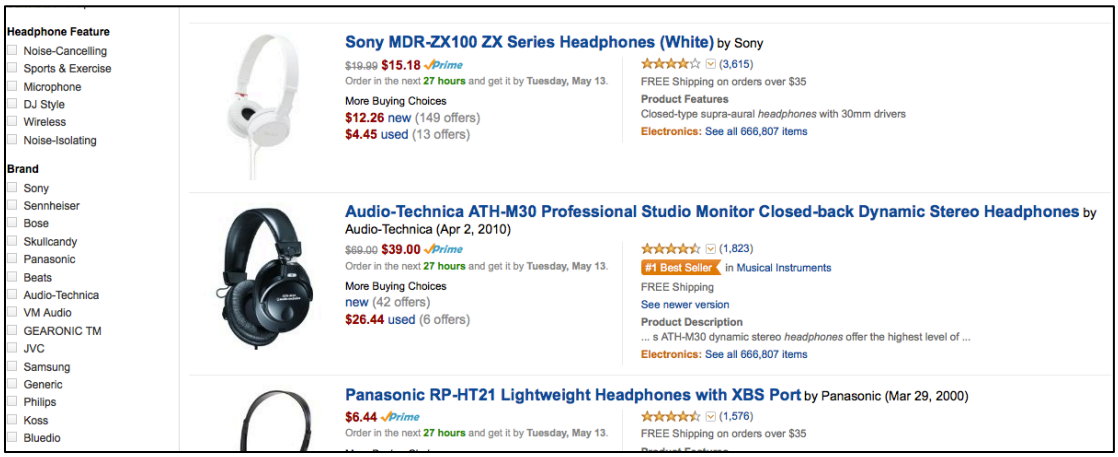

Figure 3: The use of faceted search in e-commerce website (Amazon)

Faceted search can be applied in two ways, either unidirectional or bidirectional. In unidirectional way, either applies it beforehand on a collection of selection that is browsed to construct the query or on the result of a query so it can be refined more. In the bidirectional way, it is to apply it both beforehand and afterward running the search query. Both serve the same purpose which is to personalized the search and make it easy to suit the user's needs.

Facets represent categories selected [13]. Each category consists of subcategories [13]. For example, in the pizza finder application user may choose Spicy Ingredient to get all pizza that are spicy but the category "Spicy Ingredient" could have subcategories like "Hot Pepper". Query in DL format would be [13]:

$$(S_{1,1} \vee \ldots \vee S_{1,k}) \wedge (S_{2,1} \vee \ldots \vee S_{2,k}) \wedge \ldots \wedge (S_{n,1} \vee \ldots \vee S_{n,k})$$

[1] http://www.amazon.com

[2] http://www.ebay.com

Where $(S_{1,1} \vee \ldots \vee S_{1,k})$ is the whole facet and S is the subcategory within that facet [13]. In other words, the S's are subcategories of "Spicy Ingredient" which they might include Hot Pepper as an S, and the disjunction of the S's represents the category "Spicy Ingredient".

There are some benefits of using facets [13]:

- **Guidance**: facets guide the user toward constructing valid search queries.
- **Transparency**: facets give the user idea of what is available and help in browsing the content.
- **Lucidity**: facets help in removing ambiguity caused by synonymous and homonymous query terms.
- **Relevance**: facets help with pre-compute partial results on selecting choices.

2.5 Ontology Visual Querying

The idea of visual querying is constructing a search query visually using drag and drop instead of the traditional way. Same idea can be applied on ontology-based interfaces that would be more powerful because of the benefits of the ontology-based applications. The interface would guide the user to build interactive meaningful queries by using ontologies [2, 4]. In addition, another advantage derived from the benefits of ontology-based applications is constructing only exact queries [2, 4].

According to [2], visual querying is not new. It has been there since almost the beginning of textual query languages. Almost all visual querying languages have two features in common. The two features are: (1) a model to represent the stated query and (2) a way to of constructing the query. Since visual querying languages invented to query from a data structure, it is only natural for its evolution to follow the development of data structure [2, 19]. A simple example of visual querying would in Microsoft Access.

A major benefit from ontology visual querying is the ease of querying, since user only drag and drop what needed to be queried. User does not have to remember or know the vocabulary, since user can survey the domain [2, 4]. As a result, forming queries for naïve users becomes easier [2]. In ontologies, new concept can be defined either

directly like defining class or indirectly like making inference of something. Therefore, creating query is the same as creating new concept such as the TAMBIS system and SEWASIE system [2-4]. Another advantage would be helping users, who not experienced with the system, to create satisfiable queries according to the constraints [2].

2.6 The Manchester Pizza Finder

The Manchester Pizza Finder is an application that finds specific pizza based on some topping choices. User can include and exclude any toppings, and based that the result would satisfy the query. The use of DL reasoner is present in this application, since it generated the filtering criteria (pizza topping) and their categories in the runtime. It is also make sure that the constructed queries and results are consistent. Based on the choices made the DL reasoner retrieve result that fulfill the input query. This application shows the use of ontologies, OWL API, and the power of building ontology-based interface, and faceted browsing.

The Manchester Pizza Finder is a user interface application that makes use of OWL ontology. It uses pre-defined pizza ontology that represents a domain concept of a pizza restaurant menu. For the application to be able to communicate with the pizza ontology, an API needs to be used. OWL API is an important component as any part of the application if not more important. OWL API manages all the communications between the application and the pizza ontology. OWL API is implemented using Java Pizza finder is developed using Java, since OWL API is a java API. This makes the communication between the application and the ontology easier. OWL API have full access to the pizza ontology, it can preform operations on the ontology like make sure it consistent.

Pizza finder is considered ontology-based application, since it is the ontology that derives the interface and provides a conceptual hierarchy of a pizza domain. In ontology-based application, user does not need to recall keyword or know query language on querying for specific pizza. The application itself guides the user toward building only valid queries with the ability of making complex meaningful ones. It has the ability to incrementally compose queries. User can browse around to figure

out what specific toppings are needed. It based on the knowledge of pizza domain, not on keywords.

Pizza finder personalizes the query construction process by providing some filters (Facets). As a result, the results would suite the user needs. Pizza finder uses faceted-base in query building, so the user will be guided to construct only valid queries. User does not have to recall what keyword to search for something in the domain. User has the option in querying for broad or specific pizza in the domain based on chosen facet. Figure 3 shows the use of facets in pizza finder, use can query for example for vegetable topping pizzas or can query for more specific thing in vegetable topping category such as Tomato topping.

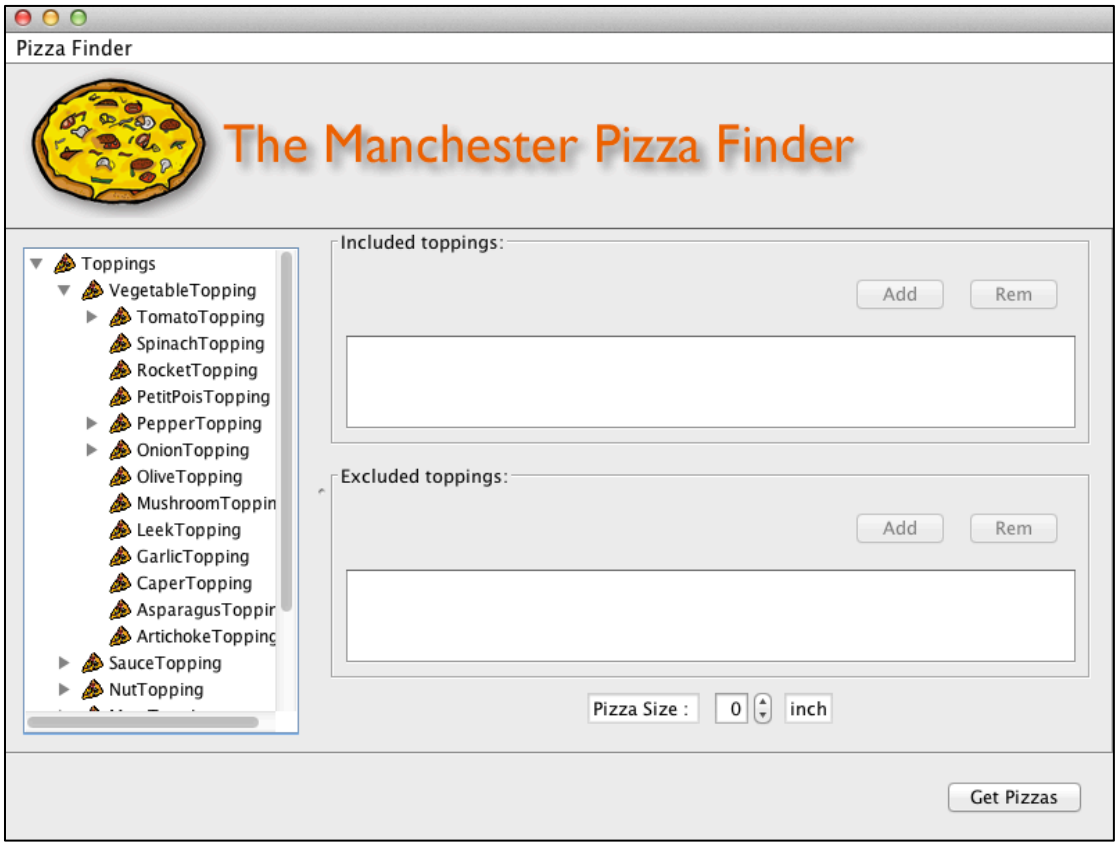

Figure 4: Facets are in the left hand side used to specify what is needed exactly.

Sushi finder is considered an extension for pizza finder. Some enhancements of pizza finder are to be introduced. In Sushi finder, the application is more flexible than pizza one. The application should be able to work for any given ontology regardless of domain, but should follow some standard annotations. Another enrichment would be

the use of annotations to drive the user interface sort of dynamically. Keeping the configurations in the ontology itself make it easier to for the user interface to be flexible. The whole application would be configurable in term of labels and languages being used. It configurable within itself, no need for changing configuration files. Facet search is introduced, user can filter the choosing options based on some configuration done in the ontology as annotations. In addition, user can apply filters on the query results.

3 RESEARCH METHODOLOGY AND PROJECT PLAN:

3.1 Research Methodology:

As mentioned in the objectives section, is to develop a system that will:

1. Find specific sushi platter based on some ingredients choices. Include and exclude criteria for the ingredients are being used.
2. Students can run the system using an ontology they have developed for their coursework, in condition that they annotate their ontology in some way. The user interface is flexible since all labels can be configured.
3. Users can view the hierarchy of the ingredients in different views like tree and lists.
4. User can filter the hierarchy of the ingredients based on some facets, as well as, filtering the result of the search query. Due to the faceted search, search will be more personalized instead of broad and general one.

To achieve the objectives of this project, it has been divided into five phases.

3.1.1 Phase 1-Requirements Gathering:

Since the project is to develop a system that behaves in a certain way, I have started with the first and important role in software development process which is requirements gathering. The stakeholders of the project are three; the end user who will use the system, the system provider who will provide the tool to the end user and probably does some configurations to the tool, and ontology engineer. Since meeting all of these stakeholders hard, I had to put myself in their shoes. Some of those stakeholders were involved in the requirements gathering process. The main

stakeholder was my supervisor as he requested for this system to be develop. Meetings have been setup to discuss the requirements (what exactly should be done?). Some conversations were held with my fallow students, who attended with me the ontology engineering for semantic web course, regarding if they had this system before how it would help them and what functionalities would be needed. All of those meetings and discussions provided more details and helped in understanding some of the requirements. Some were understood later on.

3.1.2 Phase 2-Background Study:

In the second phase, background research and survey of relevant literature was carried out along with exploring techniques to be used in the project. This has been done using journals, articles, publications and existing systems with similar functionalities. A fair amount of time spent reviewing different literatures trying to understand different aspect of the project's requirements. Reviewing relevant literature really helped in not only having a wider knowledge of the problem domain but it helped also in understanding the requirements of the project. As a result, I have good understanding of the project and approaches taken to handle such systems and it helped me in splitting the project into small tasks.

In this phase, background study is done on OWL ontology, OWL API, ontology-based systems, faceted-based search system, ontology visual querying, and finally study the Manchester Pizza finder a system that I will build my project on.

3.1.3 Phase 3-Development Phase:

In the development phase, first decide in the development tool that is java. Then, refresh myself with java specially swing components, and OWL API. Implementation started in early stage. As the strategy is to divide the development of the system into developing the main functionalities separately then combine them. Although that some functionalities were developed separately from the application itself, the official start of this phase will be after the second semester's exams. After exams, checking that the functionalities are working probably will be done and combine them.

The development is considered as enhancements of the Manchester Pizza Finder. They involve reading the configurations from the ontology file and act accordingly for more flexibility. Also, different view of the content of the conceptual model and the

result of the search query will be considered. As well as, adding a functionality to filter the content of the model based on some criteria saved in the ontology to personalize the search even further to the user.

3.1.4 Phase 4-Testing Phase:

In this phase, testing will be conducted on the application according to some scenarios that are predefined. These scenarios are called users stories which will be elaborated on later on in the report. Since the strategy of doing the project is to develop functionalities alone then combine them, testing is carried out during the development phase on functionalities separately and on the final product after combining them.

3.1.5 Phase 5-Review and Submission Phase:

After success in evaluating the product, the review and submission phase will start. An instruction file will be provided to guide the ontology developer in how to make his ontology to work with the application. The application would act as manual, so there will be no need for a guide for the system. The next step will be finishing the application and finalized the dissertation and then submitting them.

There are five milestones within this project that will guide me through the progress of the project. The milestones are:

1. Initial report.
2. Progress report.
3. Application prototype.
4. Application final product.
5. Dissertation submission.

First milestone was already completed, since initial report was submitted successfully in March. However, submission of second milestone in time was not so successful. So the original plan was altered. The new deadline will be in June 6. The last three milestones would be worked on in parallel due to time restriction. The final product is expected to finish beginning of August. Finally, the submission of the dissertation will be in the first week of September. Gantt chart is included in the next section.

3.2 Project Plan:

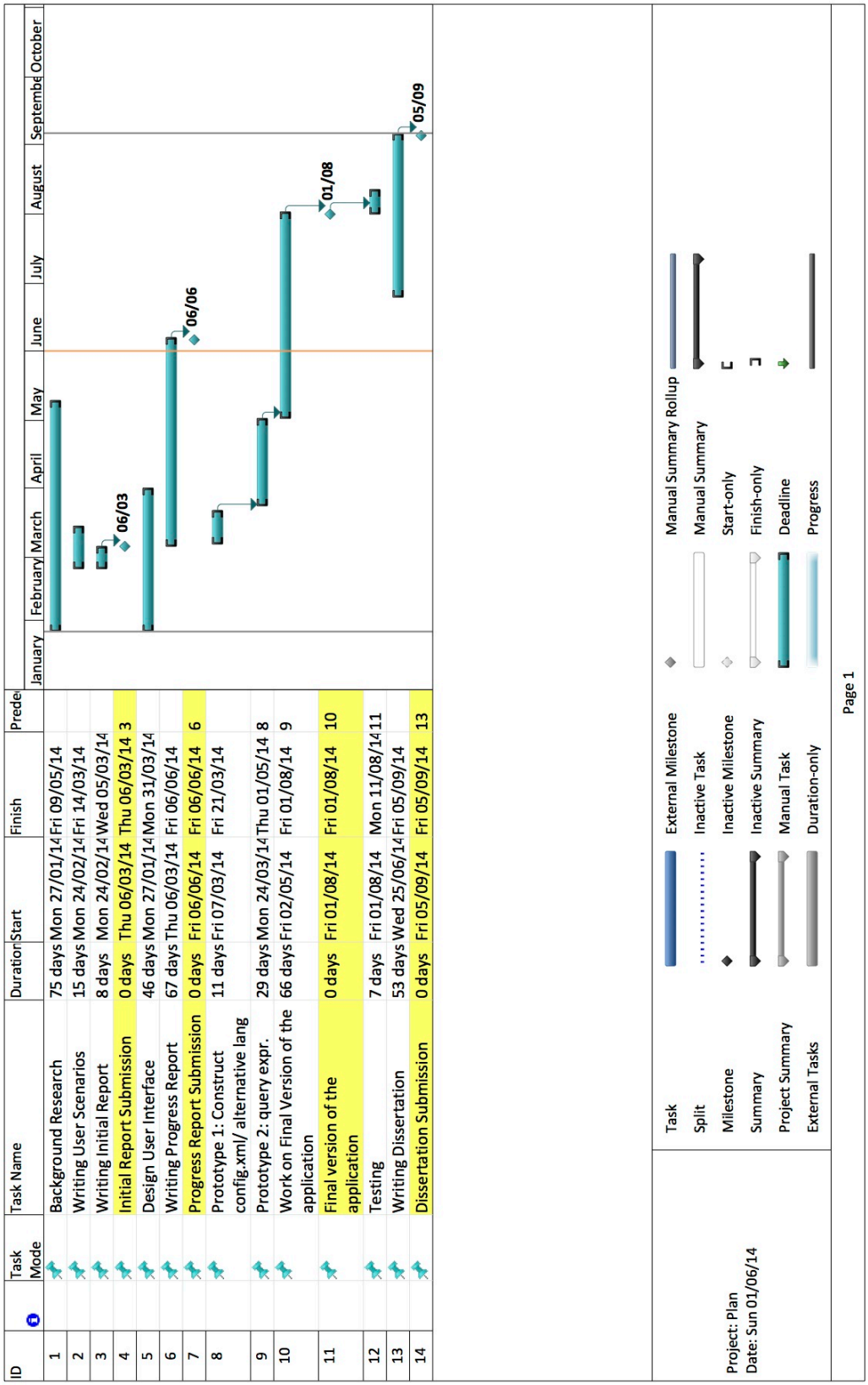

Figure 5: Project Plan

3.3 Project Deliverables:

At the end of this project the benefits of using sematic web within applications will be shown. The application will be flexible, as it will run ontology with specific annotations. In addition, it will be fully configurable. It will allow users to query for specific sushi based on some ingredients. There are three main deliverables of this project:

- User stories (scenarios) & acceptance tests.
- Final version of the application.
- Evaluation of the project.

3.4 Project Evaluation Plan:

In order to be able to evaluate the whole project, I need some measures to evaluate the project against. Some measures have been recognized to evaluate how good the system is and more importantly measure if the project is consider a success or not. In nutshell measuring the success of the objectives. These measures are:

- **Where the deliverables met?**

  A most likely way to measure if the deliverables were met or not according to a timeframe is to check if a deliverable was done within its allocated time or not.

- **Is the system accessible?**

  We can think of the system in two ways: one as a project for MSc program that should be accessible for students and lecturers who teach ontology engineering for semantic web, and as real-world application (restaurant menu). Based on that users who should access the system differ. As for the first case, since the application is desktop application and implemented on java this will assure the students and lecturers can access the system easily. In the second case end users can access the system but a web application would be more reasonable. For now it is only a desktop application.

- **Is the system (re)usable?**

  As the system designed for end users regardless of what kind of end user students or real-world users, they can determine the usability of the system. There are two concepts here determining the usability and reusability of the system. Therefore, both of them need to be checked as part of project evaluation.

- **Usability**: as an end user, one can ask several questions that will assure the system is usable. Some of these questions are:

1. How easy to use the system?

   The idea behind using ontology-based user interface is to guide the user in how to use it and ease that process. Therefore, the system should be easy to use.

2. How much time spent to figure out the system?

   This question can be answered after submitting the project.

3. Does it require experts to use the system?

   The system is designed for students who take ontology engineering for sematic web. So, some degree of expertise is needed to configure the ontology to work with the application.

4. How easy to administer the system?

   Since most of the configurations are saved within the ontology, administering the system would be a trivial task.

- **Reusability**: the system is reusable in sense of running different ontologies. This is another goal of the project, making it more flexible. The important question here is how easy it is reuse the system (running different ontologies).

- **Is the system easily configurable?**

  As most of the configurations saved in the ontology file, it would be easy to configure the application interface. Ontology developer is the only stakeholder who has to deal with the configurations which are annotations in the ontology file. They are easy to write, as the ontology developer needs to follow some instructions provided with the application. The user interface should be configured automatically using annotations in the owl file.

- **Is the result of the search query narrowed down?**

  Stakeholders such as students or real world users, like restaurant customers, will have direct contact with feature. This can be answered after submitting and using the application. Nevertheless, applying a set of filters on the search result will narrow the search for the user, since the system used faceted search method.

These are some questions that can help assessing the project. Most of them can be answered only after using the application by stakeholders. So, evaluation will help in a second version or the final product if it a prototype.

3.5 Project Tools:

Since the main part of the project is development, a programming language needs to be chosen. OWL API, which was implemented in java, is used in the project to manage the interactivity between OWL ontology and application. So, choosing java as programming language makes sense. Project is developed using NetBeans 7.4.

4 PROGRESS:

In this section, the progress of the project is summarized. It provides a brief summary of phases that were want through as well as progress on deliverables.

4.1 Project Phases:

- Gathering the requirements: the initial requirements have been gathered. So, this phase in the scope of this project has been completed. However, some new requirements could emerge during this project or even in future versions. What have been gathered should be sufficient to start the process of implementations the main functionalities.
- Background study: background research and survey of relevant literature on ontology driven interfaces has been accomplished. The review includes four pillars of the project: background research on information retrieval mechanism, ontology-based interface, faceted-based interface, and ontology visual querying. It includes as well some background information about OWL ontologies and OWL API.
- Development: an initial start on the development phase has been made. Some functionalities have been developed separately from each other. Complete all of the rest and testing them, then combine them is still to be done.
- Testing: unit testing and final test of the product is still to be conducted in different stages. As soon component (functionality) is done, it needs to be tested according to some user stories. At the end full final test on the whole application will be preformed.

4.2 Deliverable:

- Users Stories & Acceptance Tests: users stories have been developed, yet they need to be polished. These users stories based on stakeholders' scenarios. Some has been used in unit test to make sure they meet the gathered requirements, where some parts of the application are done. Unit tests at this stage act as acceptance tests of individual units. After assembling the application a full acceptance test where to be preformed.
- Prototype:

5 CONCLUSION

6 REFERENCES:

1. Ding, L., et al., *Using Ontologies in the Semantic Web: A Survey*, in *Ontologies*. 2007, Springer US. p. 79-113.
2. Bechhofer, S. and N.W. Paton, *Ontology Visual Querying*, in *Encyclopedia of Database Systems*. 2009, Springer.
3. Stevens, R., et al., *TAMBIS: Transparent Access to Multiple Bioinformatics Information Sources.* IBM System Journal. **40**(2): p. 532-551.
4. Catarci, T., et al., *An Ontology Based Visual Tool for Query Formulation Support.* ECAI, 2004: p. 308-312.
5. Bechhofer, S. *The Manchester Sushi Finder - Project Page*. 2014 [cited 2014 March 5, 2014]; Available from: http://studentnet.cs.manchester.ac.uk/pgt/2013/COMP60990/project/projectbookdetails.php?projectid=20889.
6. Wikipedia. *Application programming interface*. [cited 2014 April 21, 2014]; Available from: http://en.wikipedia.org/wiki/Application_programming_interface.
7. Group, W.C.O.W. *Web Ontology Language (OWL)*. 2012 [cited 2014 April 19, 2014]; Available from: http://www.w3.org/2001/sw/wiki/OWL.
8. Davis, R., H. Shrobe, and P. Szolovits, *What is a Knowledge Representation?*, in *AI Magazine*. 1993. p. 17-33.
9. Group, O.W. *OWL Web Ontology Language Overview*. 2004 [cited 2014 May 13, 2014]; Available from: http://www.w3.org/TR/owl-features.
10. M, H. and B. S, *The OWL API: A Java API for OWL ontologies.* Semantic Web, 2011. **2**(Number 1 / 2011): p. 11-21.
11. Group, O.W. *OWL 2 Web Ontology Language Document Overview*. 2012 [cited 2014 May 13, 2014]; Available from: http://www.w3.org/TR/owl2-overview/.
12. *The size of the World Wide Web (The Internet)*. 2014 [cited 2014 Jun 2, 2014]; Available from: http://www.worldwidewebsize.com/.
13. Hyvönen, E., S. Saarela, and K. Viljanen, *Application of ontology techniques to view-based semantic search and browsing*, in *The Semantic Web: Research and Applications*. 2004, Springer. p. 92-106.
14. Bechhofer, S., et al., *Guiding the User: An Ontology Driven Interface.*
15. Bechhofer, S. and C. Goble, *Classification Based Navigation and Retrieval for Picture Archives*, in *Database Semantics*. 1999, Springer. p. 291-310.
16. Schreiber, A.T.G., et al., *Ontology-based photo annotation.* IEEE Intelligent Systems, 2001. **16**(3): p. 66-74.
17. Wikipedia. *Faceted search*. [cited 2014 April 23, 2014]; Available from: http://en.wikipedia.org/wiki/Faceted_search.
18. Smith, D.A. and N.R. Shadbolt, *FacetOntology: Expressive Descriptions of Facets in the Semantic Web*, in *Semantic Technology*. 2013, Springer. p. 223-238.
19. Catarci, T., et al., *Visual query systems for databases: A survey.* Journal of Visual Languages and Computing, 1997. **8**: p. 215-260.